# RTI Connext Python API

**Version 7.7.0**



# Contents


**1 Getting Started** **2**

**2 API Overview** **3**

2.1 Hello World . . . 3
2.2 DomainParticipant . . . 4
2.3 Topics . . . 5
2.3.1 Special Topics . . . 5
2.4 Publications . . . 6
2.5 Subscriptions . . . 6
2.5.1 Creating a DataReader . . . 6
2.5.2 Reading data . . . 7
2.5.3 Being notified when data is available . . . 7
2.5.4 Special DataReaders . . . 8
2.6 Data Types . . . 9
2.6.1 IDL to Python mapping . . . 9
Primitive types . . . 10
Strings . . . 10
Sequences . . . 11
Arrays . . . 12
Nested collections . . . 12
Optional members . . . 13
Enumerations . . . 13
Unions . . . 14
Modules . . . 15
IDL annotations . . . 16
2.6.2 Built-in types . . . 16
2.6.3 Type support . . . 17
2.6.4 DynamicType and DynamicData . . . 18
Accessing Nested Members . . . 19
Accessing Sequences and Arrays . . . 19
2.7 XML Application Creation . . . 20
2.7.1 Defining your system in a configuration file . . . 20
2.7.2 Creating the Entities from the configuration . . . 21
2.7.3 Using IDL and Python data types . . . 21
2.8 Request-Reply and Remote Procedure Calls . . . 22
2.8.1 Requesters and Repliers . . . 22
Requester: sending requests and receiving replies . . . 23

Replier: receiving requests and sending replies . . . 24
SimpleReplier . . . 24
Waiting for a reply to a specific request . . . 25
Accessing the underlying readers and writers . . . 25
2.8.2 Remote Procedure Calls . . . 25
Concurrency model . . . 28
Exceptions . . . 29
Parameters . . . 30

**3 Quick Reference** **32**

**4 rti package** **34**
4.1 rti.connextdds . . . 34
4.1.1 rti.connextdds.compliance . . . 791
4.1.2 rti.connextdds.heap_monitoring . . . 798
4.2 rti.types . . . 803
4.2.1 rti.types.builtin . . . 804
4.3 rti.rpc . . . 805
4.4 rti.asyncio . . . 810
4.5 rti.logging . . . 812
4.5.1 Submodules . . . 812
rti.logging.distlog module . . . 812
rti.logging.handler module . . . 815
4.5.2 Module contents . . . 816

**5 Examples** **817**

**6 Copyrights and Notices** **818**

**Python Module Index** **820**

**Index** **821**

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

The *RTI Connext Python API* provides access to most *Connext* features from Python.

# Chapter 1

# Getting Started

The RTI Connext Getting Started Guide helps you install the software and run your first *Connext* Python application while learning general concepts of *Connext*.

See the Core Libraries Platform Notes for information on which *Connext* platforms and Python versions the Python API is supported on. (For example, see "Support for the Python API" in the Linux Platforms chapter.)

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 Connext Migration Guide on RTI Community provides guidance.

For more documentation, including the Connext User's Manual and the reference for the C, C++, Java and C# APIs, see RTI Community.

# Chapter 2

# API Overview

## 2.1 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*.

## 2.2 DomainParticipant

*DomainParticipants* are the focal point for creating, destroying, and managing other *Connext* objects. A DDS domain is a logical network of applications: only applications that belong to the same DDS domain may communicate using *Connext*. 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)
    # ...
```

## 2.3 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.

### 2.3.1 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*

## 2.4 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*.

## 2.5 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*.

### 2.5.1 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.

### 2.5.2 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 Connext language bindings, which return temporary loaned objects.

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

### 2.5.3 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.enabled_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`.

```
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 Connext thread, and should not block or perform CPU-heavy operations.

### 2.5.4 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`*

## 2.6 Data Types

### 2.6.1 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 Connext 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.

### 2.6.2 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))


@idl.struct
class PingType:
    number: idl.int32 = 0
```

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!"))
```

### 2.6.3 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).

### 2.6.4 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
```

## 2.7 XML Application Creation

RTI Connext 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.

### 2.7.1 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.

### 2.7.2 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()*.

### 2.7.3 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))
```

## 2.8 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.

For more information, see Request-Reply Exchanges and Remote Procedure Calls in the *RTI Connext Core Libraries User's Manual*.

A full request-reply example application is available in the Connext Examples repository.

### 2.8.1 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

First wait for the requester to discover a replier using *rti.rpc.Requester.wait_for_service_async()*:

```
await requester.wait_for_service_async(dds.Duration(10))
```

Once a replier is discovered, 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 Connext 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 Connext Core Libraries User's Manual.

### 2.8.2 Remote Procedure Calls

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();
};
```

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")
```

First, wait to discover a service using *`rti.rpc.ClientBase.wait_for_service_async()`*:

```
await client.wait_for_service_async(dds.Duration(10))
```

Once the service is discovered, the client can 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}")
```

A service may also raise any of the following built-in exceptions, which will be propagated and raised by the client:

- *rti.rpc.RemoteUnsupportedOperationError*
- *rti.rpc.RemoteInvalidArgumentError*
- *rti.rpc.RemoteOutOfResourcesError*

When an operation doesn't exist in the service, the client will raise *rti.rpc.RemoteUnknownOperationError*.

Any other exception in the service will be propagated and raised in the client as *rti.rpc.RemoteUnknownExceptionError*.

### Parameters

Operations in a service may have `in`, `out`, and `inout` parameters. The `out` and `inout` parameters can be modified by the service implementation. The modified values are sent back to the client in the response to the remote procedure call.

To specify the kind of parameter, use `rpc.in_param`, `rpc.out_param`, or `rpc.inout_param`. If not specified, the default, `rpc.in_param`, is used.

Example operation using each type of parameter:

```
@rpc.service
class RobotControl:
    @rpc.operation(
        parameter_annotations={
            "out_": [rpc.out_param],
            "inout_": [rpc.inout_param],
        },
    )
    async def operation(self, in_: Coordinates, out_: Coordinates, inout_:␣
↪Coordinates):
    ...
```

To implement the above operation and modify the `out` and `inout` parameters:

```
class RobotControlExample(RobotControl):
    async def operation(self, in_: Coordinates, out_: Coordinates, inout_:␣
↪Coordinates):
        # The in parameter flows from the client to the service.
        # The received value is the one the client used as a parameter.
        # Any modification won't affect the client parameter.
        print(in_)

        # The out parameter only flows from the service to the client.
        # The received value is the not the one the client used as a
        # parameter, it is default value.
        # Any modification will affect the client parameter.
        out_.x = 10

        # The inout parameter flows in both directions.
        # The received value is the one the client used as a parameter.
```

```
        # Any modification will affect the client parameter.
        inout_.x += 1
```

The client now can execute the operation, and the parameters will be modified.

```
# The value we assign here doesn't matter because the service will work
# with the default value of Coordinates.
out_ = Coordinates(2,2)

# The value we assign here does matter, because the service will receive
# this value.
inout_ = Coordinates(0,2)

client.operation(Coordinates(0,0), out_, inout_)
assert out_ == Coordinates(10,0)
assert inout_ == Coordinates(1,2)
```

Complex types can be modifed in the operations. However, primitive types (like `int`, `float`, and `str`) cannot be changed within the method in Python. *Connext* provides `rpc.ByRef` to encapsulate primitives values and allow modification of the values in the methods, implementing the functionality for `out` and `inout` parameters.

For example, in this operation the `out` and `inout` parameters are strings:

```
@rpc.service
class RobotControl:
    @rpc.operation(
        parameter_annotations={
            "out_": [rpc.out_param],
            "inout_": [rpc.inout_param],
        },
    )
    async def operation(self, out_: rpc.ByRef[str], inout_: rpc.ByRef[str]):
        ...
```

In the implementation, you can modify `out_.value` and `inout_.value`, and the changes will be reflected in the client parameters:

```
class RobotControlExample(RobotControl):
    async def operation(self, out_: rpc.ByRef[str], inout_: rpc.ByRef[str]):
        out_.value = "Hello World!"
        inout_.value += " World!"
```

The client now can execute the operation, and the parameters will be modified:

```
out_ = rpc.ByRef[str]("")
inout_ = rpc.ByRef[str]("Hello")

client.operation(out_, inout_)
assert out_.value == "Hello World!"
assert inout_.value == "Hello World!"
```

# Chapter 3

# Quick Reference

This section includes links to common `rti.connextdds` 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.connextdds*

# Chapter 4

# rti package

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

## 4.1 rti.connextdds

`rti.connextdds` (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*.

See the genindex for a list of all types and functions.

---

**Note:** The Python API reference includes brief descriptions for most types and functions. More detailed descriptions can be found in the Modern C++ API Reference, the C API Reference, and the Core Libraries User's Manual.

---

**class** `rti.connextdds.`**ACTEnumMember**

Bases: *DynamicType*

**INVALID_INDEX = 4294967295**

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

Test for equality.

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

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

Overloaded function.

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

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

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

Gets the maximum serialized size of samples of this type.

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

Gets the minimum serialized size of samples of this type.

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

Gets the extensibility kind.

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

Obtains the member index from its name.

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

Overloaded function.

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

Gets a member by its index.

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

Gets a member by its name.

**property member_count**

Gets the number of members.

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

Gets a copy of all the members

**class** rti.connextdds.**ACTMember**

Bases: *DynamicType*

**INVALID_INDEX = 4294967295**

**__annotations__ = {}**

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

Test for equality.

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

Overloaded function.

1. __getitem__(self: rti.connextdds.ACTMember, arg0: str) -> rti.connextdds.Member
2. __getitem__(self: rti.connextdds.ACTMember, arg0: int) -> rti.connextdds.Member

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**cdr_serialized_sample_max_size**(*self:* rti.connextdds.ACTMember, *arg0: int*) → int

Gets the maximum serialized size of samples of this type.

**cdr_serialized_sample_min_size**(*self:* rti.connextdds.ACTMember, *arg0: int*) → int

Gets the minimum serialized size of samples of this type.

**extensibility_kind**(*self:* rti.connextdds.ACTMember) → *rti.connextdds.ExtensibilityKind*

Gets the extensibility kind.

**find_member_by_name**(*self:* rti.connextdds.ACTMember, *arg0: str*) → int

Obtains the member index from its name.

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

Overloaded function.

1. member(self: rti.connextdds.ACTMember, arg0: int) -> rti.connextdds.Member

Gets a member by its index.

2. member(self: rti.connextdds.ACTMember, arg0: str) -> rti.connextdds.Member

Gets a member by its name.

**property member_count**

Gets the number of members.

**members**(*self:* rti.connextdds.ACTMember) → *rti.connextdds.MemberSeq*

Gets a copy of all the members

**class** rti.connextdds.**ACTUnionMember**

Bases: *DynamicType*

**INVALID_INDEX = 4294967295**

**__annotations__ = {}**

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

Test for equality.

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

Overloaded function.

1. __getitem__(self: rti.connextdds.ACTUnionMember, arg0: str) -> rti.connextdds.UnionMember

2. __getitem__(self: rti.connextdds.ACTUnionMember, arg0: int) -> rti.connextdds.UnionMember

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

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

Gets the maximum serialized size of samples of this type.

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

Gets the minimum serialized size of samples of this type.

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

Gets the extensibility kind.

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

Obtains the member index from its name.

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

Overloaded function.

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

Gets a member by its index.

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

Gets a member by its name.

**property member_count**

Gets the number of members.

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

Gets a copy of all the members

**class** rti.connextdds.**AckResponseData**

Bases: pybind11_object

**__annotations__ = {}**

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

Overloaded function.

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

Create an empty ack response.

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

Create an instance with a byte sequence.

**__module__ = 'rti.connextdds'**

**property value**

Gets the subscription handle of the acknowledging DataReader

**class** rti.connextdds.**AcknowledgmentInfo**

Bases: pybind11_object

**__annotations__ = {}**

**__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>**

**__annotations__ = {}**

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

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.AcknowledgmentKind.AcknowledgmentKind) → int

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

**__int__**(*self:* rti.connextdds.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.connextdds'**

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

**__repr__**(*self: object*) → str

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

**__str__**(*self: object*) → str

**property name**

**property value**

**PROTOCOL = <AcknowledgmentKind.PROTOCOL: 0>**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.AcknowledgmentKind, *arg0:* rti.connextdds.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 = 00000000000000000000000000111111**

**DEFAULT = 00000000000000000000000000111111**

**DOMAIN_ID = 00000000000000000000000000010000**

**ENTITY_KIND = 00000000000000000000000000001000**

**ENTITY_NAME = 00000000000000000000000000100000**

**GUID_PREFIX = 00000000000000000000000000000001**

**NONE = 00000000000000000000000000000000**

**TOPIC = 00000000000000000000000000000010**

**TYPE = 00000000000000000000000000000100**

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

Bitwise logical AND of masks.

**__annotations__ = {}**

**__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.

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

Create an ActivityContextMask with no bits set.

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

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.ActivityContextMask) → int

Convert mask to int.

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

Set mask to logical OR with another mask.

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

Right shift bits in mask.

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

Set mask to logical XOR with another mask.

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

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

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

Compare masks for inequality.

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

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.ActivityContextMask) → str

Convert mask to string.

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

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.ActivityContextMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.ActivityContextMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.ActivityContextMask, *arg0:* rti.connextdds.ActivityContextMask) → *rti.connextdds.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.connextdds.ActivityContextMask) -> rti.connextdds.ActivityContextMask

Flip all bits in the mask.

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

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

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

Clear all bits in the mask.

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

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

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

Set all bits in the mask.

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

Test whether the mask bit at position "pos" is set.

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

Test if all bits are set.

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

Test if any bits are set.

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

Test if none of the bits are set.

**class** rti.connextdds.**AliasType**

Bases: *DynamicType*

Represents a typedef.

**__annotations__ = {}**

**__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.

**property unit**

Retrieves the unit annotation of this alias type.

**class** rti.connextdds.**AllocationSettings**

Bases: pybind11_object

**AUTO_COUNT = -2**

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.AnyDataReaderSeq) -> None

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.AnyDataReaderSeq) → Iterator[*rti.connextdds.AnyDataReader*]

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

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.AnyDataReaderSeq, arg0: int, arg1: rti.connextdds.AnyDataReader) -> None

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

Assign list elements using a slice object

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

Add an item to the end of the list

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

Clear the contents

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

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.AnyDataReaderSeq, *i: int*, *x:* rti.connextdds.AnyDataReader) → None

Insert an item at a given position.

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

Overloaded function.

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

Remove and return the last item

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

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.AnyDataReaderSeq, *x:* rti.connextdds.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.connextdds.**AnyDataWriter**

Bases: *IAnyDataWriter*

**__annotations__ = {}**

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

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

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**AnyDataWriterListener**

Bases: pybind11_object

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

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

On application acknowledgment callback

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

On instance replaced callback.

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

Liveliness lost callback.

**on_offered_deadline_missed**(*self:* rti.connextdds.AnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.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*

**__annotations__ = {}**

**__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.connextdds.AnyDataWriterSeq, *arg0:* rti.connextdds.AnyDataWriterSeq) → bool

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

Overloaded function.

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

Retrieve list elements using a slice object

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

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__**(*self:* rti.connextdds.AnyDataWriterSeq) → Iterator[*rti.connextdds.AnyDataWriter*]

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

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.AnyDataWriterSeq, arg0: int, arg1: rti.connextdds.AnyDataWriter) -> None

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

Assign list elements using a slice object

**append**(*self:* rti.connextdds.AnyDataWriterSeq, *x:* rti.connextdds.AnyDataWriter) → None

Add an item to the end of the list

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

Clear the contents

**count**(*self:* rti.connextdds.AnyDataWriterSeq, *x:* rti.connextdds.AnyDataWriter) → int

Return the number of times x appears in the list

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

Overloaded function.

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

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

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

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

**insert**(*self:* rti.connextdds.AnyDataWriterSeq, *i: int*, *x:* rti.connextdds.AnyDataWriter) → None

Insert an item at a given position.

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

Overloaded function.

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

Remove and return the last item

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

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.AnyDataWriterSeq, *x:* rti.connextdds.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.connextdds.**AnyTopic**

Bases: *IAnyTopic*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.AnyTopic, *topic:* rti.connextdds.IAnyTopic) → None

Create an AnyTopic instance from a object that conforms to the IAnyTopic interface.

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**AnyTopicListener**

Bases: pybind11_object

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

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

Inconsistent topic callback.

**class** rti.connextdds.**AnyTopicSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

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

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

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

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

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

**__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.connextdds.AnyTopicSeq, *x:* rti.connextdds.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.connextdds.**ArrayType**

Bases: *CollectionType*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Creates a unidimensional array.

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

Create a multidimensional array.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

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

Returns the size of the ith dimension

**property dimension_count**

Number of dimensions.

**property total_element_count**

Total element count across all dimensions.

**class** rti.connextdds.**AsynchronousPublisher**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Create a default AsynchronousPublisher QoS policy.

**__module__ = 'rti.connextdds'**

**__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

**__annotations__ = {}**

**__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_endpount_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

**__annotations__ = {}**

**__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 cummulative 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 cummulative 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*

**__annotations__ = {}**

**__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[bool]

**__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_cxxabi1011__ = <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*

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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.ConnextMicroCompatibility'

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

generic_connext_micro_compatibility_2_4_9 =
'BuiltinQosLib::Generic.ConnextMicroCompatibility.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_monitoring2 = 'BuiltinQosLib::Generic.Monitoring2'

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_rpc = 'BuiltinQosLib::Pattern.RPC'

pattern_status = 'BuiltinQosLib::Pattern.Status'

pattern_streaming = 'BuiltinQosLib::Pattern.Streaming'
```

**class** rti.connextdds.**BuiltinTopicKey**

Bases: pybind11_object

**__annotations__ = {}**

**__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.connextdds.BuiltinTopicKey) → str

Returns a string representation of the key.

**property value**

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

**class** rti.connextdds.**BuiltinTopicReaderResourceLimits**

Bases: pybind11_object

**__annotations__ = {}**

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

**__hash__ = None**

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

Create a policy with default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.BuiltinTopicReaderResourceLimits, *arg0:* rti.connextdds.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.connextdds.**ByteVector**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Compare ByteVectors for equality.

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

Get the value at the specified index.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.ByteVector, buffer: Buffer) -> None

Create a ByteVector from another Python buffer.

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

Create an empty ByteVector

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

Create a ByteVector with a preallocated size.

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

Create a copy from another ByteVector.

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

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

Create a ByteVector from a list of values.

**__iter__** (*self:* rti.connextdds.ByteVector) → Iterator[int]

Iterate over the contents of the vector.

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

Get the length of the ByteVector.

**__module__ = 'rti.connextdds'**

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

Compare ByteVectors for inequality.

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

Set the value at the specified index.

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

Resize ByteVector to 0.

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

Resize ByteVector.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

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

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.CdrPaddingKind.CdrPaddingKind) → int

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

**__int__**(*self:* rti.connextdds.CdrPaddingKind.CdrPaddingKind) → int

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

**__module__ = 'rti.connextdds'**

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

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.CdrPaddingKind.CdrPaddingKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

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

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

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__** (*self:* rti.connextdds.CdrPaddingKind) → *rti.connextdds.CdrPaddingKind.CdrPaddingKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.CdrPaddingKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**ChannelSettings**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

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

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ChannelSettings, *arg0:* rti.connextdds.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.connextdds.**ChannelSettingsSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.ChannelSettingsSeq, *x:* rti.connextdds.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[*rti.connextdds.ChannelSettings*]

**__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*

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.CharSeq, *x: str*) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.CharSeq, arg0: int) -> str

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

**__hash__ = None**

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

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

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

Overloaded function.

1. __init__(self: rti.connextdds.CharSeq, arg0: Buffer) -> None
2. __init__(self: rti.connextdds.CharSeq) -> None
3. __init__(self: rti.connextdds.CharSeq, arg0: rti.connextdds.CharSeq) -> None

Copy constructor

4. __init__(self: rti.connextdds.CharSeq, arg0: Iterable) -> None
5. __init__(self: rti.connextdds.CharSeq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.CharSeq) → Iterator[str]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

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

Return the canonical string representation of this list.

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.CharSeq, arg0: int, arg1: str) -> None
2. __setitem__(self: rti.connextdds.CharSeq, arg0: slice, arg1: rti.connextdds.CharSeq) -> None

Assign list elements using a slice object

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

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

Add an item to the end of the list

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

Clear the contents

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

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

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

Overloaded function.

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

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

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

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

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

Insert an item at a given position.

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

Overloaded function.

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

Remove and return the last item

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

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.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.connextdds.CharSeq, *arg0: int*) → None

resizes the vector to the given size

**class** rti.connextdds.**CharType**

Bases: *DynamicType*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Get the singleton for CharType

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**class** rti.connextdds.**CoherentAccess**

Bases: pybind11_object

**__annotations__ = {}**

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

Start a context managed coherent access block.

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

End a context managed coherent access block.

**__init__**(*self:* rti.connextdds.CoherentAccess, *subscriber:* rti.connextdds.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.connextdds'**

**end**(*self:* rti.connextdds.CoherentAccess) → None

Explicitly indicate that the application has finished accessing the data samples in DataReader objects managed by the Subscriber.

**class** rti.connextdds.**CoherentSet**

Bases: pybind11_object

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.CoherentSet) → *rti.connextdds.CoherentSet*

Context manage the CoherentSet.

**__exit__**(*self:* rti.connextdds.CoherentSet, *arg0: object*, *arg1: object*, *arg2: object*) → None

**__init__**(*self:* rti.connextdds.CoherentSet, *publisher:* rti.connextdds.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.connextdds'**

**end**(*self:* rti.connextdds.CoherentSet) → None

Explicitly terminate a coherent set initiated by the CoherentSet constructor.

**class** rti.connextdds.**CoherentSetInfo**

Bases: pybind11_object

**UNKNOWN = <rti.connextdds.CoherentSetInfo object>**

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Create a default CoherentSetInfo.

2. __init__(self: rti.connextdds.CoherentSetInfo, group_guid: rti.connextdds.Guid, coherent_set_sequence_number: rti.connextdds.SequenceNumber, group_coherent_set_sequence_number: rti.connextdds.SequenceNumber, incomplete_coherent_set: bool) -> None

Create a CoherentSetInfo with the specified parameters.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**__str__**(*self:* rti.connextdds.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.connextdds.**CollectionType**

Bases: *DynamicType*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.CollectionType, *arg0:* rti.connextdds.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.

**__annotations__ = {}**

**__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.connextdds.CompressionIdMask) -> rti.connextdds.CompressionIdMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.CompressionIdMask, pos: int) -> rti.connextdds.CompressionIdMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.CompressionIdMask) -> rti.connextdds.CompressionIdMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.CompressionIdMask, pos: int) -> rti.connextdds.CompressionIdMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.CompressionIdMask) -> rti.connextdds.CompressionIdMask

Set all bits in the mask.

2. set(self: rti.connextdds.CompressionIdMask, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.CompressionIdMask, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all** (*self:* rti.connextdds.CompressionIdMask) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.CompressionIdMask) → bool

Test if any bits are set.

**test_none** (*self:* rti.connextdds.CompressionIdMask) → bool

Test if none of the bits are set.

**class** rti.connextdds.**CompressionSettings**

Bases: pybind11_object

**COMPRESSION_LEVEL_BEST_COMPRESSION = 10**

**COMPRESSION_LEVEL_BEST_SPEED = 1**

**COMPRESSION_LEVEL_DEFAULT = 10**

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Create a CompressionSettings object with default settings.

2. __init__(self: rti.connextdds.CompressionSettings, compression_ids: rti.connextdds.CompressionIdMask) -> None

Creates an instance with the given compression_ids.

3. __init__(self: rti.connextdds.CompressionSettings, compression_ids: rti.connextdds.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.connextdds'**

**__ne__**(*self:* rti.connextdds.CompressionSettings, *arg0:* rti.connextdds.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.connextdds.**Condition**

Bases: *ICondition*

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**ConditionSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.ConditionSeq, *x:* rti.connextdds.Condition) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ConditionSeq, arg0: int) -> rti.connextdds.Condition

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__**(*self:* rti.connextdds.ConditionSeq) → Iterator[*rti.connextdds.Condition*]

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

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.ConditionSeq, arg0: int, arg1: rti.connextdds.Condition) -> None

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

Assign list elements using a slice object

**append** (*self:* rti.connextdds.ConditionSeq, *x:* rti.connextdds.Condition) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.ConditionSeq, *x:* rti.connextdds.Condition) → int

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.ConditionSeq, *i: int*, *x:* rti.connextdds.Condition) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.ConditionSeq) -> rti.connextdds.Condition

Remove and return the last item

2. pop(self: rti.connextdds.ConditionSeq, i: int) -> rti.connextdds.Condition

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.ConditionSeq, *x:* rti.connextdds.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.connextdds.**ContentFilterBase**

Bases: pybind11_object

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**ContentFilterProperty**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ContentFilterProperty, *arg0:* rti.connextdds.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.connextdds.**ContentFilteredTopic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.ContentFilteredTopic, topic: rti.connextdds.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.ContentFilteredTopic, topic_description: rti.connextdds.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**append_to_expression_parameter**(*self:* rti.connextdds.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.ContentFilteredTopic*]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

**remove_from_expression_parameter**(*self:* rti.connextdds.ContentFilteredTopic, *index: int*, *remove_term: str*) → None

Removes the specified term from the parameter at the provided index.

**set_filter**(*self:* rti.connextdds.ContentFilteredTopic, *arg0:* rti.connextdds.Filter) → None

Set the filter.

**property topic**

Get the underlying Topic.

**class** rti.connextdds.**ContentFilteredTopicSeq**

Bases: pybind11_object

**__add__**(*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: list[*rti.connextdds.ContentFilteredTopic*]*) → list[*rti.connextdds.ContentFilteredTopic*]

**__annotations__ = {}**

**__bool__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*) → bool

Check whether the list is nonempty

**__contains__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *x:* rti.connextdds.ContentFilteredTopic) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: list[rti.connextdds.ContentFilteredTopic], arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: list[rti.connextdds.ContentFilteredTopic], arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: list[*rti.connextdds.ContentFilteredTopic*]*) → bool

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

Overloaded function.

1. __getitem__(self: list[rti.connextdds.ContentFilteredTopic], s: slice) -> list[rti.connextdds.ContentFilteredTopic]

Retrieve list elements using a slice object

2. __getitem__(self: list[rti.connextdds.ContentFilteredTopic], arg0: int) -> rti.connextdds.ContentFilteredTopic

**__hash__ = None**

**__iadd__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: list[*rti.connextdds.ContentFilteredTopic*]*) → list[*rti.connextdds.ContentFilteredTopic*]

**__imul__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: int*) → list[*rti.connextdds.ContentFilteredTopic*]

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

Overloaded function.

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

2. __init__(self: rti.connextdds.ContentFilteredTopicSeq, arg0: list[rti.connextdds.ContentFilteredTopic]) -> None

Copy constructor

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

**__iter__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*) → Iterator[*rti.connextdds.ContentFilteredTopic*]

**__len__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: int*) → list[*rti.connextdds.ContentFilteredTopic*]

**__ne__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: list[*rti.connextdds.ContentFilteredTopic*]*) → bool

**__rmul__** (*self: list[*rti.connextdds.ContentFilteredTopic*]*, *arg0: int*) → list[*rti.connextdds.ContentFilteredTopic*]

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

Overloaded function.

1. __setitem__(self: list[rti.connextdds.ContentFilteredTopic], arg0: int, arg1: rti.connextdds.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.ContentFiltered-Topic

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.

**__annotations__ = {}**

**__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.connextdds.Cookie) -> None

Creates an empty Cookie.

2. __init__(self: rti.connextdds.Cookie, bytes: rti.connextdds.ByteVector) -> None

Cookie from bytes.

**__module__ = 'rti.connextdds'**

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

Compare Cookies for inequality

**property value**

Retrieve the bytes of the Cookie's contents.

**class** rti.connextdds.**CookieSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.CookieSeq, *x:* rti.connextdds.Cookie) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.CookieSeq, arg0: int) -> rti.connextdds.Cookie

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.CookieSeq) → Iterator[*rti.connextdds.Cookie*]

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

**__module__ = 'rti.connextdds'**

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

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

**__repr__** (*self:* rti.connextdds.CookieSeq) → str

Return the canonical string representation of this list.

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.CookieSeq, arg0: int, arg1: rti.connextdds.Cookie) -> None
2. __setitem__(self: rti.connextdds.CookieSeq, arg0: slice, arg1: rti.connextdds.CookieSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.CookieSeq, *x:* rti.connextdds.Cookie) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.CookieSeq, *x:* rti.connextdds.Cookie) → int

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

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

Overloaded function.

1. extend(self: rti.connextdds.CookieSeq, L: rti.connextdds.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.

**__annotations__ = {}**

**__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[*rti.connextdds.Cookie*]

Iterate over the contents of the vector.

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

Get the length of the CookieVector.

**__module__ = 'rti.connextdds'**

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

Compare CookieVectors for inequality.

**__setitem__** (*self:* rti.connextdds.CookieVector, *arg0: int*, *arg1:* rti.connextdds.Cookie) → None

Set the value at the specified index.

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

Resize CookieVector to 0.

**resize** (*self:* rti.connextdds.CookieVector, *size: int*) → None

Resize CookieVector.

**class** rti.connextdds.**DataReader**

Bases: *IDataReader*

**class LoanedSample**

Bases: pybind11_object

**__annotations__ = {}**

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

Overloaded function.

1. __init__(self: rti.connextdds.DataReader.LoanedSample) -> None

Basic constructor

2. __init__(self: rti.connextdds.DataReader.LoanedSample, data: rti.connextdds.UserDataSample, info: rti.connextdds.SampleInfo) -> None

Construct LoanedSample with data and info.

**__iter__** (*self:* rti.connextdds.DataReader.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`

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.DataReader.LoanedSamples) → *rti.connextdds.DataReader.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

**__exit__** (*self:* rti.connextdds.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[*rti.connextdds.DataReader.LoanedSample*]

**__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

**__annotations__ = {}**

**__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.connextdds.DataReader.Selector) → list

Read copies of all available valid data based on Selector settings.

**read_loaned**(*self:* rti.connextdds.DataReader.Selector) → *rti.connextdds.DataReader.LoanedSamples*

(Advanced) Read data as a collection of loaned data in C format and info objects based on Selector settings

**state**(*self:* rti.connextdds.DataReader.Selector, *state:* rti.connextdds.DataState) → *rti.connextdds.DataReader.Selector*

Select samples with a specified data state.

**take**(*self:* rti.connextdds.DataReader.Selector) → list

Take copies of all available data and info based on Selector settings.

**take_data**(*self:* rti.connextdds.DataReader.Selector) → list

Take copies of all available valid data based on Selector settings.

**take_loaned**(*self:* rti.connextdds.DataReader.Selector) → *rti.connextdds.DataReader.LoanedSamples*

(Advanced) Take data as a collection of loaned data in C format and info objects based on Selector settings

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.DataReader) → *rti.connextdds.DataReader*

Enter a context for this DataReader, to be cleaned up on exiting context

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

Test for equality.

**__exit__**(*self:* rti.connextdds.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.DataReader, topic: rti.connextdds.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. __init__(self: rti.connextdds.DataReader, sub: rti.connextdds.Subscriber, topic: rti.connextdds.Topic) -> None

Create a DataReader in a subscriber with default QoS.

3. __init__(self: rti.connextdds.DataReader, topic: rti.connextdds.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DataReaderListener = None, mask: rti.connextdds.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 subscribrer 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

**__annotations__ = {}**

**__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.connextdds.**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>**

**__annotations__ = {}**

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

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind) → int

**__init__**(*self:* rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

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

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind, *state: int*) → None

**__str__**(*self: object*) → 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>**

**__annotations__ = {}**

**__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.connextdds.DataReaderInstanceRemovalKind, *arg0:* rti.connextdds.DataReaderInstanceRemovalKind) → bool

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.DataReaderInstanceRemovalKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**DataReaderListener**

Bases: pybind11_object

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**on_data_available** (*self:* rti.connextdds.DataReaderListener, *arg0:* rti.connextdds.DataReader) → None

Data available callback.

**on_liveliness_changed** (*self:* rti.connextdds.DataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed** (*self:* rti.connextdds.DataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos** (*self:* rti.connextdds.DataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost** (*self:* rti.connextdds.DataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.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

**__annotations__ = {}**

**__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.

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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.connextdds.DataReaderQos, *format:* rti.connextdds.QosPrintFormat = *QosPrintFormat()*, *base: Optional[*rti.connextdds.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.connextdds.**DataReaderResourceLimits**

Bases: pybind11_object

**AUTO_MAX_TOTAL_INSTANCES = 0**

**__annotations__ = {}**

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

**__hash__ = None**

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

Create a default DataReaderResourceLimits policy with default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.DataReaderResourceLimits, *arg0:* rti.connextdds.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 lazily_allocate_batch_resources**

Determines whether the DataReader lazily allocates batch resources.

**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 shmem_ref_transfer_mode_attached_segment_allocation**

The initial number of TopicQueries allocated by a DataReader.

**class**
rti.connextdds.**DataReaderResourceLimitsInstanceReplacementSettings**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.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*]

**__annotations__ = {}**

**__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.connextdds.DataReader], arg0: int) -> rti.connextdds.DataReader

**__hash__ = None**

**__iadd__** (*self: list[*rti.connextdds.DataReader*]*, *arg0: list[*rti.connextdds.DataReader*]*) → list[*rti.connextdds.DataReader*]

**__imul__** (*self: list[*rti.connextdds.DataReader*]*, *arg0: int*) → list[*rti.connextdds.DataReader*]

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

Overloaded function.

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

2. __init__(self: rti.connextdds.DataReaderSeq, arg0: list[rti.connextdds.DataReader]) -> None

Copy constructor

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

**__iter__** (*self: list[*rti.connextdds.DataReader*]*) → Iterator[*rti.connextdds.DataReader*]

**__len__** (*self: list[*rti.connextdds.DataReader*]*) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self: list[*rti.connextdds.DataReader*]*, *arg0: int*) → list[*rti.connextdds.DataReader*]

**__ne__** (*self: list[*rti.connextdds.DataReader*]*, *arg0: list[*rti.connextdds.DataReader*]*) → bool

**__rmul__** (*self: list[*rti.connextdds.DataReader*]*, *arg0: int*) → list[*rti.connextdds.DataReader*]

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

Overloaded function.

1. __setitem__(self: list[rti.connextdds.DataReader], arg0: int, arg1: rti.connextdds.DataReader) -> None

2. __setitem__(self: list[rti.connextdds.DataReader], arg0: slice, arg1: list[rti.connextdds.DataReader]) -> None

Assign list elements using a slice object

**append** (*self: list[*rti.connextdds.DataReader*]*, *x:* rti.connextdds.DataReader) → None

Add an item to the end of the list

**clear** (*self: list[*rti.connextdds.DataReader*]*) → None

Clear the contents

**count** (*self: list[*rti.connextdds.DataReader*]*, *x:* rti.connextdds.DataReader) → int

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

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

Overloaded function.

1. extend(self: list[rti.connextdds.DataReader], L: list[rti.connextdds.DataReader]) -> None

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

2. extend(self: list[rti.connextdds.DataReader], L: Iterable) -> None

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

**insert** (*self: list[*rti.connextdds.DataReader*]*, *i: int*, *x:* rti.connextdds.DataReader) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: list[rti.connextdds.DataReader]) -> rti.connextdds.DataReader

Remove and return the last item

2. pop(self: list[rti.connextdds.DataReader], i: int) -> rti.connextdds.DataReader

Remove and return the item at index `i`

**remove** (*self: list[*rti.connextdds.DataReader*]*, *x:* rti.connextdds.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.connextdds.**DataRepresentation**

Bases: `pybind11_object`

**AUTO_ID = -1**

**XCDR = 0**

**XCDR2 = 2**

**XML = 1**

**__annotations__ = {}**

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

**__hash__ = None**

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

Overloaded function.

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

Creates an instance with one element, AUTO_ID.

2. __init__(self: rti.connextdds.DataRepresentation, representations: rti.connextdds.Int16Seq) -> None

Creates an instance with a sequence of Data Representation IDs.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.DataRepresentation, *arg0:* rti.connextdds.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 commited back to the policy object via the property's setter.

**class** rti.connextdds.**DataState**

Bases: pybind11_object

**__annotations__ = {}**

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

Compare DataState objects for equality.

**__hash__ = None**

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

Overloaded function.

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

Create a DataState with InstanceState.any(), ViewState.any(), and SampleState.ANY

2. __init__(self: rti.connextdds.DataState, sample_state: rti.connextdds.SampleState) -> None

Create a DataState with InstanceState.any(), ViewState.any(), and the provided SampleState.

3. __init__(self: rti.connextdds.DataState, view_state: rti.connextdds.ViewState) -> None

Create a DataState with InstanceState.any(), SampleState.any(), and the provided ViewState.

4. __init__(self: rti.connextdds.DataState, instance_state: rti.connextdds.InstanceState) -> None

Create a DataState with ViewState.any(), SampleState.any(), and the provided InstanceState.

5. __init__(self: rti.connextdds.DataState, sample_state: rti.connextdds.SampleState, view_state: rti.connextdds.ViewState, instance_state: rti.connextdds.InstanceState) -> None

Create a DataState with the provided SampleState, ViewState, and InstanceState.

**__lshift__**(**args*, ***kwargs*)

Overloaded function.

1. __lshift__(self: rti.connextdds.DataState, arg0: rti.connextdds.SampleState) -> rti.connextdds.DataState

Set the SampleState for the DataState.

2. __lshift__(self: rti.connextdds.DataState, arg0: rti.connextdds.ViewState) -> rti.connextdds.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

**__annotations__ = {}**

**__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.

**topic_queries = <rti.connextdds.DataStateEx object>**

**property view_state**

The ViewState of the DataStateEx.

**class** rti.connextdds.**DataTag**

Bases: pybind11_object

**__annotations__ = {}**

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

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

Test for equality.

**__getitem__**(*self:* rti.connextdds.DataTag, *arg0: str*) → Optional[str]

**__hash__ = None**

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

Overloaded function.

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

Creates a policy with an empty sequence of tags.

2. __init__(self: rti.connextdds.DataTag, entries: rti.connextdds.StringPairSeq) -> None

Adds tags from a list.

3. __init__(self: rti.connextdds.DataTag, entries: dict) -> None

Adds tags from a dictionary.

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**__setitem__** (*self:* rti.connextdds.DataTag, *arg0: str*, *arg1: str*) → None

**exists** (*self:* rti.connextdds.DataTag, *key: str*) → bool

Returns true if a tag exists.

**get** (*self:* rti.connextdds.DataTag, *key: str*) → str

Returns the value of a tag identified by a key if it exists.

**get_all** (*self:* rti.connextdds.DataTag) → *rti.connextdds.StringMap*

Retrieves a copy of all the entries.

**remove** (*self:* rti.connextdds.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*

**__annotations__ = {}**

**__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 thecalling 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 Connext 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 Connext 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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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.ReliableReaderActivityChanged-Status) → 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.connextdds.DataWriterListener, *arg0:* rti.connextdds.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class** rti.connextdds.**DataWriterProtocol**

Bases: pybind11_object

**__annotations__ = {}**

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

**__hash__ = None**

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

Create a default DataWriterProtocol policy.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.DataWriterProtocol, *arg0:* rti.connextdds.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.connextdds.**DataWriterProtocolStatus**

Bases: pybind11_object

Information about the DataWriter protocol status

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__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.connextdds.DataWriterQos, *arg0:* rti.connextdds.DataWriterQos) → bool

Test for inequality.

**__rshift__**(**args*, ***kwargs*)

Overloaded function.

1. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Durability) -> rti.connextdds.DataWriterQos

Get the DurabilityQoS.

2. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DurabilityService) -> rti.connextdds.DataWriterQos

Get the DurabilityServiceQoS.

3. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Deadline) -> rti.connextdds.DataWriterQos

Get the DeadlineQoS.

4. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.LatencyBudget) -> rti.connextdds.DataWriterQos

Get the LatencyBudgetQoS.

5. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Liveliness) -> rti.connextdds.DataWriterQos

Get the LivelinessQoS.

6. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Reliability) -> rti.connextdds.DataWriterQos

Get the ReliabilityQoS.

7. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DestinationOrder) -> rti.connextdds.DataWriterQos

Get the DestinationOrderQoS.

8. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.History) -> rti.connextdds.DataWriterQos

Get the HistoryQoS.

9. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.ResourceLimits) -> rti.connextdds.DataWriterQos

Get the ResourceLimitsQoS.

10. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TransportPriority) -> rti.connextdds.DataWriterQos

Get the TransportPriorityQoS.

11. __rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Lifespan) -> rti.connextdds.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.connextdds.DataWriterQos, *format:* rti.connextdds.QosPrintFormat = *QosPrintFormat()*, *base: Optional[*rti.connextdds.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.connextdds.**DataWriterResourceLimits**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.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_remote_readers**

The initial number of remote readers supported by a 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.connextdds.**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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**UNREGISTERED = <DataWriterResourceLimitsInstanceReplacementKind.UNREGISTERED: 0>**

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__**(*self:* rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind) → *rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__**(*self:* rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**DataWriterSeq**

Bases: pybind11_object

**__add__**(*self: list[*rti.connextdds.DataWriter*]*, *arg0: list[*rti.connextdds.DataWriter*]*) → list[*rti.connextdds.DataWriter*]

**__annotations__ = {}**

**__bool__**(*self: list[*rti.connextdds.DataWriter*]*) → bool

Check whether the list is nonempty

**__contains__**(*self: list[*rti.connextdds.DataWriter*]*, *x:* rti.connextdds.DataWriter) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: list[rti.connextdds.DataWriter], arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: list[rti.connextdds.DataWriter], arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self: list[*rti.connextdds.DataWriter*]*, *arg0: list[*rti.connextdds.DataWriter*]*) → bool

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

Overloaded function.

1. __getitem__(self: list[rti.connextdds.DataWriter], s: slice) -> list[rti.connextdds.DataWriter]

Retrieve list elements using a slice object

2. __getitem__(self: list[rti.connextdds.DataWriter], arg0: int) -> rti.connextdds.DataWriter

**__hash__ = None**

**__iadd__** (*self: list[*rti.connextdds.DataWriter*]*, *arg0: list[*rti.connextdds.DataWriter*]*) → list[*rti.connextdds.DataWriter*]

**__imul__** (*self: list[*rti.connextdds.DataWriter*]*, *arg0: int*) → list[*rti.connextdds.DataWriter*]

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

Overloaded function.

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

2. __init__(self: rti.connextdds.DataWriterSeq, arg0: list[rti.connextdds.DataWriter]) -> None

Copy constructor

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

**__iter__** (*self: list[*rti.connextdds.DataWriter*]*) → Iterator[*rti.connextdds.DataWriter*]

**__len__** (*self: list[*rti.connextdds.DataWriter*]*) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self: list[*rti.connextdds.DataWriter*]*, *arg0: int*) → list[*rti.connextdds.DataWriter*]

**__ne__** (*self: list[*rti.connextdds.DataWriter*]*, *arg0: list[*rti.connextdds.DataWriter*]*) → bool

**__rmul__** (*self: list[*rti.connextdds.DataWriter*]*, *arg0: int*) → list[*rti.connextdds.DataWriter*]

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

Overloaded function.

1. __setitem__(self: list[rti.connextdds.DataWriter], arg0: int, arg1: rti.connextdds.DataWriter) -> None

2. __setitem__(self: list[rti.connextdds.DataWriter], arg0: slice, arg1: list[rti.connextdds.DataWriter]) -> None

Assign list elements using a slice object

**append** (*self: list[*rti.connextdds.DataWriter*]*, *x:* rti.connextdds.DataWriter) → None

Add an item to the end of the list

**clear** (*self: list[*rti.connextdds.DataWriter*]*) → None

Clear the contents

**count** (*self: list[*rti.connextdds.DataWriter*]*, *x:* rti.connextdds.DataWriter) → int

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

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

Overloaded function.

1. extend(self: list[rti.connextdds.DataWriter], L: list[rti.connextdds.DataWriter]) -> None

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

2. extend(self: list[rti.connextdds.DataWriter], L: Iterable) -> None

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

**insert** (*self: list[*rti.connextdds.DataWriter*]*, *i: int*, *x:* rti.connextdds.DataWriter) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: list[rti.connextdds.DataWriter]) -> rti.connextdds.DataWriter

Remove and return the last item

2. pop(self: list[rti.connextdds.DataWriter], i: int) -> rti.connextdds.DataWriter

Remove and return the item at index `i`

**remove** (*self: list[*rti.connextdds.DataWriter*]*, *x:* rti.connextdds.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.connextdds.**DataWriterShmemRefTransferModeSettings**

Bases: `pybind11_object`

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Creates an instance with the default settings.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**property enable_data_consistency_check**

Controls if samples can be checked for consistency.

**class** rti.connextdds.**DataWriterTransferMode**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Creates a DataWriterTransferMode qos policy with default values.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**property shmem_ref_settings**

Settings related to transferring data using shared memory references.

**class** rti.connextdds.**Database**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Create a Database QoS policy with default settings.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Database, *arg0:* rti.connextdds.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.connextdds.**Deadline**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Creates the default deadline, with an infinite period.

2. __init__(self: rti.connextdds.Deadline, period: rti.connextdds.Duration) -> None

Creates a deadline policy with the specified period.

3. __init__(self: rti.connextdds.Deadline, sec: int, nanosec: int) -> None

Creates a deadline policy with the specified period.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**property period**

The duration of the deadline period.

**class** rti.connextdds.**DestinationOrder**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Creates the default policy.

2. __init__(self: rti.connextdds.DestinationOrder, kind: rti.connextdds.DestinationOrderKind) -> None

Creates a policy with the specified destination order kind.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.DestinationOrder, *arg0:* rti.connextdds.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.connextdds.**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 Connext 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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__** (*self:* rti.connextdds.DestinationOrderKind) → *rti.connextdds.DestinationOrderKind.DestinationOrderKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.DestinationOrderKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**INSTANCE = <DestinationOrderScopeKind.INSTANCE: 0>**

**TOPIC = <DestinationOrderScopeKind.TOPIC: 1>**

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__** (*self:* rti.connextdds.DestinationOrderScopeKind) → *rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.DestinationOrderScopeKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**Discovery**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Creates the default policy.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Discovery, *arg0:* rti.connextdds.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.connextdds.**DiscoveryConfig**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Creates the default policy.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.DiscoveryConfig, *arg0:* rti.connextdds.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 "unreachable" 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 request_types_filter**

Topic name pattern for requesting all types via Type Lookup Service.

**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.connextdds.**DiscoveryConfigBuiltinChannelKindMask**

Bases: pybind11_object

**ALL = 1111000000000011**

**NONE = 0000000000000000**

**SERVICE_REQUEST = 0000000000000011**

**TYPE_LOOKUP_SERVICE = 1111000000000000**

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

Bitwise logical AND of masks.

**__annotations__ = {}**

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

Test if any bits are set.

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

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

Compare masks for equality.

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

Get individual mask bit.

**__hash__ = None**

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

Set mask to logical AND with another mask.

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

Left shift bits in mask.

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

Overloaded function.

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

Create a DiscoveryConfigBuiltinChannelKindMask equivalent to DiscoveryConfigBuiltinChannelKindMask.NONE

2. __init__(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connextdds.DiscoveryConfigBuiltinChannelKindMask*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0: int*) → *rti.connextdds.DiscoveryConfigBuiltinChannelKindMask*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connextdds.DiscoveryConfigBuiltinChannelKindMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0: int*) → *rti.connextdds.DiscoveryConfigBuiltinChannelKindMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

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

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connextdds.DiscoveryConfigBuiltinChannelKindMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0: int*) → *rti.connextdds.DiscoveryConfigBuiltinChannelKindMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connextdds.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.

**__annotations__ = {}**

**__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.connextdds.DiscoveryConfigBuiltinPluginKindMask) → int

Convert mask to int.

**__ior__**(*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → *rti.connextdds.DiscoveryConfigBuiltinPluginKindMask*

Set mask to logical OR with another mask.

**__irshift__**(*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0: int*) → *rti.connextdds.DiscoveryConfigBuiltinPluginKindMask*

Right shift bits in mask.

**__ixor__**(*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → *rti.connextdds.DiscoveryConfigBuiltinPluginKindMask*

Set mask to logical XOR with another mask.

**__lshift__**(*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0: int*) → *rti.connextdds.DiscoveryConfigBuiltinPluginKindMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

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

Compare masks for inequality.

**__or__**(*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → *rti.connextdds.DiscoveryConfigBuiltinPluginKindMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0: int*) → *rti.connextdds.DiscoveryConfigBuiltinPluginKindMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, *arg0:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → *rti.connextdds.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.connextdds.DiscoveryConfigBuiltinPluginKindMask) -> rti.connextdds.DiscoveryConfigBuiltinPluginKindMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, pos: int) -> rti.connextdds.DiscoveryConfigBuiltinPluginKindMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) -> rti.connextdds.DiscoveryConfigBuiltinPluginKindMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, pos: int) -> rti.connextdds.DiscoveryConfigBuiltinPluginKindMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) -> rti.connextdds.DiscoveryConfigBuiltinPluginKindMask

Set all bits in the mask.

2. set(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.DiscoveryConfigBuiltinPluginKindMask, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → bool

Test if any bits are set.

**test_none** (*self:* rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → bool

Test if none of the bits are set.

**class** rti.connextdds.**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).

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.DomainParticipant) → *rti.connextdds.DomainParticipant*

Enter a context for this Domain Participant, to be cleaned up on exiting context

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

Test for equality.

**__exit__** (*self:* rti.connextdds.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.connextdds.DomainParticipant, domain_id: int) -> None

Create a new DomainParticipant with default QoS.

2. __init__(self: rti.connextdds.DomainParticipant, domain_id: int, qos: rti.connextdds.DomainParticipantQos, listener: rti.connextdds.DomainParticipantListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a new DomainParticipant

3. __init__(self: rti.connextdds.DomainParticipant, entity: rti.connextdds.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 toparticipants 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.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.FlowController*]

Find a FlowController configured in this DomainParticipant.

**find_publisher** (*self:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.Publisher*]

Lookup a Publisher within the DomainParticipant by its entity name.

**find_publishers** (*self:* rti.connextdds.DomainParticipant) → *rti.connextdds.PublisherSeq*

Find all Publishers within the DomainParticipant.

**find_registered_content_filters** (*self:* rti.connextdds.DomainParticipant) → *rti.connextdds.StringSeq*

Retrieve a list of all registered content filter names.

**find_subscriber** (*self:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.Subscriber*]

Find a Subscriber in the DomainParticipant by its entity name.

**find_subscribers** (*self:* rti.connextdds.DomainParticipant) → *rti.connextdds.SubscriberSeq*

Find all subscribers within the DomainParticipant.

**find_topics** (*self:* rti.connextdds.DomainParticipant) → *rti.connextdds.AnyTopicSeq*

Find all Topics in the DomainParticipant.

**ignore_datareader** (*self:* rti.connextdds.DomainParticipant, *handle:* rti.connextdds.InstanceHandle) → None

Ignore a DataReader matching the provided handle.

**ignore_datareaders** (*self:* rti.connextdds.DomainParticipant, *handles:* rti.connextdds.InstanceHandleSeq) → None

Ignore a list of DataReaders specified by their handles.

**ignore_datawriter** (*self:* rti.connextdds.DomainParticipant, *handle:* rti.connextdds.InstanceHandle) → None

Ignore a DataWriter matching the provided handle.

**ignore_datawriters** (*self:* rti.connextdds.DomainParticipant, *handles:* rti.connextdds.InstanceHandleSeq) → None

Ignore a list of DataWriters specified by their handles.

**ignore_participant** (*self:* rti.connextdds.DomainParticipant, *handle:* rti.connextdds.InstanceHandle) → None

Ignore a DomainParticipant given it's handle.

**ignore_participants** (*self:* rti.connextdds.DomainParticipant, *arg0:* rti.connextdds.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 ContentFiltertedTopic.

**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.connextdds.DomainParticipant, *name: str*, *type:* rti.connextdds.DynamicType, *serialization_property:* rti.connextdds.DynamicDataTypeSerializationProperty = *DynamicDataTypeSerializationProperty.DEFAULT*) → None

Registers a DynamicType with specific serialization properties.

**remove_peer** (*self:* rti.connextdds.DomainParticipant, *peer: str*) → None

Remove a peer participant from this list that this DomainParticipant will attempt to communicate with.

**remove_peers** (*self:* rti.connextdds.DomainParticipant, *peers:* rti.connextdds.StringSeq) → None

Remove a sequence of peers from the contact list.

**resume_endpoint_discovery** (*self:* rti.connextdds.DomainParticipant, *handle:* rti.connextdds.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.connextdds.DomainParticipant, *listener:* rti.connextdds.DomainParticipantListener, *event_mask:* rti.connextdds.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.connextdds.DomainParticipant, *name: str*) → None

Unregister content filter previously registered to this DomainParticipant.

**unregister_type** (*self:* rti.connextdds.DomainParticipant, *name: str*) → None

Unregister a type previously registered to this DomainParticipant.

**class** rti.connextdds.**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'**

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self:* rti.connextdds.DomainParticipantConfigParams, *arg0:* rti.connextdds.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.connextdds.**DomainParticipantFactoryQos**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.DomainParticipantFactoryQos, *arg0:* rti.connextdds.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*

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**on_invalid_local_status_advance_notice**(*self:* rti.connextdds.DomainParticipantListener, *arg0:* rti.connextdds.DomainParticipant, *arg1:* rti.connextdds.InvalidLocalAdvanceNoticeStatus) → None

On invalid local status advance notice callback

**class** rti.connextdds.**DomainParticipantProtocolStatus**

Bases: pybind11_object

Protocol status of a DomainParticipant

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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.connextdds.**DomainParticipantResourceLimits**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Create a default policy.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.DomainParticipantResourceLimits, *arg0:* rti.connextdds.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.connextdds.**DomainParticipantSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.DomainParticipantSeq, *x:* rti.connextdds.DomainParticipant) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.DomainParticipantSeq, arg0: int) -> rti.connextdds.DomainParticipant

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.DomainParticipantSeq) → Iterator[*rti.connextdds.DomainParticipant*]

**__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.

1. __setitem__(self: rti.connextdds.DomainParticipantSeq, arg0: int, arg1: rti.connextdds.DomainParticipant) -> None

2. __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.

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

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

2. 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**

**__annotations__ = {}**

**__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.connextdds.Durability object>**

**transient_local = <rti.connextdds.Durability object>**

**volatile = <rti.connextdds.Durability object>**

**property writer_depth**

The number of samples a durable DataWriter will send to a late joining DataReader.

**class** rti.connextdds.**DurabilityKind**

Bases: pybind11_object

**class DurabilityKind**

Bases: pybind11_object

Members:

VOLATILE : [default] RTI Connext 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 Connext will only attempt to provide the data to existing subscribers.

This option does not require RTI Persistence Service.

TRANSIENT_LOCAL : RTI Connext 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 Connext 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 Connext 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 Connext 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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**PERSISTENT = <DurabilityKind.PERSISTENT: 3>**

**TRANSIENT = <DurabilityKind.TRANSIENT: 2>**

**TRANSIENT_LOCAL = <DurabilityKind.TRANSIENT_LOCAL: 1>**

**VOLATILE = <DurabilityKind.VOLATILE: 0>**

**__annotations__ = {}**

**__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.connextdds.DurabilityKind, *arg0:* rti.connextdds.DurabilityKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__**(*self:* rti.connextdds.DurabilityKind) → *rti.connextdds.DurabilityKind.DurabilityKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.DurabilityKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**DurabilityService**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Creates the default policy.

2. __init__(self: rti.connextdds.DurabilityService, service_cleanup_delay: rti.connextdds.Duration, history_kind: rti.connextdds.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.

**__annotations__ = {}**

**__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.

**__repr__** (*self:* rti.connextdds.Duration) → str
Return the detailed string representation of the Duration.

**__rmul__** (*self:* rti.connextdds.Duration, *arg0: int*) → *rti.connextdds.Duration*
Multiply a Duration by an unsigned int.

**__str__** (*self:* rti.connextdds.Duration) → str
Return the string representation of the Duration.

**__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 = <Duration(automatic)>**

**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 greate

**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 = <Duration(infinite)>**

**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 = <Duration(sec=0, nanosec=0)>**

**class** rti.connextdds.**DynamicData**

Bases: pybind11_object

**class ContentFilter**

Bases: *ContentFilterBase*

**__annotations__ = {}**

**__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.connextdds.DynamicData.ContentFilter, *compile_data: Optional[object]*, *sample:* rti.connextdds.DynamicData, *meta_data:* rti.connextdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

**finalize**(*self:* rti.connextdds.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*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.ContentFilteredTopic, topic: rti.connextdds.DynamicData.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.DynamicData.ContentFilteredTopic, topic_description: rti.connextdds.DynamicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**append_to_expression_parameter**(*self:* rti.connextdds.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.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.DynamicData.ContentFilteredTopic*]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

**remove_from_expression_parameter**(*self:* rti.connextdds.DynamicData.ContentFilteredTopic, *index: int*, *remove_term: str*) → None

Removes the specified term from the parameter at the provided index.

**set_filter**(*self:* rti.connextdds.DynamicData.ContentFilteredTopic, *arg0:* rti.connextdds.Filter) → None

Set the filter.

**property topic**

Get the underlying Topic.

**class ContentFilteredTopicSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq, *x:* rti.connextdds.DynamicData.ContentFilteredTopic) → bool

Return true the container contains x

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

Overloaded function.

1. __delitem__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: int) -> None

Delete the list elements at index i

2. __delitem__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: slice) -> None

Delete list elements using a slice object

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

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

Overloaded function.

1. __getitem__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, s: slice) -> rti.connextdds.DynamicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.DynamicData.ContentFilteredTopic

**__hash__ = None**

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

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

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq) -> None

2. __init__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: rti.connextdds.DynamicData.ContentFilteredTopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq) → Iterator[*rti.connextdds.DynamicData.ContentFilteredTopic*]

**__len__** (*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq) → int

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: int, arg1: rti.connextdds.DynamicData.ContentFilteredTopic) -> None
2. __setitem__(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connextdds.DynamicData.ContentFilteredTopicSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq, *x:* rti.connextdds.DynamicData.ContentFilteredTopic) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.DynamicData.ContentFilteredTopicSeq, *x:* rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.DynamicData.DataReader.Selector, *max: int*) → *rti.connextdds.DynamicData.DataReader.Selector*

Limit the number of samples read/taken by the Select.

**next_instance** (*self:* rti.connextdds.DynamicData.DataReader.Selector, *previous:* rti.connextdds.InstanceHandle) → *rti.connextdds.DynamicData.DataReader.Selector*

Select the instance after the specified instance to read/take.

**read** (*self:* rti.connextdds.DynamicData.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

**read_data** (*self:* rti.connextdds.DynamicData.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

**read_loaned** (*self:* rti.connextdds.DynamicData.DataReader.Selector) → *rti.connextdds.DynamicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

**state** (*self:* rti.connextdds.DynamicData.DataReader.Selector, *state:* rti.connextdds.DataState) → *rti.connextdds.DynamicData.DataReader.Selector*

Select samples with a specified data state.

**take** (*self:* rti.connextdds.DynamicData.DataReader.Selector) → list

Take copies of available samples (data and info) based on the Selector settings.

**take_data** (*self:* rti.connextdds.DynamicData.DataReader.Selector) → list

Take copies of available valid data based on the Selector settings.

**take_loaned** (*self:* rti.connextdds.DynamicData.DataReader.Selector) → *rti.connextdds.DynamicData.LoanedSamples*

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.DynamicData.DataReader) → *rti.connextdds.DynamicData.DataReader*

Enter a context for this DataReader, to be cleaned up on exiting context

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

Test for equality.

**__exit__** (*self:* rti.connextdds.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.connextdds.DynamicData.DataReader) → list

Read copies of all available samples (data and info).

**read_data** (*self:* rti.connextdds.DynamicData.DataReader) → list

Read copies of all available valid data.

**read_loaned** (*self:* rti.connextdds.DynamicData.DataReader) → *rti.connextdds.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.connextdds.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.connextdds.DynamicData.DataReader, *listener:* rti.connextdds.DynamicData.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.DynamicData.DataReader) → list

Take copies of all available samples (data and info).

**take_async** (*condition: Optional[*ReadCondition*] = None*)

**take_data** (*self:* rti.connextdds.DynamicData.DataReader) → list

Take copies of all available valid data.

**take_data_async** (*condition: Optional[*ReadCondition*] = None*)

**take_loaned** (*self:* rti.connextdds.DynamicData.DataReader) → *rti.connextdds.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`

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.DynamicData.DataReaderSeq, *arg0:* rti.connextdds.DynamicData.DataReaderSeq) → *rti.connextdds.DynamicData.DataReaderSeq*

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

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.DataReaderSeq) -> None
2. __init__(self: rti.connextdds.DynamicData.DataReaderSeq, arg0: rti.connextdds.DynamicData.DataReaderSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.DynamicData.DataReaderSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.DynamicData.DataReaderSeq) → Iterator[*rti.connextdds.DynamicData.DataReader*]

**__len__**(*self:* rti.connextdds.DynamicData.DataReaderSeq) → int

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.DynamicData.DataReaderSeq, arg0: int, arg1: rti.connextdds.DynamicData.DataReader) -> None
2. __setitem__(self: rti.connextdds.DynamicData.DataReaderSeq, arg0: slice, arg1: rti.connextdds.DynamicData.DataReaderSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.DynamicData.DataReaderSeq, *x:* rti.connextdds.DynamicData.DataReader) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.DynamicData.DataReaderSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.DynamicData.DataReaderSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__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.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.DynamicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.DynamicData.Topic) -> None

Creates a DataWriter in a publisher with default QoS.

4. __init__(self: rti.connextdds.DynamicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.DynamicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.DynamicData.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.DynamicData.DataWriter, writer: rti.connextdds.IAnyDataWriter) -> None

Create a typed DataWriter from an AnyDataWriter.

6. __init__(self: rti.connextdds.DynamicData.DataWriter, entity: rti.connextdds.IEntity) -> None

Create a typed DataWriter from an Entity.

**__lshift__**(**args*, ***kwargs*)

Overloaded function.

1. __lshift__(self: rti.connextdds.DynamicData.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.DynamicData.DataWriter

Sets the DataWriterQos.

2. __lshift__(self: rti.connextdds.DynamicData.DataWriter, arg0: tuple[rti.connextdds.DynamicData, rti.connextdds.Time]) -> rti.connextdds.DynamicData.DataWriter

Writes a paired sample with a timestamp.

3. __lshift__(self: rti.connextdds.DynamicData.DataWriter, arg0: tuple[rti.connextdds.DynamicData, rti.connextdds.InstanceHandle]) -> rti.connextdds.DynamicData.DataWriter

Writes a paired sample with an instance handle.

4. __lshift__(self: rti.connextdds.DynamicData.DataWriter, arg0: rti.connextdds.DynamicDataTimestampedSeq) -> rti.connextdds.DynamicData.DataWriter

Writes a sequence of pairs of samples with timestamps.

5. __lshift__(self: rti.connextdds.DynamicData.DataWriter, arg0: rti.connextdds.DynamicDataSeq) -> rti.connextdds.DynamicData.DataWriter

Writes a sequence of samples.

6. __lshift__(self: rti.connextdds.DynamicData.DataWriter, arg0: rti.connextdds.DynamicData) -> rti.connextdds.DynamicData.DataWriter

Writes a sample.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**__rshift__**(*self:* rti.connextdds.DynamicData.DataWriter, *arg0:* rti.connextdds.DataWriterQos) → *rti.connextdds.DynamicData.DataWriter*

Get the DataWriterQos.

**assert_liveliness** (*self:* rti.connextdds.DynamicData.DataWriter) → None

Manually asserts the liveliness of the DataWriter.

**close** (*self:* rti.connextdds.DynamicData.DataWriter) → None

Close this DataWriter.

**create_data** (*self:* rti.connextdds.DynamicData.DataWriter) → *rti.connextdds.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.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DynamicData.DataWriter

Dispose an instance.

2. dispose_instance(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.DynamicData.DataWriter

Dispose an instance with a timestamp.

3. dispose_instance(self: rti.connextdds.DynamicData.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.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object

Dispose an instance.

2. dispose_instance_async(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object

Dispose an instance with a timestamp.

3. dispose_instance_async(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData) -> object

Dispose the instance associated with key_holder.

4. dispose_instance_async(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData, timestamp: rti.connextdds.Time) -> object

Dispose the instance associated with key_holder using a timestamp

5. dispose_instance_async(self: rti.connextdds.DynamicData.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.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 Connext 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 Connext 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

**__annotations__ = {}**

**__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.connextdds.DynamicData.DataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.DynamicData.DataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.DynamicData.DataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class DataWriterSeq**

Bases: `pybind11_object`

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

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.DynamicData.DataWriterSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.DynamicData.DataWriterSeq, *x:* rti.connextdds.DynamicData.DataWriter) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: slice) -> None

Delete list elements using a slice object

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

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

Overloaded function.

1. __getitem__(self: rti.connextdds.DynamicData.DataWriterSeq, s: slice) -> rti.connextdds.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[*rti.connextdds.DynamicData.DataWriter*]

**__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`

**__annotations__ = {}**

**__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`

**__annotations__ = {}**

**__contains__**(*self:* rti.connextdds.DynamicData.FieldsView, *arg0: str*) → bool

**__init__**(*self:* rti.connextdds.DynamicData.FieldsView, *arg0:* rti.connextdds.DynamicData) → None

**__iter__**(*self:* rti.connextdds.DynamicData.FieldsView) → *rti.connextdds.DynamicData.FieldsIterator*

**__len__**(*self:* rti.connextdds.DynamicData.FieldsView) → int

**__module__ = 'rti.connextdds'**

**__reversed__**(*self:* rti.connextdds.DynamicData.FieldsView) → *rti.connextdds.DynamicData.FieldsIterator*

**class ITopicDescription**

Bases: *IEntity*

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.DynamicData.IndexIterator, *arg0:* rti.connextdds.DynamicData, *arg1: bool*) → None

**__iter__**(*self:* rti.connextdds.DynamicData.IndexIterator) → *rti.connextdds.DynamicData.IndexIterator*

**__module__ = 'rti.connextdds'**

**__next__**(*self:* rti.connextdds.DynamicData.IndexIterator) → object

**class ItemsIterator**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.DynamicData.ItemsIterator, *arg0:* rti.connextdds.DynamicData, *arg1: bool*) → None

**__iter__**(*self:* rti.connextdds.DynamicData.ItemsIterator) → *rti.connextdds.DynamicData.ItemsIterator*

**__module__ = 'rti.connextdds'**

**__next__**(*self:* rti.connextdds.DynamicData.ItemsIterator) → tuple[str, object]

**class ItemsView**

Bases: pybind11_object

**__annotations__ = {}**

**__contains__**(*self:* rti.connextdds.DynamicData.ItemsView, *arg0: tuple[str, object]*) → bool

**__init__**(*self:* rti.connextdds.DynamicData.ItemsView, *arg0:* rti.connextdds.DynamicData) → None

**__iter__**(*self:* rti.connextdds.DynamicData.ItemsView) → *rti.connextdds.DynamicData.ItemsIterator*

**__len__**(*self:* rti.connextdds.DynamicData.ItemsView) → int

**__module__ = 'rti.connextdds'**

**__reversed__**(*self:* rti.connextdds.DynamicData.ItemsView) → *rti.connextdds.DynamicData.ItemsIterator*

**class LoanedSample**

Bases: pybind11_object

**__annotations__ = {}**

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.LoanedSample) -> None

Basic constructor

2. __init__(self: rti.connextdds.DynamicData.LoanedSample, data: rti.connextdds.DynamicData, info: rti.connextdds.SampleInfo) -> None

Construct LoanedSample with data and info.

**__iter__**(*self:* rti.connextdds.DynamicData.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

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.DynamicData.LoanedSamples) → *rti.connextdds.DynamicData.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

**__exit__**(*self:* rti.connextdds.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[*rti.connextdds.DynamicData.LoanedSample*]

**__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*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.DynamicData.NoOpDataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0:* rti.connextdds.DynamicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class NoOpTopicListener**

Bases: *TopicListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.DynamicData.NoOpTopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.DynamicData.NoOpTopicListener, *arg0:* rti.connextdds.DynamicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class Sample**

Bases: pybind11_object

**__annotations__ = {}**

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.Sample, data: rti.connextdds.DynamicData, info: rti.connextdds.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

**__annotations__ = {}**

**__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[*rti.connextdds.DynamicData.LoanedSample*]

Make a sample iterator

**__len__** (*self:* rti.connextdds.DynamicData.SharedSamples) → int

Returns the number of samples.

**__module__ = 'rti.connextdds'**

**class Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.TopicDescription, topic: rti.connextdds.DynamicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. __init__(self: rti.connextdds.DynamicData.TopicDescription, entity: rti.connextdds.IEntity) -> None

Cast a Topic to a TopicDescription.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**class TopicListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.DynamicData.TopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.DynamicData.TopicListener, *arg0:* rti.connextdds.DynamicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class TopicSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.DynamicData.TopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.DynamicData.TopicSeq, *x:* rti.connextdds.DynamicData.Topic) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: rti.connextdds.DynamicData.TopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.DynamicData.TopicSeq, arg0: slice) -> None

Delete list elements using a slice object

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

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

Overloaded function.

1. __getitem__(self: rti.connextdds.DynamicData.TopicSeq, s: slice) -> rti.connextdds.DynamicData.TopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.DynamicData.TopicSeq, arg0: int) -> rti.connextdds.DynamicData.Topic

**__hash__ = None**

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

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

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

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.TopicSeq) -> None
2. __init__(self: rti.connextdds.DynamicData.TopicSeq, arg0: rti.connextdds.DynamicData.TopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.DynamicData.TopicSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.DynamicData.TopicSeq) → Iterator[*rti.connextdds.DynamicData.Topic*]

**__len__** (*self:* rti.connextdds.DynamicData.TopicSeq) → int

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.DynamicData.TopicSeq, arg0: int, arg1: rti.connextdds.DynamicData.Topic) -> None
2. __setitem__(self: rti.connextdds.DynamicData.TopicSeq, arg0: slice, arg1: rti.connextdds.DynamicData.TopicSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.DynamicData.TopicSeq, *x:* rti.connextdds.DynamicData.Topic) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.DynamicData.TopicSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.DynamicData.TopicSeq, *x:* rti.connextdds.DynamicData.Topic) → int

Return the number of times x appears in the list

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

Overloaded function.

1. extend(self: rti.connextdds.DynamicData.TopicSeq, L: rti.connextdds.DynamicData.TopicSeq) -> None

Extend the list by appending all the items in the given list
2. extend(self: rti.connextdds.DynamicData.TopicSeq, L: Iterable) -> None
Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.DynamicData.TopicSeq, *i: int*, *x:* rti.connextdds.DynamicData.Topic) → None

Insert an item at a given position.

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

Overloaded function.
1. pop(self: rti.connextdds.DynamicData.TopicSeq) -> rti.connextdds.DynamicData.Topic
Remove and return the last item
2. pop(self: rti.connextdds.DynamicData.TopicSeq, i: int) -> rti.connextdds.DynamicData.Topic
Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.DynamicData.TopicSeq, *x:* rti.connextdds.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`

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**static from_cdr_buffer** (*sample:* rti.connextdds.DynamicData, *buffer:* rti.connextdds.CharSeq) → None

Deserialize a sample from a CDR buffer.

**static initialize_sample** (*arg0:* rti.connextdds.DynamicData) → None

**static register_type** (*participant:* rti.connextdds.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`

**__annotations__ = {}**

**__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[*rti.connextdds.DynamicData.LoanedSample*]

**__module__ = 'rti.connextdds'**

**class WriterContentFilter**

Bases: *ContentFilter*

**__annotations__ = {}**

**__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.connextdds.DynamicData.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.DynamicData.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*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.DynamicData.WriterContentFilterHelper) → None

**__module__ = 'rti.connextdds'**

**add_cookie**(*self:* rti.connextdds.DynamicData.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.DynamicData.WriterContentFilterHelper, *writer_filter_data: Optional[object]*, *sample:* rti.connextdds.DynamicData, *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.

**__annotations__ = {}**

**__contains__**(**args*, ***kwargs*)

Overloaded function.

1. __contains__(self: rti.connextdds.DynamicData, arg0: str) -> bool
2. __contains__(self: rti.connextdds.DynamicData, arg0: object) -> bool

**__eq__**(**args*, ***kwargs*)

Overloaded function.

1. __eq__(self: rti.connextdds.DynamicData, arg0: rti.connextdds.DynamicData) -> bool

Test for equality.

2. __eq__(self: rti.connextdds.DynamicData, arg0: dict) -> bool

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

Overloaded function.

1. __getitem__(self: rti.connextdds.DynamicData, arg0: str) -> object
2. __getitem__(self: rti.connextdds.DynamicData, arg0: int) -> object
3. __getitem__(self: rti.connextdds.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.connextdds.DynamicData, *value: object*) → None

Append a value to a sequence.

**clear_all_members** (*self:* rti.connextdds.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.connextdds.DynamicData, name: str) -> None

Clear the contents of a single data member of this object.

2. clear_member(self: rti.connextdds.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.connextdds.DynamicData, name: str) -> None

Clear the contents of a single optional data member of this object.

2. clear_optional_member(self: rti.connextdds.DynamicData, index: int) -> None

Clear the contents of a single optional data member of this object.

**count**(*self:* rti.connextdds.DynamicData, *arg0: object*) → int

**property discriminator**

Used to get and set the discriminator value of a union. (valid for UnionType only)

**property discriminator_value**

Deprecated: use the discriminator property instead.

**extend**(*self:* rti.connextdds.DynamicData, *values: Iterable*) → None

Extend a sequence.

**fields**(*self:* rti.connextdds.DynamicData) → *rti.connextdds.DynamicData.FieldsView*

**from_cdr_buffer**(**args*, ***kwargs*)

Overloaded function.

1. from_cdr_buffer(self: rti.connextdds.DynamicData, buffer: rti.connextdds.Uint8Seq) -> rti.connextdds.DynamicData

Populates a DynamicData sample by deserializing a CDR buffer.

2. from_cdr_buffer(self: rti.connextdds.DynamicData, signed_buffer: rti.connextdds.CharSeq) -> rti.connextdds.DynamicData

Populates a DynamicData sample by deserializing a CDR buffer.

**from_json**(*self:* rti.connextdds.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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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, index: 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 a 8-bit signed int value by field name.

2. get_int8(self: rti.connextdds.DynamicData, index: int) -> int

Get a 8-bit signed int value by field name.

**get_int8_values** (**args*, ***kwargs*)

Overloaded function.

1. get_int8_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Int8Seq

Get multiple 8-bit signed int values by field name.

2. get_int8_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Int8Seq

Get multiple 8-bit signed int values by field name.

**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, index: 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, index: 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, index: 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 opaque byte value by field name.

2. get_octet(self: rti.connextdds.DynamicData, index: int) -> int

Get a opaque byte 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 opaque byte values by field name.

2. get_octet_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint8Seq

Get multiple opaque byte 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, index: 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.connextdds.DynamicData, name: str) -> rti.connextdds.Int16Seq

Get multiple 16-bit signed int values by field name.

2. get_short_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Int16Seq

Get multiple 16-bit signed int values by field name.

**get_string**(**args*, ***kwargs*)

Overloaded function.

1. get_string(self: rti.connextdds.DynamicData, name: str) -> str

Get a string value by field name.

2. get_string(self: rti.connextdds.DynamicData, index: int) -> str

Get a string value by field name.

**get_uint**(**args*, ***kwargs*)

Overloaded function.

1. get_uint(self: rti.connextdds.DynamicData, name: str) -> int

Get a 32-bit unsigned int value by field name.

2. get_uint(self: rti.connextdds.DynamicData, index: int) -> int

Get a 32-bit unsigned int value by field name.

**get_uint16**(**args*, ***kwargs*)

Overloaded function.

1. get_uint16(self: rti.connextdds.DynamicData, name: str) -> int

Get a 16-bit unsigned int value by field name.

2. get_uint16(self: rti.connextdds.DynamicData, index: 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.connextdds.DynamicData, name: str) -> rti.connextdds.Uint16Seq

Get multiple 16-bit unsigned int values by field name.

2. get_uint16_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.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, index: 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, index: 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, index: 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.connextdds.DynamicData, name: str) -> rti.connextdds.Uint8Seq

Get multiple 8-bit unsigned int values by field name.

2. get_uint8_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint8Seq

Get multiple 8-bit unsigned int values by field name.

**get_uint_values**(**args*, ***kwargs*)

Overloaded function.

1. get_uint_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Uint32Seq

Get multiple 32-bit unsigned int values by field name.

2. get_uint_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint32Seq

Get multiple 32-bit unsigned int values by field name.

**get_ulong**(**args*, ***kwargs*)

Overloaded function.

1. get_ulong(self: rti.connextdds.DynamicData, name: str) -> int

Get a 32-bit unsigned int value by field name.

2. get_ulong(self: rti.connextdds.DynamicData, index: 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.connextdds.DynamicData, name: str) -> rti.connextdds.Uint32Seq

Get multiple 32-bit unsigned int values by field name.

2. get_ulong_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint32Seq

Get multiple 32-bit unsigned int values by field name.

**get_ulonglong**(**args*, ***kwargs*)

Overloaded function.

1. get_ulonglong(self: rti.connextdds.DynamicData, name: str) -> int

Get a 64-bit unsigned int value by field name.

2. get_ulonglong(self: rti.connextdds.DynamicData, index: 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.connextdds.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, index: 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.connextdds.DynamicData, name: str, value: str) -> None

Set a character value by name.

2. set_char(self: rti.connextdds.DynamicData, index: int, value: str) -> None

Set a character value by index.

3. set_char(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a char value by field name using an int.

4. set_char(self: rti.connextdds.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.connextdds.DynamicData, name: str, values: object) -> None

Get multiple character values by field name.

2. set_char_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple character values by field name.

**set_complex**(**args*, ***kwargs*)

Overloaded function.

1. set_complex(self: rti.connextdds.DynamicData, name: str, value: rti.connextdds.DynamicData) -> None

Set a complex data type value by name.

2. set_complex(self: rti.connextdds.DynamicData, index: int, value: rti.connextdds.DynamicData) -> None

Set a complex data type value by index.

**set_complex_values**(**args*, ***kwargs*)

Overloaded function.

1. set_complex_values(self: rti.connextdds.DynamicData, name: str, values: Iterable) -> None

Set a list of complex values by field name.

2. set_complex_values(self: rti.connextdds.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.connextdds.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 a 8-bit signed int value by name.

2. set_int8(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 8-bit signed int value by index.

**set_int8_values**(**args*, ***kwargs*)

Overloaded function.

1. set_int8_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 8-bit signed int values by field name.

2. set_int8_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple 8-bit signed int values by field name.

**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.LongDouble) -> None

Set a 128-bit floating point value by name.

2. set_longdouble(self: rti.connextdds.DynamicData, index: int, value: rti.connextdds.LongDouble) -> 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 opaque byte value by name.

2. set_octet(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a opaque byte 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 opaque byte values by field name.

2. set_octet_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple opaque byte 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.connextdds.DynamicData, *format:* rti.connextdds.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.connextdds.DynamicData, *arg0: dict*) → None

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__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__**(*self: object*) → 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>**

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__**(*self:* rti.connextdds.DynamicDataEncapsulationKind) → *rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__**(*self:* rti.connextdds.DynamicDataEncapsulationKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**DynamicDataInfo**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.DynamicDataInfo, *arg0:* rti.connextdds.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.connextdds.**DynamicDataMemberInfo**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Create a DynamicDataMemberInfo object.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.DynamicDataMemberInfo, *arg0:* rti.connextdds.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.connextdds.**DynamicDataProperty**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Construct with default settings.

2. __init__(self: rti.connextdds.DynamicDataProperty, initial_buffer_size: int, max_buffer_size: int, max_buffer_size_increment: int, check_buffer_size: bool) -> None

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.DynamicDataProperty, *arg0:* rti.connextdds.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.connextdds.**DynamicDataSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.DynamicDataSeq, *x:* rti.connextdds.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[*rti.connextdds.DynamicData*]

**__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.connextdds.DynamicDataSeq, *arg0: int*) → *rti.connextdds.DynamicDataSeq*

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

Overloaded function.

1. __setitem__(self: rti.connextdds.DynamicDataSeq, arg0: int, arg1: rti.connextdds.DynamicData) -> None

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

Assign list elements using a slice object

**append** (*self:* rti.connextdds.DynamicDataSeq, *x:* rti.connextdds.DynamicData) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.DynamicDataSeq, *x:* rti.connextdds.DynamicData) → int

Return the number of times x appears in the list

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.DynamicDataSeq, *i: int*, *x:* rti.connextdds.DynamicData) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.DynamicDataSeq) -> rti.connextdds.DynamicData

Remove and return the last item

2. pop(self: rti.connextdds.DynamicDataSeq, i: int) -> rti.connextdds.DynamicData

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.DynamicDataSeq, *x:* rti.connextdds.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.connextdds.**DynamicDataTimestampedSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.DynamicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.DynamicData, rti.connextdds.Time*]*) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.DynamicDataTimestampedSeq, arg0: int) -> tuple[rti.connextdds.DynamicData, rti.connextdds.Time]

**__hash__ = None**

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

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

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

Overloaded function.

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.DynamicDataTimestampedSeq) → Iterator[tuple[*rti.connextdds.DynamicData*, *rti.connextdds.Time*]]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.DynamicDataTimestampedSeq, arg0: int, arg1: tuple[rti.connextdds.DynamicData, rti.connextdds.Time]) -> None
2. __setitem__(self: rti.connextdds.DynamicDataTimestampedSeq, arg0: slice, arg1: rti.connextdds.DynamicDataTimestampedSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.DynamicDataTimestampedSeq, *x: tuple[*rti.connextdds.DynamicData, rti.connextdds.Time*]*) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.DynamicDataTimestampedSeq, *x: tuple[*rti.connextdds.DynamicData, rti.connextdds.Time*]*) → int

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.DynamicDataTimestampedSeq, *i: int*, *x: tuple[*rti.connextdds.DynamicData, rti.connextdds.Time*]*) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.DynamicDataTimestampedSeq) -> tuple[rti.connextdds.DynamicData, rti.connextdds.Time]

Remove and return the last item

2. pop(self: rti.connextdds.DynamicDataTimestampedSeq, i: int) -> tuple[rti.connextdds.DynamicData, rti.connextdds.Time]

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.DynamicDataTimestampedSeq, *x: tuple[*rti.connextdds.DynamicData, 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.**DynamicDataTypeSerializationProperty**

Bases: `pybind11_object`

**DEFAULT = <rti.connextdds.DynamicDataTypeSerializationProperty object>**

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Construct with default settings.

2. __init__(self: rti.connextdds.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.connextdds'**

**__ne__**(*self:* rti.connextdds.DynamicDataTypeSerializationProperty, *arg0:* rti.connextdds.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.connextdds.**DynamicType**

Bases: pybind11_object

**__annotations__ = {}**

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

Compare DynamicType objects for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Compare DynamicType objects for inequality.

**__str__**(*self:* rti.connextdds.DynamicType) → str

DynamicData value to string.

**is_aggregation_type**(*self:* rti.connextdds.DynamicType) → bool

Determines if this DynamicType is an aggregation type.

**is_assignable_from**(*self:* rti.connextdds.DynamicType, *from_type:* rti.connextdds.DynamicType) → bool

Checks if this type is assignable from another type.

**Args:**
from_type (DynamicType): The type to check assignability from.

**Returns:**
bool: True if this type is assignable from the other type, False otherwise.

**is_collection_type** (*self:* rti.connextdds.DynamicType) → bool
Determins if this DynamicType is a CollectionType.

**is_constructed_type** (*self:* rti.connextdds.DynamicType) → bool
Determines if this DynamicType is a constructed type.

**is_keyed** (*self:* rti.connextdds.DynamicType) → bool
Determines if this DynamicType has a key.

**is_primitive_type** (*self:* rti.connextdds.DynamicType) → bool
Determines if this DynamicType is a PrimitiveType

**property kind**
Get the type kind.

**property name**
Gets the name.

**print_idl** (*self:* rti.connextdds.DynamicType, *index: int = 0*) → None
Prints the IDL representation of this type to the standard output.

**to_string** (*self:* rti.connextdds.DynamicType, *format:* rti.connextdds.DynamicTypePrintFormatProperty = *DynamicTypePrintFormatProperty()*) → str
Convert DynamicType to string with print format.

**class** rti.connextdds.**DynamicTypePrintFormatProperty**
Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.DynamicTypePrintFormatProperty, *arg0:* rti.connextdds.DynamicTypePrintFormatProperty) → bool
Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.DynamicTypePrintFormatProperty, *indent: int = 0*, *print_ordinals: bool = False*) → None
Construct with default settings.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.DynamicTypePrintFormatProperty, *arg0:* rti.connextdds.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.connextdds.**EndpointGroup**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.EndpointGroup, *role_name: str*, *quorum_count: int*) → None

Create an EndpointGroup with the provided parameters.

**__module__ = 'rti.connextdds'**

**property quorum_count**

Get/set the EndpointGroup's quorum count.

**property role_name**

Get/set the EndpointGroup's role name.

**class** rti.connextdds.**EndpointGroupSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.EndpointGroupSeq, *x:* rti.connextdds.EndpointGroup) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

   Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.EndpointGroupSeq, arg0: int) -> rti.connextdds.EndpointGroup

**__hash__ = None**

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

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

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

Overloaded function.

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

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

   Copy constructor

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

**__iter__**(*self:* rti.connextdds.EndpointGroupSeq) → Iterator[*rti.connextdds.EndpointGroup*]

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

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.EndpointGroupSeq, arg0: int, arg1: rti.connextdds.EndpointGroup) -> None

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

   Assign list elements using a slice object

**append**(*self:* rti.connextdds.EndpointGroupSeq, *x:* rti.connextdds.EndpointGroup) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.EndpointGroupSeq, *x:* rti.connextdds.EndpointGroup) → int

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.EndpointGroupSeq, *i: int*, *x:* rti.connextdds.EndpointGroup) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.EndpointGroupSeq) -> rti.connextdds.EndpointGroup

Remove and return the last item

2. pop(self: rti.connextdds.EndpointGroupSeq, i: int) -> rti.connextdds.EndpointGroup

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.EndpointGroupSeq, *x:* rti.connextdds.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.connextdds.**EndpointGroupVector**

Bases: `pybind11_object`

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

**__annotations__ = {}**

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

Compare EndpointGroupVectors for equality.

**__getitem__** (*self:* rti.connextdds.EndpointGroupVector, *arg0: int*) → *rti.connextdds.EndpointGroup*

Get the value at the specified index.

**__hash__ = None**

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

Overloaded function.

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

Create an empty EndpointGroupVector

2. __init__(self: rti.connextdds.EndpointGroupVector, size: int) -> None

Create a EndpointGroupVector with a preallocated size.

3. __init__(self: rti.connextdds.EndpointGroupVector, vector: rti.connextdds.EndpointGroupVector) -> None

Create a copy from another EndpointGroupVector.

4. __init__(self: rti.connextdds.EndpointGroupVector, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.EndpointGroupVector, list: rti.connextdds.EndpointGroupSeq) -> None

Create a EndpointGroupVector from a list of values.

**__iter__** (*self:* rti.connextdds.EndpointGroupVector) → Iterator[*rti.connextdds.EndpointGroup*]

Iterate over the contents of the vector.

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

Get the length of the EndpointGroupVector.

**__module__ = 'rti.connextdds'**

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

Compare EndpointGroupVectors for inequality.

**__setitem__** (*self:* rti.connextdds.EndpointGroupVector, *arg0: int*, *arg1:* rti.connextdds.EndpointGroup) → None

Set the value at the specified index.

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

Resize EndpointGroupVector to 0.

**resize** (*self:* rti.connextdds.EndpointGroupVector, *size: int*) → None

Resize EndpointGroupVector.

**class** rti.connextdds.**Entity**

Bases: *IEntity*

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**EntityFactory**

Bases: pybind11_object

**__annotations__ = {}**

**__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.**EntityLock**

Bases: pybind11_object

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.EntityLock) → *rti.connextdds.EntityLock*

**__exit__**(*self:* rti.connextdds.EntityLock, *arg0: object*, *arg1: object*, *arg2: object*) → None

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

**__module__ = 'rti.connextdds'**

**unlock**(*self:* rti.connextdds.EntityLock) → None

**class** rti.connextdds.**EntityName**

Bases: pybind11_object

**__annotations__ = {}**

**__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.connextdds'**

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

Test for inequality.

**property name**

Sets the entity name.

**property role_name**

Sets the role name.

**class** rti.connextdds.**EntitySeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.EntitySeq, *x:* rti.connextdds.Entity) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.EntitySeq, arg0: int) -> rti.connextdds.Entity

**__hash__ = None**

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

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

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

Overloaded function.

1. __init__(self: rti.connextdds.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[*rti.connextdds.Entity*]

**__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`

**__annotations__ = {}**

**__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.connextdds.EnumMemberSeq, *arg0:* rti.connextdds.EnumMemberSeq) → *rti.connextdds.EnumMemberSeq*

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.EnumMemberSeq, *x:* rti.connextdds.EnumMember) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.EnumMemberSeq, arg0: int) -> rti.connextdds.EnumMember

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.EnumMemberSeq) → Iterator[*rti.connextdds.EnumMember*]

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

**__module__ = 'rti.connextdds'**

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

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

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.EnumMemberSeq, arg0: int, arg1: rti.connextdds.EnumMember) -> None

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

Assign list elements using a slice object

**append** (*self:* rti.connextdds.EnumMemberSeq, *x:* rti.connextdds.EnumMember) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.EnumMemberSeq, *x:* rti.connextdds.EnumMember) → int

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.EnumMemberSeq, *i: int*, *x:* rti.connextdds.EnumMember) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.EnumMemberSeq) -> rti.connextdds.EnumMember

Remove and return the last item

2. pop(self: rti.connextdds.EnumMemberSeq, i: int) -> rti.connextdds.EnumMember

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.EnumMemberSeq, *x:* rti.connextdds.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.connextdds.**EnumType**

Bases: *ACTEnumMember*

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.EnumType, name: str) -> None

Creates an empty EnumType.

2. __init__(self: rti.connextdds.EnumType, name: str, members: rti.connextdds.EnumMemberSeq) -> None

Creates a enum with the members in the list.

3. __init__(self: rti.connextdds.EnumType, type: rti.connextdds.DynamicType) -> None

Cast a DynamicType to an EnumType.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**add_member** (*self:* rti.connextdds.EnumType, *member:* rti.connextdds.EnumMember) → *rti.connextdds.EnumType*

Adds a member at the end.

**add_members** (*self:* rti.connextdds.EnumType, *members:* rti.connextdds.EnumMemberSeq) → *rti.connextdds.EnumType*

Adds a member at the end.

**property extensibility_kind**

Enum's extensibility kind.

**find_member_by_ordinal** (*self:* rti.connextdds.EnumType, *ordinal: int*) → int

Gets the index of the member with this ordinal value.

**exception** rti.connextdds.**Error**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**Event**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Event, *arg0:* rti.connextdds.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

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**property change**

The incremental count.

**property total**

The total count.

**class** rti.connextdds.**EventCount64**

Bases: pybind11_object

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**property change**

The incremental count.

**property total**

The total count.

**exception** rti.connextdds.**Exception**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**__weakref__**

list of weak references to the object (if defined)

**class** rti.connextdds.**ExpressionProperty**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

1. __init__(self: rti.connextdds.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 Connext 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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**FINAL = <ExtensibilityKind.FINAL: 0>**

**MUTABLE = <ExtensibilityKind.MUTABLE: 2>**

**__annotations__ = {}**

**__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.ExtensibilityKind) -> 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.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.ExtensibilityKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**Filter**

Bases: pybind11_object

**SQL_FILTER_NAME = 'DDSSQL'**

**STRINGMATCH_FILTER_NAME = 'DDSSTRINGMATCH'**

**__annotations__ = {}**

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

Overloaded function.

1. __init__(self: rti.connextdds.Filter, expression: str) -> None

Create a Filter with the specified expression.

2. __init__(self: rti.connextdds.Filter, expression: str, parameters: rti.connextdds.StringSeq) -> None

Create a Filter with the specified expression and parameters.

**__iter__** (*self:* rti.connextdds.Filter) → Iterator[str]

**__module__ = 'rti.connextdds'**

**add_parameters** (*self:* rti.connextdds.Filter, *parameter: str*) → *rti.connextdds.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.connextdds.Filter, *arg0:* rti.connextdds.StringSeq) → None

Set the parameters for the Filter.

**class** rti.connextdds.**FilterSampleInfo**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.FilterSampleInfo, *arg0:* rti.connextdds.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_reader_guid**

The related reader GUID associated with the sample.

**property related_sample_identity**

The related sample identity associated with the sample.

**property related_source_guid**

The related source GUID associated with the sample.

**property sample_identity**

The identity of the sample.

**property source_guid**

The source GUID associated with the sample.

**class** rti.connextdds.**Float128Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Float128Seq, *arg0:* rti.connextdds.Float128Seq) → *rti.connextdds.Float128Seq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Float128Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Float128Seq, *x:* rti.connextdds.LongDouble) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: rti.connextdds.Float128Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Float128Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.Float128Seq, *arg0:* rti.connextdds.Float128Seq) → bool

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

Overloaded function.

1. __getitem__(self: rti.connextdds.Float128Seq, s: slice) -> rti.connextdds.Float128Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Float128Seq, arg0: int) -> rti.connextdds.LongDouble

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.Float128Seq, *arg0:* rti.connextdds.Float128Seq) → *rti.connextdds.Float128Seq*

**__imul__** (*self:* rti.connextdds.Float128Seq, *arg0: int*) → *rti.connextdds.Float128Seq*

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

Overloaded function.

1. __init__(self: rti.connextdds.Float128Seq) -> None

2. __init__(self: rti.connextdds.Float128Seq, arg0: rti.connextdds.Float128Seq) -> None

Copy constructor

3. __init__(self: rti.connextdds.Float128Seq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.Float128Seq) → Iterator[*rti.connextdds.LongDouble*]

**__len__** (*self:* rti.connextdds.Float128Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Float128Seq, *arg0: int*) → *rti.connextdds.Float128Seq*

**__ne__** (*self:* rti.connextdds.Float128Seq, *arg0:* rti.connextdds.Float128Seq) → bool

**__repr__** (*self:* rti.connextdds.Float128Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Float128Seq, *arg0: int*) → *rti.connextdds.Float128Seq*

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

Overloaded function.

1. __setitem__(self: rti.connextdds.Float128Seq, arg0: int, arg1: rti.connextdds.LongDouble) -> None

2. __setitem__(self: rti.connextdds.Float128Seq, arg0: slice, arg1: rti.connextdds.Float128Seq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.Float128Seq, *x:* rti.connextdds.LongDouble) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.Float128Seq) → None

Clear the contents

**count**(*self:* rti.connextdds.Float128Seq, *x:* rti.connextdds.LongDouble) → int

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

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

Overloaded function.

1. extend(self: rti.connextdds.Float128Seq, L: rti.connextdds.Float128Seq) -> None

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

2. extend(self: rti.connextdds.Float128Seq, L: Iterable) -> None

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

**insert**(*self:* rti.connextdds.Float128Seq, *i: int*, *x:* rti.connextdds.LongDouble) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.Float128Seq) -> rti.connextdds.LongDouble

Remove and return the last item

2. pop(self: rti.connextdds.Float128Seq, i: int) -> rti.connextdds.LongDouble

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.Float128Seq, *x:* rti.connextdds.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.connextdds.**Float128Type**

Bases: *DynamicType*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Float128Type, *arg0:* rti.connextdds.Float128Type) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.Float128Type) → None

Get the singleton for Float128Type

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Float128Type, *arg0:* rti.connextdds.Float128Type) → bool

Test for inequality.

**class** rti.connextdds.**Float32Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Float32Seq, *arg0:* rti.connextdds.Float32Seq) → *rti.connextdds.Float32Seq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Float32Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Float32Seq, *x: float*) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: rti.connextdds.Float32Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Float32Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.Float32Seq, *arg0:* rti.connextdds.Float32Seq) → bool

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

Overloaded function.

1. __getitem__(self: rti.connextdds.Float32Seq, s: slice) -> rti.connextdds.Float32Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Float32Seq, arg0: int) -> float

**__getstate__**(*self:* rti.connextdds.Float32Seq) → bytes

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.Float32Seq, *arg0:* rti.connextdds.Float32Seq) → *rti.connextdds.Float32Seq*

**__imul__**(*self:* rti.connextdds.Float32Seq, *arg0: int*) → *rti.connextdds.Float32Seq*

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

Overloaded function.

1. __init__(self: rti.connextdds.Float32Seq, arg0: Buffer) -> None
2. __init__(self: rti.connextdds.Float32Seq) -> None
3. __init__(self: rti.connextdds.Float32Seq, arg0: rti.connextdds.Float32Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Float32Seq, arg0: Iterable) -> None
5. __init__(self: rti.connextdds.Float32Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Float32Seq) → Iterator[float]

**__len__** (*self:* rti.connextdds.Float32Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Float32Seq, *arg0: int*) → *rti.connextdds.Float32Seq*

**__ne__** (*self:* rti.connextdds.Float32Seq, *arg0:* rti.connextdds.Float32Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Float32Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Float32Seq, *arg0: int*) → *rti.connextdds.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*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Float32Type, *arg0:* rti.connextdds.Float32Type) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.Float32Type) → None

Get the singleton for Float32Type

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Float32Type, *arg0:* rti.connextdds.Float32Type) → bool

Test for inequality.

**class** rti.connextdds.**Float64Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Float64Seq, *arg0:* rti.connextdds.Float64Seq) → *rti.connextdds.Float64Seq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Float64Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Float64Seq, *x: float*) → bool

Return true the container contains `x`

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

Overloaded function.

1. __delitem__(self: rti.connextdds.Float64Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Float64Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.Float64Seq, *arg0:* rti.connextdds.Float64Seq) → bool

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

Overloaded function.

1. __getitem__(self: rti.connextdds.Float64Seq, s: slice) -> rti.connextdds.Float64Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Float64Seq, arg0: int) -> float

**__getstate__**(*self:* rti.connextdds.Float64Seq) → bytes

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.Float64Seq, *arg0:* rti.connextdds.Float64Seq) → *rti.connextdds.Float64Seq*

**__imul__**(*self:* rti.connextdds.Float64Seq, *arg0: int*) → *rti.connextdds.Float64Seq*

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

Overloaded function.

1. __init__(self: rti.connextdds.Float64Seq, arg0: Buffer) -> None

2. __init__(self: rti.connextdds.Float64Seq) -> None

3. __init__(self: rti.connextdds.Float64Seq, arg0: rti.connextdds.Float64Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Float64Seq, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.Float64Seq, arg0: int) -> None

**__iter__**(*self:* rti.connextdds.Float64Seq) → Iterator[float]

**__len__**(*self:* rti.connextdds.Float64Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.Float64Seq, *arg0: int*) → *rti.connextdds.Float64Seq*

**__ne__**(*self:* rti.connextdds.Float64Seq, *arg0:* rti.connextdds.Float64Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__**(*self:* rti.connextdds.Float64Seq) → str

Return the canonical string representation of this list.

**__rmul__**(*self:* rti.connextdds.Float64Seq, *arg0: int*) → *rti.connextdds.Float64Seq*

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

Overloaded function.

1. __setitem__(self: rti.connextdds.Float64Seq, arg0: int, arg1: float) -> None
2. __setitem__(self: rti.connextdds.Float64Seq, arg0: slice, arg1: rti.connextdds.Float64Seq) -> None

Assign list elements using a slice object

**__setstate__**(*self:* rti.connextdds.Float64Seq, *arg0: bytes*) → None

**append**(*self:* rti.connextdds.Float64Seq, *x: float*) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.Float64Seq) → None

Clear the contents

**count**(*self:* rti.connextdds.Float64Seq, *x: float*) → int

Return the number of times x appears in the list

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

Overloaded function.

1. extend(self: rti.connextdds.Float64Seq, L: rti.connextdds.Float64Seq) -> None

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

2. extend(self: rti.connextdds.Float64Seq, L: Iterable) -> None

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

**insert**(*self:* rti.connextdds.Float64Seq, *i: int*, *x: float*) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.Float64Seq) -> float

Remove and return the last item

2. pop(self: rti.connextdds.Float64Seq, i: int) -> float

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.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.connextdds.Float64Seq, *arg0: int*) → None
resizes the vector to the given size

**class** rti.connextdds.**Float64Type**
Bases: *DynamicType*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Float64Type, *arg0:* rti.connextdds.Float64Type) → bool
Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.Float64Type) → None
Get the singleton for Float64Type

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Float64Type, *arg0:* rti.connextdds.Float64Type) → bool
Test for inequality.

rti.connextdds.**FloatSeq**
alias of *Float32Seq*

rti.connextdds.**FloatType**
alias of *Float32Type*

**class** rti.connextdds.**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'**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.FlowController, *arg0:* rti.connextdds.FlowController) → bool
Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.FlowController, *participant:* rti.connextdds.DomainParticipant, *name: str*, *token_bucket:* rti.connextdds.FlowControllerProperty = *FlowControllerProperty()*) → None
Creates a FlowController with specific properties.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**close** (*self:* rti.connextdds.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.connextdds.FlowController) → None

Disables the automatic destruction of this object.

**trigger_flow** (*self:* rti.connextdds.FlowController) → None

Provides an external way to trigger a FlowController.

**class** rti.connextdds.**FlowControllerProperty**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Creates a FlowControllerProperty with earliest-deadline-first scheduling policy and default token-bucket configuration.

2. __init__(self: rti.connextdds.FlowControllerProperty, scheduling_policy: rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy, token_bucket: rti.connextdds.FlowControllerTokenBucketProperty = FlowControllerTokenBucketProperty()) -> None

Creates a FlowControllerProperty with the specified scheduling policy and token-bucket configuration.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**property scheduling_policy**

The scheduling policy.

**property token_bucket**

The token-bucket configuration

**class** rti.connextdds.**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>**

**__annotations__ = {}**

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

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:*
rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy)
→ int

**__init__**(*self:*
rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy,
*value: int*) → None

**__int__**(*self:*
rti.connextdds.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.connextdds'**

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

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**HIGHEST_PRIORITY_FIRST = <FlowControllerSchedulingPolicy.HIGHEST_PRIORITY_FIRST: 2>**

**ROUND_ROBIN = <FlowControllerSchedulingPolicy.ROUND_ROBIN: 0>**

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__** (*self:* rti.connextdds.FlowControllerSchedulingPolicy) → *rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.FlowControllerSchedulingPolicy) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**FlowControllerTokenBucketProperty**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.FlowControllerTokenBucketProperty, *max_tokens: int = LENGTH_UNLIMITED*, *tokens_added_per_period: int = LENGTH_UNLIMITED*, *tokens_leaked_per_period: int = 0*, *period:* rti.connextdds.Duration = *Duration(1, 0)*, *bytes_per_token: int = LENGTH_UNLIMITED*) → None

Initializes the properties.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.FlowControllerTokenBucketProperty, *arg0:* rti.connextdds.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.connextdds.**GenerationCount**

Bases: pybind11_object

**__annotations__ = {}**

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

Overloaded function.

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

Create a default GenerationCount object.

2. __init__(self: rti.connextdds.GenerationCount, disposed_count: int, no_writers_count: int) -> None

Create a GenerationCount object with the provided disposed_count and no_writers count.

**__module__ = 'rti.connextdds'**

**property disposed**

Get the disposed generation count.

**property no_writers**

Get the no_writers generation count.

**class** rti.connextdds.**GroupData**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Create a GroupData instance.

2. __init__(self: rti.connextdds.GroupData, bytes: rti.connextdds.Uint8Seq) -> None

Create a GroupData instance with a sequence of bytes.

**__iter__** (*self:* rti.connextdds.GroupData) → Iterator[int]

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**property value**

The byte sequence of this GroupData.

**class** rti.connextdds.**GuardCondition**

Bases: *ICondition*

**__annotations__ = {}**

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

Compare GuardCondition objects for equality.

**__hash__ = None**

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

Overloaded function.

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

Create a GuardCondition in an untriggered state.

2. __init__(self: rti.connextdds.GuardCondition, condition: rti.connextdds.ICondition) -> None

Create a GuardCondition from a Condition.

**__module__ = 'rti.connextdds'**

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

Compare GuardCondition objects for inequality.

**reset_handler** (*self:* rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.Guid) → str

**automatic = <rti.connextdds.Guid object>**

**unknown = <rti.connextdds.Guid object>**

**class** rti.connextdds.**History**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

Overloaded function.

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

Creates a policy that keeps the last sample only.

2. __init__(self: rti.connextdds.History, kind: rti.connextdds.HistoryKind, depth: int = 1) -> None

Creates a policy with a specific history kind and optionally a history depth.

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**property depth**

The history kind.

**keep_all = <rti.connextdds.History object>**

**static keep_last** (*depth: int*) → *rti.connextdds.History*

Creates a History with HistoryKind.KEEP_LAST and the specified depth.

**property kind**

The history kind.

**class** rti.connextdds.**HistoryKind**

Bases: pybind11_object

**class HistoryKind**

Bases: pybind11_object

Members:

KEEP_LAST : [default] Keep the last depth samples.

On the publishing side, RTI Connext 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 represeting 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 represeting a dispoasal 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 Connext 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 Connext 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 Connext via the take() operation.

**KEEP_ALL = <HistoryKind.KEEP_ALL: 1>**

**KEEP_LAST = <HistoryKind.KEEP_LAST: 0>**

**__annotations__ = {}**

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

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.HistoryKind.HistoryKind) → int

**__init__**(*self:* rti.connextdds.HistoryKind.HistoryKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.HistoryKind.HistoryKind) → int

**__members__ = {'KEEP_ALL': <HistoryKind.KEEP_ALL: 1>, 'KEEP_LAST': <HistoryKind.KEEP_LAST: 0>}**

**__module__ = 'rti.connextdds'**

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

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.HistoryKind.HistoryKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**KEEP_ALL = <HistoryKind.KEEP_ALL: 1>**

**KEEP_LAST = <HistoryKind.KEEP_LAST: 0>**

**__annotations__ = {}**

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__hash__ = None**

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

Overloaded function.

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

Initializes enum to 0.

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

Copy constructor.

**__int__** (*self:* rti.connextdds.HistoryKind) → *rti.connextdds.HistoryKind.HistoryKind*

Int conversion.

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

Apply operator to underlying enumerated values.

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

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

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

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.HistoryKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**IAnyDataReader**

Bases: pybind11_object

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**close** (*self:* rti.connextdds.IAnyDataReader) → None

Close this DataReader.

**property qos**

The QoS for this AnyDataReader.

This property's getter returns a deep copy.

**retain** (*self:* rti.connextdds.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.connextdds.**IAnyDataReaderSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.IAnyDataReaderSeq, *x:* rti.connextdds.IAnyDataReader) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.IAnyDataReaderSeq, arg0: int) -> rti.connextdds.IAnyDataReader

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__**(*self:* rti.connextdds.IAnyDataReaderSeq) → Iterator[*rti.connextdds.IAnyDataReader*]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__**(*self:* rti.connextdds.IAnyDataReaderSeq) → str

Return the canonical string representation of this list.

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

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**close** (*self:* rti.connextdds.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.connextdds.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.connextdds.IAnyDataWriter, *timeout:* rti.connextdds.Duration) → None

Wait for acknowledgments from subscribers.

**class** rti.connextdds.**IAnyDataWriterSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.IAnyDataWriterSeq, *x:* rti.connextdds.IAnyDataWriter) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.IAnyDataWriterSeq, arg0: int) -> rti.connextdds.IAnyDataWriter

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__**(*self:* rti.connextdds.IAnyDataWriterSeq) → Iterator[*rti.connextdds.IAnyDataWriter*]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__**(*self:* rti.connextdds.IAnyDataWriterSeq) → str

Return the canonical string representation of this list.

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.IAnyDataWriterSeq, arg0: int, arg1: rti.connextdds.IAnyDataWriter) -> None

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

Assign list elements using a slice object

**append** (*self:* rti.connextdds.IAnyDataWriterSeq, *x:* rti.connextdds.IAnyDataWriter) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.IAnyDataWriterSeq, *x:* rti.connextdds.IAnyDataWriter) → int

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.IAnyDataWriterSeq, *i: int*, *x:* rti.connextdds.IAnyDataWriter) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.IAnyDataWriterSeq) -> rti.connextdds.IAnyDataWriter

Remove and return the last item

2. pop(self: rti.connextdds.IAnyDataWriterSeq, i: int) -> rti.connextdds.IAnyDataWriter

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.IAnyDataWriterSeq, *x:* rti.connextdds.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.connextdds.`**IAnyTopic**

Bases: `pybind11_object`

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**close** (*self:* rti.connextdds.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.connextdds.**IAnyTopicSeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.IAnyTopicSeq, *x:* rti.connextdds.IAnyTopic) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.IAnyTopicSeq, arg0: int) -> rti.connextdds.IAnyTopic

**__hash__ = None**

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

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

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

Overloaded function.

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.IAnyTopicSeq) → Iterator[*rti.connextdds.IAnyTopic*]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.IAnyTopicSeq) → str

Return the canonical string representation of this list.

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.IAnyTopicSeq, arg0: int, arg1: rti.connextdds.IAnyTopic) -> None
2. __setitem__(self: rti.connextdds.IAnyTopicSeq, arg0: slice, arg1: rti.connextdds.IAnyTopicSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.IAnyTopicSeq, *x:* rti.connextdds.IAnyTopic) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.IAnyTopicSeq, *x:* rti.connextdds.IAnyTopic) → int

Return the number of times x appears in the list

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

Overloaded function.

1. extend(self: rti.connextdds.IAnyTopicSeq, L: rti.connextdds.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

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.IConditionSeq, arg0: int) -> None

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.IConditionSeq, arg0: int) -> rti.connextdds.ICondition

**__hash__ = None**

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

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

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

Overloaded function.

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

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.IConditionSeq) → Iterator[*rti.connextdds.ICondition*]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.IConditionSeq) → str

Return the canonical string representation of this list.

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.IConditionSeq, arg0: int, arg1: rti.connextdds.ICondition) -> None

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

Assign list elements using a slice object

**append** (*self:* rti.connextdds.IConditionSeq, *x:* rti.connextdds.ICondition) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.IConditionSeq, *x:* rti.connextdds.ICondition) → int

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

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

Overloaded function.

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

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

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

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

**insert** (*self:* rti.connextdds.IConditionSeq, *i: int*, *x:* rti.connextdds.ICondition) → None

Insert an item at a given position.

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

Overloaded function.

1. pop(self: rti.connextdds.IConditionSeq) -> rti.connextdds.ICondition

Remove and return the last item

2. pop(self: rti.connextdds.IConditionSeq, i: int) -> rti.connextdds.ICondition

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.IConditionSeq, *x:* rti.connextdds.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.connextdds.**IDataReader**

Bases: *IEntity*, *IAnyDataReader*

**__annotations__ = {}**

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

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**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.

**__annotations__ = {}**

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

Test for equality.

**__hash__ = None**

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

**__module__ = 'rti.connextdds'**

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

Test for inequality.

**close** (*self:* rti.connextdds.IEntity) → None

Forces the destruction of this entity.

**property closed**

Returns a boolean indicating if this Entity is closed.

**enable** (*self:* rti.connextdds.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.

**lock** (*self:* rti.connextdds.IEntity) → *rti.connextdds.EntityLock*

Locks the entity and returns a Context Manager.

**retain** (*self:* rti.connextdds.IEntity) → None

Disables the automatic destruction of this entity.

**property status_changes**

The list of communication statuses that are triggered.

**unretain**(*self:* rti.connextdds.IEntity) → None

Decrement the retention count.

**property use_count**

Returns the internal use count value for this Entity.

**class** rti.connextdds.**IEntitySeq**

Bases: pybind11_object

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

**__annotations__ = {}**

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

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.IEntitySeq, *x:* rti.connextdds.IEntity) → bool

Return true the container contains `x`

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

Overloaded function.

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

Delete the list elements at index `i`

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

Delete list elements using a slice object

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

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

Overloaded function.

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

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.IEntitySeq, arg0: int) -> rti.connextdds.IEntity

**__hash__ = None**

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

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

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

Overloaded function.

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

Copy constructor

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

**__iter__** (*self:* rti.connextdds.IEntitySeq) → Iterator[*rti.connextdds.IEntity*]

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

**__module__ = 'rti.connextdds'**

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

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

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.IEntitySeq) → str

Return the canonical string representation of this list.

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

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

Overloaded function.

1. __setitem__(self: rti.connextdds.IEntitySeq, arg0: int, arg1: rti.connextdds.IEntity) -> None
2. __setitem__(self: rti.connextdds.IEntitySeq, arg0: slice, arg1: rti.connextdds.IEntitySeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.IEntitySeq, *x:* rti.connextdds.IEntity) → None

Add an item to the end of the list

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

Clear the contents

**count** (*self:* rti.connextdds.IEntitySeq, *x:* rti.connextdds.IEntity) → int

Return the number of times x appears in the list

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

Overloaded function.

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

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

2. extend(self: rti.connextdds.IEntitySeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.IEntitySeq, *i: int*, *x:* rti.connextdds.IEntity) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.IEntitySeq) -> rti.connextdds.IEntity

Remove and return the last item

2. pop(self: rti.connextdds.IEntitySeq, i: int) -> rti.connextdds.IEntity

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.IEntitySeq, *x:* rti.connextdds.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.connextdds.**IReadCondition**

Bases: *ICondition*

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.IReadCondition) → *rti.connextdds.IReadCondition*

**__eq__** (*self:* rti.connextdds.IReadCondition, *arg0:* rti.connextdds.IReadCondition) → bool

Test for equality.

**__exit__** (*self:* rti.connextdds.IReadCondition, *arg0: object*, *arg1: object*, *arg2: object*) → None

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.IReadCondition, *arg0:* rti.connextdds.IReadCondition) → bool

Test for inequality.

**close** (*self:* rti.connextdds.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.connextdds.**ITopicDescription**

Bases: *IEntity*

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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.connextdds.**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind) → int

**__init__**(*self:* rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind) → int

**__members__ = {'NOT_ALIVE_FIRST': <IgnoredEntityReplacementKind.NOT_ALIVE_FIRST: 1>, 'NO_REPLACEMENT': <IgnoredEntityReplacementKind.NO_REPLACEMENT: 0>}**

**__module__ = 'rti.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**NOT_ALIVE_FIRST = <IgnoredEntityReplacementKind.NOT_ALIVE_FIRST: 1>**

**NO_REPLACEMENT = <IgnoredEntityReplacementKind.NO_REPLACEMENT: 0>**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.IgnoredEntityReplacementKind, *arg0:* rti.connextdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.IgnoredEntityReplacementKind, *arg0:* rti.connextdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.IgnoredEntityReplacementKind, *arg0:* rti.connextdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.IgnoredEntityReplacementKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.IgnoredEntityReplacementKind, arg0: rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind) -> None

Copy constructor.

**__int__**(*self:* rti.connextdds.IgnoredEntityReplacementKind) → *rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind*

Int conversion.

**__le__**(*self:* rti.connextdds.IgnoredEntityReplacementKind, *arg0:* rti.connextdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

**__lt__**(*self:* rti.connextdds.IgnoredEntityReplacementKind, *arg0:* rti.connextdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.IgnoredEntityReplacementKind, *arg0:* rti.connextdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

**__str__**(*self:* rti.connextdds.IgnoredEntityReplacementKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**exception** rti.connextdds.**IllegalOperationError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**exception** rti.connextdds.**ImmutablePolicyError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**exception** rti.connextdds.**InconsistentPolicyError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**InconsistentTopicStatus**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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.connextdds.**InstanceHandle**

Bases: pybind11_object

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.InstanceHandle) → bool

**__eq__**(*self:* rti.connextdds.InstanceHandle, *arg0:* rti.connextdds.InstanceHandle) → bool

Test for equality.

**__hash__**(*self:* rti.connextdds.InstanceHandle) → int

**__init__**(*self:* rti.connextdds.InstanceHandle) → None

Create a nil InstanceHandle

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.InstanceHandle, *arg0:* rti.connextdds.InstanceHandle) → bool

Test for inequality.

**__nonzero__**(*self:* rti.connextdds.InstanceHandle) → bool

**__str__**(*self:* rti.connextdds.InstanceHandle) → str

**property is_nil**

Nil status of InstanceStatus

**static nil**() → *rti.connextdds.InstanceHandle*

Create a nil InstanceHandle.

**class** rti.connextdds.**InstanceHandleSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.InstanceHandleSeq, *arg0:* rti.connextdds.InstanceHandleSeq) → *rti.connextdds.InstanceHandleSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.InstanceHandleSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.InstanceHandleSeq, *x:* rti.connextdds.InstanceHandle) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.InstanceHandleSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.InstanceHandleSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.InstanceHandleSeq, *arg0:* rti.connextdds.InstanceHandleSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.InstanceHandleSeq, s: slice) -> rti.connextdds.InstanceHandleSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.InstanceHandleSeq, arg0: int) -> rti.connextdds.InstanceHandle

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.InstanceHandleSeq, *arg0:* rti.connextdds.InstanceHandleSeq) → *rti.connextdds.InstanceHandleSeq*

**__imul__** (*self:* rti.connextdds.InstanceHandleSeq, *arg0: int*) → *rti.connextdds.InstanceHandleSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.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[*rti.connextdds.InstanceHandle*]

**__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.connextdds.InstanceHandleSeq, *i: int*, *x:* rti.connextdds.InstanceHandle) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.InstanceHandleSeq) -> rti.connextdds.InstanceHandle

Remove and return the last item

2. pop(self: rti.connextdds.InstanceHandleSeq, i: int) -> rti.connextdds.InstanceHandle

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.InstanceHandleSeq, *x:* rti.connextdds.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.connextdds.**InstanceState**

Bases: `pybind11_object`

**ALIVE = <rti.connextdds.InstanceState object>**

**ANY = <rti.connextdds.InstanceState object>**

**NOT_ALIVE_DISPOSED = <rti.connextdds.InstanceState object>**

**NOT_ALIVE_MASK = <rti.connextdds.InstanceState object>**

**NOT_ALIVE_NO_WRITERS = <rti.connextdds.InstanceState object>**

**__and__**(*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → *rti.connextdds.InstanceState*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.InstanceState) → *rti.connextdds.InstanceState*

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → bool

**__eq__**(*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.InstanceState, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → *rti.connextdds.InstanceState*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.InstanceState, *arg0: int*) → *rti.connextdds.InstanceState*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.InstanceState) -> None

Create an InstanceState with no bits set.

2. __init__(self: rti.connextdds.InstanceState, value: int) -> None

Creates a mask from the bits in an integer.

**__int__**(*self:* rti.connextdds.InstanceState) → int

Convert mask to int.

**__ior__**(*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → *rti.connextdds.InstanceState*

Set mask to logical OR with another mask.

**__irshift__**(*self:* rti.connextdds.InstanceState, *arg0: int*) → *rti.connextdds.InstanceState*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → *rti.connextdds.InstanceState*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.InstanceState, *arg0: int*) → *rti.connextdds.InstanceState*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → *rti.connextdds.InstanceState*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.InstanceState, *arg0: int*) → *rti.connextdds.InstanceState*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.InstanceState, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.InstanceState) → str

**__xor__** (*self:* rti.connextdds.InstanceState, *arg0:* rti.connextdds.InstanceState) → *rti.connextdds.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>**

**__annotations__ = {}**

**__eq__** (*self: object*, *other: object*) → bool

**__getstate__** (*self: object*) → int

**__hash__** (*self: object*) → int

**__index__** (*self:* rti.connextdds.InstanceStateConsistencyKind) → int

**__init__** (*self:* rti.connextdds.InstanceStateConsistencyKind, *value: int*) → None

**__int__** (*self:* rti.connextdds.InstanceStateConsistencyKind) → int

**__members__ = {'NONE': <InstanceStateConsistencyKind.NONE: 0>, 'RECOVER_STATE': <InstanceStateConsistencyKind.RECOVER_STATE: 1>}**

**__module__ = 'rti.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.InstanceStateConsistencyKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**class** rti.connextdds.**Int16Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Int16Seq, *arg0:* rti.connextdds.Int16Seq) → *rti.connextdds.Int16Seq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Int16Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Int16Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Int16Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Int16Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.Int16Seq, *arg0:* rti.connextdds.Int16Seq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Int16Seq, s: slice) -> rti.connextdds.Int16Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Int16Seq, arg0: int) -> int

**__getstate__**(*self:* rti.connextdds.Int16Seq) → bytes

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.Int16Seq, *arg0:* rti.connextdds.Int16Seq) → *rti.connextdds.Int16Seq*

**__imul__** (*self:* rti.connextdds.Int16Seq, *arg0: int*) → *rti.connextdds.Int16Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Int16Seq, arg0: Buffer) -> None
2. __init__(self: rti.connextdds.Int16Seq) -> None
3. __init__(self: rti.connextdds.Int16Seq, arg0: rti.connextdds.Int16Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Int16Seq, arg0: Iterable) -> None
5. __init__(self: rti.connextdds.Int16Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Int16Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Int16Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Int16Seq, *arg0: int*) → *rti.connextdds.Int16Seq*

**__ne__** (*self:* rti.connextdds.Int16Seq, *arg0:* rti.connextdds.Int16Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Int16Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Int16Seq, *arg0: int*) → *rti.connextdds.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*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Int16Type, *arg0:* rti.connextdds.Int16Type) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.Int16Type) → None

Get the singleton for Int16Type

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Int16Type, *arg0:* rti.connextdds.Int16Type) → bool

Test for inequality.

**class** rti.connextdds.**Int32Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Int32Seq, *arg0:* rti.connextdds.Int32Seq) → *rti.connextdds.Int32Seq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Int32Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Int32Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Int32Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Int32Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.Int32Seq, *arg0:* rti.connextdds.Int32Seq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Int32Seq, s: slice) -> rti.connextdds.Int32Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Int32Seq, arg0: int) -> int

**__getstate__**(*self:* rti.connextdds.Int32Seq) → bytes

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.Int32Seq, *arg0:* rti.connextdds.Int32Seq) → *rti.connextdds.Int32Seq*

**__imul__**(*self:* rti.connextdds.Int32Seq, *arg0: int*) → *rti.connextdds.Int32Seq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Int32Seq, arg0: Buffer) -> None

2. __init__(self: rti.connextdds.Int32Seq) -> None

3. __init__(self: rti.connextdds.Int32Seq, arg0: rti.connextdds.Int32Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Int32Seq, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.Int32Seq, arg0: int) -> None

**__iter__**(*self:* rti.connextdds.Int32Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Int32Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Int32Seq, *arg0: int*) → *rti.connextdds.Int32Seq*

**__ne__** (*self:* rti.connextdds.Int32Seq, *arg0:* rti.connextdds.Int32Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Int32Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Int32Seq, *arg0: int*) → *rti.connextdds.Int32Seq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Int32Seq, arg0: int, arg1: int) -> None
2. __setitem__(self: rti.connextdds.Int32Seq, arg0: slice, arg1: rti.connextdds.Int32Seq) -> None

Assign list elements using a slice object

**__setstate__** (*self:* rti.connextdds.Int32Seq, *arg0: bytes*) → None

**append** (*self:* rti.connextdds.Int32Seq, *x: int*) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.Int32Seq) → None

Clear the contents

**count** (*self:* rti.connextdds.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*

**__annotations__ = {}**

**__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.

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Int32Vector, *arg0:* rti.connextdds.Int32Vector) → bool

Compare Int32Vectors for equality.

**__getitem__** (*self:* rti.connextdds.Int32Vector, *arg0: int*) → int

Get the value at the specified index.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Int32Vector, buffer: Buffer) -> None

Create a Int32Vector from another Python buffer.

2. __init__(self: rti.connextdds.Int32Vector) -> None

Create an empty Int32Vector

3. __init__(self: rti.connextdds.Int32Vector, size: int) -> None

Create a Int32Vector with a preallocated size.

4. __init__(self: rti.connextdds.Int32Vector, vector: rti.connextdds.Int32Vector) -> None

Create a copy from another Int32Vector.

5. __init__(self: rti.connextdds.Int32Vector, arg0: Iterable) -> None

6. __init__(self: rti.connextdds.Int32Vector, list: rti.connextdds.Int32Seq) -> None

Create a Int32Vector from a list of values.

**__iter__** (*self:* rti.connextdds.Int32Vector) → Iterator[int]

Iterate over the contents of the vector.

**__len__** (*self:* rti.connextdds.Int32Vector) → int

Get the length of the Int32Vector.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Int32Vector, *arg0:* rti.connextdds.Int32Vector) → bool

Compare Int32Vectors for inequality.

**__setitem__** (*self:* rti.connextdds.Int32Vector, *arg0: int*, *arg1: int*) → None

Set the value at the specified index.

**clear** (*self:* rti.connextdds.Int32Vector) → None

Resize Int32Vector to 0.

**resize** (*self:* rti.connextdds.Int32Vector, *size: int*) → None

Resize Int32Vector.

**class** rti.connextdds.**Int64Seq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.Int64Seq, *arg0:* rti.connextdds.Int64Seq) → *rti.connextdds.Int64Seq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.Int64Seq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.Int64Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Int64Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Int64Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.Int64Seq, *arg0:* rti.connextdds.Int64Seq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Int64Seq, s: slice) -> rti.connextdds.Int64Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Int64Seq, arg0: int) -> int

**__getstate__** (*self:* rti.connextdds.Int64Seq) → bytes

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.Int64Seq, *arg0:* rti.connextdds.Int64Seq) → *rti.connextdds.Int64Seq*

**__imul__** (*self:* rti.connextdds.Int64Seq, *arg0: int*) → *rti.connextdds.Int64Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Int64Seq, arg0: Buffer) -> None
2. __init__(self: rti.connextdds.Int64Seq) -> None
3. __init__(self: rti.connextdds.Int64Seq, arg0: rti.connextdds.Int64Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Int64Seq, arg0: Iterable) -> None
5. __init__(self: rti.connextdds.Int64Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Int64Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Int64Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Int64Seq, *arg0: int*) → *rti.connextdds.Int64Seq*

**__ne__** (*self:* rti.connextdds.Int64Seq, *arg0:* rti.connextdds.Int64Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Int64Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Int64Seq, *arg0: int*) → *rti.connextdds.Int64Seq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Int64Seq, arg0: int, arg1: int) -> None
2. __setitem__(self: rti.connextdds.Int64Seq, arg0: slice, arg1: rti.connextdds.Int64Seq) -> None

Assign list elements using a slice object

**__setstate__**(*self:* rti.connextdds.Int64Seq, *arg0: bytes*) → None

**append**(*self:* rti.connextdds.Int64Seq, *x: int*) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.Int64Seq) → None

Clear the contents

**count**(*self:* rti.connextdds.Int64Seq, *x: int*) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.Int64Seq, L: rti.connextdds.Int64Seq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.Int64Seq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.Int64Seq, *i: int*, *x: int*) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.Int64Seq) -> int

Remove and return the last item

2. pop(self: rti.connextdds.Int64Seq, i: int) -> int

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.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.connextdds.Int64Seq, *arg0: int*) → None

resizes the vector to the given size

**class** rti.connextdds.**Int64Type**

Bases: *DynamicType*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Int64Type, *arg0:* rti.connextdds.Int64Type) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.Int64Type) → None

Get the singleton for Int64Type

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Int64Type, *arg0:* rti.connextdds.Int64Type) → bool

Test for inequality.

**class** rti.connextdds.**Int8Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Int8Seq, *arg0:* rti.connextdds.Int8Seq) → *rti.connextdds.Int8Seq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Int8Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Int8Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Int8Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Int8Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.Int8Seq, *arg0:* rti.connextdds.Int8Seq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Int8Seq, s: slice) -> rti.connextdds.Int8Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Int8Seq, arg0: int) -> int

**__getstate__**(*self:* rti.connextdds.Int8Seq) → bytes

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.Int8Seq, *arg0:* rti.connextdds.Int8Seq) → *rti.connextdds.Int8Seq*

**__imul__** (*self:* rti.connextdds.Int8Seq, *arg0: int*) → *rti.connextdds.Int8Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Int8Seq, arg0: Buffer) -> None
2. __init__(self: rti.connextdds.Int8Seq) -> None
3. __init__(self: rti.connextdds.Int8Seq, arg0: rti.connextdds.Int8Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Int8Seq, arg0: Iterable) -> None
5. __init__(self: rti.connextdds.Int8Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Int8Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Int8Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Int8Seq, *arg0: int*) → *rti.connextdds.Int8Seq*

**__ne__** (*self:* rti.connextdds.Int8Seq, *arg0:* rti.connextdds.Int8Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Int8Seq) → str

Return the string representation of this list. Values are printed as integers.

**__rmul__** (*self:* rti.connextdds.Int8Seq, *arg0: int*) → *rti.connextdds.Int8Seq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Int8Seq, arg0: int, arg1: int) -> None
2. __setitem__(self: rti.connextdds.Int8Seq, arg0: slice, arg1: rti.connextdds.Int8Seq) -> None

Assign list elements using a slice object

**__setstate__** (*self:* rti.connextdds.Int8Seq, *arg0: bytes*) → None

**append** (*self:* rti.connextdds.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*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**exception** rti.connextdds.**InvalidDowncastError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**InvalidLocalAdvanceNoticeStatus**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**property expiration_code**

Get the artifact's expiration reason code

**property expiration_message**

Get the artifact's expiration reason message

**property expiration_time**

Get the artifact's expiration time

**class** rti.connextdds.**ItemsView**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__iter__**(*self:* rti.connextdds.ItemsView) → Iterator

**__len__**(*self:* rti.connextdds.ItemsView) → int

**__module__ = 'rti.connextdds'**

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**class** rti.connextdds.**KeysView**

Bases: pybind11_object

**__annotations__ = {}**

**__contains__**(*self:* rti.connextdds.KeysView, *arg0: object*) → bool

**__init__**(**args*, ***kwargs*)

**__iter__**(*self:* rti.connextdds.KeysView) → Iterator

**__len__**(*self:* rti.connextdds.KeysView) → int

**__module__ = 'rti.connextdds'**

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**class** rti.connextdds.**LatencyBudget**

Bases: pybind11_object

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Lifespan, *arg0:* rti.connextdds.Lifespan) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Lifespan) -> None

Creates the default policy with an infinite lifespan.

2. __init__(self: rti.connextdds.Lifespan, duration: rti.connextdds.Duration) -> None

Creates a policy with the specified lifespan duration.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Lifespan, *arg0:* rti.connextdds.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.connextdds.**Liveliness**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Liveliness, *arg0:* rti.connextdds.Liveliness) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Liveliness) -> None

Creates an automatic liveliness policy with infinite lease duration.

2. __init__(self: rti.connextdds.Liveliness, kind: rti.connextdds.LivelinessKind, duration: rti.connextdds.Duration) -> None

Creates an instance with the specified liveliness kind and lease duration.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Liveliness, *arg0:* rti.connextdds.Liveliness) → bool

Test for inequality.

**property assertions_per_lease_duration**

The number of assertions to send during the lease duration.

**automatic = <rti.connextdds.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

**__annotations__ = {}**

**__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 Connext 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 Connext 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 Connext 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 Connext 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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.LivelinessKind.LivelinessKind) → int

**__init__**(*self:* rti.connextdds.LivelinessKind.LivelinessKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.LivelinessKind.LivelinessKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**MANUAL_BY_PARTICIPANT = <LivelinessKind.MANUAL_BY_PARTICIPANT: 1>**

**MANUAL_BY_TOPIC = <LivelinessKind.MANUAL_BY_TOPIC: 2>**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.LivelinessKind, *arg0:* rti.connextdds.LivelinessKind) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.LivelinessKind, *arg0:* rti.connextdds.LivelinessKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.LivelinessKind, *arg0:* rti.connextdds.LivelinessKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LivelinessKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.LivelinessKind, arg0: rti.connextdds.LivelinessKind.LivelinessKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.LivelinessKind) → *rti.connextdds.LivelinessKind.LivelinessKind*

Int conversion.

**__le__** (*self:* rti.connextdds.LivelinessKind, *arg0:* rti.connextdds.LivelinessKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.LivelinessKind, *arg0:* rti.connextdds.LivelinessKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.LivelinessKind, *arg0:* rti.connextdds.LivelinessKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.LivelinessKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**LivelinessLostStatus**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**property total_count**

Total count of times that a previously alive DataWriter became not alive due to a failure to assert its liveliness signal within the agreed lease duration.

**property total_count_change**

The delta number of times liveliness has been lost since the last time the listener callback fired or this status was checked.

**class** rti.connextdds.**LoanedDynamicData**

Bases: pybind11_object

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.LoanedDynamicData) → *rti.connextdds.LoanedDynamicData*

Enter a context for the loaned field, loan returned on context exit.

**__exit__**(*self:* rti.connextdds.LoanedDynamicData, *arg0: object*, *arg1: object*, *arg2: object*) → None

Exit the context for the loaned field, returning the resources.

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**property data**

Obtains the DynamicData object representing a member of a DynamicData object.

**return_loan**(*self:* rti.connextdds.LoanedDynamicData) → None

Explicitly return a dynamic data loan.

**class** rti.connextdds.**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.

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Locator, *arg0:* rti.connextdds.Locator) → bool

Compare Locators for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.Locator, *kind:* rti.connextdds.LocatorKind.LocatorKind, *port: int*, *address:* rti.connextdds.Uint8Seq) → None

Construct a Locator with the provided kind, port, and address.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Locator, *arg0:* rti.connextdds.Locator) → bool

Compare Locators for inequality.

**property address**

Get/set the address for this Locator.

**invalid = <rti.connextdds.Locator object>**

**property kind**

Get/set the kind for this Locator.

**property port**

Get/set the port for this Locator.

**class** rti.connextdds.**LocatorFilter**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.LocatorFilter, *arg0:* rti.connextdds.LocatorFilter) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LocatorFilter) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.LocatorFilter, locator_filters: rti.connextdds.LocatorFilterElementSeq, filter_name: str = Filter.stringmatch_filter_name) -> None

Creates an instance with a sequence of filters and a filter name.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.LocatorFilter, *arg0:* rti.connextdds.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.connextdds.**LocatorFilterElement**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.LocatorFilterElement, *arg0:* rti.connextdds.LocatorFilterElement) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.LocatorFilterElement, *arg0: str*, *arg1:* rti.connextdds.LocatorSeq) → None

Creates an instance with the provided filter_expression and locators.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.LocatorFilterElement, *arg0:* rti.connextdds.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.connextdds.**LocatorFilterElementSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.LocatorFilterElementSeq, *arg0:* rti.connextdds.LocatorFilterElementSeq) → *rti.connextdds.LocatorFilterElementSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.LocatorFilterElementSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.LocatorFilterElementSeq, *x:* rti.connextdds.LocatorFilterElement) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.LocatorFilterElementSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.LocatorFilterElementSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.LocatorFilterElementSeq, *arg0:* rti.connextdds.LocatorFilterElementSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.LocatorFilterElementSeq, s: slice) -> rti.connextdds.LocatorFilterElementSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.LocatorFilterElementSeq, arg0: int) -> rti.connextdds.LocatorFilterElement

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.LocatorFilterElementSeq, *arg0:* rti.connextdds.LocatorFilterElementSeq) → *rti.connextdds.LocatorFilterElementSeq*

**__imul__**(*self:* rti.connextdds.LocatorFilterElementSeq, *arg0: int*) → *rti.connextdds.LocatorFilterElementSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LocatorFilterElementSeq) -> None

2. __init__(self: rti.connextdds.LocatorFilterElementSeq, arg0: rti.connextdds.LocatorFilterElementSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.LocatorFilterElementSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.LocatorFilterElementSeq) → Iterator[*rti.connextdds.LocatorFilterElement*]

**__len__**(*self:* rti.connextdds.LocatorFilterElementSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.LocatorFilterElementSeq, *arg0: int*) → *rti.connextdds.LocatorFilterElementSeq*

**__ne__**(*self:* rti.connextdds.LocatorFilterElementSeq, *arg0:* rti.connextdds.LocatorFilterElementSeq) → bool

**__rmul__**(*self:* rti.connextdds.LocatorFilterElementSeq, *arg0: int*) → *rti.connextdds.LocatorFilterElementSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.LocatorFilterElementSeq, arg0: int, arg1: rti.connextdds.LocatorFilterElement) -> None

2. __setitem__(self: rti.connextdds.LocatorFilterElementSeq, arg0: slice, arg1: rti.connextdds.LocatorFilterElementSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.LocatorFilterElementSeq, *x:* rti.connextdds.LocatorFilterElement) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.LocatorFilterElementSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.LocatorFilterElementSeq, *x:* rti.connextdds.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 Connext 5.1.0 or earlier.

INTRA : A locator for the Connext INTRA transport.

UDPv6 : A locator for a UDPv6 address.

UDPv6_510 : A locator for a UDPv6 address with backwards compatibility for Connext 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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.LocatorKind.LocatorKind) → int

**__init__**(*self:* rti.connextdds.LocatorKind.LocatorKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.LocatorKind.LocatorKind, *state: int*) → None

**__str__**(*self: object*) → 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>**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.LocatorKind, *arg0:* rti.connextdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.LocatorKind, *arg0:* rti.connextdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.LocatorKind, *arg0:* rti.connextdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LocatorKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.LocatorKind, arg0: rti.connextdds.LocatorKind.LocatorKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.LocatorKind) → *rti.connextdds.LocatorKind.LocatorKind*

Int conversion.

**__le__** (*self:* rti.connextdds.LocatorKind, *arg0:* rti.connextdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.LocatorKind, *arg0:* rti.connextdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.LocatorKind, *arg0:* rti.connextdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.LocatorKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**LocatorSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.LocatorSeq, *arg0:* rti.connextdds.LocatorSeq) → *rti.connextdds.LocatorSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.LocatorSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.LocatorSeq, *x:* rti.connextdds.Locator) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.LocatorSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.LocatorSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.LocatorSeq, *arg0:* rti.connextdds.LocatorSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.LocatorSeq, s: slice) -> rti.connextdds.LocatorSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.LocatorSeq, arg0: int) -> rti.connextdds.Locator

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.LocatorSeq, *arg0:* rti.connextdds.LocatorSeq) → *rti.connextdds.LocatorSeq*

**__imul__** (*self:* rti.connextdds.LocatorSeq, *arg0: int*) → *rti.connextdds.LocatorSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LocatorSeq) -> None

2. __init__(self: rti.connextdds.LocatorSeq, arg0: rti.connextdds.LocatorSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.LocatorSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.LocatorSeq) → Iterator[*rti.connextdds.Locator*]

**__len__** (*self:* rti.connextdds.LocatorSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.LocatorSeq, *arg0: int*) → *rti.connextdds.LocatorSeq*

**__ne__** (*self:* rti.connextdds.LocatorSeq, *arg0:* rti.connextdds.LocatorSeq) → bool

**__rmul__** (*self:* rti.connextdds.LocatorSeq, *arg0: int*) → *rti.connextdds.LocatorSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.LocatorSeq, arg0: int, arg1: rti.connextdds.Locator) -> None

2. __setitem__(self: rti.connextdds.LocatorSeq, arg0: slice, arg1: rti.connextdds.LocatorSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.LocatorSeq, *x:* rti.connextdds.Locator) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.LocatorSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.LocatorSeq, *x:* rti.connextdds.Locator) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.LocatorSeq, L: rti.connextdds.LocatorSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.LocatorSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.LocatorSeq, *i: int*, *x:* rti.connextdds.Locator) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.LocatorSeq) -> rti.connextdds.Locator

Remove and return the last item

2. pop(self: rti.connextdds.LocatorSeq, i: int) -> rti.connextdds.Locator

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.LocatorSeq, *x:* rti.connextdds.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.connextdds.`**LocatorVector**

Bases: `pybind11_object`

A DDS standard container with functionality similar to a C++ vector.

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.LocatorVector, *arg0:* rti.connextdds.LocatorVector) → bool

Compare LocatorVectors for equality.

**__getitem__** (*self:* rti.connextdds.LocatorVector, *arg0: int*) → *rti.connextdds.Locator*

Get the value at the specified index.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.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[*rti.connextdds.Locator*]

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 Connext 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 Connext 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 Connext (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 Connext.

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.LogCategory.LogCategory) → int

**__init__**(*self:* rti.connextdds.LogCategory.LogCategory, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__** (*self:* rti.connextdds.LogCategory.LogCategory, *state: int*) → None

**__str__** (*self: object*) → 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**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.LogCategory, *arg0:* rti.connextdds.LogCategory) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.LogCategory, *arg0:* rti.connextdds.LogCategory) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.LogCategory, *arg0:* rti.connextdds.LogCategory) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LogCategory) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.LogCategory, arg0: rti.connextdds.LogCategory.LogCategory) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.LogCategory) → *rti.connextdds.LogCategory.LogCategory*

Int conversion.

**__le__** (*self:* rti.connextdds.LogCategory, *arg0:* rti.connextdds.LogCategory) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.LogCategory, *arg0:* rti.connextdds.LogCategory) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.LogCategory, *arg0:* rti.connextdds.LogCategory) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.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.connextdds.**LogFacility**

Bases: pybind11_object

Members:

user : A log message produced by the APIs that log user messages.

security_event : A security-related message logged by the RTI Security Plugins Logging Plugin.

service : A log message produced by an Infrastructure Service, such as Routing Service.

middleware : A log message produced by RTI Connext Core Libraries.

**__annotations__ = {}**

**__eq__** (*self: object*, *other: object*) → bool

**__getstate__** (*self: object*) → int

**__hash__** (*self: object*) → int

**__index__**(*self:* rti.connextdds.LogFacility) → int

**__init__**(*self:* rti.connextdds.LogFacility, *value: int*) → None

**__int__**(*self:* rti.connextdds.LogFacility) → int

**__members__ = {'middleware': <LogFacility.middleware: 23>, 'security_event': <LogFacility.security_event: 10>, 'service': <LogFacility.service: 22>, 'user': <LogFacility.user: 1>}**

**__module__ = 'rti.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.LogFacility, *state: int*) → None

**__str__**(*self: object*) → str

**middleware = <LogFacility.middleware: 23>**

**property name**

**security_event = <LogFacility.security_event: 10>**

**service = <LogFacility.service: 22>**

**user = <LogFacility.user: 1>**

**property value**

**class** rti.connextdds.**LogLevel**

Bases: pybind11_object

**EXCEPTION = <LogLevel.EXCEPTION: 1>**

**FATAL_ERROR = <LogLevel.FATAL_ERROR: 0>**

**class LogLevel**

Bases: pybind11_object

Members:

FATAL_ERROR : The message describes a fatal error.

EXCEPTION : The message describes an error.

WARNING : The message describes a warning.

STATUS_LOCAL : The message contains information about the life-cycles of local RTI Connext objects.

STATUS_REMOTE : The message contains information about the life-cycles of remote RTI Connext objects.

STATUS_ALL : The message contains debug information that might be relevant to your application.

**EXCEPTION = <LogLevel.EXCEPTION: 1>**

**FATAL_ERROR = <LogLevel.FATAL_ERROR: 0>**

**STATUS_ALL = <LogLevel.STATUS_ALL: 5>**

**STATUS_LOCAL = <LogLevel.STATUS_LOCAL: 3>**

**STATUS_REMOTE = <LogLevel.STATUS_REMOTE: 4>**

**WARNING = <LogLevel.WARNING: 2>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.LogLevel.LogLevel) → int

**__init__**(*self:* rti.connextdds.LogLevel.LogLevel, *value: int*) → None

**__int__**(*self:* rti.connextdds.LogLevel.LogLevel) → int

**__members__ = {'EXCEPTION': <LogLevel.EXCEPTION: 1>, 'FATAL_ERROR': <LogLevel.FATAL_ERROR: 0>, 'STATUS_ALL': <LogLevel.STATUS_ALL: 5>, 'STATUS_LOCAL': <LogLevel.STATUS_LOCAL: 3>, 'STATUS_REMOTE': <LogLevel.STATUS_REMOTE: 4>, 'WARNING': <LogLevel.WARNING: 2>}**

**__module__ = 'rti.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.LogLevel.LogLevel, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**STATUS_ALL = <LogLevel.STATUS_ALL: 5>**

**STATUS_LOCAL = <LogLevel.STATUS_LOCAL: 3>**

**STATUS_REMOTE = <LogLevel.STATUS_REMOTE: 4>**

**WARNING = <LogLevel.WARNING: 2>**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.LogLevel, *arg0:* rti.connextdds.LogLevel) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.LogLevel, *arg0:* rti.connextdds.LogLevel) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.LogLevel, *arg0:* rti.connextdds.LogLevel) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LogLevel) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.LogLevel, arg0: rti.connextdds.LogLevel.LogLevel) -> None

Copy constructor.

**__int__**(*self:* rti.connextdds.LogLevel) → *rti.connextdds.LogLevel.LogLevel*

Int conversion.

**__le__**(*self:* rti.connextdds.LogLevel, *arg0:* rti.connextdds.LogLevel) → bool

Apply operator to underlying enumerated values.

**__lt__**(*self:* rti.connextdds.LogLevel, *arg0:* rti.connextdds.LogLevel) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.LogLevel, *arg0:* rti.connextdds.LogLevel) → bool

Apply operator to underlying enumerated values.

**__str__**(*self:* rti.connextdds.LogLevel) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**LogMessage**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**property facility**

The Facility associated with the log message.

**property is_license_message**

True if this message is a license message.

**property is_security_message**

True if this message is a security message.

**property level**

Log level of this message.

**property message_id**

A numeric code that identifies a specific log message.

**property text**

The text of this message.

**property timestamp**

The time when the log message was printed.

**class** rti.connextdds.**Logger**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**alert**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs an alert message in the user category.

**critical**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs a critical message in the user category.

**debug**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs a debug message in the user category.

**emergency**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs an emergency message in the user category.

**error**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs an error message in the user category.

**informational**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs an informational message in the user category.

**instance = <rti.connextdds.Logger object>**

**notice**(*self:* rti.connextdds.Logger, *msg: str*) → None

Logs a notice message in the user category.

**output_file** (*self:* rti.connextdds.Logger, *file_name: str*) → None

Set the name of the file to which the logged output is redirected.

**output_file_set** (*self:* rti.connextdds.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.connextdds.Logger, *callable_handler: object*) → None

Assigns a callable to which log messages (strings) are directed

**output_handler_extended** (*self:* rti.connextdds.Logger, *callable_handler: object*) → None

Assigns a callable that receives a LogMessage object with additional information.

**property print_format**

The message format that RTI Connext will use to log diagnostic information.

**reset_output_handler** (*self:* rti.connextdds.Logger) → None

Removes the current log handler and sends logging back to the standard output.

**property verbosity**

The verbosity at which RTI Connext is currently logging diagnostic information.

**verbosity_by_category** (**args*, ***kwargs*)

Overloaded function.

1. verbosity_by_category(self: rti.connextdds.Logger, category: rti.connextdds.LogCategory) -> rti.connextdds.Verbosity

Get the verbosity at which RTI Connext is currently logging diagnostic information in the given category.

2. verbosity_by_category(self: rti.connextdds.Logger, category: rti.connextdds.LogCategory, verbosity: rti.connextdds.Verbosity) -> None

Set the verbosity at which RTI Connext will log diagnostic information in the given category.

**warning** (*self:* rti.connextdds.Logger, *msg: str*) → None

Logs a warning message in the user category.

**class** rti.connextdds.**LongDouble**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.LongDouble, *arg0:* rti.connextdds.LongDouble) → bool

Test for equality

**__getitem__** (*self:* rti.connextdds.LongDouble, *arg0: int*) → int

Get a byte from the LongDouble buffer.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.LongDouble) -> None

Creates a LongDouble with value 0.

2. __init__(self: rti.connextdds.LongDouble, byte_sequence: rti.connextdds.Uint8Seq) -> None

Creates a LongDouble from a sequence of bytes.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.LongDouble, *arg0:* rti.connextdds.LongDouble) → bool

Test for inequality

**__setitem__** (*self:* rti.connextdds.LongDouble, *arg0: int*, *arg1: int*) → None

Set a byte in the LongDouble buffer.

**__str__** (*self:* rti.connextdds.LongDouble) → str

rti.connextdds.**LongDoubleType**

alias of *Float128Type*

rti.connextdds.**LongLongSeq**

alias of *Int64Seq*

rti.connextdds.**LongLongType**

alias of *Int64Type*

rti.connextdds.**LongSeq**

alias of *Int32Seq*

rti.connextdds.**LongType**

alias of *Int32Type*

**class** rti.connextdds.**Member**

Bases: pybind11_object

**INVALID_ID = 2147483647**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Member, *arg0:* rti.connextdds.Member) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Member, name: str, data_type: rti.connextdds.DynamicType) -> None

2. __init__(self: rti.connextdds.Member, name: str, data_type: rti.connextdds.DynamicType, id: int = -1, is_key: bool = False, is_optional: bool = False, is_pointer: bool = False, is_must_understand: bool = False) -> None

Create a Member with specified properties

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Member, *arg0:* rti.connextdds.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 must_understand**

Member must-understand status

**property name**

The member name.

**property optional**

Member optional status

**property pointer**

Member pointer status

**set_name** (*self:* rti.connextdds.Member, *arg0: str*) → *rti.connextdds.Member*

Set the member name.

**property type**

Gets the member type.

**property unit**

Retrieves the unit annotation of the member.

**class** rti.connextdds.**MemberSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.MemberSeq, *arg0:* rti.connextdds.MemberSeq) → *rti.connextdds.MemberSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.MemberSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.MemberSeq, *x:* rti.connextdds.Member) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.MemberSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.MemberSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.MemberSeq, *arg0:* rti.connextdds.MemberSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.MemberSeq, s: slice) -> rti.connextdds.MemberSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.MemberSeq, arg0: int) -> rti.connextdds.Member

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.MemberSeq, *arg0:* rti.connextdds.MemberSeq) → *rti.connextdds.MemberSeq*

**__imul__** (*self:* rti.connextdds.MemberSeq, *arg0: int*) → *rti.connextdds.MemberSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.MemberSeq) -> None

2. __init__(self: rti.connextdds.MemberSeq, arg0: rti.connextdds.MemberSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.MemberSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.MemberSeq) → Iterator[*rti.connextdds.Member*]

**__len__** (*self:* rti.connextdds.MemberSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.MemberSeq, *arg0: int*) → *rti.connextdds.MemberSeq*

**__ne__** (*self:* rti.connextdds.MemberSeq, *arg0:* rti.connextdds.MemberSeq) → bool

**__rmul__** (*self:* rti.connextdds.MemberSeq, *arg0: int*) → *rti.connextdds.MemberSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.MemberSeq, arg0: int, arg1: rti.connextdds.Member) -> None

2. __setitem__(self: rti.connextdds.MemberSeq, arg0: slice, arg1: rti.connextdds.MemberSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.MemberSeq, *x:* rti.connextdds.Member) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.MemberSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.MemberSeq, *x:* rti.connextdds.Member) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.MemberSeq, L: rti.connextdds.MemberSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.MemberSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.MemberSeq, *i: int*, *x:* rti.connextdds.Member) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.MemberSeq) -> rti.connextdds.Member

Remove and return the last item

2. pop(self: rti.connextdds.MemberSeq, i: int) -> rti.connextdds.Member

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.MemberSeq, *x:* rti.connextdds.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.connextdds.**Monitoring**

Bases: `pybind11_object`

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Monitoring, *arg0:* rti.connextdds.Monitoring) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.Monitoring) → None

Creates the default policy (monitoring is disabled).

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Monitoring, *arg0:* rti.connextdds.Monitoring) → bool

Test for inequality.

**property application_name**

The name of the resource that represents this Connext application.

**disabled = <rti.connextdds.Monitoring object>**

**property distribution_settings**

Configures the distribution of telemetry data.

**property enable**

Enables the collection and distribution of telemetry data for a Connext application using the Monitoring Library 2.0.

**enabled = <rti.connextdds.Monitoring object>**

**property telemetry_data**

Configures the collection of telemetry data.

**class** rti.connextdds.**MonitoringDedicatedParticipantSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.MonitoringLoggingDistributionSettings, *arg0:* rti.connextdds.MonitoringLoggingDistributionSettings) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.MonitoringLoggingDistributionSettings) → None

Creates an instance with the default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.MonitoringLoggingDistributionSettings, *arg0:* rti.connextdds.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.connextdds.**MonitoringLoggingForwardingSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.MonitoringLoggingForwardingSettings, *arg0:* rti.connextdds.MonitoringLoggingForwardingSettings) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.MonitoringLoggingForwardingSettings) → None

Creates an instance with the default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.MonitoringLoggingForwardingSettings, *arg0:* rti.connextdds.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.connextdds.**MonitoringMetricSelection**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.MonitoringMetricSelection, *arg0:* rti.connextdds.MonitoringMetricSelection) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.MonitoringMetricSelection) → None

Creates an instance with the default settings.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.MonitoringMetricSelection, *arg0:* rti.connextdds.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.connextdds.**MonitoringMetricSelectionSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0:* rti.connextdds.MonitoringMetricSelectionSeq) → *rti.connextdds.MonitoringMetricSelectionSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *x:* rti.connextdds.MonitoringMetricSelection) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0:* rti.connextdds.MonitoringMetricSelectionSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.MonitoringMetricSelectionSeq, s: slice) -> rti.connextdds.MonitoringMetricSelectionSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: int) -> rti.connextdds.MonitoringMetricSelection

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0:* rti.connextdds.MonitoringMetricSelectionSeq) → *rti.connextdds.MonitoringMetricSelectionSeq*

**__imul__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0: int*) → *rti.connextdds.MonitoringMetricSelectionSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.MonitoringMetricSelectionSeq) -> None

2. __init__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: rti.connextdds.MonitoringMetricSelectionSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq) → Iterator[*rti.connextdds.MonitoringMetricSelection*]

**__len__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0: int*) → *rti.connextdds.MonitoringMetricSelectionSeq*

**__ne__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0:* rti.connextdds.MonitoringMetricSelectionSeq) → bool

**__rmul__** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *arg0: int*) → *rti.connextdds.MonitoringMetricSelectionSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: int, arg1: rti.connextdds.MonitoringMetricSelection) -> None

2. __setitem__(self: rti.connextdds.MonitoringMetricSelectionSeq, arg0: slice, arg1: rti.connextdds.MonitoringMetricSelectionSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *x:* rti.connextdds.MonitoringMetricSelection) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.MonitoringMetricSelectionSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *x:* rti.connextdds.MonitoringMetricSelection) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.MonitoringMetricSelectionSeq, L: rti.connextdds.MonitoringMetricSelectionSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.MonitoringMetricSelectionSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *i: int*, *x:* rti.connextdds.MonitoringMetricSelection) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.MonitoringMetricSelectionSeq) -> rti.connextdds.MonitoringMetricSelection

Remove and return the last item

2. pop(self: rti.connextdds.MonitoringMetricSelectionSeq, i: int) -> rti.connextdds.MonitoringMetricSelection

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.MonitoringMetricSelectionSeq, *x:* rti.connextdds.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.connextdds.**MonitoringPeriodicDistributionSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.MonitoringPeriodicDistributionSettings, *arg0:* rti.connextdds.MonitoringPeriodicDistributionSettings) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.MonitoringPeriodicDistributionSettings) → None

Creates an instance with the default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.MonitoringPeriodicDistributionSettings, *arg0:* rti.connextdds.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.connextdds.**MonitoringTelemetryData**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.MonitoringTelemetryData, *arg0:* rti.connextdds.MonitoringTelemetryData) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.MonitoringTelemetryData) → None

Creates an instance with the default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.MonitoringTelemetryData, *arg0:* rti.connextdds.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.connextdds.**MultiChannel**

Bases: pybind11_object

**SQL_FILTER_NAME = 'DDSSQL'**

**STRINGMATCH_FILTER_NAME = 'DDSSTRINGMATCH'**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.MultiChannel, *arg0:* rti.connextdds.MultiChannel) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.MultiChannel) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.MultiChannel, channels: rti.connextdds.ChannelSettingsSeq, filter_name: str = Filter.stringmatch_filter_name) -> None

Creates an instance with the specified channels and filter name.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.MultiChannel, *arg0:* rti.connextdds.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.connextdds.**MulticastMapping**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.MulticastMapping, *arg0:* rti.connextdds.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*

**__annotations__ = {}**

**__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.connextdds.MulticastMappingSeq, s: slice) -> rti.connextdds.MulticastMappingSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.MulticastMappingSeq, arg0: int) -> rti.connextdds.MulticastMapping

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.MulticastMappingSeq, *arg0:* rti.connextdds.MulticastMappingSeq) → *rti.connextdds.MulticastMappingSeq*

**__imul__**(*self:* rti.connextdds.MulticastMappingSeq, *arg0: int*) → *rti.connextdds.MulticastMappingSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.MulticastMappingSeq) -> None

2. __init__(self: rti.connextdds.MulticastMappingSeq, arg0: rti.connextdds.MulticastMappingSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.MulticastMappingSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.MulticastMappingSeq) → Iterator[*rti.connextdds.MulticastMapping*]

**__len__**(*self:* rti.connextdds.MulticastMappingSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.MulticastMappingSeq, *arg0: int*) → *rti.connextdds.MulticastMappingSeq*

**__ne__**(*self:* rti.connextdds.MulticastMappingSeq, *arg0:* rti.connextdds.MulticastMappingSeq) → bool

**__rmul__**(*self:* rti.connextdds.MulticastMappingSeq, *arg0: int*) → *rti.connextdds.MulticastMappingSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.MulticastMappingSeq, arg0: int, arg1: rti.connextdds.MulticastMapping) -> None

2. __setitem__(self: rti.connextdds.MulticastMappingSeq, arg0: slice, arg1: rti.connextdds.MulticastMappingSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.MulticastMappingSeq, *x:* rti.connextdds.MulticastMapping) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.MulticastMappingSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.MulticastMappingSeq, *x:* rti.connextdds.MulticastMapping) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.MulticastMappingSeq, L: rti.connextdds.MulticastMappingSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.MulticastMappingSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.MulticastMappingSeq, *i: int*, *x:* rti.connextdds.MulticastMapping) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.MulticastMappingSeq) -> rti.connextdds.MulticastMapping

Remove and return the last item

2. pop(self: rti.connextdds.MulticastMappingSeq, i: int) -> rti.connextdds.MulticastMapping

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.MulticastMappingSeq, *x:* rti.connextdds.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.connextdds.**NoOpAnyDataReaderListener**

Bases: *AnyDataReaderListener*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.NoOpAnyDataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.NoOpAnyDataReaderListener, *arg0:* rti.connextdds.AnyDataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class** rti.connextdds.**NoOpAnyDataWriterListener**

Bases: *AnyDataWriterListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.NoOpAnyDataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.InstanceHandle) → None

On instance replaced callback.

**on_liveliness_lost**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

**on_offered_deadline_missed**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.NoOpAnyDataWriterListener, *arg0:* rti.connextdds.AnyDataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class** rti.connextdds.**NoOpAnyTopicListener**

Bases: *AnyTopicListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.NoOpAnyTopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.NoOpAnyTopicListener, *arg0:* rti.connextdds.AnyTopic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class** rti.connextdds.**NoOpDataReaderListener**

Bases: *DataReaderListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.NoOpDataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.NoOpDataReaderListener, *arg0:* rti.connextdds.DataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class** rti.connextdds.**NoOpDataWriterListener**

Bases: *DataWriterListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.NoOpDataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.NoOpDataWriterListener, *arg0:* rti.connextdds.DataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.NoOpDataWriterListener, *arg0:* rti.connextdds.DataWriter, *arg1:* rti.connextdds.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*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.NoOpDomainParticipantListener) → None

**__module__ = 'rti.connextdds'**

**on_invalid_local_status_advance_notice**(*self:* rti.connextdds.NoOpDomainParticipantListener, *arg0:* rti.connextdds.DomainParticipant, *arg1:* rti.connextdds.InvalidLocalAdvanceNoticeStatus) → None

On invalid local status advance notice callback

**class** rti.connextdds.**NoOpPublisherListener**

Bases: *PublisherListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.NoOpPublisherListener) → None

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**NoOpSubscriberListener**

Bases: *SubscriberListener*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**exception** rti.connextdds.**NotEnabledError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**exception** rti.connextdds.**NullReferenceError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

rti.connextdds.**OctetSeq**

alias of *Uint8Seq*

**class** rti.connextdds.**OctetType**

Bases: *DynamicType*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.OctetType, *arg0:* rti.connextdds.OctetType) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.OctetType) → None

Get the singleton for OctetType

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.OctetType, *arg0:* rti.connextdds.OctetType) → bool

Test for inequality.

**class** rti.connextdds.**OfferedDeadlineMissedStatus**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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.connextdds.**OfferedIncompatibleQosStatus**

Bases: pybind11_object

**__annotations__ = {}**

**__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 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.connextdds.**OutOfResourcesError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**Ownership**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Ownership, *arg0:* rti.connextdds.Ownership) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Ownership) -> None

Creates an ownership policy set to shared.

2. __init__(self: rti.connextdds.Ownership, kind: rti.connextdds.OwnershipKind) -> None

Creates an instance with the specified ownership kind.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Ownership, *arg0:* rti.connextdds.Ownership) → bool

Test for inequality.

**exclusive = <rti.connextdds.Ownership object>**

**property kind**

The ownership kind.

**shared = <rti.connextdds.Ownership object>**

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**SHARED = <OwnershipKind.SHARED: 0>**

**__annotations__ = {}**

**__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.connextdds.OwnershipKind, arg0: rti.connextdds.OwnershipKind.OwnershipKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.OwnershipKind) → *rti.connextdds.OwnershipKind.OwnershipKind*

Int conversion.

**__le__** (*self:* rti.connextdds.OwnershipKind, *arg0:* rti.connextdds.OwnershipKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.OwnershipKind, *arg0:* rti.connextdds.OwnershipKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.OwnershipKind, *arg0:* rti.connextdds.OwnershipKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.OwnershipKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**OwnershipStrength**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.OwnershipStrength, *arg0:* rti.connextdds.OwnershipStrength) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.OwnershipStrength) -> None

Creates an instance with the default strength (0).

2. __init__(self: rti.connextdds.OwnershipStrength, strength: int) -> None

Creates an instance with the specified strength value.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.OwnershipStrength, *arg0:* rti.connextdds.OwnershipStrength) → bool

Test for inequality.

**property value**

The ownership kind.

**class** rti.connextdds.**ParticipantBuiltinTopicData**

Bases: pybind11_object

**class ContentFilter**

Bases: *ContentFilterBase*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilter) → None

**__module__ = 'rti.connextdds'**

**compile**(*self:* rti.connextdds.ParticipantBuiltinTopicData.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.connextdds.ParticipantBuiltinTopicData.ContentFilter, *compile_data: Optional[object]*, *sample:* rti.connextdds.ParticipantBuiltinTopicData, *meta_data:* rti.connextdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

**finalize**(*self:* rti.connextdds.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 ContentFilteredTopic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, topic: rti.connextdds.ParticipantBuiltinTopicData.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, topic_description: rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

**append_to_expression_parameter** (*self:* rti.connextdds.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.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic*]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

**remove_from_expression_parameter** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *index: int*, *remove_term: str*) → None

Removes the specified term from the parameter at the provided index.

**set_filter** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.Filter) → None

Set the filter.

**property topic**

Get the underlying Topic.

**class ContentFilteredTopicSeq**

Bases: `pybind11_object`

**__add__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, s: slice) -> rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

**__imul__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None
2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic*]

**__len__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

**__ne__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → bool

**__rmul__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: int, arg1: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) -> None
2. __setitem__(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, L: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *i: int*, *x:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic

Remove and return the last item

2. pop(self: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, i: int) -> rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *datareader:* rti.connextdds.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.

**__annotations__ = {}**

**__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.connextdds.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.DataReader) → list

Read copies of all available samples (data and info).

**read_data** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → list

Read copies of all available valid data.

**read_loaned** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → *rti.connextdds.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::ParticipantBuiltinTopicDataImpl>, 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.connextdds.ParticipantBuiltinTopicData.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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.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*

**__annotations__ = {}**

**__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.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None
2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData.DataReader*]

**__len__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq*

**__ne__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → bool

**__rmul__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: int, arg1: rti.connextdds.ParticipantBuiltinTopicData.DataReader) -> None
2. __setitem__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: slice, arg1: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, L: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *i: int*, *x:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) -> rti.connextdds.ParticipantBuiltinTopicData.DataReader

Remove and return the last item

2. pop(self: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, i: int) -> rti.connextdds.ParticipantBuiltinTopicData.DataReader

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → *rti.connextdds.ParticipantBuiltinTopicData.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

**__eq__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → bool

Test for equality.

**__exit__** (*self:* rti.connextdds.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.connextdds.ParticipantBuiltinTopicData.DataWriter, topic: rti.connextdds.ParticipantBuiltinTopicData.Topic) -> None

Creates a DataWriter in the implicit publisher with default QoS.

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, 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 the implicit publisher with specific QoS and optionally a listener.

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.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.ParticipantBuiltinTopicDataTimestampedSeq) -> 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.connextdds.ParticipantBuiltinTopicData.DataWriter) → None

Manually asserts the liveliness of the DataWriter.

**close** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → None

Close this DataWriter.

**create_data** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → *rti.connextdds.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.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter

Dispose an instance.

2. dispose_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter

Dispose an instance with a timestamp.

3. dispose_instance(self: rti.connextdds.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object

Dispose an instance.

2. dispose_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object

Dispose an instance with a timestamp.

3. dispose_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData) -> object

Dispose the instance associated with key_holder.

4. dispose_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData, timestamp: rti.connextdds.Time) -> object

Dispose the instance associated with key_holder using a timestamp

5. dispose_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.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.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.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.DataWriter*]

Find DataWriter in publisher with a topic name, returning the first found.

**flush**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → None

Flushes the batch in progress in the context of thecalling thread.

**is_matched_subscription_active**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg0:* rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

**is_sample_app_acknowledged**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *sample_id:* rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

**key_value**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *handle:* rti.connextdds.InstanceHandle) → *rti.connextdds.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.connextdds.ParticipantBuiltinTopicData.DataWriter, *key_holder:* rti.connextdds.ParticipantBuiltinTopicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

**matched_subscription_data**(*self:* rti.connextdds.ParticipantBuiltinTopicData.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.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 Connext 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 Connext 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`

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, s: slice) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → *rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq*

**__imul__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) -> None
2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData.DataWriter*]

**__len__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0: int*) → *rti.connextdds.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 ITopicDescription**

Bases: *IEntity*

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.LoanedSample) -> None

Basic constructor

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.LoanedSample, data: rti.connextdds.ParticipantBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None

Construct LoanedSample with data and info.

**__iter__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.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

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples) → *rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

**__exit__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples, *arg0: object*, *arg1: object*, *arg2: object*) → None

Exit the context for the loaned samples, returning the resources.

**__getitem__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.LoanedSample*

Access a LoanedSample object in an array-like syntax

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples) → None

Create an empty LoanedSamples object.

**__iter__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample*]

**__len__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.LoanedSamples) → None

Returns the loan to the DataReader.

**class NoOpDataReaderListener**

Bases: *DataReaderListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1:* rti.connextdds.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*

**__annotations__ = {}**

**__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.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class NoOpTopicListener**

Bases: *TopicListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class Sample**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Sample, data: rti.connextdds.ParticipantBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None

Construct Sample with data and info.

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Sample, sample: rti.connextdds.ParticipantBuiltinTopicData.Sample) -> None

Copy constructor.

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Sample) -> None

Basic constructor

4. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Sample, loaned_sample: rti.connextdds.ParticipantBuiltinTopicData.LoanedSample) -> None

Construct a sample with a loaned sample.

**__iter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.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

**__annotations__ = {}**

**__getitem__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.SharedSamples, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicData.LoanedSample*

Get the sample at the specified index.

**__init__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.SharedSamples, *loaned_samples:* rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples) → None

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

**__iter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.SharedSamples) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample*]

Make a sample iterator

**__len__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.SharedSamples) → int

Returns the number of samples.

**__module__ = 'rti.connextdds'**

**class Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.Topic, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.Topic) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Topic, entity: rti.connextdds.IEntity) -> None

Cast an Entity to a Topic.

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Topic, topic_description: rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.Topic, topic: rti.connextdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.Topic, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.Topic) → bool

Test for inequality.

**static find**(*participant:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.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.connextdds.ParticipantBuiltinTopicData.Topic, *listener:* rti.connextdds.ParticipantBuiltinTopicData.TopicListener, *event_mask:* rti.connextdds.StatusMask) → None

Set the listener and event mask.

**class TopicDescription**

Bases: *ITopicDescription*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.TopicDescription) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicDescription, topic: rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicDescription, entity: rti.connextdds.IEntity) -> None

Cast a Topic to a TopicDescription.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.TopicDescription) → bool

Test for inequality.

**class TopicListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicListener, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class TopicSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) → *rti.connextdds.ParticipantBuiltinTopicData.TopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData.Topic) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.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[*rti.connextdds.ParticipantBuiltinTopicData.Topic*]

**__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`

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples) → *rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples*

**__exit__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples, *arg0: object*, *arg1: object*, *arg2: object*) → None

**__init__** (**args*, ***kwargs*)

**__iter__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample*]

**__module__ = 'rti.connextdds'**

**class WriterContentFilter**

Bases: *ContentFilter*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter) → None

**__module__ = 'rti.connextdds'**

**writer_attach**(*self:* rti.connextdds.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.connextdds.ParticipantBuiltinTopicData.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.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.connextdds.ParticipantBuiltinTopicData.WriterContentFilter, *writer_filter_data: Optional[object]*, *sample:* rti.connextdds.ParticipantBuiltinTopicData, *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.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.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*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilterHelper) → None

**__module__ = 'rti.connextdds'**

**add_cookie** (*self:* rti.connextdds.ParticipantBuiltinTopicData.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.ParticipantBuiltinTopicData.WriterContentFilterHelper, *writer_filter_data: Optional[object]*, *sample:* rti.connextdds.ParticipantBuiltinTopicData, *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.

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ParticipantBuiltinTopicData, *arg0:* rti.connextdds.ParticipantBuiltinTopicData) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.ParticipantBuiltinTopicData) → None

Create a default ParticipantBuiltinTopicData.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ParticipantBuiltinTopicData, *arg0:* rti.connextdds.ParticipantBuiltinTopicData) → bool

Test for inequality.

**property available_builtin_endpoints_ext**

Extended bitmap of builtin endpoints supported by the participant.

**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 Connext.

**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.connextdds.**ParticipantBuiltinTopicDataSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → *rti.connextdds.ParticipantBuiltinTopicDataSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *x:* rti.connextdds.ParticipantBuiltinTopicData) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, s: slice) -> rti.connextdds.ParticipantBuiltinTopicDataSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → *rti.connextdds.ParticipantBuiltinTopicDataSeq*

**__imul__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicDataSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> None

2. __init__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → Iterator[*rti.connextdds.ParticipantBuiltinTopicData*]

**__len__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicDataSeq*

**__ne__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataSeq) → bool

**__rmul__** (*self:* rti.connextdds.ParticipantBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.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.connextdds.**ParticipantBuiltinTopicDataTimestampedSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → *rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *x: tuple[*rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time*]*) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, s: slice) -> rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: int) -> tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time]

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → *rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq*

**__imul__**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.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[tuple[*rti.connextdds.ParticipantBuiltinTopicData*, *rti.connextdds.Time*]]

**__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_cxxabi1011__ = <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.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *i: int*, *x:* *tuple[*rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time*]*) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) -> tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time]

Remove and return the last item

2. pop(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, i: int) -> tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time]

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.ParticipantBuiltinTopicData, 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.**Partition**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Partition, *arg0:* rti.connextdds.Partition) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Partition) -> None

Creates a policy with the default partition.

2. __init__(self: rti.connextdds.Partition, partition: str) -> None

Creates a policy with a single partition with the specified name.

3. __init__(self: rti.connextdds.Partition, partitions: rti.connextdds.StringSeq) -> None

Creates a policy with the partitions specified in a sequence.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Partition, *arg0:* rti.connextdds.Partition) → bool

Test for inequality.

**property name**

The partition names specified in a sequence.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.PersistentJournalKind) → int

**__init__**(*self:* rti.connextdds.PersistentJournalKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.PersistentJournalKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**class** rti.connextdds.**PersistentStorageSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.PersistentStorageSettings, *arg0:* rti.connextdds.PersistentStorageSettings) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.PersistentStorageSettings) → None

Create a PersistentStorageSettings with default settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.PersistentStorageSettings, *arg0:* rti.connextdds.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.connextdds.**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.PersistentSynchronizationKind) → int

**__init__**(*self:* rti.connextdds.PersistentSynchronizationKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.PersistentSynchronizationKind) → int

**__members__ = {'FULL': <PersistentSynchronizationKind.FULL: 1>, 'NORMAL': <PersistentSynchronizationKind.NORMAL: 0>, 'OFF': <PersistentSynchronizationKind.OFF: 2>}**

**__module__ = 'rti.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.PersistentSynchronizationKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**exception** rti.connextdds.**PreconditionNotMetError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**Presentation**

Bases: pybind11_object

**__annotations__ = {}**

**__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.connextdds.Presentation*

Creates a Presentation instance with topic access scope.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.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__**(*self: object*) → str

**property name**

**property value**

**TOPIC = <PresentationAccessScopeKind.TOPIC: 1>**

**__annotations__ = {}**

**__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.connextdds.PresentationAccessScopeKind, arg0: rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.PresentationAccessScopeKind) → *rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind*

Int conversion.

**__le__** (*self:* rti.connextdds.PresentationAccessScopeKind, *arg0:* rti.connextdds.PresentationAccessScopeKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.PresentationAccessScopeKind, *arg0:* rti.connextdds.PresentationAccessScopeKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.PresentationAccessScopeKind, *arg0:* rti.connextdds.PresentationAccessScopeKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.PresentationAccessScopeKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.PrintFormat.PrintFormat) → int

**__init__**(*self:* rti.connextdds.PrintFormat.PrintFormat, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.PrintFormat.PrintFormat, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**TIMESTAMPED = <PrintFormat.TIMESTAMPED: 13586>**

**VERBOSE = <PrintFormat.VERBOSE: 15894>**

**VERBOSE_TIMESTAMPED = <PrintFormat.VERBOSE_TIMESTAMPED: 16150>**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.PrintFormat, *arg0:* rti.connextdds.PrintFormat) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.PrintFormat, *arg0:* rti.connextdds.PrintFormat) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.PrintFormat, *arg0:* rti.connextdds.PrintFormat) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PrintFormat) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.PrintFormat, arg0: rti.connextdds.PrintFormat.PrintFormat) -> None

Copy constructor.

**__int__**(*self:* rti.connextdds.PrintFormat) → *rti.connextdds.PrintFormat.PrintFormat*

Int conversion.

**__le__** (*self:* rti.connextdds.PrintFormat, *arg0:* rti.connextdds.PrintFormat) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.PrintFormat, *arg0:* rti.connextdds.PrintFormat) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.PrintFormat, *arg0:* rti.connextdds.PrintFormat) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.PrintFormat) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__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__** (*self: object*) → str

**property name**

**property value**

**XML = <PrintFormatKind.XML: 1>**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.PrintFormatKind, *arg0:* rti.connextdds.PrintFormatKind) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.PrintFormatKind, *arg0:* rti.connextdds.PrintFormatKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.PrintFormatKind, *arg0:* rti.connextdds.PrintFormatKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PrintFormatKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.PrintFormatKind, arg0: rti.connextdds.PrintFormatKind.PrintFormatKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.PrintFormatKind) → *rti.connextdds.PrintFormatKind.PrintFormatKind*

Int conversion.

**__le__** (*self:* rti.connextdds.PrintFormatKind, *arg0:* rti.connextdds.PrintFormatKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.PrintFormatKind, *arg0:* rti.connextdds.PrintFormatKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.PrintFormatKind, *arg0:* rti.connextdds.PrintFormatKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.PrintFormatKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**PrintFormatProperty**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.PrintFormatProperty, *kind:* rti.connextdds.PrintFormatKind = *PrintFormatKind.DEFAULT*, *is_pretty_print: bool = True*, *is_enum_as_int: bool = False*, *is_include_root_elements: bool = True*) → None

Initalize PrintFormatProperty object.

**__module__ = 'rti.connextdds'**

**default = <rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.**Property**

Bases: pybind11_object

**__annotations__ = {}**

**__contains__** (*self:* rti.connextdds.Property, *arg0: str*) → bool

**__eq__** (*self:* rti.connextdds.Property, *arg0:* rti.connextdds.Property) → bool

Test for equality.

**__getitem__** (*self:* rti.connextdds.Property, *arg0: str*) → str

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Property) -> None

Creates an empty Property container.

2. __init__(self: rti.connextdds.Property, entries: dict, propagate: bool = False) -> None

Creates a Property container with entries from a dictionary of entries

3. __init__(self: rti.connextdds.Property, entries: rti.connextdds.StringPairSeq, propagate: bool = False) -> None

Creates a Property container with entries from a sequence of entries

**__len__** (*self:* rti.connextdds.Property) → int

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Property, *arg0:* rti.connextdds.Property) → bool

Test for inequality.

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Property, arg0: str, arg1: str) -> rti.connextdds.Property
2. __setitem__(self: rti.connextdds.Property, arg0: str, arg1: tuple[str, bool]) -> rti.connextdds.Property

**exists** (*self:* rti.connextdds.Property, *key: str*) → bool

Returns true if a property exists.

**get** (*self:* rti.connextdds.Property, *key: str*) → str

Returns the value of a property identified by a key if it exists.

**get_all** (*self:* rti.connextdds.Property) → *rti.connextdds.StringMap*

Retrieves a copy of all the entries in a map.

**propagate** (*self:* rti.connextdds.Property, *key: str*) → bool

Returns whether the 'propagate' attribute for a property is set or not.

**remove** (*self:* rti.connextdds.Property, *key: str*) → bool

Removes the property identified by a key.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.Property, entry: tuple[str, str], propagate: bool = False) -> rti.connextdds.Property

Adds or assigns a property from a pair of strings.

2. set(self: rti.connextdds.Property, key: str, value: str, propagate: bool = False) -> rti.connextdds.Property

Adds or assigns a property from a key and a value.

3. set(self: rti.connextdds.Property, entries: rti.connextdds.StringPairSeq, propagate: bool = False) -> rti.connextdds.Property

Adds a range of properties.

4. set(self: rti.connextdds.Property, entries: dict, propagate: bool = False) -> None

Adds a range of properties.

**size** (*self:* rti.connextdds.Property) → int

Returns the number of properties.

**try_get** (*self:* rti.connextdds.Property, *key: str*) → Optional[str]

Returns the value of a property identified by a key if it exists.

**class** rti.connextdds.**ProtocolVersion**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ProtocolVersion, *arg0:* rti.connextdds.ProtocolVersion) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ProtocolVersion) -> None

Creates an invalid protocol version.

2. __init__(self: rti.connextdds.ProtocolVersion, major: int, minor: int) -> None

Creates a version with the given major and minor values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ProtocolVersion, *arg0:* rti.connextdds.ProtocolVersion) → bool

Test for inequality.

**current = <rti.connextdds.ProtocolVersion object>**

**property major_version**

The major version number.

**property minor_version**

The minor version number.

**class** rti.connextdds.**PublicationBuiltinTopicData**

Bases: pybind11_object

**class ContentFilter**

Bases: *ContentFilterBase*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilter) → None

**__module__ = 'rti.connextdds'**

**compile** (*self:* rti.connextdds.PublicationBuiltinTopicData.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.connextdds.PublicationBuiltinTopicData.ContentFilter, *compile_data: Optional[object]*, *sample:* rti.connextdds.PublicationBuiltinTopicData, *meta_data:* rti.connextdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

**finalize** (*self:* rti.connextdds.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*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, topic: rti.connextdds.PublicationBuiltinTopicData.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, topic_description: rti.connextdds.PublicationBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

**append_to_expression_parameter** (*self:* rti.connextdds.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.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic*]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

**remove_from_expression_parameter**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, *index: int*, *remove_term: str*) → None

Removes the specified term from the parameter at the provided index.

**set_filter**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.Filter) → None

Set the filter.

**property topic**

Get the underlying Topic.

**class ContentFilteredTopicSeq**

Bases: `pybind11_object`

**__add__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, s: slice) -> rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

**__imul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None
2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → Iterator[*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic*]

**__len__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

**__ne__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → bool

**__rmul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int, arg1: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) -> None
2. __setitem__(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, L: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *i: int*, *x:* rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic

Remove and return the last item

2. pop(self: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, i: int) -> rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *state:* rti.connextdds.DataState) → *rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector*

Select samples with a specified data state.

**take** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector) → list

Take copies of available samples (data and info) based on the Selector settings.

**take_data** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector) → list

Take copies of available valid data based on the Selector settings.

**take_loaned**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector) → *rti.connextdds.PublicationBuiltinTopicData.LoanedSamples*

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataReader) → *rti.connextdds.PublicationBuiltinTopicData.DataReader*

Enter a context for this DataReader, to be cleaned up on exiting context

**__eq__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader) → bool

Test for equality.

**__exit__**(*self:* rti.connextdds.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.connextdds.PublicationBuiltinTopicData.DataReader, topic: rti.connextdds.PublicationBuiltinTopicData.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, 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 the implicit subscriber with specific QoS and a listener.

3. __init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, cft: rti.connextdds.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.AckRe-

sponseData) -> 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.connextdds.PublicationBuiltinTopicData.DataReader, *handle:* rti.connextdds.InstanceHandle) → *rti.connextdds.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.connextdds.PublicationBuiltinTopicData.DataReader, *key_holder:* rti.connextdds.PublicationBuiltinTopicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

**matched_publication_data**(*self:* rti.connextdds.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.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.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

**__annotations__ = {}**

**__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.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class DataReaderSeq**

Bases: `pybind11_object`

**__add__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → *rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.DataReader) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.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[*rti.connextdds.PublicationBuiltinTopicData.DataReader*]

**__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*

**__annotations__ = {}**

**__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.PublicationBuiltinTopicDataTimestampedSeq) -> 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 thecalling 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.connextdds.PublicationBuiltinTopicData.DataWriter, *handle:* rti.connextdds.InstanceHandle) → *rti.connextdds.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.connextdds.PublicationBuiltinTopicData.DataWriter, *key_holder:* rti.connextdds.PublicationBuiltinTopicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

**matched_subscription_data**(*self:* rti.connextdds.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.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.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.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.PublicationBuiltinTopicData.DataWriter, samples: rti.connextdds.PublicationBuiltinTopicDataSeq, 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.PublicationBuiltinTopicData.DataWriter, sample: rti.connextdds.PublicationBuiltinTopicData, params: rti.connextdds.WriteParams) -> object

Write with advanced parameters.

**class DataWriterListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class DataWriterSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → *rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.DataWriter) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, s: slice) -> rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → *rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq*

**__imul__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) -> None
2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → Iterator[*rti.connextdds.PublicationBuiltinTopicData.DataWriter*]

**__len__** (*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq*

**__ne__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → bool

**__rmul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: int, arg1: rti.connextdds.PublicationBuiltinTopicData.DataWriter) -> None
2. __setitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, arg0: slice, arg1: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.DataWriter) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.DataWriter) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, L: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *i: int*, *x:* rti.connextdds.PublicationBuiltinTopicData.DataWriter) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter

Remove and return the last item

2. pop(self: rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, i: int) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.LoanedSample) -> None

Basic constructor

2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.LoanedSample, data: rti.connextdds.PublicationBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None

Construct LoanedSample with data and info.

**__iter__**(*self:* rti.connextdds.PublicationBuiltinTopicData.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

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.PublicationBuiltinTopicData.LoanedSamples) → *rti.connextdds.PublicationBuiltinTopicData.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

**__exit__**(*self:* rti.connextdds.PublicationBuiltinTopicData.LoanedSamples, *arg0: object*, *arg1: object*, *arg2: object*) → None

Exit the context for the loaned samples, returning the resources.

**__getitem__**(*self:* rti.connextdds.PublicationBuiltinTopicData.LoanedSamples, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.LoanedSample*

Access a LoanedSample object in an array-like syntax

**__init__**(*self:* rti.connextdds.PublicationBuiltinTopicData.LoanedSamples) → None

Create an empty LoanedSamples object.

**__iter__**(*self:* rti.connextdds.PublicationBuiltinTopicData.LoanedSamples) → Iterator[*rti.connextdds.PublicationBuiltinTopicData.LoanedSample*]

**__len__**(*self:* rti.connextdds.PublicationBuiltinTopicData.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.PublicationBuiltinTopicData.LoanedSamples) → None

Returns the loan to the DataReader.

**class NoOpDataReaderListener**

Bases: *DataReaderListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class NoOpDataWriterListener**

Bases: *DataWriterListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.InstanceHandle) → None

On instance replaced callback.

**on_liveliness_lost**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

**on_offered_deadline_missed**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted** (*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class NoOpTopicListener**

Bases: *TopicListener*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpTopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic** (*self:* rti.connextdds.PublicationBuiltinTopicData.NoOpTopicListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class Sample**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.Sample, data: rti.connextdds.PublicationBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None

Construct Sample with data and info.

2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.Sample, sample: rti.connextdds.PublicationBuiltinTopicData.Sample) -> None

Copy constructor.

3. __init__(self: rti.connextdds.PublicationBuiltinTopicData.Sample) -> None

Basic constructor

4. __init__(self: rti.connextdds.PublicationBuiltinTopicData.Sample, loaned_sample: rti.connextdds.PublicationBuiltinTopicData.LoanedSample) -> None

Construct a sample with a loaned sample.

**__iter__** (*self:* rti.connextdds.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

**__annotations__ = {}**

**__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[*rti.connextdds.PublicationBuiltinTopicData.LoanedSample*]

Make a sample iterator

**__len__**(*self:* rti.connextdds.PublicationBuiltinTopicData.SharedSamples) → int

Returns the number of samples.

**__module__ = 'rti.connextdds'**

**class Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__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.connextdds.PublicationBuiltinTopicData.Topic, *arg0:* rti.connextdds.PublicationBuiltinTopicData.Topic) → bool

Test for inequality.

**static find**(*participant:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.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.connextdds.PublicationBuiltinTopicData.Topic, *listener:* rti.connextdds.PublicationBuiltinTopicData.TopicListener, *event_mask:* rti.connextdds.StatusMask) → None

Set the listener and event mask.

**class TopicDescription**

Bases: *ITopicDescription*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.PublicationBuiltinTopicData.TopicDescription) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.TopicDescription, topic: rti.connextdds.PublicationBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.TopicDescription, entity: rti.connextdds.IEntity) -> None

Cast a Topic to a TopicDescription.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.PublicationBuiltinTopicData.TopicDescription) → bool

Test for inequality.

**class TopicListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicListener, *arg0:* rti.connextdds.PublicationBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class TopicSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → *rti.connextdds.PublicationBuiltinTopicData.TopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.Topic) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, s: slice) -> rti.connextdds.PublicationBuiltinTopicData.TopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.PublicationBuiltinTopicData.Topic

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → *rti.connextdds.PublicationBuiltinTopicData.TopicSeq*

**__imul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.TopicSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq) -> None
2. __init__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: rti.connextdds.PublicationBuiltinTopicData.TopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → Iterator[*rti.connextdds.PublicationBuiltinTopicData.Topic*]

**__len__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.TopicSeq*

**__ne__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → bool

**__rmul__**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicData.TopicSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: int, arg1: rti.connextdds.PublicationBuiltinTopicData.Topic) -> None
2. __setitem__(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, arg0: slice, arg1: rti.connextdds.PublicationBuiltinTopicData.TopicSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.Topic) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *x:* rti.connextdds.PublicationBuiltinTopicData.Topic) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, L: rti.connextdds.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`

**__annotations__ = {}**

**__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[*rti.connextdds.PublicationBuiltinTopicData.LoanedSample*]

**__module__ = 'rti.connextdds'**

**class WriterContentFilter**

Bases: *ContentFilter*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.PublicationBuiltinTopicData, *arg0:* rti.connextdds.PublicationBuiltinTopicData) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.PublicationBuiltinTopicData) → None

Create a default PublicationBuiltinTopicData.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.PublicationBuiltinTopicData, *arg0:* rti.connextdds.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 Connext.

**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.connextdds.**PublicationBuiltinTopicDataSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicDataSeq) → *rti.connextdds.PublicationBuiltinTopicDataSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.PublicationBuiltinTopicDataSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *x:* rti.connextdds.PublicationBuiltinTopicData) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicDataSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, s: slice) -> rti.connextdds.PublicationBuiltinTopicDataSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: int) -> rti.connextdds.PublicationBuiltinTopicData

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicDataSeq) → *rti.connextdds.PublicationBuiltinTopicDataSeq*

**__imul__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicDataSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublicationBuiltinTopicDataSeq) -> None

2. __init__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: rti.connextdds.PublicationBuiltinTopicDataSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq) → Iterator[*rti.connextdds.PublicationBuiltinTopicData*]

**__len__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicDataSeq*

**__ne__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0:* rti.connextdds.PublicationBuiltinTopicDataSeq) → bool

**__rmul__** (*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicDataSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: int, arg1: rti.connextdds.PublicationBuiltinTopicData) -> None

2. __setitem__(self: rti.connextdds.PublicationBuiltinTopicDataSeq, arg0: slice, arg1: rti.connextdds.PublicationBuiltinTopicDataSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *x:* rti.connextdds.PublicationBuiltinTopicData) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.PublicationBuiltinTopicDataSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.PublicationBuiltinTopicDataSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__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[tuple[*rti.connextdds.PublicationBuiltinTopicData*, *rti.connextdds.Time*]]

**__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_cxxabi1011__ = <capsule object NULL>**

**__rmul__** (*self:* rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: int, arg1: tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time]) -> None

2. __setitem__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: slice, arg1: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time*]*) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time*]*) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, L: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, *i: int*, *x:* *tuple[*rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time*]*) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) -> tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time]

Remove and return the last item

2. pop(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, i: int) -> tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.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

**__annotations__ = {}**

**__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**

**__annotations__ = {}**

**__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>**

**__annotations__ = {}**

**__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__** (*self: object*) → str

**property name**

**property value**

**SYNCHRONOUS = <PublishModeKind.SYNCHRONOUS: 0>**

**__annotations__ = {}**

**__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.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*

**__annotations__ = {}**

**__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.connextdds.Publisher, *arg0:* rti.connextdds.Publisher) → bool

Test for inequality.

**__rshift__**(*self:* rti.connextdds.Publisher, *arg0:* rti.connextdds.PublisherQos) → *rti.connextdds.Publisher*

Copy the PublisherQos.

**property default_datawriter_qos**

The default DataWriterQos.

This property's getter returns a deep copy.

**find_datawriter**(*self:* rti.connextdds.Publisher, *name: str*) → Optional[*rti.connextdds.AnyDataWriter*]

Find a DataWriter in this Publisher by its name.

**find_datawriter_by_topic_name** (*self:* rti.connextdds.Publisher, *topic_name: str*) → Optional[*rti.connextdds.AnyDataWriter*]

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.connextdds.Publisher) → *rti.connextdds.AnyDataWriterSeq*

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.connextdds.Publisher, *listener:* rti.connextdds.PublisherListener, *event_mask:* rti.connextdds.StatusMask) → None

Bind the listener and event mask to the Publisher.

**wait_for_acknowledgments** (*self:* rti.connextdds.Publisher, *max_wait:* rti.connextdds.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.connextdds.Publisher, *max_wait:* rti.connextdds.Duration) → None

Blocks until asynchronous sending is complete or timeout expires.

**class** rti.connextdds.**PublisherListener**

Bases: *AnyDataWriterListener*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.PublisherListener) → None

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**PublisherQos**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.PublisherQos, *arg0:* rti.connextdds.PublisherQos) → bool

Test for equality

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublisherQos) -> None

Create a PublisherQos with the default value for each policy.

2. __init__(self: rti.connextdds.PublisherQos, publisher: rti.connextdds.Publisher) -> None

Create a PublisherQos with the same settings as those applied to the provided Publisher object.

3. __init__(self: rti.connextdds.PublisherQos, other: rti.connextdds.PublisherQos) -> None

Create a copy of a PublisherQos object.

**__lshift__** (**args*, ***kwargs*)

Overloaded function.

1. __lshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.Presentation) -> rti.connextdds.PublisherQos

Set the PresentationQoS.

2. __lshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.Partition) -> rti.connextdds.PublisherQos

Set the PartitionQoS.

3. __lshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.GroupData) -> rti.connextdds.PublisherQos

Set the GroupDataQoS.

4. __lshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.EntityFactory) -> rti.connextdds.PublisherQos

Set the EntityFactoryQoS.

5. __lshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.AsynchronousPublisher) -> rti.connextdds.PublisherQos

Set the AsynchronousPublisherQoS.

6. __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.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 group_data**

Get/set GroupData QoS.

**property partition**

Get/set Partition QoS.

**property presentation**

Get/set Presentation QoS.

**to_string**(*self:* rti.connextdds.PublisherQos, *format:* rti.connextdds.QosPrintFormat = *QosPrintFormat()*, *base: Optional[*rti.connextdds.PublisherQos*] = None*, *print_all: bool = False*) → str

Convert QoS to string based on params.

**class** rti.connextdds.**PublisherSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.PublisherSeq, *arg0:* rti.connextdds.PublisherSeq) → *rti.connextdds.PublisherSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.PublisherSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.PublisherSeq, *x:* rti.connextdds.Publisher) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublisherSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.PublisherSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.PublisherSeq, *arg0:* rti.connextdds.PublisherSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.PublisherSeq, s: slice) -> rti.connextdds.PublisherSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.PublisherSeq, arg0: int) -> rti.connextdds.Publisher

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.PublisherSeq, *arg0:* rti.connextdds.PublisherSeq) → *rti.connextdds.PublisherSeq*

**__imul__** (*self:* rti.connextdds.PublisherSeq, *arg0: int*) → *rti.connextdds.PublisherSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.PublisherSeq) -> None

2. __init__(self: rti.connextdds.PublisherSeq, arg0: rti.connextdds.PublisherSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.PublisherSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.PublisherSeq) → Iterator[*rti.connextdds.Publisher*]

**__len__** (*self:* rti.connextdds.PublisherSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.PublisherSeq, *arg0: int*) → *rti.connextdds.PublisherSeq*

**__ne__** (*self:* rti.connextdds.PublisherSeq, *arg0:* rti.connextdds.PublisherSeq) → bool

**__rmul__** (*self:* rti.connextdds.PublisherSeq, *arg0: int*) → *rti.connextdds.PublisherSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.PublisherSeq, arg0: int, arg1: rti.connextdds.Publisher) -> None

2. __setitem__(self: rti.connextdds.PublisherSeq, arg0: slice, arg1: rti.connextdds.PublisherSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.PublisherSeq, *x:* rti.connextdds.Publisher) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.PublisherSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.PublisherSeq, *x:* rti.connextdds.Publisher) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.PublisherSeq, L: rti.connextdds.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

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.QosPolicyCountSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.QosPolicyCountSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.QosPolicyCountSeq, *arg0:* rti.connextdds.QosPolicyCountSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.QosPolicyCountSeq, s: slice) -> rti.connextdds.QosPolicyCountSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.QosPolicyCountSeq, arg0: int) -> rti.connextdds.QosPolicyCount

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.QosPolicyCountSeq, *arg0:* rti.connextdds.QosPolicyCountSeq) → *rti.connextdds.QosPolicyCountSeq*

**__imul__**(*self:* rti.connextdds.QosPolicyCountSeq, *arg0: int*) → *rti.connextdds.QosPolicyCountSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.QosPolicyCountSeq) -> None

2. __init__(self: rti.connextdds.QosPolicyCountSeq, arg0: rti.connextdds.QosPolicyCountSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.QosPolicyCountSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.QosPolicyCountSeq) → Iterator[*rti.connextdds.QosPolicyCount*]

**__len__**(*self:* rti.connextdds.QosPolicyCountSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.QosPolicyCountSeq, *arg0: int*) → *rti.connextdds.QosPolicyCountSeq*

**__ne__**(*self:* rti.connextdds.QosPolicyCountSeq, *arg0:* rti.connextdds.QosPolicyCountSeq) → bool

**__rmul__**(*self:* rti.connextdds.QosPolicyCountSeq, *arg0: int*) → *rti.connextdds.QosPolicyCountSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.QosPolicyCountSeq, arg0: int, arg1: rti.connextdds.QosPolicyCount) -> None

2. __setitem__(self: rti.connextdds.QosPolicyCountSeq, arg0: slice, arg1: rti.connextdds.QosPolicyCountSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.QosPolicyCountSeq, *x:* rti.connextdds.QosPolicyCount) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.QosPolicyCountSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.QosPolicyCountSeq, *x:* rti.connextdds.QosPolicyCount) → int

Return the number of times x appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.QosPolicyCountSeq, L: rti.connextdds.QosPolicyCountSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.QosPolicyCountSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.QosPolicyCountSeq, *i: int*, *x:* rti.connextdds.QosPolicyCount) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.QosPolicyCountSeq) -> rti.connextdds.QosPolicyCount

Remove and return the last item

2. pop(self: rti.connextdds.QosPolicyCountSeq, i: int) -> rti.connextdds.QosPolicyCount

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.QosPolicyCountSeq, *x:* rti.connextdds.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.connextdds.**QosPrintFormat**

Bases: `pybind11_object`

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.QosPrintFormat, *arg0:* rti.connextdds.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 preample toggle.

**property print_private**

Print private QoS field toggle.

**class** rti.connextdds.**QosProvider**

Bases: pybind11_object

**__annotations__ = {}**

**__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.

3. __init__(self: rti.connextdds.QosProvider, params: rti.connextdds.QosProviderParams) -> None

Create a QosProvider with the specified parameters.

**__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.connextdds.QosProvider) → None

Load the XML QoS profiles from this QosProvider.

**property participant_factory_qos**

Get a copy of the DomainParticipantFactoryQos currently associated with the QosProvider.

**participant_factory_qos_from_profile** (*self:* rti.connextdds.QosProvider, *profile_name: str*) → *rti.connextdds.DomainParticipantFactoryQos*

Get the DomainParticipantFactoryQos from a qos profile.

**property participant_qos**

Get a copy of the DomainParticipantQos currently associated with the QosProvider.

**participant_qos_from_profile** (*self:* rti.connextdds.QosProvider, *profile_name: str*) → *rti.connextdds.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.connextdds.QosProvider, *profile: str*) → *rti.connextdds.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.connextdds.QosProvider, *library: str*) → *rti.connextdds.StringSeq*

Get a list of the QoS profiles loaded the specified library of the QosProvider.

**reload_profiles** (*self:* rti.connextdds.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.connextdds.QosProvider, *profile_name: str*, *topic_name: str*) → *rti.connextdds.DataReaderQos*

Set the DataReaderQos for a given Topic name.

**set_topic_datawriter_qos** (*self:* rti.connextdds.QosProvider, *profile_name: str*, *topic_name: str*) → *rti.connextdds.DataWriterQos*

Set the DataWriterQos for a given Topic name.

**set_topic_name_qos** (*self:* rti.connextdds.QosProvider, *profile_name: str*, *topic_name: str*) → *rti.connextdds.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.connextdds.QosProvider, *profile: str*) → *rti.connextdds.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.connextdds.QosProvider, *profile_name: str*) → *rti.connextdds.TopicQos*

Get the TopicQos from a qos profile.

**type** (**args*, ***kwargs*)

Overloaded function.

1. type(self: rti.connextdds.QosProvider, library: str, name: str) -> object

Get a DynamicType from a type library in the QosProvider.

2. type(self: rti.connextdds.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.connextdds.QosProvider) → None

Unload the XML QoS profiles from this QosProvider.

**class** rti.connextdds.**QosProviderParams**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.QosProviderParams, *arg0:* rti.connextdds.QosProviderParams) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.QosProviderParams) -> None

Create a QosProviderParams with default settings.

2. __init__(self: rti.connextdds.QosProviderParams, string_profile: rti.connextdds.StringSeq = [], url_profile: rti.connextdds.StringSeq = [], ignore_user_profile: bool = False, ignore_environment_profile: bool = False, ignore_resource_profile: bool = False) -> None

Create a QosProviderParams with the specified settings.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.QosProviderParams, *arg0:* rti.connextdds.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.connextdds.**Query**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Query, *arg0:* rti.connextdds.Query) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Query, reader: rti.connextdds.IDataReader, expression: str) -> None

Init a Query for a DataReader with an expression.

2. __init__(self: rti.connextdds.Query, reader: rti.connextdds.IDataReader, expression: str, params: rti.connextdds.StringSeq) -> None

Init a Query for a DataReader with an expression and parameters.

**__iter__**(*self:* rti.connextdds.Query) → Iterator[str]

**__len__**(*self:* rti.connextdds.Query) → int

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Query, *arg0:* rti.connextdds.Query) → bool

Test for inequality.

**add_parameter**(*self:* rti.connextdds.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.connextdds.**QueryCondition**

Bases: *IReadCondition*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.QueryCondition, *arg0:* rti.connextdds.QueryCondition) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.QueryCondition, query: rti.connextdds.Query, status: rti.connextdds.DataState) -> None

Create a QueryCondition.

2. __init__(self: rti.connextdds.QueryCondition, query: rti.connextdds.Query, status: rti.connextdds.DataState, handler: Callable[[rti.connextdds.QueryCondition], None]) -> None

Create a QueryCondition.

3. __init__(self: rti.connextdds.QueryCondition, reader: rti.connextdds.Query, status: rti.connextdds.DataStateEx) -> None

Create a QueryCondition extended DataState.

4. __init__(self: rti.connextdds.QueryCondition, query: rti.connextdds.Query, status: rti.connextdds.DataStateEx, handler: Callable[[rti.connextdds.QueryCondition], None]) -> None

Create a QueryCondition with extended DataState and a query.

5. __init__(self: rti.connextdds.QueryCondition, condition: rti.connextdds.ICondition) -> None

Cast a condition to a QueryCondition.

**__len__**(*self:* rti.connextdds.QueryCondition) → int

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.QueryCondition, *arg0:* rti.connextdds.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.connextdds.QueryCondition) → None

Resets the handler for this QueryCondition.

**set_handler**(*self:* rti.connextdds.QueryCondition, *func:* *Callable[[*rti.connextdds.QueryCondition*], None]*) → None

Set a handler function for this QueryCondition.

**set_handler_no_args**(*self:* rti.connextdds.QueryCondition, *func: Callable[[], None]*) → None

Set a handler function receiving no arguments.

**class** rti.connextdds.**Rank**

Bases: pybind11_object

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.DataStateEx) -> None

Create a ReadCondition extended DataState.

3. __init__(self: rti.connextdds.ReadCondition, reader: rti.connextdds.IAnyDataReader, status: rti.connextdds.DataState, handler: Callable[[rti.connextdds.ReadCondition], None]) -> None

Create a ReadCondition with a callback.

4. __init__(self: rti.connextdds.ReadCondition, reader: rti.connextdds.IAnyDataReader, status: rti.connextdds.DataStateEx, handler: Callable[[rti.connextdds.ReadCondition], None]) -> None

Create a ReadCondition with extended DataState and a callback.

5. __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

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.ReaderDataLifecycle, *arg0:* rti.connextdds.ReaderDataLifecycle) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ReaderDataLifecycle) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.ReaderDataLifecycle, nowriter_delay: rti.connextdds.Duration, disposed_samples_delay: rti.connextdds.Duration) -> None

Creates an instance with the specified nowriter and disposed-samples purge delays.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ReaderDataLifecycle, *arg0:* rti.connextdds.ReaderDataLifecycle) → bool

Test for inequality.

**static auto_purge_disposed_samples** (*delay:* rti.connextdds.Duration) → *rti.connextdds.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.connextdds.Duration) → *rti.connextdds.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.connextdds.ReaderDataLifecycle object>**

**class** rti.connextdds.**ReceiverPool**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ReceiverPool, *arg0:* rti.connextdds.ReceiverPool) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ReceiverPool) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.ReceiverPool, thread: rti.connextdds.ThreadSettings, buffer_size: int, buffer_alignment: int) -> None

Creates an instance with the thread settings, buffer size and buffer alignment configuration.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ReceiverPool, *arg0:* rti.connextdds.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.connextdds.**Reliability**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Reliability, *arg0:* rti.connextdds.Reliability) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Reliability) -> None

Creates a best-effort policy.

2. __init__(self: rti.connextdds.Reliability, kind: rti.connextdds.ReliabilityKind, max_blocking_time: rti.connextdds.Duration = Duration.from_milliseconds(100)) -> None

Creates an instance with the specified reliability kind and optionally a specific maximum blocking time.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Reliability, *arg0:* rti.connextdds.Reliability) → bool

Test for inequality.

**property acknowledgment_kind**

The kind of reliable acknowledgment.

**best_effort = <rti.connextdds.Reliability object>**

**property exact_reliability_kind_match**

Whether the reliability kind must match exactly.

**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.connextdds.Duration = *Duration.from_milliseconds(100)*) → *rti.connextdds.Reliability*

Creates a policy with ReliabilityKind.RELIABLE and optionally a max blocking time.

**class** rti.connextdds.**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 Connext will attempt to deliver all samples in its history. Missed samples may be retried.

In steady-state (no modifications communicated via the DataWriter), RTI Connext 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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__** (*self: object*) → int

**__index__** (*self:* rti.connextdds.ReliabilityKind.ReliabilityKind) → int

**__init__** (*self:* rti.connextdds.ReliabilityKind.ReliabilityKind, *value: int*) → None

**__int__** (*self:* rti.connextdds.ReliabilityKind.ReliabilityKind) → int

**__members__ = {'BEST_EFFORT': <ReliabilityKind.BEST_EFFORT: 0>, 'RELIABLE': <ReliabilityKind.RELIABLE: 1>}**

**__module__ = 'rti.connextdds'**

**__ne__** (*self: object*, *other: object*) → bool

**__repr__** (*self: object*) → str

**__setstate__** (*self:* rti.connextdds.ReliabilityKind.ReliabilityKind, *state: int*) → None

**__str__** (*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ReliabilityKind, *arg0:* rti.connextdds.ReliabilityKind) → bool
Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.ReliabilityKind, *arg0:* rti.connextdds.ReliabilityKind) → bool
Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.ReliabilityKind, *arg0:* rti.connextdds.ReliabilityKind) → bool
Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)
Overloaded function.

1. __init__(self: rti.connextdds.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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind) → int

**__init__**(*self:* rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind) → int

**__members__ = {'LIVELINESS_BASED':**
**<RemoteParticipantPurgeKind.LIVELINESS_BASED: 0>, 'NO_PURGE':**
**<RemoteParticipantPurgeKind.NO_PURGE: 1>}**

**__module__ = 'rti.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.RemoteParticipantPurgeKind, *arg0:* rti.connextdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.RemoteParticipantPurgeKind, *arg0:* rti.connextdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.RemoteParticipantPurgeKind, *arg0:* rti.connextdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.RemoteParticipantPurgeKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.RemoteParticipantPurgeKind, arg0: rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.RemoteParticipantPurgeKind) → *rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind*

Int conversion.

**__le__** (*self:* rti.connextdds.RemoteParticipantPurgeKind, *arg0:* rti.connextdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.RemoteParticipantPurgeKind, *arg0:* rti.connextdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.RemoteParticipantPurgeKind, *arg0:* rti.connextdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.RemoteParticipantPurgeKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**RequestedDeadlineMissedStatus**

Bases: pybind11_object

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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 or data fragments.

**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 or NACK_FRAG.

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 or NACK_FRAG.

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/NACK_FRAGs 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.connextdds.**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.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → *rti.connextdds.RtpsReservedPortKindMask*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.RtpsReservedPortKindMask) → bool

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → bool

**__eq__**(**args*, ***kwargs*)

Overloaded function.

1. __eq__(self: rti.connextdds.RtpsReservedPortKindMask, arg0: rti.connextdds.RtpsReservedPortKindMask) -> bool

Compare masks for equality.

2. __eq__(self: rti.connextdds.RtpsReservedPortKindMask, arg0: rti.connextdds.RtpsReservedPortKindMask) -> bool

Test for equality.

**__getitem__**(*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → *rti.connextdds.RtpsReservedPortKindMask*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0: int*) → *rti.connextdds.RtpsReservedPortKindMask*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.RtpsReservedPortKindMask) -> None

Create a RtpsReservedPortKindMask equivalent to RtpsReservedPortKindMask.NONE

2. __init__(self: rti.connextdds.RtpsReservedPortKindMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__**(*self:* rti.connextdds.RtpsReservedPortKindMask) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → *rti.connextdds.RtpsReservedPortKindMask*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0: int*) → *rti.connextdds.RtpsReservedPortKindMask*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → *rti.connextdds.RtpsReservedPortKindMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0: int*) → *rti.connextdds.RtpsReservedPortKindMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (**args*, ***kwargs*)

Overloaded function.

1. __ne__(self: rti.connextdds.RtpsReservedPortKindMask, arg0: rti.connextdds.RtpsReservedPortKindMask) -> bool

Compare masks for inequality.

2. __ne__(self: rti.connextdds.RtpsReservedPortKindMask, arg0: rti.connextdds.RtpsReservedPortKindMask) -> bool

Test for inequality.

**__or__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → *rti.connextdds.RtpsReservedPortKindMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.RtpsReservedPortKindMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0: int*) → *rti.connextdds.RtpsReservedPortKindMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.RtpsReservedPortKindMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.RtpsReservedPortKindMask, *arg0:* rti.connextdds.RtpsReservedPortKindMask) → *rti.connextdds.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.connextdds.RtpsReservedPortKindMask) -> rti.connextdds.RtpsReservedPortKindMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.RtpsReservedPortKindMask, pos: int) -> rti.connextdds.RtpsReservedPortKindMask

Flip the mask bit at the specified position.

**reset**(**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.RtpsReservedPortKindMask) -> rti.connextdds.RtpsReservedPortKindMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.RtpsReservedPortKindMask, pos: int) -> rti.connextdds.RtpsReservedPortKindMask

Clear the mask bit at the specified position.

**set**(**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.RtpsReservedPortKindMask) -> rti.connextdds.RtpsReservedPortKindMask

Set all bits in the mask.

2. set(self: rti.connextdds.RtpsReservedPortKindMask, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.RtpsReservedPortKindMask, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all**(*self:* rti.connextdds.RtpsReservedPortKindMask) → bool

Test if all bits are set.

**test_any**(*self:* rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.**SampleFlag**

Bases: pybind11_object

**INCOMPLETE_SNAPSHOT_TOPIC_QUERY = <rti.connextdds.SampleFlag object>**

**INTERMEDIATE_REPLY_SEQUENCE = <rti.connextdds.SampleFlag object>**

**INTERMEDIATE_TOPIC_QUERY_SAMPLE = <rti.connextdds.SampleFlag object>**

**LAST_SHARED_READER_QUEUE = <rti.connextdds.SampleFlag object>**

**REDELIVERED = <rti.connextdds.SampleFlag object>**

**REPLICATE = <rti.connextdds.SampleFlag object>**

**__and__**(*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → *rti.connextdds.SampleFlag*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.SampleFlag) → bool

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → bool

**__eq__**(*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.SampleFlag, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → *rti.connextdds.SampleFlag*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.SampleFlag, *arg0: int*) → *rti.connextdds.SampleFlag*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SampleFlag) -> None

Construct an empty SampleFlag with no bits set.

2. __init__(self: rti.connextdds.SampleFlag, value: int) -> None

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.SampleFlag) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → *rti.connextdds.SampleFlag*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.SampleFlag, *arg0: int*) → *rti.connextdds.SampleFlag*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → *rti.connextdds.SampleFlag*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.SampleFlag, *arg0: int*) → *rti.connextdds.SampleFlag*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → *rti.connextdds.SampleFlag*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.SampleFlag, *arg0: int*) → *rti.connextdds.SampleFlag*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.SampleFlag, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.SampleFlag) → str

**__xor__** (*self:* rti.connextdds.SampleFlag, *arg0:* rti.connextdds.SampleFlag) → *rti.connextdds.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.connextdds.SampleFlag) -> rti.connextdds.SampleFlag

Flip all bits in the mask.

2. flip(self: rti.connextdds.SampleFlag, pos: int) -> rti.connextdds.SampleFlag

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.SampleFlag) -> rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.SampleIdentity, *arg0:* rti.connextdds.SampleIdentity) → bool

Test for inequality.

**__str__**(*self:* rti.connextdds.SequenceNumber) → str

**automatic = <rti.connextdds.SampleIdentity object>**

**property sequence_number**

Get the monotonically increasing 64-bit SequenceNumber that identifies the sample in the data source.

**unknown = <rti.connextdds.SampleIdentity object>**

**property writer_guid**

Get the 16-byte identifier identifying the virtual Guid.

**class** rti.connextdds.**SampleInfo**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.SampleInfo, *arg0:* rti.connextdds.SampleInfo) → bool

Check if two SampleInfo are equal.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.SampleInfo, *arg0:* rti.connextdds.SampleInfo) → bool

Check if two SampleInfo are not equal.

**__repr__**(*self:* rti.connextdds.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.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → *rti.connextdds.SampleLostState*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.SampleLostState) → bool

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → bool

**__eq__**(*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.SampleLostState, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → *rti.connextdds.SampleLostState*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.SampleLostState, *arg0: int*) → *rti.connextdds.SampleLostState*

Left shift bits in mask.

**__init__**(*self:* rti.connextdds.SampleLostState) → None

Creates SampleLostState.NOT_LOST

**__int__**(*self:* rti.connextdds.SampleLostState) → int

Convert mask to int.

**__ior__**(*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → *rti.connextdds.SampleLostState*

Set mask to logical OR with another mask.

**__irshift__**(*self:* rti.connextdds.SampleLostState, *arg0: int*) → *rti.connextdds.SampleLostState*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → *rti.connextdds.SampleLostState*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.SampleLostState, *arg0: int*) → *rti.connextdds.SampleLostState*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → *rti.connextdds.SampleLostState*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.SampleLostState, *arg0: int*) → *rti.connextdds.SampleLostState*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.SampleLostState, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.SampleLostState) → str

**__xor__** (*self:* rti.connextdds.SampleLostState, *arg0:* rti.connextdds.SampleLostState) → *rti.connextdds.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.connextdds.SampleLostState) -> rti.connextdds.SampleLostState

Flip all bits in the mask.

2. flip(self: rti.connextdds.SampleLostState, pos: int) -> rti.connextdds.SampleLostState

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.SampleLostState) -> rti.connextdds.SampleLostState

Clear all bits in the mask.

2. reset(self: rti.connextdds.SampleLostState, pos: int) -> rti.connextdds.SampleLostState

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.SampleLostState) -> rti.connextdds.SampleLostState

Set all bits in the mask.

2. set(self: rti.connextdds.SampleLostState, pos: int, value: bool = True) -> rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → *rti.connextdds.SampleRejectedState*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.SampleRejectedState) → bool

Test if any bits are set.

**__contains__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → bool

**__eq__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → bool

Compare masks for equality.

**__getitem__** (*self:* rti.connextdds.SampleRejectedState, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → *rti.connextdds.SampleRejectedState*

Set mask to logical AND with another mask.

**__ilshift__** (*self:* rti.connextdds.SampleRejectedState, *arg0: int*) → *rti.connextdds.SampleRejectedState*

Left shift bits in mask.

**__init__** (*self:* rti.connextdds.SampleRejectedState) → None

Creates SampleRejectedState.NOT_REJECTED

**__int__** (*self:* rti.connextdds.SampleRejectedState) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → *rti.connextdds.SampleRejectedState*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.SampleRejectedState, *arg0: int*) → *rti.connextdds.SampleRejectedState*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → *rti.connextdds.SampleRejectedState*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.SampleRejectedState, *arg0: int*) → *rti.connextdds.SampleRejectedState*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → *rti.connextdds.SampleRejectedState*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.SampleRejectedState, *arg0: int*) → *rti.connextdds.SampleRejectedState*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.SampleRejectedState, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.SampleRejectedState) → str

**__xor__** (*self:* rti.connextdds.SampleRejectedState, *arg0:* rti.connextdds.SampleRejectedState) → *rti.connextdds.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.connextdds.SampleRejectedState) -> rti.connextdds.SampleRejectedState

Flip all bits in the mask.

2. flip(self: rti.connextdds.SampleRejectedState, pos: int) -> rti.connextdds.SampleRejectedState

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.SampleRejectedState) -> rti.connextdds.SampleRejectedState

Clear all bits in the mask.

2. reset(self: rti.connextdds.SampleRejectedState, pos: int) -> rti.connextdds.SampleRejectedState

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.SampleRejectedState) -> rti.connextdds.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

**__annotations__ = {}**

**__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.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.SampleState) → *rti.connextdds.SampleState*

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → bool

**__eq__**(*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.SampleState, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → *rti.connextdds.SampleState*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.SampleState, *arg0: int*) → *rti.connextdds.SampleState*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SampleState) -> None

Create a SampleState with no bits set.

2. __init__(self: rti.connextdds.SampleState, value: int) -> None

Creates a mask from the bits in an integer.

**__int__**(*self:* rti.connextdds.SampleState) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → *rti.connextdds.SampleState*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.SampleState, *arg0: int*) → *rti.connextdds.SampleState*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → *rti.connextdds.SampleState*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.SampleState, *arg0: int*) → *rti.connextdds.SampleState*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → *rti.connextdds.SampleState*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.SampleState, *arg0: int*) → *rti.connextdds.SampleState*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.SampleState, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.SampleState) → str

**__xor__** (*self:* rti.connextdds.SampleState, *arg0:* rti.connextdds.SampleState) → *rti.connextdds.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.connextdds.SampleState) -> rti.connextdds.SampleState

Flip all bits in the mask.

2. flip(self: rti.connextdds.SampleState, pos: int) -> rti.connextdds.SampleState

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.SampleState) -> rti.connextdds.SampleState

Clear all bits in the mask.

2. reset(self: rti.connextdds.SampleState, pos: int) -> rti.connextdds.SampleState

Clear the mask bit at the specified position.

**set**(**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.SampleState) -> rti.connextdds.SampleState

Set all bits in the mask.

2. set(self: rti.connextdds.SampleState, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.SampleState, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all**(*self:* rti.connextdds.SampleState) → bool

Test if all bits are set.

**test_any**(*self:* rti.connextdds.SampleState) → bool

Test if any bits are set.

**test_none**(*self:* rti.connextdds.SampleState) → bool

Test if none of the bits are set.

**class** rti.connextdds.**SequenceNumber**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → *rti.connextdds.SequenceNumber*

Add two SequenceNumbers.

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → bool

Compare SequenceNumbers for equality.

**__ge__**(*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → bool

Compare two SequenceNumbers for a greater-than-or-equal relationship.

**__gt__**(*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → bool

Compare two SequenceNumbers for a greater-than relationship.

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → *rti.connextdds.SequenceNumber*

Add a SequenceNumber to another

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SequenceNumber) -> None

Create a default SequenceNumber, equal to SequenceNumber.unknown

2. __init__(self: rti.connextdds.SequenceNumber, value: int) -> None

Creates a SequenceNumber from an integer.

**__int__** (*self:* rti.connextdds.SequenceNumber) → int

Convert SequenceNumber to integer.

**__isub__** (*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → *rti.connextdds.SequenceNumber*

Subtract a SequenceNumber from another.

**__le__** (*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → bool

Compare two SequenceNumbers for a less-than-or-equal relationship.

**__lt__** (*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → bool

Compare two SequenceNumbers for a less-than relationship.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → bool

Compare SequenceNumbers for inequality.

**__repr__** (*self:* rti.connextdds.SequenceNumber) → str

Convert SequenceNumber to string.

**__sub__** (*self:* rti.connextdds.SequenceNumber, *arg0:* rti.connextdds.SequenceNumber) → *rti.connextdds.SequenceNumber*

Subtract one SequenceNumber from another.

**automatic = -1**

**maximum = 9223372036854775807**

**unknown = -1**

**property value**

Get/set the SequenceNumber value.

**zero = 0**

**class** rti.connextdds.**SequenceType**

Bases: *UnidimensionalCollectionType*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.SequenceType, *arg0:* rti.connextdds.SequenceType) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SequenceType, data_type: rti.connextdds.DynamicType) -> None

Creates an unbounded collection with an element type.

2. __init__(self: rti.connextdds.SequenceType, data_type: rti.connextdds.DynamicType, bounds: int) -> None

Creates a bounded collection with an element type.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.SequenceType, *arg0:* rti.connextdds.SequenceType) → bool

Test for inequality.

**class** rti.connextdds.**Service**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Service, *arg0:* rti.connextdds.Service) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Service) -> None

Creates the default policy (no service).

2. __init__(self: rti.connextdds.Service, kind: rti.connextdds.ServiceKind) -> None

Creates an instance with the specified service kind.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Service, *arg0:* rti.connextdds.Service) → bool

Test for inequality.

**property kind**

The service kind.

**class** rti.connextdds.**ServiceKind**

Bases: pybind11_object

**ADMIN_CONSOLE = <ServiceKind.ADMIN_CONSOLE: 9>**

**DATABASE_INTEGRATION = <ServiceKind.DATABASE_INTEGRATION: 6>**

**NO_SERVICE = <ServiceKind.NO_SERVICE: 0>**

**OBSERVABILITY_COLLECTOR = <ServiceKind.OBSERVABILITY_COLLECTOR: 8>**

**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.

OBSERVABILITY_COLLECTOR : The Entity is associated to RTI Observability Collector Service.

ADMIN_CONSOLE : The Entity is associated to RTI Admin Console.

**ADMIN_CONSOLE = <ServiceKind.ADMIN_CONSOLE: 9>**

**DATABASE_INTEGRATION = <ServiceKind.DATABASE_INTEGRATION: 6>**

**NO_SERVICE = <ServiceKind.NO_SERVICE: 0>**

**OBSERVABILITY_COLLECTOR = <ServiceKind.OBSERVABILITY_COLLECTOR: 8>**

**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.ServiceKind.ServiceKind) → int

**__init__**(*self:* rti.connextdds.ServiceKind.ServiceKind, *value: int*) → None

**__int__**(*self:* rti.connextdds.ServiceKind.ServiceKind) → int

**__members__ = {'ADMIN_CONSOLE': <ServiceKind.ADMIN_CONSOLE: 9>, 'DATABASE_INTEGRATION': <ServiceKind.DATABASE_INTEGRATION: 6>, 'NO_SERVICE': <ServiceKind.NO_SERVICE: 0>, 'OBSERVABILITY_COLLECTOR': <ServiceKind.OBSERVABILITY_COLLECTOR: 8>, '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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.ServiceKind.ServiceKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**WEB_INTEGRATION = <ServiceKind.WEB_INTEGRATION: 7>**

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.ServiceRequest.ContentFilter) → None

**__module__ = 'rti.connextdds'**

**compile**(*self:* rti.connextdds.ServiceRequest.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.connextdds.ServiceRequest.ContentFilter, *compile_data: Optional[object]*, *sample:* rti.connextdds.ServiceRequest, *meta_data:* rti.connextdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

**finalize**(*self:* rti.connextdds.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*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopic, *arg0:* rti.connextdds.ServiceRequest.ContentFilteredTopic) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequest.ContentFilteredTopic, topic: rti.connextdds.ServiceRequest.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.ServiceRequest.ContentFilteredTopic, topic_description: rti.connextdds.ServiceRequest.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopic, *arg0:* rti.connextdds.ServiceRequest.ContentFilteredTopic) → bool

Test for inequality.

**append_to_expression_parameter**(*self:* rti.connextdds.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*

**__annotations__ = {}**

**__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.connextdds.ServiceRequest.ContentFilteredTopicSeq, s: slice) -> rti.connextdds.ServiceRequest.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.ServiceRequest.ContentFilteredTopic

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → *rti.connextdds.ServiceRequest.ContentFilteredTopicSeq*

**__imul__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.ContentFilteredTopicSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) -> None
2. __init__(self: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, arg0: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → Iterator[*rti.connextdds.ServiceRequest.ContentFilteredTopic*]

**__len__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.ContentFilteredTopicSeq*

**__ne__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → bool

**__rmul__**(*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.ContentFilteredTopicSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int, arg1: rti.connextdds.ServiceRequest.ContentFilteredTopic) -> None
2. __setitem__(self: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *x:* rti.connextdds.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`

**__annotations__ = {}**

**__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.

**__annotations__ = {}**

**__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.connextdds.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.connextdds.ServiceRequest.DataReader, topic: rti.connextdds.ServiceRequest.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. __init__(self: rti.connextdds.ServiceRequest.DataReader, topic: rti.connextdds.ServiceRequest.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.ServiceRequest.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.ServiceRequest.DataReader, cft: rti.connextdds.ServiceRequest.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. __init__(self: rti.connextdds.ServiceRequest.DataReader, cft: rti.connextdds.ServiceRequest.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.ServiceRequest.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.ServiceRequest.DataReader, subscriber: rti.connextdds.Subscriber, topic: rti.connextdds.ServiceRequest.Topic) -> None

Create a DataReader.

6. __init__(self: rti.connextdds.ServiceRequest.DataReader, subscriber: rti.connextdds.Subscriber, topic: rti.connextdds.ServiceRequest.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.ServiceRequest.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.ServiceRequest.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.ServiceRequest.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. __init__(self: rti.connextdds.ServiceRequest.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.ServiceRequest.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.ServiceRequest.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.ServiceRequest.DataReader, reader: rti.connextdds.IAnyDataReader) -> None

Get a typed DataReader from an AnyDataReader.

10. __init__(self: rti.connextdds.ServiceRequest.DataReader, entity: rti.connextdds.IEntity) -> None

Get a typed DataReader from an Entity.

**__lshift__**(*self:* rti.connextdds.ServiceRequest.DataReader, *arg0:* rti.connextdds.DataReaderQos) → *rti.connextdds.ServiceRequest.DataReader*

Set the DataReaderQos for this DataReader.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.ServiceRequest.DataReader, *arg0:* rti.connextdds.ServiceRequest.DataReader) → bool

Test for inequality.

**__rshift__**(*self:* rti.connextdds.ServiceRequest.DataReader, *arg0:* rti.connextdds.DataReaderQos) → *rti.connextdds.ServiceRequest.DataReader*

Get the DataReaderQos from this DataReader

**acknowledge_all**(**args*, ***kwargs*)

Overloaded function.

1. acknowledge_all(self: rti.connextdds.ServiceRequest.DataReader) -> None

Acknowledge all previously accessed samples.

2. acknowledge_all(self: rti.connextdds.ServiceRequest.DataReader, arg0: rti.connextdds.AckResponseData) -> None

Acknowledge all previously accessed samples.

**acknowledge_sample**(**args*, ***kwargs*)

Overloaded function.

1. acknowledge_sample(self: rti.connextdds.ServiceRequest.DataReader, sample_info: rti.connextdds.SampleInfo) -> None

Acknowledge a single sample.

2. acknowledge_sample(self: rti.connextdds.ServiceRequest.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.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.connextdds.Subscriber, *topic_name: str*) → *rti.connextdds.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`

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.ServiceRequest.DataReaderSeq, *arg0:* rti.connextdds.ServiceRequest.DataReaderSeq) → *rti.connextdds.ServiceRequest.DataReaderSeq*

**__imul__**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.DataReaderSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequest.DataReaderSeq) -> None
2. __init__(self: rti.connextdds.ServiceRequest.DataReaderSeq, arg0: rti.connextdds.ServiceRequest.DataReaderSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ServiceRequest.DataReaderSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq) → Iterator[*rti.connextdds.ServiceRequest.DataReader*]

**__len__**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.DataReaderSeq*

**__ne__**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq, *arg0:* rti.connextdds.ServiceRequest.DataReaderSeq) → bool

**__rmul__**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.DataReaderSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.ServiceRequest.DataReaderSeq, arg0: int, arg1: rti.connextdds.ServiceRequest.DataReader) -> None

2. __setitem__(self: rti.connextdds.ServiceRequest.DataReaderSeq, arg0: slice, arg1: rti.connextdds.ServiceRequest.DataReaderSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq, *x:* rti.connextdds.ServiceRequest.DataReader) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.ServiceRequest.DataReaderSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__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.ServiceRequest.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.ServiceRequest.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.ServiceRequest.Topic) -> None

Creates a DataWriter in a publisher with default QoS.

4. __init__(self: rti.connextdds.ServiceRequest.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.ServiceRequest.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.ServiceRequest.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.ServiceRequest.DataWriter, writer: rti.connextdds.IAnyDataWriter) -> None

Create a typed DataWriter from an AnyDataWriter.

6. __init__(self: rti.connextdds.ServiceRequest.DataWriter, entity: rti.connextdds.IEntity) -> None

Create a typed DataWriter from an Entity.

**__lshift__** (**args*, ***kwargs*)

Overloaded function.

1. __lshift__(self: rti.connextdds.ServiceRequest.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.ServiceRequest.DataWriter

Sets the DataWriterQos.

2. __lshift__(self: rti.connextdds.ServiceRequest.DataWriter, arg0: tuple[rti.connextdds.ServiceRequest, rti.connextdds.Time]) -> rti.connextdds.ServiceRequest.DataWriter

Writes a paired sample with a timestamp.

3. __lshift__(self: rti.connextdds.ServiceRequest.DataWriter, arg0: tuple[rti.connextdds.ServiceRequest, rti.connextdds.InstanceHandle]) -> rti.connextdds.ServiceRequest.DataWriter

Writes a paired sample with an instance handle.

4. __lshift__(self: rti.connextdds.ServiceRequest.DataWriter, arg0: rti.connextdds.ServiceRequestTimestampedSeq) -> rti.connextdds.ServiceRequest.DataWriter

Writes a sequence of pairs of samples with timestamps.

5. __lshift__(self: rti.connextdds.ServiceRequest.DataWriter, arg0: rti.connextdds.ServiceRequestSeq) -> rti.connextdds.ServiceRequest.DataWriter

Writes a sequence of samples.

6. __lshift__(self: rti.connextdds.ServiceRequest.DataWriter, arg0: rti.connextdds.ServiceRequest) -> rti.connextdds.ServiceRequest.DataWriter

Writes a sample.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.ServiceRequest.DataWriter, *arg0:* rti.connextdds.ServiceRequest.DataWriter) → bool

Test for inequality.

**__rshift__**(*self:* rti.connextdds.ServiceRequest.DataWriter, *arg0:* rti.connextdds.DataWriterQos) → *rti.connextdds.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.connextdds.Publisher, *topic_name: str*) → *rti.connextdds.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.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.ServiceRequest.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.ServiceRequest.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.ServiceRequest.DataWriter*]

Find DataWriter in publisher with a topic name, returning the first found.

**flush**(*self:* rti.connextdds.ServiceRequest.DataWriter) → None

Flushes the batch in progress in the context of thecalling thread.

**is_matched_subscription_active**(*self:* rti.connextdds.ServiceRequest.DataWriter, *arg0:* rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

**is_sample_app_acknowledged**(*self:* rti.connextdds.ServiceRequest.DataWriter, *sample_id:* rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

**key_value**(*self:* rti.connextdds.ServiceRequest.DataWriter, *handle:* rti.connextdds.InstanceHandle) → *rti.connextdds.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.connextdds.ServiceRequest.DataWriter, *key_holder:* rti.connextdds.ServiceRequest) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

**matched_subscription_data** (*self:* rti.connextdds.ServiceRequest.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.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 Connext 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 Connext 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.con-

nextdds.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`

**__annotations__ = {}**

**__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.connextdds.ServiceRequest.DataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.ServiceRequest.DataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.ServiceRequest.DataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class DataWriterSeq**

Bases: `pybind11_object`

**__add__**(*self:* rti.connextdds.ServiceRequest.DataWriterSeq, *arg0:* rti.connextdds.ServiceRequest.DataWriterSeq) → *rti.connextdds.ServiceRequest.DataWriterSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ServiceRequest.DataWriterSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.ServiceRequest.DataWriterSeq, *x:* rti.connextdds.ServiceRequest.DataWriter) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.ServiceRequest.DataWriterSeq, *arg0:* rti.connextdds.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[*rti.connextdds.ServiceRequest.DataWriter*]

**__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.connextdds.ServiceRequest.DataWriterSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.ServiceRequest.DataWriterSeq, *x:* rti.connextdds.ServiceRequest.DataWriter) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.ServiceRequest.DataWriterSeq, L: rti.connextdds.ServiceRequest.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.ServiceRequest.DataWriterSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.ServiceRequest.DataWriterSeq, *i: int*, *x:* rti.connextdds.ServiceRequest.DataWriter) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.ServiceRequest.DataWriterSeq) -> rti.connextdds.ServiceRequest.DataWriter

Remove and return the last item

2. pop(self: rti.connextdds.ServiceRequest.DataWriterSeq, i: int) -> rti.connextdds.ServiceRequest.DataWriter

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.ServiceRequest.DataWriterSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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[*rti.connextdds.ServiceRequest.LoanedSample*]

**__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*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.ServiceRequest.NoOpDataReaderListener, *arg0:* rti.connextdds.ServiceRequest.DataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class NoOpDataWriterListener**

Bases: *DataWriterListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.InstanceHandle) → None

On instance replaced callback.

**on_liveliness_lost**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

**on_offered_deadline_missed**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.ServiceRequest.NoOpDataWriterListener, *arg0:* rti.connextdds.ServiceRequest.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class NoOpTopicListener**

Bases: *TopicListener*

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequest.Sample, data: rti.connextdds.ServiceRequest, info: rti.connextdds.SampleInfo) -> None

Construct Sample with data and info.

2. __init__(self: rti.connextdds.ServiceRequest.Sample, sample: rti.connextdds.ServiceRequest.Sample) -> None

Copy constructor.

3. __init__(self: rti.connextdds.ServiceRequest.Sample) -> None

Basic constructor

4. __init__(self: rti.connextdds.ServiceRequest.Sample, loaned_sample: rti.connextdds.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

**__annotations__ = {}**

**__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[*rti.connextdds.ServiceRequest.LoanedSample*]

Make a sample iterator

**__len__**(*self:* rti.connextdds.ServiceRequest.SharedSamples) → int

Returns the number of samples.

**__module__ = 'rti.connextdds'**

**class Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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.connextdds.ServiceRequest.TopicSeq, *arg0:* rti.connextdds.ServiceRequest.TopicSeq) → *rti.connextdds.ServiceRequest.TopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ServiceRequest.TopicSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.ServiceRequest.TopicSeq, *x:* rti.connextdds.ServiceRequest.Topic) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ServiceRequest.TopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ServiceRequest.TopicSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.ServiceRequest.TopicSeq, *arg0:* rti.connextdds.ServiceRequest.TopicSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ServiceRequest.TopicSeq, s: slice) -> rti.connextdds.ServiceRequest.TopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ServiceRequest.TopicSeq, arg0: int) -> rti.connextdds.ServiceRequest.Topic

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.ServiceRequest.TopicSeq, *arg0:* rti.connextdds.ServiceRequest.TopicSeq) → *rti.connextdds.ServiceRequest.TopicSeq*

**__imul__** (*self:* rti.connextdds.ServiceRequest.TopicSeq, *arg0: int*) → *rti.connextdds.ServiceRequest.TopicSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequest.TopicSeq) -> None
2. __init__(self: rti.connextdds.ServiceRequest.TopicSeq, arg0: rti.connextdds.ServiceRequest.TopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ServiceRequest.TopicSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.ServiceRequest.TopicSeq) → Iterator[*rti.connextdds.ServiceRequest.Topic*]

**__len__** (*self:* rti.connextdds.ServiceRequest.TopicSeq) → int

**__module__ = 'rti.connextdds'**

**__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

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.ServiceRequest.TopicSeq, L: Iterable) -> None

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

Remove and return the last item

2. pop(self: rti.connextdds.ServiceRequest.TopicSeq, i: int) -> rti.connextdds.ServiceRequest.Topic

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

**__annotations__ = {}**

**__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[*rti.connextdds.ServiceRequest.LoanedSample*]

**__module__ = 'rti.connextdds'**

**class WriterContentFilter**

Bases: *ContentFilter*

**__annotations__ = {}**

**__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.connextdds.ServiceRequest.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.ServiceRequest.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*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.ServiceRequest.WriterContentFilterHelper) → None

**__module__ = 'rti.connextdds'**

**add_cookie** (*self:* rti.connextdds.ServiceRequest.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.ServiceRequest.WriterContentFilterHelper, *writer_filter_data: Optional[object]*, *sample:* rti.connextdds.ServiceRequest, *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.

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ServiceRequest, *arg0:* rti.connextdds.ServiceRequest) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ServiceRequest, *arg0:* rti.connextdds.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.connextdds.**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().

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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.connextdds.**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.ServiceRequestId.ServiceRequestId) → int

**__init__**(*self:* rti.connextdds.ServiceRequestId.ServiceRequestId, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.ServiceRequestId.ServiceRequestId, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**TOPIC_QUERY = <ServiceRequestId.TOPIC_QUERY: 1>**

**UNKNOWN = <ServiceRequestId.UNKNOWN: 0>**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ServiceRequestId, *arg0:* rti.connextdds.ServiceRequestId) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.ServiceRequestId, *arg0:* rti.connextdds.ServiceRequestId) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.ServiceRequestId, *arg0:* rti.connextdds.ServiceRequestId) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequestId) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.ServiceRequestId, arg0: rti.connextdds.ServiceRequestId.ServiceRequestId) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.ServiceRequestId) → *rti.connextdds.ServiceRequestId.ServiceRequestId*

Int conversion.

**__le__** (*self:* rti.connextdds.ServiceRequestId, *arg0:* rti.connextdds.ServiceRequestId) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.ServiceRequestId, *arg0:* rti.connextdds.ServiceRequestId) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ServiceRequestId, *arg0:* rti.connextdds.ServiceRequestId) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.ServiceRequestId) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**ServiceRequestSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0:* rti.connextdds.ServiceRequestSeq) → *rti.connextdds.ServiceRequestSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.ServiceRequestSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.ServiceRequestSeq, *x:* rti.connextdds.ServiceRequest) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ServiceRequestSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ServiceRequestSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0:* rti.connextdds.ServiceRequestSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ServiceRequestSeq, s: slice) -> rti.connextdds.ServiceRequestSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ServiceRequestSeq, arg0: int) -> rti.connextdds.ServiceRequest

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0:* rti.connextdds.ServiceRequestSeq) → *rti.connextdds.ServiceRequestSeq*

**__imul__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0: int*) → *rti.connextdds.ServiceRequestSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequestSeq) -> None

2. __init__(self: rti.connextdds.ServiceRequestSeq, arg0: rti.connextdds.ServiceRequestSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ServiceRequestSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.ServiceRequestSeq) → Iterator[*rti.connextdds.ServiceRequest*]

**__len__** (*self:* rti.connextdds.ServiceRequestSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0: int*) → *rti.connextdds.ServiceRequestSeq*

**__ne__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0:* rti.connextdds.ServiceRequestSeq) → bool

**__rmul__** (*self:* rti.connextdds.ServiceRequestSeq, *arg0: int*) → *rti.connextdds.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*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *x:* *tuple[*rti.connextdds.ServiceRequest*,* rti.connextdds.Time*]*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *arg0:* rti.connextdds.ServiceRequestTimestampedSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.ServiceRequestTimestampedSeq, s: slice) -> rti.connextdds.ServiceRequestTimestampedSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: int) -> tuple[rti.connextdds.ServiceRequest, rti.connextdds.Time]

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *arg0:* rti.connextdds.ServiceRequestTimestampedSeq) → *rti.connextdds.ServiceRequestTimestampedSeq*

**__imul__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *arg0: int*) → *rti.connextdds.ServiceRequestTimestampedSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequestTimestampedSeq) -> None

2. __init__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: rti.connextdds.ServiceRequestTimestampedSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq) → Iterator[tuple[*rti.connextdds.ServiceRequest*, *rti.connextdds.Time*]]

**__len__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *arg0: int*) → *rti.connextdds.ServiceRequestTimestampedSeq*

**__ne__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *arg0:* rti.connextdds.ServiceRequestTimestampedSeq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__rmul__** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *arg0: int*) → *rti.connextdds.ServiceRequestTimestampedSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: int, arg1: tuple[rti.connextdds.ServiceRequest, rti.connextdds.Time]) -> None

2. __setitem__(self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: slice, arg1: rti.connextdds.ServiceRequestTimestampedSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *x:* *tuple[*rti.connextdds.ServiceRequest, rti.connextdds.Time*]*) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.ServiceRequestTimestampedSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.ServiceRequestTimestampedSeq, *x: tuple[*rti.connextdds.ServiceRequest, rti.connextdds.Time*]*) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.ServiceRequestTimestampedSeq, L: rti.connextdds.ServiceRequestTimestampedSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.ServiceRequestTimestampedSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.ServiceRequestTimestampedSeq, *i: int*, *x: tuple[*rti.connextdds.ServiceRequest, rti.connextdds.Time*]*) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.ServiceRequestTimestampedSeq) -> tuple[rti.connextdds.ServiceRequest, rti.connextdds.Time]

Remove and return the last item

2. pop(self: rti.connextdds.ServiceRequestTimestampedSeq, i: int) -> tuple[rti.connextdds.ServiceRequest, rti.connextdds.Time]

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.ServiceRequestTimestampedSeq, *x: tuple[*rti.connextdds.ServiceRequest, 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.

rti.connextdds.**ShortSeq**

alias of *Int16Seq*

rti.connextdds.**ShortType**

alias of *Int16Type*

**class** rti.connextdds.**StatusCondition**

Bases: *ICondition*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.StatusCondition, *arg0:* rti.connextdds.StatusCondition) → bool

Compare StatusCondition objects for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StatusCondition, entity: rti.connextdds.IEntity) -> None

Obtain a reference to an entity's StatusCondition object

2. __init__(self: rti.connextdds.StatusCondition, condition: rti.connextdds.ICondition) -> None

Downcast a Condition to a StatusCondition.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.StatusCondition, *arg0:* rti.connextdds.StatusCondition) → bool

Compare StatusCondition objects for inequality.

**dispatch** (*self:* rti.connextdds.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.connextdds.StatusCondition) → None

Resets the handler for this StatusCondition.

**set_handler** (*self:* rti.connextdds.StatusCondition, *func:* *Callable[[*rti.connextdds.StatusCondition*], None]*) → None

Set a handler function for this StatusCondition.

**property trigger_value**

The trigger value of the condition.

**class** rti.connextdds.**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 = 00000000000000000000001000000000**

**DESTINATION_UNREACHABLE = 01000000000000000000000000000000**

**INCONSISTENT_TOPIC = 00000000000000000000000000000001**

**INVALID_LOCAL_ADVANCE_NOTICE = 00000000000100000000000000000000**

**LIVELINESS_CHANGED = 00000000000000000001000000000000**

**LIVELINESS_LOST = 00000000000000000000100000000000**

**NONE = 00000000000000000000000000000000**

**OFFERED_DEADLINE_MISSED = 00000000000000000000000000000010**

**OFFERED_INCOMPATIBLE_QOS = 00000000000000000000000000100000**

**PUBLICATION_MATCHED = 00000000000000000010000000000000**

**RELIABLE_READER_ACTIVITY_CHANGED = 00000010000000000000000000000000**

**RELIABLE_WRITER_CACHE_CHANGED = 00000001000000000000000000000000**

**REQUESTED_DEADLINE_MISSED = 00000000000000000000000000000100**

**REQUESTED_INCOMPATIBLE_QOS = 00000000000000000000000001000000**

**SAMPLE_LOST = 00000000000000000000000010000000**

**SAMPLE_REJECTED = 00000000000000000000000100000000**

**SAMPLE_REMOVED = 10000000000000000000000000000000**

**SERVICE_REQUEST_ACCEPTED = 00000000001000000000000000000000**

**SUBSCRIPTION_MATCHED = 00000000000000000100000000000000**

**__and__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → *rti.connextdds.StatusMask*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.StatusMask) → bool

Test if any bits are set.

**__contains__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → bool

**__eq__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → bool

Compare masks for equality.

**__getitem__** (*self:* rti.connextdds.StatusMask, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → *rti.connextdds.StatusMask*

Set mask to logical AND with another mask.

**__ilshift__** (*self:* rti.connextdds.StatusMask, *arg0: int*) → *rti.connextdds.StatusMask*

Left shift bits in mask.

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StatusMask) -> None

Create a StatusMask equivalent to StatusMask.NONE

2. __init__(self: rti.connextdds.StatusMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.StatusMask) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → *rti.connextdds.StatusMask*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.StatusMask, *arg0: int*) → *rti.connextdds.StatusMask*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → *rti.connextdds.StatusMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.StatusMask, *arg0: int*) → *rti.connextdds.StatusMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → *rti.connextdds.StatusMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.StatusMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.StatusMask, *arg0: int*) → *rti.connextdds.StatusMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.StatusMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.StatusMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.StatusMask, *arg0:* rti.connextdds.StatusMask) → *rti.connextdds.StatusMask*

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.StatusMask) -> rti.connextdds.StatusMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.StatusMask, pos: int) -> rti.connextdds.StatusMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.StatusMask) -> rti.connextdds.StatusMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.StatusMask, pos: int) -> rti.connextdds.StatusMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.StatusMask) -> rti.connextdds.StatusMask

Set all bits in the mask.

2. set(self: rti.connextdds.StatusMask, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.StatusMask, *pos: int*) → bool

Test whether the mask bit at position “pos” is set.

**test_all** (*self:* rti.connextdds.StatusMask) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.StatusMask) → bool

Test if any bits are set.

**test_none**(*self:* rti.connextdds.StatusMask) → bool

Test if none of the bits are set.

**class** rti.connextdds.**StreamKind**

Bases: pybind11_object

**ANY = <rti.connextdds.StreamKind object>**

**LIVE = <rti.connextdds.StreamKind object>**

**TOPIC_QUERY = <rti.connextdds.StreamKind object>**

**__and__**(*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → *rti.connextdds.StreamKind*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.StreamKind) → *rti.connextdds.StreamKind*

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → bool

**__eq__**(*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.StreamKind, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → *rti.connextdds.StreamKind*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.StreamKind, *arg0: int*) → *rti.connextdds.StreamKind*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StreamKind) -> None

Create a StreamKind with nothing enabled.

2. __init__(self: rti.connextdds.StreamKind, value: int) -> None

Creates a mask from the bits in an integer.

**__int__**(*self:* rti.connextdds.StreamKind) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → *rti.connextdds.StreamKind*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.StreamKind, *arg0: int*) → *rti.connextdds.StreamKind*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → *rti.connextdds.StreamKind*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.StreamKind, *arg0: int*) → *rti.connextdds.StreamKind*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → *rti.connextdds.StreamKind*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.StreamKind, *arg0: int*) → *rti.connextdds.StreamKind*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.StreamKind, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.StreamKind) → str

**__xor__** (*self:* rti.connextdds.StreamKind, *arg0:* rti.connextdds.StreamKind) → *rti.connextdds.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.connextdds.StreamKind) -> rti.connextdds.StreamKind

Flip all bits in the mask.

2. flip(self: rti.connextdds.StreamKind, pos: int) -> rti.connextdds.StreamKind

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.StreamKind) -> rti.connextdds.StreamKind

Clear all bits in the mask.

2. reset(self: rti.connextdds.StreamKind, pos: int) -> rti.connextdds.StreamKind

Clear the mask bit at the specified position.

**set**(**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.StreamKind) -> rti.connextdds.StreamKind

Set all bits in the mask.

2. set(self: rti.connextdds.StreamKind, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.StreamKind, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all**(*self:* rti.connextdds.StreamKind) → bool

Test if all bits are set.

**test_any**(*self:* rti.connextdds.StreamKind) → bool

Test if any bits are set.

**test_none**(*self:* rti.connextdds.StreamKind) → bool

Test if none of the bits are set.

**class** rti.connextdds.**StringMap**

Bases: pybind11_object

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.StringMap) → bool

Check whether the map is nonempty

**__contains__**(**args*, ***kwargs*)

Overloaded function.

1. __contains__(self: rti.connextdds.StringMap, arg0: str) -> bool

2. __contains__(self: rti.connextdds.StringMap, arg0: object) -> bool

**__delitem__**(*self:* rti.connextdds.StringMap, *arg0: str*) → None

**__getitem__**(*self:* rti.connextdds.StringMap, *arg0: str*) → str

**__init__**(*self:* rti.connextdds.StringMap) → None

**__iter__**(*self:* rti.connextdds.StringMap) → Iterator[str]

**__len__** (*self:* rti.connextdds.StringMap) → int

**__module__ = 'rti.connextdds'**

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.StringMap) → str

Return the canonical string representation of this map.

**__setitem__** (*self:* rti.connextdds.StringMap, *arg0: str*, *arg1: str*) → None

**items** (*self:* rti.connextdds.StringMap) → *rti.connextdds.ItemsView*

**keys** (*self:* rti.connextdds.StringMap) → *rti.connextdds.KeysView*

**values** (*self:* rti.connextdds.StringMap) → *rti.connextdds.ValuesView*

**class** rti.connextdds.**StringPairSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.StringPairSeq, *arg0:* rti.connextdds.StringPairSeq) → *rti.connextdds.StringPairSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.StringPairSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.StringPairSeq, *x: tuple[str, str]*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.StringPairSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.StringPairSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.StringPairSeq, *arg0:* rti.connextdds.StringPairSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.StringPairSeq, s: slice) -> rti.connextdds.StringPairSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.StringPairSeq, arg0: int) -> tuple[str, str]

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.StringPairSeq, *arg0:* rti.connextdds.StringPairSeq) → *rti.connextdds.StringPairSeq*

**__imul__** (*self:* rti.connextdds.StringPairSeq, *arg0: int*) → *rti.connextdds.StringPairSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StringPairSeq) -> None
2. __init__(self: rti.connextdds.StringPairSeq, arg0: rti.connextdds.StringPairSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.StringPairSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.StringPairSeq) → Iterator[tuple[str, str]]

**__len__** (*self:* rti.connextdds.StringPairSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.StringPairSeq, *arg0: int*) → *rti.connextdds.StringPairSeq*

**__ne__** (*self:* rti.connextdds.StringPairSeq, *arg0:* rti.connextdds.StringPairSeq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <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.connextdds.StringSeq, *arg0:* rti.connextdds.StringSeq) → *rti.connextdds.StringSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.StringSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.StringSeq, *x: str*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.StringSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.StringSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.StringSeq, *arg0:* rti.connextdds.StringSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.StringSeq, s: slice) -> rti.connextdds.StringSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.StringSeq, arg0: int) -> str

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.StringSeq, *arg0:* rti.connextdds.StringSeq) → *rti.connextdds.StringSeq*

**__imul__**(*self:* rti.connextdds.StringSeq, *arg0: int*) → *rti.connextdds.StringSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StringSeq) -> None
2. __init__(self: rti.connextdds.StringSeq, arg0: rti.connextdds.StringSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.StringSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.StringSeq) → Iterator[str]

**__len__**(*self:* rti.connextdds.StringSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.StringSeq, *arg0: int*) → *rti.connextdds.StringSeq*

**__ne__**(*self:* rti.connextdds.StringSeq, *arg0:* rti.connextdds.StringSeq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__**(*self:* rti.connextdds.StringSeq) → str

Return the canonical string representation of this list.

**__rmul__**(*self:* rti.connextdds.StringSeq, *arg0: int*) → *rti.connextdds.StringSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.StringSeq, arg0: int, arg1: str) -> None
2. __setitem__(self: rti.connextdds.StringSeq, arg0: slice, arg1: rti.connextdds.StringSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.StringSeq, *x: str*) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.StringSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.StringSeq, *x: str*) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.StringSeq, L: rti.connextdds.StringSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.StringSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.StringSeq, *i: int*, *x: str*) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.StringSeq) -> str

Remove and return the last item

2. pop(self: rti.connextdds.StringSeq, i: int) -> str

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.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.connextdds.**StringType**

Bases: *UnidimensionalCollectionType*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.StringType, *arg0:* rti.connextdds.StringType) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StringType, bounds: int) -> None

Creates a bounded string.

2. __init__(self: rti.connextdds.StringType) -> None

Creates an unbounded string.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.StringType, *arg0:* rti.connextdds.StringType) → bool

Test for inequality.

**class** rti.connextdds.**StructType**

Bases: *ACTMember*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.StructType, *arg0:* rti.connextdds.StructType) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.StructType, name: str) -> None

Creates an empty StructType.

2. __init__(self: rti.connextdds.StructType, name: str, extensibility: rti.connextdds.ExtensibilityKind) -> None

Creates an empty StructType with a specified extensibility.

3. __init__(self: rti.connextdds.StructType, name: str, parent: rti.connextdds.StructType) -> None

Create an empty StructType with a base type.

4. __init__(self: rti.connextdds.StructType, name: str, parent: rti.connextdds.StructType, extensibility: rti.connextdds.ExtensibilityKind) -> None

Creates an empty StructType with a base type and specified extensibility.

5. __init__(self: rti.connextdds.StructType, name: str, members: rti.connextdds.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 specfic ID.

**property has_parent**

Indicates if this type has a base type.

**property parent**

Retrieve the base type.

**class** rti.connextdds.**Subscriber**

Bases: *IEntity*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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.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.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 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*

**__annotations__ = {}**

**__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[*rti.connextdds.Subscriber*]

**__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.connextdds.SubscriberSeq, *x:* rti.connextdds.Subscriber) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.SubscriberSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.SubscriberSeq, *x:* rti.connextdds.Subscriber) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.SubscriberSeq, L: rti.connextdds.SubscriberSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.SubscriberSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.SubscriberSeq, *i: int*, *x:* rti.connextdds.Subscriber) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.SubscriberSeq) -> rti.connextdds.Subscriber

Remove and return the last item

2. pop(self: rti.connextdds.SubscriberSeq, i: int) -> rti.connextdds.Subscriber

Remove and return the item at index i

**remove** (*self:* rti.connextdds.SubscriberSeq, *x:* rti.connextdds.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.connextdds.**SubscriptionBuiltinTopicData**

Bases: pybind11_object

**class ContentFilter**

Bases: *ContentFilterBase*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter) → None

**__module__ = 'rti.connextdds'**

**compile** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.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.connextdds.SubscriptionBuiltinTopicData.ContentFilter, *compile_data: Optional[object]*, *sample:* rti.connextdds.SubscriptionBuiltinTopicData, *meta_data:* rti.connextdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

**finalize** (*self:* rti.connextdds.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*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, topic: rti.connextdds.SubscriptionBuiltinTopicData.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, topic_description: rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

**append_to_expression_parameter**(*self:* rti.connextdds.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.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic*]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

**remove_from_expression_parameter**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *index: int*, *remove_term: str*) → None

Removes the specified term from the parameter at the provided index.

**set_filter**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.Filter) → None

Set the filter.

**property topic**

Get the underlying Topic.

**class ContentFilteredTopicSeq**

Bases: `pybind11_object`

**__add__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, s: slice) -> rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq*

**__imul__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) -> None
2. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → Iterator[*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic*]

**__len__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq*

**__ne__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → bool

**__rmul__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: int, arg1: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) -> None
2. __setitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, L: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *i: int*, *x:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic

Remove and return the last item

2. pop(self: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, i: int) -> rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *datareader:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

**__module__ = 'rti.connextdds'**

**condition**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *condition:* rti.connextdds.IReadCondition) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select samples based on a ReadCondition.

**content**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *query:* rti.connextdds.Query) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select samples based on a Query.

**instance**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *handle:* rti.connextdds.InstanceHandle) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select a specific instance to read/take.

**max_samples**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *max: int*) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Limit the number of samples read/taken by the Select.

**next_instance**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *previous:* rti.connextdds.InstanceHandle) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select the instance after the specified instance to read/take.

**read**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

**read_data**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

**read_loaned**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

**state**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector, *state:* rti.connextdds.DataState) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select samples with a specified data state.

**take**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector) → list

Take copies of available samples (data and info) based on the Selector settings.

**take_data**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector) → list

Take copies of available valid data based on the Selector settings.

**take_loaned**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples*

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → *rti.connextdds.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.SubscriptionBuiltinTopicData.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.SubscriptionBuiltinTopicData.DataReader, reader: rti.connextdds.IAnyDataReader) -> None

Get a typed DataReader from an AnyDataReader.

10. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, entity: rti.connextdds.IEntity) -> None

Get a typed DataReader from an Entity.

**__lshift__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg0:* rti.connextdds.DataReaderQos) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader*

Set the DataReaderQos for this DataReader.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → bool

Test for inequality.

**__rshift__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg0:* rti.connextdds.DataReaderQos) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReader*

Get the DataReaderQos from this DataReader

**acknowledge_all** (**args*, ***kwargs*)

Overloaded function.

1. acknowledge_all(self: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) -> None

Acknowledge all previously accessed samples.

2. acknowledge_all(self: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, arg0: rti.connextdds.AckResponseData) -> None

Acknowledge all previously accessed samples.

**acknowledge_sample** (**args*, ***kwargs*)

Overloaded function.

1. acknowledge_sample(self: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo) -> None

Acknowledge a single sample.

2. acknowledge_sample(self: rti.connextdds.SubscriptionBuiltinTopicData.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.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`

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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[*rti.connextdds.SubscriptionBuiltinTopicData.DataReader*]

**__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*

**__annotations__ = {}**

**__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.con-

nextdds.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.SubscriptionBuiltinTopicDataTimestampedSeq) -> 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 thecalling 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 Connext 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 Connext 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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.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.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class DataWriterSeq**

Bases: `pybind11_object`

**__add__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → *rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.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[*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter*]

**__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*

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample) -> None

Basic constructor

2. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample, data: rti.connextdds.SubscriptionBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None

Construct LoanedSample with data and info.

**__iter__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.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

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

**__exit__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples, *arg0: object*, *arg1: object*, *arg2: object*) → None

Exit the context for the loaned samples, returning the resources.

**__getitem__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample*

Access a LoanedSample object in an array-like syntax

**__init__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples) → None

Create an empty LoanedSamples object.

**__iter__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples) → Iterator[*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample*]

**__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*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class NoOpTopicListener**

Bases: *TopicListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.NoOpTopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.NoOpTopicListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class Sample**

Bases: pybind11_object

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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[*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample*]

Make a sample iterator

**__len__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples) → int

Returns the number of samples.

**__module__ = 'rti.connextdds'**

**class Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.Topic, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.Topic) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.Topic, entity: rti.connextdds.IEntity) -> None

Cast an Entity to a Topic.

2. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.Topic, topic_description: rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.Topic, topic: rti.connextdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.Topic, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.Topic) → bool

Test for inequality.

**static find** (*participant:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.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.connextdds.SubscriptionBuiltinTopicData.Topic, *listener:* rti.connextdds.SubscriptionBuiltinTopicData.TopicListener, *event_mask:* rti.connextdds.StatusMask) → None

Set the listener and event mask.

**class TopicDescription**

Bases: *ITopicDescription*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription, topic: rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. __init__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription, entity: rti.connextdds.IEntity) -> None

Cast a Topic to a TopicDescription.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription) → bool

Test for inequality.

**class TopicListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicListener, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class TopicSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) → *rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, *x:* rti.connextdds.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[*rti.connextdds.SubscriptionBuiltinTopicData.Topic*]

**__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`

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples) → *rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples*

**__exit__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples, *arg0: object*, *arg1: object*, *arg2: object*) → None

**__init__** (**args*, ***kwargs*)

**__iter__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples) → Iterator[*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample*]

**__module__ = 'rti.connextdds'**

**class WriterContentFilter**

Bases: *ContentFilter*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter) → None

**__module__ = 'rti.connextdds'**

**writer_attach** (*self:* rti.connextdds.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.connextdds.SubscriptionBuiltinTopicData.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.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.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter, *writer_filter_data: Optional[object]*, *sample:* rti.connextdds.SubscriptionBuiltinTopicData, *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.SubscriptionBuiltinTopicData.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.SubscriptionBuiltinTopicData.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*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper) → None

**__module__ = 'rti.connextdds'**

**add_cookie** (*self:* rti.connextdds.SubscriptionBuiltinTopicData.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.SubscriptionBuiltinTopicData.WriterContentFilterHelper, *writer_filter_data: Optional[object]*, *sample:* rti.connextdds.SubscriptionBuiltinTopicData, *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.

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData, *arg0:* rti.connextdds.SubscriptionBuiltinTopicData) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData) → None

Create a default SubscriptionBuiltinTopicData.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.SubscriptionBuiltinTopicData, *arg0:* rti.connextdds.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 Connext.

**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*

**__annotations__ = {}**

**__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[*rti.connextdds.SubscriptionBuiltinTopicData*]

**__len__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicDataSeq*

**__ne__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicDataSeq) → bool

**__rmul__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicDataSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: int, arg1: rti.connextdds.SubscriptionBuiltinTopicData) -> None

2. __setitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: slice, arg1: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *x:* rti.connextdds.SubscriptionBuiltinTopicData) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *x:* rti.connextdds.SubscriptionBuiltinTopicData) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, L: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *i: int*, *x:* rti.connextdds.SubscriptionBuiltinTopicData) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> rti.connextdds.SubscriptionBuiltinTopicData

Remove and return the last item

2. pop(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, i: int) -> rti.connextdds.SubscriptionBuiltinTopicData

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataSeq, *x:* rti.connextdds.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.connextdds.**SubscriptionBuiltinTopicDataTimestampedSeq**

Bases: `pybind11_object`

**__add__** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → *rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time*]*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, s: slice) -> rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: int) -> tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → *rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq*

**__imul__** (*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None

2. __init__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → Iterator[tuple[*rti.connextdds.SubscriptionBuiltinTopicData*, *rti.connextdds.Time*]]

**__len__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq*

**__ne__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__rmul__**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: int, arg1: tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]) -> None

2. __setitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: slice, arg1: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time*]*) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.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

**__annotations__ = {}**

**__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.connextdds.**SuspendedPublication**

Bases: pybind11_object

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.SuspendedPublication) → *rti.connextdds.SuspendedPublication*

Context manage the SuspendedPublication.

**__exit__**(*self:* rti.connextdds.SuspendedPublication, *arg0: object*, *arg1: object*, *arg2: object*) → None

**__init__**(*self:* rti.connextdds.SuspendedPublication, *publisher:* rti.connextdds.Publisher) → None

Suspends the publications of the publisher.

**__module__ = 'rti.connextdds'**

**resume**(*self:* rti.connextdds.SuspendedPublication) → None

Indicates that the application has completed these changes.

**class** rti.connextdds.**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>**

**__annotations__ = {}**

**__eq__** (*self: object*, *other: object*) → bool

**__getstate__** (*self: object*) → int

**__hash__** (*self: object*) → int

**__index__** (*self:* rti.connextdds.SyslogVerbosity) → int

**__init__** (*self:* rti.connextdds.SyslogVerbosity, *value: int*) → None

**__int__** (*self:* rti.connextdds.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.connextdds'**

**__ne__** (*self: object*, *other: object*) → bool

**__repr__** (*self: object*) → str

**__setstate__** (*self:* rti.connextdds.SyslogVerbosity, *state: int*) → None

**__str__** (*self: object*) → str

**property name**

**property value**

**class** rti.connextdds.**SystemResourceLimits**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.SystemResourceLimits, *arg0:* rti.connextdds.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

**__annotations__ = {}**

**__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.This is only required on platforms that do not support automatic thread-specific storage (TSS) cleanup. Refer to the Platform Notes to see if your platform supports automatic TSS cleanup. It will be mentioned in the 'Thread Configuration' sub-section of the platform you are using.

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**ThreadSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__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.ThreadSettingsKindMask, 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 num_uses**

**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>**

**__annotations__ = {}**

**__eq__** (*self: object*, *other: object*) → bool

**__getstate__** (*self: object*) → int

**__hash__** (*self: object*) → int

**__index__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind) → int

**__init__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind, *value: int*) → None

**__int__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind) → int

**__members__ = {'NO_ROTATION': <ThreadSettingsCpuRotationKind.NO_ROTATION: 0>, 'ROUND_ROBIN': <ThreadSettingsCpuRotationKind.ROUND_ROBIN: 1>}**

**__module__ = 'rti.connextdds'**

**__ne__** (*self: object*, *other: object*) → bool

**__repr__** (*self: object*) → str

**__setstate__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind, *state: int*) → None

**__str__** (*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind, *arg0:* rti.connextdds.ThreadSettingsCpuRotationKind) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind, *arg0:* rti.connextdds.ThreadSettingsCpuRotationKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.ThreadSettingsCpuRotationKind, *arg0:* rti.connextdds.ThreadSettingsCpuRotationKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ThreadSettingsCpuRotationKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.ThreadSettingsCpuRotationKind, arg0: rti.connextdds.ThreadSettingsCpuRotationKind.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.

**__annotations__ = {}**

**__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.connextdds.ThreadSettingsKindMask, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0:* rti.connextdds.ThreadSettingsKindMask) → *rti.connextdds.ThreadSettingsKindMask*

Set mask to logical AND with another mask.

**__ilshift__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0: int*) → *rti.connextdds.ThreadSettingsKindMask*

Left shift bits in mask.

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ThreadSettingsKindMask) -> None

Create a ThreadSettingsKindMask with default thread settings.

2. __init__(self: rti.connextdds.ThreadSettingsKindMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.ThreadSettingsKindMask) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0:* rti.connextdds.ThreadSettingsKindMask) → *rti.connextdds.ThreadSettingsKindMask*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0: int*) → *rti.connextdds.ThreadSettingsKindMask*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0:* rti.connextdds.ThreadSettingsKindMask) → *rti.connextdds.ThreadSettingsKindMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0: int*) → *rti.connextdds.ThreadSettingsKindMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0:* rti.connextdds.ThreadSettingsKindMask) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0:* rti.connextdds.ThreadSettingsKindMask) → *rti.connextdds.ThreadSettingsKindMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.ThreadSettingsKindMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0: int*) → *rti.connextdds.ThreadSettingsKindMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.ThreadSettingsKindMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.ThreadSettingsKindMask, *arg0:* rti.connextdds.ThreadSettingsKindMask) → *rti.connextdds.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.connextdds.ThreadSettingsKindMask) -> rti.connextdds.ThreadSettingsKindMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.ThreadSettingsKindMask, pos: int) -> rti.connextdds.ThreadSettingsKindMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.ThreadSettingsKindMask) -> rti.connextdds.ThreadSettingsKindMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.ThreadSettingsKindMask, pos: int) -> rti.connextdds.ThreadSettingsKindMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.ThreadSettingsKindMask) -> rti.connextdds.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.

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Topic, *arg0:* rti.connextdds.Topic) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Topic, entity: rti.connextdds.IEntity) -> None

Cast an Entity to a Topic.

2. __init__(self: rti.connextdds.Topic, topic_description: rti.connextdds.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. __init__(self: rti.connextdds.Topic, topic: rti.connextdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

4. __init__(self: rti.connextdds.Topic, participant: rti.connextdds.DomainParticipant, topic_name: str, type: object, qos: rti.connextdds.TopicQos = None, listener: rti.connextdds.TopicListener = None, mask: rti.connextdds.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.connextdds'**

**__ne__**(*self:* rti.connextdds.Topic, *arg0:* rti.connextdds.Topic) → bool

Test for inequality.

**static find**(*participant:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.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.Topic, *listener:* rti.connextdds.TopicListener, *event_mask:* rti.connextdds.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.connextdds.**TopicBuiltinTopicData**

Bases: pybind11_object

**class ContentFilter**

Bases: *ContentFilterBase*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilter) → None

**__module__ = 'rti.connextdds'**

**compile** (*self:* rti.connextdds.TopicBuiltinTopicData.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.connextdds.TopicBuiltinTopicData.ContentFilter, *compile_data: Optional[object]*, *sample:* rti.connextdds.TopicBuiltinTopicData, *meta_data:* rti.connextdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

**finalize** (*self:* rti.connextdds.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*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, topic: rti.connextdds.TopicBuiltinTopicData.Topic, name: str, contentfilter: rti.connextdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. __init__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, topic_description: rti.connextdds.TopicBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

**append_to_expression_parameter** (*self:* rti.connextdds.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.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic*]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

**remove_from_expression_parameter** (*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, *index: int*, *remove_term: str*) → None

Removes the specified term from the parameter at the provided index.

**set_filter** (*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, *arg0:* rti.connextdds.Filter) → None

Set the filter.

**property topic**

Get the underlying Topic.

**class ContentFilteredTopicSeq**

Bases: `pybind11_object`

**__add__** (*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *x:* rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.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[*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic*]

**__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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *datareader:* rti.connextdds.TopicBuiltinTopicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

**__module__ = 'rti.connextdds'**

**condition**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *condition:* rti.connextdds.IReadCondition) → *rti.connextdds.TopicBuiltinTopicData.DataReader.Selector*

Select samples based on a ReadCondition.

**content**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *query:* rti.connextdds.Query) → *rti.connextdds.TopicBuiltinTopicData.DataReader.Selector*

Select samples based on a Query.

**instance**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *handle:* rti.connextdds.InstanceHandle) → *rti.connextdds.TopicBuiltinTopicData.DataReader.Selector*

Select a specific instance to read/take.

**max_samples**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *max:* *int*) → *rti.connextdds.TopicBuiltinTopicData.DataReader.Selector*

Limit the number of samples read/taken by the Select.

**next_instance**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *previous:* rti.connextdds.InstanceHandle) → *rti.connextdds.TopicBuiltinTopicData.DataReader.Selector*

Select the instance after the specified instance to read/take.

**read**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

**read_data**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

**read_loaned** (*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector) → *rti.connextdds.TopicBuiltinTopicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

**state** (*self:* rti.connextdds.TopicBuiltinTopicData.DataReader.Selector, *state:* rti.connextdds.DataState) → *rti.connextdds.TopicBuiltinTopicData.DataReader.Selector*

Select samples with a specified data state.

**take** (*self:* rti.connextdds.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.

**__annotations__ = {}**

**__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::TopicBuiltinTopicDataImpl>, 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.TopicBuiltinTopicData.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`

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader) → None

Data available callback.

**on_liveliness_changed**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class DataReaderSeq**

Bases: `pybind11_object`

**__add__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → *rti.connextdds.TopicBuiltinTopicData.DataReaderSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.TopicBuiltinTopicData.DataReader) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, s: slice) -> rti.connextdds.TopicBuiltinTopicData.DataReaderSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.DataReader

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → *rti.connextdds.TopicBuiltinTopicData.DataReaderSeq*

**__imul__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.DataReaderSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) -> None
2. __init__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → Iterator[*rti.connextdds.TopicBuiltinTopicData.DataReader*]

**__len__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.DataReaderSeq*

**__ne__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → bool

**__rmul__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.DataReaderSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: int, arg1: rti.connextdds.TopicBuiltinTopicData.DataReader) -> None
2. __setitem__(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, arg0: slice, arg1: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.TopicBuiltinTopicData.DataReader) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *x:* rti.connextdds.TopicBuiltinTopicData.DataReader) → int

Return the number of times `x` appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, L: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *i: int*, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__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 thecalling 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 Connext 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 Connext 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.connextdds.TopicBuiltinTopicData.DataWriter, *listener:* rti.connextdds.TopicBuiltinTopicData.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.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.TopicBuiltinTopicData.DataWriter

Unregister an instance.

2. unregister_instance(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.TopicBuiltinTopicData.DataWriter

Unregister an instance with timestamp.

3. unregister_instance(self: rti.connextdds.TopicBuiltinTopicData.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.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object

Unregister an instance.

2. unregister_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object

Unregister an instance with timestamp.

3. unregister_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, key_holder: rti.connextdds.TopicBuiltinTopicData) -> object

Unregister the instance associated with key_holder.

4. unregister_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, key_holder: rti.connextdds.TopicBuiltinTopicData, timestamp: rti.connextdds.Time) -> object

Unregister the instance associate with key_holder using a timestamp.

5. unregister_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, params: rti.connextdds.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.ASYN-

CHRONOUS 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.con-

nextdds.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

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.InstanceHandle) → None

On instance replaced callback.

**on_liveliness_lost**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

**on_offered_deadline_missed**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

**on_reliable_writer_cache_changed**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

**on_service_request_accepted**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

**class DataWriterSeq**

Bases: `pybind11_object`

**__add__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) → *rti.connextdds.TopicBuiltinTopicData.DataWriterSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.TopicBuiltinTopicData.DataWriter) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, s: slice) -> rti.connextdds.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[*rti.connextdds.TopicBuiltinTopicData.DataWriter*]

**__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.connextdds.TopicBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.TopicBuiltinTopicData.DataWriter) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, L: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, *i: int*, *x:* rti.connextdds.TopicBuiltinTopicData.DataWriter) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> rti.connextdds.TopicBuiltinTopicData.DataWriter

Remove and return the last item

2. pop(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, i: int) -> rti.connextdds.TopicBuiltinTopicData.DataWriter

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, *x:* rti.connextdds.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*

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.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

**__annotations__ = {}**

**__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[*rti.connextdds.TopicBuiltinTopicData.LoanedSample*]

**__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*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener) → None

**__module__ = 'rti.connextdds'**

**on_data_available** (*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader) → None

Data available callback.

**on_liveliness_changed** (*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

**on_requested_deadline_missed**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

**on_requested_incompatible_qos**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

**on_sample_lost**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleLostStatus) → None

Sample lost callback.

**on_sample_rejected**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

**on_subscription_matched**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataReader, *arg1:* rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

**class NoOpDataWriterListener**

Bases: *DataWriterListener*

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener) → None

**__module__ = 'rti.connextdds'**

**on_application_acknowledgment**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

**on_instance_replaced**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.InstanceHandle) → None

On instance replaced callback.

**on_liveliness_lost**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

**on_offered_deadline_missed**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

**on_offered_incompatible_qos**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

**on_publication_matched**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

**on_reliable_reader_activity_changed**(*self:* rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1:* rti.connextdds.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*

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__getitem__**(*self:* rti.connextdds.TopicBuiltinTopicData.SharedSamples, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.LoanedSample*

Get the sample at the specified index.

**__init__**(*self:* rti.connextdds.TopicBuiltinTopicData.SharedSamples, *loaned_samples:* rti.connextdds.TopicBuiltinTopicData.LoanedSamples) → None

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

**__iter__**(*self:* rti.connextdds.TopicBuiltinTopicData.SharedSamples) → Iterator[*rti.connextdds.TopicBuiltinTopicData.LoanedSample*]

Make a sample iterator

**__len__**(*self:* rti.connextdds.TopicBuiltinTopicData.SharedSamples) → int

Returns the number of samples.

**__module__ = 'rti.connextdds'**

**class Topic**

Bases: *ITopicDescription*, *IAnyTopic*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TopicBuiltinTopicData.Topic, *arg0:* rti.connextdds.TopicBuiltinTopicData.Topic) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicData.Topic, entity: rti.connextdds.IEntity) -> None

Cast an Entity to a Topic.

2. __init__(self: rti.connextdds.TopicBuiltinTopicData.Topic, topic_description: rti.connextdds.TopicBuiltinTopicData.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. __init__(self: rti.connextdds.TopicBuiltinTopicData.Topic, topic: rti.connextdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TopicBuiltinTopicData.Topic, *arg0:* rti.connextdds.TopicBuiltinTopicData.Topic) → bool

Test for inequality.

**static find**(*participant:* rti.connextdds.DomainParticipant, *name: str*) → Optional[*rti.connextdds.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.connextdds.TopicBuiltinTopicData.Topic, *listener:* rti.connextdds.TopicBuiltinTopicData.TopicListener, *event_mask:* rti.connextdds.StatusMask) → None

Set the listener and event mask.

**class TopicDescription**

Bases: *ITopicDescription*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TopicBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.TopicBuiltinTopicData.TopicDescription) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicData.TopicDescription, topic: rti.connextdds.TopicBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. __init__(self: rti.connextdds.TopicBuiltinTopicData.TopicDescription, entity: rti.connextdds.IEntity) -> None

Cast a Topic to a TopicDescription.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TopicBuiltinTopicData.TopicDescription, *arg0:* rti.connextdds.TopicBuiltinTopicData.TopicDescription) → bool

Test for inequality.

**class TopicListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicListener, *arg0:* rti.connextdds.TopicBuiltinTopicData.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class TopicSeq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → *rti.connextdds.TopicBuiltinTopicData.TopicSeq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *x:* rti.connextdds.TopicBuiltinTopicData.Topic) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, s: slice) -> rti.connextdds.TopicBuiltinTopicData.TopicSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.Topic

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → *rti.connextdds.TopicBuiltinTopicData.TopicSeq*

**__imul__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.TopicSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq) -> None
2. __init__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: rti.connextdds.TopicBuiltinTopicData.TopicSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → Iterator[*rti.connextdds.TopicBuiltinTopicData.Topic*]

**__len__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.TopicSeq*

**__ne__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → bool

**__rmul__** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicData.TopicSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: int, arg1: rti.connextdds.TopicBuiltinTopicData.Topic) -> None
2. __setitem__(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, arg0: slice, arg1: rti.connextdds.TopicBuiltinTopicData.TopicSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *x:* rti.connextdds.TopicBuiltinTopicData.Topic) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *x:* rti.connextdds.TopicBuiltinTopicData.Topic) → int

Return the number of times x appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, L: rti.connextdds.TopicBuiltinTopicData.TopicSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *i: int*, *x:* rti.connextdds.TopicBuiltinTopicData.Topic) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq) -> rti.connextdds.TopicBuiltinTopicData.Topic

Remove and return the last item

2. pop(self: rti.connextdds.TopicBuiltinTopicData.TopicSeq, i: int) -> rti.connextdds.TopicBuiltinTopicData.Topic

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.TopicBuiltinTopicData.TopicSeq, *x:* rti.connextdds.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`

**__annotations__ = {}**

**__enter__** (*self:* rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples) → *rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples*

**__exit__** (*self:* rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples, *arg0: object*, *arg1: object*, *arg2: object*) → None

**__init__** (**args*, ***kwargs*)

**__iter__** (*self:* rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples) → Iterator[*rti.connextdds.TopicBuiltinTopicData.LoanedSample*]

**__module__ = 'rti.connextdds'**

**class WriterContentFilter**

Bases: *ContentFilter*

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.TopicBuiltinTopicData.WriterContentFilter) → None

**__module__ = 'rti.connextdds'**

**writer_attach** (*self:* rti.connextdds.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.connextdds.TopicBuiltinTopicData.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.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*

**__annotations__ = {}**

**__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.

**__annotations__ = {}**

**__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.connextdds'**

**__ne__** (*self:* rti.connextdds.TopicBuiltinTopicData, *arg0:* rti.connextdds.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.connextdds.**TopicBuiltinTopicDataSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataSeq) → *rti.connextdds.TopicBuiltinTopicDataSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *x:* rti.connextdds.TopicBuiltinTopicData) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TopicBuiltinTopicDataSeq, s: slice) -> rti.connextdds.TopicBuiltinTopicDataSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataSeq) → *rti.connextdds.TopicBuiltinTopicDataSeq*

**__imul__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicDataSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicDataSeq) -> None

2. __init__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: rti.connextdds.TopicBuiltinTopicDataSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq) → Iterator[*rti.connextdds.TopicBuiltinTopicData*]

**__len__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicDataSeq*

**__ne__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataSeq) → bool

**__rmul__**(*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicDataSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: int, arg1: rti.connextdds.TopicBuiltinTopicData) -> None

2. __setitem__(self: rti.connextdds.TopicBuiltinTopicDataSeq, arg0: slice, arg1: rti.connextdds.TopicBuiltinTopicDataSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *x:* rti.connextdds.TopicBuiltinTopicData) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.TopicBuiltinTopicDataSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *x:* rti.connextdds.TopicBuiltinTopicData) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TopicBuiltinTopicDataSeq, L: rti.connextdds.TopicBuiltinTopicDataSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TopicBuiltinTopicDataSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *i: int*, *x:* rti.connextdds.TopicBuiltinTopicData) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.TopicBuiltinTopicDataSeq) -> rti.connextdds.TopicBuiltinTopicData

Remove and return the last item

2. pop(self: rti.connextdds.TopicBuiltinTopicDataSeq, i: int) -> rti.connextdds.TopicBuiltinTopicData

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.TopicBuiltinTopicDataSeq, *x:* rti.connextdds.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.connextdds.**TopicBuiltinTopicDataTimestampedSeq**

Bases: `pybind11_object`

**__add__** (*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → *rti.connextdds.TopicBuiltinTopicDataTimestampedSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *x:* *tuple[*rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time*]*) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, s: slice) -> rti.connextdds.TopicBuiltinTopicDataTimestampedSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: int) -> tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → *rti.connextdds.TopicBuiltinTopicDataTimestampedSeq*

**__imul__**(*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicDataTimestampedSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) -> None

2. __init__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → Iterator[tuple[*rti.connextdds.TopicBuiltinTopicData*, *rti.connextdds.Time*]]

**__len__** (*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.TopicBuiltinTopicDataTimestampedSeq*

**__ne__** (*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__rmul__** (*self:* rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, *arg0: int*) → *rti.connextdds.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.connextdds.**TopicData**

Bases: `pybind11_object`

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TopicData, *arg0:* rti.connextdds.TopicData) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicData) -> None

Create an empty TopicData QoS policy.

2. __init__(self: rti.connextdds.TopicData, data: rti.connextdds.Uint8Seq) -> None

Create a TopicData object from a data sequence.

**__iter__** (*self:* rti.connextdds.TopicData) → Iterator[int]

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TopicData, *arg0:* rti.connextdds.TopicData) → bool

Test for inequality.

**property value**

Get/set a copy of the TopicData value.

**class** rti.connextdds.**TopicDescription**

Bases: *ITopicDescription*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TopicDescription, *arg0:* rti.connextdds.TopicDescription) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicDescription, topic: rti.connextdds.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. __init__(self: rti.connextdds.TopicDescription, entity: rti.connextdds.IEntity) -> None

Cast a Topic to a TopicDescription.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TopicDescription, *arg0:* rti.connextdds.TopicDescription) → bool

Test for inequality.

**class** rti.connextdds.**TopicListener**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(*self:* rti.connextdds.TopicListener) → None

**__module__ = 'rti.connextdds'**

**on_inconsistent_topic**(*self:* rti.connextdds.TopicListener, *arg0:* rti.connextdds.Topic, *arg1:* rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

**class** rti.connextdds.**TopicQos**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TopicQos, *arg0:* rti.connextdds.TopicQos) → bool

Test for equality

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicQos) -> None

Create a TopicQos with the default value for each policy.

2. __init__(self: rti.connextdds.TopicQos, topic: rti.connextdds.IAnyTopic) -> None

Create a TopicQos with settings equivalent to those of the provided Topic.

3. __init__(self: rti.connextdds.TopicQos, other: rti.connextdds.TopicQos) -> None

Create a copy of a TopicQos object.

**__lshift__**(**args*, ***kwargs*)

Overloaded function.

1. __lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.TopicData) -> rti.connextdds.TopicQos

Set the TopicDataQoS.

2. __lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Durability) -> rti.connextdds.TopicQos

Set the DurabilityQoS.

3. __lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.DurabilityService) -> rti.connextdds.TopicQos

Set the DurabilityServiceQoS.

4. __lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Deadline) -> rti.connextdds.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.connextdds.**TopicQuery**

Bases: pybind11_object

**__annotations__ = {}**

**__enter__**(*self:* rti.connextdds.TopicQuery) → *rti.connextdds.TopicQuery*

Enter a context managed block for a TopicQuery.

**__eq__**(*self:* rti.connextdds.TopicQuery, *arg0:* rti.connextdds.TopicQuery) → bool

Compare DataStateEx objects for equality.

**__exit__**(*self:* rti.connextdds.TopicQuery, *arg0: object*, *arg1: object*, *arg2: object*) → None

Exit a context managed block for a TopicQuery.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.TopicQuery, *reader:* rti.connextdds.IAnyDataReader, *selection:* rti.connextdds.TopicQuerySelection) → None

Creates a TopicQuery for a given DataReader.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TopicQuery, *arg0:* rti.connextdds.TopicQuery) → bool

Compare DataStateEx objects for inequality.

**close** (*self:* rti.connextdds.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.connextdds.IAnyDataReader, *guid:* rti.connextdds.Guid) → Optional[*rti.connextdds.TopicQuery*]

Lookup a TopicQuery by its GUID in the reader that createdit

**property guid**

The TopicQuery GUID.

**retain** (*self:* rti.connextdds.TopicQuery) → None

Disable automatic destruction of this TopicQuery.

**static select_all** (*reader:* rti.connextdds.IAnyDataReader) → *rti.connextdds.TopicQuery*

Create a TopicQuery that requests all data.

**unretain** (*self:* rti.connextdds.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

**__annotations__ = {}**

**__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

**__annotations__ = {}**

**__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 = <Duration(sec=1, nanosec=0)>, samples_per_period: int = -1) -> 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

**__annotations__ = {}**

**__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>**

**__annotations__ = {}**

**__eq__** (*self: object*, *other: object*) → bool

**__getstate__** (*self: object*) → int

**__hash__** (*self: object*) → int

**__index__** (*self:* rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind) → int

**__init__** (*self:* rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind, *value: int*) → None

**__int__** (*self:* rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind) → int

**__members__ = {'CONTINUOUS': <TopicQuerySelectionKind.CONTINUOUS: 1>, 'HISTORY_SNAPSHOT': <TopicQuerySelectionKind.HISTORY_SNAPSHOT: 0>}**

**__module__ = 'rti.connextdds'**

**__ne__** (*self: object*, *other: object*) → bool

**__repr__** (*self: object*) → str

**__setstate__** (*self:* rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind, *state: int*) → None

**__str__** (*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TopicQuerySelectionKind, *arg0:* rti.connextdds.TopicQuerySelectionKind) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.TopicQuerySelectionKind, *arg0:* rti.connextdds.TopicQuerySelectionKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.TopicQuerySelectionKind, *arg0:* rti.connextdds.TopicQuerySelectionKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TopicQuerySelectionKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.TopicQuerySelectionKind, arg0: rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.TopicQuerySelectionKind) → *rti.connextdds.TopicQuerySelectionKind.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*]

**__annotations__ = {}**

**__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[*rti.connextdds.Topic*]

**__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.connextdds.Topic*]*, *x:* rti.connextdds.Topic) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: list[rti.connextdds.Topic], L: list[rti.connextdds.Topic]) -> None

Extend the list by appending all the items in the given list

2. extend(self: list[rti.connextdds.Topic], L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self: list[*rti.connextdds.Topic*]*, *i: int*, *x:* rti.connextdds.Topic) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: list[rti.connextdds.Topic]) -> rti.connextdds.Topic

Remove and return the last item

2. pop(self: list[rti.connextdds.Topic], i: int) -> rti.connextdds.Topic

Remove and return the item at index `i`

**remove** (*self: list[*rti.connextdds.Topic*]*, *x:* rti.connextdds.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.connextdds.**TransportBuiltin**

Bases: `pybind11_object`

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TransportBuiltin, *arg0:* rti.connextdds.TransportBuiltin) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportBuiltin) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.TransportBuiltin, mask: rti.connextdds.TransportBuiltinMask) -> None

Creates an instnace with the transport selection that the mask specifies.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportBuiltin, *arg0:* rti.connextdds.TransportBuiltin) → bool

Test for inequality.

**all = <rti.connextdds.TransportBuiltin object>**

**property mask**

The selected transports through a mask.

**none = <rti.connextdds.TransportBuiltin object>**

**shmem = <rti.connextdds.TransportBuiltin object>**

**udpv4 = <rti.connextdds.TransportBuiltin object>**

**udpv4_wan = <rti.connextdds.TransportBuiltin object>**

**udpv6 = <rti.connextdds.TransportBuiltin object>**

**class** rti.connextdds.**TransportBuiltinMask**

Bases: pybind11_object

**ALL = 11111111**

**NONE = 00000000**

**SHMEM = 00000010**

**UDPv4 = 00000001**

**UDPv4_WAN = 00010000**

**UDPv6 = 00001000**

**__and__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → *rti.connextdds.TransportBuiltinMask*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.TransportBuiltinMask) → bool

Test if any bits are set.

**__contains__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → bool

**__eq__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → bool

Compare masks for equality.

**__getitem__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → *rti.connextdds.TransportBuiltinMask*

Set mask to logical AND with another mask.

**__ilshift__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0: int*) → *rti.connextdds.TransportBuiltinMask*

Left shift bits in mask.

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportBuiltinMask) -> None

Create a TransportBuiltinMask equivalent to TransportBuiltinMask.NONE

2. __init__(self: rti.connextdds.TransportBuiltinMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.TransportBuiltinMask) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → *rti.connextdds.TransportBuiltinMask*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0: int*) → *rti.connextdds.TransportBuiltinMask*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → *rti.connextdds.TransportBuiltinMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0: int*) → *rti.connextdds.TransportBuiltinMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → *rti.connextdds.TransportBuiltinMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.TransportBuiltinMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0: int*) → *rti.connextdds.TransportBuiltinMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.TransportBuiltinMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.TransportBuiltinMask, *arg0:* rti.connextdds.TransportBuiltinMask) → *rti.connextdds.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.connextdds.TransportBuiltinMask) -> rti.connextdds.TransportBuiltinMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.TransportBuiltinMask, pos: int) -> rti.connextdds.TransportBuiltinMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.TransportBuiltinMask) -> rti.connextdds.TransportBuiltinMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.TransportBuiltinMask, pos: int) -> rti.connextdds.TransportBuiltinMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.TransportBuiltinMask) -> rti.connextdds.TransportBuiltinMask

Set all bits in the mask.

2. set(self: rti.connextdds.TransportBuiltinMask, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.TransportBuiltinMask, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all** (*self:* rti.connextdds.TransportBuiltinMask) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.TransportClassId.TransportClassId) → int

**__init__**(*self:* rti.connextdds.TransportClassId.TransportClassId, *value: int*) → None

**__int__**(*self:* rti.connextdds.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__**(*self: object*) → 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>**

**__annotations__ = {}**

**__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.connextdds.TransportClassId, *arg0:* rti.connextdds.TransportClassId) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportClassId, *arg0:* rti.connextdds.TransportClassId) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.TransportClassId) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**TransportInfo**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportInfo) -> None

Creates an instance with default values.

2. __init__(self: rti.connextdds.TransportInfo, class_id: rti.connextdds.TransportClassId.TransportClassId, message_size_max: int) -> None

Creates an instance with the specified class id and max size.

**__module__ = 'rti.connextdds'**

**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.connextdds.**TransportInfoSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.TransportInfoSeq, *arg0:* rti.connextdds.TransportInfoSeq) → *rti.connextdds.TransportInfoSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TransportInfoSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TransportInfoSeq, *x:* rti.connextdds.TransportInfo) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TransportInfoSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TransportInfoSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.TransportInfoSeq, *arg0:* rti.connextdds.TransportInfoSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TransportInfoSeq, s: slice) -> rti.connextdds.TransportInfoSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TransportInfoSeq, arg0: int) -> rti.connextdds.TransportInfo

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.TransportInfoSeq, *arg0:* rti.connextdds.TransportInfoSeq) → *rti.connextdds.TransportInfoSeq*

**__imul__**(*self:* rti.connextdds.TransportInfoSeq, *arg0: int*) → *rti.connextdds.TransportInfoSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportInfoSeq) -> None

2. __init__(self: rti.connextdds.TransportInfoSeq, arg0: rti.connextdds.TransportInfoSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TransportInfoSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.TransportInfoSeq) → Iterator[*rti.connextdds.TransportInfo*]

**__len__** (*self:* rti.connextdds.TransportInfoSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.TransportInfoSeq, *arg0: int*) → *rti.connextdds.TransportInfoSeq*

**__ne__** (*self:* rti.connextdds.TransportInfoSeq, *arg0:* rti.connextdds.TransportInfoSeq) → bool

**__rmul__** (*self:* rti.connextdds.TransportInfoSeq, *arg0: int*) → *rti.connextdds.TransportInfoSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TransportInfoSeq, arg0: int, arg1: rti.connextdds.TransportInfo) -> None

2. __setitem__(self: rti.connextdds.TransportInfoSeq, arg0: slice, arg1: rti.connextdds.TransportInfoSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.TransportInfoSeq, *x:* rti.connextdds.TransportInfo) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.TransportInfoSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.TransportInfoSeq, *x:* rti.connextdds.TransportInfo) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TransportInfoSeq, L: rti.connextdds.TransportInfoSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TransportInfoSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.TransportInfoSeq, *i: int*, *x:* rti.connextdds.TransportInfo) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.TransportInfoSeq) -> rti.connextdds.TransportInfo

Remove and return the last item

2. pop(self: rti.connextdds.TransportInfoSeq, i: int) -> rti.connextdds.TransportInfo

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.TransportInfoSeq, *x:* rti.connextdds.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.connextdds.**TransportInfoVector**

Bases: `pybind11_object`

A DDS standard container with functionality similar to a C++ vector.

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TransportInfoVector, *arg0:* rti.connextdds.TransportInfoVector) → bool

Compare TransportInfoVectors for equality.

**__getitem__**(*self:* rti.connextdds.TransportInfoVector, *arg0: int*) → *rti.connextdds.TransportInfo*

Get the value at the specified index.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportInfoVector) -> None

Create an empty TransportInfoVector

2. __init__(self: rti.connextdds.TransportInfoVector, size: int) -> None

Create a TransportInfoVector with a preallocated size.

3. __init__(self: rti.connextdds.TransportInfoVector, vector: rti.connextdds.TransportInfoVector) -> None

Create a copy from another TransportInfoVector.

4. __init__(self: rti.connextdds.TransportInfoVector, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.TransportInfoVector, list: rti.connextdds.TransportInfoSeq) -> None

Create a TransportInfoVector from a list of values.

**__iter__**(*self:* rti.connextdds.TransportInfoVector) → Iterator[*rti.connextdds.TransportInfo*]

Iterate over the contents of the vector.

**__len__**(*self:* rti.connextdds.TransportInfoVector) → int

Get the length of the TransportInfoVector.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportInfoVector, *arg0:* rti.connextdds.TransportInfoVector) → bool

Compare TransportInfoVectors for inequality.

**__setitem__** (*self:* rti.connextdds.TransportInfoVector, *arg0: int*, *arg1:* rti.connextdds.TransportInfo) → None

Set the value at the specified index.

**clear** (*self:* rti.connextdds.TransportInfoVector) → None

Resize TransportInfoVector to 0.

**resize** (*self:* rti.connextdds.TransportInfoVector, *size: int*) → None

Resize TransportInfoVector.

**class** rti.connextdds.**TransportMulticast**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TransportMulticast, *arg0:* rti.connextdds.TransportMulticast) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.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 speicfied 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>**

**__annotations__ = {}**

**__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__**(*self: object*) → str

**property name**

**property value**

**UNICAST = <TransportMulticastKind.UNICAST: 1>**

**__annotations__ = {}**

**__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.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportMulticastKind, *arg0:* rti.connextdds.TransportMulticastKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.TransportMulticastKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**TransportMulticastMapping**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TransportMulticastMapping, *arg0:* rti.connextdds.TransportMulticastMapping) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportMulticastMapping) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.TransportMulticastMapping, mappings: rti.connextdds.MulticastMappingSeq) -> None

Creates an object with with specified mappings.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportMulticastMapping, *arg0:* rti.connextdds.TransportMulticastMapping) → bool

Test for inequality.

**property value**

A sequence of transport multicast mappings.

**class** rti.connextdds.**TransportMulticastMappingFunction**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TransportMulticastMappingFunction, *arg0:* rti.connextdds.TransportMulticastMappingFunction) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.TransportMulticastMappingFunction, *dll: str*, *function_name: str*) → None

Create a mapping function instance.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TransportMulticastMappingFunction, *arg0:* rti.connextdds.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.connextdds.**TransportMulticastSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.TransportMulticastSeq, *arg0:* rti.connextdds.TransportMulticastSeq) → *rti.connextdds.TransportMulticastSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TransportMulticastSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TransportMulticastSeq, *x:* rti.connextdds.TransportMulticast) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TransportMulticastSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TransportMulticastSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.TransportMulticastSeq, *arg0:* rti.connextdds.TransportMulticastSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TransportMulticastSeq, s: slice) -> rti.connextdds.TransportMulticastSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TransportMulticastSeq, arg0: int) -> rti.connextdds.TransportMulticast

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.TransportMulticastSeq, *arg0:* rti.connextdds.TransportMulticastSeq) → *rti.connextdds.TransportMulticastSeq*

**__imul__**(*self:* rti.connextdds.TransportMulticastSeq, *arg0: int*) → *rti.connextdds.TransportMulticastSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportMulticastSeq) -> None

2. __init__(self: rti.connextdds.TransportMulticastSeq, arg0: rti.connextdds.TransportMulticastSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TransportMulticastSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.TransportMulticastSeq) → Iterator[*rti.connextdds.TransportMulticast*]

**__len__** (*self:* rti.connextdds.TransportMulticastSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.TransportMulticastSeq, *arg0: int*) → *rti.connextdds.TransportMulticastSeq*

**__ne__** (*self:* rti.connextdds.TransportMulticastSeq, *arg0:* rti.connextdds.TransportMulticastSeq) → bool

**__rmul__** (*self:* rti.connextdds.TransportMulticastSeq, *arg0: int*) → *rti.connextdds.TransportMulticastSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TransportMulticastSeq, arg0: int, arg1: rti.connextdds.TransportMulticast) -> None

2. __setitem__(self: rti.connextdds.TransportMulticastSeq, arg0: slice, arg1: rti.connextdds.TransportMulticastSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.TransportMulticastSeq, *x:* rti.connextdds.TransportMulticast) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.TransportMulticastSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.TransportMulticastSeq, *x:* rti.connextdds.TransportMulticast) → int

Return the number of times x appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TransportMulticastSeq, L: rti.connextdds.TransportMulticastSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TransportMulticastSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.TransportMulticastSeq, *i: int*, *x:* rti.connextdds.TransportMulticast) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.TransportMulticastSeq) -> rti.connextdds.TransportMulticast

Remove and return the last item

2. pop(self: rti.connextdds.TransportMulticastSeq, i: int) -> rti.connextdds.TransportMulticast

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.TransportMulticastSeq, *x:* rti.connextdds.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.connextdds.**TransportMulticastSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TransportMulticastSettings, *arg0:* rti.connextdds.TransportMulticastSettings) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.TransportMulticastSettings, *transports:* rti.connextdds.StringSeq, *receive_address: str*, *receive_port: int*) → None

Creates an instance with the specified transport aliases, receive address and receive port.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TransportMulticastSettings, *arg0:* rti.connextdds.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.connextdds.**TransportMulticastSettingsSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0:* rti.connextdds.TransportMulticastSettingsSeq) → *rti.connextdds.TransportMulticastSettingsSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TransportMulticastSettingsSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *x:* rti.connextdds.TransportMulticastSettings) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0:* rti.connextdds.TransportMulticastSettingsSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TransportMulticastSettingsSeq, s: slice) -> rti.connextdds.TransportMulticastSettingsSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: int) -> rti.connextdds.TransportMulticastSettings

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0:* rti.connextdds.TransportMulticastSettingsSeq) → *rti.connextdds.TransportMulticastSettingsSeq*

**__imul__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0: int*) → *rti.connextdds.TransportMulticastSettingsSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportMulticastSettingsSeq) -> None

2. __init__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: rti.connextdds.TransportMulticastSettingsSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: Iterable) -> None

**__iter__**(*self:* rti.connextdds.TransportMulticastSettingsSeq) → Iterator[*rti.connextdds.TransportMulticastSettings*]

**__len__**(*self:* rti.connextdds.TransportMulticastSettingsSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0: int*) → *rti.connextdds.TransportMulticastSettingsSeq*

**__ne__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0:* rti.connextdds.TransportMulticastSettingsSeq) → bool

**__rmul__**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *arg0: int*) → *rti.connextdds.TransportMulticastSettingsSeq*

**__setitem__**(**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: int, arg1: rti.connextdds.TransportMulticastSettings) -> None

2. __setitem__(self: rti.connextdds.TransportMulticastSettingsSeq, arg0: slice, arg1: rti.connextdds.TransportMulticastSettingsSeq) -> None

Assign list elements using a slice object

**append**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *x:* rti.connextdds.TransportMulticastSettings) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.TransportMulticastSettingsSeq) → None

Clear the contents

**count**(*self:* rti.connextdds.TransportMulticastSettingsSeq, *x:* rti.connextdds.TransportMulticastSettings) → int

Return the number of times x appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TransportMulticastSettingsSeq, L: rti.connextdds.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`

**__annotations__ = {}**

**__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.connextdds.**TransportSelection**

Bases: `pybind11_object`

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TransportSelection, *arg0:* rti.connextdds.TransportSelection) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportSelection) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.TransportSelection, enabled_transports: rti.connextdds.StringSeq) -> None

Creates an instance with the specified transport aliases.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TransportSelection, *arg0:* rti.connextdds.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.connextdds.**TransportUnicast**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TransportUnicast, *arg0:* rti.connextdds.TransportUnicast) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportUnicast) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.TransportUnicast, settings: rti.connextdds.TransportUnicastSettingsSeq) -> None

Creates an instance with the specified settings.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TransportUnicast, *arg0:* rti.connextdds.TransportUnicast) → bool

Test for inequality.

**property value**

The unicast settings.

This property's getter returns a deep copy.

**class** rti.connextdds.**TransportUnicastSettings**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TransportUnicastSettings, *arg0:* rti.connextdds.TransportUnicastSettings) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportUnicastSettings) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.TransportUnicastSettings, transports: rti.connextdds.StringSeq, receive_port: int = 0) -> None

Creates the default policy.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TransportUnicastSettings, *arg0:* rti.connextdds.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.connextdds.**TransportUnicastSettingsSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0:* rti.connextdds.TransportUnicastSettingsSeq) → *rti.connextdds.TransportUnicastSettingsSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.TransportUnicastSettingsSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.TransportUnicastSettingsSeq, *x:* rti.connextdds.TransportUnicastSettings) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0:* rti.connextdds.TransportUnicastSettingsSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.TransportUnicastSettingsSeq, s: slice) -> rti.connextdds.TransportUnicastSettingsSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: int) -> rti.connextdds.TransportUnicastSettings

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0:* rti.connextdds.TransportUnicastSettingsSeq) → *rti.connextdds.TransportUnicastSettingsSeq*

**__imul__** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0: int*) → *rti.connextdds.TransportUnicastSettingsSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TransportUnicastSettingsSeq) -> None

2. __init__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: rti.connextdds.TransportUnicastSettingsSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.TransportUnicastSettingsSeq) → Iterator[*rti.connextdds.TransportUnicastSettings*]

**__len__** (*self:* rti.connextdds.TransportUnicastSettingsSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0: int*) → *rti.connextdds.TransportUnicastSettingsSeq*

**__ne__** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0:* rti.connextdds.TransportUnicastSettingsSeq) → bool

**__rmul__** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *arg0: int*) → *rti.connextdds.TransportUnicastSettingsSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: int, arg1: rti.connextdds.TransportUnicastSettings) -> None

2. __setitem__(self: rti.connextdds.TransportUnicastSettingsSeq, arg0: slice, arg1: rti.connextdds.TransportUnicastSettingsSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *x:* rti.connextdds.TransportUnicastSettings) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.TransportUnicastSettingsSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *x:* rti.connextdds.TransportUnicastSettings) → int

Return the number of times x appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.TransportUnicastSettingsSeq, L: rti.connextdds.TransportUnicastSettingsSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.TransportUnicastSettingsSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *i: int*, *x:* rti.connextdds.TransportUnicastSettings) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.TransportUnicastSettingsSeq) -> rti.connextdds.TransportUnicastSettings

Remove and return the last item

2. pop(self: rti.connextdds.TransportUnicastSettingsSeq, i: int) -> rti.connextdds.TransportUnicastSettings

Remove and return the item at index i

**remove** (*self:* rti.connextdds.TransportUnicastSettingsSeq, *x:* rti.connextdds.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.connextdds.**TriggeredConditions**

Bases: pybind11_object

**__annotations__ = {}**

**__contains__** (*self:* rti.connextdds.TriggeredConditions, *arg0:* rti.connextdds.ICondition) → bool

**__getitem__** (*self:* rti.connextdds.TriggeredConditions, *arg0: int*) → *rti.connextdds.Condition*

**__init__** (**args*, ***kwargs*)

**__iter__** (*self:* rti.connextdds.TriggeredConditions) → *rti.connextdds.TriggeredConditionsIterator*

**__len__** (*self:* rti.connextdds.TriggeredConditions) → int

**__module__ = 'rti.connextdds'**

**__reversed__** (*self:* rti.connextdds.TriggeredConditions) → *rti.connextdds.TriggeredConditionsIterator*

**class** rti.connextdds.**TriggeredConditionsIterator**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**__next__** (*self:* rti.connextdds.TriggeredConditionsIterator) → *rti.connextdds.Condition*

Get next triggered condition.

**class** rti.connextdds.**TypeConsistencyEnforcement**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TypeConsistencyEnforcement, *arg0:* rti.connextdds.TypeConsistencyEnforcement) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TypeConsistencyEnforcement) -> None

Creates the default policy.

2. __init__(self: rti.connextdds.TypeConsistencyEnforcement, kind: rti.connextdds.TypeConsistencyEnforcementKind) -> None

Creates an instance with a specific enforcement kind.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.TypeConsistencyEnforcement, *arg0:* rti.connextdds.TypeConsistencyEnforcement) → bool

Test for inequality.

**allow_type_coercion = <rti.connextdds.TypeConsistencyEnforcement object>**

**auto_type_coercion = <rti.connextdds.TypeConsistencyEnforcement object>**

**disallow_type_coercion = <rti.connextdds.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.connextdds.**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>**

**__annotations__ = {}**

**__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.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind, *state: int*) →
None

**__str__**(*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.TypeConsistencyEnforcementKind, *arg0:* rti.connextdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.TypeConsistencyEnforcementKind, *arg0:* rti.connextdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.TypeConsistencyEnforcementKind, *arg0:* rti.connextdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TypeConsistencyEnforcementKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.TypeConsistencyEnforcementKind, arg0: rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.TypeConsistencyEnforcementKind) → *rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind*

Int conversion.

**__le__** (*self:* rti.connextdds.TypeConsistencyEnforcementKind, *arg0:* rti.connextdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.TypeConsistencyEnforcementKind, *arg0:* rti.connextdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TypeConsistencyEnforcementKind, *arg0:* rti.connextdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.TypeConsistencyEnforcementKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**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>**

**INT_8_TYPE = <TypeKind.INT_8_TYPE: 16398>**

**MAP_TYPE = <TypeKind.MAP_TYPE: 33287>**

**NO_TYPE = <TypeKind.NO_TYPE: 0>**

**OCTET_TYPE = <TypeKind.OCTET_TYPE: 16399>**

**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.

OCTET_TYPE : Opaque byte.

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.

INT_8_TYPE : Signed 8-bit int 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>**

**INT_8_TYPE = <TypeKind.INT_8_TYPE: 16398>**

**MAP_TYPE = <TypeKind.MAP_TYPE: 33287>**

**NO_TYPE = <TypeKind.NO_TYPE: 0>**

**OCTET_TYPE = <TypeKind.OCTET_TYPE: 16399>**

**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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.TypeKind.TypeKind) → int

**__init__** (*self:* rti.connextdds.TypeKind.TypeKind, *value: int*) → None

**__int__** (*self:* rti.connextdds.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>, 'INT_8_TYPE': <TypeKind.INT_8_TYPE: 16398>, 'MAP_TYPE': <TypeKind.MAP_TYPE: 33287>, 'NO_TYPE': <TypeKind.NO_TYPE: 0>, 'OCTET_TYPE': <TypeKind.OCTET_TYPE: 16399>, '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.connextdds'**

**__ne__** (*self: object*, *other: object*) → bool

**__repr__** (*self: object*) → str

**__setstate__** (*self:* rti.connextdds.TypeKind.TypeKind, *state: int*) → None

**__str__** (*self: object*) → 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>**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TypeKind, *arg0:* rti.connextdds.TypeKind) → bool

Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.TypeKind, *arg0:* rti.connextdds.TypeKind) → bool

Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.TypeKind, *arg0:* rti.connextdds.TypeKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.TypeKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.TypeKind, arg0: rti.connextdds.TypeKind.TypeKind) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.TypeKind) → *rti.connextdds.TypeKind.TypeKind*

Int conversion.

**__le__** (*self:* rti.connextdds.TypeKind, *arg0:* rti.connextdds.TypeKind) → bool

Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.TypeKind, *arg0:* rti.connextdds.TypeKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TypeKind, *arg0:* rti.connextdds.TypeKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.TypeKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**TypeSupport**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.TypeSupport, *arg0:* rti.connextdds.TypeSupport) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.TypeSupport) → None

Creates the default policy.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.TypeSupport, *arg0:* rti.connextdds.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.connextdds.**ULongLongSeq**

alias of *Uint64Seq*

rti.connextdds.**ULongLongType**

alias of *Uint64Type*

rti.connextdds.**ULongSeq**

alias of *Uint32Seq*

rti.connextdds.**ULongType**

alias of *Uint32Type*

rti.connextdds.**UShortType**

alias of *Uint16Type*

**class** rti.connextdds.**Uint16Seq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.Uint16Seq, *arg0:* rti.connextdds.Uint16Seq) → *rti.connextdds.Uint16Seq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.Uint16Seq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.Uint16Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Uint16Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Uint16Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.Uint16Seq, *arg0:* rti.connextdds.Uint16Seq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Uint16Seq, s: slice) -> rti.connextdds.Uint16Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Uint16Seq, arg0: int) -> int

**__getstate__** (*self:* rti.connextdds.Uint16Seq) → bytes

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.Uint16Seq, *arg0:* rti.connextdds.Uint16Seq) → *rti.connextdds.Uint16Seq*

**__imul__** (*self:* rti.connextdds.Uint16Seq, *arg0: int*) → *rti.connextdds.Uint16Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Uint16Seq, arg0: Buffer) -> None

2. __init__(self: rti.connextdds.Uint16Seq) -> None

3. __init__(self: rti.connextdds.Uint16Seq, arg0: rti.connextdds.Uint16Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Uint16Seq, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.Uint16Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Uint16Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Uint16Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Uint16Seq, *arg0: int*) → *rti.connextdds.Uint16Seq*

**__ne__** (*self:* rti.connextdds.Uint16Seq, *arg0:* rti.connextdds.Uint16Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Uint16Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Uint16Seq, *arg0: int*) → *rti.connextdds.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*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Uint16Type, *arg0:* rti.connextdds.Uint16Type) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.Uint16Type) → None

Get the singleton for Uint16Type

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Uint16Type, *arg0:* rti.connextdds.Uint16Type) → bool

Test for inequality.

**class** rti.connextdds.**Uint32Seq**

Bases: pybind11_object

**__add__** (*self:* rti.connextdds.Uint32Seq, *arg0:* rti.connextdds.Uint32Seq) → *rti.connextdds.Uint32Seq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.Uint32Seq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.Uint32Seq, *x: int*) → bool

Return true the container contains x

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Uint32Seq, arg0: int) -> None

Delete the list elements at index i

2. __delitem__(self: rti.connextdds.Uint32Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.Uint32Seq, *arg0:* rti.connextdds.Uint32Seq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Uint32Seq, s: slice) -> rti.connextdds.Uint32Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Uint32Seq, arg0: int) -> int

**__getstate__** (*self:* rti.connextdds.Uint32Seq) → bytes

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.Uint32Seq, *arg0:* rti.connextdds.Uint32Seq) → *rti.connextdds.Uint32Seq*

**__imul__** (*self:* rti.connextdds.Uint32Seq, *arg0: int*) → *rti.connextdds.Uint32Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Uint32Seq, arg0: Buffer) -> None

2. __init__(self: rti.connextdds.Uint32Seq) -> None

3. __init__(self: rti.connextdds.Uint32Seq, arg0: rti.connextdds.Uint32Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Uint32Seq, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.Uint32Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Uint32Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Uint32Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Uint32Seq, *arg0: int*) → *rti.connextdds.Uint32Seq*

**__ne__** (*self:* rti.connextdds.Uint32Seq, *arg0:* rti.connextdds.Uint32Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Uint32Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Uint32Seq, *arg0: int*) → *rti.connextdds.Uint32Seq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Uint32Seq, arg0: int, arg1: int) -> None

2. __setitem__(self: rti.connextdds.Uint32Seq, arg0: slice, arg1: rti.connextdds.Uint32Seq) -> None

Assign list elements using a slice object

**__setstate__**(*self:* rti.connextdds.Uint32Seq, *arg0: bytes*) → None

**append**(*self:* rti.connextdds.Uint32Seq, *x: int*) → None

Add an item to the end of the list

**clear**(*self:* rti.connextdds.Uint32Seq) → None

Clear the contents

**count**(*self:* rti.connextdds.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*

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.Uint64Seq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.Uint64Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Uint64Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Uint64Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.Uint64Seq, *arg0:* rti.connextdds.Uint64Seq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Uint64Seq, s: slice) -> rti.connextdds.Uint64Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Uint64Seq, arg0: int) -> int

**__getstate__**(*self:* rti.connextdds.Uint64Seq) → bytes

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.Uint64Seq, *arg0:* rti.connextdds.Uint64Seq) → *rti.connextdds.Uint64Seq*

**__imul__**(*self:* rti.connextdds.Uint64Seq, *arg0: int*) → *rti.connextdds.Uint64Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Uint64Seq, arg0: Buffer) -> None
2. __init__(self: rti.connextdds.Uint64Seq) -> None
3. __init__(self: rti.connextdds.Uint64Seq, arg0: rti.connextdds.Uint64Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Uint64Seq, arg0: Iterable) -> None
5. __init__(self: rti.connextdds.Uint64Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Uint64Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Uint64Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Uint64Seq, *arg0: int*) → *rti.connextdds.Uint64Seq*

**__ne__** (*self:* rti.connextdds.Uint64Seq, *arg0:* rti.connextdds.Uint64Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Uint64Seq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.Uint64Seq, *arg0: int*) → *rti.connextdds.Uint64Seq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Uint64Seq, arg0: int, arg1: int) -> None
2. __setitem__(self: rti.connextdds.Uint64Seq, arg0: slice, arg1: rti.connextdds.Uint64Seq) -> None

Assign list elements using a slice object

**__setstate__** (*self:* rti.connextdds.Uint64Seq, *arg0: bytes*) → None

**append** (*self:* rti.connextdds.Uint64Seq, *x: int*) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.Uint64Seq) → None

Clear the contents

**count** (*self:* rti.connextdds.Uint64Seq, *x: int*) → int

Return the number of times x appears in the list

**extend**(**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.Uint64Seq, L: rti.connextdds.Uint64Seq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.Uint64Seq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert**(*self:* rti.connextdds.Uint64Seq, *i: int*, *x: int*) → None

Insert an item at a given position.

**pop**(**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.Uint64Seq) -> int

Remove and return the last item

2. pop(self: rti.connextdds.Uint64Seq, i: int) -> int

Remove and return the item at index `i`

**remove**(*self:* rti.connextdds.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.connextdds.Uint64Seq, *arg0: int*) → None

resizes the vector to the given size

**class** rti.connextdds.**Uint64Type**

Bases: *DynamicType*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.Uint64Type, *arg0:* rti.connextdds.Uint64Type) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.Uint64Type) → None

Get the singleton for Uint64Type

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.Uint64Type, *arg0:* rti.connextdds.Uint64Type) → bool

Test for inequality.

**class** rti.connextdds.**Uint8Seq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.Uint8Seq, *arg0:* rti.connextdds.Uint8Seq) → *rti.connextdds.Uint8Seq*

**__annotations__ = {}**

**__bool__** (*self:* rti.connextdds.Uint8Seq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.Uint8Seq, *x: int*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.Uint8Seq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.Uint8Seq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.Uint8Seq, *arg0:* rti.connextdds.Uint8Seq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.Uint8Seq, s: slice) -> rti.connextdds.Uint8Seq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.Uint8Seq, arg0: int) -> int

**__getstate__** (*self:* rti.connextdds.Uint8Seq) → bytes

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.Uint8Seq, *arg0:* rti.connextdds.Uint8Seq) → *rti.connextdds.Uint8Seq*

**__imul__** (*self:* rti.connextdds.Uint8Seq, *arg0: int*) → *rti.connextdds.Uint8Seq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.Uint8Seq, arg0: Buffer) -> None

2. __init__(self: rti.connextdds.Uint8Seq) -> None

3. __init__(self: rti.connextdds.Uint8Seq, arg0: rti.connextdds.Uint8Seq) -> None

Copy constructor

4. __init__(self: rti.connextdds.Uint8Seq, arg0: Iterable) -> None

5. __init__(self: rti.connextdds.Uint8Seq, arg0: int) -> None

**__iter__** (*self:* rti.connextdds.Uint8Seq) → Iterator[int]

**__len__** (*self:* rti.connextdds.Uint8Seq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.Uint8Seq, *arg0: int*) → *rti.connextdds.Uint8Seq*

**__ne__** (*self:* rti.connextdds.Uint8Seq, *arg0:* rti.connextdds.Uint8Seq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.Uint8Seq) → str

Return the string representation of this list. Values are printed as integers.

**__rmul__** (*self:* rti.connextdds.Uint8Seq, *arg0: int*) → *rti.connextdds.Uint8Seq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.Uint8Seq, arg0: int, arg1: int) -> None
2. __setitem__(self: rti.connextdds.Uint8Seq, arg0: slice, arg1: rti.connextdds.Uint8Seq) -> None

Assign list elements using a slice object

**__setstate__** (*self:* rti.connextdds.Uint8Seq, *arg0: bytes*) → None

**append** (*self:* rti.connextdds.Uint8Seq, *x: int*) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.Uint8Seq) → None

Clear the contents

**count** (*self:* rti.connextdds.Uint8Seq, *x: int*) → int

Return the number of times x appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.Uint8Seq, L: rti.connextdds.Uint8Seq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.Uint8Seq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.Uint8Seq, *i: int*, *x: int*) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.Uint8Seq) -> int

Remove and return the last item

2. pop(self: rti.connextdds.Uint8Seq, i: int) -> int

Remove and return the item at index i

**remove** (*self:* rti.connextdds.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.connextdds.Uint8Seq, *arg0: int*) → None

resizes the vector to the given size

**class** rti.connextdds.**Uint8Type**

Bases: *DynamicType*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Uint8Type, *arg0:* rti.connextdds.Uint8Type) → bool

Test for equality.

**__hash__ = None**

**__init__** (*self:* rti.connextdds.Uint8Type) → None

Get the singleton for Uint8Type

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Uint8Type, *arg0:* rti.connextdds.Uint8Type) → bool

Test for inequality.

**class** rti.connextdds.**UnidimensionalCollectionType**

Bases: *CollectionType*

**UNBOUNDED = 2147483647**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.UnidimensionalCollectionType, *arg0:* rti.connextdds.UnidimensionalCollectionType) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.UnidimensionalCollectionType, *arg0:* rti.connextdds.UnidimensionalCollectionType) → bool

Test for inequality.

**property bounds**

Gets the maximum length of this collection.

**class** rti.connextdds.**UnionMember**

Bases: pybind11_object

**DEFAULT_LABEL = 1073741825**

**INVALID_ID = 2147483647**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.UnionMember, *arg0:* rti.connextdds.UnionMember) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.UnionMember, name: str, dynamic_type: rti.connextdds.DynamicType, label: int) -> None

Creates a union member with a name, type, and selected by a single label.

2. __init__(self: rti.connextdds.UnionMember, name: str, dynamic_type: rti.connextdds.DynamicType, labels: rti.connextdds.Int32Seq) -> None

Create a union member with a name, type, and selected by one or more labels.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.UnionMember, *arg0:* rti.connextdds.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.

**property unit**

Retrieves the unit annotation of the member.

**class** rti.connextdds.**UnionMemberSeq**

Bases: pybind11_object

**__add__**(*self:* rti.connextdds.UnionMemberSeq, *arg0:* rti.connextdds.UnionMemberSeq) → *rti.connextdds.UnionMemberSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.UnionMemberSeq) → bool

Check whether the list is nonempty

**__contains__**(*self:* rti.connextdds.UnionMemberSeq, *x:* rti.connextdds.UnionMember) → bool

Return true the container contains `x`

**__delitem__**(**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.UnionMemberSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.UnionMemberSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__**(*self:* rti.connextdds.UnionMemberSeq, *arg0:* rti.connextdds.UnionMemberSeq) → bool

**__getitem__**(**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.UnionMemberSeq, s: slice) -> rti.connextdds.UnionMemberSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.UnionMemberSeq, arg0: int) -> rti.connextdds.UnionMember

**__hash__ = None**

**__iadd__**(*self:* rti.connextdds.UnionMemberSeq, *arg0:* rti.connextdds.UnionMemberSeq) → *rti.connextdds.UnionMemberSeq*

**__imul__**(*self:* rti.connextdds.UnionMemberSeq, *arg0: int*) → *rti.connextdds.UnionMemberSeq*

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.UnionMemberSeq) -> None

2. __init__(self: rti.connextdds.UnionMemberSeq, arg0: rti.connextdds.UnionMemberSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.UnionMemberSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.UnionMemberSeq) → Iterator[*rti.connextdds.UnionMember*]

**__len__** (*self:* rti.connextdds.UnionMemberSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.UnionMemberSeq, *arg0: int*) → *rti.connextdds.UnionMemberSeq*

**__ne__** (*self:* rti.connextdds.UnionMemberSeq, *arg0:* rti.connextdds.UnionMemberSeq) → bool

**__rmul__** (*self:* rti.connextdds.UnionMemberSeq, *arg0: int*) → *rti.connextdds.UnionMemberSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.UnionMemberSeq, arg0: int, arg1: rti.connextdds.UnionMember) -> None

2. __setitem__(self: rti.connextdds.UnionMemberSeq, arg0: slice, arg1: rti.connextdds.UnionMemberSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.UnionMemberSeq, *x:* rti.connextdds.UnionMember) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.UnionMemberSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.UnionMemberSeq, *x:* rti.connextdds.UnionMember) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.UnionMemberSeq, L: rti.connextdds.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*

**__annotations__ = {}**

**__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.connextdds.UnionType, *id: int*) → int

Gets the index of the member selected by an ID.

**find_member_by_label** (*self:* rti.connextdds.UnionType, *label: int*) → int

Gets the index of the member selected by a label.

**exception** rti.connextdds.**UnsupportedError**

Bases: *Exception*

**__annotations__ = {}**

**__module__ = 'rti.connextdds'**

**class** rti.connextdds.**UserData**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.UserData, *arg0:* rti.connextdds.UserData) → bool

Test for equality.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.UserData) -> None

Creates an instance with an empty sequence of bytes.

2. __init__(self: rti.connextdds.UserData, data: rti.connextdds.Uint8Seq) -> None

Creates an instance with a sequence of bytes.

**__iter__**(*self:* rti.connextdds.UserData) → Iterator[int]

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.UserData, *arg0:* rti.connextdds.UserData) → bool

Test for inequality.

**property value**

The user data.

**class** rti.connextdds.**UserDataSample**

Bases: pybind11_object

An alias for a Python object representing a data sample of an IDL type

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__module__ = 'rti.connextdds'**

rti.connextdds.**UshortSeq**

alias of *Uint16Seq*

**class** rti.connextdds.**ValuesView**

Bases: pybind11_object

**__annotations__ = {}**

**__init__**(**args*, ***kwargs*)

**__iter__**(*self:* rti.connextdds.ValuesView) → Iterator

**__len__**(*self:* rti.connextdds.ValuesView) → int

**__module__ = 'rti.connextdds'**

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**class** rti.connextdds.**VendorId**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.VendorId, *arg0:* rti.connextdds.VendorId) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.VendorId) → None

Creates unknown vendor ID.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.VendorId, *arg0:* rti.connextdds.VendorId) → bool

Test for inequality.

**unknown = <rti.connextdds.VendorId object>**

**property value**

Provides access to the bytes that represent the vendor id.

**class** rti.connextdds.**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 Connext. The most common cause of errors is incorrect configuration.

WARNING : Both error and warning messages will be logged.

A warning indicates that RTI Connext 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 Connext objects will be logged.

STATUS_REMOTE : Errors, warnings, and verbose information about the lifecycles of remote RTI Connext objects will be logged.

STATUS_ALL : Errors, warnings, verbose information about the lifecycles of local and remote RTI Connext objects, and periodic information about RTI Connext 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>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.Verbosity.Verbosity) → int

**__init__**(*self:* rti.connextdds.Verbosity.Verbosity, *value: int*) → None

**__int__**(*self:* rti.connextdds.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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__** (*self: object*) → str

**__setstate__** (*self:* rti.connextdds.Verbosity.Verbosity, *state: int*) → None

**__str__** (*self: object*) → str

**property name**

**property value**

**WARNING = <Verbosity.WARNING: 7>**

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.Verbosity, *arg0:* rti.connextdds.Verbosity) → bool
Apply operator to underlying enumerated values.

**__ge__** (*self:* rti.connextdds.Verbosity, *arg0:* rti.connextdds.Verbosity) → bool
Apply operator to underlying enumerated values.

**__gt__** (*self:* rti.connextdds.Verbosity, *arg0:* rti.connextdds.Verbosity) → bool
Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)
Overloaded function.

1. __init__(self: rti.connextdds.Verbosity) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.Verbosity, arg0: rti.connextdds.Verbosity.Verbosity) -> None

Copy constructor.

**__int__** (*self:* rti.connextdds.Verbosity) → *rti.connextdds.Verbosity.Verbosity*
Int conversion.

**__le__** (*self:* rti.connextdds.Verbosity, *arg0:* rti.connextdds.Verbosity) → bool
Apply operator to underlying enumerated values.

**__lt__** (*self:* rti.connextdds.Verbosity, *arg0:* rti.connextdds.Verbosity) → bool
Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.Verbosity, *arg0:* rti.connextdds.Verbosity) → bool
Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.Verbosity) → str
String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**ViewState**

Bases: pybind11_object

**ANY = <rti.connextdds.ViewState object>**

**NEW_VIEW = <rti.connextdds.ViewState object>**

**NOT_NEW_VIEW = <rti.connextdds.ViewState object>**

**__and__**(*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → *rti.connextdds.ViewState*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.ViewState) → *rti.connextdds.ViewState*

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → bool

**__eq__**(*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.ViewState, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → *rti.connextdds.ViewState*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.ViewState, *arg0: int*) → *rti.connextdds.ViewState*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.ViewState) -> None

Create a ViewState with no bits set.

2. __init__(self: rti.connextdds.ViewState, value: int) -> None

Creates a mask from the bits in an integer.

**__int__**(*self:* rti.connextdds.ViewState) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → *rti.connextdds.ViewState*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.ViewState, *arg0: int*) → *rti.connextdds.ViewState*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → *rti.connextdds.ViewState*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.ViewState, *arg0: int*) → *rti.connextdds.ViewState*

Left shift bits in mask.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → *rti.connextdds.ViewState*

Bitwise logical OR of masks.

**__rshift__** (*self:* rti.connextdds.ViewState, *arg0: int*) → *rti.connextdds.ViewState*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.ViewState, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.ViewState) → str

**__xor__** (*self:* rti.connextdds.ViewState, *arg0:* rti.connextdds.ViewState) → *rti.connextdds.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.connextdds.ViewState) -> rti.connextdds.ViewState

Flip all bits in the mask.

2. flip(self: rti.connextdds.ViewState, pos: int) -> rti.connextdds.ViewState

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.ViewState) -> rti.connextdds.ViewState

Clear all bits in the mask.

2. reset(self: rti.connextdds.ViewState, pos: int) -> rti.connextdds.ViewState

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.ViewState) -> rti.connextdds.ViewState

Set all bits in the mask.

2. set(self: rti.connextdds.ViewState, pos: int, value: bool = True) -> rti.connextdds.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.connextdds.ViewState, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all** (*self:* rti.connextdds.ViewState) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.ViewState) → bool

Test if any bits are set.

**test_none** (*self:* rti.connextdds.ViewState) → bool

Test if none of the bits are set.

**class** rti.connextdds.**WStringType**

Bases: *UnidimensionalCollectionType*

**__annotations__ = {}**

**__eq__** (*self:* rti.connextdds.WStringType, *arg0:* rti.connextdds.WStringType) → bool

Test for equality.

**__hash__ = None**

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.WStringType, bounds: int) -> None

Creates a bounded wide string.

2. __init__(self: rti.connextdds.WStringType) -> None

Creates an unbounded wide string.

**__module__ = 'rti.connextdds'**

**__ne__** (*self:* rti.connextdds.WStringType, *arg0:* rti.connextdds.WStringType) → bool

Test for inequality.

**class** rti.connextdds.**WaitSet**

Bases: pybind11_object

**__annotations__ = {}**

**__iadd__**(*self:* rti.connextdds.WaitSet, *arg0:* rti.connextdds.ICondition) → *rti.connextdds.WaitSet*

Attach a condition to a WaitSet.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.WaitSet) -> None

Create a WaitSet with no conditions attached.

2. __init__(self: rti.connextdds.WaitSet, property: rti.connextdds.WaitSetProperty) -> None

Create a WaitSet with no conditions attached with the specified WaitSetProperty settings.

**__isub__**(*self:* rti.connextdds.WaitSet, *arg0:* rti.connextdds.ICondition) → *rti.connextdds.WaitSet*

Detach a condition from a WaitSet.

**__module__ = 'rti.connextdds'**

**attach_condition**(*self:* rti.connextdds.WaitSet, *condition:* rti.connextdds.ICondition) → None

Attach a condition to the WaitSet.

**property conditions**

Get/set the attached conditions for the WaitSet.

**detach_all**(*self:* rti.connextdds.WaitSet) → None

Detach all conditions from the WaitSet.

**detach_condition**(*self:* rti.connextdds.WaitSet, *condition:* rti.connextdds.ICondition) → bool

Detach a condition from the WaitSet.

**dispatch**(**args*, ***kwargs*)

Overloaded function.

1. dispatch(self: rti.connextdds.WaitSet, timeout: rti.connextdds.Duration) -> None

Dispatch handlers for triggered conditions attached to this WaitSet with a timeout.

2. dispatch(self: rti.connextdds.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.connextdds.WaitSet, timeout: rti.connextdds.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.connextdds.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.connextdds.WaitSet, timeout: rti.connextdds.Duration) -> rti.connextdds.TriggeredConditions

Wait for conditions attached to this WaitSet to trigger with a timeout.

2. wait(self: rti.connextdds.WaitSet) -> rti.connextdds.TriggeredConditions

Wait indefinitely for conditions attached to this WaitSet to trigger.

**wait_async**(**args*, ***kwargs*)

Overloaded function.

1. wait_async(self: rti.connextdds.WaitSet, timeout: rti.connextdds.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.

**wait_one**(*self:* rti.connextdds.WaitSet, *timeout:* rti.connextdds.Duration) → bool

Wait for conditions attached to this WaitSet to trigger. Useful when only one condition is attached. Returns `true` if any condition triggered or `false` if the timeout expired.

**wait_one_async**(*self:* rti.connextdds.WaitSet, *timeout:* rti.connextdds.Duration) → object

Wait asynchronously for conditions attached to this WaitSet to trigger. Useful when only one condition is attached. Returns `true` if any condition triggered or `false` if the timeout expired.

**class** rti.connextdds.**WaitSetProperty**

Bases: `pybind11_object`

**__annotations__ = {}**

**__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*

**__annotations__ = {}**

**__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[str]

**__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_cxxabi1011__ = <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*

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.WcharType, *arg0:* rti.connextdds.WcharType) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.WcharType) → None

Get the singleton for WcharType

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.WcharType, *arg0:* rti.connextdds.WcharType) → bool

Test for inequality.

**class** rti.connextdds.**WireProtocol**

Bases: pybind11_object

**RTPS_AUTO_ID = 0**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.WireProtocol, *arg0:* rti.connextdds.WireProtocol) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.WireProtocol) → None

Creates the default policy.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.WireProtocol, *arg0:* rti.connextdds.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>**

**__annotations__ = {}**

**__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.connextdds'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.WireProtocolAutoKind, *arg0:* rti.connextdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.WireProtocolAutoKind, *arg0:* rti.connextdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.WireProtocolAutoKind, *arg0:* rti.connextdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.WireProtocolAutoKind) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.WireProtocolAutoKind, arg0: rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind) -> None

Copy constructor.

**__int__**(*self:* rti.connextdds.WireProtocolAutoKind) → *rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind*

Int conversion.

**__le__**(*self:* rti.connextdds.WireProtocolAutoKind, *arg0:* rti.connextdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

**__lt__**(*self:* rti.connextdds.WireProtocolAutoKind, *arg0:* rti.connextdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds'**

**__ne__**(*self:* rti.connextdds.WireProtocolAutoKind, *arg0:* rti.connextdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.WireProtocolAutoKind) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.**WriteParams**

Bases: pybind11_object

**__annotations__ = {}**

**__init__** (*self:* rti.connextdds.WriteParams) → None

Create a WriteParams object with default values.

**__module__ = 'rti.connextdds'**

**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

**__annotations__ = {}**

**__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.connextdds.WstringSeq, *arg0:* rti.connextdds.WstringSeq) → *rti.connextdds.WstringSeq*

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.WstringSeq) → bool

Check whether the list is nonempty

**__contains__** (*self:* rti.connextdds.WstringSeq, *x: str*) → bool

Return true the container contains `x`

**__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. __delitem__(self: rti.connextdds.WstringSeq, arg0: int) -> None

Delete the list elements at index `i`

2. __delitem__(self: rti.connextdds.WstringSeq, arg0: slice) -> None

Delete list elements using a slice object

**__eq__** (*self:* rti.connextdds.WstringSeq, *arg0:* rti.connextdds.WstringSeq) → bool

**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(self: rti.connextdds.WstringSeq, s: slice) -> rti.connextdds.WstringSeq

Retrieve list elements using a slice object

2. __getitem__(self: rti.connextdds.WstringSeq, arg0: int) -> str

**__hash__ = None**

**__iadd__** (*self:* rti.connextdds.WstringSeq, *arg0:* rti.connextdds.WstringSeq) → *rti.connextdds.WstringSeq*

**__imul__** (*self:* rti.connextdds.WstringSeq, *arg0: int*) → *rti.connextdds.WstringSeq*

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.WstringSeq) -> None

2. __init__(self: rti.connextdds.WstringSeq, arg0: rti.connextdds.WstringSeq) -> None

Copy constructor

3. __init__(self: rti.connextdds.WstringSeq, arg0: Iterable) -> None

**__iter__** (*self:* rti.connextdds.WstringSeq) → Iterator[str]

**__len__** (*self:* rti.connextdds.WstringSeq) → int

**__module__ = 'rti.connextdds'**

**__mul__** (*self:* rti.connextdds.WstringSeq, *arg0: int*) → *rti.connextdds.WstringSeq*

**__ne__** (*self:* rti.connextdds.WstringSeq, *arg0:* rti.connextdds.WstringSeq) → bool

**__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ = <capsule object NULL>**

**__repr__** (*self:* rti.connextdds.WstringSeq) → str

Return the canonical string representation of this list.

**__rmul__** (*self:* rti.connextdds.WstringSeq, *arg0: int*) → *rti.connextdds.WstringSeq*

**__setitem__** (**args*, ***kwargs*)

Overloaded function.

1. __setitem__(self: rti.connextdds.WstringSeq, arg0: int, arg1: str) -> None
2. __setitem__(self: rti.connextdds.WstringSeq, arg0: slice, arg1: rti.connextdds.WstringSeq) -> None

Assign list elements using a slice object

**append** (*self:* rti.connextdds.WstringSeq, *x: str*) → None

Add an item to the end of the list

**clear** (*self:* rti.connextdds.WstringSeq) → None

Clear the contents

**count** (*self:* rti.connextdds.WstringSeq, *x: str*) → int

Return the number of times `x` appears in the list

**extend** (**args*, ***kwargs*)

Overloaded function.

1. extend(self: rti.connextdds.WstringSeq, L: rti.connextdds.WstringSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.WstringSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

**insert** (*self:* rti.connextdds.WstringSeq, *i: int*, *x: str*) → None

Insert an item at a given position.

**pop** (**args*, ***kwargs*)

Overloaded function.

1. pop(self: rti.connextdds.WstringSeq) -> str

Remove and return the last item

2. pop(self: rti.connextdds.WstringSeq, i: int) -> str

Remove and return the item at index `i`

**remove** (*self:* rti.connextdds.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.connextdds.**enable_automatic_tss_cleanup** (*enable: bool*) → None

Enable automatic thread-specific storage (TSS) cleanup on platforms that support it.

rti.connextdds.**set_activity_context**(*attribute_mask:* rti.connextdds.ActivityContextMask) → None

Set the activity context mask.

rti.connextdds.**sleep**(*duration:* rti.connextdds.Duration) → None

Sleep for a specified duration

rti.connextdds.**spin**(*spin_count: int*) → None

Performs a spin operation (active wait) as many times as indicated.

rti.connextdds.**spin_per_microsecond**() → int

Returns the number of spin operations neededto wait 1 microsecond

rti.connextdds.**unregister_thread**() → None

Unregister a thread with the middleware

### 4.1.1 rti.connextdds.compliance

rti.connextdds.compliance contains APIs to configure compliance with certain standard specifications.

**class** rti.connextdds.compliance.**RpcMask**

Bases: pybind11_object

**DEFAULT_MASK = 00000000000000000000000000000000**

**USE_STANDARD_RELATED_SAMPLE_PID_BIT = 00000000000000000000000000000001**

**VENDOR = 00000000000000000000000000000001**

**__and__**(*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → *rti.connextdds.compliance.RpcMask*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.compliance.RpcMask) → bool

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → bool

**__eq__**(*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.compliance.RpcMask, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__** (*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → *rti.connextdds.compliance.RpcMask*

Set mask to logical AND with another mask.

**__ilshift__** (*self:* rti.connextdds.compliance.RpcMask, *arg0: int*) → *rti.connextdds.compliance.RpcMask*

Left shift bits in mask.

**__init__** (**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.compliance.RpcMask) -> None

Create an RpcMask with no bits set.

2. __init__(self: rti.connextdds.compliance.RpcMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__** (*self:* rti.connextdds.compliance.RpcMask) → int

Convert mask to int.

**__ior__** (*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → *rti.connextdds.compliance.RpcMask*

Set mask to logical OR with another mask.

**__irshift__** (*self:* rti.connextdds.compliance.RpcMask, *arg0: int*) → *rti.connextdds.compliance.RpcMask*

Right shift bits in mask.

**__ixor__** (*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → *rti.connextdds.compliance.RpcMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.compliance.RpcMask, *arg0: int*) → *rti.connextdds.compliance.RpcMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds.compliance'**

**__ne__** (*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → *rti.connextdds.compliance.RpcMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.compliance.RpcMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.compliance.RpcMask, *arg0: int*) → *rti.connextdds.compliance.RpcMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.compliance.RpcMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.compliance.RpcMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.compliance.RpcMask, *arg0:* rti.connextdds.compliance.RpcMask) → *rti.connextdds.compliance.RpcMask*

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.compliance.RpcMask) -> rti.connextdds.compliance.RpcMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.compliance.RpcMask, pos: int) -> rti.connextdds.compliance.RpcMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.compliance.RpcMask) -> rti.connextdds.compliance.RpcMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.compliance.RpcMask, pos: int) -> rti.connextdds.compliance.RpcMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.compliance.RpcMask) -> rti.connextdds.compliance.RpcMask

Set all bits in the mask.

2. set(self: rti.connextdds.compliance.RpcMask, pos: int, value: bool = True) -> rti.connextdds.compliance.RpcMask

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.compliance.RpcMask, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all** (*self:* rti.connextdds.compliance.RpcMask) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.compliance.RpcMask) → bool

Test if any bits are set.

**test_none** (*self:* rti.connextdds.compliance.RpcMask) → bool

Test if none of the bits are set.

**class** rti.connextdds.compliance.**XTypesMask**

Bases: pybind11_object

**ACCEPT_UNKNOWN_DISCRIMINATOR_BIT = 00000000000000000000000000100000**

**ACCEPT_UNKNOWN_ENUM_VALUE_BIT = 00000000000000000000000000010000**

**DEFAULT_MASK = 00000000000000000000000110001100**

**DHEADER_IN_NON_PRIMITIVE_COLLECTIONS_BIT =**
**00000000000000000000000000000001**

**ENCAPSULATION_OPTIONS_WITH_PADDING_BIT =**
**00000000000000000000000000001000**

**ENUM_AS_PRIMITIVE_IN_COLLECTIONS_BIT =**
**00000000000000000000000000000010**

**INITIALIZE_DISCRIMINATOR_TO_DEFAULT_BIT =**
**00000000000000000000000100000000**

**JAVA_KEYHASH_750_BIT = 00000000000000000000001000000000**

**MUST_UNDERSTAND_IN_KEY_AND_DISCRIMINATOR_BIT =**
**00000000000000000000100000000000**

**PARAMETER_LENGTH_WITH_PADDING_BIT =**
**00000000000000000000000000000100**

**PYTHON_INT8_730_BIT = 00000000000000000000010000000000**

**SELECT_DEFAULT_DISCRIMINATOR_BIT = 00000000000000000000000001000000**

**SENTINEL_IN_EMPTY_UNION_BIT = 00000000000000000000000010000000**

**VENDOR = 00000000000000000000100110101001**

**__and__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → *rti.connextdds.compliance.XTypesMask*

Bitwise logical AND of masks.

**__annotations__ = {}**

**__bool__**(*self:* rti.connextdds.compliance.XTypesMask) → bool

Test if any bits are set.

**__contains__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → bool

**__eq__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → bool

Compare masks for equality.

**__getitem__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0: int*) → bool

Get individual mask bit.

**__hash__ = None**

**__iand__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → *rti.connextdds.compliance.XTypesMask*

Set mask to logical AND with another mask.

**__ilshift__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0: int*) → *rti.connextdds.compliance.XTypesMask*

Left shift bits in mask.

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.compliance.XTypesMask) -> None

Create an XTypesMask with no bits set.

2. __init__(self: rti.connextdds.compliance.XTypesMask, value: int) -> None

Creates a mask from the bits in an integer.

**__int__**(*self:* rti.connextdds.compliance.XTypesMask) → int

Convert mask to int.

**__ior__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → *rti.connextdds.compliance.XTypesMask*

Set mask to logical OR with another mask.

**__irshift__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0: int*) → *rti.connextdds.compliance.XTypesMask*

Right shift bits in mask.

**__ixor__**(*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → *rti.connextdds.compliance.XTypesMask*

Set mask to logical XOR with another mask.

**__lshift__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0: int*) → *rti.connextdds.compliance.XTypesMask*

Left shift bits in mask.

**__module__ = 'rti.connextdds.compliance'**

**__ne__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → bool

Compare masks for inequality.

**__or__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → *rti.connextdds.compliance.XTypesMask*

Bitwise logical OR of masks.

**__repr__** (*self:* rti.connextdds.compliance.XTypesMask) → str

Convert mask to string.

**__rshift__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0: int*) → *rti.connextdds.compliance.XTypesMask*

Right shift bits in mask.

**__setitem__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0: int*, *arg1: bool*) → None

Set individual mask bit

**__str__** (*self:* rti.connextdds.compliance.XTypesMask) → str

Convert mask to string.

**__xor__** (*self:* rti.connextdds.compliance.XTypesMask, *arg0:* rti.connextdds.compliance.XTypesMask) → *rti.connextdds.compliance.XTypesMask*

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.compliance.XTypesMask) -> rti.connextdds.compliance.XTypesMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.compliance.XTypesMask, pos: int) -> rti.connextdds.compliance.XTypesMask

Flip the mask bit at the specified position.

**reset** (**args*, ***kwargs*)

Overloaded function.

1. reset(self: rti.connextdds.compliance.XTypesMask) -> rti.connextdds.compliance.XTypesMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.compliance.XTypesMask, pos: int) -> rti.connextdds.compliance.XTypesMask

Clear the mask bit at the specified position.

**set** (**args*, ***kwargs*)

Overloaded function.

1. set(self: rti.connextdds.compliance.XTypesMask) -> rti.connextdds.compliance.XTypesMask

Set all bits in the mask.

2. set(self: rti.connextdds.compliance.XTypesMask, pos: int, value: bool = True) -> rti.connextdds.compliance.XTypesMask

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.compliance.XTypesMask, *pos: int*) → bool

Test whether the mask bit at position "pos" is set.

**test_all** (*self:* rti.connextdds.compliance.XTypesMask) → bool

Test if all bits are set.

**test_any** (*self:* rti.connextdds.compliance.XTypesMask) → bool

Test if any bits are set.

**test_none** (*self:* rti.connextdds.compliance.XTypesMask) → bool

Test if none of the bits are set.

rti.connextdds.compliance.**get_rpc_mask**() → *rti.connextdds.compliance.RpcMask*

Gets the RPC compliance mask value for the application.

rti.connextdds.compliance.**get_xtypes_mask**() → *rti.connextdds.compliance.XTypesMask*

Gets the XTypes compliance mask value for the application.

rti.connextdds.compliance.**load_compliance_mask**() → None

Load the compliance masks from the environment variables.

rti.connextdds.compliance.**set_rpc_mask**(*mask:* rti.connextdds.compliance.RpcMask) → None

Sets the level of compliance of the application with the Rpc specification.

rti.connextdds.compliance.**set_xtypes_mask**(*mask:* rti.connextdds.compliance.XTypesMask) → None

Sets the level of compliance of the application with the XTypes specification.

### 4.1.2 rti.connextdds.heap_monitoring

rti.connextdds.heap_monitoring monitor memory allocations done by the middleware on the native heap.

**class** rti.connextdds.heap_monitoring.**HeapMonitoringParams**

Bases: pybind11_object

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.heap_monitoring.HeapMonitoringParams, *arg0:* rti.connextdds.heap_monitoring.HeapMonitoringParams) → bool

Test for equality.

**__hash__ = None**

**__init__**(*self:* rti.connextdds.heap_monitoring.HeapMonitoringParams, *snapshot_output_format:* rti.connextdds.heap_monitoring.SnapshotOutputFormat = *SnapshotOutputFormat.STANDARD*, *snapshot_content_format:* rti.connextdds.heap_monitoring.SnapshotContentFormat = *SnapshotOutputFormat.DEFAULT*) → None

Create parameters for heap monitoring.

**__module__ = 'rti.connextdds.heap_monitoring'**

**__ne__**(*self:* rti.connextdds.heap_monitoring.HeapMonitoringParams, *arg0:* rti.connextdds.heap_monitoring.HeapMonitoringParams) → bool

Test for inequality.

**property snapshot_content_format**

Information included in snapshot output.

**property snapshot_output_format**

Format of the snapshot output.

**class** rti.connextdds.heap_monitoring.**SnapshotContentFormat**

Bases: pybind11_object

**ACTIVITY = <SnapshotContentFormat.ACTIVITY: 4>**

**DEFAULT = <SnapshotContentFormat.DEFAULT: 7>**

**FUNCTION = <SnapshotContentFormat.FUNCTION: 2>**

**MINIMAL = <SnapshotContentFormat.MINIMAL: 0>**

**class SnapshotContentFormat**

Bases: pybind11_object

Members:

TOPIC : Add the topic to the snapshot of heap monitoring.

FUNCTION : Add the function name to the snapshot of heap monitoring.

ACTIVITY : Add the activity context to the snapshot of heap monitoring. The user can select the information that will be part of the activity context by using the API activity_context.set_attribute_mask.

DEFAULT : Add all the optional attributes to the snapshot of heap monitoring.

MINIMAL : Not add any optional attribute to the snapshot of heap monitoring.

**ACTIVITY = <SnapshotContentFormat.ACTIVITY: 4>**

**DEFAULT = <SnapshotContentFormat.DEFAULT: 7>**

**FUNCTION = <SnapshotContentFormat.FUNCTION: 2>**

**MINIMAL = <SnapshotContentFormat.MINIMAL: 0>**

**TOPIC = <SnapshotContentFormat.TOPIC: 1>**

**__annotations__ = {}**

**__eq__**(*self: object*, *other: object*) → bool

**__getstate__**(*self: object*) → int

**__hash__**(*self: object*) → int

**__index__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat) → int

**__init__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat, *value: int*) → None

**__int__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat) → int

**__members__ = {'ACTIVITY': <SnapshotContentFormat.ACTIVITY: 4>, 'DEFAULT': <SnapshotContentFormat.DEFAULT: 7>, 'FUNCTION': <SnapshotContentFormat.FUNCTION: 2>, 'MINIMAL': <SnapshotContentFormat.MINIMAL: 0>, 'TOPIC': <SnapshotContentFormat.TOPIC: 1>}**

**__module__ = 'rti.connextdds.heap_monitoring'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat, *state: int*) → None

**__str__**(*self: object*) → str

**property name**

**property value**

**TOPIC = <SnapshotContentFormat.TOPIC: 1>**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.heap_monitoring.SnapshotContentFormat) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.heap_monitoring.SnapshotContentFormat, arg0: rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat) -> None

Copy constructor.

**__int__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → *rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat*

Int conversion.

**__le__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → bool

Apply operator to underlying enumerated values.

**__lt__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds.heap_monitoring'**

**__ne__**(*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → bool

Apply operator to underlying enumerated values.

**__str__** (*self:* rti.connextdds.heap_monitoring.SnapshotContentFormat) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

**class** rti.connextdds.heap_monitoring.**SnapshotOutputFormat**

Bases: pybind11_object

**COMPRESSED = <SnapshotOutputFormat.COMPRESSED: 2>**

**STANDARD = <SnapshotOutputFormat.STANDARD: 1>**

**class SnapshotOutputFormat**

Bases: pybind11_object

Members:

STANDARD : The output of the snapshot will be in plain text.

COMPRESSED : The output of the snapshot will be compressed using Zlib techonology.

The file can be uncompressed using zlib-flate.

**COMPRESSED = <SnapshotOutputFormat.COMPRESSED: 2>**

**STANDARD = <SnapshotOutputFormat.STANDARD: 1>**

**__annotations__ = {}**

**__eq__** (*self: object*, *other: object*) → bool

**__getstate__** (*self: object*) → int

**__hash__** (*self: object*) → int

**__index__** (*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat) → int

**__init__** (*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat, *value: int*) → None

**__int__** (*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat) → int

**__members__ = {'COMPRESSED': <SnapshotOutputFormat.COMPRESSED: 2>, 'STANDARD': <SnapshotOutputFormat.STANDARD: 1>}**

**__module__ = 'rti.connextdds.heap_monitoring'**

**__ne__**(*self: object*, *other: object*) → bool

**__repr__**(*self: object*) → str

**__setstate__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat, *state: int*) →
None

**__str__**(*self: object*) → str

**property name**

**property value**

**__annotations__ = {}**

**__eq__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → bool

Apply operator to underlying enumerated values.

**__ge__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → bool

Apply operator to underlying enumerated values.

**__gt__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → bool

Apply operator to underlying enumerated values.

**__hash__ = None**

**__init__**(**args*, ***kwargs*)

Overloaded function.

1. __init__(self: rti.connextdds.heap_monitoring.SnapshotOutputFormat) -> None

Initializes enum to 0.

2. __init__(self: rti.connextdds.heap_monitoring.SnapshotOutputFormat, arg0: rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat) -> None

Copy constructor.

**__int__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) →
*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat*

Int conversion.

**__le__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → bool

Apply operator to underlying enumerated values.

**__lt__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → bool

Apply operator to underlying enumerated values.

**__module__ = 'rti.connextdds.heap_monitoring'**

**__ne__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat, *arg0:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → bool

Apply operator to underlying enumerated values.

**__str__**(*self:* rti.connextdds.heap_monitoring.SnapshotOutputFormat) → str

String conversion.

**property underlying**

Retrieves the actual enumerated value.

rti.connextdds.heap_monitoring.**disable**() → None

Stop monitoring the heap memory used by RTI Connext.

rti.connextdds.heap_monitoring.**enable**(**args*, ***kwargs*)

Overloaded function.

1. enable() -> bool

Start monitoring the heap memory used by RTI Connext. Must be called before any using any other int the RTI Connext library.

2. enable(params: rti.connextdds.heap_monitoring.HeapMonitoringParams) -> bool

Start monitoring the heap memory used by RTI Connext with params. Must be called before any using any other in the RTI Connext library.

rti.connextdds.heap_monitoring.**pause**() → bool

Pauses heap monitoring.

rti.connextdds.heap_monitoring.**resume**() → bool

Resumes heap monitoring.

rti.connextdds.heap_monitoring.**take_snapshot**(*filename: str*, *print_details: bool = False*) → bool

Saves the current heap memory usage in a file.

## 4.2 rti.types

This module defines the `struct`, `union`, `enum`, `alias` decorators that allow turning Python `dataclasses` into DDS topic-types.

See *Data Types* for more information.

`rti.types` also contains `rti.types.builtin`.

### 4.2.1 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[~rti.idl_impl.type_hints.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[~rti.idl_impl.type_hints.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, "My Topic", KeyedString)`.

**key: str = ''**

The key

**value: str = ''**

The string payload

**class** rti.types.builtin.**PingType**(*number: int32 = 0*)

The type used by the rtiddsping CLI utility, consisting of a single 32-bit integer.

This type can be used to create a dds.Topic to interact with rtiddsping, e.g. `topic = dds.Topic(participant, "PingTopic", PingType)`

**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, "My Topic", String)` `

**value: str = ''**

The string payload

## 4.3 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.**ByRef**(*value: T*)

This class is designed to emulate pass-by-reference behavior in Python, which natively uses pass-by-value semantics for primitive types (like *int*, *float*, and *str*). It is used to implement the *rpc.out_param* and *rpc.inout_param* for the primitive types.

For example for the following operation: ` void op(out long param); `

The Python equivalent will be: ` def op(self, param: rpc.ByRef[idl.int32]): `

This allows the modification of the parameter value in the service implementation, and this change will be reflected in the client side variable used to call the operation. `param.value = 10`

**class** rti.rpc.**ClientBase**(*participant: ~rti.connextdds.DomainParticipant*, *service_name: str*, *max_wait_per_call: ~rti.connextdds.Duration = <Duration(sec=10, nanosec=0)>*, *datawriter_qos: ~typing.Optional[~rti.connextdds.DataWriterQos] = None*, *datareader_qos: ~typing.Optional[~rti.connextdds.DataReaderQos] = None*, *publisher: ~typing.Optional[~rti.connextdds.Publisher] = None*, *subscriber: ~typing.Optional[~rti.connextdds.Subscriber] = None*, *require_matching_service_on_send_request: bool = True*)

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.

**async wait_for_service_async**(*max_wait:* Duration) → bool

Waits for a service to be discovered.

This method returns True if a service is discovered and ready to receive requests within the specified maximum wait. It returns False if it times out.

**exception** rti.rpc.**RemoteError**

Base class for all built-in exceptions that can be reported by a service and thrown by a client. This doesn't include user-defined exceptions.

**exception** rti.rpc.**RemoteInvalidArgumentError**

Exception thrown by a client operation when the server indicates that the operation failed because of an invalid argument.

**exception** rti.rpc.**RemoteOutOfResourcesError**

Exception thrown by a client operation when the server indicates that the operation failed because the server ran out of resources.

**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.

**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.**RemoteUnsupportedOperationError**

Exception thrown by a client operation when the server indicates that the operation is not supported.

**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 the QoS defined by the built-in profile dds.BuiltinProfiles.pattern_rpc).
- **datareader_qos** – QoS object to use for request reader, defaults to None (use the QoS defined by the built-in profile dds.BuiltinProfiles.pattern_rpc).
- **publisher** – Publisher used to create the request writer, defaults to None (use participant.builtin_publisher).
- **subscriber** – Subscriber used to create the reply reader, defaults to None (use participant.builtin_subscriber).
- **on_reply_available** – The callback that handles incoming requests (optional).

**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.**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*, *require_matching_service_on_send_request: bool = True*)

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 the QoS defined by the built-in profile dds.BuiltinProfiles.pattern_rpc).
- **datareader_qos** – QoS object to use for reply reader, defaults to None (use the QoS defined by the built-in profile dds.BuiltinProfiles.pattern_rpc).
- **publisher** – Publisher used to create the request writer, defaults to None (use participant.builtin_publisher).
- **subscriber** – Subscriber used to create the reply reader, defaults to None (use participant.builtin_subscriber).
- **on_reply_available** – The callback that handles incoming replies.
- **require_matching_service_on_send_request** – Indicates whether send_request fails if no service has been discovered, default True.

**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.

**async wait_for_service_async**(*max_wait:* Duration) → bool

Asynchronously waits for a Replier to be discovered.

This method returns True if a Replier is discovered and ready to receive requests within the specified maximum wait. It returns False if it times out.

**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 useses 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.**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).

**property matched_requester_count: int**

The number of discovered matched requesters.

**Getter**

Returns the number of matched requesters.

rti.rpc.**get_interface_types**(*interface: type*) → List[type]

Returns a list of all the IDL types used by this interface (call, return, in, out, result)

rti.rpc.**operation**(*funcobj=None*, *, *raises=[]*, *operation_annotations=[]*, *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.

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.

## 4.4 rti.asyncio

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 rti.asyncio.Application, an easy way to subscribe to topics by decorating functions that receive each topic update.

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, as well as perform certain cleanup tasks at the end.

See *Subscriptions*.

**class** rti.asyncio.**Application**

An rti.asyncio.Application provides an easy way to subscribe to topics by simply decorating async functions to receive topic updates.

To use this class:

1. Create an `app` instance of this class (typically one per application)
2. For each topic you want to subscribe to decorate a function with `@app.subscribe()`
3. Run the application, typically with `rti.asyncio.run(app.run(domain_id))`

For example:

```
app = rti.asyncio.Application()

@app.subscribe(topic_name="Sensor Temperature")
async def on_sensor_temperature(data: Temperature):
    print(data)

@app.subscribe(topic_name="Alerts")
async def on_alert(data: Alert):
    print(data)

rti.asyncio.run(app.run(domain_id=0))
```

**async run**(*domain_id: Union[int,* DomainParticipant*] = 0*, *qos_file: Optional[str] = None*)

Start all subscriptions and start notifying decorated functions.

This function creates a `dds.DomainParticipant` with the given domain_id or uses an existing participant if one is provided. It then creates a DataReader for each function that was decorated with `@app.subscribe`.

The optional `qos_file` argument loads an XML file used to look up any QoS profiles specified in the `@app.subscribe` decorator. By default the default *Connext* methods to look for profiles are used, including the `<working directory>/USER_QOS_PROFILES.xml` file.

The simplest way to run this method is with `rti.asyncio.run`:

```
if __name__ == "__main__":
    rti.asyncio.run(app.run(domain_id))
```

To stop the application (and all subscriptions), run as a task and cancel it when needed:

```
async def main():
    task = asyncio.create_task(app.run(domain_id))
    # subscriptions are running ...
    task.cancel()
    await task

if __name__ == "__main__":
    rti.asyncio.run(main())
```

**subscribe**(*topic_name: str*, *qos_profile: Optional[str] = None*)

Decorator for a function that will receive samples of the given topic.

The decorated function must meet the following requirements:

- It must be `async`.

- It must have at least one argument.
- The first argument must have a type annotation that corresponds to the type of the topic.

The function may have an optional second argument with the type annotation `dds.SampleInfo`.

The decorator receives the topic name and an optional QoS profile name. The QoS profile name is used to configure the DataReader and Subscriber QoS.

Example:

```
@app.subscribe(topic_name="MyTopic", qos_profile="my_library::my_
↪profile")
async def on_my_topic(data: MyType):
    print(data)
```

Decorating a function has the following effect when `run()` is called:

- A `dds.Topic` with name `"MyTopic"` and type `MyType` is created. If one already exists and the type is the same, it is re-used. If the type is different, an exception is raised.
- A `dds.DataReader` is created for the topic. If a QoS profile is provided, the profile is loaded and used to configure the DataReader and Subscriber.
- The decorated function is called with each data sample received by the DataReader.
- If the function has a second argument with the type `dds.SampleInfo`, the sample meta-data is also passed to the function. For updates with meta-data only (e.g. when an instance is disposed), the first argument is `None`.

`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. When it ends, it cleans up global resources.

## 4.5 rti.logging

### 4.5.1 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.

### 4.5.2 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.

See the genindex for a list of all types and functions.

# Chapter 5

# Examples

In addition to the code snippets in the RTI Connext Getting Started Guide and the *API Overview*, you can find full example applications in the Connext 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

# Chapter 6

# Copyrights and Notices

© 2022-2026 Real-Time Innovations, Inc. All rights reserved. April 2026

### Trademarks

RTI, Real-Time Innovations, Connext, Connext Drive, 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 agreed to in writing by RTI.

### Third-Party Software

RTI software may contain independent, third-party software or code that are subject to third-party license terms and conditions, including open source license terms and conditions. Copies of applicable third-party licenses and notices are located at community.rti.com/documentation. IT IS YOUR RESPONSIBILITY TO ENSURE THAT YOUR USE OF THIRD-PARTY SOFTWARE COMPLIES WITH THE CORRESPONDING THIRD-PARTY LICENSE TERMS AND CONDITIONS.

## Notices

*Deprecations and Removals*

*Deprecated* means that the item is still supported in this release, but will be removed in a future release. *Removed* means that the item is discontinued or no longer supported.

Any deprecations or removals noted in RTI's documentation 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. RTI's current standard terms and support and maintenance policies are available at https://www.rti.com/terms.

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/

# Python Module Index

## r

rti.asyncio, 810
rti.connextdds, 34
rti.connextdds.compliance, 791
rti.connextdds.heap_monitoring, 798
rti.logging, 816
rti.logging.distlog, 812
rti.logging.handler, 815
rti.rpc, 805
rti.types.builtin, 804

# Index

## Non-alphabetical


__add__() (*rti.connextdds.AnyDataReaderSeq method*), 46
__add__() (*rti.connextdds.AnyDataWriterSeq method*), 50
__add__() (*rti.connextdds.AnyTopicSeq method*), 52
__add__() (*rti.connextdds.BoolSeq method*), 58
__add__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__add__() (*rti.connextdds.CharSeq method*), 70
__add__() (*rti.connextdds.ConditionSeq method*), 79
__add__() (*rti.connextdds.ContentFilteredTopicSeq method*), 82
__add__() (*rti.connextdds.CookieSeq method*), 85
__add__() (*rti.connextdds.DataReaderSeq method*), 113
__add__() (*rti.connextdds.DataWriterSeq method*), 144
__add__() (*rti.connextdds.DomainParticipantSeq method*), 183
__add__() (*rti.connextdds.Duration method*), 190
__add__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__add__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 203
__add__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 214
__add__() (*rti.connextdds.DynamicDataSeq method*), 259
__add__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__add__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__add__() (*rti.connextdds.EndpointGroupSeq method*), 266
__add__() (*rti.connextdds.EntitySeq method*), 271
__add__() (*rti.connextdds.EnumMemberSeq method*), 274
__add__() (*rti.connextdds.Float32Seq method*), 285
__add__() (*rti.connextdds.Float64Seq method*), 287
__add__() (*rti.connextdds.Float128Seq method*), 282
__add__() (*rti.connextdds.IAnyDataReaderSeq method*), 302
__add__() (*rti.connextdds.IAnyDataWriterSeq method*), 305
__add__() (*rti.connextdds.IAnyTopicSeq method*), 308
__add__() (*rti.connextdds.IConditionSeq method*), 310
__add__() (*rti.connextdds.IEntitySeq method*), 314
__add__() (*rti.connextdds.InstanceHandleSeq method*), 320
__add__() (*rti.connextdds.Int8Seq method*), 335
__add__() (*rti.connextdds.Int16Seq method*), 326
__add__() (*rti.connextdds.Int32Seq method*), 328
__add__() (*rti.connextdds.Int64Seq method*), 332
__add__() (*rti.connextdds.LocatorFilterElementSeq method*), 346
__add__() (*rti.connextdds.LocatorSeq method*), 351
__add__() (*rti.connextdds.MemberSeq method*), 364
__add__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 371
__add__() (*rti.connextdds.MulticastMappingSeq method*), 376
__add__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 390
__add__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__add__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 411
__add__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 426
__add__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__add__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 421
__add__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 448
__add__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__add__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 469
__add__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 485
__add__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__add__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 479
__add__() (*rti.connextdds.PublisherSeq method*), 497
__add__() (*rti.connextdds.QosPolicyCountSeq method*), 500
__add__() (*rti.connextdds.SequenceNumber method*), 542
__add__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 549
__add__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 558
__add__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 569
__add__() (*rti.connextdds.ServiceRequestSeq method*), 585
__add__() (*rti.connextdds.ServiceRequestTimestampedSeq

*method*), 588
__add__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 578
__add__() (*rti.connextdds.StringPairSeq method*), 598
__add__() (*rti.connextdds.StringSeq method*), 600
__add__() (*rti.connextdds.SubscriberSeq method*), 608
__add__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 611
__add__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 621
__add__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 633
__add__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 649
__add__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 651
__add__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__add__() (*rti.connextdds.Time method*), 663
__add__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 668
__add__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 678
__add__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 689
__add__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 704
__add__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 706
__add__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__add__() (*rti.connextdds.TopicSeq method*), 719
__add__() (*rti.connextdds.TransportInfoSeq method*), 729
__add__() (*rti.connextdds.TransportMulticastSeq method*), 736
__add__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 738
__add__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 743
__add__() (*rti.connextdds.Uint8Seq method*), 764
__add__() (*rti.connextdds.Uint16Seq method*), 756
__add__() (*rti.connextdds.Uint32Seq method*), 759
__add__() (*rti.connextdds.Uint64Seq method*), 762
__add__() (*rti.connextdds.UnionMemberSeq method*), 769
__add__() (*rti.connextdds.WcharSeq method*), 781
__add__() (*rti.connextdds.WstringSeq method*), 788
__and__() (*rti.connextdds.ActivityContextMask method*), 41
__and__() (*rti.connextdds.compliance.RpcMask method*), 791
__and__() (*rti.connextdds.compliance.XTypesMask method*), 794
__and__() (*rti.connextdds.CompressionIdMask method*), 75
__and__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 158
__and__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 161
__and__() (*rti.connextdds.InstanceState method*), 323
__and__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__and__() (*rti.connextdds.SampleFlag method*), 528
__and__() (*rti.connextdds.SampleLostState method*), 534
__and__() (*rti.connextdds.SampleRejectedState method*), 537
__and__() (*rti.connextdds.SampleState method*), 540
__and__() (*rti.connextdds.StatusMask method*), 592
__and__() (*rti.connextdds.StreamKind method*), 595
__and__() (*rti.connextdds.ThreadSettingsKindMask method*), 660
__and__() (*rti.connextdds.TransportBuiltinMask method*), 722
__and__() (*rti.connextdds.ViewState method*), 776
__annotations__ (*rti.connextdds.AcknowledgmentInfo attribute*), 38
__annotations__ (*rti.connextdds.AcknowledgmentKind attribute*), 40
__annotations__ (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 39
__annotations__ (*rti.connextdds.AckResponseData attribute*), 37
__annotations__ (*rti.connextdds.ActivityContextMask attribute*), 41
__annotations__ (*rti.connextdds.ACTMember attribute*), 35
__annotations__ (*rti.connextdds.ACTUnionMember attribute*), 36
__annotations__ (*rti.connextdds.AliasType attribute*), 44
__annotations__ (*rti.connextdds.AllocationSettings attribute*), 45
__annotations__ (*rti.connextdds.AnyDataReader attribute*), 45
__annotations__ (*rti.connextdds.AnyDataReaderListener attribute*), 45
__annotations__ (*rti.connextdds.AnyDataReaderSeq attribute*), 46
__annotations__ (*rti.connextdds.AnyDataWriter attribute*), 49
__annotations__ (*rti.connextdds.AnyDataWriterListener attribute*), 49
__annotations__ (*rti.connextdds.AnyDataWriterSeq attribute*), 50
__annotations__ (*rti.connextdds.AnyTopic attribute*), 52
__annotations__ (*rti.connextdds.AnyTopicListener attribute*), 52
__annotations__ (*rti.connextdds.AnyTopicSeq attribute*), 53
__annotations__ (*rti.connextdds.ArrayType attribute*), 54
__annotations__ (*rti.connextdds.AsynchronousPublisher attribute*), 55
__annotations__ (*rti.connextdds.Availability attribute*), 56
__annotations__ (*rti.connextdds.Batch attribute*), 57
__annotations__ (*rti.connextdds.BoolSeq attribute*), 58

__annotations__ (*rti.connextdds.BoolType attribute*), 60
__annotations__ (*rti.connextdds.BuiltinProfiles attribute*), 60
__annotations__ (*rti.connextdds.BuiltinTopicKey attribute*), 62
__annotations__ (*rti.connextdds.BuiltinTopicReaderResourceLimits attribute*), 63
__annotations__ (*rti.connextdds.ByteVector attribute*), 64
__annotations__ (*rti.connextdds.CdrPaddingKind attribute*), 66
__annotations__ (*rti.connextdds.CdrPaddingKind.CdrPaddingKind attribute*), 66
__annotations__ (*rti.connextdds.ChannelSettings attribute*), 67
__annotations__ (*rti.connextdds.ChannelSettingsSeq attribute*), 68
__annotations__ (*rti.connextdds.CharSeq attribute*), 70
__annotations__ (*rti.connextdds.CharType attribute*), 72
__annotations__ (*rti.connextdds.CoherentAccess attribute*), 73
__annotations__ (*rti.connextdds.CoherentSet attribute*), 73
__annotations__ (*rti.connextdds.CoherentSetInfo attribute*), 73
__annotations__ (*rti.connextdds.CollectionType attribute*), 74
__annotations__ (*rti.connextdds.compliance.RpcMask attribute*), 791
__annotations__ (*rti.connextdds.compliance.XTypesMask attribute*), 794
__annotations__ (*rti.connextdds.CompressionIdMask attribute*), 75
__annotations__ (*rti.connextdds.CompressionSettings attribute*), 78
__annotations__ (*rti.connextdds.Condition attribute*), 78
__annotations__ (*rti.connextdds.ConditionSeq attribute*), 79
__annotations__ (*rti.connextdds.ContentFilterBase attribute*), 81
__annotations__ (*rti.connextdds.ContentFilteredTopic attribute*), 81
__annotations__ (*rti.connextdds.ContentFilteredTopicSeq attribute*), 82
__annotations__ (*rti.connextdds.ContentFilterProperty attribute*), 81
__annotations__ (*rti.connextdds.Cookie attribute*), 85
__annotations__ (*rti.connextdds.CookieSeq attribute*), 85
__annotations__ (*rti.connextdds.CookieVector attribute*), 87
__annotations__ (*rti.connextdds.Database attribute*), 147
__annotations__ (*rti.connextdds.DataReader attribute*), 91
__annotations__ (*rti.connextdds.DataReaderCacheStatus attribute*), 95
__annotations__ (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 99
__annotations__ (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 98
__annotations__ (*rti.connextdds.DataReaderListener attribute*), 100
__annotations__ (*rti.connextdds.DataReader.LoanedSample attribute*), 88
__annotations__ (*rti.connextdds.DataReader.LoanedSamples attribute*), 89
__annotations__ (*rti.connextdds.DataReaderProtocol attribute*), 101
__annotations__ (*rti.connextdds.DataReaderProtocolStatus attribute*), 101
__annotations__ (*rti.connextdds.DataReaderQos attribute*), 103
__annotations__ (*rti.connextdds.DataReaderResourceLimits attribute*), 110
__annotations__ (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings attribute*), 112
__annotations__ (*rti.connextdds.DataReader.Selector attribute*), 90
__annotations__ (*rti.connextdds.DataReaderSeq attribute*), 113
__annotations__ (*rti.connextdds.DataRepresentation attribute*), 115
__annotations__ (*rti.connextdds.DataState attribute*), 116
__annotations__ (*rti.connextdds.DataStateEx attribute*), 117
__annotations__ (*rti.connextdds.DataTag attribute*), 119
__annotations__ (*rti.connextdds.DataWriter attribute*), 120
__annotations__ (*rti.connextdds.DataWriterCacheStatus attribute*), 126
__annotations__ (*rti.connextdds.DataWriterListener attribute*), 127
__annotations__ (*rti.connextdds.DataWriterProtocol attribute*), 128
__annotations__ (*rti.connextdds.DataWriterProtocolStatus attribute*), 129
__annotations__ (*rti.connextdds.DataWriterQos attribute*), 131
__annotations__ (*rti.connextdds.DataWriterResourceLimits attribute*), 139
__annotations__ (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 143
__annotations__ (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142
__annotations__ (*rti.connextdds.DataWriterSeq attribute*), 144
__annotations__ (*rti.connextdds.DataWriterShmemRefTransferModeSettings attribute*), 146
__annotations__ (*rti.connextdds.DataWriterTransferMode attribute*), 147
__annotations__ (*rti.connextdds.Deadline attribute*), 148
__annotations__ (*rti.connextdds.DestinationOrder*

*attribute*), 149
__annotations__ (*rti.connextdds.DestinationOrderKind attribute*), 150
__annotations__ (*rti.connextdds.DestinationOrderKind.DestinationOrderKind attribute*), 150
__annotations__ (*rti.connextdds.DestinationOrderScopeKind attribute*), 153
__annotations__ (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 152
__annotations__ (*rti.connextdds.Discovery attribute*), 154
__annotations__ (*rti.connextdds.DiscoveryConfig attribute*), 154
__annotations__ (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 158
__annotations__ (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
__annotations__ (*rti.connextdds.DomainParticipant attribute*), 164
__annotations__ (*rti.connextdds.DomainParticipantConfigParams attribute*), 171
__annotations__ (*rti.connextdds.DomainParticipantFactoryQos attribute*), 172
__annotations__ (*rti.connextdds.DomainParticipantListener attribute*), 173
__annotations__ (*rti.connextdds.DomainParticipantProtocolStatus attribute*), 173
__annotations__ (*rti.connextdds.DomainParticipantQos attribute*), 174
__annotations__ (*rti.connextdds.DomainParticipantResourceLimits attribute*), 178
__annotations__ (*rti.connextdds.DomainParticipantSeq attribute*), 183
__annotations__ (*rti.connextdds.Durability attribute*), 185
__annotations__ (*rti.connextdds.DurabilityKind attribute*), 188
__annotations__ (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 187
__annotations__ (*rti.connextdds.DurabilityService attribute*), 189
__annotations__ (*rti.connextdds.Duration attribute*), 190
__annotations__ (*rti.connextdds.DynamicData attribute*), 229
__annotations__ (*rti.connextdds.DynamicData.ContentFilter attribute*), 192
__annotations__ (*rti.connextdds.DynamicData.ContentFilteredTopic attribute*), 193
__annotations__ (*rti.connextdds.DynamicData.ContentFilteredTopicSeq attribute*), 194
__annotations__ (*rti.connextdds.DynamicData.DataReader attribute*), 197
__annotations__ (*rti.connextdds.DynamicData.DataReaderListener attribute*), 202
__annotations__ (*rti.connextdds.DynamicData.DataReader.Selector attribute*), 196
__annotations__ (*rti.connextdds.DynamicData.DataReaderSeq attribute*), 203
__annotations__ (*rti.connextdds.DynamicData.DataWriter attribute*), 205
__annotations__ (*rti.connextdds.DynamicData.DataWriterListener attribute*), 213
__annotations__ (*rti.connextdds.DynamicData.DataWriterSeq attribute*), 214
__annotations__ (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 255
__annotations__ (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 254
__annotations__ (*rti.connextdds.DynamicData.FieldsIterator attribute*), 216
__annotations__ (*rti.connextdds.DynamicData.FieldsView attribute*), 216
__annotations__ (*rti.connextdds.DynamicData.IndexIterator attribute*), 217
__annotations__ (*rti.connextdds.DynamicDataInfo attribute*), 257
__annotations__ (*rti.connextdds.DynamicData.ItemsIterator attribute*), 217
__annotations__ (*rti.connextdds.DynamicData.ItemsView attribute*), 218
__annotations__ (*rti.connextdds.DynamicData.ITopicDescription attribute*), 217
__annotations__ (*rti.connextdds.DynamicData.LoanedSample attribute*), 218
__annotations__ (*rti.connextdds.DynamicData.LoanedSamples attribute*), 219
__annotations__ (*rti.connextdds.DynamicDataMemberInfo attribute*), 257
__annotations__ (*rti.connextdds.DynamicData.NoOpDataReaderListener attribute*), 219
__annotations__ (*rti.connextdds.DynamicData.NoOp-*

*DataWriterListener attribute*), 220
__annotations__ (*rti.connextdds.DynamicData.NoOpTopicListener attribute*), 222
__annotations__ (*rti.connextdds.DynamicDataProperty attribute*), 258
__annotations__ (*rti.connextdds.DynamicData.Sample attribute*), 222
__annotations__ (*rti.connextdds.DynamicDataSeq attribute*), 259
__annotations__ (*rti.connextdds.DynamicData.SharedSamples attribute*), 222
__annotations__ (*rti.connextdds.DynamicDataTimestampedSeq attribute*), 261
__annotations__ (*rti.connextdds.DynamicData.Topic attribute*), 223
__annotations__ (*rti.connextdds.DynamicData.TopicDescription attribute*), 224
__annotations__ (*rti.connextdds.DynamicData.TopicListener attribute*), 225
__annotations__ (*rti.connextdds.DynamicData.TopicSeq attribute*), 225
__annotations__ (*rti.connextdds.DynamicData.TopicTypeSupport attribute*), 227
__annotations__ (*rti.connextdds.DynamicDataTypeSerializationProperty attribute*), 263
__annotations__ (*rti.connextdds.DynamicData.ValidLoanedSamples attribute*), 227
__annotations__ (*rti.connextdds.DynamicData.WriterContentFilter attribute*), 228
__annotations__ (*rti.connextdds.DynamicData.WriterContentFilterHelper attribute*), 229
__annotations__ (*rti.connextdds.DynamicType attribute*), 264
__annotations__ (*rti.connextdds.DynamicTypePrintFormatProperty attribute*), 265
__annotations__ (*rti.connextdds.EndpointGroup attribute*), 266
__annotations__ (*rti.connextdds.EndpointGroupSeq attribute*), 266
__annotations__ (*rti.connextdds.EndpointGroupVector attribute*), 268
__annotations__ (*rti.connextdds.Entity attribute*), 269
__annotations__ (*rti.connextdds.EntityFactory attribute*), 269
__annotations__ (*rti.connextdds.EntityLock attribute*), 270
__annotations__ (*rti.connextdds.EntityName attribute*), 270
__annotations__ (*rti.connextdds.EntitySeq attribute*), 271
__annotations__ (*rti.connextdds.EnumMember attribute*), 273
__annotations__ (*rti.connextdds.EnumMemberSeq attribute*), 274
__annotations__ (*rti.connextdds.EnumType attribute*), 276
__annotations__ (*rti.connextdds.Error attribute*), 277
__annotations__ (*rti.connextdds.Event attribute*), 277
__annotations__ (*rti.connextdds.EventCount32 attribute*), 277
__annotations__ (*rti.connextdds.EventCount64 attribute*), 278
__annotations__ (*rti.connextdds.Exception attribute*), 278
__annotations__ (*rti.connextdds.ExpressionProperty attribute*), 278
__annotations__ (*rti.connextdds.ExtensibilityKind attribute*), 280
__annotations__ (*rti.connextdds.ExtensibilityKind.ExtensibilityKind attribute*), 279
__annotations__ (*rti.connextdds.Filter attribute*), 281
__annotations__ (*rti.connextdds.FilterSampleInfo attribute*), 282
__annotations__ (*rti.connextdds.Float32Seq attribute*), 285
__annotations__ (*rti.connextdds.Float32Type attribute*), 287
__annotations__ (*rti.connextdds.Float64Seq attribute*), 287
__annotations__ (*rti.connextdds.Float64Type attribute*), 290
__annotations__ (*rti.connextdds.Float128Seq attribute*), 282
__annotations__ (*rti.connextdds.Float128Type attribute*), 284
__annotations__ (*rti.connextdds.FlowController attribute*), 290
__annotations__ (*rti.connextdds.FlowControllerProperty attribute*), 291
__annotations__ (*rti.connextdds.FlowControllerSchedulingPolicy attribute*), 294
__annotations__ (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 293
__annotations__ (*rti.connextdds.FlowControllerTokenBucketProperty attribute*), 295
__annotations__ (*rti.connextdds.GenerationCount attribute*), 296
__annotations__ (*rti.connextdds.GroupData attribute*), 296
__annotations__ (*rti.connextdds.GuardCondition attribute*), 297
__annotations__ (*rti.connextdds.Guid attribute*), 297
__annotations__ (*rti.connextdds.heap_monitoring.HeapMonitoringParams attribute*), 798
__annotations__ (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 800
__annotations__ (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*),

799
__annotations__ (*rti.connextdds.heap_monitoring.SnapshotOutputFormat attribute*), 802
__annotations__ (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat attribute*), 801
__annotations__ (*rti.connextdds.History attribute*), 298
__annotations__ (*rti.connextdds.HistoryKind attribute*), 300
__annotations__ (*rti.connextdds.HistoryKind.HistoryKind attribute*), 300
__annotations__ (*rti.connextdds.IAnyDataReader attribute*), 301
__annotations__ (*rti.connextdds.IAnyDataReaderSeq attribute*), 302
__annotations__ (*rti.connextdds.IAnyDataWriter attribute*), 304
__annotations__ (*rti.connextdds.IAnyDataWriterSeq attribute*), 305
__annotations__ (*rti.connextdds.IAnyTopic attribute*), 307
__annotations__ (*rti.connextdds.IAnyTopicSeq attribute*), 308
__annotations__ (*rti.connextdds.ICondition attribute*), 310
__annotations__ (*rti.connextdds.IConditionSeq attribute*), 311
__annotations__ (*rti.connextdds.IDataReader attribute*), 313
__annotations__ (*rti.connextdds.IEntity attribute*), 313
__annotations__ (*rti.connextdds.IEntitySeq attribute*), 314
__annotations__ (*rti.connextdds.IgnoredEntityReplacementKind attribute*), 318
__annotations__ (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 317
__annotations__ (*rti.connextdds.IllegalOperationError attribute*), 319
__annotations__ (*rti.connextdds.ImmutablePolicyError attribute*), 319
__annotations__ (*rti.connextdds.InconsistentPolicyError attribute*), 319
__annotations__ (*rti.connextdds.InconsistentTopicStatus attribute*), 319
__annotations__ (*rti.connextdds.InstanceHandle attribute*), 320
__annotations__ (*rti.connextdds.InstanceHandleSeq attribute*), 320
__annotations__ (*rti.connextdds.InstanceState attribute*), 323
__annotations__ (*rti.connextdds.InstanceStateConsistencyKind attribute*), 325
__annotations__ (*rti.connextdds.Int8Seq attribute*), 335
__annotations__ (*rti.connextdds.Int8Type attribute*), 337
__annotations__ (*rti.connextdds.Int16Seq attribute*), 326
__annotations__ (*rti.connextdds.Int16Type attribute*), 328
__annotations__ (*rti.connextdds.Int32Seq attribute*), 329
__annotations__ (*rti.connextdds.Int32Type attribute*), 331
__annotations__ (*rti.connextdds.Int32Vector attribute*), 331
__annotations__ (*rti.connextdds.Int64Seq attribute*), 332
__annotations__ (*rti.connextdds.Int64Type attribute*), 334
__annotations__ (*rti.connextdds.InvalidArgumentError attribute*), 337
__annotations__ (*rti.connextdds.InvalidDowncastError attribute*), 338
__annotations__ (*rti.connextdds.InvalidLocalAdvanceNoticeStatus attribute*), 338
__annotations__ (*rti.connextdds.IReadCondition attribute*), 316
__annotations__ (*rti.connextdds.ItemsView attribute*), 338
__annotations__ (*rti.connextdds.ITopicDescription attribute*), 316
__annotations__ (*rti.connextdds.KeysView attribute*), 338
__annotations__ (*rti.connextdds.LatencyBudget attribute*), 339
__annotations__ (*rti.connextdds.Lifespan attribute*), 339
__annotations__ (*rti.connextdds.Liveliness attribute*), 340
__annotations__ (*rti.connextdds.LivelinessChangedStatus attribute*), 341
__annotations__ (*rti.connextdds.LivelinessKind attribute*), 342
__annotations__ (*rti.connextdds.LivelinessKind.LivelinessKind attribute*), 342
__annotations__ (*rti.connextdds.LivelinessLostStatus attribute*), 343
__annotations__ (*rti.connextdds.LoanedDynamicData attribute*), 344
__annotations__ (*rti.connextdds.Locator attribute*), 344
__annotations__ (*rti.connextdds.LocatorFilter attribute*), 345
__annotations__ (*rti.connextdds.LocatorFilterElement attribute*), 345
__annotations__ (*rti.connextdds.LocatorFilterElementSeq attribute*), 346
__annotations__ (*rti.connextdds.LocatorKind attribute*), 350
__annotations__ (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
__annotations__ (*rti.connextdds.LocatorSeq attribute*), 351
__annotations__ (*rti.connextdds.LocatorVector attribute*), 353
__annotations__ (*rti.connextdds.LogCategory attribute*), 356
__annotations__ (*rti.connextdds.LogCategory.LogCategory attribute*), 355
__annotations__ (*rti.connextdds.LogFacility attribute*), 357
__annotations__ (*rti.connextdds.Logger attribute*), 361
__annotations__ (*rti.connextdds.LogLevel attribute*), 360
__annotations__ (*rti.connextdds.LogLevel.LogLevel attribute*), 359
__annotations__ (*rti.connextdds.LogMessage attribute*), 360
__annotations__ (*rti.connextdds.LongDouble attribute*), 362
__annotations__ (*rti.connextdds.Member attribute*), 363

__annotations__ (*rti.connextdds.MemberSeq attribute*), 364
__annotations__ (*rti.connextdds.Monitoring attribute*), 366
__annotations__ (*rti.connextdds.MonitoringDedicatedParticipantSettings attribute*), 367
__annotations__ (*rti.connextdds.MonitoringDistributionSettings attribute*), 368
__annotations__ (*rti.connextdds.MonitoringEventDistributionSettings attribute*), 368
__annotations__ (*rti.connextdds.MonitoringLoggingDistributionSettings attribute*), 369
__annotations__ (*rti.connextdds.MonitoringLoggingForwardingSettings attribute*), 370
__annotations__ (*rti.connextdds.MonitoringMetricSelection attribute*), 370
__annotations__ (*rti.connextdds.MonitoringMetricSelectionSeq attribute*), 371
__annotations__ (*rti.connextdds.MonitoringPeriodicDistributionSettings attribute*), 374
__annotations__ (*rti.connextdds.MonitoringTelemetryData attribute*), 374
__annotations__ (*rti.connextdds.MulticastMapping attribute*), 375
__annotations__ (*rti.connextdds.MulticastMappingSeq attribute*), 376
__annotations__ (*rti.connextdds.MultiChannel attribute*), 375
__annotations__ (*rti.connextdds.NoOpAnyDataReaderListener attribute*), 378
__annotations__ (*rti.connextdds.NoOpAnyDataWriterListener attribute*), 379
__annotations__ (*rti.connextdds.NoOpAnyTopicListener attribute*), 380
__annotations__ (*rti.connextdds.NoOpDataReaderListener attribute*), 381
__annotations__ (*rti.connextdds.NoOpDataWriterListener attribute*), 382
__annotations__ (*rti.connextdds.NoOpDomainParticipantListener attribute*), 383
__annotations__ (*rti.connextdds.NoOpPublisherListener attribute*), 383
__annotations__ (*rti.connextdds.NoOpSubscriberListener attribute*), 383
__annotations__ (*rti.connextdds.NoOpTopicListener attribute*), 383
__annotations__ (*rti.connextdds.NotAllowedBySecurityError attribute*), 384
__annotations__ (*rti.connextdds.NotEnabledError attribute*), 384
__annotations__ (*rti.connextdds.NullReferenceError attribute*), 384
__annotations__ (*rti.connextdds.OctetType attribute*), 384
__annotations__ (*rti.connextdds.OfferedDeadlineMissedStatus attribute*), 384
__annotations__ (*rti.connextdds.OfferedIncompatibleQosStatus attribute*), 385
__annotations__ (*rti.connextdds.OutOfResourcesError attribute*), 385
__annotations__ (*rti.connextdds.Ownership attribute*), 385
__annotations__ (*rti.connextdds.OwnershipKind attribute*), 387
__annotations__ (*rti.connextdds.OwnershipKind.OwnershipKind attribute*), 386
__annotations__ (*rti.connextdds.OwnershipStrength attribute*), 388
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData attribute*), 425
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilter attribute*), 389
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic attribute*), 389
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq attribute*), 390
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReader attribute*), 394
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener attribute*), 399
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector attribute*), 393
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq attribute*), 400
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter attribute*), 402
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener attribute*), 410
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq attribute*), 412
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription attribute*), 414
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample attribute*), 414
__annotations__ (*rti.connextdds.ParticipantBuiltinTopic-*

*Data.LoanedSamples attribute*), 415
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener attribute*), 415
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener attribute*), 416
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener attribute*), 418
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.Sample attribute*), 418
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicDataSeq attribute*), 426
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.SharedSamples attribute*), 419
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq attribute*), 429
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.Topic attribute*), 419
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicDescription attribute*), 420
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicListener attribute*), 421
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq attribute*), 421
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples attribute*), 423
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter attribute*), 424
__annotations__ (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilterHelper attribute*), 424
__annotations__ (*rti.connextdds.Partition attribute*), 431
__annotations__ (*rti.connextdds.PersistentJournalKind attribute*), 432
__annotations__ (*rti.connextdds.PersistentStorageSettings attribute*), 433
__annotations__ (*rti.connextdds.PersistentSynchronizationKind attribute*), 434
__annotations__ (*rti.connextdds.PreconditionNotMetError attribute*), 435
__annotations__ (*rti.connextdds.Presentation attribute*), 435
__annotations__ (*rti.connextdds.PresentationAccessScopeKind attribute*), 437
__annotations__ (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 437
__annotations__ (*rti.connextdds.PrintFormat attribute*), 440
__annotations__ (*rti.connextdds.PrintFormatKind attribute*), 442
__annotations__ (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*), 441
__annotations__ (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
__annotations__ (*rti.connextdds.PrintFormatProperty attribute*), 443
__annotations__ (*rti.connextdds.ProductVersion attribute*), 443
__annotations__ (*rti.connextdds.Property attribute*), 444
__annotations__ (*rti.connextdds.ProtocolVersion attribute*), 446
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData attribute*), 483
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilter attribute*), 446
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic attribute*), 447
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq attribute*), 448
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataReader attribute*), 452
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener attribute*), 457
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector attribute*), 450
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq attribute*), 458
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriter attribute*), 460
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener attribute*), 468
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq attribute*), 469
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.ITopicDescription attribute*), 472
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.LoanedSample attribute*), 472
__annotations__ (*rti.connextdds.PublicationBuiltinTopic-*

*Data.LoanedSamples attribute*), 472
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener attribute*), 473
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener attribute*), 474
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.NoOpTopicListener attribute*), 476
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.Sample attribute*), 476
__annotations__ (*rti.connextdds.PublicationBuiltinTopicDataSeq attribute*), 485
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.SharedSamples attribute*), 477
__annotations__ (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq attribute*), 488
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.Topic attribute*), 477
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.TopicDescription attribute*), 478
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.TopicListener attribute*), 479
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq attribute*), 479
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples attribute*), 481
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter attribute*), 481
__annotations__ (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper attribute*), 482
__annotations__ (*rti.connextdds.PublicationMatchedStatus attribute*), 490
__annotations__ (*rti.connextdds.Publisher attribute*), 494
__annotations__ (*rti.connextdds.PublisherListener attribute*), 495
__annotations__ (*rti.connextdds.PublisherQos attribute*), 495
__annotations__ (*rti.connextdds.PublisherSeq attribute*), 498
__annotations__ (*rti.connextdds.PublishMode attribute*), 491
__annotations__ (*rti.connextdds.PublishModeKind attribute*), 493
__annotations__ (*rti.connextdds.PublishModeKind.PublishModeKind attribute*), 492
__annotations__ (*rti.connextdds.QosPolicyCount attribute*), 499
__annotations__ (*rti.connextdds.QosPolicyCountSeq attribute*), 500
__annotations__ (*rti.connextdds.QosPrintFormat attribute*), 502
__annotations__ (*rti.connextdds.QosProvider attribute*), 503
__annotations__ (*rti.connextdds.QosProviderParams attribute*), 506
__annotations__ (*rti.connextdds.Query attribute*), 507
__annotations__ (*rti.connextdds.QueryCondition attribute*), 508
__annotations__ (*rti.connextdds.Rank attribute*), 509
__annotations__ (*rti.connextdds.ReadCondition attribute*), 509
__annotations__ (*rti.connextdds.ReaderDataLifecycle attribute*), 510
__annotations__ (*rti.connextdds.ReceiverPool attribute*), 511
__annotations__ (*rti.connextdds.Reliability attribute*), 512
__annotations__ (*rti.connextdds.ReliabilityKind attribute*), 514
__annotations__ (*rti.connextdds.ReliabilityKind.ReliabilityKind attribute*), 513
__annotations__ (*rti.connextdds.ReliableReaderActivityChangedStatus attribute*), 515
__annotations__ (*rti.connextdds.ReliableWriterCacheChangedStatus attribute*), 515
__annotations__ (*rti.connextdds.RemoteParticipantPurgeKind attribute*), 517
__annotations__ (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute*), 517
__annotations__ (*rti.connextdds.RequestedDeadlineMissedStatus attribute*), 518
__annotations__ (*rti.connextdds.RequestedIncompatibleQosStatus attribute*), 519
__annotations__ (*rti.connextdds.ResourceLimits attribute*), 519
__annotations__ (*rti.connextdds.RtpsReliableReaderProtocol attribute*), 520
__annotations__ (*rti.connextdds.RtpsReliableWriterProtocol attribute*), 521
__annotations__ (*rti.connextdds.RtpsReservedPortKindMask attribute*), 524
__annotations__ (*rti.connextdds.RtpsWellKnownPorts attribute*), 527
__annotations__ (*rti.connextdds.SampleFlag attribute*), 528

__annotations__ (*rti.connextdds.SampleIdentity attribute*), 530
__annotations__ (*rti.connextdds.SampleInfo attribute*), 531
__annotations__ (*rti.connextdds.SampleLostState attribute*), 534
__annotations__ (*rti.connextdds.SampleLostStatus attribute*), 536
__annotations__ (*rti.connextdds.SampleRejectedState attribute*), 537
__annotations__ (*rti.connextdds.SampleRejectedStatus attribute*), 539
__annotations__ (*rti.connextdds.SampleState attribute*), 540
__annotations__ (*rti.connextdds.SequenceNumber attribute*), 542
__annotations__ (*rti.connextdds.SequenceType attribute*), 543
__annotations__ (*rti.connextdds.Service attribute*), 544
__annotations__ (*rti.connextdds.ServiceKind attribute*), 546
__annotations__ (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
__annotations__ (*rti.connextdds.ServiceRequest attribute*), 582
__annotations__ (*rti.connextdds.ServiceRequestAcceptedStatus attribute*), 583
__annotations__ (*rti.connextdds.ServiceRequest.ContentFilter attribute*), 547
__annotations__ (*rti.connextdds.ServiceRequest.ContentFilteredTopic attribute*), 548
__annotations__ (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq attribute*), 549
__annotations__ (*rti.connextdds.ServiceRequest.DataReader attribute*), 552
__annotations__ (*rti.connextdds.ServiceRequest.DataReaderListener attribute*), 557
__annotations__ (*rti.connextdds.ServiceRequest.DataReader.Selector attribute*), 551
__annotations__ (*rti.connextdds.ServiceRequest.DataReaderSeq attribute*), 558
__annotations__ (*rti.connextdds.ServiceRequest.DataWriter attribute*), 560
__annotations__ (*rti.connextdds.ServiceRequest.DataWriterListener attribute*), 568
__annotations__ (*rti.connextdds.ServiceRequest.DataWriterSeq attribute*), 569
__annotations__ (*rti.connextdds.ServiceRequestId attribute*), 585
__annotations__ (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
__annotations__ (*rti.connextdds.ServiceRequest.ITopicDescription attribute*), 571
__annotations__ (*rti.connextdds.ServiceRequest.LoanedSample attribute*), 572
__annotations__ (*rti.connextdds.ServiceRequest.LoanedSamples attribute*), 572
__annotations__ (*rti.connextdds.ServiceRequest.NoOpDataReaderListener attribute*), 573
__annotations__ (*rti.connextdds.ServiceRequest.NoOpDataWriterListener attribute*), 574
__annotations__ (*rti.connextdds.ServiceRequest.NoOpTopicListener attribute*), 575
__annotations__ (*rti.connextdds.ServiceRequest.Sample attribute*), 576
__annotations__ (*rti.connextdds.ServiceRequestSeq attribute*), 586
__annotations__ (*rti.connextdds.ServiceRequest.SharedSamples attribute*), 576
__annotations__ (*rti.connextdds.ServiceRequestTimestampedSeq attribute*), 588
__annotations__ (*rti.connextdds.ServiceRequest.Topic attribute*), 577
__annotations__ (*rti.connextdds.ServiceRequest.TopicDescription attribute*), 578
__annotations__ (*rti.connextdds.ServiceRequest.TopicListener attribute*), 578
__annotations__ (*rti.connextdds.ServiceRequest.TopicSeq attribute*), 578
__annotations__ (*rti.connextdds.ServiceRequest.ValidLoanedSamples attribute*), 581
__annotations__ (*rti.connextdds.ServiceRequest.WriterContentFilter attribute*), 581
__annotations__ (*rti.connextdds.ServiceRequest.WriterContentFilterHelper attribute*), 582
__annotations__ (*rti.connextdds.StatusCondition attribute*), 590
__annotations__ (*rti.connextdds.StatusMask attribute*), 592
__annotations__ (*rti.connextdds.StreamKind attribute*), 595
__annotations__ (*rti.connextdds.StringMap attribute*), 597
__annotations__ (*rti.connextdds.StringPairSeq attribute*), 598
__annotations__ (*rti.connextdds.StringSeq attribute*), 600
__annotations__ (*rti.connextdds.StringType attribute*), 602
__annotations__ (*rti.connextdds.StructType attribute*), 602
__annotations__ (*rti.connextdds.Subscriber attribute*), 604

__annotations__ (*rti.connextdds.SubscriberListener attribute*), 605
__annotations__ (*rti.connextdds.SubscriberQos attribute*), 606
__annotations__ (*rti.connextdds.SubscriberSeq attribute*), 608
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData attribute*), 647
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter attribute*), 610
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic attribute*), 610
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq attribute*), 611
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader attribute*), 615
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener attribute*), 620
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector attribute*), 614
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq attribute*), 621
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter attribute*), 623
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener attribute*), 632
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq attribute*), 633
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription attribute*), 635
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample attribute*), 636
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples attribute*), 636
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener attribute*), 637
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener attribute*), 638
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpTopicListener attribute*), 639
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.Sample attribute*), 640
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicDataSeq attribute*), 649
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples attribute*), 640
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq attribute*), 651
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.Topic attribute*), 641
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription attribute*), 642
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicListener attribute*), 642
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq attribute*), 643
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples attribute*), 645
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter attribute*), 645
__annotations__ (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper attribute*), 646
__annotations__ (*rti.connextdds.SubscriptionMatchedStatus attribute*), 654
__annotations__ (*rti.connextdds.SuspendedPublication attribute*), 654
__annotations__ (*rti.connextdds.SyslogVerbosity attribute*), 655
__annotations__ (*rti.connextdds.SystemResourceLimits attribute*), 656
__annotations__ (*rti.connextdds.ThreadContext attribute*), 657
__annotations__ (*rti.connextdds.ThreadSettings attribute*), 657
__annotations__ (*rti.connextdds.ThreadSettingsCpuRotationKind attribute*), 659
__annotations__ (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind attribute*), 658
__annotations__ (*rti.connextdds.ThreadSettingsKindMask attribute*), 660
__annotations__ (*rti.connextdds.Time attribute*), 663
__annotations__ (*rti.connextdds.TimeBasedFilter attribute*), 665
__annotations__ (*rti.connextdds.TimeoutError attribute*), 665
__annotations__ (*rti.connextdds.Topic attribute*), 665
__annotations__ (*rti.connextdds.TopicBuiltinTopicData attribute*), 703

__annotations__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilter attribute*), 667
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic attribute*), 667
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq attribute*), 668
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataReader attribute*), 672
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener attribute*), 677
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector attribute*), 671
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq attribute*), 678
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataWriter attribute*), 680
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener attribute*), 688
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq attribute*), 689
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.ITopicDescription attribute*), 692
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.LoanedSample attribute*), 692
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples attribute*), 692
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener attribute*), 693
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener attribute*), 694
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.NoOpTopicListener attribute*), 696
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.Sample attribute*), 696
__annotations__ (*rti.connextdds.TopicBuiltinTopicDataSeq attribute*), 704
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.SharedSamples attribute*), 697
__annotations__ (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq attribute*), 706
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.Topic attribute*), 697
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.TopicDescription attribute*), 698
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.TopicListener attribute*), 698
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.TopicSeq attribute*), 699
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples attribute*), 701
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter attribute*), 701
__annotations__ (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper attribute*), 702
__annotations__ (*rti.connextdds.TopicData attribute*), 709
__annotations__ (*rti.connextdds.TopicDescription attribute*), 710
__annotations__ (*rti.connextdds.TopicListener attribute*), 710
__annotations__ (*rti.connextdds.TopicQos attribute*), 710
__annotations__ (*rti.connextdds.TopicQuery attribute*), 714
__annotations__ (*rti.connextdds.TopicQueryData attribute*), 715
__annotations__ (*rti.connextdds.TopicQueryDispatch attribute*), 716
__annotations__ (*rti.connextdds.TopicQuerySelection attribute*), 716
__annotations__ (*rti.connextdds.TopicQuerySelectionKind attribute*), 718
__annotations__ (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute*), 717
__annotations__ (*rti.connextdds.TopicSeq attribute*), 719
__annotations__ (*rti.connextdds.TransportBuiltin attribute*), 721
__annotations__ (*rti.connextdds.TransportBuiltinMask attribute*), 722
__annotations__ (*rti.connextdds.TransportClassId attribute*), 727
__annotations__ (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
__annotations__ (*rti.connextdds.TransportInfo attribute*), 728
__annotations__ (*rti.connextdds.TransportInfoSeq attribute*), 729
__annotations__ (*rti.connextdds.TransportInfoVector attribute*), 731
__annotations__ (*rti.connextdds.TransportMulticast attribute*), 732
__annotations__ (*rti.connextdds.TransportMulticastKind attribute*), 733

__annotations__ (*rti.connextdds.TransportMulticastKind.TransportMulticastKind attribute*), 733
__annotations__ (*rti.connextdds.TransportMulticastMapping attribute*), 734
__annotations__ (*rti.connextdds.TransportMulticastMappingFunction attribute*), 735
__annotations__ (*rti.connextdds.TransportMulticastSeq attribute*), 736
__annotations__ (*rti.connextdds.TransportMulticastSettings attribute*), 738
__annotations__ (*rti.connextdds.TransportMulticastSettingsSeq attribute*), 738
__annotations__ (*rti.connextdds.TransportPriority attribute*), 741
__annotations__ (*rti.connextdds.TransportSelection attribute*), 741
__annotations__ (*rti.connextdds.TransportUnicast attribute*), 742
__annotations__ (*rti.connextdds.TransportUnicastSettings attribute*), 743
__annotations__ (*rti.connextdds.TransportUnicastSettingsSeq attribute*), 743
__annotations__ (*rti.connextdds.TriggeredConditions attribute*), 746
__annotations__ (*rti.connextdds.TriggeredConditionsIterator attribute*), 746
__annotations__ (*rti.connextdds.TypeConsistencyEnforcement attribute*), 746
__annotations__ (*rti.connextdds.TypeConsistencyEnforcementKind attribute*), 749
__annotations__ (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 748
__annotations__ (*rti.connextdds.TypeKind attribute*), 755
__annotations__ (*rti.connextdds.TypeKind.TypeKind attribute*), 753
__annotations__ (*rti.connextdds.TypeSupport attribute*), 756
__annotations__ (*rti.connextdds.Uint8Seq attribute*), 764
__annotations__ (*rti.connextdds.Uint8Type attribute*), 767
__annotations__ (*rti.connextdds.Uint16Seq attribute*), 756
__annotations__ (*rti.connextdds.Uint16Type attribute*), 759
__annotations__ (*rti.connextdds.Uint32Seq attribute*), 759
__annotations__ (*rti.connextdds.Uint32Type attribute*), 761
__annotations__ (*rti.connextdds.Uint64Seq attribute*), 762
__annotations__ (*rti.connextdds.Uint64Type attribute*), 764
__annotations__ (*rti.connextdds.UnidimensionalCollectionType attribute*), 767
__annotations__ (*rti.connextdds.UnionMember attribute*), 768
__annotations__ (*rti.connextdds.UnionMemberSeq attribute*), 769
__annotations__ (*rti.connextdds.UnionType attribute*), 771
__annotations__ (*rti.connextdds.UnsupportedError attribute*), 771
__annotations__ (*rti.connextdds.UserData attribute*), 772
__annotations__ (*rti.connextdds.UserDataSample attribute*), 772
__annotations__ (*rti.connextdds.ValuesView attribute*), 772
__annotations__ (*rti.connextdds.VendorId attribute*), 773
__annotations__ (*rti.connextdds.Verbosity attribute*), 775
__annotations__ (*rti.connextdds.Verbosity.Verbosity attribute*), 774
__annotations__ (*rti.connextdds.ViewState attribute*), 776
__annotations__ (*rti.connextdds.WaitSet attribute*), 779
__annotations__ (*rti.connextdds.WaitSetProperty attribute*), 780
__annotations__ (*rti.connextdds.WcharSeq attribute*), 781
__annotations__ (*rti.connextdds.WcharType attribute*), 783
__annotations__ (*rti.connextdds.WireProtocol attribute*), 783
__annotations__ (*rti.connextdds.WireProtocolAutoKind attribute*), 786
__annotations__ (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind attribute*), 785
__annotations__ (*rti.connextdds.WriteParams attribute*), 787
__annotations__ (*rti.connextdds.WriterDataLifecycle attribute*), 788
__annotations__ (*rti.connextdds.WstringSeq attribute*), 788
__annotations__ (*rti.connextdds.WStringType attribute*), 778
__bool__() (*rti.connextdds.ActivityContextMask method*), 41
__bool__() (*rti.connextdds.AnyDataReaderSeq method*), 46
__bool__() (*rti.connextdds.AnyDataWriterSeq method*), 50
__bool__() (*rti.connextdds.AnyTopicSeq method*), 53
__bool__() (*rti.connextdds.BoolSeq method*), 58
__bool__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__bool__() (*rti.connextdds.CharSeq method*), 70
__bool__() (*rti.connextdds.compliance.RpcMask method*), 791
__bool__() (*rti.connextdds.compliance.XTypesMask method*), 795
__bool__() (*rti.connextdds.CompressionIdMask method*), 75
__bool__() (*rti.connextdds.ConditionSeq method*), 79
__bool__() (*rti.connextdds.ContentFilteredTopicSeq method*), 82
__bool__() (*rti.connextdds.CookieSeq method*), 85
__bool__() (*rti.connextdds.DataReaderSeq method*), 113
__bool__() (*rti.connextdds.DataWriterSeq method*), 144
__bool__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 158
__bool__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 161
__bool__() (*rti.connextdds.DomainParticipantSeq method*), 183

__bool__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__bool__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 203
__bool__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 214
__bool__() (*rti.connextdds.DynamicDataSeq method*), 259
__bool__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__bool__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__bool__() (*rti.connextdds.EndpointGroupSeq method*), 266
__bool__() (*rti.connextdds.EntitySeq method*), 271
__bool__() (*rti.connextdds.EnumMemberSeq method*), 274
__bool__() (*rti.connextdds.Float32Seq method*), 285
__bool__() (*rti.connextdds.Float64Seq method*), 288
__bool__() (*rti.connextdds.Float128Seq method*), 282
__bool__() (*rti.connextdds.IAnyDataReaderSeq method*), 302
__bool__() (*rti.connextdds.IAnyDataWriterSeq method*), 305
__bool__() (*rti.connextdds.IAnyTopicSeq method*), 308
__bool__() (*rti.connextdds.IConditionSeq method*), 311
__bool__() (*rti.connextdds.IEntitySeq method*), 314
__bool__() (*rti.connextdds.InstanceHandle method*), 320
__bool__() (*rti.connextdds.InstanceHandleSeq method*), 320
__bool__() (*rti.connextdds.InstanceState method*), 323
__bool__() (*rti.connextdds.Int8Seq method*), 335
__bool__() (*rti.connextdds.Int16Seq method*), 326
__bool__() (*rti.connextdds.Int32Seq method*), 329
__bool__() (*rti.connextdds.Int64Seq method*), 332
__bool__() (*rti.connextdds.LocatorFilterElementSeq method*), 346
__bool__() (*rti.connextdds.LocatorSeq method*), 351
__bool__() (*rti.connextdds.MemberSeq method*), 364
__bool__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 371
__bool__() (*rti.connextdds.MulticastMappingSeq method*), 376
__bool__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 390
__bool__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__bool__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__bool__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 426
__bool__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__bool__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 421
__bool__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 448
__bool__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__bool__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 469
__bool__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 485
__bool__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__bool__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 479
__bool__() (*rti.connextdds.PublisherSeq method*), 498
__bool__() (*rti.connextdds.QosPolicyCountSeq method*), 500
__bool__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__bool__() (*rti.connextdds.SampleFlag method*), 528
__bool__() (*rti.connextdds.SampleLostState method*), 534
__bool__() (*rti.connextdds.SampleRejectedState method*), 537
__bool__() (*rti.connextdds.SampleState method*), 540
__bool__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 549
__bool__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 558
__bool__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 569
__bool__() (*rti.connextdds.ServiceRequestSeq method*), 586
__bool__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__bool__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 578
__bool__() (*rti.connextdds.StatusMask method*), 592
__bool__() (*rti.connextdds.StreamKind method*), 595
__bool__() (*rti.connextdds.StringMap method*), 597
__bool__() (*rti.connextdds.StringPairSeq method*), 598
__bool__() (*rti.connextdds.StringSeq method*), 600
__bool__() (*rti.connextdds.SubscriberSeq method*), 608
__bool__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 611
__bool__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 621
__bool__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 633
__bool__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 649
__bool__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 651
__bool__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__bool__() (*rti.connextdds.ThreadSettingsKindMask method*), 660

__bool__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__bool__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 678
__bool__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 689
__bool__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 704
__bool__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__bool__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__bool__() (*rti.connextdds.TopicSeq method*), 719
__bool__() (*rti.connextdds.TransportBuiltinMask method*), 722
__bool__() (*rti.connextdds.TransportInfoSeq method*), 729
__bool__() (*rti.connextdds.TransportMulticastSeq method*), 736
__bool__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__bool__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 743
__bool__() (*rti.connextdds.Uint8Seq method*), 764
__bool__() (*rti.connextdds.Uint16Seq method*), 756
__bool__() (*rti.connextdds.Uint32Seq method*), 759
__bool__() (*rti.connextdds.Uint64Seq method*), 762
__bool__() (*rti.connextdds.UnionMemberSeq method*), 769
__bool__() (*rti.connextdds.ViewState method*), 776
__bool__() (*rti.connextdds.WcharSeq method*), 781
__bool__() (*rti.connextdds.WstringSeq method*), 788
__contains__() (*rti.connextdds.ActivityContextMask method*), 41
__contains__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__contains__() (*rti.connextdds.AnyDataWriterSeq method*), 50
__contains__() (*rti.connextdds.AnyTopicSeq method*), 53
__contains__() (*rti.connextdds.BoolSeq method*), 58
__contains__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__contains__() (*rti.connextdds.CharSeq method*), 70
__contains__() (*rti.connextdds.compliance.RpcMask method*), 791
__contains__() (*rti.connextdds.compliance.XTypesMask method*), 795
__contains__() (*rti.connextdds.CompressionIdMask method*), 75
__contains__() (*rti.connextdds.ConditionSeq method*), 79
__contains__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__contains__() (*rti.connextdds.CookieSeq method*), 85
__contains__() (*rti.connextdds.DataReaderSeq method*), 113
__contains__() (*rti.connextdds.DataTag method*), 119
__contains__() (*rti.connextdds.DataWriterSeq method*), 144
__contains__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 158
__contains__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 161
__contains__() (*rti.connextdds.DomainParticipantSeq method*), 183
__contains__() (*rti.connextdds.DynamicData method*), 229
__contains__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__contains__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 203
__contains__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 214
__contains__() (*rti.connextdds.DynamicData.FieldsView method*), 217
__contains__() (*rti.connextdds.DynamicData.ItemsView method*), 218
__contains__() (*rti.connextdds.DynamicDataSeq method*), 259
__contains__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__contains__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__contains__() (*rti.connextdds.EndpointGroupSeq method*), 266
__contains__() (*rti.connextdds.EntitySeq method*), 271
__contains__() (*rti.connextdds.EnumMemberSeq method*), 274
__contains__() (*rti.connextdds.Float32Seq method*), 285
__contains__() (*rti.connextdds.Float64Seq method*), 288
__contains__() (*rti.connextdds.Float128Seq method*), 282
__contains__() (*rti.connextdds.IAnyDataReaderSeq method*), 302
__contains__() (*rti.connextdds.IAnyDataWriterSeq method*), 305
__contains__() (*rti.connextdds.IAnyTopicSeq method*), 308
__contains__() (*rti.connextdds.IConditionSeq method*), 311
__contains__() (*rti.connextdds.IEntitySeq method*), 314
__contains__() (*rti.connextdds.InstanceHandleSeq method*), 320
__contains__() (*rti.connextdds.InstanceState method*), 323
__contains__() (*rti.connextdds.Int8Seq method*), 335
__contains__() (*rti.connextdds.Int16Seq method*), 326
__contains__() (*rti.connextdds.Int32Seq method*), 329
__contains__() (*rti.connextdds.Int64Seq method*), 332
__contains__() (*rti.connextdds.KeysView method*), 338
__contains__() (*rti.connextdds.LocatorFilterElementSeq method*), 346
__contains__() (*rti.connextdds.LocatorSeq method*), 351
__contains__() (*rti.connextdds.MemberSeq method*), 364
__contains__()

(*rti.connextdds.MonitoringMetricSelectionSeq method*), 371
__contains__() (*rti.connextdds.MulticastMappingSeq method*), 376
__contains__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 390
__contains__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__contains__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__contains__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 426
__contains__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__contains__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 421
__contains__() (*rti.connextdds.Property method*), 444
__contains__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 448
__contains__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__contains__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__contains__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 485
__contains__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__contains__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 479
__contains__() (*rti.connextdds.PublisherSeq method*), 498
__contains__() (*rti.connextdds.QosPolicyCountSeq method*), 500
__contains__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__contains__() (*rti.connextdds.SampleFlag method*), 528
__contains__() (*rti.connextdds.SampleLostState method*), 534
__contains__() (*rti.connextdds.SampleRejectedState method*), 537
__contains__() (*rti.connextdds.SampleState method*), 540
__contains__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 549
__contains__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 558
__contains__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 569
__contains__() (*rti.connextdds.ServiceRequestSeq method*), 586
__contains__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__contains__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 578
__contains__() (*rti.connextdds.StatusMask method*), 592
__contains__() (*rti.connextdds.StreamKind method*), 595
__contains__() (*rti.connextdds.StringMap method*), 597
__contains__() (*rti.connextdds.StringPairSeq method*), 598
__contains__() (*rti.connextdds.StringSeq method*), 600
__contains__() (*rti.connextdds.SubscriberSeq method*), 608
__contains__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 611
__contains__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 621
__contains__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 633
__contains__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 649
__contains__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 651
__contains__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__contains__() (*rti.connextdds.ThreadSettingsKindMask method*), 660
__contains__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__contains__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 678
__contains__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 689
__contains__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 704
__contains__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__contains__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__contains__() (*rti.connextdds.TopicSeq method*), 719
__contains__() (*rti.connextdds.TransportBuiltinMask method*), 722
__contains__() (*rti.connextdds.TransportInfoSeq method*), 729
__contains__() (*rti.connextdds.TransportMulticastSeq method*), 736

__contains__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__contains__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 743
__contains__() (*rti.connextdds.TriggeredConditions method*), 746
__contains__() (*rti.connextdds.Uint8Seq method*), 765
__contains__() (*rti.connextdds.Uint16Seq method*), 756
__contains__() (*rti.connextdds.Uint32Seq method*), 759
__contains__() (*rti.connextdds.Uint64Seq method*), 762
__contains__() (*rti.connextdds.UnionMemberSeq method*), 769
__contains__() (*rti.connextdds.ViewState method*), 776
__contains__() (*rti.connextdds.WcharSeq method*), 781
__contains__() (*rti.connextdds.WstringSeq method*), 788
__delitem__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__delitem__() (*rti.connextdds.AnyDataWriterSeq method*), 50
__delitem__() (*rti.connextdds.AnyTopicSeq method*), 53
__delitem__() (*rti.connextdds.BoolSeq method*), 58
__delitem__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__delitem__() (*rti.connextdds.CharSeq method*), 70
__delitem__() (*rti.connextdds.ConditionSeq method*), 79
__delitem__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__delitem__() (*rti.connextdds.CookieSeq method*), 85
__delitem__() (*rti.connextdds.DataReaderSeq method*), 113
__delitem__() (*rti.connextdds.DataWriterSeq method*), 144
__delitem__() (*rti.connextdds.DomainParticipantSeq method*), 183
__delitem__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__delitem__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 203
__delitem__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 214
__delitem__() (*rti.connextdds.DynamicDataSeq method*), 259
__delitem__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__delitem__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__delitem__() (*rti.connextdds.EndpointGroupSeq method*), 266
__delitem__() (*rti.connextdds.EntitySeq method*), 271
__delitem__() (*rti.connextdds.EnumMemberSeq method*), 274
__delitem__() (*rti.connextdds.Float32Seq method*), 285
__delitem__() (*rti.connextdds.Float64Seq method*), 288
__delitem__() (*rti.connextdds.Float128Seq method*), 282
__delitem__() (*rti.connextdds.IAnyDataReaderSeq method*), 302
__delitem__() (*rti.connextdds.IAnyDataWriterSeq method*), 305
__delitem__() (*rti.connextdds.IAnyTopicSeq method*), 308
__delitem__() (*rti.connextdds.IConditionSeq method*), 311
__delitem__() (*rti.connextdds.IEntitySeq method*), 314
__delitem__() (*rti.connextdds.InstanceHandleSeq method*), 320
__delitem__() (*rti.connextdds.Int8Seq method*), 335
__delitem__() (*rti.connextdds.Int16Seq method*), 326
__delitem__() (*rti.connextdds.Int32Seq method*), 329
__delitem__() (*rti.connextdds.Int64Seq method*), 332
__delitem__() (*rti.connextdds.LocatorFilterElementSeq method*), 346
__delitem__() (*rti.connextdds.LocatorSeq method*), 352
__delitem__() (*rti.connextdds.MemberSeq method*), 365
__delitem__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 371
__delitem__() (*rti.connextdds.MulticastMappingSeq method*), 376
__delitem__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 390
__delitem__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__delitem__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__delitem__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 426
__delitem__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__delitem__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 421
__delitem__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 448
__delitem__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__delitem__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__delitem__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 485
__delitem__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__delitem__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 479
__delitem__() (*rti.connextdds.PublisherSeq method*), 498
__delitem__() (*rti.connextdds.QosPolicyCountSeq method*), 500
__delitem__() (*rti.connextdds.ServiceRequest.ContentFil-*

*teredTopicSeq method*), 549
__delitem__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 558
__delitem__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__delitem__() (*rti.connextdds.ServiceRequestSeq method*), 586
__delitem__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__delitem__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__delitem__() (*rti.connextdds.StringMap method*), 597
__delitem__() (*rti.connextdds.StringPairSeq method*), 598
__delitem__() (*rti.connextdds.StringSeq method*), 600
__delitem__() (*rti.connextdds.SubscriberSeq method*), 608
__delitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__delitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 621
__delitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 633
__delitem__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 649
__delitem__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__delitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__delitem__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__delitem__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 678
__delitem__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__delitem__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 704
__delitem__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__delitem__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__delitem__() (*rti.connextdds.TopicSeq method*), 719
__delitem__() (*rti.connextdds.TransportInfoSeq method*), 729
__delitem__() (*rti.connextdds.TransportMulticastSeq method*), 736
__delitem__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__delitem__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 743
__delitem__() (*rti.connextdds.Uint8Seq method*), 765
__delitem__() (*rti.connextdds.Uint16Seq method*), 757
__delitem__() (*rti.connextdds.Uint32Seq method*), 759
__delitem__() (*rti.connextdds.Uint64Seq method*), 762
__delitem__() (*rti.connextdds.UnionMemberSeq method*), 769
__delitem__() (*rti.connextdds.WcharSeq method*), 781
__delitem__() (*rti.connextdds.WstringSeq method*), 789
__enter__() (*rti.connextdds.CoherentAccess method*), 73
__enter__() (*rti.connextdds.CoherentSet method*), 73
__enter__() (*rti.connextdds.DataReader method*), 91
__enter__() (*rti.connextdds.DataReader.LoanedSamples method*), 89
__enter__() (*rti.connextdds.DataWriter method*), 120
__enter__() (*rti.connextdds.DomainParticipant method*), 164
__enter__() (*rti.connextdds.DynamicData.DataReader method*), 197
__enter__() (*rti.connextdds.DynamicData.DataWriter method*), 205
__enter__() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
__enter__() (*rti.connextdds.DynamicData.ValidLoanedSamples method*), 227
__enter__() (*rti.connextdds.EntityLock method*), 270
__enter__() (*rti.connextdds.IReadCondition method*), 316
__enter__() (*rti.connextdds.LoanedDynamicData method*), 344
__enter__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 394
__enter__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 402
__enter__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
__enter__() (*rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples method*), 423
__enter__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 452
__enter__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 460
__enter__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 472
__enter__() (*rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples method*), 481
__enter__() (*rti.connextdds.ServiceRequest.DataReader method*), 552

__enter__() (*rti.connextdds.ServiceRequest.DataWriter method*), 560
__enter__() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 572
__enter__() (*rti.connextdds.ServiceRequest.ValidLoanedSamples method*), 581
__enter__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 615
__enter__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 624
__enter__() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples method*), 636
__enter__() (*rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples method*), 645
__enter__() (*rti.connextdds.SuspendedPublication method*), 654
__enter__() (*rti.connextdds.ThreadContext method*), 657
__enter__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 672
__enter__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 680
__enter__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 692
__enter__() (*rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples method*), 701
__enter__() (*rti.connextdds.TopicQuery method*), 714
__eq__() (*rti.connextdds.AcknowledgmentKind method*), 40
__eq__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 39
__eq__() (*rti.connextdds.ACTEnumMember method*), 34
__eq__() (*rti.connextdds.ActivityContextMask method*), 41
__eq__() (*rti.connextdds.ACTMember method*), 35
__eq__() (*rti.connextdds.ACTUnionMember method*), 36
__eq__() (*rti.connextdds.AliasType method*), 44
__eq__() (*rti.connextdds.AllocationSettings method*), 45
__eq__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__eq__() (*rti.connextdds.AnyDataWriterSeq method*), 50
__eq__() (*rti.connextdds.AnyTopicSeq method*), 53
__eq__() (*rti.connextdds.ArrayType method*), 55
__eq__() (*rti.connextdds.AsynchronousPublisher method*), 55
__eq__() (*rti.connextdds.Availability method*), 56
__eq__() (*rti.connextdds.Batch method*), 57
__eq__() (*rti.connextdds.BoolSeq method*), 58
__eq__() (*rti.connextdds.BoolType method*), 60
__eq__() (*rti.connextdds.BuiltinTopicKey method*), 63
__eq__() (*rti.connextdds.BuiltinTopicReaderResourceLimits method*), 63
__eq__() (*rti.connextdds.ByteVector method*), 64
__eq__() (*rti.connextdds.CdrPaddingKind method*), 66
__eq__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__eq__() (*rti.connextdds.ChannelSettings method*), 67
__eq__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__eq__() (*rti.connextdds.CharSeq method*), 70
__eq__() (*rti.connextdds.CharType method*), 72
__eq__() (*rti.connextdds.CoherentSetInfo method*), 73
__eq__() (*rti.connextdds.CollectionType method*), 74
__eq__() (*rti.connextdds.compliance.RpcMask method*), 791
__eq__() (*rti.connextdds.compliance.XTypesMask method*), 795
__eq__() (*rti.connextdds.CompressionIdMask method*), 75
__eq__() (*rti.connextdds.CompressionSettings method*), 78
__eq__() (*rti.connextdds.ConditionSeq method*), 79
__eq__() (*rti.connextdds.ContentFilteredTopic method*), 81
__eq__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__eq__() (*rti.connextdds.ContentFilterProperty method*), 81
__eq__() (*rti.connextdds.Cookie method*), 85
__eq__() (*rti.connextdds.CookieSeq method*), 86
__eq__() (*rti.connextdds.CookieVector method*), 87
__eq__() (*rti.connextdds.Database method*), 147
__eq__() (*rti.connextdds.DataReader method*), 91
__eq__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__eq__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__eq__() (*rti.connextdds.DataReaderProtocol method*), 101
__eq__() (*rti.connextdds.DataReaderQos method*), 103
__eq__() (*rti.connextdds.DataReaderResourceLimits method*), 110
__eq__() (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings method*), 112
__eq__() (*rti.connextdds.DataReaderSeq method*), 114
__eq__() (*rti.connextdds.DataRepresentation method*), 115
__eq__() (*rti.connextdds.DataState method*), 116
__eq__() (*rti.connextdds.DataStateEx method*), 117
__eq__() (*rti.connextdds.DataTag method*), 119
__eq__() (*rti.connextdds.DataWriter method*), 120
__eq__() (*rti.connextdds.DataWriterProtocol method*), 128
__eq__() (*rti.connextdds.DataWriterQos method*), 131
__eq__() (*rti.connextdds.DataWriterResourceLimits method*), 139
__eq__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__eq__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 142
__eq__() (*rti.connextdds.DataWriterSeq method*), 145
__eq__() (*rti.connextdds.DataWriterShmemRefTransferModeSettings method*), 146
__eq__() (*rti.connextdds.DataWriterTransferMode method*), 147
__eq__() (*rti.connextdds.Deadline method*), 148
__eq__() (*rti.connextdds.DestinationOrder method*), 149

__eq__() (*rti.connextdds.DestinationOrderKind method*), 150
__eq__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__eq__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__eq__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__eq__() (*rti.connextdds.Discovery method*), 154
__eq__() (*rti.connextdds.DiscoveryConfig method*), 155
__eq__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 158
__eq__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 161
__eq__() (*rti.connextdds.DomainParticipant method*), 164
__eq__() (*rti.connextdds.DomainParticipantConfigParams method*), 171
__eq__() (*rti.connextdds.DomainParticipantFactoryQos method*), 172
__eq__() (*rti.connextdds.DomainParticipantQos method*), 174
__eq__() (*rti.connextdds.DomainParticipantResourceLimits method*), 178
__eq__() (*rti.connextdds.DomainParticipantSeq method*), 183
__eq__() (*rti.connextdds.Durability method*), 186
__eq__() (*rti.connextdds.DurabilityKind method*), 188
__eq__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 187
__eq__() (*rti.connextdds.DurabilityService method*), 189
__eq__() (*rti.connextdds.Duration method*), 190
__eq__() (*rti.connextdds.DynamicData method*), 229
__eq__() (*rti.connextdds.DynamicData.ContentFilteredTopic method*), 193
__eq__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__eq__() (*rti.connextdds.DynamicData.DataReader method*), 197
__eq__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 203
__eq__() (*rti.connextdds.DynamicData.DataWriter method*), 205
__eq__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 214
__eq__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__eq__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__eq__() (*rti.connextdds.DynamicDataInfo method*), 257
__eq__() (*rti.connextdds.DynamicDataMemberInfo method*), 257
__eq__() (*rti.connextdds.DynamicDataProperty method*), 258
__eq__() (*rti.connextdds.DynamicDataSeq method*), 259
__eq__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__eq__() (*rti.connextdds.DynamicData.Topic method*), 223
__eq__() (*rti.connextdds.DynamicData.TopicDescription method*), 224
__eq__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__eq__() (*rti.connextdds.DynamicDataTypeSerializationProperty method*), 263
__eq__() (*rti.connextdds.DynamicType method*), 264
__eq__() (*rti.connextdds.DynamicTypePrintFormatProperty method*), 265
__eq__() (*rti.connextdds.EndpointGroupSeq method*), 266
__eq__() (*rti.connextdds.EndpointGroupVector method*), 268
__eq__() (*rti.connextdds.EntityFactory method*), 269
__eq__() (*rti.connextdds.EntityName method*), 270
__eq__() (*rti.connextdds.EntitySeq method*), 271
__eq__() (*rti.connextdds.EnumMember method*), 273
__eq__() (*rti.connextdds.EnumMemberSeq method*), 274
__eq__() (*rti.connextdds.EnumType method*), 276
__eq__() (*rti.connextdds.Event method*), 277
__eq__() (*rti.connextdds.ExpressionProperty method*), 278
__eq__() (*rti.connextdds.ExtensibilityKind method*), 280
__eq__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 279
__eq__() (*rti.connextdds.FilterSampleInfo method*), 282
__eq__() (*rti.connextdds.Float32Seq method*), 285
__eq__() (*rti.connextdds.Float32Type method*), 287
__eq__() (*rti.connextdds.Float64Seq method*), 288
__eq__() (*rti.connextdds.Float64Type method*), 290
__eq__() (*rti.connextdds.Float128Seq method*), 283
__eq__() (*rti.connextdds.Float128Type method*), 284
__eq__() (*rti.connextdds.FlowController method*), 290
__eq__() (*rti.connextdds.FlowControllerProperty method*), 291
__eq__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 294
__eq__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 293
__eq__() (*rti.connextdds.FlowControllerTokenBucketProperty method*), 295
__eq__() (*rti.connextdds.GroupData method*), 296
__eq__() (*rti.connextdds.GuardCondition method*), 297
__eq__() (*rti.connextdds.Guid method*), 298
__eq__() (*rti.connextdds.heap_monitoring.HeapMonitoringParams method*), 798
__eq__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__eq__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__eq__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 802
__eq__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__eq__() (*rti.connextdds.History method*), 298

__eq__() (*rti.connextdds.HistoryKind method*), 300
__eq__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__eq__() (*rti.connextdds.IAnyDataReader method*), 301
__eq__() (*rti.connextdds.IAnyDataReaderSeq method*), 302
__eq__() (*rti.connextdds.IAnyDataWriter method*), 304
__eq__() (*rti.connextdds.IAnyDataWriterSeq method*), 305
__eq__() (*rti.connextdds.IAnyTopic method*), 307
__eq__() (*rti.connextdds.IAnyTopicSeq method*), 308
__eq__() (*rti.connextdds.ICondition method*), 310
__eq__() (*rti.connextdds.IConditionSeq method*), 311
__eq__() (*rti.connextdds.IEntity method*), 313
__eq__() (*rti.connextdds.IEntitySeq method*), 314
__eq__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 318
__eq__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 317
__eq__() (*rti.connextdds.InstanceHandle method*), 320
__eq__() (*rti.connextdds.InstanceHandleSeq method*), 321
__eq__() (*rti.connextdds.InstanceState method*), 323
__eq__() (*rti.connextdds.InstanceStateConsistencyKind method*), 325
__eq__() (*rti.connextdds.Int8Seq method*), 335
__eq__() (*rti.connextdds.Int8Type method*), 337
__eq__() (*rti.connextdds.Int16Seq method*), 326
__eq__() (*rti.connextdds.Int16Type method*), 328
__eq__() (*rti.connextdds.Int32Seq method*), 329
__eq__() (*rti.connextdds.Int32Type method*), 331
__eq__() (*rti.connextdds.Int32Vector method*), 331
__eq__() (*rti.connextdds.Int64Seq method*), 333
__eq__() (*rti.connextdds.Int64Type method*), 335
__eq__() (*rti.connextdds.IReadCondition method*), 316
__eq__() (*rti.connextdds.LatencyBudget method*), 339
__eq__() (*rti.connextdds.Lifespan method*), 339
__eq__() (*rti.connextdds.Liveliness method*), 340
__eq__() (*rti.connextdds.LivelinessKind method*), 342
__eq__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__eq__() (*rti.connextdds.Locator method*), 344
__eq__() (*rti.connextdds.LocatorFilter method*), 345
__eq__() (*rti.connextdds.LocatorFilterElement method*), 345
__eq__() (*rti.connextdds.LocatorFilterElementSeq method*), 346
__eq__() (*rti.connextdds.LocatorKind method*), 350
__eq__() (*rti.connextdds.LocatorKind.LocatorKind method*), 349
__eq__() (*rti.connextdds.LocatorSeq method*), 352
__eq__() (*rti.connextdds.LocatorVector method*), 353
__eq__() (*rti.connextdds.LogCategory method*), 356
__eq__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__eq__() (*rti.connextdds.LogFacility method*), 357
__eq__() (*rti.connextdds.LogLevel method*), 360
__eq__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__eq__() (*rti.connextdds.LongDouble method*), 362
__eq__() (*rti.connextdds.Member method*), 363
__eq__() (*rti.connextdds.MemberSeq method*), 365
__eq__() (*rti.connextdds.Monitoring method*), 366
__eq__() (*rti.connextdds.MonitoringDedicatedParticipantSettings method*), 367
__eq__() (*rti.connextdds.MonitoringDistributionSettings method*), 368
__eq__() (*rti.connextdds.MonitoringEventDistributionSettings method*), 369
__eq__() (*rti.connextdds.MonitoringLoggingDistributionSettings method*), 369
__eq__() (*rti.connextdds.MonitoringLoggingForwardingSettings method*), 370
__eq__() (*rti.connextdds.MonitoringMetricSelection method*), 370
__eq__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__eq__() (*rti.connextdds.MonitoringPeriodicDistributionSettings method*), 374
__eq__() (*rti.connextdds.MonitoringTelemetryData method*), 374
__eq__() (*rti.connextdds.MulticastMapping method*), 375
__eq__() (*rti.connextdds.MulticastMappingSeq method*), 377
__eq__() (*rti.connextdds.MultiChannel method*), 375
__eq__() (*rti.connextdds.OctetType method*), 384
__eq__() (*rti.connextdds.Ownership method*), 385
__eq__() (*rti.connextdds.OwnershipKind method*), 387
__eq__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__eq__() (*rti.connextdds.OwnershipStrength method*), 388
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData method*), 425
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 389
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 394
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 402
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__eq__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__eq__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.Topic method*), 419
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.Top-*

*icDescription method*), 420
__eq__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 421
__eq__() (*rti.connextdds.Partition method*), 431
__eq__() (*rti.connextdds.PersistentJournalKind method*), 432
__eq__() (*rti.connextdds.PersistentStorageSettings method*), 433
__eq__() (*rti.connextdds.PersistentSynchronizationKind method*), 434
__eq__() (*rti.connextdds.Presentation method*), 435
__eq__() (*rti.connextdds.PresentationAccessScopeKind method*), 437
__eq__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__eq__() (*rti.connextdds.PrintFormat method*), 440
__eq__() (*rti.connextdds.PrintFormatKind method*), 442
__eq__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 441
__eq__() (*rti.connextdds.PrintFormat.PrintFormat method*), 439
__eq__() (*rti.connextdds.ProductVersion method*), 443
__eq__() (*rti.connextdds.Property method*), 444
__eq__() (*rti.connextdds.ProtocolVersion method*), 446
__eq__() (*rti.connextdds.PublicationBuiltinTopicData method*), 483
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 447
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 448
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 452
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 460
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__eq__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__eq__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.Topic method*), 477
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.TopicDescription method*), 478
__eq__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 479
__eq__() (*rti.connextdds.Publisher method*), 494
__eq__() (*rti.connextdds.PublisherQos method*), 495
__eq__() (*rti.connextdds.PublisherSeq method*), 498
__eq__() (*rti.connextdds.PublishMode method*), 491
__eq__() (*rti.connextdds.PublishModeKind method*), 493
__eq__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__eq__() (*rti.connextdds.QosPolicyCount method*), 500
__eq__() (*rti.connextdds.QosPolicyCountSeq method*), 500
__eq__() (*rti.connextdds.QosPrintFormat method*), 502
__eq__() (*rti.connextdds.QosProvider method*), 503
__eq__() (*rti.connextdds.QosProviderParams method*), 506
__eq__() (*rti.connextdds.Query method*), 507
__eq__() (*rti.connextdds.QueryCondition method*), 508
__eq__() (*rti.connextdds.ReaderDataLifecycle method*), 510
__eq__() (*rti.connextdds.ReceiverPool method*), 511
__eq__() (*rti.connextdds.Reliability method*), 512
__eq__() (*rti.connextdds.ReliabilityKind method*), 514
__eq__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 513
__eq__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 517
__eq__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__eq__() (*rti.connextdds.ResourceLimits method*), 519
__eq__() (*rti.connextdds.RtpsReliableReaderProtocol method*), 520
__eq__() (*rti.connextdds.RtpsReliableWriterProtocol method*), 521
__eq__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__eq__() (*rti.connextdds.RtpsWellKnownPorts method*), 527
__eq__() (*rti.connextdds.SampleFlag method*), 528
__eq__() (*rti.connextdds.SampleIdentity method*), 530
__eq__() (*rti.connextdds.SampleInfo method*), 531
__eq__() (*rti.connextdds.SampleLostState method*), 534
__eq__() (*rti.connextdds.SampleRejectedState method*), 537
__eq__() (*rti.connextdds.SampleState method*), 540
__eq__() (*rti.connextdds.SequenceNumber method*), 542
__eq__() (*rti.connextdds.SequenceType method*), 544
__eq__() (*rti.connextdds.Service method*), 544
__eq__() (*rti.connextdds.ServiceKind method*), 546
__eq__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__eq__() (*rti.connextdds.ServiceRequest method*), 582
__eq__() (*rti.connextdds.ServiceRequest.ContentFilteredTopic method*), 548
__eq__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 549
__eq__() (*rti.connextdds.ServiceRequest.DataReader method*), 553
__eq__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__eq__() (*rti.connextdds.ServiceRequest.DataWriter method*), 560
__eq__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__eq__() (*rti.connextdds.ServiceRequestId method*), 585

__eq__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__eq__() (*rti.connextdds.ServiceRequestSeq method*), 586
__eq__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__eq__() (*rti.connextdds.ServiceRequest.Topic method*), 577
__eq__() (*rti.connextdds.ServiceRequest.TopicDescription method*), 578
__eq__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__eq__() (*rti.connextdds.StatusCondition method*), 590
__eq__() (*rti.connextdds.StatusMask method*), 592
__eq__() (*rti.connextdds.StreamKind method*), 595
__eq__() (*rti.connextdds.StringPairSeq method*), 598
__eq__() (*rti.connextdds.StringSeq method*), 600
__eq__() (*rti.connextdds.StringType method*), 602
__eq__() (*rti.connextdds.StructType method*), 603
__eq__() (*rti.connextdds.Subscriber method*), 604
__eq__() (*rti.connextdds.SubscriberQos method*), 606
__eq__() (*rti.connextdds.SubscriberSeq method*), 608
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData method*), 647
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 610
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 615
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 624
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 649
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.Topic method*), 641
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription method*), 642
__eq__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__eq__() (*rti.connextdds.SyslogVerbosity method*), 655
__eq__() (*rti.connextdds.SystemResourceLimits method*), 656
__eq__() (*rti.connextdds.ThreadSettings method*), 657
__eq__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 659
__eq__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 658
__eq__() (*rti.connextdds.ThreadSettingsKindMask method*), 660
__eq__() (*rti.connextdds.Time method*), 663
__eq__() (*rti.connextdds.TimeBasedFilter method*), 665
__eq__() (*rti.connextdds.Topic method*), 666
__eq__() (*rti.connextdds.TopicBuiltinTopicData method*), 703
__eq__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 667
__eq__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__eq__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 672
__eq__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 678
__eq__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 680
__eq__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__eq__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 704
__eq__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__eq__() (*rti.connextdds.TopicBuiltinTopicData.Topic method*), 697
__eq__() (*rti.connextdds.TopicBuiltinTopicData.TopicDescription method*), 698
__eq__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__eq__() (*rti.connextdds.TopicData method*), 709
__eq__() (*rti.connextdds.TopicDescription method*), 710
__eq__() (*rti.connextdds.TopicQos method*), 710
__eq__() (*rti.connextdds.TopicQuery method*), 714
__eq__() (*rti.connextdds.TopicQueryDispatch method*), 716
__eq__() (*rti.connextdds.TopicQuerySelectionKind method*), 718
__eq__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 717
__eq__() (*rti.connextdds.TopicSeq method*), 719
__eq__() (*rti.connextdds.TransportBuiltin method*), 721
__eq__() (*rti.connextdds.TransportBuiltinMask method*), 722
__eq__() (*rti.connextdds.TransportClassId method*), 727
__eq__() (*rti.connextdds.TransportClassId.TransportClassId method*), 726
__eq__() (*rti.connextdds.TransportInfoSeq method*), 729
__eq__() (*rti.connextdds.TransportInfoVector method*), 731
__eq__() (*rti.connextdds.TransportMulticast method*), 732
__eq__() (*rti.connextdds.TransportMulticastKind method*), 733

__eq__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__eq__() (*rti.connextdds.TransportMulticastMapping method*), 734
__eq__() (*rti.connextdds.TransportMulticastMappingFunction method*), 735
__eq__() (*rti.connextdds.TransportMulticastSeq method*), 736
__eq__() (*rti.connextdds.TransportMulticastSettings method*), 738
__eq__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__eq__() (*rti.connextdds.TransportPriority method*), 741
__eq__() (*rti.connextdds.TransportSelection method*), 741
__eq__() (*rti.connextdds.TransportUnicast method*), 742
__eq__() (*rti.connextdds.TransportUnicastSettings method*), 743
__eq__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__eq__() (*rti.connextdds.TypeConsistencyEnforcement method*), 746
__eq__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 749
__eq__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 748
__eq__() (*rti.connextdds.TypeKind method*), 755
__eq__() (*rti.connextdds.TypeKind.TypeKind method*), 753
__eq__() (*rti.connextdds.TypeSupport method*), 756
__eq__() (*rti.connextdds.Uint8Seq method*), 765
__eq__() (*rti.connextdds.Uint8Type method*), 767
__eq__() (*rti.connextdds.Uint16Seq method*), 757
__eq__() (*rti.connextdds.Uint16Type method*), 759
__eq__() (*rti.connextdds.Uint32Seq method*), 759
__eq__() (*rti.connextdds.Uint32Type method*), 761
__eq__() (*rti.connextdds.Uint64Seq method*), 762
__eq__() (*rti.connextdds.Uint64Type method*), 764
__eq__() (*rti.connextdds.UnidimensionalCollectionType method*), 767
__eq__() (*rti.connextdds.UnionMember method*), 768
__eq__() (*rti.connextdds.UnionMemberSeq method*), 769
__eq__() (*rti.connextdds.UnionType method*), 771
__eq__() (*rti.connextdds.UserData method*), 772
__eq__() (*rti.connextdds.VendorId method*), 773
__eq__() (*rti.connextdds.Verbosity method*), 775
__eq__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__eq__() (*rti.connextdds.ViewState method*), 776
__eq__() (*rti.connextdds.WaitSetProperty method*), 780
__eq__() (*rti.connextdds.WcharSeq method*), 781
__eq__() (*rti.connextdds.WcharType method*), 783
__eq__() (*rti.connextdds.WireProtocol method*), 783
__eq__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__eq__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__eq__() (*rti.connextdds.WriterDataLifecycle method*), 788
__eq__() (*rti.connextdds.WstringSeq method*), 789
__eq__() (*rti.connextdds.WStringType method*), 778
__exit__() (*rti.connextdds.CoherentAccess method*), 73
__exit__() (*rti.connextdds.CoherentSet method*), 73
__exit__() (*rti.connextdds.DataReader method*), 91
__exit__() (*rti.connextdds.DataReader.LoanedSamples method*), 89
__exit__() (*rti.connextdds.DataWriter method*), 120
__exit__() (*rti.connextdds.DomainParticipant method*), 164
__exit__() (*rti.connextdds.DynamicData.DataReader method*), 197
__exit__() (*rti.connextdds.DynamicData.DataWriter method*), 205
__exit__() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
__exit__() (*rti.connextdds.DynamicData.ValidLoanedSamples method*), 227
__exit__() (*rti.connextdds.EntityLock method*), 270
__exit__() (*rti.connextdds.IReadCondition method*), 316
__exit__() (*rti.connextdds.LoanedDynamicData method*), 344
__exit__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 394
__exit__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 402
__exit__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
__exit__() (*rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples method*), 423
__exit__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 452
__exit__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 460
__exit__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 473
__exit__() (*rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples method*), 481
__exit__() (*rti.connextdds.ServiceRequest.DataReader method*), 553
__exit__() (*rti.connextdds.ServiceRequest.DataWriter method*), 561
__exit__() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 572
__exit__() (*rti.connextdds.ServiceRequest.ValidLoanedSamples method*), 581
__exit__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 615
__exit__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 624
__exit__() (*rti.connextdds.SubscriptionBuiltinTopic-*

*Data.LoanedSamples method*), 636
__exit__() (*rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples method*), 645
__exit__() (*rti.connextdds.SuspendedPublication method*), 655
__exit__() (*rti.connextdds.ThreadContext method*), 657
__exit__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 672
__exit__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 680
__exit__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 693
__exit__() (*rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples method*), 701
__exit__() (*rti.connextdds.TopicQuery method*), 714
__float__() (*rti.connextdds.Duration method*), 190
__ge__() (*rti.connextdds.AcknowledgmentKind method*), 40
__ge__() (*rti.connextdds.CdrPaddingKind method*), 66
__ge__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__ge__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__ge__() (*rti.connextdds.DestinationOrderKind method*), 151
__ge__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__ge__() (*rti.connextdds.DurabilityKind method*), 188
__ge__() (*rti.connextdds.Duration method*), 190
__ge__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__ge__() (*rti.connextdds.ExtensibilityKind method*), 280
__ge__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 294
__ge__() (*rti.connextdds.Guid method*), 298
__ge__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__ge__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 802
__ge__() (*rti.connextdds.HistoryKind method*), 300
__ge__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 318
__ge__() (*rti.connextdds.LivelinessKind method*), 342
__ge__() (*rti.connextdds.LocatorKind method*), 351
__ge__() (*rti.connextdds.LogCategory method*), 356
__ge__() (*rti.connextdds.LogLevel method*), 360
__ge__() (*rti.connextdds.OwnershipKind method*), 387
__ge__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__ge__() (*rti.connextdds.PrintFormat method*), 440
__ge__() (*rti.connextdds.PrintFormatKind method*), 442
__ge__() (*rti.connextdds.PublishModeKind method*), 493
__ge__() (*rti.connextdds.ReliabilityKind method*), 514
__ge__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 517
__ge__() (*rti.connextdds.SequenceNumber method*), 542
__ge__() (*rti.connextdds.ServiceKind method*), 547
__ge__() (*rti.connextdds.ServiceRequestId method*), 585
__ge__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 659
__ge__() (*rti.connextdds.Time method*), 663
__ge__() (*rti.connextdds.TopicQuerySelectionKind method*), 718
__ge__() (*rti.connextdds.TransportClassId method*), 727
__ge__() (*rti.connextdds.TransportMulticastKind method*), 733
__ge__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 749
__ge__() (*rti.connextdds.TypeKind method*), 755
__ge__() (*rti.connextdds.Verbosity method*), 775
__ge__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__getattr__() (*rti.connextdds.ACTEnumMember method*), 34
__getitem__() (*rti.connextdds.ACTEnumMember method*), 34
__getitem__() (*rti.connextdds.ActivityContextMask method*), 41
__getitem__() (*rti.connextdds.ACTMember method*), 35
__getitem__() (*rti.connextdds.ACTUnionMember method*), 36
__getitem__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__getitem__() (*rti.connextdds.AnyDataWriterSeq method*), 50
__getitem__() (*rti.connextdds.AnyTopicSeq method*), 53
__getitem__() (*rti.connextdds.BoolSeq method*), 58
__getitem__() (*rti.connextdds.ByteVector method*), 64
__getitem__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__getitem__() (*rti.connextdds.CharSeq method*), 70
__getitem__() (*rti.connextdds.compliance.RpcMask method*), 791
__getitem__() (*rti.connextdds.compliance.XTypesMask method*), 795
__getitem__() (*rti.connextdds.CompressionIdMask method*), 75
__getitem__() (*rti.connextdds.ConditionSeq method*), 79
__getitem__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__getitem__() (*rti.connextdds.CookieSeq method*), 86
__getitem__() (*rti.connextdds.CookieVector method*), 87
__getitem__() (*rti.connextdds.DataReader.LoanedSamples method*), 89
__getitem__() (*rti.connextdds.DataReaderSeq method*), 114
__getitem__() (*rti.connextdds.DataTag method*), 119
__getitem__() (*rti.connextdds.DataWriterSeq method*), 145
__getitem__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 158
__getitem__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*),

161
__getitem__() (*rti.connextdds.DomainParticipantSeq method*), 184
__getitem__() (*rti.connextdds.DynamicData method*), 229
__getitem__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__getitem__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 203
__getitem__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 214
__getitem__() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
__getitem__() (*rti.connextdds.DynamicDataSeq method*), 259
__getitem__() (*rti.connextdds.DynamicData.SharedSamples method*), 222
__getitem__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__getitem__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__getitem__() (*rti.connextdds.EndpointGroupSeq method*), 266
__getitem__() (*rti.connextdds.EndpointGroupVector method*), 268
__getitem__() (*rti.connextdds.EntitySeq method*), 271
__getitem__() (*rti.connextdds.EnumMemberSeq method*), 274
__getitem__() (*rti.connextdds.Float32Seq method*), 285
__getitem__() (*rti.connextdds.Float64Seq method*), 288
__getitem__() (*rti.connextdds.Float128Seq method*), 283
__getitem__() (*rti.connextdds.Guid method*), 298
__getitem__() (*rti.connextdds.IAnyDataReaderSeq method*), 302
__getitem__() (*rti.connextdds.IAnyDataWriterSeq method*), 305
__getitem__() (*rti.connextdds.IAnyTopicSeq method*), 308
__getitem__() (*rti.connextdds.IConditionSeq method*), 311
__getitem__() (*rti.connextdds.IEntitySeq method*), 314
__getitem__() (*rti.connextdds.InstanceHandleSeq method*), 321
__getitem__() (*rti.connextdds.InstanceState method*), 323
__getitem__() (*rti.connextdds.Int8Seq method*), 335
__getitem__() (*rti.connextdds.Int16Seq method*), 326
__getitem__() (*rti.connextdds.Int32Seq method*), 329
__getitem__() (*rti.connextdds.Int32Vector method*), 331
__getitem__() (*rti.connextdds.Int64Seq method*), 333
__getitem__() (*rti.connextdds.LocatorFilterElementSeq method*), 346
__getitem__() (*rti.connextdds.LocatorSeq method*), 352
__getitem__() (*rti.connextdds.LocatorVector method*), 354
__getitem__() (*rti.connextdds.LongDouble method*), 362
__getitem__() (*rti.connextdds.MemberSeq method*), 365
__getitem__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__getitem__() (*rti.connextdds.MulticastMappingSeq method*), 377
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicData.SharedSamples method*), 419
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__getitem__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 421
__getitem__() (*rti.connextdds.Property method*), 444
__getitem__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 448
__getitem__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__getitem__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__getitem__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 473
__getitem__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__getitem__() (*rti.connextdds.PublicationBuiltinTopicData.SharedSamples method*), 477
__getitem__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__getitem__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 479
__getitem__() (*rti.connextdds.PublisherSeq method*), 498
__getitem__() (*rti.connextdds.QosPolicyCountSeq method*), 500
__getitem__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__getitem__() (*rti.connextdds.SampleFlag method*), 528
__getitem__() (*rti.connextdds.SampleLostState method*), 534

__getitem__() (*rti.connextdds.SampleRejectedState method*), 537
__getitem__() (*rti.connextdds.SampleState method*), 540
__getitem__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 549
__getitem__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__getitem__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__getitem__() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 572
__getitem__() (*rti.connextdds.ServiceRequestSeq method*), 586
__getitem__() (*rti.connextdds.ServiceRequest.SharedSamples method*), 576
__getitem__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__getitem__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__getitem__() (*rti.connextdds.StatusMask method*), 592
__getitem__() (*rti.connextdds.StreamKind method*), 595
__getitem__() (*rti.connextdds.StringMap method*), 597
__getitem__() (*rti.connextdds.StringPairSeq method*), 598
__getitem__() (*rti.connextdds.StringSeq method*), 600
__getitem__() (*rti.connextdds.SubscriberSeq method*), 608
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples method*), 636
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 649
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples method*), 640
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__getitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__getitem__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__getitem__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__getitem__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 678
__getitem__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__getitem__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 693
__getitem__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 704
__getitem__() (*rti.connextdds.TopicBuiltinTopicData.SharedSamples method*), 697
__getitem__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__getitem__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__getitem__() (*rti.connextdds.TopicSeq method*), 719
__getitem__() (*rti.connextdds.TransportBuiltinMask method*), 722
__getitem__() (*rti.connextdds.TransportInfoSeq method*), 729
__getitem__() (*rti.connextdds.TransportInfoVector method*), 731
__getitem__() (*rti.connextdds.TransportMulticastSeq method*), 736
__getitem__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__getitem__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__getitem__() (*rti.connextdds.TriggeredConditions method*), 746
__getitem__() (*rti.connextdds.Uint8Seq method*), 765
__getitem__() (*rti.connextdds.Uint16Seq method*), 757
__getitem__() (*rti.connextdds.Uint32Seq method*), 759
__getitem__() (*rti.connextdds.Uint64Seq method*), 762
__getitem__() (*rti.connextdds.UnionMemberSeq method*), 769
__getitem__() (*rti.connextdds.ViewState method*), 776
__getitem__() (*rti.connextdds.WcharSeq method*), 781
__getitem__() (*rti.connextdds.WstringSeq method*), 789
__getstate__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 39
__getstate__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__getstate__() (*rti.connextdds.CharSeq method*), 70
__getstate__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__getstate__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 142

__getstate__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__getstate__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__getstate__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 187
__getstate__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__getstate__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 279
__getstate__() (*rti.connextdds.Float32Seq method*), 285
__getstate__() (*rti.connextdds.Float64Seq method*), 288
__getstate__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 293
__getstate__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__getstate__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__getstate__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__getstate__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 317
__getstate__() (*rti.connextdds.InstanceStateConsistencyKind method*), 325
__getstate__() (*rti.connextdds.Int8Seq method*), 335
__getstate__() (*rti.connextdds.Int16Seq method*), 326
__getstate__() (*rti.connextdds.Int32Seq method*), 329
__getstate__() (*rti.connextdds.Int64Seq method*), 333
__getstate__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__getstate__() (*rti.connextdds.LocatorKind.LocatorKind method*), 349
__getstate__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__getstate__() (*rti.connextdds.LogFacility method*), 357
__getstate__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__getstate__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__getstate__() (*rti.connextdds.PersistentJournalKind method*), 432
__getstate__() (*rti.connextdds.PersistentSynchronizationKind method*), 434
__getstate__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__getstate__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 441
__getstate__() (*rti.connextdds.PrintFormat.PrintFormat method*), 439
__getstate__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__getstate__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 513
__getstate__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__getstate__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__getstate__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__getstate__() (*rti.connextdds.SyslogVerbosity method*), 656
__getstate__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 658
__getstate__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 717
__getstate__() (*rti.connextdds.TransportClassId.TransportClassId method*), 726
__getstate__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__getstate__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 748
__getstate__() (*rti.connextdds.TypeKind.TypeKind method*), 753
__getstate__() (*rti.connextdds.Uint8Seq method*), 765
__getstate__() (*rti.connextdds.Uint16Seq method*), 757
__getstate__() (*rti.connextdds.Uint32Seq method*), 760
__getstate__() (*rti.connextdds.Uint64Seq method*), 762
__getstate__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__getstate__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__gt__() (*rti.connextdds.AcknowledgmentKind method*), 40
__gt__() (*rti.connextdds.CdrPaddingKind method*), 66
__gt__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__gt__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__gt__() (*rti.connextdds.DestinationOrderKind method*), 151
__gt__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__gt__() (*rti.connextdds.DurabilityKind method*), 188
__gt__() (*rti.connextdds.Duration method*), 190

__gt__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__gt__() (*rti.connextdds.ExtensibilityKind method*), 280
__gt__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 294
__gt__() (*rti.connextdds.Guid method*), 298
__gt__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__gt__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 802
__gt__() (*rti.connextdds.HistoryKind method*), 300
__gt__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 318
__gt__() (*rti.connextdds.LivelinessKind method*), 343
__gt__() (*rti.connextdds.LocatorKind method*), 351
__gt__() (*rti.connextdds.LogCategory method*), 356
__gt__() (*rti.connextdds.LogLevel method*), 360
__gt__() (*rti.connextdds.OwnershipKind method*), 387
__gt__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__gt__() (*rti.connextdds.PrintFormat method*), 440
__gt__() (*rti.connextdds.PrintFormatKind method*), 442
__gt__() (*rti.connextdds.PublishModeKind method*), 493
__gt__() (*rti.connextdds.ReliabilityKind method*), 514
__gt__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 517
__gt__() (*rti.connextdds.SequenceNumber method*), 542
__gt__() (*rti.connextdds.ServiceKind method*), 547
__gt__() (*rti.connextdds.ServiceRequestId method*), 585
__gt__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 659
__gt__() (*rti.connextdds.Time method*), 663
__gt__() (*rti.connextdds.TopicQuerySelectionKind method*), 718
__gt__() (*rti.connextdds.TransportClassId method*), 727
__gt__() (*rti.connextdds.TransportMulticastKind method*), 734
__gt__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 749
__gt__() (*rti.connextdds.TypeKind method*), 755
__gt__() (*rti.connextdds.Verbosity method*), 775
__gt__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__hash__ (*rti.connextdds.AcknowledgmentKind attribute*), 40
__hash__ (*rti.connextdds.ACTEnumMember attribute*), 34
__hash__ (*rti.connextdds.ActivityContextMask attribute*), 42
__hash__ (*rti.connextdds.ACTMember attribute*), 36
__hash__ (*rti.connextdds.ACTUnionMember attribute*), 37
__hash__ (*rti.connextdds.AliasType attribute*), 44
__hash__ (*rti.connextdds.AllocationSettings attribute*), 45
__hash__ (*rti.connextdds.AnyDataReaderSeq attribute*), 47
__hash__ (*rti.connextdds.AnyDataWriterSeq attribute*), 51
__hash__ (*rti.connextdds.AnyTopicSeq attribute*), 53
__hash__ (*rti.connextdds.ArrayType attribute*), 55
__hash__ (*rti.connextdds.AsynchronousPublisher attribute*), 55
__hash__ (*rti.connextdds.Availability attribute*), 56
__hash__ (*rti.connextdds.Batch attribute*), 57
__hash__ (*rti.connextdds.BoolSeq attribute*), 58
__hash__ (*rti.connextdds.BoolType attribute*), 60
__hash__ (*rti.connextdds.BuiltinTopicKey attribute*), 63
__hash__ (*rti.connextdds.BuiltinTopicReaderResourceLimits attribute*), 63
__hash__ (*rti.connextdds.ByteVector attribute*), 64
__hash__ (*rti.connextdds.CdrPaddingKind attribute*), 67
__hash__ (*rti.connextdds.ChannelSettings attribute*), 67
__hash__ (*rti.connextdds.ChannelSettingsSeq attribute*), 68
__hash__ (*rti.connextdds.CharSeq attribute*), 71
__hash__ (*rti.connextdds.CharType attribute*), 72
__hash__ (*rti.connextdds.CoherentSetInfo attribute*), 73
__hash__ (*rti.connextdds.CollectionType attribute*), 74
__hash__ (*rti.connextdds.compliance.RpcMask attribute*), 791
__hash__ (*rti.connextdds.compliance.XTypesMask attribute*), 795
__hash__ (*rti.connextdds.CompressionIdMask attribute*), 75
__hash__ (*rti.connextdds.CompressionSettings attribute*), 78
__hash__ (*rti.connextdds.ConditionSeq attribute*), 79
__hash__ (*rti.connextdds.ContentFilteredTopic attribute*), 81
__hash__ (*rti.connextdds.ContentFilteredTopicSeq attribute*), 83
__hash__ (*rti.connextdds.ContentFilterProperty attribute*), 81
__hash__ (*rti.connextdds.Cookie attribute*), 85
__hash__ (*rti.connextdds.CookieSeq attribute*), 86
__hash__ (*rti.connextdds.CookieVector attribute*), 88
__hash__ (*rti.connextdds.Database attribute*), 147
__hash__ (*rti.connextdds.DataReader attribute*), 91
__hash__ (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 99
__hash__ (*rti.connextdds.DataReaderProtocol attribute*), 101
__hash__ (*rti.connextdds.DataReaderQos attribute*), 103
__hash__ (*rti.connextdds.DataReaderResourceLimits attribute*), 110
__hash__ (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings attribute*), 112
__hash__ (*rti.connextdds.DataReaderSeq attribute*), 114
__hash__ (*rti.connextdds.DataRepresentation attribute*), 115
__hash__ (*rti.connextdds.DataState attribute*), 116
__hash__ (*rti.connextdds.DataStateEx attribute*), 117
__hash__ (*rti.connextdds.DataTag attribute*), 119
__hash__ (*rti.connextdds.DataWriter attribute*), 121
__hash__ (*rti.connextdds.DataWriterProtocol attribute*), 128
__hash__ (*rti.connextdds.DataWriterQos attribute*), 131
__hash__ (*rti.connextdds.DataWriterResourceLimits attribute*), 139
__hash__ (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 143
__hash__ (*rti.connextdds.DataWriterSeq attribute*), 145
__hash__ (*rti.connextdds.DataWriterShmemRefTransferModeSettings attribute*), 146
__hash__ (*rti.connextdds.DataWriterTransferMode attribute*), 147
__hash__ (*rti.connextdds.Deadline attribute*), 148
__hash__ (*rti.connextdds.DestinationOrder attribute*), 149
__hash__ (*rti.connextdds.DestinationOrderKind attribute*), 151

__hash__ (*rti.connextdds.DestinationOrderScopeKind attribute*), 153
__hash__ (*rti.connextdds.Discovery attribute*), 154
__hash__ (*rti.connextdds.DiscoveryConfig attribute*), 155
__hash__ (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 158
__hash__ (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
__hash__ (*rti.connextdds.DomainParticipant attribute*), 164
__hash__ (*rti.connextdds.DomainParticipantConfigParams attribute*), 171
__hash__ (*rti.connextdds.DomainParticipantFactoryQos attribute*), 172
__hash__ (*rti.connextdds.DomainParticipantQos attribute*), 174
__hash__ (*rti.connextdds.DomainParticipantResourceLimits attribute*), 178
__hash__ (*rti.connextdds.DomainParticipantSeq attribute*), 184
__hash__ (*rti.connextdds.Durability attribute*), 186
__hash__ (*rti.connextdds.DurabilityKind attribute*), 188
__hash__ (*rti.connextdds.DurabilityService attribute*), 189
__hash__ (*rti.connextdds.Duration attribute*), 190
__hash__ (*rti.connextdds.DynamicData attribute*), 229
__hash__ (*rti.connextdds.DynamicData.ContentFilteredTopic attribute*), 193
__hash__ (*rti.connextdds.DynamicData.ContentFilteredTopicSeq attribute*), 194
__hash__ (*rti.connextdds.DynamicData.DataReader attribute*), 197
__hash__ (*rti.connextdds.DynamicData.DataReaderSeq attribute*), 204
__hash__ (*rti.connextdds.DynamicData.DataWriter attribute*), 205
__hash__ (*rti.connextdds.DynamicData.DataWriterSeq attribute*), 215
__hash__ (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 256
__hash__ (*rti.connextdds.DynamicDataInfo attribute*), 257
__hash__ (*rti.connextdds.DynamicDataMemberInfo attribute*), 257
__hash__ (*rti.connextdds.DynamicDataProperty attribute*), 258
__hash__ (*rti.connextdds.DynamicDataSeq attribute*), 259
__hash__ (*rti.connextdds.DynamicDataTimestampedSeq attribute*), 261
__hash__ (*rti.connextdds.DynamicData.Topic attribute*), 223
__hash__ (*rti.connextdds.DynamicData.TopicDescription attribute*), 224
__hash__ (*rti.connextdds.DynamicData.TopicSeq attribute*), 225
__hash__ (*rti.connextdds.DynamicDataTypeSerializationProperty attribute*), 263
__hash__ (*rti.connextdds.DynamicType attribute*), 264
__hash__ (*rti.connextdds.DynamicTypePrintFormatProperty attribute*), 265
__hash__ (*rti.connextdds.EndpointGroupSeq attribute*), 267
__hash__ (*rti.connextdds.EndpointGroupVector attribute*), 268
__hash__ (*rti.connextdds.EntityFactory attribute*), 269
__hash__ (*rti.connextdds.EntityName attribute*), 270
__hash__ (*rti.connextdds.EntitySeq attribute*), 271
__hash__ (*rti.connextdds.EnumMember attribute*), 273
__hash__ (*rti.connextdds.EnumMemberSeq attribute*), 274
__hash__ (*rti.connextdds.EnumType attribute*), 276
__hash__ (*rti.connextdds.Event attribute*), 277
__hash__ (*rti.connextdds.ExpressionProperty attribute*), 278
__hash__ (*rti.connextdds.ExtensibilityKind attribute*), 280
__hash__ (*rti.connextdds.FilterSampleInfo attribute*), 282
__hash__ (*rti.connextdds.Float32Seq attribute*), 285
__hash__ (*rti.connextdds.Float32Type attribute*), 287
__hash__ (*rti.connextdds.Float64Seq attribute*), 288
__hash__ (*rti.connextdds.Float64Type attribute*), 290
__hash__ (*rti.connextdds.Float128Seq attribute*), 283
__hash__ (*rti.connextdds.Float128Type attribute*), 284
__hash__ (*rti.connextdds.FlowController attribute*), 290
__hash__ (*rti.connextdds.FlowControllerProperty attribute*), 291
__hash__ (*rti.connextdds.FlowControllerSchedulingPolicy attribute*), 294
__hash__ (*rti.connextdds.FlowControllerTokenBucketProperty attribute*), 295
__hash__ (*rti.connextdds.GroupData attribute*), 296
__hash__ (*rti.connextdds.GuardCondition attribute*), 297
__hash__ (*rti.connextdds.Guid attribute*), 298
__hash__ (*rti.connextdds.heap_monitoring.HeapMonitoringParams attribute*), 798
__hash__ (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 800
__hash__ (*rti.connextdds.heap_monitoring.SnapshotOutputFormat attribute*), 802
__hash__ (*rti.connextdds.History attribute*), 299
__hash__ (*rti.connextdds.HistoryKind attribute*), 301
__hash__ (*rti.connextdds.IAnyDataReader attribute*), 301
__hash__ (*rti.connextdds.IAnyDataReaderSeq attribute*), 303
__hash__ (*rti.connextdds.IAnyDataWriter attribute*), 304
__hash__ (*rti.connextdds.IAnyDataWriterSeq attribute*), 306
__hash__ (*rti.connextdds.IAnyTopic attribute*), 307
__hash__ (*rti.connextdds.IAnyTopicSeq attribute*), 308
__hash__ (*rti.connextdds.ICondition attribute*), 310
__hash__ (*rti.connextdds.IConditionSeq attribute*), 311
__hash__ (*rti.connextdds.IEntity attribute*), 313
__hash__ (*rti.connextdds.IEntitySeq attribute*), 314
__hash__ (*rti.connextdds.IgnoredEntityReplacementKind attribute*), 318
__hash__ (*rti.connextdds.InstanceHandleSeq attribute*), 321
__hash__ (*rti.connextdds.InstanceState attribute*), 323
__hash__ (*rti.connextdds.Int8Seq attribute*), 335
__hash__ (*rti.connextdds.Int8Type attribute*), 337
__hash__ (*rti.connextdds.Int16Seq attribute*), 326
__hash__ (*rti.connextdds.Int16Type attribute*), 328
__hash__ (*rti.connextdds.Int32Seq attribute*), 329
__hash__ (*rti.connextdds.Int32Type attribute*), 331
__hash__ (*rti.connextdds.Int32Vector attribute*), 331

__hash__ (*rti.connextdds.Int64Seq attribute*), 333
__hash__ (*rti.connextdds.Int64Type attribute*), 335
__hash__ (*rti.connextdds.IReadCondition attribute*), 316
__hash__ (*rti.connextdds.LatencyBudget attribute*), 339
__hash__ (*rti.connextdds.Lifespan attribute*), 339
__hash__ (*rti.connextdds.Liveliness attribute*), 340
__hash__ (*rti.connextdds.LivelinessKind attribute*), 343
__hash__ (*rti.connextdds.Locator attribute*), 344
__hash__ (*rti.connextdds.LocatorFilter attribute*), 345
__hash__ (*rti.connextdds.LocatorFilterElement attribute*), 345
__hash__ (*rti.connextdds.LocatorFilterElementSeq attribute*), 347
__hash__ (*rti.connextdds.LocatorKind attribute*), 351
__hash__ (*rti.connextdds.LocatorSeq attribute*), 352
__hash__ (*rti.connextdds.LocatorVector attribute*), 354
__hash__ (*rti.connextdds.LogCategory attribute*), 356
__hash__ (*rti.connextdds.LogLevel attribute*), 360
__hash__ (*rti.connextdds.LongDouble attribute*), 362
__hash__ (*rti.connextdds.Member attribute*), 363
__hash__ (*rti.connextdds.MemberSeq attribute*), 365
__hash__ (*rti.connextdds.Monitoring attribute*), 366
__hash__ (*rti.connextdds.MonitoringDedicatedParticipantSettings attribute*), 367
__hash__ (*rti.connextdds.MonitoringDistributionSettings attribute*), 368
__hash__ (*rti.connextdds.MonitoringEventDistributionSettings attribute*), 369
__hash__ (*rti.connextdds.MonitoringLoggingDistributionSettings attribute*), 369
__hash__ (*rti.connextdds.MonitoringLoggingForwardingSettings attribute*), 370
__hash__ (*rti.connextdds.MonitoringMetricSelection attribute*), 371
__hash__ (*rti.connextdds.MonitoringMetricSelectionSeq attribute*), 372
__hash__ (*rti.connextdds.MonitoringPeriodicDistributionSettings attribute*), 374
__hash__ (*rti.connextdds.MonitoringTelemetryData attribute*), 374
__hash__ (*rti.connextdds.MulticastMapping attribute*), 376
__hash__ (*rti.connextdds.MulticastMappingSeq attribute*), 377
__hash__ (*rti.connextdds.MultiChannel attribute*), 375
__hash__ (*rti.connextdds.OctetType attribute*), 384
__hash__ (*rti.connextdds.Ownership attribute*), 386
__hash__ (*rti.connextdds.OwnershipKind attribute*), 387
__hash__ (*rti.connextdds.OwnershipStrength attribute*), 388
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData attribute*), 425
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic attribute*), 389
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq attribute*), 391
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReader attribute*), 394
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq attribute*), 400
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter attribute*), 402
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq attribute*), 412
__hash__ (*rti.connextdds.ParticipantBuiltinTopicDataSeq attribute*), 427
__hash__ (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq attribute*), 429
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.Topic attribute*), 419
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicDescription attribute*), 420
__hash__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq attribute*), 422
__hash__ (*rti.connextdds.Partition attribute*), 431
__hash__ (*rti.connextdds.PersistentStorageSettings attribute*), 433
__hash__ (*rti.connextdds.Presentation attribute*), 435
__hash__ (*rti.connextdds.PresentationAccessScopeKind attribute*), 438
__hash__ (*rti.connextdds.PrintFormat attribute*), 440
__hash__ (*rti.connextdds.PrintFormatKind attribute*), 442
__hash__ (*rti.connextdds.ProductVersion attribute*), 443
__hash__ (*rti.connextdds.Property attribute*), 444
__hash__ (*rti.connextdds.ProtocolVersion attribute*), 446
__hash__ (*rti.connextdds.PublicationBuiltinTopicData attribute*), 483
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic attribute*), 447
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq attribute*), 449
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.DataReader attribute*), 452
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq attribute*), 458
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriter attribute*), 460
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq attribute*), 470
__hash__ (*rti.connextdds.PublicationBuiltinTopicDataSeq attribute*), 486
__hash__ (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq attribute*),

488
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.Topic attribute*), 477
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.TopicDescription attribute*), 478
__hash__ (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq attribute*), 479
__hash__ (*rti.connextdds.Publisher attribute*), 494
__hash__ (*rti.connextdds.PublisherQos attribute*), 495
__hash__ (*rti.connextdds.PublisherSeq attribute*), 498
__hash__ (*rti.connextdds.PublishMode attribute*), 491
__hash__ (*rti.connextdds.PublishModeKind attribute*), 493
__hash__ (*rti.connextdds.QosPolicyCount attribute*), 500
__hash__ (*rti.connextdds.QosPolicyCountSeq attribute*), 501
__hash__ (*rti.connextdds.QosPrintFormat attribute*), 502
__hash__ (*rti.connextdds.QosProvider attribute*), 503
__hash__ (*rti.connextdds.QosProviderParams attribute*), 506
__hash__ (*rti.connextdds.Query attribute*), 507
__hash__ (*rti.connextdds.QueryCondition attribute*), 508
__hash__ (*rti.connextdds.ReaderDataLifecycle attribute*), 510
__hash__ (*rti.connextdds.ReceiverPool attribute*), 511
__hash__ (*rti.connextdds.Reliability attribute*), 512
__hash__ (*rti.connextdds.ReliabilityKind attribute*), 514
__hash__ (*rti.connextdds.RemoteParticipantPurgeKind attribute*), 518
__hash__ (*rti.connextdds.ResourceLimits attribute*), 519
__hash__ (*rti.connextdds.RtpsReliableReaderProtocol attribute*), 520
__hash__ (*rti.connextdds.RtpsReliableWriterProtocol attribute*), 521
__hash__ (*rti.connextdds.RtpsReservedPortKindMask attribute*), 524
__hash__ (*rti.connextdds.RtpsWellKnownPorts attribute*), 527
__hash__ (*rti.connextdds.SampleFlag attribute*), 528
__hash__ (*rti.connextdds.SampleIdentity attribute*), 530
__hash__ (*rti.connextdds.SampleInfo attribute*), 531
__hash__ (*rti.connextdds.SampleLostState attribute*), 534
__hash__ (*rti.connextdds.SampleRejectedState attribute*), 537
__hash__ (*rti.connextdds.SampleState attribute*), 540
__hash__ (*rti.connextdds.SequenceNumber attribute*), 542
__hash__ (*rti.connextdds.SequenceType attribute*), 544
__hash__ (*rti.connextdds.Service attribute*), 544
__hash__ (*rti.connextdds.ServiceKind attribute*), 547
__hash__ (*rti.connextdds.ServiceRequest attribute*), 582
__hash__ (*rti.connextdds.ServiceRequest.ContentFilteredTopic attribute*), 548
__hash__ (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq attribute*), 550
__hash__ (*rti.connextdds.ServiceRequest.DataReader attribute*), 553
__hash__ (*rti.connextdds.ServiceRequest.DataReaderSeq attribute*), 559
__hash__ (*rti.connextdds.ServiceRequest.DataWriter attribute*), 561
__hash__ (*rti.connextdds.ServiceRequest.DataWriterSeq attribute*), 570
__hash__ (*rti.connextdds.ServiceRequestId attribute*), 585
__hash__ (*rti.connextdds.ServiceRequestSeq attribute*), 586
__hash__ (*rti.connextdds.ServiceRequestTimestampedSeq attribute*), 588
__hash__ (*rti.connextdds.ServiceRequest.Topic attribute*), 577
__hash__ (*rti.connextdds.ServiceRequest.TopicDescription attribute*), 578
__hash__ (*rti.connextdds.ServiceRequest.TopicSeq attribute*), 579
__hash__ (*rti.connextdds.StatusCondition attribute*), 590
__hash__ (*rti.connextdds.StatusMask attribute*), 592
__hash__ (*rti.connextdds.StreamKind attribute*), 595
__hash__ (*rti.connextdds.StringPairSeq attribute*), 598
__hash__ (*rti.connextdds.StringSeq attribute*), 600
__hash__ (*rti.connextdds.StringType attribute*), 602
__hash__ (*rti.connextdds.StructType attribute*), 603
__hash__ (*rti.connextdds.Subscriber attribute*), 604
__hash__ (*rti.connextdds.SubscriberQos attribute*), 606
__hash__ (*rti.connextdds.SubscriberSeq attribute*), 608
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData attribute*), 647
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic attribute*), 610
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq attribute*), 612
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader attribute*), 615
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq attribute*), 622
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter attribute*), 624
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq attribute*), 634
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicDataSeq attribute*), 650
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq attribute*), 652
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.Topic attribute*), 641
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription attribute*), 642
__hash__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq attribute*), 643
__hash__ (*rti.connextdds.SystemResourceLimits attribute*), 656
__hash__ (*rti.connextdds.ThreadSettings attribute*), 657
__hash__ (*rti.connextdds.ThreadSettingsCpuRotationKind attribute*), 659
__hash__ (*rti.connextdds.ThreadSettingsKindMask attribute*), 661
__hash__ (*rti.connextdds.Time attribute*), 663

__hash__ (*rti.connextdds.TimeBasedFilter attribute*), 665
__hash__ (*rti.connextdds.Topic attribute*), 666
__hash__ (*rti.connextdds.TopicBuiltinTopicData attribute*), 703
__hash__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic attribute*), 667
__hash__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq attribute*), 669
__hash__ (*rti.connextdds.TopicBuiltinTopicData.DataReader attribute*), 672
__hash__ (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq attribute*), 679
__hash__ (*rti.connextdds.TopicBuiltinTopicData.DataWriter attribute*), 680
__hash__ (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq attribute*), 690
__hash__ (*rti.connextdds.TopicBuiltinTopicDataSeq attribute*), 705
__hash__ (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq attribute*), 707
__hash__ (*rti.connextdds.TopicBuiltinTopicData.Topic attribute*), 697
__hash__ (*rti.connextdds.TopicBuiltinTopicData.TopicDescription attribute*), 698
__hash__ (*rti.connextdds.TopicBuiltinTopicData.TopicSeq attribute*), 699
__hash__ (*rti.connextdds.TopicData attribute*), 709
__hash__ (*rti.connextdds.TopicDescription attribute*), 710
__hash__ (*rti.connextdds.TopicQos attribute*), 710
__hash__ (*rti.connextdds.TopicQuery attribute*), 714
__hash__ (*rti.connextdds.TopicQueryDispatch attribute*), 716
__hash__ (*rti.connextdds.TopicQuerySelectionKind attribute*), 718
__hash__ (*rti.connextdds.TopicSeq attribute*), 720
__hash__ (*rti.connextdds.TransportBuiltin attribute*), 721
__hash__ (*rti.connextdds.TransportBuiltinMask attribute*), 722
__hash__ (*rti.connextdds.TransportClassId attribute*), 727
__hash__ (*rti.connextdds.TransportInfoSeq attribute*), 729
__hash__ (*rti.connextdds.TransportInfoVector attribute*), 731
__hash__ (*rti.connextdds.TransportMulticast attribute*), 732
__hash__ (*rti.connextdds.TransportMulticastKind attribute*), 734
__hash__ (*rti.connextdds.TransportMulticastMapping attribute*), 735
__hash__ (*rti.connextdds.TransportMulticastMappingFunction attribute*), 735
__hash__ (*rti.connextdds.TransportMulticastSeq attribute*), 736
__hash__ (*rti.connextdds.TransportMulticastSettings attribute*), 738
__hash__ (*rti.connextdds.TransportMulticastSettingsSeq attribute*), 739
__hash__ (*rti.connextdds.TransportPriority attribute*), 741
__hash__ (*rti.connextdds.TransportSelection attribute*), 741
__hash__ (*rti.connextdds.TransportUnicast attribute*), 742
__hash__ (*rti.connextdds.TransportUnicastSettings attribute*), 743
__hash__ (*rti.connextdds.TransportUnicastSettingsSeq attribute*), 744
__hash__ (*rti.connextdds.TypeConsistencyEnforcement attribute*), 746
__hash__ (*rti.connextdds.TypeConsistencyEnforcementKind attribute*), 749
__hash__ (*rti.connextdds.TypeKind attribute*), 755
__hash__ (*rti.connextdds.TypeSupport attribute*), 756
__hash__ (*rti.connextdds.Uint8Seq attribute*), 765
__hash__ (*rti.connextdds.Uint8Type attribute*), 767
__hash__ (*rti.connextdds.Uint16Seq attribute*), 757
__hash__ (*rti.connextdds.Uint16Type attribute*), 759
__hash__ (*rti.connextdds.Uint32Seq attribute*), 760
__hash__ (*rti.connextdds.Uint32Type attribute*), 761
__hash__ (*rti.connextdds.Uint64Seq attribute*), 762
__hash__ (*rti.connextdds.Uint64Type attribute*), 764
__hash__ (*rti.connextdds.UnidimensionalCollectionType attribute*), 767
__hash__ (*rti.connextdds.UnionMember attribute*), 768
__hash__ (*rti.connextdds.UnionMemberSeq attribute*), 769
__hash__ (*rti.connextdds.UnionType attribute*), 771
__hash__ (*rti.connextdds.UserData attribute*), 772
__hash__ (*rti.connextdds.VendorId attribute*), 773
__hash__ (*rti.connextdds.Verbosity attribute*), 775
__hash__ (*rti.connextdds.ViewState attribute*), 776
__hash__ (*rti.connextdds.WaitSetProperty attribute*), 780
__hash__ (*rti.connextdds.WcharSeq attribute*), 781
__hash__ (*rti.connextdds.WcharType attribute*), 783
__hash__ (*rti.connextdds.WireProtocol attribute*), 783
__hash__ (*rti.connextdds.WireProtocolAutoKind attribute*), 786
__hash__ (*rti.connextdds.WriterDataLifecycle attribute*), 788
__hash__ (*rti.connextdds.WstringSeq attribute*), 789
__hash__ (*rti.connextdds.WStringType attribute*), 778
__hash__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 39
__hash__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__hash__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__hash__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 142
__hash__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__hash__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__hash__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 187
__hash__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255

__hash__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 279
__hash__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 293
__hash__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__hash__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__hash__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__hash__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 317
__hash__() (*rti.connextdds.InstanceHandle method*), 320
__hash__() (*rti.connextdds.InstanceStateConsistencyKind method*), 325
__hash__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__hash__() (*rti.connextdds.LocatorKind.LocatorKind method*), 349
__hash__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__hash__() (*rti.connextdds.LogFacility method*), 357
__hash__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__hash__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__hash__() (*rti.connextdds.PersistentJournalKind method*), 432
__hash__() (*rti.connextdds.PersistentSynchronizationKind method*), 434
__hash__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__hash__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 441
__hash__() (*rti.connextdds.PrintFormat.PrintFormat method*), 439
__hash__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__hash__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 513
__hash__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__hash__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__hash__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__hash__() (*rti.connextdds.SyslogVerbosity method*), 656
__hash__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 658
__hash__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 717
__hash__() (*rti.connextdds.TransportClassId.TransportClassId method*), 726
__hash__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__hash__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 748
__hash__() (*rti.connextdds.TypeKind.TypeKind method*), 753
__hash__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__hash__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__iadd__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__iadd__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__iadd__() (*rti.connextdds.AnyTopicSeq method*), 53
__iadd__() (*rti.connextdds.BoolSeq method*), 58
__iadd__() (*rti.connextdds.ChannelSettingsSeq method*), 68
__iadd__() (*rti.connextdds.CharSeq method*), 71
__iadd__() (*rti.connextdds.ConditionSeq method*), 79
__iadd__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__iadd__() (*rti.connextdds.CookieSeq method*), 86
__iadd__() (*rti.connextdds.DataReaderSeq method*), 114
__iadd__() (*rti.connextdds.DataWriterSeq method*), 145
__iadd__() (*rti.connextdds.DomainParticipantSeq method*), 184
__iadd__() (*rti.connextdds.Duration method*), 190
__iadd__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__iadd__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__iadd__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__iadd__() (*rti.connextdds.DynamicDataSeq method*), 259
__iadd__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__iadd__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__iadd__() (*rti.connextdds.EndpointGroupSeq method*), 267
__iadd__() (*rti.connextdds.EntitySeq method*), 271
__iadd__() (*rti.connextdds.EnumMemberSeq method*), 274
__iadd__() (*rti.connextdds.Float32Seq method*), 285
__iadd__() (*rti.connextdds.Float64Seq method*), 288
__iadd__() (*rti.connextdds.Float128Seq method*), 283
__iadd__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__iadd__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__iadd__() (*rti.connextdds.IAnyTopicSeq method*), 308
__iadd__() (*rti.connextdds.IConditionSeq method*), 311
__iadd__() (*rti.connextdds.IEntitySeq method*), 314
__iadd__() (*rti.connextdds.InstanceHandleSeq method*), 321
__iadd__() (*rti.connextdds.Int8Seq method*), 335
__iadd__() (*rti.connextdds.Int16Seq method*), 326
__iadd__() (*rti.connextdds.Int32Seq method*), 329
__iadd__() (*rti.connextdds.Int64Seq method*), 333

__iadd__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__iadd__() (*rti.connextdds.LocatorSeq method*), 352
__iadd__() (*rti.connextdds.MemberSeq method*), 365
__iadd__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__iadd__() (*rti.connextdds.MulticastMappingSeq method*), 377
__iadd__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__iadd__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__iadd__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__iadd__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__iadd__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__iadd__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__iadd__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__iadd__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__iadd__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__iadd__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__iadd__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__iadd__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__iadd__() (*rti.connextdds.PublisherSeq method*), 498
__iadd__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__iadd__() (*rti.connextdds.SequenceNumber method*), 542
__iadd__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__iadd__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__iadd__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__iadd__() (*rti.connextdds.ServiceRequestSeq method*), 586
__iadd__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__iadd__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__iadd__() (*rti.connextdds.StringPairSeq method*), 598
__iadd__() (*rti.connextdds.StringSeq method*), 601
__iadd__() (*rti.connextdds.SubscriberSeq method*), 608
__iadd__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__iadd__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__iadd__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__iadd__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__iadd__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__iadd__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__iadd__() (*rti.connextdds.Time method*), 663
__iadd__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__iadd__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__iadd__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__iadd__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__iadd__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__iadd__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__iadd__() (*rti.connextdds.TopicSeq method*), 720
__iadd__() (*rti.connextdds.TransportInfoSeq method*), 729
__iadd__() (*rti.connextdds.TransportMulticastSeq method*), 736
__iadd__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__iadd__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__iadd__() (*rti.connextdds.Uint8Seq method*), 765
__iadd__() (*rti.connextdds.Uint16Seq method*), 757
__iadd__() (*rti.connextdds.Uint32Seq method*), 760
__iadd__() (*rti.connextdds.Uint64Seq method*), 762
__iadd__() (*rti.connextdds.UnionMemberSeq method*), 769
__iadd__() (*rti.connextdds.WaitSet method*), 779
__iadd__() (*rti.connextdds.WcharSeq method*), 781
__iadd__() (*rti.connextdds.WstringSeq method*), 789
__iand__() (*rti.connextdds.ActivityContextMask method*), 42
__iand__() (*rti.connextdds.compliance.RpcMask method*), 791
__iand__() (*rti.connextdds.compliance.XTypesMask method*), 795
__iand__() (*rti.connextdds.CompressionIdMask method*), 75
__iand__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*),

158
__iand__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__iand__() (*rti.connextdds.InstanceState method*), 323
__iand__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__iand__() (*rti.connextdds.SampleFlag method*), 528
__iand__() (*rti.connextdds.SampleLostState method*), 534
__iand__() (*rti.connextdds.SampleRejectedState method*), 537
__iand__() (*rti.connextdds.SampleState method*), 540
__iand__() (*rti.connextdds.StatusMask method*), 592
__iand__() (*rti.connextdds.StreamKind method*), 595
__iand__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__iand__() (*rti.connextdds.TransportBuiltinMask method*), 722
__iand__() (*rti.connextdds.ViewState method*), 776
__ilshift__() (*rti.connextdds.ActivityContextMask method*), 42
__ilshift__() (*rti.connextdds.compliance.RpcMask method*), 792
__ilshift__() (*rti.connextdds.compliance.XTypesMask method*), 795
__ilshift__() (*rti.connextdds.CompressionIdMask method*), 75
__ilshift__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 158
__ilshift__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__ilshift__() (*rti.connextdds.InstanceState method*), 323
__ilshift__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__ilshift__() (*rti.connextdds.SampleFlag method*), 528
__ilshift__() (*rti.connextdds.SampleLostState method*), 534
__ilshift__() (*rti.connextdds.SampleRejectedState method*), 537
__ilshift__() (*rti.connextdds.SampleState method*), 540
__ilshift__() (*rti.connextdds.StatusMask method*), 593
__ilshift__() (*rti.connextdds.StreamKind method*), 595
__ilshift__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__ilshift__() (*rti.connextdds.TransportBuiltinMask method*), 723
__ilshift__() (*rti.connextdds.ViewState method*), 776
__imul__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__imul__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__imul__() (*rti.connextdds.AnyTopicSeq method*), 53
__imul__() (*rti.connextdds.BoolSeq method*), 58
__imul__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__imul__() (*rti.connextdds.CharSeq method*), 71
__imul__() (*rti.connextdds.ConditionSeq method*), 79
__imul__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__imul__() (*rti.connextdds.CookieSeq method*), 86
__imul__() (*rti.connextdds.DataReaderSeq method*), 114
__imul__() (*rti.connextdds.DataWriterSeq method*), 145
__imul__() (*rti.connextdds.DomainParticipantSeq method*), 184
__imul__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__imul__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__imul__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__imul__() (*rti.connextdds.DynamicDataSeq method*), 259
__imul__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 261
__imul__() (*rti.connextdds.DynamicData.TopicSeq method*), 225
__imul__() (*rti.connextdds.EndpointGroupSeq method*), 267
__imul__() (*rti.connextdds.EntitySeq method*), 271
__imul__() (*rti.connextdds.EnumMemberSeq method*), 274
__imul__() (*rti.connextdds.Float32Seq method*), 285
__imul__() (*rti.connextdds.Float64Seq method*), 288
__imul__() (*rti.connextdds.Float128Seq method*), 283
__imul__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__imul__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__imul__() (*rti.connextdds.IAnyTopicSeq method*), 309
__imul__() (*rti.connextdds.IConditionSeq method*), 311
__imul__() (*rti.connextdds.IEntitySeq method*), 314
__imul__() (*rti.connextdds.InstanceHandleSeq method*), 321
__imul__() (*rti.connextdds.Int8Seq method*), 335
__imul__() (*rti.connextdds.Int16Seq method*), 327
__imul__() (*rti.connextdds.Int32Seq method*), 329
__imul__() (*rti.connextdds.Int64Seq method*), 333
__imul__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__imul__() (*rti.connextdds.LocatorSeq method*), 352
__imul__() (*rti.connextdds.MemberSeq method*), 365
__imul__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__imul__() (*rti.connextdds.MulticastMappingSeq method*), 377
__imul__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__imul__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 400
__imul__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__imul__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__imul__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__imul__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__imul__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449

__imul__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__imul__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__imul__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__imul__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__imul__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__imul__() (*rti.connextdds.PublisherSeq method*), 498
__imul__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__imul__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__imul__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__imul__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__imul__() (*rti.connextdds.ServiceRequestSeq method*), 586
__imul__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__imul__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__imul__() (*rti.connextdds.StringPairSeq method*), 599
__imul__() (*rti.connextdds.StringSeq method*), 601
__imul__() (*rti.connextdds.SubscriberSeq method*), 608
__imul__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__imul__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__imul__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__imul__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__imul__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__imul__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__imul__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__imul__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__imul__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__imul__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__imul__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__imul__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 699
__imul__() (*rti.connextdds.TopicSeq method*), 720
__imul__() (*rti.connextdds.TransportInfoSeq method*), 729
__imul__() (*rti.connextdds.TransportMulticastSeq method*), 736
__imul__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__imul__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__imul__() (*rti.connextdds.Uint8Seq method*), 765
__imul__() (*rti.connextdds.Uint16Seq method*), 757
__imul__() (*rti.connextdds.Uint32Seq method*), 760
__imul__() (*rti.connextdds.Uint64Seq method*), 762
__imul__() (*rti.connextdds.UnionMemberSeq method*), 769
__imul__() (*rti.connextdds.WcharSeq method*), 782
__imul__() (*rti.connextdds.WstringSeq method*), 789
__index__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 39
__index__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__index__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__index__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 142
__index__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__index__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__index__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 187
__index__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__index__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 279
__index__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 293
__index__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__index__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__index__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__index__() (*rti.connextdds.IgnoredEntityReplacemen-*

*tKind.IgnoredEntityReplacementKind method*), 317
__index__() (*rti.connextdds.InstanceStateConsistencyKind method*), 325
__index__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__index__() (*rti.connextdds.LocatorKind.LocatorKind method*), 349
__index__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__index__() (*rti.connextdds.LogFacility method*), 357
__index__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__index__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__index__() (*rti.connextdds.PersistentJournalKind method*), 432
__index__() (*rti.connextdds.PersistentSynchronizationKind method*), 434
__index__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__index__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 441
__index__() (*rti.connextdds.PrintFormat.PrintFormat method*), 439
__index__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__index__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__index__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__index__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__index__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__index__() (*rti.connextdds.SyslogVerbosity method*), 656
__index__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 658
__index__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 717
__index__() (*rti.connextdds.TransportClassId.TransportClassId method*), 726
__index__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__index__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 748
__index__() (*rti.connextdds.TypeKind.TypeKind method*), 753
__index__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__index__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__init__() (*rti.connextdds.AcknowledgmentInfo method*), 38
__init__() (*rti.connextdds.AcknowledgmentKind method*), 40
__init__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 39
__init__() (*rti.connextdds.AckResponseData method*), 38
__init__() (*rti.connextdds.ACTEnumMember method*), 35
__init__() (*rti.connextdds.ActivityContextMask method*), 42
__init__() (*rti.connextdds.ACTMember method*), 36
__init__() (*rti.connextdds.ACTUnionMember method*), 37
__init__() (*rti.connextdds.AliasType method*), 44
__init__() (*rti.connextdds.AllocationSettings method*), 45
__init__() (*rti.connextdds.AnyDataReader method*), 45
__init__() (*rti.connextdds.AnyDataReaderListener method*), 46
__init__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__init__() (*rti.connextdds.AnyDataWriter method*), 49
__init__() (*rti.connextdds.AnyDataWriterListener method*), 49
__init__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__init__() (*rti.connextdds.AnyTopic method*), 52
__init__() (*rti.connextdds.AnyTopicListener method*), 52
__init__() (*rti.connextdds.AnyTopicSeq method*), 53
__init__() (*rti.connextdds.ArrayType method*), 55
__init__() (*rti.connextdds.AsynchronousPublisher method*), 55
__init__() (*rti.connextdds.Availability method*), 56
__init__() (*rti.connextdds.Batch method*), 57
__init__() (*rti.connextdds.BoolSeq method*), 58
__init__() (*rti.connextdds.BoolType method*), 60
__init__() (*rti.connextdds.BuiltinProfiles method*), 60
__init__() (*rti.connextdds.BuiltinTopicKey method*), 63
__init__() (*rti.connextdds.BuiltinTopicReaderResourceLimits method*), 63
__init__() (*rti.connextdds.ByteVector method*), 64
__init__() (*rti.connextdds.CdrPaddingKind method*), 67
__init__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__init__() (*rti.connextdds.ChannelSettings method*), 67
__init__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__init__() (*rti.connextdds.CharSeq method*), 71
__init__() (*rti.connextdds.CharType method*), 72
__init__() (*rti.connextdds.CoherentAccess method*), 73
__init__() (*rti.connextdds.CoherentSet method*), 73
__init__() (*rti.connextdds.CoherentSetInfo method*), 73
__init__() (*rti.connextdds.CollectionType method*), 74
__init__() (*rti.connextdds.compliance.RpcMask method*), 792
__init__() (*rti.connextdds.compliance.XTypesMask method*), 795
__init__() (*rti.connextdds.CompressionIdMask method*), 75
__init__() (*rti.connextdds.CompressionSettings method*), 78
__init__() (*rti.connextdds.Condition method*), 78
__init__() (*rti.connextdds.ConditionSeq method*), 79
__init__() (*rti.connextdds.ContentFilterBase method*), 81

__init__() (*rti.connextdds.ContentFilteredTopic method*), 82
__init__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__init__() (*rti.connextdds.ContentFilterProperty method*), 81
__init__() (*rti.connextdds.Cookie method*), 85
__init__() (*rti.connextdds.CookieSeq method*), 86
__init__() (*rti.connextdds.CookieVector method*), 88
__init__() (*rti.connextdds.Database method*), 147
__init__() (*rti.connextdds.DataReader method*), 91
__init__() (*rti.connextdds.DataReaderCacheStatus method*), 95
__init__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__init__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__init__() (*rti.connextdds.DataReaderListener method*), 100
__init__() (*rti.connextdds.DataReader.LoanedSample method*), 88
__init__() (*rti.connextdds.DataReader.LoanedSamples method*), 89
__init__() (*rti.connextdds.DataReaderProtocol method*), 101
__init__() (*rti.connextdds.DataReaderProtocolStatus method*), 101
__init__() (*rti.connextdds.DataReaderQos method*), 103
__init__() (*rti.connextdds.DataReaderResourceLimits method*), 110
__init__() (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings method*), 112
__init__() (*rti.connextdds.DataReader.Selector method*), 90
__init__() (*rti.connextdds.DataReaderSeq method*), 114
__init__() (*rti.connextdds.DataRepresentation method*), 115
__init__() (*rti.connextdds.DataState method*), 116
__init__() (*rti.connextdds.DataStateEx method*), 117
__init__() (*rti.connextdds.DataTag method*), 119
__init__() (*rti.connextdds.DataWriter method*), 121
__init__() (*rti.connextdds.DataWriterCacheStatus method*), 126
__init__() (*rti.connextdds.DataWriterListener method*), 127
__init__() (*rti.connextdds.DataWriterProtocol method*), 128
__init__() (*rti.connextdds.DataWriterProtocolStatus method*), 129
__init__() (*rti.connextdds.DataWriterQos method*), 131
__init__() (*rti.connextdds.DataWriterResourceLimits method*), 139
__init__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 144
__init__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 142
__init__() (*rti.connextdds.DataWriterSeq method*), 145
__init__() (*rti.connextdds.DataWriterShmemRefTransferModeSettings method*), 146
__init__() (*rti.connextdds.DataWriterTransferMode method*), 147
__init__() (*rti.connextdds.Deadline method*), 148
__init__() (*rti.connextdds.DestinationOrder method*), 149
__init__() (*rti.connextdds.DestinationOrderKind method*), 151
__init__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__init__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__init__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__init__() (*rti.connextdds.Discovery method*), 154
__init__() (*rti.connextdds.DiscoveryConfig method*), 155
__init__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__init__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__init__() (*rti.connextdds.DomainParticipant method*), 165
__init__() (*rti.connextdds.DomainParticipantConfigParams method*), 171
__init__() (*rti.connextdds.DomainParticipantFactoryQos method*), 172
__init__() (*rti.connextdds.DomainParticipantListener method*), 173
__init__() (*rti.connextdds.DomainParticipantProtocolStatus method*), 173
__init__() (*rti.connextdds.DomainParticipantQos method*), 174
__init__() (*rti.connextdds.DomainParticipantResourceLimits method*), 178
__init__() (*rti.connextdds.DomainParticipantSeq method*), 184
__init__() (*rti.connextdds.Durability method*), 186
__init__() (*rti.connextdds.DurabilityKind method*), 188
__init__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 187
__init__() (*rti.connextdds.DurabilityService method*), 189
__init__() (*rti.connextdds.Duration method*), 190
__init__() (*rti.connextdds.DynamicData method*), 229
__init__() (*rti.connextdds.DynamicData.ContentFilter method*), 192
__init__() (*rti.connextdds.DynamicData.ContentFilteredTopic method*), 193
__init__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 194
__init__() (*rti.connextdds.DynamicData.DataReader method*), 197
__init__() (*rti.connextdds.DynamicData.DataReaderListener method*), 202
__init__() (*rti.connextdds.DynamicData.DataReader.Selector method*), 196

__init__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__init__() (*rti.connextdds.DynamicData.DataWriter method*), 205
__init__() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
__init__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__init__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__init__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__init__() (*rti.connextdds.DynamicData.FieldsIterator method*), 216
__init__() (*rti.connextdds.DynamicData.FieldsView method*), 217
__init__() (*rti.connextdds.DynamicData.IndexIterator method*), 217
__init__() (*rti.connextdds.DynamicDataInfo method*), 257
__init__() (*rti.connextdds.DynamicData.ItemsIterator method*), 218
__init__() (*rti.connextdds.DynamicData.ItemsView method*), 218
__init__() (*rti.connextdds.DynamicData.ITopicDescription method*), 217
__init__() (*rti.connextdds.DynamicData.LoanedSample method*), 218
__init__() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
__init__() (*rti.connextdds.DynamicDataMemberInfo method*), 257
__init__() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 219
__init__() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 220
__init__() (*rti.connextdds.DynamicData.NoOpTopicListener method*), 222
__init__() (*rti.connextdds.DynamicDataProperty method*), 258
__init__() (*rti.connextdds.DynamicData.Sample method*), 222
__init__() (*rti.connextdds.DynamicDataSeq method*), 259
__init__() (*rti.connextdds.DynamicData.SharedSamples method*), 222
__init__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__init__() (*rti.connextdds.DynamicData.Topic method*), 223
__init__() (*rti.connextdds.DynamicData.TopicDescription method*), 224
__init__() (*rti.connextdds.DynamicData.TopicListener method*), 225
__init__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__init__() (*rti.connextdds.DynamicData.TopicTypeSupport method*), 227
__init__() (*rti.connextdds.DynamicDataTypeSerializationProperty method*), 263
__init__() (*rti.connextdds.DynamicData.ValidLoanedSamples method*), 228
__init__() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
__init__() (*rti.connextdds.DynamicData.WriterContentFilterHelper method*), 229
__init__() (*rti.connextdds.DynamicType method*), 264
__init__() (*rti.connextdds.DynamicTypePrintFormatProperty method*), 265
__init__() (*rti.connextdds.EndpointGroup method*), 266
__init__() (*rti.connextdds.EndpointGroupSeq method*), 267
__init__() (*rti.connextdds.EndpointGroupVector method*), 268
__init__() (*rti.connextdds.Entity method*), 269
__init__() (*rti.connextdds.EntityFactory method*), 270
__init__() (*rti.connextdds.EntityLock method*), 270
__init__() (*rti.connextdds.EntityName method*), 270
__init__() (*rti.connextdds.EntitySeq method*), 272
__init__() (*rti.connextdds.EnumMember method*), 273
__init__() (*rti.connextdds.EnumMemberSeq method*), 274
__init__() (*rti.connextdds.EnumType method*), 276
__init__() (*rti.connextdds.Event method*), 277
__init__() (*rti.connextdds.EventCount32 method*), 277
__init__() (*rti.connextdds.EventCount64 method*), 278
__init__() (*rti.connextdds.ExpressionProperty method*), 278
__init__() (*rti.connextdds.ExtensibilityKind method*), 280
__init__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 279
__init__() (*rti.connextdds.Filter method*), 281
__init__() (*rti.connextdds.FilterSampleInfo method*), 282
__init__() (*rti.connextdds.Float32Seq method*), 285
__init__() (*rti.connextdds.Float32Type method*), 287
__init__() (*rti.connextdds.Float64Seq method*), 288
__init__() (*rti.connextdds.Float64Type method*), 290
__init__() (*rti.connextdds.Float128Seq method*), 283
__init__() (*rti.connextdds.Float128Type method*), 285
__init__() (*rti.connextdds.FlowController method*), 290
__init__() (*rti.connextdds.FlowControllerProperty method*), 291
__init__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 294
__init__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 293
__init__() (*rti.connextdds.FlowControllerTokenBucketProperty method*), 295
__init__() (*rti.connextdds.GenerationCount method*), 296
__init__() (*rti.connextdds.GroupData method*), 296
__init__() (*rti.connextdds.GuardCondition method*), 297
__init__() (*rti.connextdds.Guid method*), 298
__init__() (*rti.connextdds.heap_monitoring.HeapMonitoringParams method*), 798

__init__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__init__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__init__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 802
__init__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__init__() (*rti.connextdds.History method*), 299
__init__() (*rti.connextdds.HistoryKind method*), 301
__init__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__init__() (*rti.connextdds.IAnyDataReader method*), 301
__init__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__init__() (*rti.connextdds.IAnyDataWriter method*), 304
__init__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__init__() (*rti.connextdds.IAnyTopic method*), 307
__init__() (*rti.connextdds.IAnyTopicSeq method*), 309
__init__() (*rti.connextdds.ICondition method*), 310
__init__() (*rti.connextdds.IConditionSeq method*), 311
__init__() (*rti.connextdds.IDataReader method*), 313
__init__() (*rti.connextdds.IEntity method*), 313
__init__() (*rti.connextdds.IEntitySeq method*), 314
__init__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 318
__init__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 317
__init__() (*rti.connextdds.InconsistentTopicStatus method*), 319
__init__() (*rti.connextdds.InstanceHandle method*), 320
__init__() (*rti.connextdds.InstanceHandleSeq method*), 321
__init__() (*rti.connextdds.InstanceState method*), 323
__init__() (*rti.connextdds.InstanceStateConsistencyKind method*), 325
__init__() (*rti.connextdds.Int8Seq method*), 336
__init__() (*rti.connextdds.Int8Type method*), 337
__init__() (*rti.connextdds.Int16Seq method*), 327
__init__() (*rti.connextdds.Int16Type method*), 328
__init__() (*rti.connextdds.Int32Seq method*), 329
__init__() (*rti.connextdds.Int32Type method*), 331
__init__() (*rti.connextdds.Int32Vector method*), 331
__init__() (*rti.connextdds.Int64Seq method*), 333
__init__() (*rti.connextdds.Int64Type method*), 335
__init__() (*rti.connextdds.InvalidLocalAdvanceNoticeStatus method*), 338
__init__() (*rti.connextdds.IReadCondition method*), 316
__init__() (*rti.connextdds.ItemsView method*), 338
__init__() (*rti.connextdds.ITopicDescription method*), 317
__init__() (*rti.connextdds.KeysView method*), 338
__init__() (*rti.connextdds.LatencyBudget method*), 339
__init__() (*rti.connextdds.Lifespan method*), 339
__init__() (*rti.connextdds.Liveliness method*), 340
__init__() (*rti.connextdds.LivelinessChangedStatus method*), 341
__init__() (*rti.connextdds.LivelinessKind method*), 343
__init__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__init__() (*rti.connextdds.LivelinessLostStatus method*), 343
__init__() (*rti.connextdds.LoanedDynamicData method*), 344
__init__() (*rti.connextdds.Locator method*), 344
__init__() (*rti.connextdds.LocatorFilter method*), 345
__init__() (*rti.connextdds.LocatorFilterElement method*), 345
__init__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__init__() (*rti.connextdds.LocatorKind method*), 351
__init__() (*rti.connextdds.LocatorKind.LocatorKind method*), 350
__init__() (*rti.connextdds.LocatorSeq method*), 352
__init__() (*rti.connextdds.LocatorVector method*), 354
__init__() (*rti.connextdds.LogCategory method*), 356
__init__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__init__() (*rti.connextdds.LogFacility method*), 358
__init__() (*rti.connextdds.Logger method*), 361
__init__() (*rti.connextdds.LogLevel method*), 360
__init__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__init__() (*rti.connextdds.LogMessage method*), 360
__init__() (*rti.connextdds.LongDouble method*), 362
__init__() (*rti.connextdds.Member method*), 363
__init__() (*rti.connextdds.MemberSeq method*), 365
__init__() (*rti.connextdds.Monitoring method*), 367
__init__() (*rti.connextdds.MonitoringDedicatedParticipantSettings method*), 367
__init__() (*rti.connextdds.MonitoringDistributionSettings method*), 368
__init__() (*rti.connextdds.MonitoringEventDistributionSettings method*), 369
__init__() (*rti.connextdds.MonitoringLoggingDistributionSettings method*), 369
__init__() (*rti.connextdds.MonitoringLoggingForwardingSettings method*), 370
__init__() (*rti.connextdds.MonitoringMetricSelection method*), 371
__init__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__init__() (*rti.connextdds.MonitoringPeriodicDistributionSettings method*), 374
__init__() (*rti.connextdds.MonitoringTelemetryData method*), 374
__init__() (*rti.connextdds.MulticastMapping method*), 376
__init__() (*rti.connextdds.MulticastMappingSeq method*), 377
__init__() (*rti.connextdds.MultiChannel method*), 375
__init__() (*rti.connextdds.NoOpAnyDataReaderListener method*), 378

__init__() (*rti.connextdds.NoOpAnyDataWriterListener method*), 379
__init__() (*rti.connextdds.NoOpAnyTopicListener method*), 380
__init__() (*rti.connextdds.NoOpDataReaderListener method*), 381
__init__() (*rti.connextdds.NoOpDataWriterListener method*), 382
__init__() (*rti.connextdds.NoOpDomainParticipantListener method*), 383
__init__() (*rti.connextdds.NoOpPublisherListener method*), 383
__init__() (*rti.connextdds.NoOpSubscriberListener method*), 383
__init__() (*rti.connextdds.NoOpTopicListener method*), 383
__init__() (*rti.connextdds.OctetType method*), 384
__init__() (*rti.connextdds.OfferedDeadlineMissedStatus method*), 385
__init__() (*rti.connextdds.OfferedIncompatibleQosStatus method*), 385
__init__() (*rti.connextdds.Ownership method*), 386
__init__() (*rti.connextdds.OwnershipKind method*), 387
__init__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__init__() (*rti.connextdds.OwnershipStrength method*), 388
__init__() (*rti.connextdds.ParticipantBuiltinTopicData method*), 425
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilter method*), 389
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 389
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 394
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 402
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 410
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription method*), 414
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample method*), 414
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 415
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 416
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener method*), 418
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.Sample method*), 418
__init__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.SharedSamples method*), 419
__init__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 429
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.Topic method*), 419
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicDescription method*), 420
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicListener method*), 421
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples method*), 423
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
__init__() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilterHelper method*), 424
__init__() (*rti.connextdds.Partition method*), 431
__init__() (*rti.connextdds.PersistentJournalKind method*), 432
__init__() (*rti.connextdds.PersistentStorageSettings method*), 433
__init__() (*rti.connextdds.PersistentSynchronizationKind method*), 434
__init__() (*rti.connextdds.Presentation method*), 435
__init__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__init__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*),

437
__init__() (*rti.connextdds.PrintFormat method*), 440
__init__() (*rti.connextdds.PrintFormatKind method*), 442
__init__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 441
__init__() (*rti.connextdds.PrintFormat.PrintFormat method*), 439
__init__() (*rti.connextdds.PrintFormatProperty method*), 443
__init__() (*rti.connextdds.ProductVersion method*), 443
__init__() (*rti.connextdds.Property method*), 444
__init__() (*rti.connextdds.ProtocolVersion method*), 446
__init__() (*rti.connextdds.PublicationBuiltinTopicData method*), 483
__init__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilter method*), 446
__init__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 447
__init__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 452
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 450
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 458
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 460
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 468
__init__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__init__() (*rti.connextdds.PublicationBuiltinTopicData.ITopicDescription method*), 472
__init__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSample method*), 472
__init__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 473
__init__() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 473
__init__() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 474
__init__() (*rti.connextdds.PublicationBuiltinTopicData.NoOpTopicListener method*), 476
__init__() (*rti.connextdds.PublicationBuiltinTopicData.Sample method*), 476
__init__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__init__() (*rti.connextdds.PublicationBuiltinTopicData.SharedSamples method*), 477
__init__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 488
__init__() (*rti.connextdds.PublicationBuiltinTopicData.Topic method*), 477
__init__() (*rti.connextdds.PublicationBuiltinTopicData.TopicDescription method*), 478
__init__() (*rti.connextdds.PublicationBuiltinTopicData.TopicListener method*), 479
__init__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__init__() (*rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples method*), 481
__init__() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
__init__() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper method*), 482
__init__() (*rti.connextdds.PublicationMatchedStatus method*), 490
__init__() (*rti.connextdds.Publisher method*), 494
__init__() (*rti.connextdds.PublisherListener method*), 495
__init__() (*rti.connextdds.PublisherQos method*), 495
__init__() (*rti.connextdds.PublisherSeq method*), 498
__init__() (*rti.connextdds.PublishMode method*), 491
__init__() (*rti.connextdds.PublishModeKind method*), 493
__init__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__init__() (*rti.connextdds.QosPolicyCount method*), 500
__init__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__init__() (*rti.connextdds.QosPrintFormat method*), 502
__init__() (*rti.connextdds.QosProvider method*), 503
__init__() (*rti.connextdds.QosProviderParams method*), 506
__init__() (*rti.connextdds.Query method*), 507
__init__() (*rti.connextdds.QueryCondition method*), 508
__init__() (*rti.connextdds.Rank method*), 509
__init__() (*rti.connextdds.ReadCondition method*), 509
__init__() (*rti.connextdds.ReaderDataLifecycle method*), 510
__init__() (*rti.connextdds.ReceiverPool method*), 511
__init__() (*rti.connextdds.Reliability method*), 512

__init__() (*rti.connextdds.ReliabilityKind method*), 514
__init__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__init__() (*rti.connextdds.ReliableReaderActivityChangedStatus method*), 515
__init__() (*rti.connextdds.ReliableWriterCacheChangedStatus method*), 515
__init__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 518
__init__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__init__() (*rti.connextdds.RequestedDeadlineMissedStatus method*), 518
__init__() (*rti.connextdds.RequestedIncompatibleQosStatus method*), 519
__init__() (*rti.connextdds.ResourceLimits method*), 519
__init__() (*rti.connextdds.RtpsReliableReaderProtocol method*), 520
__init__() (*rti.connextdds.RtpsReliableWriterProtocol method*), 521
__init__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__init__() (*rti.connextdds.RtpsWellKnownPorts method*), 527
__init__() (*rti.connextdds.SampleFlag method*), 528
__init__() (*rti.connextdds.SampleIdentity method*), 530
__init__() (*rti.connextdds.SampleInfo method*), 531
__init__() (*rti.connextdds.SampleLostState method*), 534
__init__() (*rti.connextdds.SampleLostStatus method*), 536
__init__() (*rti.connextdds.SampleRejectedState method*), 537
__init__() (*rti.connextdds.SampleRejectedStatus method*), 539
__init__() (*rti.connextdds.SampleState method*), 540
__init__() (*rti.connextdds.SequenceNumber method*), 543
__init__() (*rti.connextdds.SequenceType method*), 544
__init__() (*rti.connextdds.Service method*), 544
__init__() (*rti.connextdds.ServiceKind method*), 547
__init__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__init__() (*rti.connextdds.ServiceRequest method*), 582
__init__() (*rti.connextdds.ServiceRequestAcceptedStatus method*), 583
__init__() (*rti.connextdds.ServiceRequest.ContentFilter method*), 547
__init__() (*rti.connextdds.ServiceRequest.ContentFilteredTopic method*), 548
__init__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__init__() (*rti.connextdds.ServiceRequest.DataReader method*), 553
__init__() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 557
__init__() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 551
__init__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__init__() (*rti.connextdds.ServiceRequest.DataWriter method*), 561
__init__() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 568
__init__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__init__() (*rti.connextdds.ServiceRequestId method*), 585
__init__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__init__() (*rti.connextdds.ServiceRequest.ITopicDescription method*), 572
__init__() (*rti.connextdds.ServiceRequest.LoanedSample method*), 572
__init__() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 573
__init__() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 573
__init__() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 574
__init__() (*rti.connextdds.ServiceRequest.NoOpTopicListener method*), 575
__init__() (*rti.connextdds.ServiceRequest.Sample method*), 576
__init__() (*rti.connextdds.ServiceRequestSeq method*), 586
__init__() (*rti.connextdds.ServiceRequest.SharedSamples method*), 576
__init__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 588
__init__() (*rti.connextdds.ServiceRequest.Topic method*), 577
__init__() (*rti.connextdds.ServiceRequest.TopicDescription method*), 578
__init__() (*rti.connextdds.ServiceRequest.TopicListener method*), 578
__init__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__init__() (*rti.connextdds.ServiceRequest.ValidLoanedSamples method*), 581
__init__() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 581
__init__() (*rti.connextdds.ServiceRequest.WriterContentFilterHelper method*), 582
__init__() (*rti.connextdds.StatusCondition method*), 590
__init__() (*rti.connextdds.StatusMask method*), 593
__init__() (*rti.connextdds.StreamKind method*), 595
__init__() (*rti.connextdds.StringMap method*), 597
__init__() (*rti.connextdds.StringPairSeq method*), 599
__init__() (*rti.connextdds.StringSeq method*), 601

__init__() (*rti.connextdds.StringType method*), 602
__init__() (*rti.connextdds.StructType method*), 603
__init__() (*rti.connextdds.Subscriber method*), 604
__init__() (*rti.connextdds.SubscriberListener method*), 606
__init__() (*rti.connextdds.SubscriberQos method*), 606
__init__() (*rti.connextdds.SubscriberSeq method*), 608
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData method*), 647
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter method*), 610
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 610
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 615
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 620
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 624
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription method*), 635
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample method*), 636
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples method*), 636
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 637
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 638
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpTopicListener method*), 640
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.Sample method*), 640
__init__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples method*), 640
__init__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.Topic method*), 641
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription method*), 642
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicListener method*), 642
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 643
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples method*), 645
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 645
__init__() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper method*), 646
__init__() (*rti.connextdds.SubscriptionMatchedStatus method*), 654
__init__() (*rti.connextdds.SuspendedPublication method*), 655
__init__() (*rti.connextdds.SyslogVerbosity method*), 656
__init__() (*rti.connextdds.SystemResourceLimits method*), 656
__init__() (*rti.connextdds.ThreadContext method*), 657
__init__() (*rti.connextdds.ThreadSettings method*), 657
__init__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 659
__init__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 658
__init__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__init__() (*rti.connextdds.Time method*), 663
__init__() (*rti.connextdds.TimeBasedFilter method*), 665
__init__() (*rti.connextdds.Topic method*), 666
__init__() (*rti.connextdds.TopicBuiltinTopicData method*), 703
__init__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilter method*), 667
__init__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 667
__init__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669

__init__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 672
__init__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 677
__init__() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
__init__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__init__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 680
__init__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 688
__init__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__init__() (*rti.connextdds.TopicBuiltinTopicData.ITopicDescription method*), 692
__init__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSample method*), 692
__init__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 693
__init__() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 693
__init__() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 694
__init__() (*rti.connextdds.TopicBuiltinTopicData.NoOpTopicListener method*), 696
__init__() (*rti.connextdds.TopicBuiltinTopicData.Sample method*), 696
__init__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__init__() (*rti.connextdds.TopicBuiltinTopicData.SharedSamples method*), 697
__init__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 707
__init__() (*rti.connextdds.TopicBuiltinTopicData.Topic method*), 697
__init__() (*rti.connextdds.TopicBuiltinTopicData.TopicDescription method*), 698
__init__() (*rti.connextdds.TopicBuiltinTopicData.TopicListener method*), 698
__init__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__init__() (*rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples method*), 701
__init__() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 701
__init__() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper method*), 702
__init__() (*rti.connextdds.TopicData method*), 709
__init__() (*rti.connextdds.TopicDescription method*), 710
__init__() (*rti.connextdds.TopicListener method*), 710
__init__() (*rti.connextdds.TopicQos method*), 710
__init__() (*rti.connextdds.TopicQuery method*), 715
__init__() (*rti.connextdds.TopicQueryData method*), 715
__init__() (*rti.connextdds.TopicQueryDispatch method*), 716
__init__() (*rti.connextdds.TopicQuerySelection method*), 716
__init__() (*rti.connextdds.TopicQuerySelectionKind method*), 718
__init__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 717
__init__() (*rti.connextdds.TopicSeq method*), 720
__init__() (*rti.connextdds.TransportBuiltin method*), 721
__init__() (*rti.connextdds.TransportBuiltinMask method*), 723
__init__() (*rti.connextdds.TransportClassId method*), 727
__init__() (*rti.connextdds.TransportClassId.TransportClassId method*), 726
__init__() (*rti.connextdds.TransportInfo method*), 728
__init__() (*rti.connextdds.TransportInfoSeq method*), 729
__init__() (*rti.connextdds.TransportInfoVector method*), 731
__init__() (*rti.connextdds.TransportMulticast method*), 732
__init__() (*rti.connextdds.TransportMulticastKind method*), 734
__init__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__init__() (*rti.connextdds.TransportMulticastMapping method*), 735
__init__() (*rti.connextdds.TransportMulticastMappingFunction method*), 735
__init__() (*rti.connextdds.TransportMulticastSeq method*), 736
__init__() (*rti.connextdds.TransportMulticastSettings method*), 738
__init__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__init__() (*rti.connextdds.TransportPriority method*), 741
__init__() (*rti.connextdds.TransportSelection method*), 742
__init__() (*rti.connextdds.TransportUnicast method*), 742
__init__() (*rti.connextdds.TransportUnicastSettings method*), 743
__init__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__init__() (*rti.connextdds.TriggeredConditions method*), 746

__init__() (*rti.connextdds.TriggeredConditionsIterator method*), 746
__init__() (*rti.connextdds.TypeConsistencyEnforcement method*), 746
__init__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 749
__init__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 748
__init__() (*rti.connextdds.TypeKind method*), 755
__init__() (*rti.connextdds.TypeKind.TypeKind method*), 753
__init__() (*rti.connextdds.TypeSupport method*), 756
__init__() (*rti.connextdds.Uint8Seq method*), 765
__init__() (*rti.connextdds.Uint8Type method*), 767
__init__() (*rti.connextdds.Uint16Seq method*), 757
__init__() (*rti.connextdds.Uint16Type method*), 759
__init__() (*rti.connextdds.Uint32Seq method*), 760
__init__() (*rti.connextdds.Uint32Type method*), 762
__init__() (*rti.connextdds.Uint64Seq method*), 762
__init__() (*rti.connextdds.Uint64Type method*), 764
__init__() (*rti.connextdds.UnidimensionalCollectionType method*), 767
__init__() (*rti.connextdds.UnionMember method*), 768
__init__() (*rti.connextdds.UnionMemberSeq method*), 769
__init__() (*rti.connextdds.UnionType method*), 771
__init__() (*rti.connextdds.UserData method*), 772
__init__() (*rti.connextdds.UserDataSample method*), 772
__init__() (*rti.connextdds.ValuesView method*), 772
__init__() (*rti.connextdds.VendorId method*), 773
__init__() (*rti.connextdds.Verbosity method*), 775
__init__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__init__() (*rti.connextdds.ViewState method*), 776
__init__() (*rti.connextdds.WaitSet method*), 779
__init__() (*rti.connextdds.WaitSetProperty method*), 780
__init__() (*rti.connextdds.WcharSeq method*), 782
__init__() (*rti.connextdds.WcharType method*), 783
__init__() (*rti.connextdds.WireProtocol method*), 784
__init__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__init__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__init__() (*rti.connextdds.WriteParams method*), 787
__init__() (*rti.connextdds.WriterDataLifecycle method*), 788
__init__() (*rti.connextdds.WstringSeq method*), 789
__init__() (*rti.connextdds.WStringType method*), 778
__int__() (*rti.connextdds.AcknowledgmentKind method*), 40
__int__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 39
__int__() (*rti.connextdds.ActivityContextMask method*), 42
__int__() (*rti.connextdds.CdrPaddingKind method*), 67
__int__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__int__() (*rti.connextdds.compliance.RpcMask method*), 792
__int__() (*rti.connextdds.compliance.XTypesMask method*), 795
__int__() (*rti.connextdds.CompressionIdMask method*), 76
__int__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__int__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__int__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 144
__int__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 142
__int__() (*rti.connextdds.DestinationOrderKind method*), 151
__int__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__int__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__int__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__int__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__int__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__int__() (*rti.connextdds.DurabilityKind method*), 188
__int__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 187
__int__() (*rti.connextdds.Duration method*), 191
__int__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__int__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__int__() (*rti.connextdds.EnumMember method*), 273
__int__() (*rti.connextdds.ExtensibilityKind method*), 280
__int__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 279
__int__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 295
__int__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 293
__int__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__int__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__int__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 802
__int__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__int__() (*rti.connextdds.HistoryKind method*), 301
__int__() (*rti.connextdds.HistoryKind.HistoryKind method*),

300
__int__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 318
__int__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 317
__int__() (*rti.connextdds.InstanceState method*), 323
__int__() (*rti.connextdds.InstanceStateConsistencyKind method*), 325
__int__() (*rti.connextdds.LivelinessKind method*), 343
__int__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__int__() (*rti.connextdds.LocatorKind method*), 351
__int__() (*rti.connextdds.LocatorKind.LocatorKind method*), 350
__int__() (*rti.connextdds.LogCategory method*), 356
__int__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__int__() (*rti.connextdds.LogFacility method*), 358
__int__() (*rti.connextdds.LogLevel method*), 360
__int__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__int__() (*rti.connextdds.OwnershipKind method*), 388
__int__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__int__() (*rti.connextdds.PersistentJournalKind method*), 432
__int__() (*rti.connextdds.PersistentSynchronizationKind method*), 434
__int__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__int__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__int__() (*rti.connextdds.PrintFormat method*), 440
__int__() (*rti.connextdds.PrintFormatKind method*), 442
__int__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 441
__int__() (*rti.connextdds.PrintFormat.PrintFormat method*), 439
__int__() (*rti.connextdds.PublishModeKind method*), 493
__int__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__int__() (*rti.connextdds.ReliabilityKind method*), 514
__int__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__int__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 518
__int__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__int__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__int__() (*rti.connextdds.SampleFlag method*), 529
__int__() (*rti.connextdds.SampleLostState method*), 534
__int__() (*rti.connextdds.SampleRejectedState method*), 537
__int__() (*rti.connextdds.SampleState method*), 540
__int__() (*rti.connextdds.SequenceNumber method*), 543
__int__() (*rti.connextdds.ServiceKind method*), 547
__int__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__int__() (*rti.connextdds.ServiceRequestId method*), 585
__int__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__int__() (*rti.connextdds.StatusMask method*), 593
__int__() (*rti.connextdds.StreamKind method*), 595
__int__() (*rti.connextdds.SyslogVerbosity method*), 656
__int__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 660
__int__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 659
__int__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__int__() (*rti.connextdds.TopicQuerySelectionKind method*), 718
__int__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 718
__int__() (*rti.connextdds.TransportBuiltinMask method*), 723
__int__() (*rti.connextdds.TransportClassId method*), 728
__int__() (*rti.connextdds.TransportClassId.TransportClassId method*), 726
__int__() (*rti.connextdds.TransportMulticastKind method*), 734
__int__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__int__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 749
__int__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 748
__int__() (*rti.connextdds.TypeKind method*), 755
__int__() (*rti.connextdds.TypeKind.TypeKind method*), 754
__int__() (*rti.connextdds.Verbosity method*), 775
__int__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__int__() (*rti.connextdds.ViewState method*), 776
__int__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__int__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__ior__() (*rti.connextdds.ActivityContextMask method*), 42
__ior__() (*rti.connextdds.compliance.RpcMask method*), 792
__ior__() (*rti.connextdds.compliance.XTypesMask method*), 795
__ior__() (*rti.connextdds.CompressionIdMask method*), 76
__ior__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__ior__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__ior__() (*rti.connextdds.InstanceState method*), 323
__ior__() (*rti.connextdds.RtpsReservedPortKindMask method*), 524
__ior__() (*rti.connextdds.SampleFlag method*), 529

__ior__() (*rti.connextdds.SampleLostState method*), 534
__ior__() (*rti.connextdds.SampleRejectedState method*), 537
__ior__() (*rti.connextdds.SampleState method*), 540
__ior__() (*rti.connextdds.StatusMask method*), 593
__ior__() (*rti.connextdds.StreamKind method*), 595
__ior__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__ior__() (*rti.connextdds.TransportBuiltinMask method*), 723
__ior__() (*rti.connextdds.ViewState method*), 776
__irshift__() (*rti.connextdds.ActivityContextMask method*), 42
__irshift__() (*rti.connextdds.compliance.RpcMask method*), 792
__irshift__() (*rti.connextdds.compliance.XTypesMask method*), 795
__irshift__() (*rti.connextdds.CompressionIdMask method*), 76
__irshift__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__irshift__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__irshift__() (*rti.connextdds.InstanceState method*), 323
__irshift__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__irshift__() (*rti.connextdds.SampleFlag method*), 529
__irshift__() (*rti.connextdds.SampleLostState method*), 534
__irshift__() (*rti.connextdds.SampleRejectedState method*), 538
__irshift__() (*rti.connextdds.SampleState method*), 541
__irshift__() (*rti.connextdds.StatusMask method*), 593
__irshift__() (*rti.connextdds.StreamKind method*), 596
__irshift__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__irshift__() (*rti.connextdds.TransportBuiltinMask method*), 723
__irshift__() (*rti.connextdds.ViewState method*), 777
__isub__() (*rti.connextdds.Duration method*), 191
__isub__() (*rti.connextdds.SequenceNumber method*), 543
__isub__() (*rti.connextdds.Time method*), 664
__isub__() (*rti.connextdds.WaitSet method*), 779
__iter__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__iter__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__iter__() (*rti.connextdds.AnyTopicSeq method*), 53
__iter__() (*rti.connextdds.BoolSeq method*), 59
__iter__() (*rti.connextdds.ByteVector method*), 65
__iter__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__iter__() (*rti.connextdds.CharSeq method*), 71
__iter__() (*rti.connextdds.ConditionSeq method*), 79
__iter__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__iter__() (*rti.connextdds.CookieSeq method*), 86
__iter__() (*rti.connextdds.CookieVector method*), 88
__iter__() (*rti.connextdds.DataReader.LoanedSample method*), 89
__iter__() (*rti.connextdds.DataReader.LoanedSamples method*), 89
__iter__() (*rti.connextdds.DataReaderSeq method*), 114
__iter__() (*rti.connextdds.DataWriterSeq method*), 145
__iter__() (*rti.connextdds.DomainParticipantSeq method*), 184
__iter__() (*rti.connextdds.DynamicData method*), 230
__iter__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
__iter__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__iter__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__iter__() (*rti.connextdds.DynamicData.FieldsIterator method*), 216
__iter__() (*rti.connextdds.DynamicData.FieldsView method*), 217
__iter__() (*rti.connextdds.DynamicData.IndexIterator method*), 217
__iter__() (*rti.connextdds.DynamicData.ItemsIterator method*), 218
__iter__() (*rti.connextdds.DynamicData.ItemsView method*), 218
__iter__() (*rti.connextdds.DynamicData.LoanedSample method*), 218
__iter__() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
__iter__() (*rti.connextdds.DynamicData.Sample method*), 222
__iter__() (*rti.connextdds.DynamicDataSeq method*), 259
__iter__() (*rti.connextdds.DynamicData.SharedSamples method*), 223
__iter__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__iter__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__iter__() (*rti.connextdds.DynamicData.ValidLoanedSamples method*), 228
__iter__() (*rti.connextdds.EndpointGroupSeq method*), 267
__iter__() (*rti.connextdds.EndpointGroupVector method*), 269
__iter__() (*rti.connextdds.EntitySeq method*), 272
__iter__() (*rti.connextdds.EnumMemberSeq method*), 275
__iter__() (*rti.connextdds.Filter method*), 281
__iter__() (*rti.connextdds.Float32Seq method*), 286
__iter__() (*rti.connextdds.Float64Seq method*), 288
__iter__() (*rti.connextdds.Float128Seq method*), 283
__iter__() (*rti.connextdds.GroupData method*), 297
__iter__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__iter__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__iter__() (*rti.connextdds.IAnyTopicSeq method*), 309
__iter__() (*rti.connextdds.IConditionSeq method*), 311
__iter__() (*rti.connextdds.IEntitySeq method*), 315
__iter__() (*rti.connextdds.InstanceHandleSeq method*), 321
__iter__() (*rti.connextdds.Int8Seq method*), 336
__iter__() (*rti.connextdds.Int16Seq method*), 327
__iter__() (*rti.connextdds.Int32Seq method*), 329
__iter__() (*rti.connextdds.Int32Vector method*), 332
__iter__() (*rti.connextdds.Int64Seq method*), 333

__iter__() (*rti.connextdds.ItemsView method*), 338
__iter__() (*rti.connextdds.KeysView method*), 338
__iter__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__iter__() (*rti.connextdds.LocatorSeq method*), 352
__iter__() (*rti.connextdds.LocatorVector method*), 354
__iter__() (*rti.connextdds.MemberSeq method*), 365
__iter__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__iter__() (*rti.connextdds.MulticastMappingSeq method*), 377
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 412
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample method*), 414
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.Sample method*), 418
__iter__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.SharedSamples method*), 419
__iter__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__iter__() (*rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples method*), 423
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSample method*), 472
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 473
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.Sample method*), 476
__iter__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.SharedSamples method*), 477
__iter__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__iter__() (*rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples method*), 481
__iter__() (*rti.connextdds.PublisherSeq method*), 498
__iter__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__iter__() (*rti.connextdds.Query method*), 507
__iter__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__iter__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__iter__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__iter__() (*rti.connextdds.ServiceRequest.LoanedSample method*), 572
__iter__() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 573
__iter__() (*rti.connextdds.ServiceRequest.Sample method*), 576
__iter__() (*rti.connextdds.ServiceRequestSeq method*), 586
__iter__() (*rti.connextdds.ServiceRequest.SharedSamples method*), 576
__iter__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
__iter__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__iter__() (*rti.connextdds.ServiceRequest.ValidLoanedSamples method*), 581
__iter__() (*rti.connextdds.StringMap method*), 597
__iter__() (*rti.connextdds.StringPairSeq method*), 599
__iter__() (*rti.connextdds.StringSeq method*), 601
__iter__() (*rti.connextdds.SubscriberSeq method*), 609
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample method*), 636
__iter__() (*rti.connextdds.SubscriptionBuiltinTopic-*

*Data.LoanedSamples method*), 636
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.Sample method*), 640
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples method*), 641
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 652
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
__iter__() (*rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples method*), 645
__iter__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__iter__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__iter__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__iter__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSample method*), 692
__iter__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 693
__iter__() (*rti.connextdds.TopicBuiltinTopicData.Sample method*), 696
__iter__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__iter__() (*rti.connextdds.TopicBuiltinTopicData.SharedSamples method*), 697
__iter__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
__iter__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__iter__() (*rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples method*), 701
__iter__() (*rti.connextdds.TopicData method*), 709
__iter__() (*rti.connextdds.TopicSeq method*), 720
__iter__() (*rti.connextdds.TransportInfoSeq method*), 730
__iter__() (*rti.connextdds.TransportInfoVector method*), 731
__iter__() (*rti.connextdds.TransportMulticastSeq method*), 737
__iter__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 739
__iter__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__iter__() (*rti.connextdds.TriggeredConditions method*), 746
__iter__() (*rti.connextdds.Uint8Seq method*), 765
__iter__() (*rti.connextdds.Uint16Seq method*), 757
__iter__() (*rti.connextdds.Uint32Seq method*), 760
__iter__() (*rti.connextdds.Uint64Seq method*), 763
__iter__() (*rti.connextdds.UnionMemberSeq method*), 769
__iter__() (*rti.connextdds.UserData method*), 772
__iter__() (*rti.connextdds.ValuesView method*), 773
__iter__() (*rti.connextdds.WcharSeq method*), 782
__iter__() (*rti.connextdds.WstringSeq method*), 789
__ixor__() (*rti.connextdds.ActivityContextMask method*), 42
__ixor__() (*rti.connextdds.compliance.RpcMask method*), 792
__ixor__() (*rti.connextdds.compliance.XTypesMask method*), 795
__ixor__() (*rti.connextdds.CompressionIdMask method*), 76
__ixor__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__ixor__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__ixor__() (*rti.connextdds.InstanceState method*), 323
__ixor__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__ixor__() (*rti.connextdds.SampleFlag method*), 529
__ixor__() (*rti.connextdds.SampleLostState method*), 534
__ixor__() (*rti.connextdds.SampleRejectedState method*), 538
__ixor__() (*rti.connextdds.SampleState method*), 541
__ixor__() (*rti.connextdds.StatusMask method*), 593
__ixor__() (*rti.connextdds.StreamKind method*), 596
__ixor__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__ixor__() (*rti.connextdds.TransportBuiltinMask method*), 723
__ixor__() (*rti.connextdds.ViewState method*), 777
__le__() (*rti.connextdds.AcknowledgmentKind method*), 40
__le__() (*rti.connextdds.CdrPaddingKind method*), 67
__le__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__le__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 144
__le__() (*rti.connextdds.DestinationOrderKind method*), 151
__le__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__le__() (*rti.connextdds.DurabilityKind method*), 188
__le__() (*rti.connextdds.Duration method*), 191
__le__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__le__() (*rti.connextdds.ExtensibilityKind method*), 280
__le__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 295
__le__() (*rti.connextdds.Guid method*), 298
__le__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__le__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*),

802
__le__() (*rti.connextdds.HistoryKind method*), 301
__le__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 319
__le__() (*rti.connextdds.LivelinessKind method*), 343
__le__() (*rti.connextdds.LocatorKind method*), 351
__le__() (*rti.connextdds.LogCategory method*), 356
__le__() (*rti.connextdds.LogLevel method*), 360
__le__() (*rti.connextdds.OwnershipKind method*), 388
__le__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__le__() (*rti.connextdds.PrintFormat method*), 440
__le__() (*rti.connextdds.PrintFormatKind method*), 442
__le__() (*rti.connextdds.PublishModeKind method*), 493
__le__() (*rti.connextdds.ReliabilityKind method*), 514
__le__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 518
__le__() (*rti.connextdds.SequenceNumber method*), 543
__le__() (*rti.connextdds.ServiceKind method*), 547
__le__() (*rti.connextdds.ServiceRequestId method*), 585
__le__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 660
__le__() (*rti.connextdds.Time method*), 664
__le__() (*rti.connextdds.TopicQuerySelectionKind method*), 719
__le__() (*rti.connextdds.TransportClassId method*), 728
__le__() (*rti.connextdds.TransportMulticastKind method*), 734
__le__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 750
__le__() (*rti.connextdds.TypeKind method*), 755
__le__() (*rti.connextdds.Verbosity method*), 775
__le__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__len__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__len__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__len__() (*rti.connextdds.AnyTopicSeq method*), 53
__len__() (*rti.connextdds.BoolSeq method*), 59
__len__() (*rti.connextdds.ByteVector method*), 65
__len__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__len__() (*rti.connextdds.CharSeq method*), 71
__len__() (*rti.connextdds.ConditionSeq method*), 79
__len__() (*rti.connextdds.ContentFilteredTopicSeq method*), 83
__len__() (*rti.connextdds.CookieSeq method*), 86
__len__() (*rti.connextdds.CookieVector method*), 88
__len__() (*rti.connextdds.DataReader.LoanedSamples method*), 89
__len__() (*rti.connextdds.DataReaderSeq method*), 114
__len__() (*rti.connextdds.DataTag method*), 119
__len__() (*rti.connextdds.DataWriterSeq method*), 145
__len__() (*rti.connextdds.DomainParticipantSeq method*), 184
__len__() (*rti.connextdds.DynamicData method*), 230
__len__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
__len__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__len__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__len__() (*rti.connextdds.DynamicData.FieldsView method*), 217
__len__() (*rti.connextdds.DynamicData.ItemsView method*), 218
__len__() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
__len__() (*rti.connextdds.DynamicDataSeq method*), 260
__len__() (*rti.connextdds.DynamicData.SharedSamples method*), 223
__len__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__len__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__len__() (*rti.connextdds.EndpointGroupSeq method*), 267
__len__() (*rti.connextdds.EndpointGroupVector method*), 269
__len__() (*rti.connextdds.EntitySeq method*), 272
__len__() (*rti.connextdds.EnumMemberSeq method*), 275
__len__() (*rti.connextdds.Float32Seq method*), 286
__len__() (*rti.connextdds.Float64Seq method*), 288
__len__() (*rti.connextdds.Float128Seq method*), 283
__len__() (*rti.connextdds.Guid method*), 298
__len__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__len__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__len__() (*rti.connextdds.IAnyTopicSeq method*), 309
__len__() (*rti.connextdds.IConditionSeq method*), 311
__len__() (*rti.connextdds.IEntitySeq method*), 315
__len__() (*rti.connextdds.InstanceHandleSeq method*), 321
__len__() (*rti.connextdds.Int8Seq method*), 336
__len__() (*rti.connextdds.Int16Seq method*), 327
__len__() (*rti.connextdds.Int32Seq method*), 329
__len__() (*rti.connextdds.Int32Vector method*), 332
__len__() (*rti.connextdds.Int64Seq method*), 333
__len__() (*rti.connextdds.ItemsView method*), 338
__len__() (*rti.connextdds.KeysView method*), 338
__len__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__len__() (*rti.connextdds.LocatorSeq method*), 352
__len__() (*rti.connextdds.LocatorVector method*), 354
__len__() (*rti.connextdds.MemberSeq method*), 365
__len__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__len__() (*rti.connextdds.MulticastMappingSeq method*), 377
__len__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__len__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__len__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
__len__() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
__len__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427

__len__() (*rti.connextdds.ParticipantBuiltinTopicData.SharedSamples method*), 419
__len__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
__len__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__len__() (*rti.connextdds.Property method*), 444
__len__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__len__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
__len__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 470
__len__() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 473
__len__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__len__() (*rti.connextdds.PublicationBuiltinTopicData.SharedSamples method*), 477
__len__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
__len__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__len__() (*rti.connextdds.PublisherSeq method*), 498
__len__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__len__() (*rti.connextdds.Query method*), 507
__len__() (*rti.connextdds.QueryCondition method*), 508
__len__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__len__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__len__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__len__() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 573
__len__() (*rti.connextdds.ServiceRequestSeq method*), 586
__len__() (*rti.connextdds.ServiceRequest.SharedSamples method*), 576
__len__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
__len__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__len__() (*rti.connextdds.StringMap method*), 597
__len__() (*rti.connextdds.StringPairSeq method*), 599
__len__() (*rti.connextdds.StringSeq method*), 601
__len__() (*rti.connextdds.SubscriberSeq method*), 609
__len__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 612
__len__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__len__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__len__() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples method*), 637
__len__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__len__() (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples method*), 641
__len__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
__len__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
__len__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 669
__len__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__len__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__len__() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 693
__len__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__len__() (*rti.connextdds.TopicBuiltinTopicData.SharedSamples method*), 697
__len__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
__len__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__len__() (*rti.connextdds.TopicSeq method*), 720
__len__() (*rti.connextdds.TransportInfoSeq method*), 730
__len__() (*rti.connextdds.TransportInfoVector method*), 731
__len__() (*rti.connextdds.TransportMulticastSeq method*), 737
__len__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
__len__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__len__() (*rti.connextdds.TriggeredConditions method*), 746
__len__() (*rti.connextdds.Uint8Seq method*), 765
__len__() (*rti.connextdds.Uint16Seq method*), 757
__len__() (*rti.connextdds.Uint32Seq method*), 760
__len__() (*rti.connextdds.Uint64Seq method*), 763
__len__() (*rti.connextdds.UnionMemberSeq method*), 770
__len__() (*rti.connextdds.ValuesView method*), 773

__len__() (*rti.connextdds.WcharSeq method*), 782
__len__() (*rti.connextdds.WstringSeq method*), 789
__long__() (*rti.connextdds.EnumMember method*), 273
__lshift__() (*rti.connextdds.ActivityContextMask method*), 42
__lshift__() (*rti.connextdds.compliance.RpcMask method*), 792
__lshift__() (*rti.connextdds.compliance.XTypesMask method*), 795
__lshift__() (*rti.connextdds.CompressionIdMask method*), 76
__lshift__() (*rti.connextdds.DataReader method*), 92
__lshift__() (*rti.connextdds.DataReaderQos method*), 103
__lshift__() (*rti.connextdds.DataState method*), 116
__lshift__() (*rti.connextdds.DataStateEx method*), 118
__lshift__() (*rti.connextdds.DataWriter method*), 121
__lshift__() (*rti.connextdds.DataWriterQos method*), 131
__lshift__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__lshift__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__lshift__() (*rti.connextdds.DomainParticipant method*), 165
__lshift__() (*rti.connextdds.DomainParticipantFactoryQos method*), 172
__lshift__() (*rti.connextdds.DomainParticipantQos method*), 174
__lshift__() (*rti.connextdds.DynamicData.DataReader method*), 198
__lshift__() (*rti.connextdds.DynamicData.DataWriter method*), 206
__lshift__() (*rti.connextdds.InstanceState method*), 324
__lshift__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 395
__lshift__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 403
__lshift__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 453
__lshift__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 461
__lshift__() (*rti.connextdds.Publisher method*), 494
__lshift__() (*rti.connextdds.PublisherQos method*), 496
__lshift__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__lshift__() (*rti.connextdds.SampleFlag method*), 529
__lshift__() (*rti.connextdds.SampleLostState method*), 535
__lshift__() (*rti.connextdds.SampleRejectedState method*), 538
__lshift__() (*rti.connextdds.SampleState method*), 541
__lshift__() (*rti.connextdds.ServiceRequest.DataReader method*), 554
__lshift__() (*rti.connextdds.ServiceRequest.DataWriter method*), 561
__lshift__() (*rti.connextdds.StatusMask method*), 593
__lshift__() (*rti.connextdds.StreamKind method*), 596
__lshift__() (*rti.connextdds.SubscriberQos method*), 606
__lshift__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 616
__lshift__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 624
__lshift__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__lshift__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 673
__lshift__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 681
__lshift__() (*rti.connextdds.TopicQos method*), 711
__lshift__() (*rti.connextdds.TransportBuiltinMask method*), 723
__lshift__() (*rti.connextdds.ViewState method*), 777
__lt__() (*rti.connextdds.AcknowledgmentKind method*), 40
__lt__() (*rti.connextdds.CdrPaddingKind method*), 67
__lt__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__lt__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 144
__lt__() (*rti.connextdds.DestinationOrderKind method*), 151
__lt__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__lt__() (*rti.connextdds.DurabilityKind method*), 189
__lt__() (*rti.connextdds.Duration method*), 191
__lt__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__lt__() (*rti.connextdds.ExtensibilityKind method*), 280
__lt__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 295
__lt__() (*rti.connextdds.Guid method*), 298
__lt__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__lt__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 802
__lt__() (*rti.connextdds.HistoryKind method*), 301
__lt__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 319
__lt__() (*rti.connextdds.LivelinessKind method*), 343
__lt__() (*rti.connextdds.LocatorKind method*), 351
__lt__() (*rti.connextdds.LogCategory method*), 357
__lt__() (*rti.connextdds.LogLevel method*), 360
__lt__() (*rti.connextdds.OwnershipKind method*), 388
__lt__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__lt__() (*rti.connextdds.PrintFormat method*), 441
__lt__() (*rti.connextdds.PrintFormatKind method*), 442
__lt__() (*rti.connextdds.PublishModeKind method*), 493
__lt__() (*rti.connextdds.ReliabilityKind method*), 514

__lt__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 518
__lt__() (*rti.connextdds.SequenceNumber method*), 543
__lt__() (*rti.connextdds.ServiceKind method*), 547
__lt__() (*rti.connextdds.ServiceRequestId method*), 585
__lt__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 660
__lt__() (*rti.connextdds.Time method*), 664
__lt__() (*rti.connextdds.TopicQuerySelectionKind method*), 719
__lt__() (*rti.connextdds.TransportClassId method*), 728
__lt__() (*rti.connextdds.TransportMulticastKind method*), 734
__lt__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 750
__lt__() (*rti.connextdds.TypeKind method*), 755
__lt__() (*rti.connextdds.Verbosity method*), 775
__lt__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__members__ (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 39
__members__ (*rti.connextdds.CdrPaddingKind.CdrPaddingKind attribute*), 66
__members__ (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 98
__members__ (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 143
__members__ (*rti.connextdds.DestinationOrderKind.DestinationOrderKind attribute*), 150
__members__ (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 152
__members__ (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 187
__members__ (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 255
__members__ (*rti.connextdds.ExtensibilityKind.ExtensibilityKind attribute*), 279
__members__ (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 294
__members__ (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*), 799
__members__ (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat attribute*), 801
__members__ (*rti.connextdds.HistoryKind.HistoryKind attribute*), 300
__members__ (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 317
__members__ (*rti.connextdds.InstanceStateConsistencyKind attribute*), 325
__members__ (*rti.connextdds.LivelinessKind.LivelinessKind attribute*), 342
__members__ (*rti.connextdds.LocatorKind.LocatorKind attribute*), 350
__members__ (*rti.connextdds.LogCategory.LogCategory attribute*), 355
__members__ (*rti.connextdds.LogFacility attribute*), 358
__members__ (*rti.connextdds.LogLevel.LogLevel attribute*), 359
__members__ (*rti.connextdds.OwnershipKind.OwnershipKind attribute*), 387
__members__ (*rti.connextdds.PersistentJournalKind attribute*), 432
__members__ (*rti.connextdds.PersistentSynchronizationKind attribute*), 434
__members__ (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 437
__members__ (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*), 442
__members__ (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
__members__ (*rti.connextdds.PublishModeKind.PublishModeKind attribute*), 492
__members__ (*rti.connextdds.ReliabilityKind.ReliabilityKind attribute*), 514
__members__ (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute*), 517
__members__ (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
__members__ (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
__members__ (*rti.connextdds.SyslogVerbosity attribute*), 656
__members__ (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind attribute*), 659
__members__ (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute*), 718
__members__ (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
__members__ (*rti.connextdds.TransportMulticastKind.TransportMulticastKind attribute*), 733
__members__ (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 748
__members__ (*rti.connextdds.TypeKind.TypeKind attribute*), 754
__members__ (*rti.connextdds.Verbosity.Verbosity attribute*), 774
__members__ (*rti.connextdds.WireProtocolAutoKind.WirePro-*

*tocolAutoKind attribute*), 785
__module__ (*rti.connextdds.AcknowledgmentInfo attribute*), 38
__module__ (*rti.connextdds.AcknowledgmentKind attribute*), 41
__module__ (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 40
__module__ (*rti.connextdds.AckResponseData attribute*), 38
__module__ (*rti.connextdds.ACTEnumMember attribute*), 35
__module__ (*rti.connextdds.ActivityContextMask attribute*), 42
__module__ (*rti.connextdds.ACTMember attribute*), 36
__module__ (*rti.connextdds.ACTUnionMember attribute*), 37
__module__ (*rti.connextdds.AliasType attribute*), 44
__module__ (*rti.connextdds.AllocationSettings attribute*), 45
__module__ (*rti.connextdds.AlreadyClosedError attribute*), 45
__module__ (*rti.connextdds.AnyDataReader attribute*), 45
__module__ (*rti.connextdds.AnyDataReaderListener attribute*), 46
__module__ (*rti.connextdds.AnyDataReaderSeq attribute*), 47
__module__ (*rti.connextdds.AnyDataWriter attribute*), 49
__module__ (*rti.connextdds.AnyDataWriterListener attribute*), 49
__module__ (*rti.connextdds.AnyDataWriterSeq attribute*), 51
__module__ (*rti.connextdds.AnyTopic attribute*), 52
__module__ (*rti.connextdds.AnyTopicListener attribute*), 52
__module__ (*rti.connextdds.AnyTopicSeq attribute*), 53
__module__ (*rti.connextdds.ArrayType attribute*), 55
__module__ (*rti.connextdds.AsynchronousPublisher attribute*), 55
__module__ (*rti.connextdds.Availability attribute*), 56
__module__ (*rti.connextdds.Batch attribute*), 57
__module__ (*rti.connextdds.BoolSeq attribute*), 59
__module__ (*rti.connextdds.BoolType attribute*), 60
__module__ (*rti.connextdds.BuiltinProfiles attribute*), 60
__module__ (*rti.connextdds.BuiltinTopicKey attribute*), 63
__module__ (*rti.connextdds.BuiltinTopicReaderResourceLimits attribute*), 63
__module__ (*rti.connextdds.ByteVector attribute*), 65
__module__ (*rti.connextdds.CdrPaddingKind attribute*), 67
__module__ (*rti.connextdds.CdrPaddingKind.CdrPaddingKind attribute*), 66
__module__ (*rti.connextdds.ChannelSettings attribute*), 67
__module__ (*rti.connextdds.ChannelSettingsSeq attribute*), 69
__module__ (*rti.connextdds.CharSeq attribute*), 71
__module__ (*rti.connextdds.CharType attribute*), 72
__module__ (*rti.connextdds.CoherentAccess attribute*), 73
__module__ (*rti.connextdds.CoherentSet attribute*), 73
__module__ (*rti.connextdds.CoherentSetInfo attribute*), 74
__module__ (*rti.connextdds.CollectionType attribute*), 74
__module__ (*rti.connextdds.compliance.RpcMask attribute*), 792
__module__ (*rti.connextdds.compliance.XTypesMask attribute*), 796
__module__ (*rti.connextdds.CompressionIdMask attribute*), 76
__module__ (*rti.connextdds.CompressionSettings attribute*), 78
__module__ (*rti.connextdds.Condition attribute*), 78
__module__ (*rti.connextdds.ConditionSeq attribute*), 80
__module__ (*rti.connextdds.ContentFilterBase attribute*), 81
__module__ (*rti.connextdds.ContentFilteredTopic attribute*), 82
__module__ (*rti.connextdds.ContentFilteredTopicSeq attribute*), 84
__module__ (*rti.connextdds.ContentFilterProperty attribute*), 81
__module__ (*rti.connextdds.Cookie attribute*), 85
__module__ (*rti.connextdds.CookieSeq attribute*), 86
__module__ (*rti.connextdds.CookieVector attribute*), 88
__module__ (*rti.connextdds.Database attribute*), 147
__module__ (*rti.connextdds.DataReader attribute*), 92
__module__ (*rti.connextdds.DataReaderCacheStatus attribute*), 96
__module__ (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 99
__module__ (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 98
__module__ (*rti.connextdds.DataReaderListener attribute*), 100
__module__ (*rti.connextdds.DataReader.LoanedSample attribute*), 89
__module__ (*rti.connextdds.DataReader.LoanedSamples attribute*), 89
__module__ (*rti.connextdds.DataReaderProtocol attribute*), 101
__module__ (*rti.connextdds.DataReaderProtocolStatus attribute*), 101
__module__ (*rti.connextdds.DataReaderQos attribute*), 106
__module__ (*rti.connextdds.DataReaderResourceLimits attribute*), 110
__module__ (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings attribute*), 113
__module__ (*rti.connextdds.DataReader.Selector attribute*), 90
__module__ (*rti.connextdds.DataReaderSeq attribute*), 114
__module__ (*rti.connextdds.DataRepresentation attribute*), 116
__module__ (*rti.connextdds.DataState attribute*), 117
__module__ (*rti.connextdds.DataStateEx attribute*), 118
__module__ (*rti.connextdds.DataTag attribute*), 119
__module__ (*rti.connextdds.DataWriter attribute*), 122
__module__ (*rti.connextdds.DataWriterCacheStatus attribute*), 126
__module__ (*rti.connextdds.DataWriterListener attribute*), 127
__module__ (*rti.connextdds.DataWriterProtocol attribute*), 128
__module__ (*rti.connextdds.DataWriterProtocolStatus attribute*), 129
__module__ (*rti.connextdds.DataWriterQos attribute*), 134
__module__ (*rti.connextdds.DataWriterResourceLimits attribute*), 139
__module__ (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 144
__module__ (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 143
__module__ (*rti.connextdds.DataWriterSeq attribute*), 145
__module__ (*rti.connextdds.DataWriterShmemRefTransferModeSettings attribute*),

146
__module__ (*rti.connextdds.DataWriterTransferMode attribute*), 147
__module__ (*rti.connextdds.Deadline attribute*), 148
__module__ (*rti.connextdds.DestinationOrder attribute*), 149
__module__ (*rti.connextdds.DestinationOrderKind attribute*), 151
__module__ (*rti.connextdds.DestinationOrderKind.DestinationOrderKind attribute*), 150
__module__ (*rti.connextdds.DestinationOrderScopeKind attribute*), 153
__module__ (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 152
__module__ (*rti.connextdds.Discovery attribute*), 154
__module__ (*rti.connextdds.DiscoveryConfig attribute*), 155
__module__ (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 159
__module__ (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 162
__module__ (*rti.connextdds.DomainParticipant attribute*), 165
__module__ (*rti.connextdds.DomainParticipantConfigParams attribute*), 171
__module__ (*rti.connextdds.DomainParticipantFactoryQos attribute*), 172
__module__ (*rti.connextdds.DomainParticipantListener attribute*), 173
__module__ (*rti.connextdds.DomainParticipantProtocolStatus attribute*), 173
__module__ (*rti.connextdds.DomainParticipantQos attribute*), 176
__module__ (*rti.connextdds.DomainParticipantResourceLimits attribute*), 178
__module__ (*rti.connextdds.DomainParticipantSeq attribute*), 184
__module__ (*rti.connextdds.Durability attribute*), 186
__module__ (*rti.connextdds.DurabilityKind attribute*), 189
__module__ (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 188
__module__ (*rti.connextdds.DurabilityService attribute*), 189
__module__ (*rti.connextdds.Duration attribute*), 191
__module__ (*rti.connextdds.DynamicData attribute*), 230
__module__ (*rti.connextdds.DynamicData.ContentFilter attribute*), 192
__module__ (*rti.connextdds.DynamicData.ContentFilteredTopic attribute*), 193
__module__ (*rti.connextdds.DynamicData.ContentFilteredTopicSeq attribute*), 195
__module__ (*rti.connextdds.DynamicData.DataReader attribute*), 198
__module__ (*rti.connextdds.DynamicData.DataReaderListener attribute*), 202
__module__ (*rti.connextdds.DynamicData.DataReader.Selector attribute*), 196
__module__ (*rti.connextdds.DynamicData.DataReaderSeq attribute*), 204
__module__ (*rti.connextdds.DynamicData.DataWriter attribute*), 206
__module__ (*rti.connextdds.DynamicData.DataWriterListener attribute*), 213
__module__ (*rti.connextdds.DynamicData.DataWriterSeq attribute*), 215
__module__ (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 256
__module__ (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 255
__module__ (*rti.connextdds.DynamicData.FieldsIterator attribute*), 216
__module__ (*rti.connextdds.DynamicData.FieldsView attribute*), 217
__module__ (*rti.connextdds.DynamicData.IndexIterator attribute*), 217
__module__ (*rti.connextdds.DynamicDataInfo attribute*), 257
__module__ (*rti.connextdds.DynamicData.ItemsIterator attribute*), 218
__module__ (*rti.connextdds.DynamicData.ItemsView attribute*), 218
__module__ (*rti.connextdds.DynamicData.ITopicDescription attribute*), 217
__module__ (*rti.connextdds.DynamicData.LoanedSample attribute*), 218
__module__ (*rti.connextdds.DynamicData.LoanedSamples attribute*), 219
__module__ (*rti.connextdds.DynamicDataMemberInfo attribute*), 257
__module__ (*rti.connextdds.DynamicData.NoOpDataReaderListener attribute*), 219
__module__ (*rti.connextdds.DynamicData.NoOpDataWriterListener attribute*), 220
__module__ (*rti.connextdds.DynamicData.NoOpTopicListener attribute*), 222
__module__ (*rti.connextdds.DynamicDataProperty attribute*), 258
__module__ (*rti.connextdds.DynamicData.Sample attribute*), 222
__module__ (*rti.connextdds.DynamicDataSeq attribute*), 260
__module__ (*rti.connextdds.DynamicData.SharedSamples attribute*), 223
__module__ (*rti.connextdds.DynamicDataTimestampedSeq attribute*), 262
__module__ (*rti.connextdds.DynamicData.Topic attribute*), 223
__module__ (*rti.connextdds.DynamicData.TopicDescription attribute*), 224
__module__ (*rti.connextdds.DynamicData.TopicListener attribute*), 225
__module__ (*rti.connextdds.DynamicData.TopicSeq attribute*), 226

__module__ (*rti.connextdds.DynamicData.TopicTypeSupport attribute*), 227
__module__ (*rti.connextdds.DynamicDataTypeSerializationProperty attribute*), 264
__module__ (*rti.connextdds.DynamicData.ValidLoanedSamples attribute*), 228
__module__ (*rti.connextdds.DynamicData.WriterContentFilter attribute*), 228
__module__ (*rti.connextdds.DynamicData.WriterContentFilterHelper attribute*), 229
__module__ (*rti.connextdds.DynamicType attribute*), 264
__module__ (*rti.connextdds.DynamicTypePrintFormatProperty attribute*), 265
__module__ (*rti.connextdds.EndpointGroup attribute*), 266
__module__ (*rti.connextdds.EndpointGroupSeq attribute*), 267
__module__ (*rti.connextdds.EndpointGroupVector attribute*), 269
__module__ (*rti.connextdds.Entity attribute*), 269
__module__ (*rti.connextdds.EntityFactory attribute*), 270
__module__ (*rti.connextdds.EntityLock attribute*), 270
__module__ (*rti.connextdds.EntityName attribute*), 271
__module__ (*rti.connextdds.EntitySeq attribute*), 272
__module__ (*rti.connextdds.EnumMember attribute*), 273
__module__ (*rti.connextdds.EnumMemberSeq attribute*), 275
__module__ (*rti.connextdds.EnumType attribute*), 276
__module__ (*rti.connextdds.Error attribute*), 277
__module__ (*rti.connextdds.Event attribute*), 277
__module__ (*rti.connextdds.EventCount32 attribute*), 278
__module__ (*rti.connextdds.EventCount64 attribute*), 278
__module__ (*rti.connextdds.Exception attribute*), 278
__module__ (*rti.connextdds.ExpressionProperty attribute*), 279
__module__ (*rti.connextdds.ExtensibilityKind attribute*), 281
__module__ (*rti.connextdds.ExtensibilityKind.ExtensibilityKind attribute*), 280
__module__ (*rti.connextdds.Filter attribute*), 281
__module__ (*rti.connextdds.FilterSampleInfo attribute*), 282
__module__ (*rti.connextdds.Float32Seq attribute*), 286
__module__ (*rti.connextdds.Float32Type attribute*), 287
__module__ (*rti.connextdds.Float64Seq attribute*), 288
__module__ (*rti.connextdds.Float64Type attribute*), 290
__module__ (*rti.connextdds.Float128Seq attribute*), 283
__module__ (*rti.connextdds.Float128Type attribute*), 285
__module__ (*rti.connextdds.FlowController attribute*), 290
__module__ (*rti.connextdds.FlowControllerProperty attribute*), 291
__module__ (*rti.connextdds.FlowControllerSchedulingPolicy attribute*), 295
__module__ (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 294
__module__ (*rti.connextdds.FlowControllerTokenBucketProperty attribute*), 295
__module__ (*rti.connextdds.GenerationCount attribute*), 296
__module__ (*rti.connextdds.GroupData attribute*), 297
__module__ (*rti.connextdds.GuardCondition attribute*), 297
__module__ (*rti.connextdds.Guid attribute*), 298
__module__ (*rti.connextdds.heap_monitoring.HeapMonitoringParams attribute*), 798
__module__ (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 800
__module__ (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*), 799
__module__ (*rti.connextdds.heap_monitoring.SnapshotOutputFormat attribute*), 802
__module__ (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat attribute*), 801
__module__ (*rti.connextdds.History attribute*), 299
__module__ (*rti.connextdds.HistoryKind attribute*), 301
__module__ (*rti.connextdds.HistoryKind.HistoryKind attribute*), 300
__module__ (*rti.connextdds.IAnyDataReader attribute*), 301
__module__ (*rti.connextdds.IAnyDataReaderSeq attribute*), 303
__module__ (*rti.connextdds.IAnyDataWriter attribute*), 304
__module__ (*rti.connextdds.IAnyDataWriterSeq attribute*), 306
__module__ (*rti.connextdds.IAnyTopic attribute*), 307
__module__ (*rti.connextdds.IAnyTopicSeq attribute*), 309
__module__ (*rti.connextdds.ICondition attribute*), 310
__module__ (*rti.connextdds.IConditionSeq attribute*), 311
__module__ (*rti.connextdds.IDataReader attribute*), 313
__module__ (*rti.connextdds.IEntity attribute*), 313
__module__ (*rti.connextdds.IEntitySeq attribute*), 315
__module__ (*rti.connextdds.IgnoredEntityReplacementKind attribute*), 319
__module__ (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 318
__module__ (*rti.connextdds.IllegalOperationError attribute*), 319
__module__ (*rti.connextdds.ImmutablePolicyError attribute*), 319
__module__ (*rti.connextdds.InconsistentPolicyError attribute*), 319
__module__ (*rti.connextdds.InconsistentTopicStatus attribute*), 319
__module__ (*rti.connextdds.InstanceHandle attribute*), 320
__module__ (*rti.connextdds.InstanceHandleSeq attribute*), 321
__module__ (*rti.connextdds.InstanceState attribute*), 324
__module__ (*rti.connextdds.InstanceStateConsistencyKind attribute*), 326
__module__ (*rti.connextdds.Int8Seq attribute*), 336
__module__ (*rti.connextdds.Int8Type attribute*), 337
__module__ (*rti.connextdds.Int16Seq attribute*), 327
__module__ (*rti.connextdds.Int16Type attribute*), 328
__module__ (*rti.connextdds.Int32Seq attribute*), 330
__module__ (*rti.connextdds.Int32Type attribute*), 331
__module__ (*rti.connextdds.Int32Vector attribute*), 332
__module__ (*rti.connextdds.Int64Seq attribute*), 333

__module__ (*rti.connextdds.Int64Type attribute*), 335
__module__ (*rti.connextdds.InvalidArgumentError attribute*), 337
__module__ (*rti.connextdds.InvalidDowncastError attribute*), 338
__module__ (*rti.connextdds.InvalidLocalAdvanceNoticeStatus attribute*), 338
__module__ (*rti.connextdds.IReadCondition attribute*), 316
__module__ (*rti.connextdds.ItemsView attribute*), 338
__module__ (*rti.connextdds.ITopicDescription attribute*), 317
__module__ (*rti.connextdds.KeysView attribute*), 338
__module__ (*rti.connextdds.LatencyBudget attribute*), 339
__module__ (*rti.connextdds.Lifespan attribute*), 339
__module__ (*rti.connextdds.Liveliness attribute*), 340
__module__ (*rti.connextdds.LivelinessChangedStatus attribute*), 341
__module__ (*rti.connextdds.LivelinessKind attribute*), 343
__module__ (*rti.connextdds.LivelinessKind.LivelinessKind attribute*), 342
__module__ (*rti.connextdds.LivelinessLostStatus attribute*), 343
__module__ (*rti.connextdds.LoanedDynamicData attribute*), 344
__module__ (*rti.connextdds.Locator attribute*), 344
__module__ (*rti.connextdds.LocatorFilter attribute*), 345
__module__ (*rti.connextdds.LocatorFilterElement attribute*), 346
__module__ (*rti.connextdds.LocatorFilterElementSeq attribute*), 347
__module__ (*rti.connextdds.LocatorKind attribute*), 351
__module__ (*rti.connextdds.LocatorKind.LocatorKind attribute*), 350
__module__ (*rti.connextdds.LocatorSeq attribute*), 352
__module__ (*rti.connextdds.LocatorVector attribute*), 354
__module__ (*rti.connextdds.LogCategory attribute*), 357
__module__ (*rti.connextdds.LogCategory.LogCategory attribute*), 355
__module__ (*rti.connextdds.LogFacility attribute*), 358
__module__ (*rti.connextdds.Logger attribute*), 361
__module__ (*rti.connextdds.LogLevel attribute*), 360
__module__ (*rti.connextdds.LogLevel.LogLevel attribute*), 359
__module__ (*rti.connextdds.LogMessage attribute*), 360
__module__ (*rti.connextdds.LongDouble attribute*), 363
__module__ (*rti.connextdds.Member attribute*), 364
__module__ (*rti.connextdds.MemberSeq attribute*), 365
__module__ (*rti.connextdds.Monitoring attribute*), 367
__module__ (*rti.connextdds.MonitoringDedicatedParticipantSettings attribute*), 367
__module__ (*rti.connextdds.MonitoringDistributionSettings attribute*), 368
__module__ (*rti.connextdds.MonitoringEventDistributionSettings attribute*), 369
__module__ (*rti.connextdds.MonitoringLoggingDistributionSettings attribute*), 369
__module__ (*rti.connextdds.MonitoringLoggingForwardingSettings attribute*), 370
__module__ (*rti.connextdds.MonitoringMetricSelection attribute*), 371
__module__ (*rti.connextdds.MonitoringMetricSelectionSeq attribute*), 372
__module__ (*rti.connextdds.MonitoringPeriodicDistributionSettings attribute*), 374
__module__ (*rti.connextdds.MonitoringTelemetryData attribute*), 374
__module__ (*rti.connextdds.MulticastMapping attribute*), 376
__module__ (*rti.connextdds.MulticastMappingSeq attribute*), 377
__module__ (*rti.connextdds.MultiChannel attribute*), 375
__module__ (*rti.connextdds.NoOpAnyDataReaderListener attribute*), 378
__module__ (*rti.connextdds.NoOpAnyDataWriterListener attribute*), 379
__module__ (*rti.connextdds.NoOpAnyTopicListener attribute*), 381
__module__ (*rti.connextdds.NoOpDataReaderListener attribute*), 381
__module__ (*rti.connextdds.NoOpDataWriterListener attribute*), 382
__module__ (*rti.connextdds.NoOpDomainParticipantListener attribute*), 383
__module__ (*rti.connextdds.NoOpPublisherListener attribute*), 383
__module__ (*rti.connextdds.NoOpSubscriberListener attribute*), 383
__module__ (*rti.connextdds.NoOpTopicListener attribute*), 384
__module__ (*rti.connextdds.NotAllowedBySecurityError attribute*), 384
__module__ (*rti.connextdds.NotEnabledError attribute*), 384
__module__ (*rti.connextdds.NullReferenceError attribute*), 384
__module__ (*rti.connextdds.OctetType attribute*), 384
__module__ (*rti.connextdds.OfferedDeadlineMissedStatus attribute*), 385
__module__ (*rti.connextdds.OfferedIncompatibleQosStatus attribute*), 385
__module__ (*rti.connextdds.OutOfResourcesError attribute*), 385
__module__ (*rti.connextdds.Ownership attribute*), 386
__module__ (*rti.connextdds.OwnershipKind attribute*), 388
__module__ (*rti.connextdds.OwnershipKind.OwnershipKind attribute*), 387
__module__ (*rti.connextdds.OwnershipStrength attribute*), 388
__module__ (*rti.connextdds.ParticipantBuiltinTopicData attribute*), 425
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilter attribute*), 389
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic attribute*), 389
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq attribute*), 391
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReader attribute*),

395
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener attribute*), 399
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector attribute*), 393
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq attribute*), 401
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter attribute*), 403
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener attribute*), 410
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq attribute*), 413
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription attribute*), 414
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample attribute*), 414
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples attribute*), 415
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener attribute*), 415
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener attribute*), 416
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener attribute*), 418
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.Sample attribute*), 419
__module__ (*rti.connextdds.ParticipantBuiltinTopicDataSeq attribute*), 427
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.SharedSamples attribute*), 419
__module__ (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq attribute*), 430
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.Topic attribute*), 420
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicDescription attribute*), 420
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicListener attribute*), 421
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq attribute*), 422
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples attribute*), 423
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter attribute*), 424
__module__ (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilterHelper attribute*), 425
__module__ (*rti.connextdds.Partition attribute*), 432
__module__ (*rti.connextdds.PersistentJournalKind attribute*), 433
__module__ (*rti.connextdds.PersistentStorageSettings attribute*), 433
__module__ (*rti.connextdds.PersistentSynchronizationKind attribute*), 435
__module__ (*rti.connextdds.PreconditionNotMetError attribute*), 435
__module__ (*rti.connextdds.Presentation attribute*), 435
__module__ (*rti.connextdds.PresentationAccessScopeKind attribute*), 438
__module__ (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 437
__module__ (*rti.connextdds.PrintFormat attribute*), 441
__module__ (*rti.connextdds.PrintFormatKind attribute*), 442
__module__ (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*), 442
__module__ (*rti.connextdds.PrintFormat.PrintFormat attribute*), 440
__module__ (*rti.connextdds.PrintFormatProperty attribute*), 443
__module__ (*rti.connextdds.ProductVersion attribute*), 443
__module__ (*rti.connextdds.Property attribute*), 445
__module__ (*rti.connextdds.ProtocolVersion attribute*), 446
__module__ (*rti.connextdds.PublicationBuiltinTopicData attribute*), 483
__module__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilter attribute*), 446
__module__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic attribute*), 447
__module__ (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq attribute*), 449
__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataReader attribute*), 453
__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener attribute*), 457
__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector attribute*), 450
__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq attribute*), 459

__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriter attribute*), 461
__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener attribute*), 468
__module__ (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq attribute*), 470
__module__ (*rti.connextdds.PublicationBuiltinTopicData.ITopicDescription attribute*), 472
__module__ (*rti.connextdds.PublicationBuiltinTopicData.LoanedSample attribute*), 472
__module__ (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples attribute*), 473
__module__ (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener attribute*), 473
__module__ (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener attribute*), 474
__module__ (*rti.connextdds.PublicationBuiltinTopicData.NoOpTopicListener attribute*), 476
__module__ (*rti.connextdds.PublicationBuiltinTopicData.Sample attribute*), 476
__module__ (*rti.connextdds.PublicationBuiltinTopicDataSeq attribute*), 486
__module__ (*rti.connextdds.PublicationBuiltinTopicData.SharedSamples attribute*), 477
__module__ (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq attribute*), 489
__module__ (*rti.connextdds.PublicationBuiltinTopicData.Topic attribute*), 477
__module__ (*rti.connextdds.PublicationBuiltinTopicData.TopicDescription attribute*), 478
__module__ (*rti.connextdds.PublicationBuiltinTopicData.TopicListener attribute*), 479
__module__ (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq attribute*), 480
__module__ (*rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples attribute*), 481
__module__ (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter attribute*), 482
__module__ (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper attribute*), 482
__module__ (*rti.connextdds.PublicationMatchedStatus attribute*), 490
__module__ (*rti.connextdds.Publisher attribute*), 494
__module__ (*rti.connextdds.PublisherListener attribute*), 495
__module__ (*rti.connextdds.PublisherQos attribute*), 496
__module__ (*rti.connextdds.PublisherSeq attribute*), 498
__module__ (*rti.connextdds.PublishMode attribute*), 491
__module__ (*rti.connextdds.PublishModeKind attribute*), 493
__module__ (*rti.connextdds.PublishModeKind.PublishModeKind attribute*), 492
__module__ (*rti.connextdds.QosPolicyCount attribute*), 500
__module__ (*rti.connextdds.QosPolicyCountSeq attribute*), 501
__module__ (*rti.connextdds.QosPrintFormat attribute*), 502
__module__ (*rti.connextdds.QosProvider attribute*), 503
__module__ (*rti.connextdds.QosProviderParams attribute*), 506
__module__ (*rti.connextdds.Query attribute*), 507
__module__ (*rti.connextdds.QueryCondition attribute*), 508
__module__ (*rti.connextdds.Rank attribute*), 509
__module__ (*rti.connextdds.ReadCondition attribute*), 510
__module__ (*rti.connextdds.ReaderDataLifecycle attribute*), 510
__module__ (*rti.connextdds.ReceiverPool attribute*), 512
__module__ (*rti.connextdds.Reliability attribute*), 512
__module__ (*rti.connextdds.ReliabilityKind attribute*), 515
__module__ (*rti.connextdds.ReliabilityKind.ReliabilityKind attribute*), 514
__module__ (*rti.connextdds.ReliableReaderActivityChangedStatus attribute*), 515
__module__ (*rti.connextdds.ReliableWriterCacheChangedStatus attribute*), 515
__module__ (*rti.connextdds.RemoteParticipantPurgeKind attribute*), 518
__module__ (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute*), 517
__module__ (*rti.connextdds.RequestedDeadlineMissedStatus attribute*), 518
__module__ (*rti.connextdds.RequestedIncompatibleQosStatus attribute*), 519
__module__ (*rti.connextdds.ResourceLimits attribute*), 519
__module__ (*rti.connextdds.RtpsReliableReaderProtocol attribute*), 520
__module__ (*rti.connextdds.RtpsReliableWriterProtocol attribute*), 521
__module__ (*rti.connextdds.RtpsReservedPortKindMask attribute*), 525
__module__ (*rti.connextdds.RtpsWellKnownPorts attribute*), 527
__module__ (*rti.connextdds.SampleFlag attribute*), 529
__module__ (*rti.connextdds.SampleIdentity attribute*), 531
__module__ (*rti.connextdds.SampleInfo attribute*), 531
__module__ (*rti.connextdds.SampleLostState attribute*), 535
__module__ (*rti.connextdds.SampleLostStatus attribute*), 536
__module__ (*rti.connextdds.SampleRejectedState attribute*), 538

__module__ (*rti.connextdds.SampleRejectedStatus attribute*), 539
__module__ (*rti.connextdds.SampleState attribute*), 541
__module__ (*rti.connextdds.SequenceNumber attribute*), 543
__module__ (*rti.connextdds.SequenceType attribute*), 544
__module__ (*rti.connextdds.Service attribute*), 544
__module__ (*rti.connextdds.ServiceKind attribute*), 547
__module__ (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
__module__ (*rti.connextdds.ServiceRequest attribute*), 582
__module__ (*rti.connextdds.ServiceRequestAcceptedStatus attribute*), 583
__module__ (*rti.connextdds.ServiceRequest.ContentFilter attribute*), 547
__module__ (*rti.connextdds.ServiceRequest.ContentFilteredTopic attribute*), 548
__module__ (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq attribute*), 550
__module__ (*rti.connextdds.ServiceRequest.DataReader attribute*), 554
__module__ (*rti.connextdds.ServiceRequest.DataReaderListener attribute*), 557
__module__ (*rti.connextdds.ServiceRequest.DataReader.Selector attribute*), 551
__module__ (*rti.connextdds.ServiceRequest.DataReaderSeq attribute*), 559
__module__ (*rti.connextdds.ServiceRequest.DataWriter attribute*), 562
__module__ (*rti.connextdds.ServiceRequest.DataWriterListener attribute*), 568
__module__ (*rti.connextdds.ServiceRequest.DataWriterSeq attribute*), 570
__module__ (*rti.connextdds.ServiceRequestId attribute*), 585
__module__ (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
__module__ (*rti.connextdds.ServiceRequest.ITopicDescription attribute*), 572
__module__ (*rti.connextdds.ServiceRequest.LoanedSample attribute*), 572
__module__ (*rti.connextdds.ServiceRequest.LoanedSamples attribute*), 573
__module__ (*rti.connextdds.ServiceRequest.NoOpDataReaderListener attribute*), 573
__module__ (*rti.connextdds.ServiceRequest.NoOpDataWriterListener attribute*), 574
__module__ (*rti.connextdds.ServiceRequest.NoOpTopicListener attribute*), 575
__module__ (*rti.connextdds.ServiceRequest.Sample attribute*), 576
__module__ (*rti.connextdds.ServiceRequestSeq attribute*), 587
__module__ (*rti.connextdds.ServiceRequest.SharedSamples attribute*), 576
__module__ (*rti.connextdds.ServiceRequestTimestampedSeq attribute*), 589
__module__ (*rti.connextdds.ServiceRequest.Topic attribute*), 577
__module__ (*rti.connextdds.ServiceRequest.TopicDescription attribute*), 578
__module__ (*rti.connextdds.ServiceRequest.TopicListener attribute*), 578
__module__ (*rti.connextdds.ServiceRequest.TopicSeq attribute*), 579
__module__ (*rti.connextdds.ServiceRequest.ValidLoanedSamples attribute*), 581
__module__ (*rti.connextdds.ServiceRequest.WriterContentFilter attribute*), 581
__module__ (*rti.connextdds.ServiceRequest.WriterContentFilterHelper attribute*), 582
__module__ (*rti.connextdds.StatusCondition attribute*), 591
__module__ (*rti.connextdds.StatusMask attribute*), 593
__module__ (*rti.connextdds.StreamKind attribute*), 596
__module__ (*rti.connextdds.StringMap attribute*), 598
__module__ (*rti.connextdds.StringPairSeq attribute*), 599
__module__ (*rti.connextdds.StringSeq attribute*), 601
__module__ (*rti.connextdds.StringType attribute*), 602
__module__ (*rti.connextdds.StructType attribute*), 603
__module__ (*rti.connextdds.Subscriber attribute*), 604
__module__ (*rti.connextdds.SubscriberListener attribute*), 606
__module__ (*rti.connextdds.SubscriberQos attribute*), 607
__module__ (*rti.connextdds.SubscriberSeq attribute*), 609
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData attribute*), 647
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter attribute*), 610
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic attribute*), 611
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq attribute*), 612
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader attribute*), 616
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener attribute*), 620
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector attribute*), 614
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq attribute*), 622
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter attribute*), 625
__module__ (*rti.connextdds.SubscriptionBuiltinTopic-*

*Data.DataWriterListener attribute*), 632
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq attribute*), 634
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription attribute*), 635
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample attribute*), 636
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples attribute*), 637
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener attribute*), 637
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener attribute*), 638
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpTopicListener attribute*), 640
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.Sample attribute*), 640
__module__ (*rti.connextdds.SubscriptionBuiltinTopicDataSeq attribute*), 650
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples attribute*), 641
__module__ (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq attribute*), 653
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.Topic attribute*), 641
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription attribute*), 642
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicListener attribute*), 642
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq attribute*), 644
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples attribute*), 645
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter attribute*), 645
__module__ (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper attribute*), 646
__module__ (*rti.connextdds.SubscriptionMatchedStatus attribute*), 654
__module__ (*rti.connextdds.SuspendedPublication attribute*), 655
__module__ (*rti.connextdds.SyslogVerbosity attribute*), 656
__module__ (*rti.connextdds.SystemResourceLimits attribute*), 656
__module__ (*rti.connextdds.ThreadContext attribute*), 657
__module__ (*rti.connextdds.ThreadSettings attribute*), 657
__module__ (*rti.connextdds.ThreadSettingsCpuRotationKind attribute*), 660
__module__ (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind attribute*), 659
__module__ (*rti.connextdds.ThreadSettingsKindMask attribute*), 661
__module__ (*rti.connextdds.Time attribute*), 664
__module__ (*rti.connextdds.TimeBasedFilter attribute*), 665
__module__ (*rti.connextdds.TimeoutError attribute*), 665
__module__ (*rti.connextdds.Topic attribute*), 666
__module__ (*rti.connextdds.TopicBuiltinTopicData attribute*), 703
__module__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilter attribute*), 667
__module__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic attribute*), 668
__module__ (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq attribute*), 670
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataReader attribute*), 673
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener attribute*), 677
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector attribute*), 671
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq attribute*), 679
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataWriter attribute*), 681
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener attribute*), 688
__module__ (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq attribute*), 690
__module__ (*rti.connextdds.TopicBuiltinTopicData.ITopicDescription attribute*), 692
__module__ (*rti.connextdds.TopicBuiltinTopicData.LoanedSample attribute*), 692
__module__ (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples attribute*), 693
__module__ (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener attribute*),

693
__module__ (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener attribute*), 694
__module__ (*rti.connextdds.TopicBuiltinTopicData.NoOpTopicListener attribute*), 696
__module__ (*rti.connextdds.TopicBuiltinTopicData.Sample attribute*), 696
__module__ (*rti.connextdds.TopicBuiltinTopicDataSeq attribute*), 705
__module__ (*rti.connextdds.TopicBuiltinTopicData.SharedSamples attribute*), 697
__module__ (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq attribute*), 708
__module__ (*rti.connextdds.TopicBuiltinTopicData.Topic attribute*), 697
__module__ (*rti.connextdds.TopicBuiltinTopicData.TopicDescription attribute*), 698
__module__ (*rti.connextdds.TopicBuiltinTopicData.TopicListener attribute*), 699
__module__ (*rti.connextdds.TopicBuiltinTopicData.TopicSeq attribute*), 700
__module__ (*rti.connextdds.TopicBuiltinTopicData.ValidLoanedSamples attribute*), 701
__module__ (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter attribute*), 701
__module__ (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper attribute*), 702
__module__ (*rti.connextdds.TopicData attribute*), 709
__module__ (*rti.connextdds.TopicDescription attribute*), 710
__module__ (*rti.connextdds.TopicListener attribute*), 710
__module__ (*rti.connextdds.TopicQos attribute*), 712
__module__ (*rti.connextdds.TopicQuery attribute*), 715
__module__ (*rti.connextdds.TopicQueryData attribute*), 715
__module__ (*rti.connextdds.TopicQueryDispatch attribute*), 716
__module__ (*rti.connextdds.TopicQuerySelection attribute*), 717
__module__ (*rti.connextdds.TopicQuerySelectionKind attribute*), 719
__module__ (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute*), 718
__module__ (*rti.connextdds.TopicSeq attribute*), 720
__module__ (*rti.connextdds.TransportBuiltin attribute*), 721
__module__ (*rti.connextdds.TransportBuiltinMask attribute*), 723
__module__ (*rti.connextdds.TransportClassId attribute*), 728
__module__ (*rti.connextdds.TransportClassId.TransportClassId attribute*), 727
__module__ (*rti.connextdds.TransportInfo attribute*), 728
__module__ (*rti.connextdds.TransportInfoSeq attribute*), 730
__module__ (*rti.connextdds.TransportInfoVector attribute*), 731
__module__ (*rti.connextdds.TransportMulticast attribute*), 732
__module__ (*rti.connextdds.TransportMulticastKind attribute*), 734
__module__ (*rti.connextdds.TransportMulticastKind.TransportMulticastKind attribute*), 733
__module__ (*rti.connextdds.TransportMulticastMapping attribute*), 735
__module__ (*rti.connextdds.TransportMulticastMappingFunction attribute*), 735
__module__ (*rti.connextdds.TransportMulticastSeq attribute*), 737
__module__ (*rti.connextdds.TransportMulticastSettings attribute*), 738
__module__ (*rti.connextdds.TransportMulticastSettingsSeq attribute*), 740
__module__ (*rti.connextdds.TransportPriority attribute*), 741
__module__ (*rti.connextdds.TransportSelection attribute*), 742
__module__ (*rti.connextdds.TransportUnicast attribute*), 742
__module__ (*rti.connextdds.TransportUnicastSettings attribute*), 743
__module__ (*rti.connextdds.TransportUnicastSettingsSeq attribute*), 744
__module__ (*rti.connextdds.TriggeredConditions attribute*), 746
__module__ (*rti.connextdds.TriggeredConditionsIterator attribute*), 746
__module__ (*rti.connextdds.TypeConsistencyEnforcement attribute*), 747
__module__ (*rti.connextdds.TypeConsistencyEnforcementKind attribute*), 750
__module__ (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 749
__module__ (*rti.connextdds.TypeKind attribute*), 755
__module__ (*rti.connextdds.TypeKind.TypeKind attribute*), 754
__module__ (*rti.connextdds.TypeSupport attribute*), 756
__module__ (*rti.connextdds.Uint8Seq attribute*), 765
__module__ (*rti.connextdds.Uint8Type attribute*), 767
__module__ (*rti.connextdds.Uint16Seq attribute*), 757
__module__ (*rti.connextdds.Uint16Type attribute*), 759
__module__ (*rti.connextdds.Uint32Seq attribute*), 760
__module__ (*rti.connextdds.Uint32Type attribute*), 762
__module__ (*rti.connextdds.Uint64Seq attribute*), 763
__module__ (*rti.connextdds.Uint64Type attribute*), 764
__module__ (*rti.connextdds.UnidimensionalCollectionType attribute*), 767
__module__ (*rti.connextdds.UnionMember attribute*), 768
__module__ (*rti.connextdds.UnionMemberSeq attribute*), 770
__module__ (*rti.connextdds.UnionType attribute*), 771
__module__ (*rti.connextdds.UnsupportedError attribute*), 772
__module__ (*rti.connextdds.UserData attribute*), 772
__module__ (*rti.connextdds.UserDataSample attribute*), 772

__module__ (*rti.connextdds.ValuesView attribute*), 773
__module__ (*rti.connextdds.VendorId attribute*), 773
__module__ (*rti.connextdds.Verbosity attribute*), 775
__module__ (*rti.connextdds.Verbosity.Verbosity attribute*), 774
__module__ (*rti.connextdds.ViewState attribute*), 777
__module__ (*rti.connextdds.WaitSet attribute*), 779
__module__ (*rti.connextdds.WaitSetProperty attribute*), 781
__module__ (*rti.connextdds.WcharSeq attribute*), 782
__module__ (*rti.connextdds.WcharType attribute*), 783
__module__ (*rti.connextdds.WireProtocol attribute*), 784
__module__ (*rti.connextdds.WireProtocolAutoKind attribute*), 786
__module__ (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind attribute*), 785
__module__ (*rti.connextdds.WriteParams attribute*), 787
__module__ (*rti.connextdds.WriterDataLifecycle attribute*), 788
__module__ (*rti.connextdds.WstringSeq attribute*), 789
__module__ (*rti.connextdds.WStringType attribute*), 778
__mul__() (*rti.connextdds.AnyDataReaderSeq method*), 47
__mul__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__mul__() (*rti.connextdds.AnyTopicSeq method*), 53
__mul__() (*rti.connextdds.BoolSeq method*), 59
__mul__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__mul__() (*rti.connextdds.CharSeq method*), 71
__mul__() (*rti.connextdds.ConditionSeq method*), 80
__mul__() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
__mul__() (*rti.connextdds.CookieSeq method*), 86
__mul__() (*rti.connextdds.DataReaderSeq method*), 114
__mul__() (*rti.connextdds.DataWriterSeq method*), 145
__mul__() (*rti.connextdds.DomainParticipantSeq method*), 184
__mul__() (*rti.connextdds.Duration method*), 191
__mul__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
__mul__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__mul__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__mul__() (*rti.connextdds.DynamicDataSeq method*), 260
__mul__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__mul__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__mul__() (*rti.connextdds.EndpointGroupSeq method*), 267
__mul__() (*rti.connextdds.EntitySeq method*), 272
__mul__() (*rti.connextdds.EnumMemberSeq method*), 275
__mul__() (*rti.connextdds.Float32Seq method*), 286
__mul__() (*rti.connextdds.Float64Seq method*), 289
__mul__() (*rti.connextdds.Float128Seq method*), 283
__mul__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__mul__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__mul__() (*rti.connextdds.IAnyTopicSeq method*), 309
__mul__() (*rti.connextdds.IConditionSeq method*), 311
__mul__() (*rti.connextdds.IEntitySeq method*), 315
__mul__() (*rti.connextdds.InstanceHandleSeq method*), 321
__mul__() (*rti.connextdds.Int8Seq method*), 336
__mul__() (*rti.connextdds.Int16Seq method*), 327
__mul__() (*rti.connextdds.Int32Seq method*), 330
__mul__() (*rti.connextdds.Int64Seq method*), 333
__mul__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__mul__() (*rti.connextdds.LocatorSeq method*), 352
__mul__() (*rti.connextdds.MemberSeq method*), 365
__mul__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__mul__() (*rti.connextdds.MulticastMappingSeq method*), 377
__mul__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__mul__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__mul__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
__mul__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__mul__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
__mul__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__mul__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__mul__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
__mul__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
__mul__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__mul__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
__mul__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__mul__() (*rti.connextdds.PublisherSeq method*), 498
__mul__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__mul__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__mul__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__mul__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__mul__() (*rti.connextdds.ServiceRequestSeq method*), 587
__mul__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
__mul__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579

__mul__() (*rti.connextdds.StringPairSeq method*), 599
__mul__() (*rti.connextdds.StringSeq method*), 601
__mul__() (*rti.connextdds.SubscriberSeq method*), 609
__mul__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
__mul__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__mul__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__mul__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__mul__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
__mul__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
__mul__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
__mul__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__mul__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 690
__mul__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__mul__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
__mul__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__mul__() (*rti.connextdds.TopicSeq method*), 720
__mul__() (*rti.connextdds.TransportInfoSeq method*), 730
__mul__() (*rti.connextdds.TransportMulticastSeq method*), 737
__mul__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
__mul__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__mul__() (*rti.connextdds.Uint8Seq method*), 765
__mul__() (*rti.connextdds.Uint16Seq method*), 757
__mul__() (*rti.connextdds.Uint32Seq method*), 760
__mul__() (*rti.connextdds.Uint64Seq method*), 763
__mul__() (*rti.connextdds.UnionMemberSeq method*), 770
__mul__() (*rti.connextdds.WcharSeq method*), 782
__mul__() (*rti.connextdds.WstringSeq method*), 789
__ne__() (*rti.connextdds.AcknowledgmentKind method*), 41
__ne__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 40
__ne__() (*rti.connextdds.ACTEnumMember method*), 35
__ne__() (*rti.connextdds.ActivityContextMask method*), 42
__ne__() (*rti.connextdds.ACTMember method*), 36
__ne__() (*rti.connextdds.ACTUnionMember method*), 37
__ne__() (*rti.connextdds.AliasType method*), 44
__ne__() (*rti.connextdds.AllocationSettings method*), 45
__ne__() (*rti.connextdds.AnyDataReaderSeq method*), 48
__ne__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__ne__() (*rti.connextdds.AnyTopicSeq method*), 54
__ne__() (*rti.connextdds.ArrayType method*), 55
__ne__() (*rti.connextdds.AsynchronousPublisher method*), 55
__ne__() (*rti.connextdds.Availability method*), 56
__ne__() (*rti.connextdds.Batch method*), 57
__ne__() (*rti.connextdds.BoolSeq method*), 59
__ne__() (*rti.connextdds.BoolType method*), 60
__ne__() (*rti.connextdds.BuiltinTopicKey method*), 63
__ne__() (*rti.connextdds.BuiltinTopicReaderResourceLimits method*), 63
__ne__() (*rti.connextdds.ByteVector method*), 65
__ne__() (*rti.connextdds.CdrPaddingKind method*), 67
__ne__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__ne__() (*rti.connextdds.ChannelSettings method*), 67
__ne__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__ne__() (*rti.connextdds.CharSeq method*), 71
__ne__() (*rti.connextdds.CharType method*), 72
__ne__() (*rti.connextdds.CoherentSetInfo method*), 74
__ne__() (*rti.connextdds.CollectionType method*), 74
__ne__() (*rti.connextdds.compliance.RpcMask method*), 792
__ne__() (*rti.connextdds.compliance.XTypesMask method*), 796
__ne__() (*rti.connextdds.CompressionIdMask method*), 76
__ne__() (*rti.connextdds.CompressionSettings method*), 78
__ne__() (*rti.connextdds.ConditionSeq method*), 80
__ne__() (*rti.connextdds.ContentFilteredTopic method*), 82
__ne__() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
__ne__() (*rti.connextdds.ContentFilterProperty method*), 81
__ne__() (*rti.connextdds.Cookie method*), 85
__ne__() (*rti.connextdds.CookieSeq method*), 86
__ne__() (*rti.connextdds.CookieVector method*), 88
__ne__() (*rti.connextdds.Database method*), 147
__ne__() (*rti.connextdds.DataReader method*), 92
__ne__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__ne__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__ne__() (*rti.connextdds.DataReaderProtocol method*), 101
__ne__() (*rti.connextdds.DataReaderQos method*), 106
__ne__() (*rti.connextdds.DataReaderResourceLimits method*), 110
__ne__() (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings method*), 113
__ne__() (*rti.connextdds.DataReaderSeq method*), 114
__ne__() (*rti.connextdds.DataRepresentation method*), 116
__ne__() (*rti.connextdds.DataState method*), 117
__ne__() (*rti.connextdds.DataStateEx method*), 118
__ne__() (*rti.connextdds.DataTag method*), 120
__ne__() (*rti.connextdds.DataWriter method*), 122
__ne__() (*rti.connextdds.DataWriterProtocol method*), 128
__ne__() (*rti.connextdds.DataWriterQos method*), 134

__ne__() (*rti.connextdds.DataWriterResourceLimits method*), 139
__ne__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 144
__ne__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__ne__() (*rti.connextdds.DataWriterSeq method*), 145
__ne__() (*rti.connextdds.DataWriterShmemRefTransferModeSettings method*), 147
__ne__() (*rti.connextdds.DataWriterTransferMode method*), 147
__ne__() (*rti.connextdds.Deadline method*), 148
__ne__() (*rti.connextdds.DestinationOrder method*), 149
__ne__() (*rti.connextdds.DestinationOrderKind method*), 151
__ne__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__ne__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__ne__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__ne__() (*rti.connextdds.Discovery method*), 154
__ne__() (*rti.connextdds.DiscoveryConfig method*), 155
__ne__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__ne__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__ne__() (*rti.connextdds.DomainParticipant method*), 165
__ne__() (*rti.connextdds.DomainParticipantConfigParams method*), 171
__ne__() (*rti.connextdds.DomainParticipantFactoryQos method*), 172
__ne__() (*rti.connextdds.DomainParticipantQos method*), 176
__ne__() (*rti.connextdds.DomainParticipantResourceLimits method*), 178
__ne__() (*rti.connextdds.DomainParticipantSeq method*), 184
__ne__() (*rti.connextdds.Durability method*), 186
__ne__() (*rti.connextdds.DurabilityKind method*), 189
__ne__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 188
__ne__() (*rti.connextdds.DurabilityService method*), 189
__ne__() (*rti.connextdds.Duration method*), 191
__ne__() (*rti.connextdds.DynamicData method*), 230
__ne__() (*rti.connextdds.DynamicData.ContentFilteredTopic method*), 193
__ne__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
__ne__() (*rti.connextdds.DynamicData.DataReader method*), 198
__ne__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__ne__() (*rti.connextdds.DynamicData.DataWriter method*), 206
__ne__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__ne__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__ne__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__ne__() (*rti.connextdds.DynamicDataInfo method*), 257
__ne__() (*rti.connextdds.DynamicDataMemberInfo method*), 257
__ne__() (*rti.connextdds.DynamicDataProperty method*), 258
__ne__() (*rti.connextdds.DynamicDataSeq method*), 260
__ne__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__ne__() (*rti.connextdds.DynamicData.Topic method*), 224
__ne__() (*rti.connextdds.DynamicData.TopicDescription method*), 224
__ne__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__ne__() (*rti.connextdds.DynamicDataTypeSerializationProperty method*), 264
__ne__() (*rti.connextdds.DynamicType method*), 264
__ne__() (*rti.connextdds.DynamicTypePrintFormatProperty method*), 265
__ne__() (*rti.connextdds.EndpointGroupSeq method*), 267
__ne__() (*rti.connextdds.EndpointGroupVector method*), 269
__ne__() (*rti.connextdds.EntityFactory method*), 270
__ne__() (*rti.connextdds.EntityName method*), 271
__ne__() (*rti.connextdds.EntitySeq method*), 272
__ne__() (*rti.connextdds.EnumMember method*), 273
__ne__() (*rti.connextdds.EnumMemberSeq method*), 275
__ne__() (*rti.connextdds.EnumType method*), 276
__ne__() (*rti.connextdds.Event method*), 277
__ne__() (*rti.connextdds.ExpressionProperty method*), 279
__ne__() (*rti.connextdds.ExtensibilityKind method*), 281
__ne__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 280
__ne__() (*rti.connextdds.FilterSampleInfo method*), 282
__ne__() (*rti.connextdds.Float32Seq method*), 286
__ne__() (*rti.connextdds.Float32Type method*), 287
__ne__() (*rti.connextdds.Float64Seq method*), 289
__ne__() (*rti.connextdds.Float64Type method*), 290
__ne__() (*rti.connextdds.Float128Seq method*), 283
__ne__() (*rti.connextdds.Float128Type method*), 285
__ne__() (*rti.connextdds.FlowController method*), 291
__ne__() (*rti.connextdds.FlowControllerProperty method*), 292
__ne__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 295
__ne__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 294
__ne__() (*rti.connextdds.FlowControllerTokenBucketProperty method*), 295
__ne__() (*rti.connextdds.GroupData method*), 297
__ne__() (*rti.connextdds.GuardCondition method*), 297

__ne__() (*rti.connextdds.Guid method*), 298
__ne__() (*rti.connextdds.heap_monitoring.HeapMonitoringParams method*), 798
__ne__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__ne__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__ne__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 803
__ne__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 801
__ne__() (*rti.connextdds.History method*), 299
__ne__() (*rti.connextdds.HistoryKind method*), 301
__ne__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__ne__() (*rti.connextdds.IAnyDataReader method*), 301
__ne__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__ne__() (*rti.connextdds.IAnyDataWriter method*), 304
__ne__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__ne__() (*rti.connextdds.IAnyTopic method*), 307
__ne__() (*rti.connextdds.IAnyTopicSeq method*), 309
__ne__() (*rti.connextdds.ICondition method*), 310
__ne__() (*rti.connextdds.IConditionSeq method*), 312
__ne__() (*rti.connextdds.IEntity method*), 313
__ne__() (*rti.connextdds.IEntitySeq method*), 315
__ne__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 319
__ne__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 318
__ne__() (*rti.connextdds.InstanceHandle method*), 320
__ne__() (*rti.connextdds.InstanceHandleSeq method*), 321
__ne__() (*rti.connextdds.InstanceState method*), 324
__ne__() (*rti.connextdds.InstanceStateConsistencyKind method*), 326
__ne__() (*rti.connextdds.Int8Seq method*), 336
__ne__() (*rti.connextdds.Int8Type method*), 337
__ne__() (*rti.connextdds.Int16Seq method*), 327
__ne__() (*rti.connextdds.Int16Type method*), 328
__ne__() (*rti.connextdds.Int32Seq method*), 330
__ne__() (*rti.connextdds.Int32Type method*), 331
__ne__() (*rti.connextdds.Int32Vector method*), 332
__ne__() (*rti.connextdds.Int64Seq method*), 333
__ne__() (*rti.connextdds.Int64Type method*), 335
__ne__() (*rti.connextdds.IReadCondition method*), 316
__ne__() (*rti.connextdds.LatencyBudget method*), 339
__ne__() (*rti.connextdds.Lifespan method*), 340
__ne__() (*rti.connextdds.Liveliness method*), 340
__ne__() (*rti.connextdds.LivelinessKind method*), 343
__ne__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__ne__() (*rti.connextdds.Locator method*), 344
__ne__() (*rti.connextdds.LocatorFilter method*), 345
__ne__() (*rti.connextdds.LocatorFilterElement method*), 346
__ne__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__ne__() (*rti.connextdds.LocatorKind method*), 351
__ne__() (*rti.connextdds.LocatorKind.LocatorKind method*), 350
__ne__() (*rti.connextdds.LocatorSeq method*), 352
__ne__() (*rti.connextdds.LocatorVector method*), 354
__ne__() (*rti.connextdds.LogCategory method*), 357
__ne__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__ne__() (*rti.connextdds.LogFacility method*), 358
__ne__() (*rti.connextdds.LogLevel method*), 360
__ne__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__ne__() (*rti.connextdds.LongDouble method*), 363
__ne__() (*rti.connextdds.Member method*), 364
__ne__() (*rti.connextdds.MemberSeq method*), 365
__ne__() (*rti.connextdds.Monitoring method*), 367
__ne__() (*rti.connextdds.MonitoringDedicatedParticipantSettings method*), 367
__ne__() (*rti.connextdds.MonitoringDistributionSettings method*), 368
__ne__() (*rti.connextdds.MonitoringEventDistributionSettings method*), 369
__ne__() (*rti.connextdds.MonitoringLoggingDistributionSettings method*), 369
__ne__() (*rti.connextdds.MonitoringLoggingForwardingSettings method*), 370
__ne__() (*rti.connextdds.MonitoringMetricSelection method*), 371
__ne__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__ne__() (*rti.connextdds.MonitoringPeriodicDistributionSettings method*), 374
__ne__() (*rti.connextdds.MonitoringTelemetryData method*), 374
__ne__() (*rti.connextdds.MulticastMapping method*), 376
__ne__() (*rti.connextdds.MulticastMappingSeq method*), 377
__ne__() (*rti.connextdds.MultiChannel method*), 375
__ne__() (*rti.connextdds.OctetType method*), 384
__ne__() (*rti.connextdds.Ownership method*), 386
__ne__() (*rti.connextdds.OwnershipKind method*), 388
__ne__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__ne__() (*rti.connextdds.OwnershipStrength method*), 388
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData method*), 425
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 389
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 395

__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 403
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
__ne__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__ne__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.Topic method*), 420
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicDescription method*), 421
__ne__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__ne__() (*rti.connextdds.Partition method*), 432
__ne__() (*rti.connextdds.PersistentJournalKind method*), 433
__ne__() (*rti.connextdds.PersistentStorageSettings method*), 433
__ne__() (*rti.connextdds.PersistentSynchronizationKind method*), 435
__ne__() (*rti.connextdds.Presentation method*), 435
__ne__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__ne__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__ne__() (*rti.connextdds.PrintFormat method*), 441
__ne__() (*rti.connextdds.PrintFormatKind method*), 443
__ne__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 442
__ne__() (*rti.connextdds.PrintFormat.PrintFormat method*), 440
__ne__() (*rti.connextdds.ProductVersion method*), 444
__ne__() (*rti.connextdds.Property method*), 445
__ne__() (*rti.connextdds.ProtocolVersion method*), 446
__ne__() (*rti.connextdds.PublicationBuiltinTopicData method*), 483
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 447
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 453
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 461
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
__ne__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__ne__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.Topic method*), 477
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.TopicDescription method*), 478
__ne__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__ne__() (*rti.connextdds.Publisher method*), 494
__ne__() (*rti.connextdds.PublisherQos method*), 496
__ne__() (*rti.connextdds.PublisherSeq method*), 499
__ne__() (*rti.connextdds.PublishMode method*), 491
__ne__() (*rti.connextdds.PublishModeKind method*), 493
__ne__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__ne__() (*rti.connextdds.QosPolicyCount method*), 500
__ne__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__ne__() (*rti.connextdds.QosPrintFormat method*), 502
__ne__() (*rti.connextdds.QosProvider method*), 503
__ne__() (*rti.connextdds.QosProviderParams method*), 506
__ne__() (*rti.connextdds.Query method*), 507
__ne__() (*rti.connextdds.QueryCondition method*), 508
__ne__() (*rti.connextdds.ReaderDataLifecycle method*), 510
__ne__() (*rti.connextdds.ReceiverPool method*), 512
__ne__() (*rti.connextdds.Reliability method*), 512
__ne__() (*rti.connextdds.ReliabilityKind method*), 515
__ne__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__ne__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 518
__ne__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__ne__() (*rti.connextdds.ResourceLimits method*), 520
__ne__() (*rti.connextdds.RtpsReliableReaderProtocol method*), 520
__ne__() (*rti.connextdds.RtpsReliableWriterProtocol method*), 521
__ne__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__ne__() (*rti.connextdds.RtpsWellKnownPorts method*), 527
__ne__() (*rti.connextdds.SampleFlag method*), 529
__ne__() (*rti.connextdds.SampleIdentity method*), 531
__ne__() (*rti.connextdds.SampleInfo method*), 531
__ne__() (*rti.connextdds.SampleLostState method*), 535
__ne__() (*rti.connextdds.SampleRejectedState method*), 538
__ne__() (*rti.connextdds.SampleState method*), 541
__ne__() (*rti.connextdds.SequenceNumber method*), 543
__ne__() (*rti.connextdds.SequenceType method*), 544
__ne__() (*rti.connextdds.Service method*), 544
__ne__() (*rti.connextdds.ServiceKind method*), 547

__ne__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__ne__() (*rti.connextdds.ServiceRequest method*), 582
__ne__() (*rti.connextdds.ServiceRequest.ContentFilteredTopic method*), 548
__ne__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__ne__() (*rti.connextdds.ServiceRequest.DataReader method*), 554
__ne__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__ne__() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
__ne__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__ne__() (*rti.connextdds.ServiceRequestId method*), 585
__ne__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__ne__() (*rti.connextdds.ServiceRequestSeq method*), 587
__ne__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
__ne__() (*rti.connextdds.ServiceRequest.Topic method*), 577
__ne__() (*rti.connextdds.ServiceRequest.TopicDescription method*), 578
__ne__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 579
__ne__() (*rti.connextdds.StatusCondition method*), 591
__ne__() (*rti.connextdds.StatusMask method*), 593
__ne__() (*rti.connextdds.StreamKind method*), 596
__ne__() (*rti.connextdds.StringPairSeq method*), 599
__ne__() (*rti.connextdds.StringSeq method*), 601
__ne__() (*rti.connextdds.StringType method*), 602
__ne__() (*rti.connextdds.StructType method*), 603
__ne__() (*rti.connextdds.Subscriber method*), 604
__ne__() (*rti.connextdds.SubscriberQos method*), 607
__ne__() (*rti.connextdds.SubscriberSeq method*), 609
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData method*), 647
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 611
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 616
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 625
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.Topic method*), 641
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicDescription method*), 642
__ne__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
__ne__() (*rti.connextdds.SyslogVerbosity method*), 656
__ne__() (*rti.connextdds.SystemResourceLimits method*), 657
__ne__() (*rti.connextdds.ThreadSettings method*), 657
__ne__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 660
__ne__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 659
__ne__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__ne__() (*rti.connextdds.Time method*), 664
__ne__() (*rti.connextdds.TimeBasedFilter method*), 665
__ne__() (*rti.connextdds.Topic method*), 666
__ne__() (*rti.connextdds.TopicBuiltinTopicData method*), 703
__ne__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 668
__ne__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
__ne__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 673
__ne__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__ne__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
__ne__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
__ne__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__ne__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
__ne__() (*rti.connextdds.TopicBuiltinTopicData.Topic method*), 697
__ne__() (*rti.connextdds.TopicBuiltinTopicData.TopicDescription method*), 698
__ne__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__ne__() (*rti.connextdds.TopicData method*), 709
__ne__() (*rti.connextdds.TopicDescription method*), 710
__ne__() (*rti.connextdds.TopicQos method*), 712
__ne__() (*rti.connextdds.TopicQuery method*), 715
__ne__() (*rti.connextdds.TopicQueryDispatch method*), 716

__ne__() (*rti.connextdds.TopicQuerySelectionKind method*), 719
__ne__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 718
__ne__() (*rti.connextdds.TopicSeq method*), 720
__ne__() (*rti.connextdds.TransportBuiltin method*), 721
__ne__() (*rti.connextdds.TransportBuiltinMask method*), 723
__ne__() (*rti.connextdds.TransportClassId method*), 728
__ne__() (*rti.connextdds.TransportClassId.TransportClassId method*), 727
__ne__() (*rti.connextdds.TransportInfoSeq method*), 730
__ne__() (*rti.connextdds.TransportInfoVector method*), 732
__ne__() (*rti.connextdds.TransportMulticast method*), 732
__ne__() (*rti.connextdds.TransportMulticastKind method*), 734
__ne__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__ne__() (*rti.connextdds.TransportMulticastMapping method*), 735
__ne__() (*rti.connextdds.TransportMulticastMappingFunction method*), 735
__ne__() (*rti.connextdds.TransportMulticastSeq method*), 737
__ne__() (*rti.connextdds.TransportMulticastSettings method*), 738
__ne__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
__ne__() (*rti.connextdds.TransportPriority method*), 741
__ne__() (*rti.connextdds.TransportSelection method*), 742
__ne__() (*rti.connextdds.TransportUnicast method*), 742
__ne__() (*rti.connextdds.TransportUnicastSettings method*), 743
__ne__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 744
__ne__() (*rti.connextdds.TypeConsistencyEnforcement method*), 747
__ne__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 750
__ne__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 749
__ne__() (*rti.connextdds.TypeKind method*), 755
__ne__() (*rti.connextdds.TypeKind.TypeKind method*), 754
__ne__() (*rti.connextdds.TypeSupport method*), 756
__ne__() (*rti.connextdds.Uint8Seq method*), 766
__ne__() (*rti.connextdds.Uint8Type method*), 767
__ne__() (*rti.connextdds.Uint16Seq method*), 757
__ne__() (*rti.connextdds.Uint16Type method*), 759
__ne__() (*rti.connextdds.Uint32Seq method*), 760
__ne__() (*rti.connextdds.Uint32Type method*), 762
__ne__() (*rti.connextdds.Uint64Seq method*), 763
__ne__() (*rti.connextdds.Uint64Type method*), 764
__ne__() (*rti.connextdds.UnidimensionalCollectionType method*), 767
__ne__() (*rti.connextdds.UnionMember method*), 768
__ne__() (*rti.connextdds.UnionMemberSeq method*), 770
__ne__() (*rti.connextdds.UnionType method*), 771
__ne__() (*rti.connextdds.UserData method*), 772
__ne__() (*rti.connextdds.VendorId method*), 773
__ne__() (*rti.connextdds.Verbosity method*), 775
__ne__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__ne__() (*rti.connextdds.ViewState method*), 777
__ne__() (*rti.connextdds.WaitSetProperty method*), 781
__ne__() (*rti.connextdds.WcharSeq method*), 782
__ne__() (*rti.connextdds.WcharType method*), 783
__ne__() (*rti.connextdds.WireProtocol method*), 784
__ne__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__ne__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__ne__() (*rti.connextdds.WriterDataLifecycle method*), 788
__ne__() (*rti.connextdds.WstringSeq method*), 789
__ne__() (*rti.connextdds.WStringType method*), 778
__next__() (*rti.connextdds.DynamicData.FieldsIterator method*), 216
__next__() (*rti.connextdds.DynamicData.IndexIterator method*), 217
__next__() (*rti.connextdds.DynamicData.ItemsIterator method*), 218
__next__() (*rti.connextdds.TriggeredConditionsIterator method*), 746
__nonzero__() (*rti.connextdds.InstanceHandle method*), 320
__or__() (*rti.connextdds.ActivityContextMask method*), 42
__or__() (*rti.connextdds.compliance.RpcMask method*), 792
__or__() (*rti.connextdds.compliance.XTypesMask method*), 796
__or__() (*rti.connextdds.CompressionIdMask method*), 76
__or__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__or__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 162
__or__() (*rti.connextdds.InstanceState method*), 324
__or__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__or__() (*rti.connextdds.SampleFlag method*), 529
__or__() (*rti.connextdds.SampleLostState method*), 535
__or__() (*rti.connextdds.SampleRejectedState method*), 538
__or__() (*rti.connextdds.SampleState method*), 541
__or__() (*rti.connextdds.StatusMask method*), 593
__or__() (*rti.connextdds.StreamKind method*), 596
__or__() (*rti.connextdds.ThreadSettingsKindMask method*), 661
__or__() (*rti.connextdds.TransportBuiltinMask method*), 723
__or__() (*rti.connextdds.ViewState method*), 777
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.BoolSeq attribute*), 59
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.CharSeq attribute*), 71
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.DynamicDataTimestampedSeq attribute*), 262

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Float32Seq attribute*), 286
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Float64Seq attribute*), 289
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.IAnyDataReaderSeq attribute*), 303
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.IAnyDataWriterSeq attribute*), 306
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.IAnyTopicSeq attribute*), 309
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.IConditionSeq attribute*), 312
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.IEntitySeq attribute*), 315
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Int8Seq attribute*), 336
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Int16Seq attribute*), 327
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Int32Seq attribute*), 330
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Int64Seq attribute*), 333
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.ItemsView attribute*), 338
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.KeysView attribute*), 339
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq attribute*), 430
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq attribute*), 489
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.ServiceRequestTimestampedSeq attribute*), 589
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.StringMap attribute*), 598
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.StringPairSeq attribute*), 599
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.StringSeq attribute*), 601
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq attribute*), 653
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq attribute*), 708
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Uint8Seq attribute*), 766
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Uint16Seq attribute*), 758
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Uint32Seq attribute*), 760
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.Uint64Seq attribute*), 763
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.ValuesView attribute*), 773
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.WcharSeq attribute*), 782
__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1011__ (*rti.connextdds.WstringSeq attribute*), 789
__radd__() (*rti.connextdds.Time method*), 664
__repr__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 40
__repr__() (*rti.connextdds.ActivityContextMask method*), 42
__repr__() (*rti.connextdds.BoolSeq method*), 59
__repr__() (*rti.connextdds.BuiltinTopicKey method*), 63
__repr__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__repr__() (*rti.connextdds.CharSeq method*), 71
__repr__() (*rti.connextdds.compliance.RpcMask method*), 792
__repr__() (*rti.connextdds.compliance.XTypesMask method*), 796
__repr__() (*rti.connextdds.CompressionIdMask method*), 76
__repr__() (*rti.connextdds.CookieSeq method*), 86
__repr__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__repr__() (*rti.connextdds.DataState method*), 117
__repr__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__repr__() (*rti.connextdds.DestinationOrderKind.Destina-*

*tionOrderKind method*), 150
__repr__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__repr__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__repr__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
__repr__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 188
__repr__() (*rti.connextdds.Duration method*), 191
__repr__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__repr__() (*rti.connextdds.DynamicDataSeq method*), 260
__repr__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 280
__repr__() (*rti.connextdds.Float32Seq method*), 286
__repr__() (*rti.connextdds.Float64Seq method*), 289
__repr__() (*rti.connextdds.Float128Seq method*), 283
__repr__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 294
__repr__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__repr__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 802
__repr__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__repr__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__repr__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__repr__() (*rti.connextdds.IAnyTopicSeq method*), 309
__repr__() (*rti.connextdds.IConditionSeq method*), 312
__repr__() (*rti.connextdds.IEntitySeq method*), 315
__repr__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 318
__repr__() (*rti.connextdds.InstanceHandleSeq method*), 321
__repr__() (*rti.connextdds.InstanceStateConsistencyKind method*), 326
__repr__() (*rti.connextdds.Int8Seq method*), 336
__repr__() (*rti.connextdds.Int16Seq method*), 327
__repr__() (*rti.connextdds.Int32Seq method*), 330
__repr__() (*rti.connextdds.Int64Seq method*), 333
__repr__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__repr__() (*rti.connextdds.LocatorKind.LocatorKind method*), 350
__repr__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__repr__() (*rti.connextdds.LogFacility method*), 358
__repr__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__repr__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__repr__() (*rti.connextdds.PersistentJournalKind method*), 433
__repr__() (*rti.connextdds.PersistentSynchronizationKind method*), 435
__repr__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__repr__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 442
__repr__() (*rti.connextdds.PrintFormat.PrintFormat method*), 440
__repr__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 492
__repr__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__repr__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__repr__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__repr__() (*rti.connextdds.SampleInfo method*), 531
__repr__() (*rti.connextdds.SequenceNumber method*), 543
__repr__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__repr__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__repr__() (*rti.connextdds.StatusMask method*), 593
__repr__() (*rti.connextdds.StringMap method*), 598
__repr__() (*rti.connextdds.StringSeq method*), 601
__repr__() (*rti.connextdds.SyslogVerbosity method*), 656
__repr__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 659
__repr__() (*rti.connextdds.ThreadSettingsKindMask method*), 662
__repr__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 718
__repr__() (*rti.connextdds.TransportBuiltinMask method*), 723
__repr__() (*rti.connextdds.TransportClassId.TransportClassId method*), 727
__repr__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__repr__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 749
__repr__() (*rti.connextdds.TypeKind.TypeKind method*), 754
__repr__() (*rti.connextdds.Uint8Seq method*), 766
__repr__() (*rti.connextdds.Uint16Seq method*), 758
__repr__() (*rti.connextdds.Uint32Seq method*), 760
__repr__() (*rti.connextdds.Uint64Seq method*), 763
__repr__() (*rti.connextdds.Verbosity.Verbosity method*), 774
__repr__() (*rti.connextdds.WcharSeq method*), 782
__repr__() (*rti.connextdds.WireProtocolAutoKind.WireProto-*

*colAutoKind method*), 785
__repr__() (*rti.connextdds.WstringSeq method*), 789
__reversed__() (*rti.connextdds.DynamicData method*), 230
__reversed__() (*rti.connextdds.DynamicData.FieldsView method*), 217
__reversed__() (*rti.connextdds.DynamicData.ItemsView method*), 218
__reversed__() (*rti.connextdds.TriggeredConditions method*), 746
__rmul__() (*rti.connextdds.AnyDataReaderSeq method*), 48
__rmul__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__rmul__() (*rti.connextdds.AnyTopicSeq method*), 54
__rmul__() (*rti.connextdds.BoolSeq method*), 59
__rmul__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__rmul__() (*rti.connextdds.CharSeq method*), 71
__rmul__() (*rti.connextdds.ConditionSeq method*), 80
__rmul__() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
__rmul__() (*rti.connextdds.CookieSeq method*), 86
__rmul__() (*rti.connextdds.DataReaderSeq method*), 114
__rmul__() (*rti.connextdds.DataWriterSeq method*), 145
__rmul__() (*rti.connextdds.DomainParticipantSeq method*), 184
__rmul__() (*rti.connextdds.Duration method*), 191
__rmul__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
__rmul__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__rmul__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__rmul__() (*rti.connextdds.DynamicDataSeq method*), 260
__rmul__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__rmul__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__rmul__() (*rti.connextdds.EndpointGroupSeq method*), 267
__rmul__() (*rti.connextdds.EntitySeq method*), 272
__rmul__() (*rti.connextdds.EnumMemberSeq method*), 275
__rmul__() (*rti.connextdds.Float32Seq method*), 286
__rmul__() (*rti.connextdds.Float64Seq method*), 289
__rmul__() (*rti.connextdds.Float128Seq method*), 283
__rmul__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__rmul__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__rmul__() (*rti.connextdds.IAnyTopicSeq method*), 309
__rmul__() (*rti.connextdds.IConditionSeq method*), 312
__rmul__() (*rti.connextdds.IEntitySeq method*), 315
__rmul__() (*rti.connextdds.InstanceHandleSeq method*), 321
__rmul__() (*rti.connextdds.Int8Seq method*), 336
__rmul__() (*rti.connextdds.Int16Seq method*), 327
__rmul__() (*rti.connextdds.Int32Seq method*), 330
__rmul__() (*rti.connextdds.Int64Seq method*), 333
__rmul__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__rmul__() (*rti.connextdds.LocatorSeq method*), 352
__rmul__() (*rti.connextdds.MemberSeq method*), 365
__rmul__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__rmul__() (*rti.connextdds.MulticastMappingSeq method*), 377
__rmul__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__rmul__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__rmul__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
__rmul__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__rmul__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
__rmul__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__rmul__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__rmul__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
__rmul__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
__rmul__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486
__rmul__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
__rmul__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__rmul__() (*rti.connextdds.PublisherSeq method*), 499
__rmul__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__rmul__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__rmul__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__rmul__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 570
__rmul__() (*rti.connextdds.ServiceRequestSeq method*), 587
__rmul__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
__rmul__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
__rmul__() (*rti.connextdds.StringPairSeq method*), 599
__rmul__() (*rti.connextdds.StringSeq method*), 601
__rmul__() (*rti.connextdds.SubscriberSeq method*), 609
__rmul__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
__rmul__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 622

__rmul__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__rmul__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__rmul__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
__rmul__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
__rmul__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
__rmul__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__rmul__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
__rmul__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__rmul__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
__rmul__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__rmul__() (*rti.connextdds.TopicSeq method*), 720
__rmul__() (*rti.connextdds.TransportInfoSeq method*), 730
__rmul__() (*rti.connextdds.TransportMulticastSeq method*), 737
__rmul__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
__rmul__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
__rmul__() (*rti.connextdds.Uint8Seq method*), 766
__rmul__() (*rti.connextdds.Uint16Seq method*), 758
__rmul__() (*rti.connextdds.Uint32Seq method*), 760
__rmul__() (*rti.connextdds.Uint64Seq method*), 763
__rmul__() (*rti.connextdds.UnionMemberSeq method*), 770
__rmul__() (*rti.connextdds.WcharSeq method*), 782
__rmul__() (*rti.connextdds.WstringSeq method*), 790
__rshift__() (*rti.connextdds.ActivityContextMask method*), 43
__rshift__() (*rti.connextdds.compliance.RpcMask method*), 792
__rshift__() (*rti.connextdds.compliance.XTypesMask method*), 796
__rshift__() (*rti.connextdds.CompressionIdMask method*), 76
__rshift__() (*rti.connextdds.DataReader method*), 92
__rshift__() (*rti.connextdds.DataReaderQos method*), 106
__rshift__() (*rti.connextdds.DataWriter method*), 122
__rshift__() (*rti.connextdds.DataWriterQos method*), 134
__rshift__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 159
__rshift__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
__rshift__() (*rti.connextdds.DomainParticipant method*), 165
__rshift__() (*rti.connextdds.DomainParticipantFactoryQos method*), 172
__rshift__() (*rti.connextdds.DomainParticipantQos method*), 176
__rshift__() (*rti.connextdds.DynamicData.DataReader method*), 198
__rshift__() (*rti.connextdds.DynamicData.DataWriter method*), 206
__rshift__() (*rti.connextdds.InstanceState method*), 324
__rshift__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 395
__rshift__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 403
__rshift__() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 453
__rshift__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 461
__rshift__() (*rti.connextdds.Publisher method*), 494
__rshift__() (*rti.connextdds.PublisherQos method*), 496
__rshift__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__rshift__() (*rti.connextdds.SampleFlag method*), 529
__rshift__() (*rti.connextdds.SampleLostState method*), 535
__rshift__() (*rti.connextdds.SampleRejectedState method*), 538
__rshift__() (*rti.connextdds.SampleState method*), 541
__rshift__() (*rti.connextdds.ServiceRequest.DataReader method*), 554
__rshift__() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
__rshift__() (*rti.connextdds.StatusMask method*), 593
__rshift__() (*rti.connextdds.StreamKind method*), 596
__rshift__() (*rti.connextdds.SubscriberQos method*), 607
__rshift__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 617
__rshift__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 625
__rshift__() (*rti.connextdds.ThreadSettingsKindMask method*), 662
__rshift__() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 673
__rshift__() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
__rshift__() (*rti.connextdds.TopicQos method*), 712
__rshift__() (*rti.connextdds.TransportBuiltinMask method*), 723
__rshift__() (*rti.connextdds.ViewState method*), 777

__setitem__() (*rti.connextdds.ActivityContextMask method*), 43
__setitem__() (*rti.connextdds.AnyDataReaderSeq method*), 48
__setitem__() (*rti.connextdds.AnyDataWriterSeq method*), 51
__setitem__() (*rti.connextdds.AnyTopicSeq method*), 54
__setitem__() (*rti.connextdds.BoolSeq method*), 59
__setitem__() (*rti.connextdds.ByteVector method*), 65
__setitem__() (*rti.connextdds.ChannelSettingsSeq method*), 69
__setitem__() (*rti.connextdds.CharSeq method*), 71
__setitem__() (*rti.connextdds.compliance.RpcMask method*), 793
__setitem__() (*rti.connextdds.compliance.XTypesMask method*), 796
__setitem__() (*rti.connextdds.CompressionIdMask method*), 76
__setitem__() (*rti.connextdds.ConditionSeq method*), 80
__setitem__() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
__setitem__() (*rti.connextdds.CookieSeq method*), 86
__setitem__() (*rti.connextdds.CookieVector method*), 88
__setitem__() (*rti.connextdds.DataReaderSeq method*), 114
__setitem__() (*rti.connextdds.DataTag method*), 120
__setitem__() (*rti.connextdds.DataWriterSeq method*), 145
__setitem__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 160
__setitem__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
__setitem__() (*rti.connextdds.DomainParticipantSeq method*), 184
__setitem__() (*rti.connextdds.DynamicData method*), 230
__setitem__() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
__setitem__() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
__setitem__() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
__setitem__() (*rti.connextdds.DynamicDataSeq method*), 260
__setitem__() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
__setitem__() (*rti.connextdds.DynamicData.TopicSeq method*), 226
__setitem__() (*rti.connextdds.EndpointGroupSeq method*), 267
__setitem__() (*rti.connextdds.EndpointGroupVector method*), 269
__setitem__() (*rti.connextdds.EntitySeq method*), 272
__setitem__() (*rti.connextdds.EnumMemberSeq method*), 275
__setitem__() (*rti.connextdds.Float32Seq method*), 286
__setitem__() (*rti.connextdds.Float64Seq method*), 289
__setitem__() (*rti.connextdds.Float128Seq method*), 283
__setitem__() (*rti.connextdds.Guid method*), 298
__setitem__() (*rti.connextdds.IAnyDataReaderSeq method*), 303
__setitem__() (*rti.connextdds.IAnyDataWriterSeq method*), 306
__setitem__() (*rti.connextdds.IAnyTopicSeq method*), 309
__setitem__() (*rti.connextdds.IConditionSeq method*), 312
__setitem__() (*rti.connextdds.IEntitySeq method*), 315
__setitem__() (*rti.connextdds.InstanceHandleSeq method*), 322
__setitem__() (*rti.connextdds.InstanceState method*), 324
__setitem__() (*rti.connextdds.Int8Seq method*), 336
__setitem__() (*rti.connextdds.Int16Seq method*), 327
__setitem__() (*rti.connextdds.Int32Seq method*), 330
__setitem__() (*rti.connextdds.Int32Vector method*), 332
__setitem__() (*rti.connextdds.Int64Seq method*), 333
__setitem__() (*rti.connextdds.LocatorFilterElementSeq method*), 347
__setitem__() (*rti.connextdds.LocatorSeq method*), 352
__setitem__() (*rti.connextdds.LocatorVector method*), 354
__setitem__() (*rti.connextdds.LongDouble method*), 363
__setitem__() (*rti.connextdds.MemberSeq method*), 365
__setitem__() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 372
__setitem__() (*rti.connextdds.MulticastMappingSeq method*), 377
__setitem__() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 391
__setitem__() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
__setitem__() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
__setitem__() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 427
__setitem__() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
__setitem__() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
__setitem__() (*rti.connextdds.Property method*), 445
__setitem__() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
__setitem__() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
__setitem__() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
__setitem__() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 486

__setitem__() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
__setitem__() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
__setitem__() (*rti.connextdds.PublisherSeq method*), 499
__setitem__() (*rti.connextdds.QosPolicyCountSeq method*), 501
__setitem__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__setitem__() (*rti.connextdds.SampleFlag method*), 529
__setitem__() (*rti.connextdds.SampleLostState method*), 535
__setitem__() (*rti.connextdds.SampleRejectedState method*), 538
__setitem__() (*rti.connextdds.SampleState method*), 541
__setitem__() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
__setitem__() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 559
__setitem__() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
__setitem__() (*rti.connextdds.ServiceRequestSeq method*), 587
__setitem__() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
__setitem__() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
__setitem__() (*rti.connextdds.StatusMask method*), 593
__setitem__() (*rti.connextdds.StreamKind method*), 596
__setitem__() (*rti.connextdds.StringMap method*), 598
__setitem__() (*rti.connextdds.StringPairSeq method*), 599
__setitem__() (*rti.connextdds.StringSeq method*), 601
__setitem__() (*rti.connextdds.SubscriberSeq method*), 609
__setitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
__setitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
__setitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 634
__setitem__() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
__setitem__() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
__setitem__() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
__setitem__() (*rti.connextdds.ThreadSettingsKindMask method*), 662
__setitem__() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
__setitem__() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
__setitem__() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
__setitem__() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 705
__setitem__() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
__setitem__() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
__setitem__() (*rti.connextdds.TopicSeq method*), 720
__setitem__() (*rti.connextdds.TransportBuiltinMask method*), 724
__setitem__() (*rti.connextdds.TransportInfoSeq method*), 730
__setitem__() (*rti.connextdds.TransportInfoVector method*), 732
__setitem__() (*rti.connextdds.TransportMulticastSeq method*), 737
__setitem__() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
__setitem__() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
__setitem__() (*rti.connextdds.Uint8Seq method*), 766
__setitem__() (*rti.connextdds.Uint16Seq method*), 758
__setitem__() (*rti.connextdds.Uint32Seq method*), 760
__setitem__() (*rti.connextdds.Uint64Seq method*), 763
__setitem__() (*rti.connextdds.UnionMemberSeq method*), 770
__setitem__() (*rti.connextdds.ViewState method*), 777
__setitem__() (*rti.connextdds.WcharSeq method*), 782
__setitem__() (*rti.connextdds.WstringSeq method*), 790
__setstate__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 40
__setstate__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__setstate__() (*rti.connextdds.CharSeq method*), 71
__setstate__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98
__setstate__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__setstate__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__setstate__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152

__setstate__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 188
__setstate__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__setstate__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 280
__setstate__() (*rti.connextdds.Float32Seq method*), 286
__setstate__() (*rti.connextdds.Float64Seq method*), 289
__setstate__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 294
__setstate__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__setstate__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 802
__setstate__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__setstate__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 318
__setstate__() (*rti.connextdds.InstanceStateConsistencyKind method*), 326
__setstate__() (*rti.connextdds.Int8Seq method*), 336
__setstate__() (*rti.connextdds.Int16Seq method*), 327
__setstate__() (*rti.connextdds.Int32Seq method*), 330
__setstate__() (*rti.connextdds.Int64Seq method*), 334
__setstate__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__setstate__() (*rti.connextdds.LocatorKind.LocatorKind method*), 350
__setstate__() (*rti.connextdds.LogCategory.LogCategory method*), 355
__setstate__() (*rti.connextdds.LogFacility method*), 358
__setstate__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__setstate__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__setstate__() (*rti.connextdds.PersistentJournalKind method*), 433
__setstate__() (*rti.connextdds.PersistentSynchronizationKind method*), 435
__setstate__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__setstate__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 442
__setstate__() (*rti.connextdds.PrintFormat.PrintFormat method*), 440
__setstate__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 493
__setstate__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__setstate__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517
__setstate__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__setstate__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__setstate__() (*rti.connextdds.SyslogVerbosity method*), 656
__setstate__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 659
__setstate__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 718
__setstate__() (*rti.connextdds.TransportClassId.TransportClassId method*), 727
__setstate__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__setstate__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 749
__setstate__() (*rti.connextdds.TypeKind.TypeKind method*), 754
__setstate__() (*rti.connextdds.Uint8Seq method*), 766
__setstate__() (*rti.connextdds.Uint16Seq method*), 758
__setstate__() (*rti.connextdds.Uint32Seq method*), 761
__setstate__() (*rti.connextdds.Uint64Seq method*), 763
__setstate__() (*rti.connextdds.Verbosity.Verbosity method*), 775
__setstate__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__str__() (*rti.connextdds.AcknowledgmentKind method*), 41
__str__() (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind method*), 40
__str__() (*rti.connextdds.ActivityContextMask method*), 43
__str__() (*rti.connextdds.CdrPaddingKind method*), 67
__str__() (*rti.connextdds.CdrPaddingKind.CdrPaddingKind method*), 66
__str__() (*rti.connextdds.CoherentSetInfo method*), 74
__str__() (*rti.connextdds.compliance.RpcMask method*), 793
__str__() (*rti.connextdds.compliance.XTypesMask method*), 796
__str__() (*rti.connextdds.CompressionIdMask method*), 76
__str__() (*rti.connextdds.DataReaderInstanceRemovalKind method*), 99
__str__() (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 98

__str__() (*rti.connextdds.DataReaderQos method*), 108
__str__() (*rti.connextdds.DataWriterQos method*), 137
__str__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind method*), 144
__str__() (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 143
__str__() (*rti.connextdds.DestinationOrderKind method*), 151
__str__() (*rti.connextdds.DestinationOrderKind.DestinationOrderKind method*), 150
__str__() (*rti.connextdds.DestinationOrderScopeKind method*), 153
__str__() (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 152
__str__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 160
__str__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
__str__() (*rti.connextdds.DomainParticipantFactoryQos method*), 173
__str__() (*rti.connextdds.DomainParticipantQos method*), 177
__str__() (*rti.connextdds.DurabilityKind method*), 189
__str__() (*rti.connextdds.DurabilityKind.DurabilityKind method*), 188
__str__() (*rti.connextdds.Duration method*), 191
__str__() (*rti.connextdds.DynamicData method*), 230
__str__() (*rti.connextdds.DynamicDataEncapsulationKind method*), 256
__str__() (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 255
__str__() (*rti.connextdds.DynamicType method*), 264
__str__() (*rti.connextdds.EnumMember method*), 274
__str__() (*rti.connextdds.ExtensibilityKind method*), 281
__str__() (*rti.connextdds.ExtensibilityKind.ExtensibilityKind method*), 280
__str__() (*rti.connextdds.FlowControllerSchedulingPolicy method*), 295
__str__() (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 294
__str__() (*rti.connextdds.Guid method*), 298
__str__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat method*), 800
__str__() (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat method*), 799
__str__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat method*), 803
__str__() (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat method*), 802
__str__() (*rti.connextdds.HistoryKind method*), 301
__str__() (*rti.connextdds.HistoryKind.HistoryKind method*), 300
__str__() (*rti.connextdds.IgnoredEntityReplacementKind method*), 319
__str__() (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 318
__str__() (*rti.connextdds.InstanceHandle method*), 320
__str__() (*rti.connextdds.InstanceState method*), 324
__str__() (*rti.connextdds.InstanceStateConsistencyKind method*), 326
__str__() (*rti.connextdds.LivelinessKind method*), 343
__str__() (*rti.connextdds.LivelinessKind.LivelinessKind method*), 342
__str__() (*rti.connextdds.LocatorKind method*), 351
__str__() (*rti.connextdds.LocatorKind.LocatorKind method*), 350
__str__() (*rti.connextdds.LogCategory method*), 357
__str__() (*rti.connextdds.LogCategory.LogCategory method*), 356
__str__() (*rti.connextdds.LogFacility method*), 358
__str__() (*rti.connextdds.LogLevel method*), 360
__str__() (*rti.connextdds.LogLevel.LogLevel method*), 359
__str__() (*rti.connextdds.LongDouble method*), 363
__str__() (*rti.connextdds.OwnershipKind method*), 388
__str__() (*rti.connextdds.OwnershipKind.OwnershipKind method*), 387
__str__() (*rti.connextdds.PersistentJournalKind method*), 433
__str__() (*rti.connextdds.PersistentSynchronizationKind method*), 435
__str__() (*rti.connextdds.PresentationAccessScopeKind method*), 438
__str__() (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 437
__str__() (*rti.connextdds.PrintFormat method*), 441
__str__() (*rti.connextdds.PrintFormatKind method*), 443
__str__() (*rti.connextdds.PrintFormatKind.PrintFormatKind method*), 442
__str__() (*rti.connextdds.PrintFormat.PrintFormat method*), 440
__str__() (*rti.connextdds.ProductVersion method*), 444
__str__() (*rti.connextdds.PublisherQos method*), 497
__str__() (*rti.connextdds.PublishModeKind method*), 493
__str__() (*rti.connextdds.PublishModeKind.PublishModeKind method*), 493
__str__() (*rti.connextdds.ReliabilityKind method*), 515
__str__() (*rti.connextdds.ReliabilityKind.ReliabilityKind method*), 514
__str__() (*rti.connextdds.RemoteParticipantPurgeKind method*), 518
__str__() (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 517

__str__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__str__() (*rti.connextdds.SampleFlag method*), 529
__str__() (*rti.connextdds.SampleIdentity method*), 531
__str__() (*rti.connextdds.SampleLostState method*), 535
__str__() (*rti.connextdds.SampleRejectedState method*), 538
__str__() (*rti.connextdds.SampleState method*), 541
__str__() (*rti.connextdds.ServiceKind method*), 547
__str__() (*rti.connextdds.ServiceKind.ServiceKind method*), 546
__str__() (*rti.connextdds.ServiceRequestId method*), 585
__str__() (*rti.connextdds.ServiceRequestId.ServiceRequestId method*), 584
__str__() (*rti.connextdds.StatusMask method*), 593
__str__() (*rti.connextdds.StreamKind method*), 596
__str__() (*rti.connextdds.SubscriberQos method*), 607
__str__() (*rti.connextdds.SyslogVerbosity method*), 656
__str__() (*rti.connextdds.ThreadSettingsCpuRotationKind method*), 660
__str__() (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 659
__str__() (*rti.connextdds.ThreadSettingsKindMask method*), 662
__str__() (*rti.connextdds.TopicQos method*), 713
__str__() (*rti.connextdds.TopicQuerySelectionKind method*), 719
__str__() (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 718
__str__() (*rti.connextdds.TransportBuiltinMask method*), 724
__str__() (*rti.connextdds.TransportClassId method*), 728
__str__() (*rti.connextdds.TransportClassId.TransportClassId method*), 727
__str__() (*rti.connextdds.TransportMulticastKind method*), 734
__str__() (*rti.connextdds.TransportMulticastKind.TransportMulticastKind method*), 733
__str__() (*rti.connextdds.TypeConsistencyEnforcementKind method*), 750
__str__() (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 749
__str__() (*rti.connextdds.TypeKind method*), 755
__str__() (*rti.connextdds.TypeKind.TypeKind method*), 754
__str__() (*rti.connextdds.Verbosity method*), 775
__str__() (*rti.connextdds.Verbosity.Verbosity method*), 775
__str__() (*rti.connextdds.ViewState method*), 777
__str__() (*rti.connextdds.WireProtocolAutoKind method*), 786
__str__() (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 785
__sub__() (*rti.connextdds.Duration method*), 191
__sub__() (*rti.connextdds.SequenceNumber method*), 543
__sub__() (*rti.connextdds.Time method*), 664
__truediv__() (*rti.connextdds.Duration method*), 191
__weakref__ (*rti.connextdds.Exception attribute*), 278
__xor__() (*rti.connextdds.ActivityContextMask method*), 43
__xor__() (*rti.connextdds.compliance.RpcMask method*), 793
__xor__() (*rti.connextdds.compliance.XTypesMask method*), 796
__xor__() (*rti.connextdds.CompressionIdMask method*), 76
__xor__() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 160
__xor__() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
__xor__() (*rti.connextdds.InstanceState method*), 324
__xor__() (*rti.connextdds.RtpsReservedPortKindMask method*), 525
__xor__() (*rti.connextdds.SampleFlag method*), 529
__xor__() (*rti.connextdds.SampleLostState method*), 535
__xor__() (*rti.connextdds.SampleRejectedState method*), 538
__xor__() (*rti.connextdds.SampleState method*), 541
__xor__() (*rti.connextdds.StatusMask method*), 594
__xor__() (*rti.connextdds.StreamKind method*), 596
__xor__() (*rti.connextdds.ThreadSettingsKindMask method*), 662
__xor__() (*rti.connextdds.TransportBuiltinMask method*), 724
__xor__() (*rti.connextdds.ViewState method*), 777

## A

absolute_generation (*rti.connextdds.Rank property*), 509
ACCEPT_UNKNOWN_DISCRIMINATOR_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
ACCEPT_UNKNOWN_ENUM_VALUE_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
accept_unknown_peers (*rti.connextdds.Discovery property*), 154
access_scope (*rti.connextdds.Presentation property*), 435
acknowledge_all() (*rti.connextdds.DataReader method*), 92
acknowledge_all() (*rti.connextdds.DynamicData.DataReader method*), 199
acknowledge_all() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 395
acknowledge_all() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 453
acknowledge_all() (*rti.connextdds.ServiceRequest.DataReader method*), 554
acknowledge_all() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 617
acknowledge_all() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 673
acknowledge_sample() (*rti.connextdds.DataReader method*), 92

acknowledge_sample() (*rti.connextdds.DynamicData.DataReader method*), 199
acknowledge_sample() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 395
acknowledge_sample() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 453
acknowledge_sample() (*rti.connextdds.ServiceRequest.DataReader method*), 554
acknowledge_sample() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 617
acknowledge_sample() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 674
acknowledgment_kind (*rti.connextdds.Reliability property*), 512
AcknowledgmentInfo (*class in rti.connextdds*), 38
AcknowledgmentKind (*class in rti.connextdds*), 38
AcknowledgmentKind.AcknowledgmentKind (*class in rti.connextdds*), 38
AckResponseData (*class in rti.connextdds*), 37
ACTEnumMember (*class in rti.connextdds*), 34
active_count (*rti.connextdds.ReliableReaderActivityChangedStatus property*), 515
ACTIVITY (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 798
ACTIVITY (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*), 799
ActivityContextMask (*class in rti.connextdds*), 41
ACTMember (*class in rti.connextdds*), 35
ACTUnionMember (*class in rti.connextdds*), 36
add_cookie() (*rti.connextdds.DynamicData.WriterContentFilterHelper method*), 229
add_cookie() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilterHelper method*), 425
add_cookie() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper method*), 483
add_cookie() (*rti.connextdds.ServiceRequest.WriterContentFilterHelper method*), 582
add_cookie() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper method*), 646
add_cookie() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper method*), 702
add_member() (*rti.connextdds.EnumType method*), 276
add_member() (*rti.connextdds.StructType method*), 603
add_members() (*rti.connextdds.EnumType method*), 276
add_members() (*rti.connextdds.StructType method*), 604
add_members() (*rti.connextdds.UnionType method*), 771
add_parameter() (*rti.connextdds.Query method*), 507
add_parameters() (*rti.connextdds.Filter method*), 281
add_peer() (*rti.connextdds.DomainParticipant method*), 165
add_peers() (*rti.connextdds.DomainParticipant method*), 165
address (*rti.connextdds.Locator property*), 344
addresses (*rti.connextdds.MulticastMapping property*), 376
ADMIN_CONSOLE (*rti.connextdds.ServiceKind attribute*), 545
ADMIN_CONSOLE (*rti.connextdds.ServiceKind.ServiceKind attribute*), 545
AGGREGATION_TYPE (*rti.connextdds.TypeKind attribute*), 750
AGGREGATION_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
ALERT (*rti.connextdds.SyslogVerbosity attribute*), 655
alert() (*rti.connextdds.Logger method*), 361
ALIAS_TYPE (*rti.connextdds.TypeKind attribute*), 750
ALIAS_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
AliasType (*class in rti.connextdds*), 44
ALIVE (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 140
ALIVE (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142
ALIVE (*rti.connextdds.InstanceState attribute*), 322
alive_count (*rti.connextdds.LivelinessChangedStatus property*), 341
alive_count_change (*rti.connextdds.LivelinessChangedStatus property*), 341
alive_instance_count (*rti.connextdds.DataReaderCacheStatus property*), 96
alive_instance_count (*rti.connextdds.DataWriterCacheStatus property*), 127
alive_instance_count_peak (*rti.connextdds.DataReaderCacheStatus property*), 96
alive_instance_count_peak (*rti.connextdds.DataWriterCacheStatus property*), 127
alive_instance_removal (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings property*), 113
ALIVE_OR_DISPOSED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 140
ALIVE_OR_DISPOSED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142
ALIVE_THEN_DISPOSED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 141
ALIVE_THEN_DISPOSED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142

ALL (*rti.connextdds.ActivityContextMask attribute*), 41
ALL (*rti.connextdds.CompressionIdMask attribute*), 75
ALL (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 158
ALL (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
ALL (*rti.connextdds.StatusMask attribute*), 591
all (*rti.connextdds.TransportBuiltin attribute*), 722
ALL (*rti.connextdds.TransportBuiltinMask attribute*), 722
all_categories (*rti.connextdds.LogCategory attribute*), 357
all_categories (*rti.connextdds.LogCategory.LogCategory attribute*), 356
AllocationSettings (*class in rti.connextdds*), 44
allow_type_coercion (*rti.connextdds.TypeConsistencyEnforcement attribute*), 747
ALLOW_TYPE_COERCION (*rti.connextdds.TypeConsistencyEnforcementKind attribute*), 747
ALLOW_TYPE_COERCION (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 748
AlreadyClosedError, 45
ANNOTATION_TYPE (*rti.connextdds.TypeKind attribute*), 750
ANNOTATION_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
any (*rti.connextdds.DataState attribute*), 117
any (*rti.connextdds.DataStateEx attribute*), 118
ANY (*rti.connextdds.InstanceState attribute*), 322
ANY (*rti.connextdds.LocatorKind attribute*), 348
ANY (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
ANY (*rti.connextdds.SampleState attribute*), 540
ANY (*rti.connextdds.StreamKind attribute*), 595
ANY (*rti.connextdds.TransportClassId attribute*), 725
ANY (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
ANY (*rti.connextdds.ViewState attribute*), 776
any_data (*rti.connextdds.DataState attribute*), 117
any_data (*rti.connextdds.DataStateEx attribute*), 118
ANY_INSTANCE (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 97
ANY_INSTANCE (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 97
AnyDataReader (*class in rti.connextdds*), 45
AnyDataReaderListener (*class in rti.connextdds*), 45
AnyDataReaderSeq (*class in rti.connextdds*), 46
AnyDataWriter (*class in rti.connextdds*), 48
AnyDataWriterListener (*class in rti.connextdds*), 49
AnyDataWriterSeq (*class in rti.connextdds*), 50
AnyTopic (*class in rti.connextdds*), 52
AnyTopicListener (*class in rti.connextdds*), 52
AnyTopicSeq (*class in rti.connextdds*), 52
api (*rti.connextdds.LogCategory attribute*), 357
api (*rti.connextdds.LogCategory.LogCategory attribute*), 356
app_ack_period (*rti.connextdds.RtpsReliableReaderProtocol property*), 520
append() (*rti.connextdds.AnyDataReaderSeq method*), 48
append() (*rti.connextdds.AnyDataWriterSeq method*), 51
append() (*rti.connextdds.AnyTopicSeq method*), 54
append() (*rti.connextdds.BoolSeq method*), 59
append() (*rti.connextdds.ChannelSettingsSeq method*), 69
append() (*rti.connextdds.CharSeq method*), 71
append() (*rti.connextdds.ConditionSeq method*), 80
append() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
append() (*rti.connextdds.CookieSeq method*), 87
append() (*rti.connextdds.DataReaderSeq method*), 114
append() (*rti.connextdds.DataWriterSeq method*), 146
append() (*rti.connextdds.DomainParticipantSeq method*), 185
append() (*rti.connextdds.DynamicData method*), 230
append() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
append() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
append() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
append() (*rti.connextdds.DynamicDataSeq method*), 260
append() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
append() (*rti.connextdds.DynamicData.TopicSeq method*), 226
append() (*rti.connextdds.EndpointGroupSeq method*), 267
append() (*rti.connextdds.EntitySeq method*), 272
append() (*rti.connextdds.EnumMemberSeq method*), 275
append() (*rti.connextdds.Float32Seq method*), 286
append() (*rti.connextdds.Float64Seq method*), 289
append() (*rti.connextdds.Float128Seq method*), 284
append() (*rti.connextdds.IAnyDataReaderSeq method*), 303
append() (*rti.connextdds.IAnyDataWriterSeq method*), 307
append() (*rti.connextdds.IAnyTopicSeq method*), 309
append() (*rti.connextdds.IConditionSeq method*), 312
append() (*rti.connextdds.IEntitySeq method*), 315
append() (*rti.connextdds.InstanceHandleSeq method*), 322
append() (*rti.connextdds.Int8Seq method*), 336
append() (*rti.connextdds.Int16Seq method*), 327
append() (*rti.connextdds.Int32Seq method*), 330
append() (*rti.connextdds.Int64Seq method*), 334
append() (*rti.connextdds.LocatorFilterElementSeq method*), 347
append() (*rti.connextdds.LocatorSeq method*), 353
append() (*rti.connextdds.MemberSeq method*), 366
append() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
append() (*rti.connextdds.MulticastMappingSeq method*), 378
append() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
append() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
append() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
append() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
append() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*),

430
append() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
append() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 449
append() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
append() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
append() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
append() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
append() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
append() (*rti.connextdds.PublisherSeq method*), 499
append() (*rti.connextdds.QosPolicyCountSeq method*), 501
append() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 550
append() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
append() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
append() (*rti.connextdds.ServiceRequestSeq method*), 587
append() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
append() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
append() (*rti.connextdds.StringPairSeq method*), 599
append() (*rti.connextdds.StringSeq method*), 601
append() (*rti.connextdds.SubscriberSeq method*), 609
append() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
append() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
append() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
append() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
append() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
append() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
append() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
append() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
append() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
append() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
append() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
append() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
append() (*rti.connextdds.TopicSeq method*), 720
append() (*rti.connextdds.TransportInfoSeq method*), 730
append() (*rti.connextdds.TransportMulticastSeq method*), 737
append() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
append() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
append() (*rti.connextdds.Uint8Seq method*), 766
append() (*rti.connextdds.Uint16Seq method*), 758
append() (*rti.connextdds.Uint32Seq method*), 761
append() (*rti.connextdds.Uint64Seq method*), 763
append() (*rti.connextdds.UnionMemberSeq method*), 770
append() (*rti.connextdds.WcharSeq method*), 782
append() (*rti.connextdds.WstringSeq method*), 790
append_to_expression_parameter() (*rti.connextdds.ContentFilteredTopic method*), 82
append_to_expression_parameter() (*rti.connextdds.DynamicData.ContentFilteredTopic method*), 193
append_to_expression_parameter() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 390
append_to_expression_parameter() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 447
append_to_expression_parameter() (*rti.connextdds.ServiceRequest.ContentFilteredTopic method*), 548
append_to_expression_parameter() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 611
append_to_expression_parameter() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 668
Application (*class in rti.asyncio*), 810
APPLICATION_AUTO (*rti.connextdds.AcknowledgmentKind attribute*), 38
APPLICATION_AUTO (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 39
APPLICATION_EXPLICIT (*rti.connextdds.AcknowledgmentKind attribute*), 38
APPLICATION_EXPLICIT (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 39

application_kind (*rti.logging.distlog.LoggerOptions property*), 814
application_name (*rti.connextdds.Monitoring property*), 367
APPLICATION_ORDERED (*rti.connextdds.AcknowledgmentKind attribute*), 38
APPLICATION_ORDERED (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 39
ARRAY_TYPE (*rti.connextdds.TypeKind attribute*), 750
ARRAY_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
ArrayType (*class in rti.connextdds*), 54
assert_liveliness() (*rti.connextdds.DataWriter method*), 122
assert_liveliness() (*rti.connextdds.DomainParticipant method*), 165
assert_liveliness() (*rti.connextdds.DynamicData.DataWriter method*), 207
assert_liveliness() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 404
assert_liveliness() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 462
assert_liveliness() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
assert_liveliness() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 625
assert_liveliness() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
assertions_per_lease_duration (*rti.connextdds.Liveliness property*), 340
ASYNCHRONOUS (*rti.connextdds.PublishModeKind attribute*), 492
ASYNCHRONOUS (*rti.connextdds.PublishModeKind.PublishModeKind attribute*), 492
asynchronous() (*rti.connextdds.PublishMode static method*), 491
asynchronous_batch_thread (*rti.connextdds.AsynchronousPublisher property*), 55
asynchronous_publisher (*rti.connextdds.DiscoveryConfig property*), 155
asynchronous_publisher (*rti.connextdds.PublisherQos property*), 497
AsynchronousPublisher (*class in rti.connextdds*), 55
attach_condition() (*rti.connextdds.WaitSet method*), 779
AUTO (*rti.connextdds.CdrPaddingKind attribute*), 65
AUTO (*rti.connextdds.CdrPaddingKind.CdrPaddingKind attribute*), 66
AUTO_COUNT (*rti.connextdds.AllocationSettings attribute*), 45
auto_enable (*rti.connextdds.EntityFactory attribute*), 270
AUTO_ID (*rti.connextdds.DataRepresentation attribute*), 115
AUTO_MAX_TOTAL_INSTANCES (*rti.connextdds.DataReaderResourceLimits attribute*), 110
auto_purge_disposed_samples() (*rti.connextdds.ReaderDataLifecycle static method*), 511
auto_purge_no_writer_samples() (*rti.connextdds.ReaderDataLifecycle static method*), 511
auto_type_coercion (*rti.connextdds.TypeConsistencyEnforcement attribute*), 747
AUTO_TYPE_COERCION (*rti.connextdds.TypeConsistencyEnforcementKind attribute*), 747
AUTO_TYPE_COERCION (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 748
AUTO_WRITER_DEPTH (*rti.connextdds.Durability attribute*), 185
autodispose_unregistered_instances (*rti.connextdds.WriterDataLifecycle property*), 788
autoenable_created_entities (*rti.connextdds.EntityFactory property*), 270
automatic (*rti.connextdds.Duration attribute*), 191
automatic (*rti.connextdds.Guid attribute*), 298
automatic (*rti.connextdds.Liveliness attribute*), 340
AUTOMATIC (*rti.connextdds.LivelinessKind attribute*), 341
AUTOMATIC (*rti.connextdds.LivelinessKind.LivelinessKind attribute*), 342
automatic (*rti.connextdds.SampleIdentity attribute*), 531
automatic (*rti.connextdds.SequenceNumber attribute*), 543
AUTOMATIC (*rti.connextdds.TransportMulticastKind attribute*), 732
AUTOMATIC (*rti.connextdds.TransportMulticastKind.TransportMulticastKind attribute*), 733
autopurge_disposed_instances_delay (*rti.connextdds.ReaderDataLifecycle property*), 511
autopurge_disposed_instances_delay (*rti.connextdds.WriterDataLifecycle property*), 788
autopurge_disposed_samples_delay (*rti.connextdds.ReaderDataLifecycle property*), 511
autopurge_nowriter_instances_delay (*rti.connextdds.ReaderDataLifecycle property*), 511
autopurge_nowriter_samples_delay (*rti.connextdds.ReaderDataLifecycle property*), 511
autopurge_remote_not_alive_writer_delay (*rti.connextdds.DataReaderResourceLimits property*), 110
autopurge_remote_virtual_writer_delay (*rti.connextdds.DataReaderResourceLimits property*), 110
autopurge_unregistered_instances_delay (*rti.connextdds.WriterDataLifecycle property*), 788
autoregister_instances (*rti.connextdds.DataWriterResourceLimits property*), 139
Availability (*class in rti.connextdds*), 56
availability (*rti.connextdds.DataReaderQos property*), 108

availability (*rti.connextdds.DataWriterQos property*), 137
available_builtin_endpoints_ext (*rti.connextdds.ParticipantBuiltinTopicData property*), 425

## B

backwards_compatible (*rti.connextdds.RtpsWellKnownPorts attribute*), 527
banish_ignored_participants() (*rti.connextdds.DomainParticipant method*), 165
baseline (*rti.connextdds.BuiltinProfiles attribute*), 60
baseline_5_0_0 (*rti.connextdds.BuiltinProfiles attribute*), 60
baseline_5_1_0 (*rti.connextdds.BuiltinProfiles attribute*), 60
baseline_5_2_0 (*rti.connextdds.BuiltinProfiles attribute*), 60
baseline_5_3_0 (*rti.connextdds.BuiltinProfiles attribute*), 60
baseline_6_0_0 (*rti.connextdds.BuiltinProfiles attribute*), 60
Batch (*class in rti.connextdds*), 57
batch (*rti.connextdds.DataWriterQos property*), 137
best_effort (*rti.connextdds.Reliability attribute*), 512
BEST_EFFORT (*rti.connextdds.ReliabilityKind attribute*), 513
BEST_EFFORT (*rti.connextdds.ReliabilityKind.ReliabilityKind attribute*), 513
bitset (*rti.connextdds.Member property*), 364
BITSET_TYPE (*rti.connextdds.TypeKind attribute*), 750
BITSET_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
BOOLEAN_TYPE (*rti.connextdds.TypeKind attribute*), 750
BOOLEAN_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
BoolSeq (*class in rti.connextdds*), 58
BoolType (*class in rti.connextdds*), 60
bounds (*rti.connextdds.UnidimensionalCollectionType property*), 767
buffer_alignment (*rti.connextdds.ReceiverPool property*), 512
buffer_size (*rti.connextdds.ReceiverPool property*), 512
builtin_discovery_plugins (*rti.connextdds.DiscoveryConfig property*), 155
BUILTIN_MULTICAST (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
builtin_multicast_port_offset (*rti.connextdds.RtpsWellKnownPorts property*), 527
builtin_subscriber (*rti.connextdds.DomainParticipant property*), 165
builtin_topic_name (*rti.connextdds.PublicationBuiltinTopicData attribute*), 483
builtin_topic_name (*rti.connextdds.SubscriptionBuiltinTopicData attribute*), 647
BUILTIN_UNICAST (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
builtin_unicast_port_offset (*rti.connextdds.RtpsWellKnownPorts property*), 527
BuiltinProfiles (*class in rti.connextdds*), 60
BuiltinTopicKey (*class in rti.connextdds*), 62
BuiltinTopicReaderResourceLimits (*class in rti.connextdds*), 63
BY_RECEPTION_TIMESTAMP (*rti.connextdds.DestinationOrderKind attribute*), 149
BY_RECEPTION_TIMESTAMP (*rti.connextdds.DestinationOrderKind.DestinationOrderKind attribute*), 150
BY_SOURCE_TIMESTAMP (*rti.connextdds.DestinationOrderKind attribute*), 149
BY_SOURCE_TIMESTAMP (*rti.connextdds.DestinationOrderKind.DestinationOrderKind attribute*), 150
ByRef (*class in rti.rpc*), 805
Bytes (*class in rti.types.builtin*), 804
bytes_per_token (*rti.connextdds.FlowControllerTokenBucketProperty property*), 295
ByteVector (*class in rti.connextdds*), 64
BZIP2 (*rti.connextdds.CompressionIdMask attribute*), 75

## C

CANCEL_ASYNCHRONOUS (*rti.connextdds.ThreadSettingsKindMask attribute*), 660
category (*rti.logging.distlog.MessageParams property*), 815
CDR_BIG_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 254
CDR_BIG_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 254
CDR_LITTLE_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 254
CDR_LITTLE_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 254
cdr_padding_kind (*rti.connextdds.TypeSupport property*), 756
cdr_serialized_sample_max_size() (*rti.connextdds.ACTEnumMember method*), 35
cdr_serialized_sample_max_size() (*rti.connextdds.ACTMember method*), 36
cdr_serialized_sample_max_size() (*rti.connextdds.ACTUnionMember method*), 37
cdr_serialized_sample_min_size() (*rti.connextdds.ACTEnumMember method*), 35
cdr_serialized_sample_min_size() (*rti.connextdds.ACTMember method*), 36
cdr_serialized_sample_min_size() (*rti.connextdds.ACTUnionMember method*), 37
CdrPaddingKind (*class in rti.connextdds*), 65
CdrPaddingKind.CdrPaddingKind (*class in rti.connextdds*), 65
change (*rti.connextdds.EventCount32 property*), 278
change (*rti.connextdds.EventCount64 property*), 278
channel_filter_expression_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 179

channel_seq_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 179
channels (*rti.connextdds.MultiChannel property*), 375
ChannelSettings (*class in rti.connextdds*), 67
ChannelSettingsSeq (*class in rti.connextdds*), 68
CHAR_8_TYPE (*rti.connextdds.TypeKind attribute*), 750
CHAR_8_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
CHAR_16_TYPE (*rti.connextdds.TypeKind attribute*), 750
CHAR_16_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
CharSeq (*class in rti.connextdds*), 70
CharType (*class in rti.connextdds*), 72
check_buffer_size (*rti.connextdds.DynamicDataProperty property*), 258
check_crc (*rti.connextdds.WireProtocol property*), 784
class_id (*rti.connextdds.TransportInfo property*), 728
cleanup_period (*rti.connextdds.Database property*), 147
clear() (*rti.connextdds.AnyDataReaderSeq method*), 48
clear() (*rti.connextdds.AnyDataWriterSeq method*), 51
clear() (*rti.connextdds.AnyTopicSeq method*), 54
clear() (*rti.connextdds.BoolSeq method*), 59
clear() (*rti.connextdds.ByteVector method*), 65
clear() (*rti.connextdds.ChannelSettingsSeq method*), 69
clear() (*rti.connextdds.CharSeq method*), 71
clear() (*rti.connextdds.ConditionSeq method*), 80
clear() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
clear() (*rti.connextdds.CookieSeq method*), 87
clear() (*rti.connextdds.CookieVector method*), 88
clear() (*rti.connextdds.DataReaderSeq method*), 114
clear() (*rti.connextdds.DataWriterSeq method*), 146
clear() (*rti.connextdds.DomainParticipantSeq method*), 185
clear() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
clear() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
clear() (*rti.connextdds.DynamicData.DataWriterSeq method*), 215
clear() (*rti.connextdds.DynamicDataSeq method*), 260
clear() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
clear() (*rti.connextdds.DynamicData.TopicSeq method*), 226
clear() (*rti.connextdds.EndpointGroupSeq method*), 267
clear() (*rti.connextdds.EndpointGroupVector method*), 269
clear() (*rti.connextdds.EntitySeq method*), 272
clear() (*rti.connextdds.EnumMemberSeq method*), 275
clear() (*rti.connextdds.Float32Seq method*), 286
clear() (*rti.connextdds.Float64Seq method*), 289
clear() (*rti.connextdds.Float128Seq method*), 284
clear() (*rti.connextdds.IAnyDataReaderSeq method*), 304
clear() (*rti.connextdds.IAnyDataWriterSeq method*), 307
clear() (*rti.connextdds.IAnyTopicSeq method*), 309
clear() (*rti.connextdds.IConditionSeq method*), 312
clear() (*rti.connextdds.IEntitySeq method*), 315
clear() (*rti.connextdds.InstanceHandleSeq method*), 322
clear() (*rti.connextdds.Int8Seq method*), 336
clear() (*rti.connextdds.Int16Seq method*), 327
clear() (*rti.connextdds.Int32Seq method*), 330
clear() (*rti.connextdds.Int32Vector method*), 332
clear() (*rti.connextdds.Int64Seq method*), 334
clear() (*rti.connextdds.LocatorFilterElementSeq method*), 347
clear() (*rti.connextdds.LocatorSeq method*), 353
clear() (*rti.connextdds.LocatorVector method*), 354
clear() (*rti.connextdds.MemberSeq method*), 366
clear() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
clear() (*rti.connextdds.MulticastMappingSeq method*), 378
clear() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
clear() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
clear() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
clear() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
clear() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
clear() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 422
clear() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 450
clear() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
clear() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
clear() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
clear() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
clear() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
clear() (*rti.connextdds.PublisherSeq method*), 499
clear() (*rti.connextdds.QosPolicyCountSeq method*), 501
clear() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 551
clear() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
clear() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
clear() (*rti.connextdds.ServiceRequestSeq method*), 587
clear() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
clear() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
clear() (*rti.connextdds.StringPairSeq method*), 599
clear() (*rti.connextdds.StringSeq method*), 601
clear() (*rti.connextdds.SubscriberSeq method*), 609
clear() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*),

613
clear() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
clear() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
clear() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 650
clear() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
clear() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
clear() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
clear() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
clear() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
clear() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
clear() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
clear() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
clear() (*rti.connextdds.TopicSeq method*), 720
clear() (*rti.connextdds.TransportInfoSeq method*), 730
clear() (*rti.connextdds.TransportInfoVector method*), 732
clear() (*rti.connextdds.TransportMulticastSeq method*), 737
clear() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
clear() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
clear() (*rti.connextdds.Uint8Seq method*), 766
clear() (*rti.connextdds.Uint16Seq method*), 758
clear() (*rti.connextdds.Uint32Seq method*), 761
clear() (*rti.connextdds.Uint64Seq method*), 763
clear() (*rti.connextdds.UnionMemberSeq method*), 770
clear() (*rti.connextdds.WcharSeq method*), 782
clear() (*rti.connextdds.WstringSeq method*), 790
clear_all_members() (*rti.connextdds.DynamicData method*), 230
clear_member() (*rti.connextdds.DynamicData method*), 230
clear_optional_member() (*rti.connextdds.DynamicData method*), 231
ClientBase (*class in rti.rpc*), 805
close() (*rti.connextdds.DataReader method*), 92
close() (*rti.connextdds.DataWriter method*), 122
close() (*rti.connextdds.DynamicData.DataReader method*), 199
close() (*rti.connextdds.DynamicData.DataWriter method*), 207
close() (*rti.connextdds.FlowController method*), 291
close() (*rti.connextdds.IAnyDataReader method*), 301
close() (*rti.connextdds.IAnyDataWriter method*), 304
close() (*rti.connextdds.IAnyTopic method*), 308
close() (*rti.connextdds.IEntity method*), 313
close() (*rti.connextdds.IReadCondition method*), 316
close() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 396
close() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 404
close() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 454
close() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 462
close() (*rti.connextdds.ServiceRequest.DataReader method*), 554
close() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
close() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 617
close() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 625
close() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 674
close() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
close() (*rti.connextdds.TopicQuery method*), 715
close() (*rti.logging.handler.DistlogHandler method*), 815
close() (*rti.rpc.ClientBase method*), 805
close() (*rti.rpc.Service method*), 808
close_contained_entities() (*rti.connextdds.DomainParticipant method*), 165
closed (*rti.connextdds.FlowController property*), 291
closed (*rti.connextdds.IEntity property*), 313
closed (*rti.connextdds.IReadCondition property*), 316
closed (*rti.connextdds.TopicQuery property*), 715
coherent_access (*rti.connextdds.Presentation property*), 436
coherent_set_info (*rti.connextdds.SampleInfo property*), 531
coherent_set_sequence_number (*rti.connextdds.CoherentSetInfo property*), 74
CoherentAccess (*class in rti.connextdds*), 72
CoherentSet (*class in rti.connextdds*), 73
CoherentSetInfo (*class in rti.connextdds*), 73
COLLECTION_TYPE (*rti.connextdds.TypeKind attribute*), 750
COLLECTION_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
CollectionType (*class in rti.connextdds*), 74
collector_initial_peers (*rti.connextdds.MonitoringDedicatedParticipantSettings property*), 367
communication (*rti.connextdds.LogCategory attribute*), 357
communication (*rti.connextdds.LogCategory.LogCategory attribute*), 356
compare() (*rti.connextdds.Duration method*), 191
compare() (*rti.connextdds.Time method*), 664

compile() (*rti.connextdds.DynamicData.ContentFilter method*), 192
compile() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilter method*), 389
compile() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilter method*), 446
compile() (*rti.connextdds.ServiceRequest.ContentFilter method*), 548
compile() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter method*), 610
compile() (*rti.connextdds.TopicBuiltinTopicData.ContentFilter method*), 667
COMPRESSED (*rti.connextdds.heap_monitoring.SnapshotOutputFormat attribute*), 801
COMPRESSED (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat attribute*), 801
compressed_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 96
compression_ids (*rti.connextdds.CompressionSettings property*), 78
COMPRESSION_LEVEL_BEST_COMPRESSION (*rti.connextdds.CompressionSettings attribute*), 77
COMPRESSION_LEVEL_BEST_SPEED (*rti.connextdds.CompressionSettings attribute*), 77
COMPRESSION_LEVEL_DEFAULT (*rti.connextdds.CompressionSettings attribute*), 78
compression_settings (*rti.connextdds.DataRepresentation property*), 116
CompressionIdMask (*class in rti.connextdds*), 74
CompressionSettings (*class in rti.connextdds*), 77
compute_crc (*rti.connextdds.WireProtocol property*), 784
concurrency_level (*rti.connextdds.MonitoringEventDistributionSettings property*), 369
concurrency_level (*rti.connextdds.MonitoringLoggingDistributionSettings property*), 369
Condition (*class in rti.connextdds*), 78
condition() (*rti.connextdds.DataReader.Selector method*), 90
condition() (*rti.connextdds.DynamicData.DataReader.Selector method*), 196
condition() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
condition() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 450
condition() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 551
condition() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
condition() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
conditions (*rti.connextdds.WaitSet property*), 779
ConditionSeq (*class in rti.connextdds*), 78
CONSTRUCTED_TYPE (*rti.connextdds.TypeKind attribute*), 750
CONSTRUCTED_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
contains_entity() (*rti.connextdds.DomainParticipant method*), 165
content() (*rti.connextdds.DataReader.Selector method*), 90
content() (*rti.connextdds.DynamicData.DataReader.Selector method*), 196
content() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
content() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
content() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
content() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
content() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
content_filter_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 179
content_filter_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 96
content_filter_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 179
content_filter_property (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
content_filter_topic_name (*rti.connextdds.ContentFilterProperty property*), 81
content_filtered_topic_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 179
content_filtered_topic_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 179
content_type (*rti.connextdds.CollectionType property*), 74
contentfilter_property_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 179
ContentFilterBase (*class in rti.connextdds*), 81
ContentFilteredTopic (*class in rti.connextdds*), 81
ContentFilteredTopicSeq (*class in rti.connextdds*), 82
ContentFilterProperty (*class in rti.connextdds*), 81
CONTINUOUS (*rti.connextdds.TopicQuerySelectionKind attribute*), 717
CONTINUOUS (*rti.connextdds.TopicQuerySelectionKind.Topic-*

*QuerySelectionKind attribute*), 717
Cookie (*class in rti.connextdds*), 85
cookie (*rti.connextdds.AcknowledgmentInfo property*), 38
cookie (*rti.connextdds.WriteParams property*), 787
cookie_max_length (*rti.connextdds.DataWriterResourceLimits property*), 139
CookieSeq (*class in rti.connextdds*), 85
CookieVector (*class in rti.connextdds*), 87
corrupted_rtps_message_count (*rti.connextdds.DomainParticipantProtocolStatus property*), 174
corrupted_rtps_message_count_change (*rti.connextdds.DomainParticipantProtocolStatus property*), 174
count (*rti.connextdds.ActivityContextMask property*), 43
count (*rti.connextdds.compliance.RpcMask property*), 793
count (*rti.connextdds.compliance.XTypesMask property*), 796
count (*rti.connextdds.CompressionIdMask property*), 76
count (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask property*), 160
count (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask property*), 163
count (*rti.connextdds.InstanceState property*), 324
count (*rti.connextdds.QosPolicyCount property*), 500
count (*rti.connextdds.RtpsReservedPortKindMask property*), 525
count (*rti.connextdds.SampleFlag property*), 529
count (*rti.connextdds.SampleLostState property*), 535
count (*rti.connextdds.SampleRejectedState property*), 538
count (*rti.connextdds.SampleState property*), 541
count (*rti.connextdds.StatusMask property*), 594
count (*rti.connextdds.StreamKind property*), 596
count (*rti.connextdds.ThreadSettingsKindMask property*), 662
count (*rti.connextdds.TransportBuiltinMask property*), 724
count (*rti.connextdds.ViewState property*), 777
count() (*rti.connextdds.AnyDataReaderSeq method*), 48
count() (*rti.connextdds.AnyDataWriterSeq method*), 51
count() (*rti.connextdds.AnyTopicSeq method*), 54
count() (*rti.connextdds.BoolSeq method*), 59
count() (*rti.connextdds.ChannelSettingsSeq method*), 69
count() (*rti.connextdds.CharSeq method*), 72
count() (*rti.connextdds.ConditionSeq method*), 80
count() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
count() (*rti.connextdds.CookieSeq method*), 87
count() (*rti.connextdds.DataReaderSeq method*), 115
count() (*rti.connextdds.DataWriterSeq method*), 146
count() (*rti.connextdds.DomainParticipantSeq method*), 185
count() (*rti.connextdds.DynamicData method*), 231
count() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
count() (*rti.connextdds.DynamicData.DataReaderSeq method*), 204
count() (*rti.connextdds.DynamicData.DataWriterSeq method*), 216
count() (*rti.connextdds.DynamicDataSeq method*), 260
count() (*rti.connextdds.DynamicDataTimestampedSeq method*), 262
count() (*rti.connextdds.DynamicData.TopicSeq method*), 226
count() (*rti.connextdds.EndpointGroupSeq method*), 267
count() (*rti.connextdds.EntitySeq method*), 272
count() (*rti.connextdds.EnumMemberSeq method*), 275
count() (*rti.connextdds.Float32Seq method*), 286
count() (*rti.connextdds.Float64Seq method*), 289
count() (*rti.connextdds.Float128Seq method*), 284
count() (*rti.connextdds.IAnyDataReaderSeq method*), 304
count() (*rti.connextdds.IAnyDataWriterSeq method*), 307
count() (*rti.connextdds.IAnyTopicSeq method*), 309
count() (*rti.connextdds.IConditionSeq method*), 312
count() (*rti.connextdds.IEntitySeq method*), 315
count() (*rti.connextdds.InstanceHandleSeq method*), 322
count() (*rti.connextdds.Int8Seq method*), 336
count() (*rti.connextdds.Int16Seq method*), 327
count() (*rti.connextdds.Int32Seq method*), 330
count() (*rti.connextdds.Int64Seq method*), 334
count() (*rti.connextdds.LocatorFilterElementSeq method*), 348
count() (*rti.connextdds.LocatorSeq method*), 353
count() (*rti.connextdds.MemberSeq method*), 366
count() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
count() (*rti.connextdds.MulticastMappingSeq method*), 378
count() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
count() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
count() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
count() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
count() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
count() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 423
count() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 450
count() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
count() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
count() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
count() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
count() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 480
count() (*rti.connextdds.PublisherSeq method*), 499
count() (*rti.connextdds.QosPolicyCountSeq method*), 501
count() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*),

551
count() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
count() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
count() (*rti.connextdds.ServiceRequestSeq method*), 587
count() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
count() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
count() (*rti.connextdds.StringPairSeq method*), 599
count() (*rti.connextdds.StringSeq method*), 601
count() (*rti.connextdds.SubscriberSeq method*), 609
count() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
count() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
count() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
count() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 651
count() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
count() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
count() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
count() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 679
count() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
count() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
count() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
count() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
count() (*rti.connextdds.TopicSeq method*), 720
count() (*rti.connextdds.TransportInfoSeq method*), 730
count() (*rti.connextdds.TransportMulticastSeq method*), 737
count() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
count() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
count() (*rti.connextdds.Uint8Seq method*), 766
count() (*rti.connextdds.Uint16Seq method*), 758
count() (*rti.connextdds.Uint32Seq method*), 761
count() (*rti.connextdds.Uint64Seq method*), 763
count() (*rti.connextdds.UnionMemberSeq method*), 770
count() (*rti.connextdds.WcharSeq method*), 782
count() (*rti.connextdds.WstringSeq method*), 790
cpu_list (*rti.connextdds.ThreadSettings property*), 658
cpu_rotation (*rti.connextdds.ThreadSettings property*), 658
create_data() (*rti.connextdds.DynamicData.DataWriter method*), 207
create_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 404
create_data() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 462
create_data() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
create_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 625
create_data() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
create_from_service_request() (*rti.connextdds.TopicQueryData static method*), 715
create_participant_from_config() (*rti.connextdds.QosProvider method*), 503
CRITICAL (*rti.connextdds.SyslogVerbosity attribute*), 655
critical() (*rti.connextdds.Logger method*), 361
current (*rti.connextdds.ProductVersion attribute*), 444
current (*rti.connextdds.ProtocolVersion attribute*), 446
current_count (*rti.connextdds.PublicationMatchedStatus property*), 490
current_count (*rti.connextdds.ServiceRequestAcceptedStatus property*), 583
current_count (*rti.connextdds.SubscriptionMatchedStatus property*), 654
current_count_change (*rti.connextdds.PublicationMatchedStatus property*), 490
current_count_change (*rti.connextdds.SubscriptionMatchedStatus property*), 654
current_count_peak (*rti.connextdds.PublicationMatchedStatus property*), 490
current_count_peak (*rti.connextdds.SubscriptionMatchedStatus property*), 654
current_time (*rti.connextdds.DomainParticipant property*), 166

## D

data (*rti.connextdds.DataReader.LoanedSample property*), 89
data (*rti.connextdds.DynamicData.LoanedSample property*), 218
data (*rti.connextdds.DynamicData.Sample property*), 222
data (*rti.connextdds.LoanedDynamicData property*), 344
data (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample property*), 414
data (*rti.connextdds.ParticipantBuiltinTopicData.Sample property*), 419
data (*rti.connextdds.PublicationBuiltinTopicData.LoanedSample property*), 472
data (*rti.connextdds.PublicationBuiltinTopicData.Sample property*), 476
data (*rti.connextdds.ServiceRequest.LoanedSample property*), 572

data (*rti.connextdds.ServiceRequest.Sample property*), 576
data (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample property*), 636
data (*rti.connextdds.SubscriptionBuiltinTopicData.Sample property*), 640
data (*rti.connextdds.TopicBuiltinTopicData.LoanedSample property*), 692
data (*rti.connextdds.TopicBuiltinTopicData.Sample property*), 696
DATA_AVAILABLE (*rti.connextdds.StatusMask attribute*), 591
DATA_ON_READERS (*rti.connextdds.StatusMask attribute*), 591
data_reader (*rti.connextdds.IReadCondition property*), 316
data_reader (*rti.connextdds.Query property*), 507
data_reader_protocol (*rti.connextdds.DataReaderQos property*), 108
data_reader_resource_limits (*rti.connextdds.DataReaderQos property*), 108
data_representation (*rti.connextdds.DataReaderQos property*), 108
data_representation (*rti.connextdds.DataWriterQos property*), 137
data_representation (*rti.connextdds.TopicQos property*), 713
data_state (*rti.connextdds.DataStateEx property*), 118
data_tag (*rti.connextdds.DataReaderQos property*), 108
data_tag (*rti.connextdds.DataWriterQos property*), 137
data_tag (*rti.connextdds.PublicationBuiltinTopicData property*), 483
data_tag (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
data_writer_protocol (*rti.connextdds.DataWriterQos property*), 137
data_writer_resource_limits (*rti.connextdds.DataWriterQos property*), 137
data_writer_transfer_mode (*rti.connextdds.DataWriterQos property*), 137
Database (*class in rti.connextdds*), 147
database (*rti.connextdds.DomainParticipantQos property*), 177
database (*rti.connextdds.LogCategory attribute*), 357
database (*rti.connextdds.LogCategory.LogCategory attribute*), 356
DATABASE_INTEGRATION (*rti.connextdds.ServiceKind attribute*), 545
DATABASE_INTEGRATION (*rti.connextdds.ServiceKind.ServiceKind attribute*), 545
DataReader (*class in rti.connextdds*), 88
datareader (*rti.connextdds.TopicQuery property*), 715
DATAREADER_CACHE (*rti.connextdds.StatusMask attribute*), 591
datareader_cache_status (*rti.connextdds.DataReader property*), 93
datareader_cache_status (*rti.connextdds.DynamicData.DataReader property*), 199
datareader_cache_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 396
datareader_cache_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 454
datareader_cache_status (*rti.connextdds.ServiceRequest.DataReader property*), 554
datareader_cache_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 617
datareader_cache_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 674
DATAREADER_PROTOCOL (*rti.connextdds.StatusMask attribute*), 591
datareader_protocol_status (*rti.connextdds.DataReader property*), 93
datareader_protocol_status (*rti.connextdds.DynamicData.DataReader property*), 199
datareader_protocol_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 396
datareader_protocol_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 454
datareader_protocol_status (*rti.connextdds.ServiceRequest.DataReader property*), 554
datareader_protocol_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 617
datareader_protocol_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 674
datareader_qos (*rti.connextdds.QosProvider property*), 503
datareader_qos_from_profile() (*rti.connextdds.QosProvider method*), 504
DataReaderCacheStatus (*class in rti.connextdds*), 95
DataReaderInstanceRemovalKind (*class in rti.connextdds*), 97
DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind (*class in rti.connextdds*), 97
DataReaderListener (*class in rti.connextdds*), 100
DataReader.LoanedSample (*class in rti.connextdds*), 88
DataReader.LoanedSamples (*class in rti.connextdds*), 89
DataReaderProtocol (*class in rti.connextdds*), 100
DataReaderProtocolStatus (*class in rti.connextdds*), 101
DataReaderQos (*class in rti.connextdds*), 103
DataReaderResourceLimits (*class in rti.connextdds*), 110
DataReaderResourceLimitsInstanceReplacementSettings (*class in rti.connextdds*), 112
DataReader.Selector (*class in rti.connextdds*), 89
DataReaderSeq (*class in rti.connextdds*), 113
DataRepresentation (*class in rti.connextdds*), 115

DataState (*class in rti.connextdds*), 116
DataStateEx (*class in rti.connextdds*), 117
DataTag (*class in rti.connextdds*), 119
DataWriter (*class in rti.connextdds*), 120
DATAWRITER_APPLICATION_ACKNOWLEDGMENT (*rti.connextdds.StatusMask attribute*), 591
DATAWRITER_CACHE (*rti.connextdds.StatusMask attribute*), 591
datawriter_cache_status (*rti.connextdds.DataWriter property*), 122
datawriter_cache_status (*rti.connextdds.DynamicData.DataWriter property*), 207
datawriter_cache_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 404
datawriter_cache_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 462
datawriter_cache_status (*rti.connextdds.ServiceRequest.DataWriter property*), 562
datawriter_cache_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 625
datawriter_cache_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 682
DATAWRITER_INSTANCE_REPLACED (*rti.connextdds.StatusMask attribute*), 591
DATAWRITER_PROTOCOL (*rti.connextdds.StatusMask attribute*), 591
datawriter_protocol_status (*rti.connextdds.DataWriter property*), 122
datawriter_protocol_status (*rti.connextdds.DynamicData.DataWriter property*), 207
datawriter_protocol_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 404
datawriter_protocol_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 462
datawriter_protocol_status (*rti.connextdds.ServiceRequest.DataWriter property*), 562
datawriter_protocol_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 625
datawriter_protocol_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 682
datawriter_qos (*rti.connextdds.QosProvider property*), 504
datawriter_qos_from_profile() (*rti.connextdds.QosProvider method*), 504
datawriter_qos_profile_name (*rti.connextdds.MonitoringEventDistributionSettings property*), 369
datawriter_qos_profile_name (*rti.connextdds.MonitoringLoggingDistributionSettings property*), 370
datawriter_qos_profile_name (*rti.connextdds.MonitoringPeriodicDistributionSettings property*), 374
DataWriterCacheStatus (*class in rti.connextdds*), 126
DataWriterListener (*class in rti.connextdds*), 127
DataWriterProtocol (*class in rti.connextdds*), 128
DataWriterProtocolStatus (*class in rti.connextdds*), 129
DataWriterQos (*class in rti.connextdds*), 131
DataWriterResourceLimits (*class in rti.connextdds*), 139
DataWriterResourceLimitsInstanceReplacementKind (*class in rti.connextdds*), 140
DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind (*class in rti.connextdds*), 141
DataWriterSeq (*class in rti.connextdds*), 144
DataWriterShmemRefTransferModeSettings (*class in rti.connextdds*), 146
DataWriterTransferMode (*class in rti.connextdds*), 147
dds_builtin_endpoints (*rti.connextdds.ParticipantBuiltinTopicData property*), 425
Deadline (*class in rti.connextdds*), 148
deadline (*rti.connextdds.DataReaderQos property*), 108
deadline (*rti.connextdds.DataWriterQos property*), 137
deadline (*rti.connextdds.PublicationBuiltinTopicData property*), 483
deadline (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
deadline (*rti.connextdds.TopicBuiltinTopicData property*), 703
deadline (*rti.connextdds.TopicQos property*), 713
DEBUG (*rti.connextdds.PrintFormat attribute*), 438
DEBUG (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
DEBUG (*rti.connextdds.SyslogVerbosity attribute*), 655
DEBUG (*rti.logging.distlog.LogLevel attribute*), 813
debug() (*rti.connextdds.Logger method*), 361
debug() (*rti.logging.distlog.Logger static method*), 813
dedicated_participant (*rti.connextdds.MonitoringDistributionSettings property*), 368
DEFAULT (*rti.connextdds.ActivityContextMask attribute*), 41
DEFAULT (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 254
DEFAULT (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 254
DEFAULT (*rti.connextdds.DynamicDataTypeSerializationProperty attribute*), 263
DEFAULT (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 798
DEFAULT (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*),

799
DEFAULT (*rti.connextdds.PrintFormat attribute*), 439
DEFAULT (*rti.connextdds.PrintFormatKind attribute*), 441
DEFAULT (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*), 441
DEFAULT (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
default (*rti.connextdds.PrintFormatProperty attribute*), 443
default (*rti.connextdds.QosProvider attribute*), 504
default_datareader_qos (*rti.connextdds.DomainParticipant property*), 166
default_datareader_qos (*rti.connextdds.Subscriber property*), 604
default_datawriter_qos (*rti.connextdds.DomainParticipant property*), 166
default_datawriter_qos (*rti.connextdds.Publisher property*), 494
default_domain_announcement_period (*rti.connextdds.DiscoveryConfig property*), 155
default_filter_state (*rti.connextdds.DataReader property*), 93
default_filter_state (*rti.connextdds.DynamicData.DataReader property*), 199
default_filter_state (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 396
default_filter_state (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 454
default_filter_state (*rti.connextdds.ServiceRequest.DataReader property*), 554
default_filter_state (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 617
default_filter_state (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 674
DEFAULT_LABEL (*rti.connextdds.UnionMember attribute*), 767
default_library (*rti.connextdds.QosProvider property*), 504
DEFAULT_MASK (*rti.connextdds.compliance.RpcMask attribute*), 791
DEFAULT_MASK (*rti.connextdds.compliance.XTypesMask attribute*), 794
DEFAULT_MASK (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
DEFAULT_NAME (*rti.connextdds.FlowController attribute*), 290
default_participant_qos (*rti.connextdds.DomainParticipant attribute*), 166
default_profile (*rti.connextdds.QosProvider property*), 504
default_profile_library (*rti.connextdds.QosProvider property*), 504
default_provider_params (*rti.connextdds.QosProvider attribute*), 504
DEFAULT_PUBLICATION (*rti.connextdds.CompressionIdMask attribute*), 75
default_publisher_qos (*rti.connextdds.DomainParticipant property*), 166
default_subscriber_qos (*rti.connextdds.DomainParticipant property*), 166
DEFAULT_SUBSCRIPTION (*rti.connextdds.CompressionIdMask attribute*), 75
default_topic_qos (*rti.connextdds.DomainParticipant property*), 166
default_unicast (*rti.connextdds.DomainParticipantQos property*), 177
default_unicast_locators (*rti.connextdds.ParticipantBuiltinTopicData property*), 425
DELETE (*rti.connextdds.PersistentJournalKind attribute*), 432
delete_durable_subscription() (*rti.connextdds.DomainParticipant method*), 166
depth (*rti.connextdds.History property*), 299
deserialized_type_object_dynamic_allocation_threshold (*rti.connextdds.DomainParticipantResourceLimits property*), 179
destination_order (*rti.connextdds.DataReaderQos property*), 108
destination_order (*rti.connextdds.DataWriterQos property*), 137
destination_order (*rti.connextdds.PublicationBuiltinTopicData property*), 483
destination_order (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
destination_order (*rti.connextdds.TopicBuiltinTopicData property*), 703
destination_order (*rti.connextdds.TopicQos property*), 713
DESTINATION_UNREACHABLE (*rti.connextdds.StatusMask attribute*), 591
DestinationOrder (*class in rti.connextdds*), 148
DestinationOrderKind (*class in rti.connextdds*), 149
DestinationOrderKind.DestinationOrderKind (*class in rti.connextdds*), 149
DestinationOrderScopeKind (*class in rti.connextdds*), 151
DestinationOrderScopeKind.DestinationOrderScopeKind (*class in rti.connextdds*), 151
detach_all() (*rti.connextdds.WaitSet method*), 779
detach_condition() (*rti.connextdds.WaitSet method*), 779
detached_instance_count (*rti.connextdds.DataReaderCacheStatus property*), 96
detached_instance_count_peak (*rti.connextdds.DataReaderCacheStatus property*), 96
DHEADER_IN_NON_PRIMITIVE_COLLECTIONS_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
dimension() (*rti.connextdds.ArrayType method*), 55
dimension_count (*rti.connextdds.ArrayType property*), 55
direct_communication (*rti.connextdds.Durability property*), 186

disable() (*in module rti.connextdds.heap_monitoring*), 803
disable_asynchronous_batch (*rti.connextdds.AsynchronousPublisher property*), 56
disable_asynchronous_write (*rti.connextdds.AsynchronousPublisher property*), 56
disable_fragmentation_support (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 63
disable_fragmentation_support (*rti.connextdds.DataReaderResourceLimits property*), 110
disable_inline_keyhash (*rti.connextdds.DataWriterProtocol property*), 128
disable_positive_acks (*rti.connextdds.DataReaderProtocol property*), 101
disable_positive_acks (*rti.connextdds.DataWriterProtocol property*), 128
disable_positive_acks (*rti.connextdds.PublicationBuiltinTopicData property*), 483
disable_positive_acks (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
disable_positive_acks_decrease_sample_keep_duration_factor (*rti.connextdds.RtpsReliableWriterProtocol property*), 521
disable_positive_acks_enable_adaptive_sample_keep_duration (*rti.connextdds.RtpsReliableWriterProtocol property*), 521
disable_positive_acks_increase_sample_keep_duration_factor (*rti.connextdds.RtpsReliableWriterProtocol property*), 521
disable_positive_acks_max_sample_keep_duration (*rti.connextdds.RtpsReliableWriterProtocol property*), 521
disable_positive_acks_min_sample_keep_duration (*rti.connextdds.RtpsReliableWriterProtocol property*), 521
disable_repair_piggyback_heartbeat (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
disable_topic_query_publication (*rti.connextdds.AsynchronousPublisher property*), 56
disabled (*rti.connextdds.AsynchronousPublisher attribute*), 56
disabled (*rti.connextdds.Batch attribute*), 57
disabled (*rti.connextdds.Monitoring attribute*), 367
disabled_metrics_selection (*rti.connextdds.MonitoringMetricSelection property*), 371
disallow_type_coercion (*rti.connextdds.TypeConsistencyEnforcement attribute*), 747
DISALLOW_TYPE_COERCION (*rti.connextdds.TypeConsistencyEnforcementKind attribute*), 747
DISALLOW_TYPE_COERCION (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 748
discovered_participant_data() (*rti.connextdds.DomainParticipant method*), 166
discovered_participant_subject_name() (*rti.connextdds.DomainParticipant method*), 166
discovered_participants() (*rti.connextdds.DomainParticipant method*), 167
discovered_participants_from_subject_name() (*rti.connextdds.DomainParticipant method*), 167
discovered_topic_data() (*rti.connextdds.DomainParticipant method*), 167
discovered_topics() (*rti.connextdds.DomainParticipant method*), 167
Discovery (*class in rti.connextdds*), 154
discovery (*rti.connextdds.DomainParticipantQos property*), 177
discovery (*rti.connextdds.LogCategory attribute*), 357
discovery (*rti.connextdds.LogCategory.LogCategory attribute*), 356
discovery_config (*rti.connextdds.DomainParticipantQos property*), 177
DiscoveryConfig (*class in rti.connextdds*), 154
DiscoveryConfigBuiltinChannelKindMask (*class in rti.connextdds*), 158
DiscoveryConfigBuiltinPluginKindMask (*class in rti.connextdds*), 161
discriminator (*rti.connextdds.DynamicData property*), 231
discriminator (*rti.connextdds.UnionType property*), 771
discriminator_value (*rti.connextdds.DynamicData property*), 231
dispatch() (*rti.connextdds.ICondition method*), 310
dispatch() (*rti.connextdds.StatusCondition method*), 591
dispatch() (*rti.connextdds.WaitSet method*), 779
dispatch_async() (*rti.connextdds.WaitSet method*), 779
dispose_instance() (*rti.connextdds.DataWriter method*), 122
dispose_instance() (*rti.connextdds.DynamicData.DataWriter method*), 207
dispose_instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 404
dispose_instance() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 462
dispose_instance() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
dispose_instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 625
dispose_instance() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
dispose_instance_async() (*rti.connextdds.DynamicData.DataWriter method*), 207

dispose_instance_async() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 404
dispose_instance_async() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 462
dispose_instance_async() (*rti.connextdds.ServiceRequest.DataWriter method*), 562
dispose_instance_async() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 626
dispose_instance_async() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 682
DISPOSED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 141
DISPOSED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142
disposed (*rti.connextdds.GenerationCount property*), 296
disposed_instance_count (*rti.connextdds.DataReaderCacheStatus property*), 96
disposed_instance_count (*rti.connextdds.DataWriterCacheStatus property*), 127
disposed_instance_count_peak (*rti.connextdds.DataReaderCacheStatus property*), 96
disposed_instance_count_peak (*rti.connextdds.DataWriterCacheStatus property*), 127
disposed_instance_removal (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings property*), 113
DISPOSED_THEN_ALIVE (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 141
DISPOSED_THEN_ALIVE (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142
DistlogHandler (*class in rti.logging.handler*), 815
distribution_settings (*rti.connextdds.Monitoring property*), 367
dll (*rti.connextdds.TransportMulticastMappingFunction property*), 735
dns_tracker_polling_period (*rti.connextdds.DiscoveryConfig property*), 155
domain_entity_qos_library_name (*rti.connextdds.DomainParticipantConfigParams property*), 171
domain_entity_qos_profile_name (*rti.connextdds.DomainParticipantConfigParams property*), 171
DOMAIN_ID (*rti.connextdds.ActivityContextMask attribute*), 41
domain_id (*rti.connextdds.DomainParticipant property*), 167
domain_id (*rti.connextdds.DomainParticipantConfigParams property*), 171
domain_id (*rti.connextdds.MonitoringDedicatedParticipantSettings property*), 367
domain_id (*rti.connextdds.ParticipantBuiltinTopicData property*), 425
domain_id (*rti.logging.distlog.LoggerOptions property*), 814
domain_id_gain (*rti.connextdds.RtpsWellKnownPorts property*), 527
DOMAIN_ID_USE_XML_CONFIG (*rti.connextdds.DomainParticipantConfigParams attribute*), 171
DomainParticipant (*class in rti.connextdds*), 164
DomainParticipantConfigParams (*class in rti.connextdds*), 170
DomainParticipantFactoryQos (*class in rti.connextdds*), 172
DomainParticipantListener (*class in rti.connextdds*), 173
DomainParticipantProtocolStatus (*class in rti.connextdds*), 173
DomainParticipantQos (*class in rti.connextdds*), 174
DomainParticipantResourceLimits (*class in rti.connextdds*), 178
DomainParticipantSeq (*class in rti.connextdds*), 183
DoubleSeq (*in module rti.connextdds*), 185
DoubleType (*in module rti.connextdds*), 185
drop_incomplete_coherent_set (*rti.connextdds.Presentation property*), 436
dropped_fragment_count (*rti.connextdds.DataReaderProtocolStatus property*), 101
duplicate_sample_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 102
duplicate_sample_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
Durability (*class in rti.connextdds*), 185
durability (*rti.connextdds.DataReaderQos property*), 108
durability (*rti.connextdds.DataWriterQos property*), 137
durability (*rti.connextdds.PublicationBuiltinTopicData property*), 483
durability (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
durability (*rti.connextdds.TopicBuiltinTopicData property*), 703
durability (*rti.connextdds.TopicQos property*), 713
durability_service (*rti.connextdds.DataWriterQos property*), 137
durability_service (*rti.connextdds.PublicationBuiltinTopicData property*), 483
durability_service (*rti.connextdds.TopicBuiltinTopicData property*), 703
durability_service (*rti.connextdds.TopicQos property*), 714
DurabilityKind (*class in rti.connextdds*), 186
DurabilityKind.DurabilityKind (*class in rti.connextdds*), 186
DurabilityService (*class in rti.connextdds*), 189

Duration (*class in rti.connextdds*), 190
duration (*rti.connextdds.LatencyBudget property*), 339
duration (*rti.connextdds.Lifespan property*), 340
dynamically_allocate_fragmented_samples (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 63
dynamically_allocate_fragmented_samples (*rti.connextdds.DataReaderResourceLimits property*), 110
DynamicData (*class in rti.connextdds*), 192
DynamicData.ContentFilter (*class in rti.connextdds*), 192
DynamicData.ContentFilteredTopic (*class in rti.connextdds*), 192
DynamicData.ContentFilteredTopicSeq (*class in rti.connextdds*), 194
DynamicData.DataReader (*class in rti.connextdds*), 196
DynamicData.DataReaderListener (*class in rti.connextdds*), 202
DynamicData.DataReader.Selector (*class in rti.connextdds*), 196
DynamicData.DataReaderSeq (*class in rti.connextdds*), 203
DynamicData.DataWriter (*class in rti.connextdds*), 205
DynamicData.DataWriterListener (*class in rti.connextdds*), 213
DynamicData.DataWriterSeq (*class in rti.connextdds*), 214
DynamicDataEncapsulationKind (*class in rti.connextdds*), 254
DynamicDataEncapsulationKind.DynamicDataEncapsulationKind (*class in rti.connextdds*), 254
DynamicData.FieldsIterator (*class in rti.connextdds*), 216
DynamicData.FieldsView (*class in rti.connextdds*), 216
DynamicData.IndexIterator (*class in rti.connextdds*), 217
DynamicDataInfo (*class in rti.connextdds*), 256
DynamicData.ItemsIterator (*class in rti.connextdds*), 217
DynamicData.ItemsView (*class in rti.connextdds*), 218
DynamicData.ITopicDescription (*class in rti.connextdds*), 217
DynamicData.LoanedSample (*class in rti.connextdds*), 218
DynamicData.LoanedSamples (*class in rti.connextdds*), 219
DynamicDataMemberInfo (*class in rti.connextdds*), 257
DynamicData.NoOpDataReaderListener (*class in rti.connextdds*), 219
DynamicData.NoOpDataWriterListener (*class in rti.connextdds*), 220
DynamicData.NoOpTopicListener (*class in rti.connextdds*), 222
DynamicDataProperty (*class in rti.connextdds*), 258
DynamicData.Sample (*class in rti.connextdds*), 222
DynamicDataSeq (*class in rti.connextdds*), 258
DynamicData.SharedSamples (*class in rti.connextdds*), 222
DynamicDataTimestampedSeq (*class in rti.connextdds*), 261
DynamicData.Topic (*class in rti.connextdds*), 223
DynamicData.TopicDescription (*class in rti.connextdds*), 224
DynamicData.TopicListener (*class in rti.connextdds*), 224
DynamicData.TopicSeq (*class in rti.connextdds*), 225
DynamicData.TopicTypeSupport (*class in rti.connextdds*), 227
DynamicDataTypeSerializationProperty (*class in rti.connextdds*), 263
DynamicData.ValidLoanedSamples (*class in rti.connextdds*), 227
DynamicData.WriterContentFilter (*class in rti.connextdds*), 228
DynamicData.WriterContentFilterHelper (*class in rti.connextdds*), 228
DynamicType (*class in rti.connextdds*), 264
DynamicTypePrintFormatProperty (*class in rti.connextdds*), 265

## E

EARLIEST_DEADLINE_FIRST (*rti.connextdds.FlowControllerSchedulingPolicy attribute*), 292
EARLIEST_DEADLINE_FIRST (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 293
echo_to_stdout (*rti.logging.distlog.LoggerOptions property*), 814
element_count (*rti.connextdds.DynamicDataMemberInfo property*), 257
element_kind (*rti.connextdds.DynamicDataMemberInfo property*), 257
EMERGENCY (*rti.connextdds.SyslogVerbosity attribute*), 655
emergency() (*rti.connextdds.Logger method*), 361
emit() (*rti.logging.handler.DistlogHandler method*), 815
EMPTY_INSTANCES (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 98
EMPTY_INSTANCES (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 97
empty_reliable_writer_cache (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 515
enable (*rti.connextdds.Batch property*), 57
enable (*rti.connextdds.Monitoring property*), 367
enable (*rti.connextdds.MonitoringDedicatedParticipantSettings property*), 368
enable (*rti.connextdds.PersistentStorageSettings property*), 433
enable (*rti.connextdds.TopicQueryDispatch property*), 716
enable() (*in module rti.connextdds.heap_monitoring*), 803
enable() (*rti.connextdds.IEntity method*), 313

enable_automatic_tss_cleanup() (*in module rti.connextdds*), 790
enable_data_consistency_check (*rti.connextdds.DataWriterShmemRefTransferModeSettings property*), 147
enable_endpoint_discovery (*rti.connextdds.Discovery property*), 154
enable_multicast_periodic_heartbeat (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
enable_required_subscriptions (*rti.connextdds.Availability property*), 56
enabled (*rti.connextdds.Batch attribute*), 57
enabled (*rti.connextdds.IEntity property*), 313
enabled (*rti.connextdds.Monitoring attribute*), 367
enabled() (*rti.connextdds.AsynchronousPublisher static method*), 56
enabled_builtin_channels (*rti.connextdds.DiscoveryConfig property*), 155
enabled_metrics_selection (*rti.connextdds.MonitoringMetricSelection property*), 371
enabled_statuses (*rti.connextdds.StatusCondition property*), 591
enabled_transports (*rti.connextdds.Discovery property*), 154
enabled_with_max_data_bytes() (*rti.connextdds.Batch static method*), 57
enabled_with_max_samples() (*rti.connextdds.Batch static method*), 57
encapsulation_id (*rti.connextdds.SampleInfo property*), 531
encapsulation_kind (*rti.connextdds.DynamicDataInfo property*), 257
ENCAPSULATION_OPTIONS_WITH_PADDING_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
end() (*rti.connextdds.CoherentAccess method*), 73
end() (*rti.connextdds.CoherentSet method*), 73
endpoint_type_object_lb_serialization_threshold (*rti.connextdds.DiscoveryConfig property*), 155
EndpointGroup (*class in rti.connextdds*), 266
EndpointGroupSeq (*class in rti.connextdds*), 266
EndpointGroupVector (*class in rti.connextdds*), 268
entities (*rti.connextdds.LogCategory attribute*), 357
entities (*rti.connextdds.LogCategory.LogCategory attribute*), 356
Entity (*class in rti.connextdds*), 269
entity (*rti.connextdds.StatusCondition property*), 591
entity_factory (*rti.connextdds.DomainParticipantFactoryQos property*), 173
entity_factory (*rti.connextdds.DomainParticipantQos property*), 177
entity_factory (*rti.connextdds.PublisherQos property*), 497
entity_factory (*rti.connextdds.SubscriberQos property*), 607
ENTITY_KIND (*rti.connextdds.ActivityContextMask attribute*), 41
ENTITY_NAME (*rti.connextdds.ActivityContextMask attribute*), 41
entity_name (*rti.connextdds.DataReaderQos property*), 108
entity_name (*rti.connextdds.DataWriterQos property*), 137
entity_name (*rti.connextdds.PublisherQos property*), 497
entity_name (*rti.connextdds.SubscriberQos property*), 607
ENTITY_NAME_USE_XML_CONFIG (*rti.connextdds.DomainParticipantConfigParams attribute*), 171
EntityFactory (*class in rti.connextdds*), 269
EntityLock (*class in rti.connextdds*), 270
EntityName (*class in rti.connextdds*), 270
EntitySeq (*class in rti.connextdds*), 271
enum_as_int (*rti.connextdds.PrintFormatProperty property*), 443
ENUM_AS_PRIMITIVE_IN_COLLECTIONS_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
ENUMERATION_TYPE (*rti.connextdds.TypeKind attribute*), 750
ENUMERATION_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
EnumMember (*class in rti.connextdds*), 273
EnumMemberSeq (*class in rti.connextdds*), 274
EnumType (*class in rti.connextdds*), 276
Error, 277
ERROR (*rti.connextdds.SyslogVerbosity attribute*), 655
ERROR (*rti.logging.distlog.LogLevel attribute*), 813
error() (*rti.connextdds.Logger method*), 361
error() (*rti.logging.distlog.Logger static method*), 813
evaluate() (*rti.connextdds.DynamicData.ContentFilter method*), 192
evaluate() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilter method*), 389
evaluate() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilter method*), 447
evaluate() (*rti.connextdds.ServiceRequest.ContentFilter method*), 548
evaluate() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter method*), 610
evaluate() (*rti.connextdds.TopicBuiltinTopicData.ContentFilter method*), 667
Event (*class in rti.connextdds*), 277
event (*rti.connextdds.DomainParticipantQos property*), 177
event_count (*rti.connextdds.WaitSetProperty property*), 781
event_delay (*rti.connextdds.WaitSetProperty property*), 781
event_settings (*rti.connextdds.MonitoringDistributionSettings property*), 368
EventCount32 (*class in rti.connextdds*), 277
EventCount64 (*class in rti.connextdds*), 278
exact_reliability_kind_match (*rti.connextdds.Reliability property*), 512
Exception, 278

EXCEPTION (*rti.connextdds.LogLevel attribute*), 358
EXCEPTION (*rti.connextdds.LogLevel.LogLevel attribute*), 359
EXCEPTION (*rti.connextdds.Verbosity attribute*), 773
EXCEPTION (*rti.connextdds.Verbosity.Verbosity attribute*), 774
exclusive (*rti.connextdds.Ownership attribute*), 386
EXCLUSIVE (*rti.connextdds.OwnershipKind attribute*), 386
EXCLUSIVE (*rti.connextdds.OwnershipKind.OwnershipKind attribute*), 386
exists() (*rti.connextdds.DataTag method*), 120
exists() (*rti.connextdds.Property method*), 445
expects_inline_qos (*rti.connextdds.DataReaderProtocol property*), 101
expiration_code (*rti.connextdds.InvalidLocalAdvanceNoticeStatus property*), 338
expiration_message (*rti.connextdds.InvalidLocalAdvanceNoticeStatus property*), 338
expiration_time (*rti.connextdds.InvalidLocalAdvanceNoticeStatus property*), 338
expired_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 96
expression (*rti.connextdds.Filter property*), 281
expression (*rti.connextdds.Query property*), 507
expression (*rti.connextdds.QueryCondition property*), 508
ExpressionProperty (*class in rti.connextdds*), 278
extend() (*rti.connextdds.AnyDataReaderSeq method*), 48
extend() (*rti.connextdds.AnyDataWriterSeq method*), 51
extend() (*rti.connextdds.AnyTopicSeq method*), 54
extend() (*rti.connextdds.BoolSeq method*), 59
extend() (*rti.connextdds.ChannelSettingsSeq method*), 69
extend() (*rti.connextdds.CharSeq method*), 72
extend() (*rti.connextdds.ConditionSeq method*), 80
extend() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
extend() (*rti.connextdds.CookieSeq method*), 87
extend() (*rti.connextdds.DataReaderSeq method*), 115
extend() (*rti.connextdds.DataWriterSeq method*), 146
extend() (*rti.connextdds.DomainParticipantSeq method*), 185
extend() (*rti.connextdds.DynamicData method*), 231
extend() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
extend() (*rti.connextdds.DynamicData.DataReaderSeq method*), 205
extend() (*rti.connextdds.DynamicData.DataWriterSeq method*), 216
extend() (*rti.connextdds.DynamicDataSeq method*), 260
extend() (*rti.connextdds.DynamicDataTimestampedSeq method*), 263
extend() (*rti.connextdds.DynamicData.TopicSeq method*), 226
extend() (*rti.connextdds.EndpointGroupSeq method*), 268
extend() (*rti.connextdds.EntitySeq method*), 272
extend() (*rti.connextdds.EnumMemberSeq method*), 275
extend() (*rti.connextdds.Float32Seq method*), 286
extend() (*rti.connextdds.Float64Seq method*), 289
extend() (*rti.connextdds.Float128Seq method*), 284
extend() (*rti.connextdds.IAnyDataReaderSeq method*), 304
extend() (*rti.connextdds.IAnyDataWriterSeq method*), 307
extend() (*rti.connextdds.IAnyTopicSeq method*), 309
extend() (*rti.connextdds.IConditionSeq method*), 312
extend() (*rti.connextdds.IEntitySeq method*), 315
extend() (*rti.connextdds.InstanceHandleSeq method*), 322
extend() (*rti.connextdds.Int8Seq method*), 336
extend() (*rti.connextdds.Int16Seq method*), 328
extend() (*rti.connextdds.Int32Seq method*), 330
extend() (*rti.connextdds.Int64Seq method*), 334
extend() (*rti.connextdds.LocatorFilterElementSeq method*), 348
extend() (*rti.connextdds.LocatorSeq method*), 353
extend() (*rti.connextdds.MemberSeq method*), 366
extend() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
extend() (*rti.connextdds.MulticastMappingSeq method*), 378
extend() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
extend() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 401
extend() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
extend() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
extend() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 430
extend() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 423
extend() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 450
extend() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
extend() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
extend() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
extend() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 489
extend() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 481
extend() (*rti.connextdds.PublisherSeq method*), 499
extend() (*rti.connextdds.QosPolicyCountSeq method*), 501
extend() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 551
extend() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
extend() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
extend() (*rti.connextdds.ServiceRequestSeq method*), 587

extend() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 589
extend() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
extend() (*rti.connextdds.StringPairSeq method*), 599
extend() (*rti.connextdds.StringSeq method*), 601
extend() (*rti.connextdds.SubscriberSeq method*), 609
extend() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
extend() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
extend() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
extend() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 651
extend() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
extend() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
extend() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
extend() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 680
extend() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
extend() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
extend() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 708
extend() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 700
extend() (*rti.connextdds.TopicSeq method*), 721
extend() (*rti.connextdds.TransportInfoSeq method*), 730
extend() (*rti.connextdds.TransportMulticastSeq method*), 737
extend() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
extend() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
extend() (*rti.connextdds.Uint8Seq method*), 766
extend() (*rti.connextdds.Uint16Seq method*), 758
extend() (*rti.connextdds.Uint32Seq method*), 761
extend() (*rti.connextdds.Uint64Seq method*), 763
extend() (*rti.connextdds.UnionMemberSeq method*), 770
extend() (*rti.connextdds.WcharSeq method*), 782
extend() (*rti.connextdds.WstringSeq method*), 790
extensibility_kind (*rti.connextdds.EnumType property*), 276
extensibility_kind (*rti.connextdds.StructType property*), 604
extensibility_kind (*rti.connextdds.UnionType property*), 771
extensibility_kind() (*rti.connextdds.ACTEnumMember method*), 35
extensibility_kind() (*rti.connextdds.ACTMember method*), 36
extensibility_kind() (*rti.connextdds.ACTUnionMember method*), 37
ExtensibilityKind (*class in rti.connextdds*), 279
ExtensibilityKind.ExtensibilityKind (*class in rti.connextdds*), 279
EXTENSIBLE (*rti.connextdds.ExtensibilityKind attribute*), 279
EXTENSIBLE (*rti.connextdds.ExtensibilityKind.ExtensibilityKind attribute*), 279

## F

facility (*rti.connextdds.LogMessage property*), 360
fast_heartbeat_period (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
FATAL (*rti.logging.distlog.LogLevel attribute*), 813
fatal() (*rti.logging.distlog.Logger static method*), 813
FATAL_ERROR (*rti.connextdds.LogLevel attribute*), 358
FATAL_ERROR (*rti.connextdds.LogLevel.LogLevel attribute*), 359
fields() (*rti.connextdds.DynamicData method*), 231
file_name (*rti.connextdds.PersistentStorageSettings property*), 433
Filter (*class in rti.connextdds*), 281
filter (*rti.connextdds.TopicQuerySelection property*), 717
filter_class_name (*rti.connextdds.ContentFilterProperty property*), 81
filter_expression (*rti.connextdds.ChannelSettings property*), 68
filter_expression (*rti.connextdds.ContentFilteredTopic property*), 82
filter_expression (*rti.connextdds.ContentFilterProperty property*), 81
filter_expression (*rti.connextdds.DynamicData.ContentFilteredTopic property*), 193
filter_expression (*rti.connextdds.LocatorFilterElement property*), 346
filter_expression (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic property*), 390
filter_expression (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic property*), 447
filter_expression (*rti.connextdds.ServiceRequest.ContentFilteredTopic property*), 548
filter_expression (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic property*), 611
filter_expression (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic property*), 668

filter_level (*rti.logging.distlog.LoggerOptions property*), 814
filter_level() (*rti.logging.distlog.Logger static method*), 813
filter_name (*rti.connextdds.LocatorFilter property*), 345
filter_name (*rti.connextdds.MultiChannel property*), 375
filter_parameters (*rti.connextdds.ContentFilteredTopic property*), 82
filter_parameters (*rti.connextdds.DynamicData.ContentFilteredTopic property*), 193
filter_parameters (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic property*), 390
filter_parameters (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic property*), 447
filter_parameters (*rti.connextdds.ServiceRequest.ContentFilteredTopic property*), 549
filter_parameters (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic property*), 611
filter_parameters (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic property*), 668
FilterSampleInfo (*class in rti.connextdds*), 281
FINAL (*rti.connextdds.ExtensibilityKind attribute*), 280
FINAL (*rti.connextdds.ExtensibilityKind.ExtensibilityKind attribute*), 279
finalize() (*rti.connextdds.DynamicData.ContentFilter method*), 192
finalize() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilter method*), 389
finalize() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilter method*), 447
finalize() (*rti.connextdds.ServiceRequest.ContentFilter method*), 548
finalize() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilter method*), 610
finalize() (*rti.connextdds.TopicBuiltinTopicData.ContentFilter method*), 667
finalize() (*rti.logging.distlog.Logger static method*), 813
finalize_participant_factory() (*rti.connextdds.DomainParticipant static method*), 167
find() (*rti.connextdds.ContentFilteredTopic static method*), 82
find() (*rti.connextdds.DomainParticipant static method*), 167
find() (*rti.connextdds.DynamicData.ContentFilteredTopic static method*), 193
find() (*rti.connextdds.DynamicData.Topic static method*), 224
find() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic static method*), 390
find() (*rti.connextdds.ParticipantBuiltinTopicData.Topic static method*), 420
find() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic static method*), 448
find() (*rti.connextdds.PublicationBuiltinTopicData.Topic static method*), 478
find() (*rti.connextdds.ServiceRequest.ContentFilteredTopic static method*), 549
find() (*rti.connextdds.ServiceRequest.Topic static method*), 577
find() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic static method*), 611
find() (*rti.connextdds.SubscriptionBuiltinTopicData.Topic static method*), 641
find() (*rti.connextdds.Topic static method*), 666
find() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic static method*), 668
find() (*rti.connextdds.TopicBuiltinTopicData.Topic static method*), 697
find() (*rti.connextdds.TopicQuery static method*), 715
find_all_by_topic() (*rti.connextdds.DataReader static method*), 93
find_all_by_topic() (*rti.connextdds.DataWriter static method*), 122
find_all_by_topic() (*rti.connextdds.DynamicData.DataReader static method*), 199
find_all_by_topic() (*rti.connextdds.DynamicData.DataWriter static method*), 207
find_all_by_topic() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader static method*), 396
find_all_by_topic() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter static method*), 405
find_all_by_topic() (*rti.connextdds.PublicationBuiltinTopicData.DataReader static method*), 454
find_all_by_topic() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter static method*), 462
find_all_by_topic() (*rti.connextdds.ServiceRequest.DataReader static method*), 555
find_all_by_topic() (*rti.connextdds.ServiceRequest.DataWriter static method*), 563
find_all_by_topic() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader static method*), 617
find_all_by_topic() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter static method*), 626
find_all_by_topic() (*rti.connextdds.TopicBuiltinTopicData.DataReader static method*), 674
find_all_by_topic() (*rti.connextdds.TopicBuiltinTopicData.DataWriter*

*static method*), 683
find_by_name() (*rti.connextdds.DataReader static method*), 93
find_by_name() (*rti.connextdds.DataWriter static method*), 122
find_by_name() (*rti.connextdds.DynamicData.DataReader static method*), 199
find_by_name() (*rti.connextdds.DynamicData.DataWriter static method*), 208
find_by_name() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader static method*), 396
find_by_name() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter static method*), 405
find_by_name() (*rti.connextdds.PublicationBuiltinTopicData.DataReader static method*), 454
find_by_name() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter static method*), 463
find_by_name() (*rti.connextdds.ServiceRequest.DataReader static method*), 555
find_by_name() (*rti.connextdds.ServiceRequest.DataWriter static method*), 563
find_by_name() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader static method*), 617
find_by_name() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter static method*), 626
find_by_name() (*rti.connextdds.TopicBuiltinTopicData.DataReader static method*), 674
find_by_name() (*rti.connextdds.TopicBuiltinTopicData.DataWriter static method*), 683
find_by_topic() (*rti.connextdds.DataReader static method*), 93
find_by_topic() (*rti.connextdds.DataWriter static method*), 123
find_by_topic() (*rti.connextdds.DynamicData.DataReader static method*), 199
find_by_topic() (*rti.connextdds.DynamicData.DataWriter static method*), 208
find_by_topic() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader static method*), 396
find_by_topic() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter static method*), 405
find_by_topic() (*rti.connextdds.PublicationBuiltinTopicData.DataReader static method*), 454
find_by_topic() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter static method*), 463
find_by_topic() (*rti.connextdds.ServiceRequest.DataReader static method*), 555
find_by_topic() (*rti.connextdds.ServiceRequest.DataWriter static method*), 563
find_by_topic() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader static method*), 618
find_by_topic() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter static method*), 626
find_by_topic() (*rti.connextdds.TopicBuiltinTopicData.DataReader static method*), 674
find_by_topic() (*rti.connextdds.TopicBuiltinTopicData.DataWriter static method*), 683
find_contentfilter() (*rti.connextdds.DomainParticipant method*), 168
find_datareader() (*rti.connextdds.DomainParticipant method*), 168
find_datareader() (*rti.connextdds.Subscriber method*), 605
find_datareader_by_topic_name() (*rti.connextdds.Subscriber method*), 605
find_datareaders() (*rti.connextdds.Subscriber method*), 605
find_datawriter() (*rti.connextdds.DomainParticipant method*), 168
find_datawriter() (*rti.connextdds.Publisher method*), 494
find_datawriter_by_topic_name() (*rti.connextdds.Publisher method*), 494
find_datawriters() (*rti.connextdds.Publisher method*), 495
find_flow_controller() (*rti.connextdds.DomainParticipant method*), 168
find_member_by_id() (*rti.connextdds.StructType method*), 604
find_member_by_id() (*rti.connextdds.UnionType method*), 771
find_member_by_label() (*rti.connextdds.UnionType method*), 771
find_member_by_name() (*rti.connextdds.ACTEnumMember method*), 35
find_member_by_name() (*rti.connextdds.ACTMember method*), 36
find_member_by_name() (*rti.connextdds.ACTUnionMember method*), 37
find_member_by_ordinal() (*rti.connextdds.EnumType method*), 277
find_publisher() (*rti.connextdds.DomainParticipant method*), 168
find_publishers() (*rti.connextdds.DomainParticipant method*), 168
find_registered_content_filters() (*rti.connextdds.DomainParticipant method*), 168
find_subscriber() (*rti.connextdds.DomainParticipant method*), 168
find_subscribers() (*rti.connextdds.DomainParticipant method*), 168

find_topics() (*rti.connextdds.DomainParticipant method*), 168
first_available_sample_sequence_number (*rti.connextdds.DataReaderProtocolStatus property*), 102
first_available_sample_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 129
first_available_sample_virtual_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 129
first_unacknowledged_sample_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 129
first_unacknowledged_sample_subscription_handle (*rti.connextdds.DataWriterProtocolStatus property*), 129
first_unacknowledged_sample_virtual_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 129
first_unelapsed_keep_duration_sample_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 130
FIXED_RATE_NAME (*rti.connextdds.FlowController attribute*), 290
flag (*rti.connextdds.SampleInfo property*), 531
flag (*rti.connextdds.WriteParams property*), 787
flip() (*rti.connextdds.ActivityContextMask method*), 43
flip() (*rti.connextdds.compliance.RpcMask method*), 793
flip() (*rti.connextdds.compliance.XTypesMask method*), 796
flip() (*rti.connextdds.CompressionIdMask method*), 76
flip() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 160
flip() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
flip() (*rti.connextdds.InstanceState method*), 324
flip() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
flip() (*rti.connextdds.SampleFlag method*), 529
flip() (*rti.connextdds.SampleLostState method*), 535
flip() (*rti.connextdds.SampleRejectedState method*), 538
flip() (*rti.connextdds.SampleState method*), 541
flip() (*rti.connextdds.StatusMask method*), 594
flip() (*rti.connextdds.StreamKind method*), 596
flip() (*rti.connextdds.ThreadSettingsKindMask method*), 662
flip() (*rti.connextdds.TransportBuiltinMask method*), 724
flip() (*rti.connextdds.ViewState method*), 777
Float32Seq (*class in rti.connextdds*), 285
Float32Type (*class in rti.connextdds*), 287
Float64Seq (*class in rti.connextdds*), 287
Float64Type (*class in rti.connextdds*), 290
Float128Seq (*class in rti.connextdds*), 282
Float128Type (*class in rti.connextdds*), 284
FLOAT_32_TYPE (*rti.connextdds.TypeKind attribute*), 750
FLOAT_32_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
FLOAT_64_TYPE (*rti.connextdds.TypeKind attribute*), 750
FLOAT_64_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
FLOAT_128_TYPE (*rti.connextdds.TypeKind attribute*), 750
FLOAT_128_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 752
FLOATING_POINT (*rti.connextdds.ThreadSettingsKindMask attribute*), 660
FloatSeq (*in module rti.connextdds*), 290
FloatType (*in module rti.connextdds*), 290
flow_controller_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 179
flow_controller_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 179
flow_controller_name (*rti.connextdds.PublishMode property*), 491
FlowController (*class in rti.connextdds*), 290
FlowControllerProperty (*class in rti.connextdds*), 291
FlowControllerSchedulingPolicy (*class in rti.connextdds*), 292
FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy (*class in rti.connextdds*), 292
FlowControllerTokenBucketProperty (*class in rti.connextdds*), 295
flush() (*rti.connextdds.DataWriter method*), 123
flush() (*rti.connextdds.DynamicData.DataWriter method*), 208
flush() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 405
flush() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
flush() (*rti.connextdds.ServiceRequest.DataWriter method*), 563
flush() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 626
flush() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 683
force_type_validation (*rti.connextdds.TypeConsistencyEnforcement property*), 747
from_cdr_buffer() (*rti.connextdds.DynamicData method*), 231
from_cdr_buffer() (*rti.connextdds.DynamicData.TopicTypeSupport static method*), 227
from_json() (*rti.connextdds.DynamicData method*), 231
from_microseconds() (*rti.connextdds.Duration static method*), 191
from_microseconds() (*rti.connextdds.Time static method*), 664
from_milliseconds() (*rti.connextdds.Duration static*

*method*), 191
from_milliseconds() (*rti.connextdds.Time static method*), 664
from_seconds() (*rti.connextdds.Duration static method*), 191
from_seconds() (*rti.connextdds.Time static method*), 664
from_string() (*rti.connextdds.DynamicData method*), 231
FULL (*rti.connextdds.PersistentSynchronizationKind attribute*), 434
full_reliable_writer_cache (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 515
FULLY_PROCESSED_INSTANCES (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 98
FULLY_PROCESSED_INSTANCES (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 97
FUNCTION (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 798
FUNCTION (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*), 799
function_name (*rti.connextdds.TransportMulticastMappingFunction property*), 735

## G

generation (*rti.connextdds.Rank property*), 509
generation_count (*rti.connextdds.SampleInfo property*), 531
GenerationCount (*class in rti.connextdds*), 296
generic_auto_tuning (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_best_effort (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_common (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_connext_micro_compatibility (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_connext_micro_compatibility_2_4_3 (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_connext_micro_compatibility_2_4_9 (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_large_data (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_large_data_fast_flow (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_large_data_medium_flow (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_large_data_slow_flow (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_persistent (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_transient (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_keep_last_reliable_transient_local (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_minimal_memory_footprint (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_monitoring2 (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_monitoring_common (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_other_dds_vendor_compatibility (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_participant_large_data (*rti.connextdds.BuiltinProfiles attribute*), 61
generic_participant_large_data_monitoring (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_security (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable_high_throughput (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable_large_data (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable_large_data_fast_flow (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable_large_data_medium_flow (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable_large_data_slow_flow (*rti.connextdds.BuiltinProfiles attribute*), 62
generic_strict_reliable_low_latency (*rti.connextdds.BuiltinProfiles attribute*), 62
get() (*rti.connextdds.DataTag method*), 120
get() (*rti.connextdds.Property method*), 445
get_all() (*rti.connextdds.DataTag method*), 120
get_all() (*rti.connextdds.Property method*), 445
get_boolean() (*rti.connextdds.DynamicData method*), 231
get_cdr_buffer() (*rti.connextdds.DynamicData method*), 232
get_char() (*rti.connextdds.DynamicData method*), 232
get_char_values() (*rti.connextdds.DynamicData method*), 232
get_complex() (*rti.connextdds.DynamicData method*), 232
get_complex_values() (*rti.connextdds.DynamicData method*), 232
get_double() (*rti.connextdds.DynamicData method*), 232
get_double_values() (*rti.connextdds.DynamicData method*), 233
get_float() (*rti.connextdds.DynamicData method*), 233
get_float32() (*rti.connextdds.DynamicData method*), 233
get_float32_values() (*rti.connextdds.DynamicData method*), 233
get_float64() (*rti.connextdds.DynamicData method*), 234

get_float64_values() (*rti.connextdds.DynamicData method*), 234
get_float128() (*rti.connextdds.DynamicData method*), 233
get_float_values() (*rti.connextdds.DynamicData method*), 234
get_int() (*rti.connextdds.DynamicData method*), 234
get_int8() (*rti.connextdds.DynamicData method*), 236
get_int8_values() (*rti.connextdds.DynamicData method*), 236
get_int16() (*rti.connextdds.DynamicData method*), 234
get_int16_values() (*rti.connextdds.DynamicData method*), 235
get_int32() (*rti.connextdds.DynamicData method*), 235
get_int32_values() (*rti.connextdds.DynamicData method*), 235
get_int64() (*rti.connextdds.DynamicData method*), 235
get_int64_values() (*rti.connextdds.DynamicData method*), 235
get_int_values() (*rti.connextdds.DynamicData method*), 236
get_interface_types() (*in module rti.rpc*), 810
get_long() (*rti.connextdds.DynamicData method*), 236
get_long_values() (*rti.connextdds.DynamicData method*), 236
get_longdouble() (*rti.connextdds.DynamicData method*), 236
get_longlong() (*rti.connextdds.DynamicData method*), 237
get_longlong_values() (*rti.connextdds.DynamicData method*), 237
get_octet() (*rti.connextdds.DynamicData method*), 237
get_octet_values() (*rti.connextdds.DynamicData method*), 237
get_rpc_mask() (*in module rti.connextdds.compliance*), 797
get_short() (*rti.connextdds.DynamicData method*), 237
get_short_values() (*rti.connextdds.DynamicData method*), 238
get_string() (*rti.connextdds.DynamicData method*), 238
get_topic_datareader_qos() (*rti.connextdds.QosProvider method*), 504
get_topic_datawriter_qos() (*rti.connextdds.QosProvider method*), 504
get_topic_name_qos() (*rti.connextdds.QosProvider method*), 504
get_uint() (*rti.connextdds.DynamicData method*), 238
get_uint8() (*rti.connextdds.DynamicData method*), 239
get_uint8_values() (*rti.connextdds.DynamicData method*), 239
get_uint16() (*rti.connextdds.DynamicData method*), 238
get_uint16_values() (*rti.connextdds.DynamicData method*), 238
get_uint32() (*rti.connextdds.DynamicData method*), 239
get_uint32_values() (*rti.connextdds.DynamicData method*), 239
get_uint64() (*rti.connextdds.DynamicData method*), 239
get_uint64_values() (*rti.connextdds.DynamicData method*), 239
get_uint_values() (*rti.connextdds.DynamicData method*), 240
get_ulong() (*rti.connextdds.DynamicData method*), 240
get_ulong_values() (*rti.connextdds.DynamicData method*), 240
get_ulonglong() (*rti.connextdds.DynamicData method*), 240
get_ulonglong_values() (*rti.connextdds.DynamicData method*), 240
get_ushort() (*rti.connextdds.DynamicData method*), 241
get_ushort_values() (*rti.connextdds.DynamicData method*), 241
get_value() (*rti.connextdds.DynamicData method*), 241
get_values() (*rti.connextdds.DynamicData method*), 241
get_wchar() (*rti.connextdds.DynamicData method*), 241
get_wstring() (*rti.connextdds.DynamicData method*), 242
get_xtypes_mask() (*in module rti.connextdds.compliance*), 797
GROUP (*rti.connextdds.PresentationAccessScopeKind attribute*), 436
GROUP (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 436
group_access_scope() (*rti.connextdds.Presentation static method*), 436
group_coherent_set_sequence_number (*rti.connextdds.CoherentSetInfo property*), 74
group_data (*rti.connextdds.PublicationBuiltinTopicData property*), 483
group_data (*rti.connextdds.PublisherQos property*), 497
group_data (*rti.connextdds.SubscriberQos property*), 607
group_data (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
group_guid (*rti.connextdds.CoherentSetInfo property*), 74
GroupData (*class in rti.connextdds*), 296
GuardCondition (*class in rti.connextdds*), 297
Guid (*class in rti.connextdds*), 297
guid (*rti.connextdds.TopicQuery property*), 715
GUID_PREFIX (*rti.connextdds.ActivityContextMask attribute*), 41

## H

handle (*rti.connextdds.WriteParams property*), 787
has_id (*rti.connextdds.UnionMember property*), 768
has_parent (*rti.connextdds.StructType property*), 604
HeapMonitoringParams (*class in rti.connextdds.heap_monitoring*), 798
heartbeat_period (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
heartbeat_suppression_duration (*rti.connextdds.RtpsReliableReaderProtocol property*), 520
heartbeats_per_max_samples (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
high_watermark (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
high_watermark_reliable_writer_cache (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 515

HIGHEST_OFFERED (*rti.connextdds.PresentationAccessScopeKind attribute*), 436
HIGHEST_OFFERED (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 436
HIGHEST_PRIORITY_FIRST (*rti.connextdds.FlowControllerSchedulingPolicy attribute*), 294
HIGHEST_PRIORITY_FIRST (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 293
History (*class in rti.connextdds*), 298
history (*rti.connextdds.DataReaderQos property*), 108
history (*rti.connextdds.DataWriterQos property*), 138
history (*rti.connextdds.TopicBuiltinTopicData property*), 703
history (*rti.connextdds.TopicQos property*), 714
history_depth (*rti.connextdds.DurabilityService property*), 189
history_kind (*rti.connextdds.DurabilityService property*), 189
HISTORY_SNAPSHOT (*rti.connextdds.TopicQuerySelectionKind attribute*), 717
HISTORY_SNAPSHOT (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute*), 717
HistoryKind (*class in rti.connextdds*), 299
HistoryKind.HistoryKind (*class in rti.connextdds*), 299

## I

IAnyDataReader (*class in rti.connextdds*), 301
IAnyDataReaderSeq (*class in rti.connextdds*), 302
IAnyDataWriter (*class in rti.connextdds*), 304
IAnyDataWriterSeq (*class in rti.connextdds*), 305
IAnyTopic (*class in rti.connextdds*), 307
IAnyTopicSeq (*class in rti.connextdds*), 308
ICondition (*class in rti.connextdds*), 310
IConditionSeq (*class in rti.connextdds*), 310
id (*rti.connextdds.Member property*), 364
id (*rti.connextdds.UnionMember property*), 768
IDataReader (*class in rti.connextdds*), 313
identity (*rti.connextdds.WriteParams property*), 787
IEntity (*class in rti.connextdds*), 313
IEntitySeq (*class in rti.connextdds*), 314
ignore_datareader() (*rti.connextdds.DomainParticipant method*), 168
ignore_datareaders() (*rti.connextdds.DomainParticipant method*), 168
ignore_datawriter() (*rti.connextdds.DomainParticipant method*), 169
ignore_datawriters() (*rti.connextdds.DomainParticipant method*), 169
ignore_default_domain_announcements (*rti.connextdds.DiscoveryConfig property*), 155
ignore_enum_literal_names (*rti.connextdds.TypeConsistencyEnforcement property*), 747
ignore_environment_profile (*rti.connextdds.QosProviderParams property*), 506
ignore_member_names (*rti.connextdds.TypeConsistencyEnforcement property*), 747
ignore_participant() (*rti.connextdds.DomainParticipant method*), 169
ignore_participants() (*rti.connextdds.DomainParticipant method*), 169
ignore_resource_profile (*rti.connextdds.QosProviderParams property*), 506
ignore_sequence_bounds (*rti.connextdds.TypeConsistencyEnforcement property*), 747
ignore_string_bounds (*rti.connextdds.TypeConsistencyEnforcement property*), 747
ignore_topic() (*rti.connextdds.DomainParticipant method*), 169
ignore_topics() (*rti.connextdds.DomainParticipant method*), 169
ignore_user_profile (*rti.connextdds.QosProviderParams property*), 507
ignored_entity_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 179
ignored_entity_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 179
ignored_entity_replacement_kind (*rti.connextdds.DomainParticipantResourceLimits property*), 179
IgnoredEntityReplacementKind (*class in rti.connextdds*), 317
IgnoredEntityReplacementKind.IgnoredEntityReplacementKind (*class in rti.connextdds*), 317
IllegalOperationError, 319
ImmutablePolicyError, 319
implicit_publisher (*rti.connextdds.DomainParticipant property*), 169
implicit_subscriber (*rti.connextdds.DomainParticipant property*), 169
inactivate_nonprogressing_readers (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
inactive_count (*rti.connextdds.ReliableReaderActivityChangedStatus property*), 515
include_root_elements (*rti.connextdds.PrintFormatProperty property*), 443
incomplete_coherent_set (*rti.connextdds.CoherentSetInfo property*), 74
INCOMPLETE_SNAPSHOT_TOPIC_QUERY (*rti.connextdds.SampleFlag attribute*), 528
INCONSISTENT_TOPIC (*rti.connextdds.StatusMask attribute*), 591
inconsistent_topic_status (*rti.connextdds.DynamicData.Topic property*), 224

inconsistent_topic_status (*rti.connextdds.ParticipantBuiltinTopicData.Topic property*), 420
inconsistent_topic_status (*rti.connextdds.PublicationBuiltinTopicData.Topic property*), 478
inconsistent_topic_status (*rti.connextdds.ServiceRequest.Topic property*), 577
inconsistent_topic_status (*rti.connextdds.SubscriptionBuiltinTopicData.Topic property*), 641
inconsistent_topic_status (*rti.connextdds.Topic property*), 666
inconsistent_topic_status (*rti.connextdds.TopicBuiltinTopicData.Topic property*), 698
InconsistentPolicyError, 319
InconsistentTopicStatus (*class in rti.connextdds*), 319
incremental_count (*rti.connextdds.AllocationSettings property*), 45
indent (*rti.connextdds.DynamicTypePrintFormatProperty property*), 265
indent (*rti.connextdds.QosPrintFormat property*), 502
index (*rti.connextdds.DynamicDataMemberInfo property*), 258
infinite (*rti.connextdds.Duration attribute*), 192
info (*rti.connextdds.DataReader.LoanedSample property*), 89
info (*rti.connextdds.DynamicData property*), 242
info (*rti.connextdds.DynamicData.LoanedSample property*), 218
info (*rti.connextdds.DynamicData.Sample property*), 222
info (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSample property*), 414
info (*rti.connextdds.ParticipantBuiltinTopicData.Sample property*), 419
info (*rti.connextdds.PublicationBuiltinTopicData.LoanedSample property*), 472
info (*rti.connextdds.PublicationBuiltinTopicData.Sample property*), 476
info (*rti.connextdds.ServiceRequest.LoanedSample property*), 572
info (*rti.connextdds.ServiceRequest.Sample property*), 576
info (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample property*), 636
info (*rti.connextdds.SubscriptionBuiltinTopicData.Sample property*), 640
info (*rti.connextdds.TopicBuiltinTopicData.LoanedSample property*), 692
info (*rti.connextdds.TopicBuiltinTopicData.Sample property*), 696
INFO (*rti.logging.distlog.LogLevel attribute*), 813
info() (*rti.logging.distlog.Logger static method*), 813
INFORMATIONAL (*rti.connextdds.SyslogVerbosity attribute*), 655
informational() (*rti.connextdds.Logger method*), 361
init() (*rti.logging.distlog.Logger static method*), 813
initial_active_topic_queries (*rti.connextdds.DataWriterResourceLimits property*), 139
initial_batches (*rti.connextdds.DataWriterResourceLimits property*), 139
initial_buffer_size (*rti.connextdds.DynamicDataProperty property*), 258
initial_concurrent_blocking_threads (*rti.connextdds.DataWriterResourceLimits property*), 139
initial_count (*rti.connextdds.AllocationSettings property*), 45
initial_count (*rti.connextdds.Event property*), 277
initial_fragmented_samples (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 63
initial_fragmented_samples (*rti.connextdds.DataReaderResourceLimits property*), 110
initial_infos (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 63
initial_infos (*rti.connextdds.DataReaderResourceLimits property*), 110
initial_instances (*rti.connextdds.ResourceLimits property*), 520
initial_objects_per_thread (*rti.connextdds.SystemResourceLimits property*), 657
initial_outstanding_reads (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 63
initial_outstanding_reads (*rti.connextdds.DataReaderResourceLimits property*), 111
initial_participant_announcements (*rti.connextdds.DiscoveryConfig property*), 155
initial_peers (*rti.connextdds.Discovery property*), 154
initial_records (*rti.connextdds.Database property*), 147
initial_remote_readers (*rti.connextdds.DataWriterResourceLimits property*), 140
initial_remote_virtual_writers (*rti.connextdds.DataReaderResourceLimits property*), 111
initial_remote_virtual_writers_per_instance (*rti.connextdds.DataReaderResourceLimits property*), 111
initial_remote_writers (*rti.connextdds.DataReaderResourceLimits property*), 111
initial_remote_writers_per_instance (*rti.connextdds.DataReaderResourceLimits property*), 111
initial_samples (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
initial_samples (*rti.connextdds.ResourceLimits property*), 520
initial_topic_queries (*rti.connextdds.DataReaderResourceLimits property*), 111
initial_virtual_sequence_number

(*rti.connextdds.DataWriterProtocol property*), 129
initial_virtual_writers (*rti.connextdds.DataWriterResourceLimits property*), 140
initial_weak_references (*rti.connextdds.Database property*), 148
INITIALIZE_DISCRIMINATOR_TO_DEFAULT_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
initialize_sample() (*rti.connextdds.DynamicData.TopicTypeSupport static method*), 227
initialize_writer_loaned_sample (*rti.connextdds.DataWriterResourceLimits property*), 140
insert() (*rti.connextdds.AnyDataReaderSeq method*), 48
insert() (*rti.connextdds.AnyDataWriterSeq method*), 52
insert() (*rti.connextdds.AnyTopicSeq method*), 54
insert() (*rti.connextdds.BoolSeq method*), 60
insert() (*rti.connextdds.ChannelSettingsSeq method*), 70
insert() (*rti.connextdds.CharSeq method*), 72
insert() (*rti.connextdds.ConditionSeq method*), 80
insert() (*rti.connextdds.ContentFilteredTopicSeq method*), 84
insert() (*rti.connextdds.CookieSeq method*), 87
insert() (*rti.connextdds.DataReaderSeq method*), 115
insert() (*rti.connextdds.DataWriterSeq method*), 146
insert() (*rti.connextdds.DomainParticipantSeq method*), 185
insert() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 195
insert() (*rti.connextdds.DynamicData.DataReaderSeq method*), 205
insert() (*rti.connextdds.DynamicData.DataWriterSeq method*), 216
insert() (*rti.connextdds.DynamicDataSeq method*), 260
insert() (*rti.connextdds.DynamicDataTimestampedSeq method*), 263
insert() (*rti.connextdds.DynamicData.TopicSeq method*), 227
insert() (*rti.connextdds.EndpointGroupSeq method*), 268
insert() (*rti.connextdds.EntitySeq method*), 272
insert() (*rti.connextdds.EnumMemberSeq method*), 275
insert() (*rti.connextdds.Float32Seq method*), 287
insert() (*rti.connextdds.Float64Seq method*), 289
insert() (*rti.connextdds.Float128Seq method*), 284
insert() (*rti.connextdds.IAnyDataReaderSeq method*), 304
insert() (*rti.connextdds.IAnyDataWriterSeq method*), 307
insert() (*rti.connextdds.IAnyTopicSeq method*), 310
insert() (*rti.connextdds.IConditionSeq method*), 312
insert() (*rti.connextdds.IEntitySeq method*), 316
insert() (*rti.connextdds.InstanceHandleSeq method*), 322
insert() (*rti.connextdds.Int8Seq method*), 337
insert() (*rti.connextdds.Int16Seq method*), 328
insert() (*rti.connextdds.Int32Seq method*), 330
insert() (*rti.connextdds.Int64Seq method*), 334
insert() (*rti.connextdds.LocatorFilterElementSeq method*), 348
insert() (*rti.connextdds.LocatorSeq method*), 353
insert() (*rti.connextdds.MemberSeq method*), 366
insert() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
insert() (*rti.connextdds.MulticastMappingSeq method*), 378
insert() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
insert() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 402
insert() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
insert() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
insert() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 431
insert() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 423
insert() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 450
insert() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 459
insert() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
insert() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
insert() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 490
insert() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 481
insert() (*rti.connextdds.PublisherSeq method*), 499
insert() (*rti.connextdds.QosPolicyCountSeq method*), 502
insert() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 551
insert() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
insert() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
insert() (*rti.connextdds.ServiceRequestSeq method*), 587
insert() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 590
insert() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
insert() (*rti.connextdds.StringPairSeq method*), 600
insert() (*rti.connextdds.StringSeq method*), 602
insert() (*rti.connextdds.SubscriberSeq method*), 609
insert() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
insert() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623

insert() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
insert() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 651
insert() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 653
insert() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 644
insert() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
insert() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 680
insert() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
insert() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
insert() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 709
insert() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 701
insert() (*rti.connextdds.TopicSeq method*), 721
insert() (*rti.connextdds.TransportInfoSeq method*), 730
insert() (*rti.connextdds.TransportMulticastSeq method*), 737
insert() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740
insert() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
insert() (*rti.connextdds.Uint8Seq method*), 766
insert() (*rti.connextdds.Uint16Seq method*), 758
insert() (*rti.connextdds.Uint32Seq method*), 761
insert() (*rti.connextdds.Uint64Seq method*), 764
insert() (*rti.connextdds.UnionMemberSeq method*), 770
insert() (*rti.connextdds.WcharSeq method*), 783
insert() (*rti.connextdds.WstringSeq method*), 790
INSTANCE (*rti.connextdds.DestinationOrderScopeKind attribute*), 153
INSTANCE (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 152
instance (*rti.connextdds.Logger attribute*), 361
INSTANCE (*rti.connextdds.PresentationAccessScopeKind attribute*), 436
INSTANCE (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 437
instance() (*rti.connextdds.DataReader.Selector method*), 90
instance() (*rti.connextdds.DynamicData.DataReader.Selector method*), 196
instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
instance() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
instance() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
instance() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
instance_access_scope() (*rti.connextdds.Presentation static method*), 436
instance_handle (*rti.connextdds.IEntity property*), 313
instance_handle (*rti.connextdds.SampleInfo property*), 532
instance_hash_buckets (*rti.connextdds.ResourceLimits property*), 520
instance_id (*rti.connextdds.ServiceRequest property*), 582
instance_replacement (*rti.connextdds.DataReaderResourceLimits property*), 111
instance_replacement (*rti.connextdds.DataWriterResourceLimits property*), 140
instance_state (*rti.connextdds.DataState property*), 117
instance_state (*rti.connextdds.DataStateEx property*), 119
instance_state_consistency_kind (*rti.connextdds.Reliability property*), 512
InstanceHandle (*class in rti.connextdds*), 320
InstanceHandleSeq (*class in rti.connextdds*), 320
InstanceState (*class in rti.connextdds*), 322
InstanceStateConsistencyKind (*class in rti.connextdds*), 325
Int8Seq (*class in rti.connextdds*), 335
Int8Type (*class in rti.connextdds*), 337
Int16Seq (*class in rti.connextdds*), 326
Int16Type (*class in rti.connextdds*), 328
Int32Seq (*class in rti.connextdds*), 328
Int32Type (*class in rti.connextdds*), 331
Int32Vector (*class in rti.connextdds*), 331
Int64Seq (*class in rti.connextdds*), 332
Int64Type (*class in rti.connextdds*), 334
INT_8_TYPE (*rti.connextdds.TypeKind attribute*), 751
INT_8_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
INT_16_TYPE (*rti.connextdds.TypeKind attribute*), 751
INT_16_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
INT_32_TYPE (*rti.connextdds.TypeKind attribute*), 751
INT_32_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
INT_64_TYPE (*rti.connextdds.TypeKind attribute*), 751
INT_64_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
INTERMEDIATE_REPLY_SEQUENCE (*rti.connextdds.SampleFlag attribute*), 528
INTERMEDIATE_TOPIC_QUERY_SAMPLE (*rti.connextdds.SampleFlag attribute*), 528

interoperable (*rti.connextdds.RtpsWellKnownPorts attribute*), 527
INTRA (*rti.connextdds.LocatorKind attribute*), 348
INTRA (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
INTRA (*rti.connextdds.TransportClassId attribute*), 725
INTRA (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
invalid (*rti.connextdds.Locator attribute*), 344
INVALID (*rti.connextdds.LocatorKind attribute*), 348
INVALID (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
invalid (*rti.connextdds.Time attribute*), 664
INVALID (*rti.connextdds.TransportClassId attribute*), 725
INVALID (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
INVALID_ID (*rti.connextdds.Member attribute*), 363
INVALID_ID (*rti.connextdds.UnionMember attribute*), 768
INVALID_INDEX (*rti.connextdds.ACTEnumMember attribute*), 34
INVALID_INDEX (*rti.connextdds.ACTMember attribute*), 35
INVALID_INDEX (*rti.connextdds.ACTUnionMember attribute*), 36
INVALID_LOCAL_ADVANCE_NOTICE (*rti.connextdds.StatusMask attribute*), 592
InvalidArgumentError, 337
InvalidDowncastError, 337
InvalidLocalAdvanceNoticeStatus (*class in rti.connextdds*), 338
IReadCondition (*class in rti.connextdds*), 316
is_aggregation_type() (*rti.connextdds.DynamicType method*), 264
is_assignable_from() (*rti.connextdds.DynamicType method*), 264
is_cdr (*rti.connextdds.DynamicData property*), 242
is_collection_type() (*rti.connextdds.DynamicType method*), 265
is_constructed_type() (*rti.connextdds.DynamicType method*), 265
is_key (*rti.connextdds.Member property*), 364
is_keyed() (*rti.connextdds.DynamicType method*), 265
is_license_message (*rti.connextdds.LogMessage property*), 361
is_matched_publication_alive() (*rti.connextdds.DataReader method*), 93
is_matched_publication_alive() (*rti.connextdds.DynamicData.DataReader method*), 199
is_matched_publication_alive() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 396
is_matched_publication_alive() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 454
is_matched_publication_alive() (*rti.connextdds.ServiceRequest.DataReader method*), 555
is_matched_publication_alive() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 618
is_matched_publication_alive() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 674
is_matched_subscription_active() (*rti.connextdds.DataWriter method*), 123
is_matched_subscription_active() (*rti.connextdds.DynamicData.DataWriter method*), 208
is_matched_subscription_active() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 405
is_matched_subscription_active() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
is_matched_subscription_active() (*rti.connextdds.ServiceRequest.DataWriter method*), 563
is_matched_subscription_active() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
is_matched_subscription_active() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 683
is_member_key() (*rti.connextdds.DynamicData method*), 242
is_nil (*rti.connextdds.InstanceHandle property*), 320
is_optimized_storage (*rti.connextdds.DynamicDataInfo property*), 257
is_pointer (*rti.connextdds.AliasType property*), 44
is_primitive_type() (*rti.connextdds.DynamicType method*), 265
is_sample_app_acknowledged() (*rti.connextdds.DataWriter method*), 123
is_sample_app_acknowledged() (*rti.connextdds.DynamicData.DataWriter method*), 208
is_sample_app_acknowledged() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 405
is_sample_app_acknowledged() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
is_sample_app_acknowledged() (*rti.connextdds.ServiceRequest.DataWriter method*), 563
is_sample_app_acknowledged() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
is_sample_app_acknowledged() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 683
is_security_message (*rti.connextdds.LogMessage property*), 361
is_standalone (*rti.connextdds.QosPrintFormat property*), 503
is_type_registered() (*rti.connextdds.DomainParticipant method*), 169
items() (*rti.connextdds.DynamicData method*), 242
items() (*rti.connextdds.StringMap method*), 598

ItemsView (*class in rti.connextdds*), 338
ITopicDescription (*class in rti.connextdds*), 316

## J

JAVA_KEYHASH_750_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
journal_kind (*rti.connextdds.PersistentStorageSettings property*), 433
JSON (*rti.connextdds.PrintFormatKind attribute*), 441
JSON (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*), 441
json (*rti.connextdds.PrintFormatProperty attribute*), 443

## K

keep_all (*rti.connextdds.History attribute*), 299
KEEP_ALL (*rti.connextdds.HistoryKind attribute*), 300
KEEP_ALL (*rti.connextdds.HistoryKind.HistoryKind attribute*), 300
KEEP_LAST (*rti.connextdds.HistoryKind attribute*), 300
KEEP_LAST (*rti.connextdds.HistoryKind.HistoryKind attribute*), 300
keep_last() (*rti.connextdds.History static method*), 299
keep_minimum_state_for_instances (*rti.connextdds.DataReaderResourceLimits property*), 111
key (*rti.connextdds.ParticipantBuiltinTopicData property*), 425
key (*rti.connextdds.PublicationBuiltinTopicData property*), 484
key (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
key (*rti.connextdds.TopicBuiltinTopicData property*), 703
key (*rti.types.builtin.KeyedBytes attribute*), 804
key (*rti.types.builtin.KeyedString attribute*), 804
key_only_filter (*rti.connextdds.ExpressionProperty property*), 279
key_value() (*rti.connextdds.DataReader method*), 93
key_value() (*rti.connextdds.DataWriter method*), 123
key_value() (*rti.connextdds.DynamicData.DataReader method*), 200
key_value() (*rti.connextdds.DynamicData.DataWriter method*), 208
key_value() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 396
key_value() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 405
key_value() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 454
key_value() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
key_value() (*rti.connextdds.ServiceRequest.DataReader method*), 555
key_value() (*rti.connextdds.ServiceRequest.DataWriter method*), 563
key_value() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 618
key_value() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
key_value() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 674
key_value() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 683
KeyedBytes (*class in rti.types.builtin*), 804
KeyedString (*class in rti.types.builtin*), 804
keys() (*rti.connextdds.StringMap method*), 598
KeysView (*class in rti.connextdds*), 338
kind (*rti.connextdds.DestinationOrder property*), 149
kind (*rti.connextdds.Durability property*), 186
kind (*rti.connextdds.DynamicDataMemberInfo property*), 258
kind (*rti.connextdds.DynamicType property*), 265
kind (*rti.connextdds.History property*), 299
kind (*rti.connextdds.Liveliness property*), 340
kind (*rti.connextdds.Locator property*), 344
kind (*rti.connextdds.Ownership property*), 386
kind (*rti.connextdds.PrintFormatProperty property*), 443
kind (*rti.connextdds.PublishMode property*), 491
kind (*rti.connextdds.Reliability property*), 513
kind (*rti.connextdds.Service property*), 544
kind (*rti.connextdds.TopicQuerySelection property*), 717
kind (*rti.connextdds.TransportMulticast property*), 732
kind (*rti.connextdds.TypeConsistencyEnforcement property*), 747

## L

label_count (*rti.connextdds.UnionMember property*), 768
labels (*rti.connextdds.UnionMember property*), 768
last_available_sample_sequence_number (*rti.connextdds.DataReaderProtocolStatus property*), 102
last_available_sample_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 130
last_available_sample_virtual_sequence_number (*rti.connextdds.DataWriterProtocolStatus property*), 130
last_committed_sample_sequence_number (*rti.connextdds.DataReaderProtocolStatus property*), 102
last_corrupted_message_timestamp (*rti.connextdds.DomainParticipantProtocolStatus property*), 174
last_instance_handle (*rti.connextdds.OfferedDeadlineMissedStatus property*), 385
last_instance_handle (*rti.connextdds.ReliableReaderActivityChangedStatus property*), 515
last_instance_handle (*rti.connextdds.RequestedDeadlineMissedStatus property*), 518

last_instance_handle (*rti.connextdds.SampleRejectedStatus property*), 539
last_policy (*rti.connextdds.OfferedIncompatibleQosStatus property*), 385
last_policy (*rti.connextdds.RequestedIncompatibleQosStatus property*), 519
last_publication_handle (*rti.connextdds.LivelinessChangedStatus property*), 341
last_publication_handle (*rti.connextdds.SubscriptionMatchedStatus property*), 654
last_reason (*rti.connextdds.SampleLostStatus property*), 536
last_reason (*rti.connextdds.SampleRejectedStatus property*), 539
last_request_handle (*rti.connextdds.ServiceRequestAcceptedStatus property*), 583
LAST_SHARED_READER_QUEUE (*rti.connextdds.SampleFlag attribute*), 528
last_subscription_handle (*rti.connextdds.PublicationMatchedStatus property*), 490
late_joiner_heartbeat_period (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
latency_budget (*rti.connextdds.DataReaderQos property*), 109
latency_budget (*rti.connextdds.DataWriterQos property*), 138
latency_budget (*rti.connextdds.PublicationBuiltinTopicData property*), 484
latency_budget (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
latency_budget (*rti.connextdds.TopicBuiltinTopicData property*), 703
latency_budget (*rti.connextdds.TopicQos property*), 714
LatencyBudget (*class in rti.connextdds*), 339
lazily_allocate_batch_resources (*rti.connextdds.DataReaderResourceLimits property*), 111
lease_duration (*rti.connextdds.Liveliness property*), 340
length (*rti.connextdds.DataReader.LoanedSamples property*), 89
length (*rti.connextdds.DynamicData.LoanedSamples property*), 219
length (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples property*), 415
length (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples property*), 473
length (*rti.connextdds.ServiceRequest.LoanedSamples property*), 573
length (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples property*), 637
length (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples property*), 693
level (*rti.connextdds.LogMessage property*), 361
library_name (*rti.connextdds.BuiltinProfiles attribute*), 62
Lifespan (*class in rti.connextdds*), 339
lifespan (*rti.connextdds.DataWriterQos property*), 138
lifespan (*rti.connextdds.PublicationBuiltinTopicData property*), 484
lifespan (*rti.connextdds.TopicBuiltinTopicData property*), 703
lifespan (*rti.connextdds.TopicQos property*), 714
listener (*rti.connextdds.DataReader property*), 93
listener (*rti.connextdds.DataWriter property*), 123
listener (*rti.connextdds.DomainParticipant property*), 169
listener (*rti.connextdds.DynamicData.DataReader property*), 200
listener (*rti.connextdds.DynamicData.DataWriter property*), 208
listener (*rti.connextdds.DynamicData.Topic property*), 224
listener (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 396
listener (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 405
listener (*rti.connextdds.ParticipantBuiltinTopicData.Topic property*), 420
listener (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 454
listener (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 463
listener (*rti.connextdds.PublicationBuiltinTopicData.Topic property*), 478
listener (*rti.connextdds.Publisher property*), 495
listener (*rti.connextdds.ServiceRequest.DataReader property*), 555
listener (*rti.connextdds.ServiceRequest.DataWriter property*), 563
listener (*rti.connextdds.ServiceRequest.Topic property*), 577
listener (*rti.connextdds.Subscriber property*), 605
listener (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 618
listener (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 627
listener (*rti.connextdds.SubscriptionBuiltinTopicData.Topic property*), 641
listener (*rti.connextdds.Topic property*), 666
listener (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 675
listener (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 683
listener (*rti.connextdds.TopicBuiltinTopicData.Topic property*), 698
LIVE (*rti.connextdds.StreamKind attribute*), 595
Liveliness (*class in rti.connextdds*), 340
liveliness (*rti.connextdds.DataReaderQos property*), 109

liveliness (*rti.connextdds.DataWriterQos property*), 138
liveliness (*rti.connextdds.PublicationBuiltinTopicData property*), 484
liveliness (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
liveliness (*rti.connextdds.TopicBuiltinTopicData property*), 703
liveliness (*rti.connextdds.TopicQos property*), 714
LIVELINESS_BASED (*rti.connextdds.RemoteParticipantPurgeKind attribute*), 516
LIVELINESS_BASED (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute*), 516
LIVELINESS_CHANGED (*rti.connextdds.StatusMask attribute*), 592
liveliness_changed_status (*rti.connextdds.DataReader property*), 93
liveliness_changed_status (*rti.connextdds.DynamicData.DataReader property*), 200
liveliness_changed_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 396
liveliness_changed_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 454
liveliness_changed_status (*rti.connextdds.ServiceRequest.DataReader property*), 555
liveliness_changed_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 618
liveliness_changed_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 675
LIVELINESS_LOST (*rti.connextdds.StatusMask attribute*), 592
liveliness_lost_status (*rti.connextdds.DataWriter property*), 123
liveliness_lost_status (*rti.connextdds.DynamicData.DataWriter property*), 208
liveliness_lost_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 405
liveliness_lost_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 463
liveliness_lost_status (*rti.connextdds.ServiceRequest.DataWriter property*), 564
liveliness_lost_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 627
liveliness_lost_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 683
LivelinessChangedStatus (*class in rti.connextdds*), 341
LivelinessKind (*class in rti.connextdds*), 341
LivelinessKind.LivelinessKind (*class in rti.connextdds*), 341
LivelinessLostStatus (*class in rti.connextdds*), 343
load_compliance_mask() (*in module rti.connextdds.compliance*), 797
load_profiles() (*rti.connextdds.QosProvider method*), 504
loan_value() (*rti.connextdds.DynamicData method*), 242
LoanedDynamicData (*class in rti.connextdds*), 344
local_publisher_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 179
local_publisher_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_reader_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_reader_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_subscriber_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_subscriber_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_topic_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_topic_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_writer_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 180
local_writer_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 180
Locator (*class in rti.connextdds*), 344
locator_filter (*rti.connextdds.PublicationBuiltinTopicData property*), 484
locator_filters (*rti.connextdds.LocatorFilter property*), 345
LOCATOR_REACHABILITY (*rti.connextdds.ServiceRequestId attribute*), 583
LOCATOR_REACHABILITY (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
locator_reachability_assert_period (*rti.connextdds.DiscoveryConfig property*), 155
locator_reachability_change_detection_period (*rti.connextdds.DiscoveryConfig property*), 155
locator_reachability_lease_duration (*rti.connextdds.DiscoveryConfig property*), 156
LocatorFilter (*class in rti.connextdds*), 345
LocatorFilterElement (*class in rti.connextdds*), 345
LocatorFilterElementSeq (*class in rti.connextdds*), 346

LocatorKind (*class in rti.connextdds*), 348
LocatorKind.LocatorKind (*class in rti.connextdds*), 348
locators (*rti.connextdds.LocatorFilterElement property*), 346
LocatorSeq (*class in rti.connextdds*), 351
LocatorVector (*class in rti.connextdds*), 353
lock() (*rti.connextdds.IEntity method*), 313
log() (*rti.logging.distlog.Logger static method*), 813
log_infrastructure_messages (*rti.logging.distlog.LoggerOptions property*), 814
log_level (*rti.logging.distlog.MessageParams property*), 815
LogCategory (*class in rti.connextdds*), 354
LogCategory.LogCategory (*class in rti.connextdds*), 354
LogFacility (*class in rti.connextdds*), 357
Logger (*class in rti.connextdds*), 361
Logger (*class in rti.logging.distlog*), 813
LoggerOptions (*class in rti.logging.distlog*), 814
logging_settings (*rti.connextdds.MonitoringDistributionSettings property*), 368
LogLevel (*class in rti.connextdds*), 358
LogLevel (*class in rti.logging.distlog*), 812
LogLevel.LogLevel (*class in rti.connextdds*), 358
LogMessage (*class in rti.connextdds*), 360
logs (*rti.connextdds.MonitoringTelemetryData property*), 374
LongDouble (*class in rti.connextdds*), 362
LongDoubleType (*in module rti.connextdds*), 363
LongLongSeq (*in module rti.connextdds*), 363
LongLongType (*in module rti.connextdds*), 363
LongSeq (*in module rti.connextdds*), 363
LongType (*in module rti.connextdds*), 363
lookup_instance() (*rti.connextdds.DataReader method*), 93
lookup_instance() (*rti.connextdds.DataWriter method*), 123
lookup_instance() (*rti.connextdds.DynamicData.DataReader method*), 200
lookup_instance() (*rti.connextdds.DynamicData.DataWriter method*), 208
lookup_instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
lookup_instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 405
lookup_instance() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 454
lookup_instance() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
lookup_instance() (*rti.connextdds.ServiceRequest.DataReader method*), 555
lookup_instance() (*rti.connextdds.ServiceRequest.DataWriter method*), 564
lookup_instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 618
lookup_instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
lookup_instance() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
lookup_instance() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 683
LOST_BY_AVAILABILITY_WAITING_TIME (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_DECODE_FAILURE (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_DESERIALIZATION_FAILURE (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_INCOMPLETE_COHERENT_SET (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_INSTANCES_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_LARGE_COHERENT_SET (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_OUT_OF_MEMORY (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_REMOTE_WRITERS_PER_INSTANCE_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_REMOTE_WRITERS_PER_SAMPLE_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_REMOTE_WRITERS_PER_VIRTUAL_QUEUE_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_SAMPLES_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_SAMPLES_PER_INSTANCE_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_SAMPLES_PER_REMOTE_WRITER_LIMIT (*rti.connextdds.SampleLostState attribute*), 533
LOST_BY_UNKNOWN_INSTANCE (*rti.connextdds.SampleLostState attribute*), 534
LOST_BY_VIRTUAL_WRITERS_LIMIT (*rti.connextdds.SampleLostState attribute*), 534
LOST_BY_WRITER (*rti.connextdds.SampleLostState attribute*), 534
low_watermark (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
low_watermark_reliable_writer_cache (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 515
LZ4 (*rti.connextdds.CompressionIdMask attribute*), 75

## M

major_version (*rti.connextdds.ProductVersion property*), 444
major_version (*rti.connextdds.ProtocolVersion property*), 446
MANUAL_BY_PARTICIPANT (*rti.connextdds.LivelinessKind attribute*), 342

MANUAL_BY_PARTICIPANT (*rti.connextdds.LivelinessKind.LivelinessKind attribute*), 342
manual_by_participant() (*rti.connextdds.Liveliness static method*), 340
MANUAL_BY_TOPIC (*rti.connextdds.LivelinessKind attribute*), 342
MANUAL_BY_TOPIC (*rti.connextdds.LivelinessKind.LivelinessKind attribute*), 342
manual_by_topic() (*rti.connextdds.Liveliness static method*), 340
manually_enable (*rti.connextdds.EntityFactory attribute*), 270
MAP_TYPE (*rti.connextdds.TypeKind attribute*), 751
MAP_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
mapping_function (*rti.connextdds.MulticastMapping property*), 376
mask (*rti.connextdds.ThreadSettings property*), 658
mask (*rti.connextdds.TransportBuiltin property*), 722
matched_client_count (*rti.rpc.Service property*), 809
matched_publication_data() (*rti.connextdds.DataReader method*), 94
matched_publication_data() (*rti.connextdds.DynamicData.DataReader method*), 200
matched_publication_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
matched_publication_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 454
matched_publication_data() (*rti.connextdds.ServiceRequest.DataReader method*), 555
matched_publication_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 618
matched_publication_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
matched_publication_datareader_protocol_status() (*rti.connextdds.DataReader method*), 94
matched_publication_datareader_protocol_status() (*rti.connextdds.DynamicData.DataReader method*), 200
matched_publication_datareader_protocol_status() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
matched_publication_datareader_protocol_status() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 455
matched_publication_datareader_protocol_status() (*rti.connextdds.ServiceRequest.DataReader method*), 555
matched_publication_datareader_protocol_status() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 618
matched_publication_datareader_protocol_status() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
matched_publication_participant_data() (*rti.connextdds.DataReader method*), 94
matched_publication_participant_data() (*rti.connextdds.DynamicData.DataReader method*), 200
matched_publication_participant_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
matched_publication_participant_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 455
matched_publication_participant_data() (*rti.connextdds.ServiceRequest.DataReader method*), 556
matched_publication_participant_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 618
matched_publication_participant_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
matched_publications (*rti.connextdds.DataReader property*), 94
matched_publications (*rti.connextdds.DynamicData.DataReader property*), 200
matched_publications (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 397
matched_publications (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 455
matched_publications (*rti.connextdds.ServiceRequest.DataReader property*), 556
matched_publications (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
matched_publications (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 675
matched_requester_count (*rti.rpc.SimpleReplier property*), 810
matched_service_count (*rti.rpc.ClientBase property*), 805

matched_subscription_data() (*rti.connextdds.DataWriter method*), 123
matched_subscription_data() (*rti.connextdds.DynamicData.DataWriter method*), 208
matched_subscription_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 405
matched_subscription_data() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
matched_subscription_data() (*rti.connextdds.ServiceRequest.DataWriter method*), 564
matched_subscription_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
matched_subscription_data() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 684
matched_subscription_datawriter_protocol_status() (*rti.connextdds.DataWriter method*), 123
matched_subscription_datawriter_protocol_status() (*rti.connextdds.DynamicData.DataWriter method*), 208
matched_subscription_datawriter_protocol_status() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 406
matched_subscription_datawriter_protocol_status() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 463
matched_subscription_datawriter_protocol_status() (*rti.connextdds.ServiceRequest.DataWriter method*), 564
matched_subscription_datawriter_protocol_status() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
matched_subscription_datawriter_protocol_status() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 684
matched_subscription_participant_data() (*rti.connextdds.DataWriter method*), 124
matched_subscription_participant_data() (*rti.connextdds.DynamicData.DataWriter method*), 209
matched_subscription_participant_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 406
matched_subscription_participant_data() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 464
matched_subscription_participant_data() (*rti.connextdds.ServiceRequest.DataWriter method*), 564
matched_subscription_participant_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 627
matched_subscription_participant_data() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 684
matched_subscriptions (*rti.connextdds.DataWriter property*), 124
matched_subscriptions (*rti.connextdds.DynamicData.DataWriter property*), 209
matched_subscriptions (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
matched_subscriptions (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
matched_subscriptions (*rti.connextdds.ServiceRequest.DataWriter property*), 564
matched_subscriptions (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
matched_subscriptions (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
matched_subscriptions_locators (*rti.connextdds.DataWriter property*), 124
matched_subscriptions_locators (*rti.connextdds.DynamicData.DataWriter property*), 209
matched_subscriptions_locators (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
matched_subscriptions_locators (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
matched_subscriptions_locators (*rti.connextdds.ServiceRequest.DataWriter property*), 564
matched_subscriptions_locators (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
matched_subscriptions_locators (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
matching_reader_writer_pair_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 180
matching_reader_writer_pair_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits*

*property*), 180
matching_writer_reader_pair_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 180
matching_writer_reader_pair_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 180
max_active_topic_queries (*rti.connextdds.DataWriterResourceLimits property*), 140
max_app_ack_remote_readers (*rti.connextdds.DataWriterResourceLimits property*), 140
max_app_ack_response_length (*rti.connextdds.DataReaderResourceLimits property*), 111
max_batches (*rti.connextdds.DataWriterResourceLimits property*), 140
max_blocking_time (*rti.connextdds.Reliability property*), 513
max_buffer_size (*rti.connextdds.DynamicDataProperty property*), 258
max_buffer_size_increment (*rti.connextdds.DynamicDataProperty property*), 258
max_bytes_per_nack_response (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
max_concurrent_blocking_threads (*rti.connextdds.DataWriterResourceLimits property*), 140
max_count (*rti.connextdds.AllocationSettings property*), 45
max_count (*rti.connextdds.Event property*), 277
max_data_availability_waiting_time (*rti.connextdds.Availability property*), 57
max_data_bytes (*rti.connextdds.Batch property*), 57
max_endpoint_availability_waiting_time (*rti.connextdds.Availability property*), 57
max_endpoint_group_cumulative_characters (*rti.connextdds.DomainParticipantResourceLimits property*), 181
max_endpoint_groups (*rti.connextdds.DomainParticipantResourceLimits property*), 181
max_flush_delay (*rti.connextdds.Batch property*), 57
max_fragmented_samples (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_fragmented_samples (*rti.connextdds.DataReaderResourceLimits property*), 111
max_fragmented_samples_per_remote_writer (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_fragmented_samples_per_remote_writer (*rti.connextdds.DataReaderResourceLimits property*), 111
max_fragments_per_sample (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_fragments_per_sample (*rti.connextdds.DataReaderResourceLimits property*), 111
max_gather_destinations (*rti.connextdds.DomainParticipantResourceLimits property*), 181
max_heartbeat_response_delay (*rti.connextdds.RtpsReliableReaderProtocol property*), 520
max_heartbeat_retries (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
max_historical_logs (*rti.connextdds.MonitoringLoggingDistributionSettings property*), 370
max_infos (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_infos (*rti.connextdds.DataReaderResourceLimits property*), 111
max_initial_participant_announcement_period (*rti.connextdds.DiscoveryConfig property*), 156
max_instances (*rti.connextdds.DurabilityService property*), 189
max_instances (*rti.connextdds.ResourceLimits property*), 520
max_liveliness_loss_detection_period (*rti.connextdds.DiscoveryConfig property*), 156
max_nack_response_delay (*rti.connextdds.RtpsReliableWriterProtocol property*), 522
max_objects_per_thread (*rti.connextdds.SystemResourceLimits property*), 657
max_outstanding_reads (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_outstanding_reads (*rti.connextdds.DataReaderResourceLimits property*), 111
max_partition_cumulative_characters (*rti.connextdds.DomainParticipantResourceLimits property*), 181
max_partitions (*rti.connextdds.DomainParticipantResourceLimits property*), 181
max_query_condition_filters (*rti.connextdds.DataReaderResourceLimits property*), 111
max_remote_reader_filters (*rti.connextdds.DataWriterResourceLimits property*), 140
max_remote_readers (*rti.connextdds.DataWriterResourceLimits property*), 140
max_remote_virtual_writers (*rti.connextdds.DataReaderResourceLimits property*), 112
max_remote_virtual_writers_per_instance (*rti.connextdds.DataReaderResourceLimits property*),

112
max_remote_writers (*rti.connextdds.DataReaderResourceLimits property*), 112
max_remote_writers_per_instance (*rti.connextdds.DataReaderResourceLimits property*), 112
max_remote_writers_per_sample (*rti.connextdds.DataReaderResourceLimits property*), 112
max_samples (*rti.connextdds.Batch property*), 58
max_samples (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_samples (*rti.connextdds.DurabilityService property*), 189
max_samples (*rti.connextdds.ResourceLimits property*), 520
max_samples() (*rti.connextdds.DataReader.Selector method*), 90
max_samples() (*rti.connextdds.DynamicData.DataReader.Selector method*), 196
max_samples() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
max_samples() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
max_samples() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
max_samples() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
max_samples() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
max_samples_per_instance (*rti.connextdds.DurabilityService property*), 190
max_samples_per_instance (*rti.connextdds.ResourceLimits property*), 520
max_samples_per_read (*rti.connextdds.BuiltinTopicReaderResourceLimits property*), 64
max_samples_per_read (*rti.connextdds.DataReaderResourceLimits property*), 112
max_samples_per_remote_writer (*rti.connextdds.DataReaderResourceLimits property*), 112
max_send_window_size (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
max_serialized_size (*rti.connextdds.DynamicDataTypeSerializationProperty property*), 264
max_skiplist_level (*rti.connextdds.Database property*), 148
max_tokens (*rti.connextdds.FlowControllerTokenBucketProperty property*), 295
max_topic_queries (*rti.connextdds.DataReaderResourceLimits property*), 112
max_total_instances (*rti.connextdds.DataReaderResourceLimits property*), 112
max_virtual_writers (*rti.connextdds.DataWriterResourceLimits property*), 140
max_weak_references (*rti.connextdds.Database property*), 148
MAXIMAL (*rti.connextdds.PrintFormat attribute*), 439
MAXIMAL (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
maximum (*rti.connextdds.SequenceNumber attribute*), 543
maximum (*rti.connextdds.Time attribute*), 664
Member (*class in rti.connextdds*), 363
member() (*rti.connextdds.ACTEnumMember method*), 35
member() (*rti.connextdds.ACTMember method*), 36
member() (*rti.connextdds.ACTUnionMember method*), 37
member_count (*rti.connextdds.ACTEnumMember property*), 35
member_count (*rti.connextdds.ACTMember property*), 36
member_count (*rti.connextdds.ACTUnionMember property*), 37
member_count (*rti.connextdds.DynamicData property*), 242
member_count (*rti.connextdds.DynamicDataInfo property*), 257
member_exists() (*rti.connextdds.DynamicData method*), 242
member_exists_in_type() (*rti.connextdds.DynamicData method*), 243
member_index() (*rti.connextdds.DynamicData method*), 243
member_info() (*rti.connextdds.DynamicData method*), 243
members() (*rti.connextdds.ACTEnumMember method*), 35
members() (*rti.connextdds.ACTMember method*), 36
members() (*rti.connextdds.ACTUnionMember method*), 37
MemberSeq (*class in rti.connextdds*), 364
MEMORY (*rti.connextdds.PersistentJournalKind attribute*), 432
message (*rti.logging.distlog.MessageParams property*), 815
message_id (*rti.connextdds.LogMessage property*), 361
message_size_max (*rti.connextdds.TransportInfo property*), 728
MessageParams (*class in rti.logging.distlog*), 815
metatraffic_transport_priority (*rti.connextdds.Discovery property*), 154
metrics (*rti.connextdds.MonitoringTelemetryData property*), 375
middleware (*rti.connextdds.LogFacility attribute*), 358
middleware_forwarding_level (*rti.connextdds.MonitoringLoggingForwardingSettings property*), 370
min_heartbeat_response_delay (*rti.connextdds.RtpsReliableReaderProtocol property*), 521
min_initial_participant_announcement_period (*rti.connextdds.DiscoveryConfig property*), 156

min_nack_response_delay (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
min_send_window_size (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
min_serialized_size (*rti.connextdds.DynamicDataTypeSerializationProperty property*), 264
min_serialized_size (*rti.connextdds.DynamicTypePrintFormatProperty property*), 266
MINIMAL (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 798
MINIMAL (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*), 799
MINIMAL (*rti.connextdds.PrintFormat attribute*), 439
MINIMAL (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
minimum_separation (*rti.connextdds.TimeBasedFilter property*), 665
minor_version (*rti.connextdds.ProductVersion property*), 444
minor_version (*rti.connextdds.ProtocolVersion property*), 446
module
    rti.asyncio, 810
    rti.connextdds, 34
    rti.connextdds.compliance, 791
    rti.connextdds.heap_monitoring, 798
    rti.logging, 816
    rti.logging.distlog, 812
    rti.logging.handler, 815
    rti.rpc, 805
    rti.types.builtin, 804
Monitoring (*class in rti.connextdds*), 366
monitoring (*rti.connextdds.DomainParticipantFactoryQos property*), 173
MONITORING_LIBRARY_COMMAND (*rti.connextdds.ServiceRequestId attribute*), 583
MONITORING_LIBRARY_COMMAND (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
MONITORING_LIBRARY_REPLY (*rti.connextdds.ServiceRequestId attribute*), 583
MONITORING_LIBRARY_REPLY (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
MonitoringDedicatedParticipantSettings (*class in rti.connextdds*), 367
MonitoringDistributionSettings (*class in rti.connextdds*), 368
MonitoringEventDistributionSettings (*class in rti.connextdds*), 368
MonitoringLoggingDistributionSettings (*class in rti.connextdds*), 369
MonitoringLoggingForwardingSettings (*class in rti.connextdds*), 370
MonitoringMetricSelection (*class in rti.connextdds*), 370
MonitoringMetricSelectionSeq (*class in rti.connextdds*), 371
MonitoringPeriodicDistributionSettings (*class in rti.connextdds*), 373
MonitoringTelemetryData (*class in rti.connextdds*), 374
multi_channel (*rti.connextdds.DataWriterQos property*), 138
multicast_locators (*rti.connextdds.SubscriptionBuiltinTopicData property*), 647
multicast_receive_addresses (*rti.connextdds.Discovery property*), 154
multicast_resend_threshold (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
multicast_settings (*rti.connextdds.ChannelSettings property*), 68
MulticastMapping (*class in rti.connextdds*), 375
MulticastMappingSeq (*class in rti.connextdds*), 376
MultiChannel (*class in rti.connextdds*), 375
must_understand (*rti.connextdds.Member property*), 364
MUST_UNDERSTAND_IN_KEY_AND_DISCRIMINATOR_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
MUTABLE (*rti.connextdds.ExtensibilityKind attribute*), 280
MUTABLE (*rti.connextdds.ExtensibilityKind.ExtensibilityKind attribute*), 279

## N

nack_period (*rti.connextdds.RtpsReliableReaderProtocol property*), 521
nack_suppression_duration (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
name (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind property*), 40
name (*rti.connextdds.CdrPaddingKind.CdrPaddingKind property*), 66
name (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind property*), 98
name (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind property*), 143
name (*rti.connextdds.DestinationOrderKind.DestinationOrderKind property*), 150
name (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind property*), 152
name (*rti.connextdds.DurabilityKind.DurabilityKind property*), 188

name (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind property*), 255
name (*rti.connextdds.DynamicData.ITopicDescription property*), 217
name (*rti.connextdds.DynamicDataMemberInfo property*), 258
name (*rti.connextdds.DynamicType property*), 265
name (*rti.connextdds.EntityName property*), 271
name (*rti.connextdds.EnumMember property*), 274
name (*rti.connextdds.ExtensibilityKind.ExtensibilityKind property*), 280
name (*rti.connextdds.Filter property*), 281
name (*rti.connextdds.FlowController property*), 291
name (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy property*), 294
name (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat property*), 800
name (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat property*), 802
name (*rti.connextdds.HistoryKind.HistoryKind property*), 300
name (*rti.connextdds.IAnyTopic property*), 308
name (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind property*), 318
name (*rti.connextdds.InstanceStateConsistencyKind property*), 326
name (*rti.connextdds.ITopicDescription property*), 317
name (*rti.connextdds.LivelinessKind.LivelinessKind property*), 342
name (*rti.connextdds.LocatorKind.LocatorKind property*), 350
name (*rti.connextdds.LogCategory.LogCategory property*), 356
name (*rti.connextdds.LogFacility property*), 358
name (*rti.connextdds.LogLevel.LogLevel property*), 359
name (*rti.connextdds.Member property*), 364
name (*rti.connextdds.OwnershipKind.OwnershipKind property*), 387
name (*rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription property*), 414
name (*rti.connextdds.Partition property*), 432
name (*rti.connextdds.PersistentJournalKind property*), 433
name (*rti.connextdds.PersistentSynchronizationKind property*), 435
name (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind property*), 437
name (*rti.connextdds.PrintFormatKind.PrintFormatKind property*), 442
name (*rti.connextdds.PrintFormat.PrintFormat property*), 440
name (*rti.connextdds.PublicationBuiltinTopicData.ITopicDescription property*), 472
name (*rti.connextdds.PublishModeKind.PublishModeKind property*), 493
name (*rti.connextdds.Query property*), 507
name (*rti.connextdds.ReliabilityKind.ReliabilityKind property*), 514
name (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind property*), 517
name (*rti.connextdds.ServiceKind.ServiceKind property*), 546
name (*rti.connextdds.ServiceRequestId.ServiceRequestId property*), 584
name (*rti.connextdds.ServiceRequest.ITopicDescription property*), 572
name (*rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription property*), 635
name (*rti.connextdds.SyslogVerbosity property*), 656
name (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind property*), 659
name (*rti.connextdds.TopicBuiltinTopicData property*), 703
name (*rti.connextdds.TopicBuiltinTopicData.ITopicDescription property*), 692
name (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind property*), 718
name (*rti.connextdds.TransportClassId.TransportClassId property*), 727
name (*rti.connextdds.TransportMulticastKind.TransportMulticastKind property*), 733
name (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind property*), 749
name (*rti.connextdds.TypeKind.TypeKind property*), 754
name (*rti.connextdds.UnionMember property*), 768
name (*rti.connextdds.Verbosity.Verbosity property*), 775
name (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind property*), 786
name (*rti.logging.distlog.LogLevel property*), 813
nanosec (*rti.connextdds.Duration property*), 192
nanosec (*rti.connextdds.Time property*), 664
native_build_id (*rti.connextdds.ProductVersion property*), 444
new_data (*rti.connextdds.DataState attribute*), 117
new_data (*rti.connextdds.DataStateEx attribute*), 119
new_instance (*rti.connextdds.DataState attribute*), 117
new_instance (*rti.connextdds.DataStateEx attribute*), 119
new_remote_participant_announcements (*rti.connextdds.DiscoveryConfig property*), 156
NEW_VIEW (*rti.connextdds.ViewState attribute*), 776
next_instance() (*rti.connextdds.DataReader.Selector method*), 90
next_instance() (*rti.connextdds.DynamicData.DataReader.Selector method*), 196
next_instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
next_instance() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*),

451
next_instance() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
next_instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
next_instance() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
nil() (*rti.connextdds.InstanceHandle static method*), 320
no_auto_purge (*rti.connextdds.ReaderDataLifecycle attribute*), 511
NO_INSTANCE (*rti.connextdds.DataReaderInstanceRemovalKind attribute*), 99
NO_INSTANCE (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 98
NO_PURGE (*rti.connextdds.RemoteParticipantPurgeKind attribute*), 516
NO_PURGE (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute*), 517
NO_REPLACEMENT (*rti.connextdds.IgnoredEntityReplacementKind attribute*), 318
NO_REPLACEMENT (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 317
NO_ROTATION (*rti.connextdds.ThreadSettingsCpuRotationKind attribute*), 658
NO_ROTATION (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind attribute*), 658
NO_SERVICE (*rti.connextdds.ServiceKind attribute*), 545
NO_SERVICE (*rti.connextdds.ServiceKind.ServiceKind attribute*), 545
NO_TYPE (*rti.connextdds.TypeKind attribute*), 751
NO_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
no_writers (*rti.connextdds.GenerationCount property*), 296
no_writers_instance_count (*rti.connextdds.DataReaderCacheStatus property*), 96
no_writers_instance_count_peak (*rti.connextdds.DataReaderCacheStatus property*), 96
no_writers_instance_removal (*rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings property*), 113
NONE (*rti.connextdds.ActivityContextMask attribute*), 41
NONE (*rti.connextdds.CompressionIdMask attribute*), 75
NONE (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 158
NONE (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
NONE (*rti.connextdds.InstanceStateConsistencyKind attribute*), 325
NONE (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
NONE (*rti.connextdds.StatusMask attribute*), 592
none (*rti.connextdds.TransportBuiltin attribute*), 722
NONE (*rti.connextdds.TransportBuiltinMask attribute*), 722
NoOpAnyDataReaderListener (*class in rti.connextdds*), 378
NoOpAnyDataWriterListener (*class in rti.connextdds*), 379
NoOpAnyTopicListener (*class in rti.connextdds*), 380
NoOpDataReaderListener (*class in rti.connextdds*), 381
NoOpDataWriterListener (*class in rti.connextdds*), 382
NoOpDomainParticipantListener (*class in rti.connextdds*), 383
NoOpPublisherListener (*class in rti.connextdds*), 383
NoOpSubscriberListener (*class in rti.connextdds*), 383
NoOpTopicListener (*class in rti.connextdds*), 383
NORMAL (*rti.connextdds.PersistentSynchronizationKind attribute*), 434
not_alive_count (*rti.connextdds.LivelinessChangedStatus property*), 341
not_alive_count_change (*rti.connextdds.LivelinessChangedStatus property*), 341
NOT_ALIVE_DISPOSED (*rti.connextdds.InstanceState attribute*), 323
NOT_ALIVE_FIRST (*rti.connextdds.IgnoredEntityReplacementKind attribute*), 318
NOT_ALIVE_FIRST (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 317
NOT_ALIVE_MASK (*rti.connextdds.InstanceState attribute*), 323
NOT_ALIVE_NO_WRITERS (*rti.connextdds.InstanceState attribute*), 323
NOT_LOST (*rti.connextdds.SampleLostState attribute*), 534
NOT_NEW_VIEW (*rti.connextdds.ViewState attribute*), 776
NOT_READ (*rti.connextdds.SampleState attribute*), 540
NOT_REJECTED (*rti.connextdds.SampleRejectedState attribute*), 536
NOT_SET (*rti.connextdds.CdrPaddingKind attribute*), 66
NOT_SET (*rti.connextdds.CdrPaddingKind.CdrPaddingKind attribute*), 66
NotAllowedBySecurityError, 384
NotEnabledError, 384
NOTICE (*rti.connextdds.SyslogVerbosity attribute*), 655
NOTICE (*rti.logging.distlog.LogLevel attribute*), 813
notice() (*rti.connextdds.Logger method*), 361
notice() (*rti.logging.distlog.Logger static method*), 814
notify_datareaders() (*rti.connextdds.Subscriber method*), 605
NullReferenceError, 384
num_uses (*rti.connextdds.ThreadSettings property*), 658

## O

OBSERVABILITY_COLLECTOR (*rti.connextdds.ServiceKind attribute*), 545
OBSERVABILITY_COLLECTOR (*rti.connextdds.ServiceKind.ServiceKind attribute*), 545
OCTET_TYPE (*rti.connextdds.TypeKind attribute*), 751

OCTET_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
OctetSeq (*in module rti.connextdds*), 384
OctetType (*class in rti.connextdds*), 384
OFF (*rti.connextdds.PersistentJournalKind attribute*), 432
OFF (*rti.connextdds.PersistentSynchronizationKind attribute*), 434
OFFERED_DEADLINE_MISSED (*rti.connextdds.StatusMask attribute*), 592
offered_deadline_missed_status (*rti.connextdds.DataWriter property*), 124
offered_deadline_missed_status (*rti.connextdds.DynamicData.DataWriter property*), 209
offered_deadline_missed_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
offered_deadline_missed_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
offered_deadline_missed_status (*rti.connextdds.ServiceRequest.DataWriter property*), 564
offered_deadline_missed_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
offered_deadline_missed_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
OFFERED_INCOMPATIBLE_QOS (*rti.connextdds.StatusMask attribute*), 592
offered_incompatible_qos_status (*rti.connextdds.DataWriter property*), 124
offered_incompatible_qos_status (*rti.connextdds.DynamicData.DataWriter property*), 209
offered_incompatible_qos_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
offered_incompatible_qos_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
offered_incompatible_qos_status (*rti.connextdds.ServiceRequest.DataWriter property*), 564
offered_incompatible_qos_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
offered_incompatible_qos_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
OfferedDeadlineMissedStatus (*class in rti.connextdds*), 384
OfferedIncompatibleQosStatus (*class in rti.connextdds*), 385
old_source_timestamp_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 96
on_application_acknowledgment() (*rti.connextdds.AnyDataWriterListener method*), 49
on_application_acknowledgment() (*rti.connextdds.DataWriterListener method*), 127
on_application_acknowledgment() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_application_acknowledgment() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 220
on_application_acknowledgment() (*rti.connextdds.NoOpAnyDataWriterListener method*), 379
on_application_acknowledgment() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_application_acknowledgment() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 410
on_application_acknowledgment() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 416
on_application_acknowledgment() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 468
on_application_acknowledgment() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 474
on_application_acknowledgment() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 568
on_application_acknowledgment() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 574
on_application_acknowledgment() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_application_acknowledgment() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 638
on_application_acknowledgment() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 688
on_application_acknowledgment() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 694
on_data_available() (*rti.connextdds.AnyDataReaderListener method*), 46
on_data_available() (*rti.connextdds.DataReaderListener method*), 100
on_data_available() (*rti.connextdds.DynamicData.DataReaderListener method*), 202
on_data_available() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*),

219
on_data_available() (*rti.connextdds.NoOpAnyDataReaderListener method*), 378
on_data_available() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_data_available() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
on_data_available() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 415
on_data_available() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_data_available() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 473
on_data_available() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 557
on_data_available() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 573
on_data_available() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 620
on_data_available() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 637
on_data_available() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 677
on_data_available() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 693
on_data_on_readers() (*rti.connextdds.NoOpSubscriberListener method*), 383
on_data_on_readers() (*rti.connextdds.SubscriberListener method*), 606
ON_DEMAND_NAME (*rti.connextdds.FlowController attribute*), 290
on_inconsistent_topic() (*rti.connextdds.AnyTopicListener method*), 52
on_inconsistent_topic() (*rti.connextdds.DynamicData.NoOpTopicListener method*), 222
on_inconsistent_topic() (*rti.connextdds.DynamicData.TopicListener method*), 225
on_inconsistent_topic() (*rti.connextdds.NoOpAnyTopicListener method*), 381
on_inconsistent_topic() (*rti.connextdds.NoOpTopicListener method*), 384
on_inconsistent_topic() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener method*), 418
on_inconsistent_topic() (*rti.connextdds.ParticipantBuiltinTopicData.TopicListener method*), 421
on_inconsistent_topic() (*rti.connextdds.PublicationBuiltinTopicData.NoOpTopicListener method*), 476
on_inconsistent_topic() (*rti.connextdds.PublicationBuiltinTopicData.TopicListener method*), 479
on_inconsistent_topic() (*rti.connextdds.ServiceRequest.NoOpTopicListener method*), 575
on_inconsistent_topic() (*rti.connextdds.ServiceRequest.TopicListener method*), 578
on_inconsistent_topic() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpTopicListener method*), 640
on_inconsistent_topic() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicListener method*), 642
on_inconsistent_topic() (*rti.connextdds.TopicBuiltinTopicData.NoOpTopicListener method*), 696
on_inconsistent_topic() (*rti.connextdds.TopicBuiltinTopicData.TopicListener method*), 699
on_inconsistent_topic() (*rti.connextdds.TopicListener method*), 710
on_instance_replaced() (*rti.connextdds.AnyDataWriterListener method*), 49
on_instance_replaced() (*rti.connextdds.DataWriterListener method*), 127
on_instance_replaced() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_instance_replaced() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 220
on_instance_replaced() (*rti.connextdds.NoOpAnyDataWriterListener method*), 379
on_instance_replaced() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_instance_replaced() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 410
on_instance_replaced() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_instance_replaced() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 468
on_instance_replaced() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_instance_replaced() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 568

on_instance_replaced() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 574
on_instance_replaced() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_instance_replaced() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 638
on_instance_replaced() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 688
on_instance_replaced() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 694
on_invalid_local_status_advance_notice() (*rti.connextdds.DomainParticipantListener method*), 173
on_invalid_local_status_advance_notice() (*rti.connextdds.NoOpDomainParticipantListener method*), 383
on_liveliness_changed() (*rti.connextdds.AnyDataReaderListener method*), 46
on_liveliness_changed() (*rti.connextdds.DataReaderListener method*), 100
on_liveliness_changed() (*rti.connextdds.DynamicData.DataReaderListener method*), 202
on_liveliness_changed() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 219
on_liveliness_changed() (*rti.connextdds.NoOpAnyDataReaderListener method*), 379
on_liveliness_changed() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_liveliness_changed() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
on_liveliness_changed() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 415
on_liveliness_changed() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_liveliness_changed() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 473
on_liveliness_changed() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 557
on_liveliness_changed() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 573
on_liveliness_changed() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 620
on_liveliness_changed() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 637
on_liveliness_changed() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 677
on_liveliness_changed() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 693
on_liveliness_lost() (*rti.connextdds.AnyDataWriterListener method*), 49
on_liveliness_lost() (*rti.connextdds.DataWriterListener method*), 127
on_liveliness_lost() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_liveliness_lost() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_liveliness_lost() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_liveliness_lost() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_liveliness_lost() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 410
on_liveliness_lost() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_liveliness_lost() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 468
on_liveliness_lost() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_liveliness_lost() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 568
on_liveliness_lost() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 574
on_liveliness_lost() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_liveliness_lost() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 638
on_liveliness_lost() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 688
on_liveliness_lost() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 695
on_offered_deadline_missed() (*rti.connextdds.AnyDataWriterListener method*), 49
on_offered_deadline_missed() (*rti.connextdds.DataWriterListener method*), 127
on_offered_deadline_missed()

(*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_offered_deadline_missed() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_offered_deadline_missed() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_offered_deadline_missed() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_offered_deadline_missed() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 410
on_offered_deadline_missed() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_offered_deadline_missed() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 468
on_offered_deadline_missed() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_offered_deadline_missed() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 568
on_offered_deadline_missed() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 574
on_offered_deadline_missed() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_offered_deadline_missed() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 638
on_offered_deadline_missed() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 688
on_offered_deadline_missed() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 695
on_offered_incompatible_qos() (*rti.connextdds.AnyDataWriterListener method*), 49
on_offered_incompatible_qos() (*rti.connextdds.DataWriterListener method*), 128
on_offered_incompatible_qos() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_offered_incompatible_qos() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_offered_incompatible_qos() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_offered_incompatible_qos() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_offered_incompatible_qos() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 411
on_offered_incompatible_qos() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_offered_incompatible_qos() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 468
on_offered_incompatible_qos() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_offered_incompatible_qos() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 568
on_offered_incompatible_qos() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 574
on_offered_incompatible_qos() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_offered_incompatible_qos() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 639
on_offered_incompatible_qos() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 688
on_offered_incompatible_qos() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 695
on_publication_matched() (*rti.connextdds.AnyDataWriterListener method*), 49
on_publication_matched() (*rti.connextdds.DataWriterListener method*), 128
on_publication_matched() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_publication_matched() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_publication_matched() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_publication_matched() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_publication_matched() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 411
on_publication_matched() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_publication_matched() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 469
on_publication_matched() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_publication_matched() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 569

on_publication_matched() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 575
on_publication_matched() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_publication_matched() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 639
on_publication_matched() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 689
on_publication_matched() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 695
on_reliable_reader_activity_changed() (*rti.connextdds.AnyDataWriterListener method*), 49
on_reliable_reader_activity_changed() (*rti.connextdds.DataWriterListener method*), 128
on_reliable_reader_activity_changed() (*rti.connextdds.DynamicData.DataWriterListener method*), 213
on_reliable_reader_activity_changed() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_reliable_reader_activity_changed() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_reliable_reader_activity_changed() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_reliable_reader_activity_changed() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 411
on_reliable_reader_activity_changed() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_reliable_reader_activity_changed() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 469
on_reliable_reader_activity_changed() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_reliable_reader_activity_changed() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 569
on_reliable_reader_activity_changed() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 575
on_reliable_reader_activity_changed() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 632
on_reliable_reader_activity_changed() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 639
on_reliable_reader_activity_changed() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 689
on_reliable_reader_activity_changed() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 695
on_reliable_writer_cache_changed() (*rti.connextdds.AnyDataWriterListener method*), 50
on_reliable_writer_cache_changed() (*rti.connextdds.DataWriterListener method*), 128
on_reliable_writer_cache_changed() (*rti.connextdds.DynamicData.DataWriterListener method*), 214
on_reliable_writer_cache_changed() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_reliable_writer_cache_changed() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_reliable_writer_cache_changed() (*rti.connextdds.NoOpDataWriterListener method*), 382
on_reliable_writer_cache_changed() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 411
on_reliable_writer_cache_changed() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 417
on_reliable_writer_cache_changed() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 469
on_reliable_writer_cache_changed() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 475
on_reliable_writer_cache_changed() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 569
on_reliable_writer_cache_changed() (*rti.connextdds.ServiceRequest.NoOpDataWriterListener method*), 575
on_reliable_writer_cache_changed() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 633
on_reliable_writer_cache_changed() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 639
on_reliable_writer_cache_changed() (*rti.con-*

*nextdds.TopicBuiltinTopicData.DataWriterListener method*), 689
on_reliable_writer_cache_changed() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 695
on_requested_deadline_missed() (*rti.connextdds.AnyDataReaderListener method*), 46
on_requested_deadline_missed() (*rti.connextdds.DataReaderListener method*), 100
on_requested_deadline_missed() (*rti.connextdds.DynamicData.DataReaderListener method*), 202
on_requested_deadline_missed() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 220
on_requested_deadline_missed() (*rti.connextdds.NoOpAnyDataReaderListener method*), 379
on_requested_deadline_missed() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_requested_deadline_missed() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
on_requested_deadline_missed() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 416
on_requested_deadline_missed() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_requested_deadline_missed() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 473
on_requested_deadline_missed() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 558
on_requested_deadline_missed() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 573
on_requested_deadline_missed() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 620
on_requested_deadline_missed() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 637
on_requested_deadline_missed() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 677
on_requested_deadline_missed() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 693
on_requested_incompatible_qos() (*rti.connextdds.AnyDataReaderListener method*), 46
on_requested_incompatible_qos() (*rti.connextdds.DataReaderListener method*), 100
on_requested_incompatible_qos() (*rti.connextdds.DynamicData.DataReaderListener method*), 202
on_requested_incompatible_qos() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 220
on_requested_incompatible_qos() (*rti.connextdds.NoOpAnyDataReaderListener method*), 379
on_requested_incompatible_qos() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_requested_incompatible_qos() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
on_requested_incompatible_qos() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 416
on_requested_incompatible_qos() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_requested_incompatible_qos() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 474
on_requested_incompatible_qos() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 558
on_requested_incompatible_qos() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 573
on_requested_incompatible_qos() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 621
on_requested_incompatible_qos() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 637
on_requested_incompatible_qos() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 677
on_requested_incompatible_qos() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 694
on_sample_lost() (*rti.connextdds.AnyDataReaderListener method*), 46
on_sample_lost() (*rti.connextdds.DataReaderListener method*), 100
on_sample_lost() (*rti.connextdds.DynamicData.DataReaderListener*

*method*), 203
on_sample_lost() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 220
on_sample_lost() (*rti.connextdds.NoOpAnyDataReaderListener method*), 379
on_sample_lost() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_sample_lost() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
on_sample_lost() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 416
on_sample_lost() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_sample_lost() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 474
on_sample_lost() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 558
on_sample_lost() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 574
on_sample_lost() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 621
on_sample_lost() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 637
on_sample_lost() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 677
on_sample_lost() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 694
on_sample_rejected() (*rti.connextdds.AnyDataReaderListener method*), 46
on_sample_rejected() (*rti.connextdds.DataReaderListener method*), 100
on_sample_rejected() (*rti.connextdds.DynamicData.DataReaderListener method*), 203
on_sample_rejected() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 220
on_sample_rejected() (*rti.connextdds.NoOpAnyDataReaderListener method*), 379
on_sample_rejected() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_sample_rejected() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 399
on_sample_rejected() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 416
on_sample_rejected() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_sample_rejected() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 474
on_sample_rejected() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 558
on_sample_rejected() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 574
on_sample_rejected() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 621
on_sample_rejected() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 638
on_sample_rejected() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 678
on_sample_rejected() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 694
on_service_request_accepted() (*rti.connextdds.AnyDataWriterListener method*), 50
on_service_request_accepted() (*rti.connextdds.DataWriterListener method*), 128
on_service_request_accepted() (*rti.connextdds.DynamicData.DataWriterListener method*), 214
on_service_request_accepted() (*rti.connextdds.DynamicData.NoOpDataWriterListener method*), 221
on_service_request_accepted() (*rti.connextdds.NoOpAnyDataWriterListener method*), 380
on_service_request_accepted() (*rti.connextdds.NoOpDataWriterListener method*), 383
on_service_request_accepted() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener method*), 411
on_service_request_accepted() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 418
on_service_request_accepted() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterListener method*), 469
on_service_request_accepted() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 476
on_service_request_accepted() (*rti.connextdds.ServiceRequest.DataWriterListener method*), 569
on_service_request_accepted() (*rti.connextdds.Ser-*

*viceRequest.NoOpDataWriterListener method*), 575
on_service_request_accepted() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 633
on_service_request_accepted() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 639
on_service_request_accepted() (*rti.connextdds.TopicBuiltinTopicData.DataWriterListener method*), 689
on_service_request_accepted() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 696
on_subscription_matched() (*rti.connextdds.AnyDataReaderListener method*), 46
on_subscription_matched() (*rti.connextdds.DataReaderListener method*), 100
on_subscription_matched() (*rti.connextdds.DynamicData.DataReaderListener method*), 203
on_subscription_matched() (*rti.connextdds.DynamicData.NoOpDataReaderListener method*), 220
on_subscription_matched() (*rti.connextdds.NoOpAnyDataReaderListener method*), 379
on_subscription_matched() (*rti.connextdds.NoOpDataReaderListener method*), 381
on_subscription_matched() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener method*), 400
on_subscription_matched() (*rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 416
on_subscription_matched() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderListener method*), 457
on_subscription_matched() (*rti.connextdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 474
on_subscription_matched() (*rti.connextdds.ServiceRequest.DataReaderListener method*), 558
on_subscription_matched() (*rti.connextdds.ServiceRequest.NoOpDataReaderListener method*), 574
on_subscription_matched() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 621
on_subscription_matched() (*rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 638
on_subscription_matched() (*rti.connextdds.TopicBuiltinTopicData.DataReaderListener method*), 678
on_subscription_matched() (*rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 694
operation() (*in module rti.rpc*), 810
optional (*rti.connextdds.Member property*), 364
ordered_access (*rti.connextdds.Presentation property*), 436
ordinal (*rti.connextdds.EnumMember property*), 274
original_publication_virtual_guid (*rti.connextdds.SampleInfo property*), 532
original_publication_virtual_sample_identity (*rti.connextdds.SampleInfo property*), 532
original_publication_virtual_sequence_number (*rti.connextdds.SampleInfo property*), 532
original_related_reader_guid (*rti.connextdds.TopicQueryData property*), 715
out_of_range_rejected_sample_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
OutOfResourcesError, 385
output_file() (*rti.connextdds.Logger method*), 361
output_file_set() (*rti.connextdds.Logger method*), 362
output_handler() (*rti.connextdds.Logger method*), 362
output_handler_extended() (*rti.connextdds.Logger method*), 362
outstanding_asynchronous_sample_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 181
Ownership (*class in rti.connextdds*), 385
ownership (*rti.connextdds.DataReaderQos property*), 109
ownership (*rti.connextdds.DataWriterQos property*), 138
ownership (*rti.connextdds.PublicationBuiltinTopicData property*), 484
ownership (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
ownership (*rti.connextdds.TopicBuiltinTopicData property*), 703
ownership (*rti.connextdds.TopicQos property*), 714
ownership_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 96
ownership_strength (*rti.connextdds.DataWriterQos property*), 138
ownership_strength (*rti.connextdds.PublicationBuiltinTopicData property*), 484
OwnershipKind (*class in rti.connextdds*), 386
OwnershipKind.OwnershipKind (*class in rti.connextdds*), 386
OwnershipStrength (*class in rti.connextdds*), 388

## P

parameter_count (*rti.connextdds.Filter property*), 281
PARAMETER_LENGTH_WITH_PADDING_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
parameters (*rti.connextdds.Query property*), 508
parameters (*rti.connextdds.QueryCondition property*), 508
parameters_length (*rti.connextdds.Query property*), 508
parameters_length (*rti.connextdds.QueryCondition property*), 509

parent (*rti.connextdds.StructType property*), 604
partial_configuration (*rti.connextdds.ParticipantBuiltinTopicData property*), 425
participant (*rti.connextdds.DynamicData.ITopicDescription property*), 217
participant (*rti.connextdds.FlowController property*), 291
participant (*rti.connextdds.ITopicDescription property*), 317
participant (*rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription property*), 414
participant (*rti.connextdds.PublicationBuiltinTopicData.ITopicDescription property*), 472
participant (*rti.connextdds.Publisher property*), 495
participant (*rti.connextdds.ServiceRequest.ITopicDescription property*), 572
participant (*rti.connextdds.Subscriber property*), 605
participant (*rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription property*), 636
participant (*rti.connextdds.TopicBuiltinTopicData.ITopicDescription property*), 692
participant (*rti.logging.distlog.LoggerOptions property*), 814
participant_announcement_period (*rti.connextdds.DiscoveryConfig property*), 156
participant_factory_qos (*rti.connextdds.DomainParticipant attribute*), 169
participant_factory_qos (*rti.connextdds.QosProvider property*), 504
participant_factory_qos_from_profile() (*rti.connextdds.QosProvider method*), 504
participant_id (*rti.connextdds.WireProtocol property*), 784
participant_id_gain (*rti.connextdds.RtpsWellKnownPorts property*), 527
participant_key (*rti.connextdds.PublicationBuiltinTopicData property*), 484
participant_key (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
participant_liveliness_assert_period (*rti.connextdds.DiscoveryConfig property*), 156
participant_liveliness_lease_duration (*rti.connextdds.DiscoveryConfig property*), 156
participant_message_reader (*rti.connextdds.DiscoveryConfig property*), 156
participant_message_reader_reliability_kind (*rti.connextdds.DiscoveryConfig property*), 156
participant_message_writer (*rti.connextdds.DiscoveryConfig property*), 156
participant_name (*rti.connextdds.DomainParticipantConfigParams property*), 171
participant_name (*rti.connextdds.DomainParticipantQos property*), 177
participant_name (*rti.connextdds.ParticipantBuiltinTopicData property*), 425
participant_property_list_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 181
participant_property_string_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 181
participant_protocol_status (*rti.connextdds.DomainParticipant property*), 169
participant_qos (*rti.connextdds.QosProvider property*), 504
participant_qos_from_profile() (*rti.connextdds.QosProvider method*), 505
participant_qos_library_name (*rti.connextdds.DomainParticipantConfigParams property*), 172
participant_qos_profile_name (*rti.connextdds.DomainParticipantConfigParams property*), 172
participant_qos_profile_name (*rti.connextdds.MonitoringDedicatedParticipantSettings property*), 368
participant_reader (*rti.connextdds.DomainParticipant property*), 169
participant_reader_resource_limits (*rti.connextdds.DiscoveryConfig property*), 156
participant_user_data_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 181
ParticipantBuiltinTopicData (*class in rti.connextdds*), 388
ParticipantBuiltinTopicData.ContentFilter (*class in rti.connextdds*), 388
ParticipantBuiltinTopicData.ContentFilteredTopic (*class in rti.connextdds*), 389
ParticipantBuiltinTopicData.ContentFilteredTopicSeq (*class in rti.connextdds*), 390
ParticipantBuiltinTopicData.DataReader (*class in rti.connextdds*), 392
ParticipantBuiltinTopicData.DataReaderListener (*class in rti.connextdds*), 399
ParticipantBuiltinTopicData.DataReader.Selector (*class in rti.connextdds*), 392
ParticipantBuiltinTopicData.DataReaderSeq (*class in rti.connextdds*), 400
ParticipantBuiltinTopicData.DataWriter (*class in rti.connextdds*), 402
ParticipantBuiltinTopicData.DataWriterListener (*class in rti.connextdds*), 410
ParticipantBuiltinTopicData.DataWriterSeq (*class in rti.connextdds*), 411
ParticipantBuiltinTopicData.ITopicDescription (*class in rti.connextdds*),

414
ParticipantBuiltinTopicData.LoanedSample (*class in rti.connextdds*), 414
ParticipantBuiltinTopicData.LoanedSamples (*class in rti.connextdds*), 414
ParticipantBuiltinTopicData.NoOpDataReaderListener (*class in rti.connextdds*), 415
ParticipantBuiltinTopicData.NoOpDataWriterListener (*class in rti.connextdds*), 416
ParticipantBuiltinTopicData.NoOpTopicListener (*class in rti.connextdds*), 418
ParticipantBuiltinTopicData.Sample (*class in rti.connextdds*), 418
ParticipantBuiltinTopicDataSeq (*class in rti.connextdds*), 426
ParticipantBuiltinTopicData.SharedSamples (*class in rti.connextdds*), 419
ParticipantBuiltinTopicDataTimestampedSeq (*class in rti.connextdds*), 428
ParticipantBuiltinTopicData.Topic (*class in rti.connextdds*), 419
ParticipantBuiltinTopicData.TopicDescription (*class in rti.connextdds*), 420
ParticipantBuiltinTopicData.TopicListener (*class in rti.connextdds*), 421
ParticipantBuiltinTopicData.TopicSeq (*class in rti.connextdds*), 421
ParticipantBuiltinTopicData.ValidLoanedSamples (*class in rti.connextdds*), 423
ParticipantBuiltinTopicData.WriterContentFilter (*class in rti.connextdds*), 423
ParticipantBuiltinTopicData.WriterContentFilterHelper (*class in rti.connextdds*), 424
Partition (*class in rti.connextdds*), 431
partition (*rti.connextdds.DomainParticipantQos property*), 178
partition (*rti.connextdds.PublicationBuiltinTopicData property*), 484
partition (*rti.connextdds.PublisherQos property*), 497
partition (*rti.connextdds.SubscriberQos property*), 607
partition (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
pattern_alarm_event (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_alarm_status (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_event (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_last_value_cache (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_periodic_data (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_reliable_streaming (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_rpc (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_status (*rti.connextdds.BuiltinProfiles attribute*), 62
pattern_streaming (*rti.connextdds.BuiltinProfiles attribute*), 62
pause() (*in module rti.connextdds.heap_monitoring*), 803
period (*rti.connextdds.Deadline property*), 148
period (*rti.connextdds.FlowControllerTokenBucketProperty property*), 296
periodic_settings (*rti.connextdds.MonitoringDistributionSettings property*), 368
PERSIST (*rti.connextdds.PersistentJournalKind attribute*), 432
PERSISTENCE (*rti.connextdds.ServiceKind attribute*), 545
PERSISTENCE (*rti.connextdds.ServiceKind.ServiceKind attribute*), 545
persistent (*rti.connextdds.Durability attribute*), 186
PERSISTENT (*rti.connextdds.DurabilityKind attribute*), 188
PERSISTENT (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 187
PersistentJournalKind (*class in rti.connextdds*), 432
PersistentStorageSettings (*class in rti.connextdds*), 433
PersistentSynchronizationKind (*class in rti.connextdds*), 434
PingType (*class in rti.types.builtin*), 804
PL_CDR_BIG_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 255
PL_CDR_BIG_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 254
PL_CDR_LITTLE_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind attribute*), 255
PL_CDR_LITTLE_ENDIAN (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 254
platform (*rti.connextdds.LogCategory attribute*), 357
platform (*rti.connextdds.LogCategory.LogCategory attribute*), 356
pointer (*rti.connextdds.Member property*), 364
pointer (*rti.connextdds.UnionMember property*), 768
policies (*rti.connextdds.OfferedIncompatibleQosStatus property*), 385
policies (*rti.connextdds.RequestedIncompatibleQosStatus property*), 519
policy (*rti.connextdds.QosPolicyCount property*), 500
polling_period (*rti.connextdds.MonitoringPeriodicDistributionSettings property*), 374
pop() (*rti.connextdds.AnyDataReaderSeq method*), 48
pop() (*rti.connextdds.AnyDataWriterSeq method*), 52
pop() (*rti.connextdds.AnyTopicSeq method*), 54
pop() (*rti.connextdds.BoolSeq method*), 60
pop() (*rti.connextdds.ChannelSettingsSeq method*), 70
pop() (*rti.connextdds.CharSeq method*), 72
pop() (*rti.connextdds.ConditionSeq method*), 80
pop() (*rti.connextdds.ContentFilteredTopicSeq method*), 84

pop() (*rti.connextdds.CookieSeq method*), 87
pop() (*rti.connextdds.DataReaderSeq method*), 115
pop() (*rti.connextdds.DataWriterSeq method*), 146
pop() (*rti.connextdds.DomainParticipantSeq method*), 185
pop() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 196
pop() (*rti.connextdds.DynamicData.DataReaderSeq method*), 205
pop() (*rti.connextdds.DynamicData.DataWriterSeq method*), 216
pop() (*rti.connextdds.DynamicDataSeq method*), 260
pop() (*rti.connextdds.DynamicDataTimestampedSeq method*), 263
pop() (*rti.connextdds.DynamicData.TopicSeq method*), 227
pop() (*rti.connextdds.EndpointGroupSeq method*), 268
pop() (*rti.connextdds.EntitySeq method*), 273
pop() (*rti.connextdds.EnumMemberSeq method*), 276
pop() (*rti.connextdds.Float32Seq method*), 287
pop() (*rti.connextdds.Float64Seq method*), 289
pop() (*rti.connextdds.Float128Seq method*), 284
pop() (*rti.connextdds.IAnyDataReaderSeq method*), 304
pop() (*rti.connextdds.IAnyDataWriterSeq method*), 307
pop() (*rti.connextdds.IAnyTopicSeq method*), 310
pop() (*rti.connextdds.IConditionSeq method*), 312
pop() (*rti.connextdds.IEntitySeq method*), 316
pop() (*rti.connextdds.InstanceHandleSeq method*), 322
pop() (*rti.connextdds.Int8Seq method*), 337
pop() (*rti.connextdds.Int16Seq method*), 328
pop() (*rti.connextdds.Int32Seq method*), 330
pop() (*rti.connextdds.Int64Seq method*), 334
pop() (*rti.connextdds.LocatorFilterElementSeq method*), 348
pop() (*rti.connextdds.LocatorSeq method*), 353
pop() (*rti.connextdds.MemberSeq method*), 366
pop() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
pop() (*rti.connextdds.MulticastMappingSeq method*), 378
pop() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
pop() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 402
pop() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 413
pop() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
pop() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 431
pop() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 423
pop() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 450
pop() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 460
pop() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 471
pop() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
pop() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 490
pop() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 481
pop() (*rti.connextdds.PublisherSeq method*), 499
pop() (*rti.connextdds.QosPolicyCountSeq method*), 502
pop() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 551
pop() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
pop() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
pop() (*rti.connextdds.ServiceRequestSeq method*), 587
pop() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 590
pop() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
pop() (*rti.connextdds.StringPairSeq method*), 600
pop() (*rti.connextdds.StringSeq method*), 602
pop() (*rti.connextdds.SubscriberSeq method*), 609
pop() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 613
pop() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
pop() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
pop() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 651
pop() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 654
pop() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 645
pop() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 670
pop() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 680
pop() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
pop() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
pop() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 709
pop() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 701
pop() (*rti.connextdds.TopicSeq method*), 721
pop() (*rti.connextdds.TransportInfoSeq method*), 730
pop() (*rti.connextdds.TransportMulticastSeq method*), 737
pop() (*rti.connextdds.TransportMulticastSettingsSeq method*), 740

pop() (*rti.connextdds.TransportUnicastSettingsSeq method*), 745
pop() (*rti.connextdds.Uint8Seq method*), 766
pop() (*rti.connextdds.Uint16Seq method*), 758
pop() (*rti.connextdds.Uint32Seq method*), 761
pop() (*rti.connextdds.Uint64Seq method*), 764
pop() (*rti.connextdds.UnionMemberSeq method*), 770
pop() (*rti.connextdds.WcharSeq method*), 783
pop() (*rti.connextdds.WstringSeq method*), 790
port (*rti.connextdds.Locator property*), 345
port_base (*rti.connextdds.RtpsWellKnownPorts property*), 527
PreconditionNotMetError, 435
Presentation (*class in rti.connextdds*), 435
presentation (*rti.connextdds.PublicationBuiltinTopicData property*), 484
presentation (*rti.connextdds.PublisherQos property*), 497
presentation (*rti.connextdds.SubscriberQos property*), 607
presentation (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
PresentationAccessScopeKind (*class in rti.connextdds*), 436
PresentationAccessScopeKind.PresentationAccessScopeKind (*class in rti.connextdds*), 436
pretty_print (*rti.connextdds.PrintFormatProperty property*), 443
prevent_type_widening (*rti.connextdds.TypeConsistencyEnforcement property*), 747
PRIMITIVE_TYPE (*rti.connextdds.TypeKind attribute*), 751
PRIMITIVE_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
print_format (*rti.connextdds.Logger property*), 362
print_format() (*rti.logging.distlog.Logger static method*), 814
print_idl() (*rti.connextdds.DynamicType method*), 265
print_private (*rti.connextdds.QosPrintFormat property*), 503
PrintFormat (*class in rti.connextdds*), 438
PrintFormatKind (*class in rti.connextdds*), 441
PrintFormatKind.PrintFormatKind (*class in rti.connextdds*), 441
PrintFormat.PrintFormat (*class in rti.connextdds*), 439
PrintFormatProperty (*class in rti.connextdds*), 443
priority (*rti.connextdds.ChannelSettings property*), 68
priority (*rti.connextdds.FilterSampleInfo property*), 282
priority (*rti.connextdds.PublishMode property*), 491
priority (*rti.connextdds.ThreadSettings property*), 658
priority (*rti.connextdds.WriteParams property*), 787
PRIORITY_ENFORCE (*rti.connextdds.ThreadSettingsKindMask attribute*), 660
product_version (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
product_version (*rti.connextdds.PublicationBuiltinTopicData property*), 484
product_version (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
ProductVersion (*class in rti.connextdds*), 443
profiles_loaded (*rti.connextdds.QosProvider property*), 505
propagate() (*rti.connextdds.Property method*), 445
propagate_app_ack_with_no_response (*rti.connextdds.DataWriterProtocol property*), 129
propagate_dispose_of_unregistered_instances (*rti.connextdds.DataReaderProtocol property*), 101
propagate_unregister_of_disposed_instances (*rti.connextdds.DataReaderProtocol property*), 101
Property (*class in rti.connextdds*), 444
property (*rti.connextdds.DataReaderQos property*), 109
property (*rti.connextdds.DataWriterQos property*), 138
property (*rti.connextdds.DomainParticipantQos property*), 178
property (*rti.connextdds.FlowController property*), 291
property (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
property (*rti.connextdds.PublicationBuiltinTopicData property*), 484
property (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
property (*rti.connextdds.WaitSet property*), 780
PROTOCOL (*rti.connextdds.AcknowledgmentKind attribute*), 40
PROTOCOL (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 39
protocol (*rti.connextdds.DataReaderQos property*), 109
protocol (*rti.connextdds.DataWriterQos property*), 138
ProtocolVersion (*class in rti.connextdds*), 446
provider_params (*rti.connextdds.QosProvider property*), 505
publication_handle (*rti.connextdds.SampleInfo property*), 532
PUBLICATION_MATCHED (*rti.connextdds.StatusMask attribute*), 592
publication_matched_status (*rti.connextdds.DataWriter property*), 124
publication_matched_status (*rti.connextdds.DynamicData.DataWriter property*), 209
publication_matched_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
publication_matched_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
publication_matched_status (*rti.connextdds.ServiceRequest.DataWriter property*), 564
publication_matched_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
publication_matched_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
publication_name

(*rti.connextdds.PublicationBuiltinTopicData property*), 484
publication_period (*rti.connextdds.MonitoringEventDistributionSettings property*), 369
publication_period (*rti.connextdds.MonitoringLoggingDistributionSettings property*), 370
publication_period (*rti.connextdds.TopicQueryDispatch property*), 716
PUBLICATION_PRIORITY_UNDEFINED (*rti.connextdds.PublishMode attribute*), 491
publication_reader (*rti.connextdds.DiscoveryConfig property*), 156
publication_reader (*rti.connextdds.DomainParticipant property*), 169
publication_reader_resource_limits (*rti.connextdds.DiscoveryConfig property*), 157
publication_sequence_number (*rti.connextdds.SampleInfo property*), 532
publication_writer (*rti.connextdds.DiscoveryConfig property*), 157
publication_writer_data_lifecycle (*rti.connextdds.DiscoveryConfig property*), 157
publication_writer_publish_mode (*rti.connextdds.DiscoveryConfig property*), 157
PublicationBuiltinTopicData (*class in rti.connextdds*), 446
PublicationBuiltinTopicData.ContentFilter (*class in rti.connextdds*), 446
PublicationBuiltinTopicData.ContentFilteredTopic (*class in rti.connextdds*), 447
PublicationBuiltinTopicData.ContentFilteredTopicSeq (*class in rti.connextdds*), 448
PublicationBuiltinTopicData.DataReader (*class in rti.connextdds*), 450
PublicationBuiltinTopicData.DataReaderListener (*class in rti.connextdds*), 457
PublicationBuiltinTopicData.DataReader.Selector (*class in rti.connextdds*), 450
PublicationBuiltinTopicData.DataReaderSeq (*class in rti.connextdds*), 458
PublicationBuiltinTopicData.DataWriter (*class in rti.connextdds*), 460
PublicationBuiltinTopicData.DataWriterListener (*class in rti.connextdds*), 468
PublicationBuiltinTopicData.DataWriterSeq (*class in rti.connextdds*), 469
PublicationBuiltinTopicData.ITopicDescription (*class in rti.connextdds*), 472
PublicationBuiltinTopicData.LoanedSample (*class in rti.connextdds*), 472
PublicationBuiltinTopicData.LoanedSamples (*class in rti.connextdds*), 472
PublicationBuiltinTopicData.NoOpDataReaderListener (*class in rti.connextdds*), 473
PublicationBuiltinTopicData.NoOpDataWriterListener (*class in rti.connextdds*), 474
PublicationBuiltinTopicData.NoOpTopicListener (*class in rti.connextdds*), 476
PublicationBuiltinTopicData.Sample (*class in rti.connextdds*), 476
PublicationBuiltinTopicDataSeq (*class in rti.connextdds*), 485
PublicationBuiltinTopicData.SharedSamples (*class in rti.connextdds*), 477
PublicationBuiltinTopicDataTimestampedSeq (*class in rti.connextdds*), 487
PublicationBuiltinTopicData.Topic (*class in rti.connextdds*), 477
PublicationBuiltinTopicData.TopicDescription (*class in rti.connextdds*), 478
PublicationBuiltinTopicData.TopicListener (*class in rti.connextdds*), 478
PublicationBuiltinTopicData.TopicSeq (*class in rti.connextdds*), 479
PublicationBuiltinTopicData.ValidLoanedSamples (*class in rti.connextdds*), 481
PublicationBuiltinTopicData.WriterContentFilter (*class in rti.connextdds*), 481
PublicationBuiltinTopicData.WriterContentFilterHelper (*class in rti.connextdds*), 482
PublicationMatchedStatus (*class in rti.connextdds*), 490
publish_mode (*rti.connextdds.DataWriterQos property*), 138
Publisher (*class in rti.connextdds*), 494
publisher (*rti.connextdds.DataWriter property*), 124
publisher (*rti.connextdds.DynamicData.DataWriter property*), 209
publisher (*rti.connextdds.IAnyDataWriter property*), 305
publisher (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
publisher (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
publisher (*rti.connextdds.ServiceRequest.DataWriter property*), 564
publisher (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
publisher (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
publisher_group_data_max_length (*rti.connextdds.DomainParticipantResourceLimits*

*property*), 181
publisher_key (*rti.connextdds.PublicationBuiltinTopicData property*), 484
publisher_qos (*rti.connextdds.QosProvider property*), 505
publisher_qos_from_profile() (*rti.connextdds.QosProvider method*), 505
publisher_qos_profile_name (*rti.connextdds.MonitoringDistributionSettings property*), 368
PublisherListener (*class in rti.connextdds*), 495
PublisherQos (*class in rti.connextdds*), 495
PublisherSeq (*class in rti.connextdds*), 497
PublishMode (*class in rti.connextdds*), 491
PublishModeKind (*class in rti.connextdds*), 491
PublishModeKind.PublishModeKind (*class in rti.connextdds*), 492
pulled_fragment_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
pulled_fragment_count (*rti.connextdds.DataWriterProtocolStatus property*), 130
pulled_sample_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
pulled_sample_count (*rti.connextdds.DataWriterProtocolStatus property*), 130
push_on_write (*rti.connextdds.DataWriterProtocol property*), 129
pushed_fragment_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
pushed_fragment_count (*rti.connextdds.DataWriterProtocolStatus property*), 130
pushed_sample_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
pushed_sample_count (*rti.connextdds.DataWriterProtocolStatus property*), 130
PYTHON_INT8_730_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794

## Q

qos (*rti.connextdds.DataReader property*), 94
qos (*rti.connextdds.DataWriter property*), 124
qos (*rti.connextdds.DomainParticipant property*), 169
qos (*rti.connextdds.DynamicData.DataReader property*), 200
qos (*rti.connextdds.DynamicData.DataWriter property*), 209
qos (*rti.connextdds.DynamicData.Topic property*), 224
qos (*rti.connextdds.IAnyDataReader property*), 302
qos (*rti.connextdds.IAnyDataWriter property*), 305
qos (*rti.connextdds.IAnyTopic property*), 308
qos (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 397
qos (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 406
qos (*rti.connextdds.ParticipantBuiltinTopicData.Topic property*), 420
qos (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 455
qos (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 464
qos (*rti.connextdds.PublicationBuiltinTopicData.Topic property*), 478
qos (*rti.connextdds.Publisher property*), 495
qos (*rti.connextdds.ServiceRequest.DataReader property*), 556
qos (*rti.connextdds.ServiceRequest.DataWriter property*), 564
qos (*rti.connextdds.ServiceRequest.Topic property*), 577
qos (*rti.connextdds.Subscriber property*), 605
qos (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
qos (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
qos (*rti.connextdds.SubscriptionBuiltinTopicData.Topic property*), 642
qos (*rti.connextdds.Topic property*), 666
qos (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 675
qos (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 684
qos (*rti.connextdds.TopicBuiltinTopicData.Topic property*), 698
QOS_ELEMENT_NAME_USE_XML_CONFIG (*rti.connextdds.DomainParticipantConfigParams attribute*), 171
qos_library (*rti.logging.distlog.LoggerOptions property*), 815
qos_profile (*rti.logging.distlog.LoggerOptions property*), 815
qos_profile_libraries (*rti.connextdds.QosProvider property*), 505
qos_profiles() (*rti.connextdds.QosProvider method*), 505
QosPolicyCount (*class in rti.connextdds*), 499
QosPolicyCountSeq (*class in rti.connextdds*), 500
QosPrintFormat (*class in rti.connextdds*), 502
QosProvider (*class in rti.connextdds*), 503
QosProviderParams (*class in rti.connextdds*), 506
Query (*class in rti.connextdds*), 507
query_condition_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 181
QueryCondition (*class in rti.connextdds*), 508
queue_size (*rti.logging.distlog.LoggerOptions property*), 815
QUEUING (*rti.connextdds.ServiceKind attribute*), 545
QUEUING (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
quorum_count (*rti.connextdds.EndpointGroup property*), 266

## R

Rank (*class in rti.connextdds*), 509
rank (*rti.connextdds.SampleInfo property*), 532
reachability_lease_duration (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
READ (*rti.connextdds.SampleState attribute*), 540

read() (*rti.connextdds.DataReader method*), 94
read() (*rti.connextdds.DataReader.Selector method*), 90
read() (*rti.connextdds.DynamicData.DataReader method*), 200
read() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
read() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
read() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
read() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 455
read() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
read() (*rti.connextdds.ServiceRequest.DataReader method*), 556
read() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
read() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 619
read() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 614
read() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
read() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
read_condition_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 181
read_data() (*rti.connextdds.DataReader method*), 94
read_data() (*rti.connextdds.DataReader.Selector method*), 90
read_data() (*rti.connextdds.DynamicData.DataReader method*), 200
read_data() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
read_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
read_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
read_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 455
read_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
read_data() (*rti.connextdds.ServiceRequest.DataReader method*), 556
read_data() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
read_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 619
read_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 615
read_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
read_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
read_loaned() (*rti.connextdds.DataReader method*), 94
read_loaned() (*rti.connextdds.DataReader.Selector method*), 90
read_loaned() (*rti.connextdds.DynamicData.DataReader method*), 201
read_loaned() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
read_loaned() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 397
read_loaned() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
read_loaned() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 455
read_loaned() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
read_loaned() (*rti.connextdds.ServiceRequest.DataReader method*), 556
read_loaned() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
read_loaned() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 619
read_loaned() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 615
read_loaned() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 675
read_loaned() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 671
ReadCondition (*class in rti.connextdds*), 509
reader_checkpoint_frequency (*rti.connextdds.PersistentStorageSettings property*), 433
reader_data_lifecycle (*rti.connextdds.DataReaderQos property*), 109
reader_data_tag_list_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 181
reader_data_tag_string_max_length

(*rti.connextdds.DomainParticipantResourceLimits property*), 181
reader_property_list_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 181
reader_property_string_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 182
reader_resource_limits (*rti.connextdds.DataReaderQos property*), 109
reader_user_data_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 182
ReaderDataLifecycle (*class in rti.connextdds*), 510
REALTIME_PRIORITY (*rti.connextdds.ThreadSettingsKindMask attribute*), 660
reassembled_sample_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
receive_address (*rti.connextdds.TransportMulticastSettings property*), 738
receive_port (*rti.connextdds.TransportMulticastSettings property*), 738
receive_port (*rti.connextdds.TransportUnicastSettings property*), 743
receive_window_size (*rti.connextdds.RtpsReliableReaderProtocol property*), 521
received_ack_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
received_ack_count (*rti.connextdds.DataWriterProtocolStatus property*), 130
received_fragment_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
received_gap_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 102
received_gap_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
received_heartbeat_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 102
received_heartbeat_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
received_nack_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
received_nack_count (*rti.connextdds.DataWriterProtocolStatus property*), 130
received_nack_fragment_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 130
received_nack_fragment_count (*rti.connextdds.DataWriterProtocolStatus property*), 131
received_sample_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 102
received_sample_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
receiver_pool (*rti.connextdds.DomainParticipantQos property*), 178
ReceiverPool (*class in rti.connextdds*), 511
reception_sequence_number (*rti.connextdds.SampleInfo property*), 532
reception_timestamp (*rti.connextdds.SampleInfo property*), 532
RECORDING (*rti.connextdds.ServiceKind attribute*), 545
RECORDING (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
RECOVER_STATE (*rti.connextdds.InstanceStateConsistencyKind attribute*), 325
REDELIVERED (*rti.connextdds.SampleFlag attribute*), 528
register_contentfilter() (*rti.connextdds.DomainParticipant method*), 170
register_durable_subscription() (*rti.connextdds.DomainParticipant method*), 170
register_idl_type() (*rti.connextdds.DomainParticipant static method*), 170
register_instance() (*rti.connextdds.DataWriter method*), 124
register_instance() (*rti.connextdds.DynamicData.DataWriter method*), 209
register_instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 406
register_instance() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 464
register_instance() (*rti.connextdds.ServiceRequest.DataWriter method*), 565
register_instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 628
register_instance() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 684
register_type() (*rti.connextdds.DomainParticipant method*), 170
register_type() (*rti.connextdds.DynamicData.TopicTypeSupport static method*), 227
REJECTED_BY_DECODE_FAILURE (*rti.connextdds.SampleRejectedState attribute*), 536
REJECTED_BY_INSTANCES_LIMIT (*rti.connextdds.SampleRejectedState attribute*), 536
REJECTED_BY_REMOTE_WRITERS_PER_VIRTUAL_QUEUE_LIMIT

(*rti.connextdds.SampleRejectedState attribute*), 537
REJECTED_BY_SAMPLES_LIMIT (*rti.connextdds.SampleRejectedState attribute*), 537
REJECTED_BY_SAMPLES_PER_INSTANCE_LIMIT (*rti.connextdds.SampleRejectedState attribute*), 537
REJECTED_BY_SAMPLES_PER_REMOTE_WRITER_LIMIT (*rti.connextdds.SampleRejectedState attribute*), 537
rejected_sample_count (*rti.connextdds.DataReaderProtocolStatus property*), 102
rejected_sample_count (*rti.connextdds.DataWriterProtocolStatus property*), 131
related_original_publication_virtual_guid (*rti.connextdds.SampleInfo property*), 532
related_original_publication_virtual_sample_identity (*rti.connextdds.SampleInfo property*), 532
related_original_publication_virtual_sequence_number (*rti.connextdds.SampleInfo property*), 532
related_reader_guid (*rti.connextdds.FilterSampleInfo property*), 282
related_reader_guid (*rti.connextdds.WriteParams property*), 787
related_sample_identity (*rti.connextdds.FilterSampleInfo property*), 282
related_sample_identity (*rti.connextdds.WriteParams property*), 787
related_source_guid (*rti.connextdds.FilterSampleInfo property*), 282
related_source_guid (*rti.connextdds.SampleInfo property*), 532
related_source_guid (*rti.connextdds.WriteParams property*), 787
related_subscription_guid (*rti.connextdds.SampleInfo property*), 532
related_topic_name (*rti.connextdds.ContentFilterProperty property*), 81
related_type() (*rti.connextdds.AliasType method*), 44
release_version (*rti.connextdds.ProductVersion property*), 444
Reliability (*class in rti.connextdds*), 512
reliability (*rti.connextdds.DataReaderQos property*), 109
reliability (*rti.connextdds.DataWriterQos property*), 138
reliability (*rti.connextdds.PublicationBuiltinTopicData property*), 484
reliability (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
reliability (*rti.connextdds.TopicBuiltinTopicData property*), 704
reliability (*rti.connextdds.TopicQos property*), 714
ReliabilityKind (*class in rti.connextdds*), 513
ReliabilityKind.ReliabilityKind (*class in rti.connextdds*), 513
RELIABLE (*rti.connextdds.ReliabilityKind attribute*), 513
RELIABLE (*rti.connextdds.ReliabilityKind.ReliabilityKind attribute*), 513
reliable() (*rti.connextdds.Reliability static method*), 513
RELIABLE_READER_ACTIVITY_CHANGED (*rti.connextdds.StatusMask attribute*), 592
reliable_reader_activity_changed_status (*rti.connextdds.DataWriter property*), 124
reliable_reader_activity_changed_status (*rti.connextdds.DynamicData.DataWriter property*), 210
reliable_reader_activity_changed_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 407
reliable_reader_activity_changed_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 465
reliable_reader_activity_changed_status (*rti.connextdds.ServiceRequest.DataWriter property*), 565
reliable_reader_activity_changed_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
reliable_reader_activity_changed_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 685
RELIABLE_WRITER_CACHE_CHANGED (*rti.connextdds.StatusMask attribute*), 592
reliable_writer_cache_changed_status (*rti.connextdds.DataWriter property*), 124
reliable_writer_cache_changed_status (*rti.connextdds.DynamicData.DataWriter property*), 210
reliable_writer_cache_changed_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 407
reliable_writer_cache_changed_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 465
reliable_writer_cache_changed_status (*rti.connextdds.ServiceRequest.DataWriter property*), 565
reliable_writer_cache_changed_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 628
reliable_writer_cache_changed_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 685
ReliableReaderActivityChangedStatus (*class in rti.connextdds*), 515
ReliableWriterCacheChangedStatus (*class in rti.connextdds*), 515
reload_profiles() (*rti.connextdds.QosProvider method*), 505
remote_administration_enabled (*rti.logging.distlog.LoggerOptions property*), 815
remote_participant_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 182

remote_participant_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 182
remote_participant_purge_kind (*rti.connextdds.DiscoveryConfig property*), 157
remote_reader_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 182
remote_reader_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 182
remote_topic_query_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 182
remote_topic_query_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 182
remote_writer_allocation (*rti.connextdds.DomainParticipantResourceLimits property*), 182
remote_writer_hash_buckets (*rti.connextdds.DomainParticipantResourceLimits property*), 182
RemoteError, 805
RemoteInvalidArgumentError, 806
RemoteOutOfResourcesError, 806
RemoteParticipantPurgeKind (*class in rti.connextdds*), 516
RemoteParticipantPurgeKind.RemoteParticipantPurgeKind (*class in rti.connextdds*), 516
RemoteUnknownExceptionError, 806
RemoteUnknownOperationError, 806
RemoteUnsupportedOperationError, 806
remove() (*rti.connextdds.AnyDataReaderSeq method*), 48
remove() (*rti.connextdds.AnyDataWriterSeq method*), 52
remove() (*rti.connextdds.AnyTopicSeq method*), 54
remove() (*rti.connextdds.BoolSeq method*), 60
remove() (*rti.connextdds.ChannelSettingsSeq method*), 70
remove() (*rti.connextdds.CharSeq method*), 72
remove() (*rti.connextdds.ConditionSeq method*), 80
remove() (*rti.connextdds.ContentFilteredTopicSeq method*), 85
remove() (*rti.connextdds.CookieSeq method*), 87
remove() (*rti.connextdds.DataReaderSeq method*), 115
remove() (*rti.connextdds.DataTag method*), 120
remove() (*rti.connextdds.DataWriterSeq method*), 146
remove() (*rti.connextdds.DomainParticipantSeq method*), 185
remove() (*rti.connextdds.DynamicData.ContentFilteredTopicSeq method*), 196
remove() (*rti.connextdds.DynamicData.DataReaderSeq method*), 205
remove() (*rti.connextdds.DynamicData.DataWriterSeq method*), 216
remove() (*rti.connextdds.DynamicDataSeq method*), 261
remove() (*rti.connextdds.DynamicDataTimestampedSeq method*), 263
remove() (*rti.connextdds.DynamicData.TopicSeq method*), 227
remove() (*rti.connextdds.EndpointGroupSeq method*), 268
remove() (*rti.connextdds.EntitySeq method*), 273
remove() (*rti.connextdds.EnumMemberSeq method*), 276
remove() (*rti.connextdds.Float32Seq method*), 287
remove() (*rti.connextdds.Float64Seq method*), 290
remove() (*rti.connextdds.Float128Seq method*), 284
remove() (*rti.connextdds.IAnyDataReaderSeq method*), 304
remove() (*rti.connextdds.IAnyDataWriterSeq method*), 307
remove() (*rti.connextdds.IAnyTopicSeq method*), 310
remove() (*rti.connextdds.IConditionSeq method*), 312
remove() (*rti.connextdds.IEntitySeq method*), 316
remove() (*rti.connextdds.InstanceHandleSeq method*), 322
remove() (*rti.connextdds.Int8Seq method*), 337
remove() (*rti.connextdds.Int16Seq method*), 328
remove() (*rti.connextdds.Int32Seq method*), 331
remove() (*rti.connextdds.Int64Seq method*), 334
remove() (*rti.connextdds.LocatorFilterElementSeq method*), 348
remove() (*rti.connextdds.LocatorSeq method*), 353
remove() (*rti.connextdds.MemberSeq method*), 366
remove() (*rti.connextdds.MonitoringMetricSelectionSeq method*), 373
remove() (*rti.connextdds.MulticastMappingSeq method*), 378
remove() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 392
remove() (*rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 402
remove() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 414
remove() (*rti.connextdds.ParticipantBuiltinTopicDataSeq method*), 428
remove() (*rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 431
remove() (*rti.connextdds.ParticipantBuiltinTopicData.TopicSeq method*), 423
remove() (*rti.connextdds.Property method*), 445
remove() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 450
remove() (*rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq method*), 460
remove() (*rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq method*), 472
remove() (*rti.connextdds.PublicationBuiltinTopicDataSeq method*), 487
remove() (*rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq method*), 490
remove() (*rti.connextdds.PublicationBuiltinTopicData.TopicSeq method*), 481
remove() (*rti.connextdds.PublisherSeq method*), 499
remove() (*rti.connextdds.QosPolicyCountSeq method*), 502

remove() (*rti.connextdds.ServiceRequest.ContentFilteredTopicSeq method*), 551
remove() (*rti.connextdds.ServiceRequest.DataReaderSeq method*), 560
remove() (*rti.connextdds.ServiceRequest.DataWriterSeq method*), 571
remove() (*rti.connextdds.ServiceRequestSeq method*), 587
remove() (*rti.connextdds.ServiceRequestTimestampedSeq method*), 590
remove() (*rti.connextdds.ServiceRequest.TopicSeq method*), 580
remove() (*rti.connextdds.StringPairSeq method*), 600
remove() (*rti.connextdds.StringSeq method*), 602
remove() (*rti.connextdds.SubscriberSeq method*), 610
remove() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 614
remove() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 623
remove() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 635
remove() (*rti.connextdds.SubscriptionBuiltinTopicDataSeq method*), 651
remove() (*rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 654
remove() (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq method*), 645
remove() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 671
remove() (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq method*), 680
remove() (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq method*), 691
remove() (*rti.connextdds.TopicBuiltinTopicDataSeq method*), 706
remove() (*rti.connextdds.TopicBuiltinTopicDataTimestampedSeq method*), 709
remove() (*rti.connextdds.TopicBuiltinTopicData.TopicSeq method*), 701
remove() (*rti.connextdds.TopicSeq method*), 721
remove() (*rti.connextdds.TransportInfoSeq method*), 731
remove() (*rti.connextdds.TransportMulticastSeq method*), 738
remove() (*rti.connextdds.TransportMulticastSettingsSeq method*), 741
remove() (*rti.connextdds.TransportUnicastSettingsSeq method*), 746
remove() (*rti.connextdds.Uint8Seq method*), 767
remove() (*rti.connextdds.Uint16Seq method*), 758
remove() (*rti.connextdds.Uint32Seq method*), 761
remove() (*rti.connextdds.Uint64Seq method*), 764
remove() (*rti.connextdds.UnionMemberSeq method*), 770
remove() (*rti.connextdds.WcharSeq method*), 783
remove() (*rti.connextdds.WstringSeq method*), 790
remove_from_expression_parameter() (*rti.connextdds.ContentFilteredTopic method*), 82
remove_from_expression_parameter() (*rti.connextdds.DynamicData.ContentFilteredTopic method*), 193
remove_from_expression_parameter() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 390
remove_from_expression_parameter() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 448
remove_from_expression_parameter() (*rti.connextdds.ServiceRequest.ContentFilteredTopic method*), 549
remove_from_expression_parameter() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 611
remove_from_expression_parameter() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 668
remove_peer() (*rti.connextdds.DomainParticipant method*), 170
remove_peers() (*rti.connextdds.DomainParticipant method*), 170
replace_automatic_values (*rti.connextdds.WriteParams property*), 787
replace_empty_instances (*rti.connextdds.DataWriterResourceLimits property*), 140
replaced_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 97
replaced_unacknowledged_sample_count (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 516
REPLAY (*rti.connextdds.ServiceKind attribute*), 545
REPLAY (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
REPLICATE (*rti.connextdds.SampleFlag attribute*), 528
Replier (*class in rti.rpc*), 806
representation (*rti.connextdds.PublicationBuiltinTopicData property*), 484
representation (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
representation (*rti.connextdds.TopicBuiltinTopicData property*), 704
request_body (*rti.connextdds.ServiceRequest property*), 582
request_types_filter (*rti.connextdds.DiscoveryConfig property*), 157
REQUESTED_DEADLINE_MISSED (*rti.connextdds.StatusMask attribute*), 592
requested_deadline_missed_status (*rti.connextdds.DataReader property*), 94

requested_deadline_missed_status (*rti.connextdds.DynamicData.DataReader property*), 201
requested_deadline_missed_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 397
requested_deadline_missed_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 455
requested_deadline_missed_status (*rti.connextdds.ServiceRequest.DataReader property*), 556
requested_deadline_missed_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
requested_deadline_missed_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
REQUESTED_INCOMPATIBLE_QOS (*rti.connextdds.StatusMask attribute*), 592
requested_incompatible_qos_status (*rti.connextdds.DataReader property*), 94
requested_incompatible_qos_status (*rti.connextdds.DynamicData.DataReader property*), 201
requested_incompatible_qos_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 397
requested_incompatible_qos_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 455
requested_incompatible_qos_status (*rti.connextdds.ServiceRequest.DataReader property*), 556
requested_incompatible_qos_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
requested_incompatible_qos_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
RequestedDeadlineMissedStatus (*class in rti.connextdds*), 518
RequestedIncompatibleQosStatus (*class in rti.connextdds*), 519
Requester (*class in rti.rpc*), 807
required_matched_endpoint_groups (*rti.connextdds.Availability property*), 57
RESERVED (*rti.connextdds.LocatorKind attribute*), 350
RESERVED (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
RESERVED_RANGE (*rti.connextdds.TransportClassId attribute*), 725
RESERVED_RANGE (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
reset() (*rti.connextdds.ActivityContextMask method*), 43
reset() (*rti.connextdds.compliance.RpcMask method*), 793
reset() (*rti.connextdds.compliance.XTypesMask method*), 796
reset() (*rti.connextdds.CompressionIdMask method*), 77
reset() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 160
reset() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
reset() (*rti.connextdds.InstanceState method*), 324
reset() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
reset() (*rti.connextdds.SampleFlag method*), 530
reset() (*rti.connextdds.SampleLostState method*), 535
reset() (*rti.connextdds.SampleRejectedState method*), 539
reset() (*rti.connextdds.SampleState method*), 541
reset() (*rti.connextdds.StatusMask method*), 594
reset() (*rti.connextdds.StreamKind method*), 596
reset() (*rti.connextdds.ThreadSettingsKindMask method*), 662
reset() (*rti.connextdds.TransportBuiltinMask method*), 724
reset() (*rti.connextdds.ViewState method*), 777
reset() (*rti.connextdds.WriteParams method*), 787
reset_default() (*rti.connextdds.QosProvider static method*), 505
reset_handler() (*rti.connextdds.GuardCondition method*), 297
reset_handler() (*rti.connextdds.QueryCondition method*), 509
reset_handler() (*rti.connextdds.ReadCondition method*), 510
reset_handler() (*rti.connextdds.StatusCondition method*), 591
reset_output_handler() (*rti.connextdds.Logger method*), 362
resize() (*rti.connextdds.ByteVector method*), 65
resize() (*rti.connextdds.CharSeq method*), 72
resize() (*rti.connextdds.CookieVector method*), 88
resize() (*rti.connextdds.EndpointGroupVector method*), 269
resize() (*rti.connextdds.Float32Seq method*), 287
resize() (*rti.connextdds.Float64Seq method*), 290
resize() (*rti.connextdds.Int8Seq method*), 337
resize() (*rti.connextdds.Int16Seq method*), 328
resize() (*rti.connextdds.Int32Seq method*), 331
resize() (*rti.connextdds.Int32Vector method*), 332
resize() (*rti.connextdds.Int64Seq method*), 334
resize() (*rti.connextdds.LocatorVector method*), 354
resize() (*rti.connextdds.TransportInfoVector method*), 732
resize() (*rti.connextdds.Uint8Seq method*), 767
resize() (*rti.connextdds.Uint16Seq method*), 759
resize() (*rti.connextdds.Uint32Seq method*), 761
resize() (*rti.connextdds.Uint64Seq method*), 764
resolve() (*rti.connextdds.AliasType method*), 44
resolve_type() (*rti.connextdds.AliasType static method*), 44
resource_limits (*rti.connextdds.DataReaderQos property*), 109
resource_limits (*rti.connextdds.DataWriterQos property*), 138
resource_limits (*rti.connextdds.DomainParticipantQos property*), 178
resource_limits (*rti.connextdds.TopicBuiltinTopicData property*), 704

resource_limits (*rti.connextdds.TopicQos property*), 714
resource_selection (*rti.connextdds.MonitoringMetricSelection property*), 371
ResourceLimits (*class in rti.connextdds*), 519
response_data (*rti.connextdds.AcknowledgmentInfo property*), 38
restore (*rti.connextdds.PersistentStorageSettings property*), 433
resume() (*in module rti.connextdds.heap_monitoring*), 803
resume() (*rti.connextdds.SuspendedPublication method*), 655
resume_endpoint_discovery() (*rti.connextdds.DomainParticipant method*), 170
retain() (*rti.connextdds.FlowController method*), 291
retain() (*rti.connextdds.IAnyDataReader method*), 302
retain() (*rti.connextdds.IAnyDataWriter method*), 305
retain() (*rti.connextdds.IEntity method*), 313
retain() (*rti.connextdds.TopicQuery method*), 715
return_loan() (*rti.connextdds.DataReader.LoanedSamples method*), 89
return_loan() (*rti.connextdds.DynamicData.LoanedSamples method*), 219
return_loan() (*rti.connextdds.LoanedDynamicData method*), 344
return_loan() (*rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples method*), 415
return_loan() (*rti.connextdds.PublicationBuiltinTopicData.LoanedSamples method*), 473
return_loan() (*rti.connextdds.ServiceRequest.LoanedSamples method*), 573
return_loan() (*rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples method*), 637
return_loan() (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples method*), 693
revision_version (*rti.connextdds.ProductVersion property*), 444
role_name (*rti.connextdds.EndpointGroup property*), 266
role_name (*rti.connextdds.EntityName property*), 271
ROUND_ROBIN (*rti.connextdds.FlowControllerSchedulingPolicy attribute*), 294
ROUND_ROBIN (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 293
ROUND_ROBIN (*rti.connextdds.ThreadSettingsCpuRotationKind attribute*), 658
ROUND_ROBIN (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind attribute*), 658
round_trip_time (*rti.connextdds.RtpsReliableReaderProtocol property*), 521
ROUTING (*rti.connextdds.ServiceKind attribute*), 545
ROUTING (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
RpcMask (*class in rti.connextdds.compliance*), 791
rti.asyncio
    module, 810
rti.connextdds
    module, 34
rti.connextdds.compliance
    module, 791
rti.connextdds.heap_monitoring
    module, 798
rti.logging
    module, 816
rti.logging.distlog
    module, 812
rti.logging.handler
    module, 815
rti.rpc
    module, 805
rti.types.builtin
    module, 804
rtps_app_id (*rti.connextdds.WireProtocol property*), 784
RTPS_AUTO_ID (*rti.connextdds.WireProtocol attribute*), 783
RTPS_AUTO_ID_FROM_IP (*rti.connextdds.WireProtocolAutoKind attribute*), 784
RTPS_AUTO_ID_FROM_IP (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind attribute*), 785
RTPS_AUTO_ID_FROM_MAC (*rti.connextdds.WireProtocolAutoKind attribute*), 784
RTPS_AUTO_ID_FROM_MAC (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind attribute*), 785
RTPS_AUTO_ID_FROM_UUID (*rti.connextdds.WireProtocolAutoKind attribute*), 784
RTPS_AUTO_ID_FROM_UUID (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind attribute*), 785
rtps_auto_id_kind (*rti.connextdds.WireProtocol property*), 784
rtps_host_id (*rti.connextdds.WireProtocol property*), 784
rtps_instance_id (*rti.connextdds.WireProtocol property*), 784
rtps_object_id (*rti.connextdds.DataReaderProtocol property*), 101
rtps_object_id (*rti.connextdds.DataWriterProtocol property*), 129
rtps_protocol_version (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
rtps_protocol_version (*rti.connextdds.PublicationBuiltinTopicData property*), 484
rtps_protocol_version (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
rtps_reliable_reader (*rti.connextdds.DataReaderProtocol property*), 101
rtps_reliable_writer (*rti.connextdds.DataWriterProtocol property*), 129

rtps_reserved_port_mask (*rti.connextdds.WireProtocol property*), 784
rtps_vendor_id (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
rtps_vendor_id (*rti.connextdds.PublicationBuiltinTopicData property*), 485
rtps_vendor_id (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
rtps_well_known_ports (*rti.connextdds.WireProtocol property*), 784
RtpsReliableReaderProtocol (*class in rti.connextdds*), 520
RtpsReliableWriterProtocol (*class in rti.connextdds*), 521
RtpsReservedPortKindMask (*class in rti.connextdds*), 523
RtpsWellKnownPorts (*class in rti.connextdds*), 527
run() (*in module rti.asyncio*), 812
run() (*rti.asyncio.Application method*), 811
run() (*rti.rpc.Service method*), 809

## S

sample (*rti.connextdds.Rank property*), 509
sample_count (*rti.connextdds.DataReaderCacheStatus property*), 97
sample_count (*rti.connextdds.DataWriterCacheStatus property*), 127
sample_count_peak (*rti.connextdds.DataReaderCacheStatus property*), 97
sample_count_peak (*rti.connextdds.DataWriterCacheStatus property*), 127
sample_identity (*rti.connextdds.AcknowledgmentInfo property*), 38
sample_identity (*rti.connextdds.FilterSampleInfo property*), 282
SAMPLE_LOST (*rti.connextdds.StatusMask attribute*), 592
sample_lost_status (*rti.connextdds.DataReader property*), 94
sample_lost_status (*rti.connextdds.DynamicData.DataReader property*), 201
sample_lost_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
sample_lost_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 455
sample_lost_status (*rti.connextdds.ServiceRequest.DataReader property*), 556
sample_lost_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
sample_lost_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
SAMPLE_REJECTED (*rti.connextdds.StatusMask attribute*), 592
sample_rejected_status (*rti.connextdds.DataReader property*), 94
sample_rejected_status (*rti.connextdds.DynamicData.DataReader property*), 201
sample_rejected_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
sample_rejected_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 455
sample_rejected_status (*rti.connextdds.ServiceRequest.DataReader property*), 556
sample_rejected_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
sample_rejected_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
SAMPLE_REMOVED (*rti.connextdds.StatusMask attribute*), 592
sample_state (*rti.connextdds.DataState property*), 117
sample_state (*rti.connextdds.DataStateEx property*), 119
SampleFlag (*class in rti.connextdds*), 528
SampleIdentity (*class in rti.connextdds*), 530
SampleInfo (*class in rti.connextdds*), 531
SampleLostState (*class in rti.connextdds*), 533
SampleLostStatus (*class in rti.connextdds*), 536
SampleRejectedState (*class in rti.connextdds*), 536
SampleRejectedStatus (*class in rti.connextdds*), 539
samples_per_app_ack (*rti.connextdds.RtpsReliableReaderProtocol property*), 521
samples_per_period (*rti.connextdds.TopicQueryDispatch property*), 716
samples_per_virtual_heartbeat (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
SampleState (*class in rti.connextdds*), 540
scheduling_policy (*rti.connextdds.FlowControllerProperty property*), 292
scope (*rti.connextdds.DestinationOrder property*), 149
SDP (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
SDP2 (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
sec (*rti.connextdds.Duration property*), 192
sec (*rti.connextdds.Time property*), 664
secure_volatile_reader (*rti.connextdds.DiscoveryConfig property*), 157
secure_volatile_writer (*rti.connextdds.DiscoveryConfig property*), 157
secure_volatile_writer_publish_mode (*rti.connextdds.DiscoveryConfig property*), 157
security (*rti.connextdds.LogCategory attribute*), 357
security (*rti.connextdds.LogCategory.LogCategory attribute*), 356
security_event (*rti.connextdds.LogFacility attribute*), 358

security_event_forwarding_level (*rti.connextdds.MonitoringLoggingForwardingSettings property*), 370
SEDP (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
select() (*rti.connextdds.DataReader method*), 94
select() (*rti.connextdds.DynamicData.DataReader method*), 201
select() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
select() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
select() (*rti.connextdds.ServiceRequest.DataReader method*), 556
select() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 619
select() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 676
select_all() (*rti.connextdds.TopicQuery static method*), 715
SELECT_DEFAULT_DISCRIMINATOR_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
selection (*rti.connextdds.TopicQueryData property*), 716
send_window_decrease_factor (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
send_window_increase_factor (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
send_window_size (*rti.connextdds.DataWriterProtocolStatus property*), 131
send_window_update_period (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
sent_ack_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 102
sent_ack_count (*rti.connextdds.DataReaderProtocolStatus property*), 103
sent_gap_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 131
sent_gap_count (*rti.connextdds.DataWriterProtocolStatus property*), 131
sent_heartbeat_bytes (*rti.connextdds.DataWriterProtocolStatus property*), 131
sent_heartbeat_count (*rti.connextdds.DataWriterProtocolStatus property*), 131
sent_nack_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 103
sent_nack_count (*rti.connextdds.DataReaderProtocolStatus property*), 103
sent_nack_fragment_bytes (*rti.connextdds.DataReaderProtocolStatus property*), 103
sent_nack_fragment_count (*rti.connextdds.DataReaderProtocolStatus property*), 103
SENTINEL_IN_EMPTY_UNION_BIT (*rti.connextdds.compliance.XTypesMask attribute*), 794
sequence_number (*rti.connextdds.SampleIdentity property*), 531
SEQUENCE_TYPE (*rti.connextdds.TypeKind attribute*), 751
SEQUENCE_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
SequenceNumber (*class in rti.connextdds*), 542
SequenceType (*class in rti.connextdds*), 543
serialize_key_with_dispose (*rti.connextdds.DataWriterProtocol property*), 129
serialized_type_object_dynamic_allocation_threshold (*rti.connextdds.DomainParticipantResourceLimits property*), 182
Service (*class in rti.connextdds*), 544
Service (*class in rti.rpc*), 808
service (*rti.connextdds.DataReaderQos property*), 109
service (*rti.connextdds.DataWriterQos property*), 138
service (*rti.connextdds.DomainParticipantQos property*), 178
service (*rti.connextdds.LogFacility attribute*), 358
service (*rti.connextdds.PublicationBuiltinTopicData property*), 485
service (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
service() (*in module rti.rpc*), 810
service_cleanup_delay (*rti.connextdds.DurabilityService property*), 190
service_forwarding_level (*rti.connextdds.MonitoringLoggingForwardingSettings property*), 370
service_id (*rti.connextdds.ServiceRequest property*), 583
service_id (*rti.connextdds.ServiceRequestAcceptedStatus property*), 583
SERVICE_REQUEST (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 158
SERVICE_REQUEST_ACCEPTED (*rti.connextdds.StatusMask attribute*), 592
service_request_accepted_status (*rti.connextdds.DataWriter property*), 125
service_request_accepted_status (*rti.connextdds.DynamicData.DataWriter property*), 210
service_request_accepted_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 407
service_request_accepted_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 465
service_request_accepted_status (*rti.connextdds.ServiceRequest.DataWriter property*), 565
service_request_accepted_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter*

*property*), 628
service_request_accepted_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 685
service_request_reader (*rti.connextdds.DiscoveryConfig property*), 157
service_request_reader (*rti.connextdds.DomainParticipant property*), 170
service_request_writer (*rti.connextdds.DiscoveryConfig property*), 157
service_request_writer_data_lifecycle (*rti.connextdds.DiscoveryConfig property*), 157
service_request_writer_publish_mode (*rti.connextdds.DiscoveryConfig property*), 157
ServiceKind (*class in rti.connextdds*), 544
ServiceKind.ServiceKind (*class in rti.connextdds*), 545
ServiceRequest (*class in rti.connextdds*), 547
ServiceRequestAcceptedStatus (*class in rti.connextdds*), 583
ServiceRequest.ContentFilter (*class in rti.connextdds*), 547
ServiceRequest.ContentFilteredTopic (*class in rti.connextdds*), 548
ServiceRequest.ContentFilteredTopicSeq (*class in rti.connextdds*), 549
ServiceRequest.DataReader (*class in rti.connextdds*), 551
ServiceRequest.DataReaderListener (*class in rti.connextdds*), 557
ServiceRequest.DataReader.Selector (*class in rti.connextdds*), 551
ServiceRequest.DataReaderSeq (*class in rti.connextdds*), 558
ServiceRequest.DataWriter (*class in rti.connextdds*), 560
ServiceRequest.DataWriterListener (*class in rti.connextdds*), 568
ServiceRequest.DataWriterSeq (*class in rti.connextdds*), 569
ServiceRequestId (*class in rti.connextdds*), 583
ServiceRequestId.ServiceRequestId (*class in rti.connextdds*), 583
ServiceRequest.ITopicDescription (*class in rti.connextdds*), 571
ServiceRequest.LoanedSample (*class in rti.connextdds*), 572
ServiceRequest.LoanedSamples (*class in rti.connextdds*), 572
ServiceRequest.NoOpDataReaderListener (*class in rti.connextdds*), 573
ServiceRequest.NoOpDataWriterListener (*class in rti.connextdds*), 574
ServiceRequest.NoOpTopicListener (*class in rti.connextdds*), 575
ServiceRequest.Sample (*class in rti.connextdds*), 575
ServiceRequestSeq (*class in rti.connextdds*), 585
ServiceRequest.SharedSamples (*class in rti.connextdds*), 576
ServiceRequestTimestampedSeq (*class in rti.connextdds*), 588
ServiceRequest.Topic (*class in rti.connextdds*), 577
ServiceRequest.TopicDescription (*class in rti.connextdds*), 577
ServiceRequest.TopicListener (*class in rti.connextdds*), 578
ServiceRequest.TopicSeq (*class in rti.connextdds*), 578
ServiceRequest.ValidLoanedSamples (*class in rti.connextdds*), 580
ServiceRequest.WriterContentFilter (*class in rti.connextdds*), 581
ServiceRequest.WriterContentFilterHelper (*class in rti.connextdds*), 582
set() (*rti.connextdds.ActivityContextMask method*), 43
set() (*rti.connextdds.compliance.RpcMask method*), 793
set() (*rti.connextdds.compliance.XTypesMask method*), 797
set() (*rti.connextdds.CompressionIdMask method*), 77
set() (*rti.connextdds.DataTag method*), 120
set() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 160
set() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 163
set() (*rti.connextdds.InstanceState method*), 324
set() (*rti.connextdds.Property method*), 445
set() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
set() (*rti.connextdds.SampleFlag method*), 530
set() (*rti.connextdds.SampleLostState method*), 535
set() (*rti.connextdds.SampleRejectedState method*), 539
set() (*rti.connextdds.SampleState method*), 542
set() (*rti.connextdds.StatusMask method*), 594
set() (*rti.connextdds.StreamKind method*), 597
set() (*rti.connextdds.ThreadSettingsKindMask method*), 662
set() (*rti.connextdds.TransportBuiltinMask method*), 724
set() (*rti.connextdds.ViewState method*), 778
set_activity_context() (*in module rti.connextdds*), 790
set_boolean() (*rti.connextdds.DynamicData method*), 243
set_cdr_buffer() (*rti.connextdds.DynamicData method*), 243
set_char() (*rti.connextdds.DynamicData method*), 243
set_char_values() (*rti.connextdds.DynamicData method*), 244
set_complex() (*rti.connextdds.DynamicData method*), 244
set_complex_values() (*rti.connextdds.DynamicData method*), 244
set_double() (*rti.connextdds.DynamicData method*), 244
set_double_values() (*rti.connextdds.DynamicData method*), 245
set_filter() (*rti.connextdds.ContentFilteredTopic method*), 82
set_filter() (*rti.connextdds.DynamicData.ContentFilteredTopic method*), 193
set_filter() (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 390
set_filter() (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 448

set_filter() (*rti.connextdds.ServiceRequest.ContentFilteredTopic method*), 549
set_filter() (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 611
set_filter() (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 668
set_float() (*rti.connextdds.DynamicData method*), 245
set_float32() (*rti.connextdds.DynamicData method*), 245
set_float32_values() (*rti.connextdds.DynamicData method*), 245
set_float64() (*rti.connextdds.DynamicData method*), 246
set_float64_values() (*rti.connextdds.DynamicData method*), 246
set_float128() (*rti.connextdds.DynamicData method*), 245
set_float_values() (*rti.connextdds.DynamicData method*), 246
set_handler() (*rti.connextdds.GuardCondition method*), 297
set_handler() (*rti.connextdds.QueryCondition method*), 509
set_handler() (*rti.connextdds.StatusCondition method*), 591
set_handler_no_args() (*rti.connextdds.GuardCondition method*), 297
set_handler_no_args() (*rti.connextdds.QueryCondition method*), 509
set_handler_no_args() (*rti.connextdds.ReadCondition method*), 510
set_int() (*rti.connextdds.DynamicData method*), 246
set_int8() (*rti.connextdds.DynamicData method*), 247
set_int8_values() (*rti.connextdds.DynamicData method*), 247
set_int16() (*rti.connextdds.DynamicData method*), 246
set_int16_values() (*rti.connextdds.DynamicData method*), 246
set_int32() (*rti.connextdds.DynamicData method*), 247
set_int32_values() (*rti.connextdds.DynamicData method*), 247
set_int64() (*rti.connextdds.DynamicData method*), 247
set_int64_values() (*rti.connextdds.DynamicData method*), 247
set_int_values() (*rti.connextdds.DynamicData method*), 248
set_listener() (*rti.connextdds.DataReader method*), 95
set_listener() (*rti.connextdds.DataWriter method*), 125
set_listener() (*rti.connextdds.DomainParticipant method*), 170
set_listener() (*rti.connextdds.DynamicData.DataReader method*), 201
set_listener() (*rti.connextdds.DynamicData.DataWriter method*), 210
set_listener() (*rti.connextdds.DynamicData.Topic method*), 224
set_listener() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
set_listener() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 407
set_listener() (*rti.connextdds.ParticipantBuiltinTopicData.Topic method*), 420
set_listener() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
set_listener() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 465
set_listener() (*rti.connextdds.PublicationBuiltinTopicData.Topic method*), 478
set_listener() (*rti.connextdds.Publisher method*), 495
set_listener() (*rti.connextdds.ServiceRequest.DataReader method*), 556
set_listener() (*rti.connextdds.ServiceRequest.DataWriter method*), 565
set_listener() (*rti.connextdds.ServiceRequest.Topic method*), 577
set_listener() (*rti.connextdds.Subscriber method*), 605
set_listener() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 619
set_listener() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 628
set_listener() (*rti.connextdds.SubscriptionBuiltinTopicData.Topic method*), 642
set_listener() (*rti.connextdds.Topic method*), 666
set_listener() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 676
set_listener() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 685
set_listener() (*rti.connextdds.TopicBuiltinTopicData.Topic method*), 698
set_long() (*rti.connextdds.DynamicData method*), 248
set_long_values() (*rti.connextdds.DynamicData method*), 248
set_longdouble() (*rti.connextdds.DynamicData method*), 248
set_longlong() (*rti.connextdds.DynamicData method*), 248
set_longlong_values() (*rti.connextdds.DynamicData method*), 249
set_name() (*rti.connextdds.Member method*), 364
set_octet() (*rti.connextdds.DynamicData method*), 249
set_octet_values() (*rti.connextdds.DynamicData method*), 249
set_parameters() (*rti.connextdds.Filter method*), 281
set_rpc_mask() (*in module rti.connextdds.compliance*), 797
set_short() (*rti.connextdds.DynamicData method*), 249
set_short_values() (*rti.connextdds.DynamicData method*), 249
set_string() (*rti.connextdds.DynamicData method*), 250

set_topic_datareader_qos() (*rti.connextdds.QosProvider method*), 505
set_topic_datawriter_qos() (*rti.connextdds.QosProvider method*), 505
set_topic_name_qos() (*rti.connextdds.QosProvider method*), 505
set_uint() (*rti.connextdds.DynamicData method*), 250
set_uint8() (*rti.connextdds.DynamicData method*), 251
set_uint8_values() (*rti.connextdds.DynamicData method*), 251
set_uint16() (*rti.connextdds.DynamicData method*), 250
set_uint16_values() (*rti.connextdds.DynamicData method*), 250
set_uint32() (*rti.connextdds.DynamicData method*), 250
set_uint32_values() (*rti.connextdds.DynamicData method*), 250
set_uint64() (*rti.connextdds.DynamicData method*), 251
set_uint64_values() (*rti.connextdds.DynamicData method*), 251
set_uint_values() (*rti.connextdds.DynamicData method*), 251
set_ulong() (*rti.connextdds.DynamicData method*), 252
set_ulong_values() (*rti.connextdds.DynamicData method*), 252
set_ulonglong() (*rti.connextdds.DynamicData method*), 252
set_ulonglong_values() (*rti.connextdds.DynamicData method*), 252
set_ushort() (*rti.connextdds.DynamicData method*), 252
set_ushort_values() (*rti.connextdds.DynamicData method*), 252
set_value() (*rti.connextdds.DynamicData method*), 253
set_values() (*rti.connextdds.DynamicData method*), 253
set_wchar() (*rti.connextdds.DynamicData method*), 253
set_wstring() (*rti.connextdds.DynamicData method*), 253
set_xtypes_mask() (*in module rti.connextdds.compliance*), 797
SEVERE (*rti.logging.distlog.LogLevel attribute*), 813
severe() (*rti.logging.distlog.Logger static method*), 814
shared (*rti.connextdds.Ownership attribute*), 386
SHARED (*rti.connextdds.OwnershipKind attribute*), 387
SHARED (*rti.connextdds.OwnershipKind.OwnershipKind attribute*), 386
SHMEM (*rti.connextdds.LocatorKind attribute*), 350
SHMEM (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
shmem (*rti.connextdds.TransportBuiltin attribute*), 722
SHMEM (*rti.connextdds.TransportBuiltinMask attribute*), 722
SHMEM (*rti.connextdds.TransportClassId attribute*), 725
SHMEM (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
SHMEM_510 (*rti.connextdds.LocatorKind attribute*), 350
SHMEM_510 (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
SHMEM_510 (*rti.connextdds.TransportClassId attribute*), 725
SHMEM_510 (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
shmem_ref_settings (*rti.connextdds.DataWriterTransferMode property*), 147
shmem_ref_transfer_mode_attached_segment_allocation (*rti.connextdds.DataReaderResourceLimits property*), 112
shmem_ref_transfer_mode_max_segments (*rti.connextdds.DomainParticipantResourceLimits property*), 182
ShortSeq (*in module rti.connextdds*), 590
ShortType (*in module rti.connextdds*), 590
shutdown_cleanup_period (*rti.connextdds.Database property*), 148
shutdown_timeout (*rti.connextdds.Database property*), 148
SILENT (*rti.connextdds.SyslogVerbosity attribute*), 655
SILENT (*rti.connextdds.Verbosity attribute*), 773
SILENT (*rti.connextdds.Verbosity.Verbosity attribute*), 774
SILENT (*rti.logging.distlog.LogLevel attribute*), 813
SimpleReplier (*class in rti.rpc*), 809
size (*rti.connextdds.ActivityContextMask property*), 43
size (*rti.connextdds.compliance.RpcMask property*), 793
size (*rti.connextdds.compliance.XTypesMask property*), 797
size (*rti.connextdds.CompressionIdMask property*), 77
size (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask property*), 161
size (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask property*), 164
size (*rti.connextdds.InstanceState property*), 325
size (*rti.connextdds.RtpsReservedPortKindMask property*), 526
size (*rti.connextdds.SampleFlag property*), 530
size (*rti.connextdds.SampleLostState property*), 536
size (*rti.connextdds.SampleRejectedState property*), 539
size (*rti.connextdds.SampleState property*), 542
size (*rti.connextdds.StatusMask property*), 594
size (*rti.connextdds.StreamKind property*), 597
size (*rti.connextdds.ThreadSettingsKindMask property*), 663
size (*rti.connextdds.TransportBuiltinMask property*), 724
size (*rti.connextdds.ViewState property*), 778
size() (*rti.connextdds.DataTag method*), 120
size() (*rti.connextdds.Property method*), 445
skip_deserialization (*rti.connextdds.DynamicDataTypeSerializationProperty property*), 264
sleep() (*in module rti.connextdds*), 791
snapshot_content_format (*rti.connextdds.heap_monitoring.HeapMonitoringParams property*), 798
snapshot_output_format (*rti.connextdds.heap_monitoring.HeapMonitoringParams property*), 798
SnapshotContentFormat (*class in rti.connextdds.heap_monitoring*), 798
SnapshotContentFormat.SnapshotContentFormat (*class in rti.connextdds.heap_monitoring*), 798
SnapshotOutputFormat (*class in rti.connextdds.heap_monitoring*), 801
SnapshotOutputFormat.SnapshotOutputFormat (*class in rti.connextdds.heap_monitoring*), 801
source_guid (*rti.connextdds.FilterSampleInfo property*), 282
source_guid (*rti.connextdds.SampleInfo property*), 532

source_guid (*rti.connextdds.WriteParams property*), 787
source_timestamp (*rti.connextdds.SampleInfo property*), 533
source_timestamp (*rti.connextdds.WriteParams property*), 787
source_timestamp_resolution (*rti.connextdds.Batch property*), 58
source_timestamp_tolerance (*rti.connextdds.DestinationOrder property*), 149
SPDP (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
SPDP2 (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask attribute*), 161
spin() (*in module rti.connextdds*), 791
spin_per_microsecond() (*in module rti.connextdds*), 791
SQL_FILTER_NAME (*rti.connextdds.Filter attribute*), 281
SQL_FILTER_NAME (*rti.connextdds.MultiChannel attribute*), 375
stack_size (*rti.connextdds.ThreadSettings property*), 658
STANDARD (*rti.connextdds.heap_monitoring.SnapshotOutputFormat attribute*), 801
STANDARD (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat attribute*), 801
state (*rti.connextdds.SampleInfo property*), 533
state() (*rti.connextdds.DataReader.Selector method*), 90
state() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
state() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 393
state() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
state() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
state() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 615
state() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 672
state_filter (*rti.connextdds.IReadCondition property*), 316
STATUS_ALL (*rti.connextdds.LogLevel attribute*), 359
STATUS_ALL (*rti.connextdds.LogLevel.LogLevel attribute*), 359
STATUS_ALL (*rti.connextdds.Verbosity attribute*), 773
STATUS_ALL (*rti.connextdds.Verbosity.Verbosity attribute*), 774
status_changes (*rti.connextdds.IEntity property*), 313
STATUS_LOCAL (*rti.connextdds.LogLevel attribute*), 359
STATUS_LOCAL (*rti.connextdds.LogLevel.LogLevel attribute*), 359
STATUS_LOCAL (*rti.connextdds.Verbosity attribute*), 773
STATUS_LOCAL (*rti.connextdds.Verbosity.Verbosity attribute*), 774
STATUS_REMOTE (*rti.connextdds.LogLevel attribute*), 359
STATUS_REMOTE (*rti.connextdds.LogLevel.LogLevel attribute*), 359
STATUS_REMOTE (*rti.connextdds.Verbosity attribute*), 773
STATUS_REMOTE (*rti.connextdds.Verbosity.Verbosity attribute*), 774
StatusCondition (*class in rti.connextdds*), 590
StatusMask (*class in rti.connextdds*), 591
STDIO (*rti.connextdds.ThreadSettingsKindMask attribute*), 660
storage_settings (*rti.connextdds.Durability property*), 186
stored_size (*rti.connextdds.DynamicDataInfo property*), 257
stream_kind (*rti.connextdds.DataStateEx property*), 119
StreamKind (*class in rti.connextdds*), 595
String (*class in rti.types.builtin*), 804
string_profile (*rti.connextdds.QosProviderParams property*), 507
STRING_TYPE (*rti.connextdds.TypeKind attribute*), 751
STRING_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
StringMap (*class in rti.connextdds*), 597
STRINGMATCH_FILTER_NAME (*rti.connextdds.Filter attribute*), 281
STRINGMATCH_FILTER_NAME (*rti.connextdds.MultiChannel attribute*), 375
StringPairSeq (*class in rti.connextdds*), 598
StringSeq (*class in rti.connextdds*), 600
StringType (*class in rti.connextdds*), 602
StructType (*class in rti.connextdds*), 602
STRUCTURE_FWD_DECL_TYPE (*rti.connextdds.TypeKind attribute*), 751
STRUCTURE_FWD_DECL_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
STRUCTURE_TYPE (*rti.connextdds.TypeKind attribute*), 751
STRUCTURE_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
subscribe() (*rti.asyncio.Application method*), 811
Subscriber (*class in rti.connextdds*), 604
subscriber (*rti.connextdds.DataReader property*), 95
subscriber (*rti.connextdds.DynamicData.DataReader property*), 201
subscriber (*rti.connextdds.IAnyDataReader property*), 302
subscriber (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
subscriber (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 456
subscriber (*rti.connextdds.ServiceRequest.DataReader property*), 556
subscriber (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
subscriber (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
subscriber_group_data_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 182
subscriber_key (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
subscriber_qos (*rti.connextdds.QosProvider property*), 505

subscriber_qos_from_profile() (*rti.connextdds.QosProvider method*), 505
SubscriberListener (*class in rti.connextdds*), 605
SubscriberQos (*class in rti.connextdds*), 606
SubscriberSeq (*class in rti.connextdds*), 608
subscription_handle (*rti.connextdds.AcknowledgmentInfo property*), 38
SUBSCRIPTION_MATCHED (*rti.connextdds.StatusMask attribute*), 592
subscription_matched_status (*rti.connextdds.DataReader property*), 95
subscription_matched_status (*rti.connextdds.DynamicData.DataReader property*), 201
subscription_matched_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
subscription_matched_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 456
subscription_matched_status (*rti.connextdds.ServiceRequest.DataReader property*), 557
subscription_matched_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 619
subscription_matched_status (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
subscription_name (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
subscription_reader (*rti.connextdds.DiscoveryConfig property*), 157
subscription_reader (*rti.connextdds.DomainParticipant property*), 170
subscription_reader_resource_limits (*rti.connextdds.DiscoveryConfig property*), 157
subscription_writer (*rti.connextdds.DiscoveryConfig property*), 158
subscription_writer_data_lifecycle (*rti.connextdds.DiscoveryConfig property*), 158
subscription_writer_publish_mode (*rti.connextdds.DiscoveryConfig property*), 158
SubscriptionBuiltinTopicData (*class in rti.connextdds*), 610
SubscriptionBuiltinTopicData.ContentFilter (*class in rti.connextdds*), 610
SubscriptionBuiltinTopicData.ContentFilteredTopic (*class in rti.connextdds*), 610
SubscriptionBuiltinTopicData.ContentFilteredTopicSeq (*class in rti.connextdds*), 611
SubscriptionBuiltinTopicData.DataReader (*class in rti.connextdds*), 614
SubscriptionBuiltinTopicData.DataReaderListener (*class in rti.connextdds*), 620
SubscriptionBuiltinTopicData.DataReader.Selector (*class in rti.connextdds*), 614
SubscriptionBuiltinTopicData.DataReaderSeq (*class in rti.connextdds*), 621
SubscriptionBuiltinTopicData.DataWriter (*class in rti.connextdds*), 623
SubscriptionBuiltinTopicData.DataWriterListener (*class in rti.connextdds*), 632
SubscriptionBuiltinTopicData.DataWriterSeq (*class in rti.connextdds*), 633
SubscriptionBuiltinTopicData.ITopicDescription (*class in rti.connextdds*), 635
SubscriptionBuiltinTopicData.LoanedSample (*class in rti.connextdds*), 636
SubscriptionBuiltinTopicData.LoanedSamples (*class in rti.connextdds*), 636
SubscriptionBuiltinTopicData.NoOpDataReaderListener (*class in rti.connextdds*), 637
SubscriptionBuiltinTopicData.NoOpDataWriterListener (*class in rti.connextdds*), 638
SubscriptionBuiltinTopicData.NoOpTopicListener (*class in rti.connextdds*), 639
SubscriptionBuiltinTopicData.Sample (*class in rti.connextdds*), 640
SubscriptionBuiltinTopicDataSeq (*class in rti.connextdds*), 649
SubscriptionBuiltinTopicData.SharedSamples (*class in rti.connextdds*), 640
SubscriptionBuiltinTopicDataTimestampedSeq (*class in rti.connextdds*), 651
SubscriptionBuiltinTopicData.Topic (*class in rti.connextdds*), 641
SubscriptionBuiltinTopicData.TopicDescription (*class in rti.connextdds*), 642
SubscriptionBuiltinTopicData.TopicListener (*class in rti.connextdds*), 642
SubscriptionBuiltinTopicData.TopicSeq (*class in rti.connextdds*), 643
SubscriptionBuiltinTopicData.ValidLoanedSamples (*class in rti.connextdds*), 645
SubscriptionBuiltinTopicData.WriterContentFilter (*class in rti.connextdds*), 645
SubscriptionBuiltinTopicData.WriterContentFilterHelper (*class in rti.connextdds*), 646
SubscriptionMatchedStatus (*class in rti.connextdds*), 654
SuspendedPublication (*class in rti.connextdds*), 654
synchronization_kind (*rti.connextdds.PersistentStorageSettings property*), 434

synchronous (*rti.connextdds.PublishMode attribute*), 491
SYNCHRONOUS (*rti.connextdds.PublishModeKind attribute*), 493
SYNCHRONOUS (*rti.connextdds.PublishModeKind.PublishModeKind attribute*), 492
SyslogVerbosity (*class in rti.connextdds*), 655
system_resource_limits (*rti.connextdds.DomainParticipantFactoryQos property*), 173
SystemResourceLimits (*class in rti.connextdds*), 656

## T

take() (*rti.connextdds.DataReader method*), 95
take() (*rti.connextdds.DataReader.Selector method*), 90
take() (*rti.connextdds.DynamicData.DataReader method*), 201
take() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
take() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
take() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 394
take() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
take() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
take() (*rti.connextdds.ServiceRequest.DataReader method*), 557
take() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
take() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 619
take() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 615
take() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 676
take() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 672
take_async() (*rti.connextdds.DataReader method*), 95
take_async() (*rti.connextdds.DynamicData.DataReader method*), 201
take_data() (*rti.connextdds.DataReader method*), 95
take_data() (*rti.connextdds.DataReader.Selector method*), 90
take_data() (*rti.connextdds.DynamicData.DataReader method*), 201
take_data() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
take_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
take_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 394
take_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
take_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
take_data() (*rti.connextdds.ServiceRequest.DataReader method*), 557
take_data() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
take_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 620
take_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 615
take_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 676
take_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 672
take_data_async() (*rti.connextdds.DataReader method*), 95
take_data_async() (*rti.connextdds.DynamicData.DataReader method*), 201
take_loaned() (*rti.connextdds.DataReader method*), 95
take_loaned() (*rti.connextdds.DataReader.Selector method*), 90
take_loaned() (*rti.connextdds.DynamicData.DataReader method*), 201
take_loaned() (*rti.connextdds.DynamicData.DataReader.Selector method*), 197
take_loaned() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
take_loaned() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 394
take_loaned() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
take_loaned() (*rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector method*), 451
take_loaned() (*rti.connextdds.ServiceRequest.DataReader method*), 557
take_loaned() (*rti.connextdds.ServiceRequest.DataReader.Selector method*), 552
take_loaned() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 620

take_loaned() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 615
take_loaned() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 676
take_loaned() (*rti.connextdds.TopicBuiltinTopicData.DataReader.Selector method*), 672
take_snapshot() (*in module rti.connextdds.heap_monitoring*), 803
TCPV4_LAN (*rti.connextdds.LocatorKind attribute*), 350
TCPV4_LAN (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
TCPV4_LAN (*rti.connextdds.TransportClassId attribute*), 725
TCPV4_LAN (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
TCPV4_WAN (*rti.connextdds.LocatorKind attribute*), 350
TCPV4_WAN (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
TCPV4_WAN (*rti.connextdds.TransportClassId attribute*), 725
TCPV4_WAN (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
telemetry_data (*rti.connextdds.Monitoring property*), 367
test() (*rti.connextdds.ActivityContextMask method*), 44
test() (*rti.connextdds.compliance.RpcMask method*), 793
test() (*rti.connextdds.compliance.XTypesMask method*), 797
test() (*rti.connextdds.CompressionIdMask method*), 77
test() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 161
test() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 164
test() (*rti.connextdds.InstanceState method*), 325
test() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
test() (*rti.connextdds.SampleFlag method*), 530
test() (*rti.connextdds.SampleLostState method*), 536
test() (*rti.connextdds.SampleRejectedState method*), 539
test() (*rti.connextdds.SampleState method*), 542
test() (*rti.connextdds.StatusMask method*), 594
test() (*rti.connextdds.StreamKind method*), 597
test() (*rti.connextdds.ThreadSettingsKindMask method*), 663
test() (*rti.connextdds.TransportBuiltinMask method*), 724
test() (*rti.connextdds.ViewState method*), 778
test_all() (*rti.connextdds.ActivityContextMask method*), 44
test_all() (*rti.connextdds.compliance.RpcMask method*), 794
test_all() (*rti.connextdds.compliance.XTypesMask method*), 797
test_all() (*rti.connextdds.CompressionIdMask method*), 77
test_all() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 161
test_all() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 164
test_all() (*rti.connextdds.InstanceState method*), 325
test_all() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
test_all() (*rti.connextdds.SampleFlag method*), 530
test_all() (*rti.connextdds.SampleLostState method*), 536
test_all() (*rti.connextdds.SampleRejectedState method*), 539
test_all() (*rti.connextdds.SampleState method*), 542
test_all() (*rti.connextdds.StatusMask method*), 594
test_all() (*rti.connextdds.StreamKind method*), 597
test_all() (*rti.connextdds.ThreadSettingsKindMask method*), 663
test_all() (*rti.connextdds.TransportBuiltinMask method*), 725
test_all() (*rti.connextdds.ViewState method*), 778
test_any() (*rti.connextdds.ActivityContextMask method*), 44
test_any() (*rti.connextdds.compliance.RpcMask method*), 794
test_any() (*rti.connextdds.compliance.XTypesMask method*), 797
test_any() (*rti.connextdds.CompressionIdMask method*), 77
test_any() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 161
test_any() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 164
test_any() (*rti.connextdds.InstanceState method*), 325
test_any() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
test_any() (*rti.connextdds.SampleFlag method*), 530
test_any() (*rti.connextdds.SampleLostState method*), 536
test_any() (*rti.connextdds.SampleRejectedState method*), 539
test_any() (*rti.connextdds.SampleState method*), 542
test_any() (*rti.connextdds.StatusMask method*), 594
test_any() (*rti.connextdds.StreamKind method*), 597
test_any() (*rti.connextdds.ThreadSettingsKindMask method*), 663
test_any() (*rti.connextdds.TransportBuiltinMask method*), 725
test_any() (*rti.connextdds.ViewState method*), 778
test_none() (*rti.connextdds.ActivityContextMask method*), 44
test_none() (*rti.connextdds.compliance.RpcMask method*), 794
test_none() (*rti.connextdds.compliance.XTypesMask method*), 797
test_none() (*rti.connextdds.CompressionIdMask method*), 77
test_none() (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask method*), 161
test_none() (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask method*), 164
test_none() (*rti.connextdds.InstanceState method*), 325
test_none() (*rti.connextdds.RtpsReservedPortKindMask method*), 526
test_none() (*rti.connextdds.SampleFlag method*), 530
test_none() (*rti.connextdds.SampleLostState method*), 536
test_none() (*rti.connextdds.SampleRejectedState method*), 539
test_none() (*rti.connextdds.SampleState method*), 542

test_none() (*rti.connextdds.StatusMask method*), 594
test_none() (*rti.connextdds.StreamKind method*), 597
test_none() (*rti.connextdds.ThreadSettingsKindMask method*), 663
test_none() (*rti.connextdds.TransportBuiltinMask method*), 725
test_none() (*rti.connextdds.ViewState method*), 778
text (*rti.connextdds.LogMessage property*), 361
thread (*rti.connextdds.AsynchronousPublisher property*), 56
thread (*rti.connextdds.Database property*), 148
thread (*rti.connextdds.Event property*), 277
thread (*rti.connextdds.MonitoringEventDistributionSettings property*), 369
thread (*rti.connextdds.MonitoringLoggingDistributionSettings property*), 370
thread (*rti.connextdds.MonitoringPeriodicDistributionSettings property*), 374
thread (*rti.connextdds.ReceiverPool property*), 512
thread_safe_write (*rti.connextdds.Batch property*), 58
thread_settings (*rti.logging.distlog.LoggerOptions property*), 815
ThreadContext (*class in rti.connextdds*), 657
ThreadSettings (*class in rti.connextdds*), 657
ThreadSettingsCpuRotationKind (*class in rti.connextdds*), 658
ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind (*class in rti.connextdds*), 658
ThreadSettingsKindMask (*class in rti.connextdds*), 660
Time (*class in rti.connextdds*), 663
time_based_filter (*rti.connextdds.DataReaderQos property*), 109
time_based_filter (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
time_based_filter_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 97
TimeBasedFilter (*class in rti.connextdds*), 665
TimeoutError, 665
timestamp (*rti.connextdds.LogMessage property*), 361
timestamp (*rti.logging.distlog.MessageParams property*), 815
TIMESTAMPED (*rti.connextdds.PrintFormat attribute*), 440
TIMESTAMPED (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
TLSV4_LAN (*rti.connextdds.LocatorKind attribute*), 350
TLSV4_LAN (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
TLSV4_LAN (*rti.connextdds.TransportClassId attribute*), 725
TLSV4_LAN (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
TLSV4_WAN (*rti.connextdds.LocatorKind attribute*), 350
TLSV4_WAN (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
TLSV4_WAN (*rti.connextdds.TransportClassId attribute*), 725
TLSV4_WAN (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
to_cdr_buffer() (*rti.connextdds.DynamicData method*), 253
to_cdr_buffer() (*rti.connextdds.DynamicData.TopicTypeSupport static method*), 227
to_json() (*rti.connextdds.DynamicData method*), 253
to_microseconds() (*rti.connextdds.Duration method*), 192
to_microseconds() (*rti.connextdds.Time method*), 665
to_milliseconds() (*rti.connextdds.Duration method*), 192
to_milliseconds() (*rti.connextdds.Time method*), 665
to_seconds() (*rti.connextdds.Duration method*), 192
to_seconds() (*rti.connextdds.Time method*), 665
to_string() (*rti.connextdds.DataReaderQos method*), 109
to_string() (*rti.connextdds.DataWriterQos method*), 138
to_string() (*rti.connextdds.DomainParticipantFactoryQos method*), 173
to_string() (*rti.connextdds.DomainParticipantQos method*), 178
to_string() (*rti.connextdds.DynamicData method*), 254
to_string() (*rti.connextdds.DynamicType method*), 265
to_string() (*rti.connextdds.PublisherQos method*), 497
to_string() (*rti.connextdds.SubscriberQos method*), 607
to_string() (*rti.connextdds.TopicQos method*), 714
to_timedelta() (*rti.connextdds.Duration method*), 192
token_bucket (*rti.connextdds.FlowControllerProperty property*), 292
tokens_added_per_period (*rti.connextdds.FlowControllerTokenBucketProperty property*), 296
tokens_leaked_per_period (*rti.connextdds.FlowControllerTokenBucketProperty property*), 296
tolerance_source_timestamp_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 97
Topic (*class in rti.connextdds*), 665
TOPIC (*rti.connextdds.ActivityContextMask attribute*), 41
topic (*rti.connextdds.ContentFilteredTopic property*), 82
topic (*rti.connextdds.DataWriter property*), 125
TOPIC (*rti.connextdds.DestinationOrderScopeKind attribute*), 153
TOPIC (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 152
topic (*rti.connextdds.DynamicData.ContentFilteredTopic property*), 193
topic (*rti.connextdds.DynamicData.DataWriter property*), 210
TOPIC (*rti.connextdds.heap_monitoring.SnapshotContentFormat attribute*), 800
TOPIC (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat attribute*), 799
topic (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic property*), 390
topic (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 407
TOPIC (*rti.connextdds.PresentationAccessScopeKind attribute*), 437
TOPIC (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*),

437
topic (*rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic property*), 448
topic (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 465
topic (*rti.connextdds.ServiceRequest.ContentFilteredTopic property*), 549
topic (*rti.connextdds.ServiceRequest.DataWriter property*), 565
topic (*rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic property*), 611
topic (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 629
topic (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic property*), 668
topic (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 685
topic_access_scope() (*rti.connextdds.Presentation static method*), 436
topic_data (*rti.connextdds.PublicationBuiltinTopicData property*), 485
topic_data (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
topic_data (*rti.connextdds.TopicBuiltinTopicData property*), 704
topic_data (*rti.connextdds.TopicQos property*), 714
topic_data_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 182
topic_description (*rti.connextdds.DataReader property*), 95
topic_description (*rti.connextdds.DynamicData.DataReader property*), 201
topic_description (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
topic_description (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 456
topic_description (*rti.connextdds.ServiceRequest.DataReader property*), 557
topic_description (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 620
topic_description (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
topic_expression (*rti.connextdds.MulticastMapping property*), 376
topic_name (*rti.connextdds.DataReader property*), 95
topic_name (*rti.connextdds.DataWriter property*), 125
topic_name (*rti.connextdds.DynamicData.DataReader property*), 202
topic_name (*rti.connextdds.DynamicData.DataWriter property*), 210
topic_name (*rti.connextdds.IAnyDataReader property*), 302
topic_name (*rti.connextdds.IAnyDataWriter property*), 305
topic_name (*rti.connextdds.ParticipantBuiltinTopicData attribute*), 426
topic_name (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
topic_name (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 407
topic_name (*rti.connextdds.PublicationBuiltinTopicData property*), 485
topic_name (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 456
topic_name (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 465
topic_name (*rti.connextdds.ServiceRequest attribute*), 583
topic_name (*rti.connextdds.ServiceRequest.DataReader property*), 557
topic_name (*rti.connextdds.ServiceRequest.DataWriter property*), 565
topic_name (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
topic_name (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 620
topic_name (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 629
topic_name (*rti.connextdds.TopicBuiltinTopicData attribute*), 704
topic_name (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
topic_name (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 685
topic_name (*rti.connextdds.TopicQueryData property*), 716
topic_qos (*rti.connextdds.QosProvider property*), 505
topic_qos_from_profile() (*rti.connextdds.QosProvider method*), 506
topic_queries (*rti.connextdds.DataStateEx attribute*), 119
TOPIC_QUERY (*rti.connextdds.ServiceRequestId attribute*), 585
TOPIC_QUERY (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
TOPIC_QUERY (*rti.connextdds.StreamKind attribute*), 595
topic_query_dispatch (*rti.connextdds.DataWriterQos property*), 138
topic_query_guid (*rti.connextdds.SampleInfo property*), 533
topic_query_publication_thread (*rti.connextdds.AsynchronousPublisher property*), 56
TopicBuiltinTopicData (*class in rti.connextdds*), 667
TopicBuiltinTopicData.ContentFilter (*class in rti.connextdds*), 667
TopicBuiltinTopicData.ContentFilteredTopic

(*class in rti.connextdds*), 667
TopicBuiltinTopicData.ContentFiltered-TopicSeq (*class in rti.connextdds*), 668
TopicBuiltinTopicData.DataReader (*class in rti.connextdds*), 671
TopicBuiltinTopicData.DataReaderListener (*class in rti.connextdds*), 677
TopicBuiltinTopicData.DataReader.Selector (*class in rti.connextdds*), 671
TopicBuiltinTopicData.DataReaderSeq (*class in rti.connextdds*), 678
TopicBuiltinTopicData.DataWriter (*class in rti.connextdds*), 680
TopicBuiltinTopicData.DataWriterListener (*class in rti.connextdds*), 688
TopicBuiltinTopicData.DataWriterSeq (*class in rti.connextdds*), 689
TopicBuiltinTopicData.ITopicDescription (*class in rti.connextdds*), 692
TopicBuiltinTopicData.LoanedSample (*class in rti.connextdds*), 692
TopicBuiltinTopicData.LoanedSamples (*class in rti.connextdds*), 692
TopicBuiltinTopicData.NoOpDataReaderListener (*class in rti.connextdds*), 693
TopicBuiltinTopicData.NoOpDataWriterListener (*class in rti.connextdds*), 694
TopicBuiltinTopicData.NoOpTopicListener (*class in rti.connextdds*), 696
TopicBuiltinTopicData.Sample (*class in rti.connextdds*), 696
TopicBuiltinTopicDataSeq (*class in rti.connextdds*), 704
TopicBuiltinTopicData.SharedSamples (*class in rti.connextdds*), 696
TopicBuiltinTopicDataTimestampedSeq (*class in rti.connextdds*), 706
TopicBuiltinTopicData.Topic (*class in rti.connextdds*), 697
TopicBuiltinTopicData.TopicDescription (*class in rti.connextdds*), 698
TopicBuiltinTopicData.TopicListener (*class in rti.connextdds*), 698
TopicBuiltinTopicData.TopicSeq (*class in rti.connextdds*), 699
TopicBuiltinTopicData.ValidLoanedSamples (*class in rti.connextdds*), 701
TopicBuiltinTopicData.WriterContentFilter (*class in rti.connextdds*), 701
TopicBuiltinTopicData.WriterContentFilter-Helper (*class in rti.connextdds*), 702
TopicData (*class in rti.connextdds*), 709
TopicDescription (*class in rti.connextdds*), 709
TopicListener (*class in rti.connextdds*), 710
TopicQos (*class in rti.connextdds*), 710
TopicQuery (*class in rti.connextdds*), 714
TopicQueryData (*class in rti.connextdds*), 715
TopicQueryDispatch (*class in rti.connextdds*), 716
TopicQuerySelection (*class in rti.connextdds*), 716
TopicQuerySelectionKind (*class in rti.connextdds*), 717
TopicQuerySelectionKind.TopicQuerySelectionKind (*class in rti.connextdds*), 717
TopicSeq (*class in rti.connextdds*), 719
total (*rti.connextdds.EventCount32 property*), 278
total (*rti.connextdds.EventCount64 property*), 278
total_count (*rti.connextdds.InconsistentTopicStatus property*), 320
total_count (*rti.connextdds.LivelinessLostStatus property*), 343
total_count (*rti.connextdds.OfferedDeadlineMissedStatus property*), 385
total_count (*rti.connextdds.OfferedIncompatibleQosStatus property*), 385
total_count (*rti.connextdds.PublicationMatchedStatus property*), 490
total_count (*rti.connextdds.RequestedDeadlineMissedStatus property*), 518
total_count (*rti.connextdds.SampleLostStatus property*), 536
total_count (*rti.connextdds.SampleRejectedStatus property*), 539
total_count (*rti.connextdds.ServiceRequestAcceptedStatus property*), 583
total_count (*rti.connextdds.SubscriptionMatchedStatus property*), 654
total_count() (*rti.connextdds.RequestedIncompatibleQosStatus method*), 519
total_count_change (*rti.connextdds.InconsistentTopicStatus property*), 320
total_count_change (*rti.connextdds.LivelinessLostStatus property*), 343
total_count_change (*rti.connextdds.OfferedDeadlineMissedStatus property*), 385
total_count_change (*rti.connextdds.OfferedIncompatibleQosStatus property*), 385
total_count_change (*rti.connextdds.PublicationMatchedStatus property*), 491
total_count_change (*rti.connextdds.RequestedDeadlineMissedStatus property*), 519
total_count_change (*rti.connextdds.RequestedIncompatibleQosStatus property*), 519
total_count_change (*rti.connextdds.SampleLostStatus property*), 536
total_count_change (*rti.connextdds.SampleRejectedStatus property*), 540
total_count_change (*rti.connextdds.SubscriptionMatchedStatus property*), 654

total_element_count (*rti.connextdds.ArrayType property*), 55
total_samples_dropped_by_instance_replacement (*rti.connextdds.DataReaderCacheStatus property*), 97
TRACE (*rti.logging.distlog.LogLevel attribute*), 813
trace() (*rti.logging.distlog.Logger static method*), 814
trace_file_name (*rti.connextdds.PersistentStorageSettings property*), 434
transient (*rti.connextdds.Durability attribute*), 186
TRANSIENT (*rti.connextdds.DurabilityKind attribute*), 188
TRANSIENT (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 187
transient_local (*rti.connextdds.Durability attribute*), 186
TRANSIENT_LOCAL (*rti.connextdds.DurabilityKind attribute*), 188
TRANSIENT_LOCAL (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 187
transport_builtin (*rti.connextdds.DomainParticipantQos property*), 178
transport_info (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
transport_info_list_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 182
transport_multicast (*rti.connextdds.DataReaderQos property*), 109
transport_multicast_mapping (*rti.connextdds.DomainParticipantQos property*), 178
transport_priority (*rti.connextdds.DataReaderQos property*), 109
transport_priority (*rti.connextdds.DataWriterQos property*), 138
transport_priority (*rti.connextdds.TopicBuiltinTopicData property*), 704
transport_priority (*rti.connextdds.TopicQos property*), 714
transport_selection (*rti.connextdds.DataReaderQos property*), 109
transport_selection (*rti.connextdds.DataWriterQos property*), 138
transport_unicast (*rti.connextdds.DataReaderQos property*), 109
transport_unicast (*rti.connextdds.DataWriterQos property*), 139
TransportBuiltin (*class in rti.connextdds*), 721
TransportBuiltinMask (*class in rti.connextdds*), 722
TransportClassId (*class in rti.connextdds*), 725
TransportClassId.TransportClassId (*class in rti.connextdds*), 725
TransportInfo (*class in rti.connextdds*), 728
TransportInfoSeq (*class in rti.connextdds*), 729
TransportInfoVector (*class in rti.connextdds*), 731
TransportMulticast (*class in rti.connextdds*), 732
TransportMulticastKind (*class in rti.connextdds*), 732
TransportMulticastKind.TransportMulticastKind (*class in rti.connextdds*), 732
TransportMulticastMapping (*class in rti.connextdds*), 734
TransportMulticastMappingFunction (*class in rti.connextdds*), 735
TransportMulticastSeq (*class in rti.connextdds*), 735
TransportMulticastSettings (*class in rti.connextdds*), 738
TransportMulticastSettingsSeq (*class in rti.connextdds*), 738
TransportPriority (*class in rti.connextdds*), 741
transports (*rti.connextdds.TransportMulticastSettings property*), 738
transports (*rti.connextdds.TransportUnicastSettings property*), 743
TransportSelection (*class in rti.connextdds*), 741
TransportUnicast (*class in rti.connextdds*), 742
TransportUnicastSettings (*class in rti.connextdds*), 742
TransportUnicastSettingsSeq (*class in rti.connextdds*), 743
trigger_flow() (*rti.connextdds.FlowController method*), 291
trigger_value (*rti.connextdds.GuardCondition property*), 297
trigger_value (*rti.connextdds.ICondition property*), 310
trigger_value (*rti.connextdds.StatusCondition property*), 591
TriggeredConditions (*class in rti.connextdds*), 746
TriggeredConditionsIterator (*class in rti.connextdds*), 746
trim_to_size (*rti.connextdds.DynamicDataTypeSerializationProperty property*), 264
TRUNCATE (*rti.connextdds.PersistentJournalKind attribute*), 432
try_get() (*rti.connextdds.DataTag method*), 120
try_get() (*rti.connextdds.Property method*), 445
TYPE (*rti.connextdds.ActivityContextMask attribute*), 41
type (*rti.connextdds.DynamicData property*), 254
type (*rti.connextdds.Member property*), 364
type (*rti.connextdds.PublicationBuiltinTopicData property*), 485
type (*rti.connextdds.SubscriptionBuiltinTopicData property*), 648
type (*rti.connextdds.Topic property*), 667
type (*rti.connextdds.UnionMember property*), 768
type() (*rti.connextdds.QosProvider method*), 506
type_code_max_serialized_length (*rti.connextdds.DomainParticipantResourceLimits property*), 183
type_consistency (*rti.connextdds.DataReaderQos property*), 109
type_consistency_enforcement (*rti.connextdds.DataReaderQos property*), 110
type_kind (*rti.connextdds.DynamicData property*), 254
type_libraries (*rti.connextdds.QosProvider property*), 506
TYPE_LOOKUP_SERVICE (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask attribute*), 158
type_name (*rti.connextdds.DataReader property*), 95
type_name (*rti.connextdds.DataWriter property*), 125

type_name (*rti.connextdds.DynamicData.DataReader property*), 202
type_name (*rti.connextdds.DynamicData.DataWriter property*), 210
type_name (*rti.connextdds.DynamicData.ITopicDescription property*), 217
type_name (*rti.connextdds.IAnyDataReader property*), 302
type_name (*rti.connextdds.IAnyDataWriter property*), 305
type_name (*rti.connextdds.IAnyTopic property*), 308
type_name (*rti.connextdds.ITopicDescription property*), 317
type_name (*rti.connextdds.ParticipantBuiltinTopicData.DataReader property*), 398
type_name (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter property*), 407
type_name (*rti.connextdds.ParticipantBuiltinTopicData.ITopicDescription property*), 414
type_name (*rti.connextdds.PublicationBuiltinTopicData property*), 485
type_name (*rti.connextdds.PublicationBuiltinTopicData.DataReader property*), 456
type_name (*rti.connextdds.PublicationBuiltinTopicData.DataWriter property*), 465
type_name (*rti.connextdds.PublicationBuiltinTopicData.ITopicDescription property*), 472
type_name (*rti.connextdds.ServiceRequest.DataReader property*), 557
type_name (*rti.connextdds.ServiceRequest.DataWriter property*), 565
type_name (*rti.connextdds.ServiceRequest.ITopicDescription property*), 572
type_name (*rti.connextdds.SubscriptionBuiltinTopicData property*), 649
type_name (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader property*), 620
type_name (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property*), 629
type_name (*rti.connextdds.SubscriptionBuiltinTopicData.ITopicDescription property*), 636
type_name (*rti.connextdds.TopicBuiltinTopicData property*), 704
type_name (*rti.connextdds.TopicBuiltinTopicData.DataReader property*), 676
type_name (*rti.connextdds.TopicBuiltinTopicData.DataWriter property*), 685
type_name (*rti.connextdds.TopicBuiltinTopicData.ITopicDescription property*), 692
type_object_max_deserialized_length (*rti.connextdds.DomainParticipantResourceLimits property*), 183
type_object_max_serialized_length (*rti.connextdds.DomainParticipantResourceLimits property*), 183
type_support (*rti.connextdds.DataReaderQos property*), 110
type_support (*rti.connextdds.DataWriterQos property*), 139
type_support (*rti.connextdds.Topic property*), 667
TypeConsistencyEnforcement (*class in rti.connextdds*), 746
TypeConsistencyEnforcementKind (*class in rti.connextdds*), 747
TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind (*class in rti.connextdds*), 748
TypeKind (*class in rti.connextdds*), 750
TypeKind.TypeKind (*class in rti.connextdds*), 751
TypeSupport (*class in rti.connextdds*), 756

## U

UDPv4 (*rti.connextdds.LocatorKind attribute*), 350
UDPv4 (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
udpv4 (*rti.connextdds.TransportBuiltin attribute*), 722
UDPv4 (*rti.connextdds.TransportBuiltinMask attribute*), 722
UDPv4 (*rti.connextdds.TransportClassId attribute*), 727
UDPv4 (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
udpv4_wan (*rti.connextdds.TransportBuiltin attribute*), 722
UDPv4_WAN (*rti.connextdds.TransportBuiltinMask attribute*), 722
UDPv4_WAN (*rti.connextdds.TransportClassId attribute*), 727
UDPv4_WAN (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
UDPv6 (*rti.connextdds.LocatorKind attribute*), 350
UDPv6 (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
udpv6 (*rti.connextdds.TransportBuiltin attribute*), 722
UDPv6 (*rti.connextdds.TransportBuiltinMask attribute*), 722
UDPv6 (*rti.connextdds.TransportClassId attribute*), 727
UDPv6 (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
UDPv6_510 (*rti.connextdds.LocatorKind attribute*), 350
UDPv6_510 (*rti.connextdds.LocatorKind.LocatorKind attribute*), 349
UDPv6_510 (*rti.connextdds.TransportClassId attribute*), 727
UDPv6_510 (*rti.connextdds.TransportClassId.TransportClassId attribute*), 726
Uint8Seq (*class in rti.connextdds*), 764
Uint8Type (*class in rti.connextdds*), 767
Uint16Seq (*class in rti.connextdds*), 756
Uint16Type (*class in rti.connextdds*), 759
Uint32Seq (*class in rti.connextdds*), 759
Uint32Type (*class in rti.connextdds*), 761
Uint64Seq (*class in rti.connextdds*), 762
Uint64Type (*class in rti.connextdds*), 764
UINT_8_TYPE (*rti.connextdds.TypeKind attribute*), 755
UINT_8_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
UINT_16_TYPE (*rti.connextdds.TypeKind attribute*), 755
UINT_16_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753

UINT_32_TYPE (*rti.connextdds.TypeKind attribute*), 755
UINT_32_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
UINT_64_TYPE (*rti.connextdds.TypeKind attribute*), 755
UINT_64_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
ULongLongSeq (*in module rti.connextdds*), 756
ULongLongType (*in module rti.connextdds*), 756
ULongSeq (*in module rti.connextdds*), 756
ULongType (*in module rti.connextdds*), 756
unacknowledged_sample_count (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 516
unacknowledged_sample_count_peak (*rti.connextdds.ReliableWriterCacheChangedStatus property*), 516
UNBOUNDED (*rti.connextdds.UnidimensionalCollectionType attribute*), 767
uncommitted_sample_count (*rti.connextdds.DataReaderProtocolStatus property*), 103
underlying (*rti.connextdds.AcknowledgmentKind property*), 41
underlying (*rti.connextdds.CdrPaddingKind property*), 67
underlying (*rti.connextdds.DataReaderInstanceRemovalKind property*), 100
underlying (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind property*), 144
underlying (*rti.connextdds.DestinationOrderKind property*), 151
underlying (*rti.connextdds.DestinationOrderScopeKind property*), 154
underlying (*rti.connextdds.DurabilityKind property*), 189
underlying (*rti.connextdds.DynamicDataEncapsulationKind property*), 256
underlying (*rti.connextdds.ExtensibilityKind property*), 281
underlying (*rti.connextdds.FlowControllerSchedulingPolicy property*), 295
underlying (*rti.connextdds.heap_monitoring.SnapshotContentFormat property*), 801
underlying (*rti.connextdds.heap_monitoring.SnapshotOutputFormat property*), 803
underlying (*rti.connextdds.HistoryKind property*), 301
underlying (*rti.connextdds.IgnoredEntityReplacementKind property*), 319
underlying (*rti.connextdds.LivelinessKind property*), 343
underlying (*rti.connextdds.LocatorKind property*), 351
underlying (*rti.connextdds.LogCategory property*), 357
underlying (*rti.connextdds.LogLevel property*), 360
underlying (*rti.connextdds.OwnershipKind property*), 388
underlying (*rti.connextdds.PresentationAccessScopeKind property*), 438
underlying (*rti.connextdds.PrintFormat property*), 441
underlying (*rti.connextdds.PrintFormatKind property*), 443
underlying (*rti.connextdds.PublishModeKind property*), 493
underlying (*rti.connextdds.ReliabilityKind property*), 515
underlying (*rti.connextdds.RemoteParticipantPurgeKind property*), 518
underlying (*rti.connextdds.ServiceKind property*), 547
underlying (*rti.connextdds.ServiceRequestId property*), 585
underlying (*rti.connextdds.ThreadSettingsCpuRotationKind property*), 660
underlying (*rti.connextdds.TopicQuerySelectionKind property*), 719
underlying (*rti.connextdds.TransportClassId property*), 728
underlying (*rti.connextdds.TransportMulticastKind property*), 734
underlying (*rti.connextdds.TypeConsistencyEnforcementKind property*), 750
underlying (*rti.connextdds.TypeKind property*), 756
underlying (*rti.connextdds.Verbosity property*), 775
underlying (*rti.connextdds.WireProtocolAutoKind property*), 787
UNICAST (*rti.connextdds.TransportMulticastKind attribute*), 733
UNICAST (*rti.connextdds.TransportMulticastKind.TransportMulticastKind attribute*), 733
unicast_locators (*rti.connextdds.PublicationBuiltinTopicData property*), 485
unicast_locators (*rti.connextdds.SubscriptionBuiltinTopicData property*), 649
UnidimensionalCollectionType (*class in rti.connextdds*), 767
UNION_FWD_DECL_TYPE (*rti.connextdds.TypeKind attribute*), 755
UNION_FWD_DECL_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
UNION_TYPE (*rti.connextdds.TypeKind attribute*), 755
UNION_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
UnionMember (*class in rti.connextdds*), 767
UnionMemberSeq (*class in rti.connextdds*), 768
UnionType (*class in rti.connextdds*), 771
unit (*rti.connextdds.AliasType property*), 44
unit (*rti.connextdds.Member property*), 364
unit (*rti.connextdds.UnionMember property*), 768
UNKNOWN (*rti.connextdds.CoherentSetInfo attribute*), 73
unknown (*rti.connextdds.Guid attribute*), 298
unknown (*rti.connextdds.ProductVersion attribute*), 444
unknown (*rti.connextdds.SampleIdentity attribute*), 531
unknown (*rti.connextdds.SequenceNumber attribute*), 543
UNKNOWN (*rti.connextdds.ServiceRequestId attribute*), 585
UNKNOWN (*rti.connextdds.ServiceRequestId.ServiceRequestId attribute*), 584
unknown (*rti.connextdds.VendorId attribute*), 773
unload_profiles() (*rti.connextdds.QosProvider method*), 506
unlock() (*rti.connextdds.EntityLock method*), 270
unregister_contentfilter() (*rti.connextdds.DomainParticipant method*), 170
unregister_instance() (*rti.connextdds.DataWriter method*), 125

unregister_instance() (*rti.connextdds.DynamicData.DataWriter method*), 210
unregister_instance() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 407
unregister_instance() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 465
unregister_instance() (*rti.connextdds.ServiceRequest.DataWriter method*), 565
unregister_instance() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 629
unregister_instance() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 685
unregister_instance_async() (*rti.connextdds.DynamicData.DataWriter method*), 210
unregister_instance_async() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 407
unregister_instance_async() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 465
unregister_instance_async() (*rti.connextdds.ServiceRequest.DataWriter method*), 566
unregister_instance_async() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 629
unregister_instance_async() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 685
unregister_thread() (*in module rti.connextdds*), 791
unregister_type() (*rti.connextdds.DomainParticipant method*), 170
UNREGISTERED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 143
UNREGISTERED (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 142
unregistered_instance_count (*rti.connextdds.DataWriterCacheStatus property*), 127
unregistered_instance_count_peak (*rti.connextdds.DataWriterCacheStatus property*), 127
unretain() (*rti.connextdds.IEntity method*), 314
unretain() (*rti.connextdds.TopicQuery method*), 715
UnsupportedError, 771
update() (*rti.connextdds.DynamicData method*), 254
url_profile (*rti.connextdds.QosProviderParams property*), 507
use_count (*rti.connextdds.IEntity property*), 314
use_reader_content_filter() (*rti.connextdds.TopicQuery static method*), 715
USE_STANDARD_RELATED_SAMPLE_PID_BIT (*rti.connextdds.compliance.RpcMask attribute*), 791
user (*rti.connextdds.LogCategory attribute*), 357
user (*rti.connextdds.LogCategory.LogCategory attribute*), 356
user (*rti.connextdds.LogFacility attribute*), 358
user_data (*rti.connextdds.DataReaderQos property*), 110
user_data (*rti.connextdds.DataWriterQos property*), 139
user_data (*rti.connextdds.DomainParticipantQos property*), 178
user_data (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
user_data (*rti.connextdds.PublicationBuiltinTopicData property*), 485
user_data (*rti.connextdds.SubscriptionBuiltinTopicData property*), 649
user_forwarding_level (*rti.connextdds.MonitoringLoggingForwardingSettings property*), 370
USER_MULTICAST (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
user_multicast_port_offset (*rti.connextdds.RtpsWellKnownPorts property*), 527
USER_UNICAST (*rti.connextdds.RtpsReservedPortKindMask attribute*), 523
user_unicast_port_offset (*rti.connextdds.RtpsWellKnownPorts property*), 528
UserData (*class in rti.connextdds*), 772
UserDataSample (*class in rti.connextdds*), 772
UshortSeq (*in module rti.connextdds*), 772
UShortType (*in module rti.connextdds*), 756

## V

vacuum (*rti.connextdds.PersistentStorageSettings property*), 434
valid (*rti.connextdds.SampleInfo property*), 533
valid_response_data (*rti.connextdds.AcknowledgmentInfo property*), 38
value (*rti.connextdds.AcknowledgmentKind.AcknowledgmentKind property*), 40
value (*rti.connextdds.AckResponseData property*), 38
value (*rti.connextdds.BuiltinTopicKey property*), 63
value (*rti.connextdds.CdrPaddingKind.CdrPaddingKind property*), 66
value (*rti.connextdds.Cookie property*), 85
value (*rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind property*), 98
value (*rti.connextdds.DataRepresentation property*), 116
value (*rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind property*), 143
value (*rti.connextdds.DestinationOrderKind.DestinationOrderKind property*), 150
value (*rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind property*), 152

value (*rti.connextdds.DurabilityKind.DurabilityKind property*), 188
value (*rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind property*), 255
value (*rti.connextdds.ExtensibilityKind.ExtensibilityKind property*), 280
value (*rti.connextdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy property*), 294
value (*rti.connextdds.GroupData property*), 297
value (*rti.connextdds.heap_monitoring.SnapshotContentFormat.SnapshotContentFormat property*), 800
value (*rti.connextdds.heap_monitoring.SnapshotOutputFormat.SnapshotOutputFormat property*), 802
value (*rti.connextdds.HistoryKind.HistoryKind property*), 300
value (*rti.connextdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind property*), 318
value (*rti.connextdds.InstanceStateConsistencyKind property*), 326
value (*rti.connextdds.LivelinessKind.LivelinessKind property*), 342
value (*rti.connextdds.LocatorKind.LocatorKind property*), 350
value (*rti.connextdds.LogCategory.LogCategory property*), 356
value (*rti.connextdds.LogFacility property*), 358
value (*rti.connextdds.LogLevel.LogLevel property*), 359
value (*rti.connextdds.OwnershipKind.OwnershipKind property*), 387
value (*rti.connextdds.OwnershipStrength property*), 388
value (*rti.connextdds.PersistentJournalKind property*), 433
value (*rti.connextdds.PersistentSynchronizationKind property*), 435
value (*rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind property*), 437
value (*rti.connextdds.PrintFormatKind.PrintFormatKind property*), 442
value (*rti.connextdds.PrintFormat.PrintFormat property*), 440
value (*rti.connextdds.PublishModeKind.PublishModeKind property*), 493
value (*rti.connextdds.ReliabilityKind.ReliabilityKind property*), 514
value (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind property*), 517
value (*rti.connextdds.SequenceNumber property*), 543
value (*rti.connextdds.ServiceKind.ServiceKind property*), 546
value (*rti.connextdds.ServiceRequestId.ServiceRequestId property*), 584
value (*rti.connextdds.SyslogVerbosity property*), 656
value (*rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind property*), 659
value (*rti.connextdds.TopicData property*), 709
value (*rti.connextdds.TopicQuerySelectionKind.TopicQuerySelectionKind property*), 718
value (*rti.connextdds.TransportClassId.TransportClassId property*), 727
value (*rti.connextdds.TransportMulticast property*), 732
value (*rti.connextdds.TransportMulticastKind.TransportMulticastKind property*), 733
value (*rti.connextdds.TransportMulticastMapping property*), 735
value (*rti.connextdds.TransportPriority property*), 741
value (*rti.connextdds.TransportSelection property*), 742
value (*rti.connextdds.TransportUnicast property*), 742
value (*rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind property*), 749
value (*rti.connextdds.TypeKind.TypeKind property*), 754
value (*rti.connextdds.UserData property*), 772
value (*rti.connextdds.VendorId property*), 773
value (*rti.connextdds.Verbosity.Verbosity property*), 775
value (*rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind property*), 786
value (*rti.logging.distlog.LogLevel property*), 813
value (*rti.types.builtin.Bytes attribute*), 804
value (*rti.types.builtin.KeyedBytes attribute*), 804
value (*rti.types.builtin.KeyedString attribute*), 804
value (*rti.types.builtin.String attribute*), 804
values() (*rti.connextdds.StringMap method*), 598
ValuesView (*class in rti.connextdds*), 772
VENDOR (*rti.connextdds.compliance.RpcMask attribute*), 791
VENDOR (*rti.connextdds.compliance.XTypesMask attribute*), 794
vendor_builtin_endpoints (*rti.connextdds.ParticipantBuiltinTopicData property*), 426
VendorId (*class in rti.connextdds*), 773
VERBOSE (*rti.connextdds.PrintFormat attribute*), 440
VERBOSE (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
VERBOSE_TIMESTAMPED (*rti.connextdds.PrintFormat attribute*), 440
VERBOSE_TIMESTAMPED (*rti.connextdds.PrintFormat.PrintFormat attribute*), 439
Verbosity (*class in rti.connextdds*), 773
verbosity (*rti.connextdds.Logger property*), 362
verbosity() (*rti.logging.distlog.Logger static method*), 814
verbosity_by_category() (*rti.connextdds.Logger method*), 362
Verbosity.Verbosity (*class in rti.connextdds*), 773
view_state (*rti.connextdds.DataState property*), 117
view_state (*rti.connextdds.DataStateEx property*), 119
ViewState (*class in rti.connextdds*), 776
virtual_duplicate_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 97
virtual_guid (*rti.connextdds.DataReaderProtocol property*), 101
virtual_guid (*rti.connextdds.DataWriterProtocol property*), 129

virtual_guid (*rti.connextdds.PublicationBuiltinTopicData property*), 485
virtual_guid (*rti.connextdds.SubscriptionBuiltinTopicData property*), 649
virtual_heartbeat_period (*rti.connextdds.RtpsReliableWriterProtocol property*), 523
volatile (*rti.connextdds.Durability attribute*), 186
VOLATILE (*rti.connextdds.DurabilityKind attribute*), 188
VOLATILE (*rti.connextdds.DurabilityKind.DurabilityKind attribute*), 187

## W

wait() (*rti.connextdds.WaitSet method*), 780
wait_async() (*rti.connextdds.WaitSet method*), 780
wait_for_acknowledgments() (*rti.connextdds.DataWriter method*), 125
wait_for_acknowledgments() (*rti.connextdds.DynamicData.DataWriter method*), 211
wait_for_acknowledgments() (*rti.connextdds.IAnyDataWriter method*), 305
wait_for_acknowledgments() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 408
wait_for_acknowledgments() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 466
wait_for_acknowledgments() (*rti.connextdds.Publisher method*), 495
wait_for_acknowledgments() (*rti.connextdds.ServiceRequest.DataWriter method*), 566
wait_for_acknowledgments() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 629
wait_for_acknowledgments() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 686
wait_for_asynchronous_publishing() (*rti.connextdds.DataWriter method*), 125
wait_for_asynchronous_publishing() (*rti.connextdds.DynamicData.DataWriter method*), 211
wait_for_asynchronous_publishing() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 408
wait_for_asynchronous_publishing() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 466
wait_for_asynchronous_publishing() (*rti.connextdds.Publisher method*), 495
wait_for_asynchronous_publishing() (*rti.connextdds.ServiceRequest.DataWriter method*), 566
wait_for_asynchronous_publishing() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 630
wait_for_asynchronous_publishing() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 686
wait_for_asynchronous_publishing_async() (*rti.connextdds.DataWriter method*), 126
wait_for_asynchronous_publishing_async() (*rti.connextdds.DynamicData.DataWriter method*), 211
wait_for_asynchronous_publishing_async() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 408
wait_for_asynchronous_publishing_async() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 466
wait_for_asynchronous_publishing_async() (*rti.connextdds.ServiceRequest.DataWriter method*), 566
wait_for_asynchronous_publishing_async() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 630
wait_for_asynchronous_publishing_async() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 686
wait_for_historical_data() (*rti.connextdds.DataReader method*), 95
wait_for_historical_data() (*rti.connextdds.DynamicData.DataReader method*), 202
wait_for_historical_data() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
wait_for_historical_data() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
wait_for_historical_data() (*rti.connextdds.ServiceRequest.DataReader method*), 557
wait_for_historical_data() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 620
wait_for_historical_data() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 677
wait_for_historical_data_async() (*rti.connextdds.DataReader method*), 95
wait_for_historical_data_async() (*rti.connextdds.DynamicData.DataReader method*), 202
wait_for_historical_data_async() (*rti.connextdds.ParticipantBuiltinTopicData.DataReader method*), 398
wait_for_historical_data_async() (*rti.connextdds.PublicationBuiltinTopicData.DataReader method*), 456
wait_for_historical_data_async() (*rti.connextdds.ServiceRequest.DataReader method*),

557
wait_for_historical_data_async() (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader method*), 620
wait_for_historical_data_async() (*rti.connextdds.TopicBuiltinTopicData.DataReader method*), 677
wait_for_replies_async() (*rti.rpc.Requester method*), 808
wait_for_requests_async() (*rti.rpc.Replier method*), 807
wait_for_service_async() (*rti.rpc.ClientBase method*), 805
wait_for_service_async() (*rti.rpc.Requester method*), 808
wait_one() (*rti.connextdds.WaitSet method*), 780
wait_one_async() (*rti.connextdds.WaitSet method*), 780
WaitSet (*class in rti.connextdds*), 778
WaitSetProperty (*class in rti.connextdds*), 780
WAL (*rti.connextdds.PersistentJournalKind attribute*), 432
WARNING (*rti.connextdds.LogLevel attribute*), 359
WARNING (*rti.connextdds.LogLevel.LogLevel attribute*), 359
WARNING (*rti.connextdds.SyslogVerbosity attribute*), 655
WARNING (*rti.connextdds.Verbosity attribute*), 775
WARNING (*rti.connextdds.Verbosity.Verbosity attribute*), 774
WARNING (*rti.logging.distlog.LogLevel attribute*), 813
warning() (*rti.connextdds.Logger method*), 362
warning() (*rti.logging.distlog.Logger static method*), 814
WcharSeq (*class in rti.connextdds*), 781
WcharType (*class in rti.connextdds*), 783
WEB_INTEGRATION (*rti.connextdds.ServiceKind attribute*), 546
WEB_INTEGRATION (*rti.connextdds.ServiceKind.ServiceKind attribute*), 546
wire_protocol (*rti.connextdds.DomainParticipantQos property*), 178
WireProtocol (*class in rti.connextdds*), 783
WireProtocolAutoKind (*class in rti.connextdds*), 784
WireProtocolAutoKind.WireProtocolAutoKind (*class in rti.connextdds*), 785
write() (*rti.connextdds.DataWriter method*), 126
write() (*rti.connextdds.DynamicData.DataWriter method*), 211
write() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 409
write() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 466
write() (*rti.connextdds.ServiceRequest.DataWriter method*), 567
write() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 630
write() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 687
write_async() (*rti.connextdds.DynamicData.DataWriter method*), 212
write_async() (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter method*), 409
write_async() (*rti.connextdds.PublicationBuiltinTopicData.DataWriter method*), 467
write_async() (*rti.connextdds.ServiceRequest.DataWriter method*), 567
write_async() (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter method*), 631
write_async() (*rti.connextdds.TopicBuiltinTopicData.DataWriter method*), 687
WriteParams (*class in rti.connextdds*), 787
writer_attach() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
writer_attach() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
writer_attach() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
writer_attach() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 581
writer_attach() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 645
writer_attach() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 701
writer_compile() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
writer_compile() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
writer_compile() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
writer_compile() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 581
writer_compile() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 645
writer_compile() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 701
writer_compression_level (*rti.connextdds.CompressionSettings property*), 78
writer_compression_threshold (*rti.connextdds.CompressionSettings property*), 78
writer_data_lifecycle (*rti.connextdds.DataWriterQos property*), 139
writer_data_tag_list_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 183
writer_data_tag_string_max_length

(*rti.connextdds.DomainParticipantResourceLimits property*), 183
writer_depth (*rti.connextdds.Durability property*), 186
writer_detach() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
writer_detach() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
writer_detach() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
writer_detach() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 581
writer_detach() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 646
writer_detach() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 702
writer_evaluate() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
writer_evaluate() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
writer_evaluate() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
writer_evaluate() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 581
writer_evaluate() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 646
writer_evaluate() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 702
writer_evaluate_helper() (*rti.connextdds.DynamicData.WriterContentFilterHelper method*), 229
writer_evaluate_helper() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilterHelper method*), 425
writer_evaluate_helper() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper method*), 483
writer_evaluate_helper() (*rti.connextdds.ServiceRequest.WriterContentFilterHelper method*), 582
writer_evaluate_helper() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper method*), 646
writer_evaluate_helper() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper method*), 702
writer_finalize() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
writer_finalize() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
writer_finalize() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
writer_finalize() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 581
writer_finalize() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 646
writer_finalize() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 702
writer_guid (*rti.connextdds.SampleIdentity property*), 531
writer_instance_cache_allocation (*rti.connextdds.PersistentStorageSettings property*), 434
writer_loaned_sample_allocation (*rti.connextdds.DataWriterResourceLimits property*), 140
writer_memory_state (*rti.connextdds.PersistentStorageSettings property*), 434
writer_property_list_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 183
writer_property_string_max_length (*rti.connextdds.DomainParticipantResourceLimits property*), 183
writer_removed_batch_sample_dropped_sample_count (*rti.connextdds.DataReaderCacheStatus property*), 97
writer_resource_limits (*rti.connextdds.DataWriterQos property*), 139
writer_return_loan() (*rti.connextdds.DynamicData.WriterContentFilter method*), 228
writer_return_loan() (*rti.connextdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 424
writer_return_loan() (*rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter method*), 482
writer_return_loan() (*rti.connextdds.ServiceRequest.WriterContentFilter method*), 582
writer_return_loan() (*rti.connextdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 646
writer_return_loan() (*rti.connextdds.TopicBuiltinTopicData.WriterContentFilter method*), 702
writer_sample_cache_allocation (*rti.connextdds.PersistentStorageSettings property*), 434
writer_side_filter_optimization (*rti.connextdds.ExpressionProperty property*), 279
writer_user_data_max_length (*rti.connextdds.DomainParticipantResourceLimits*

*property*), 183
WriterDataLifecycle (*class in rti.connextdds*), 787
WSTRING_TYPE (*rti.connextdds.TypeKind attribute*), 755
WSTRING_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*), 753
WstringSeq (*class in rti.connextdds*), 788
WStringType (*class in rti.connextdds*), 778

## X

XCDR (*rti.connextdds.DataRepresentation attribute*), 115
XCDR2 (*rti.connextdds.DataRepresentation attribute*), 115
XML (*rti.connextdds.DataRepresentation attribute*), 115
XML (*rti.connextdds.PrintFormatKind attribute*), 442
XML (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*), 441
xml (*rti.connextdds.PrintFormatProperty attribute*), 443
XTypesMask (*class in rti.connextdds.compliance*), 794

## Z

ZERO (*rti.connextdds.CdrPaddingKind attribute*), 66
ZERO (*rti.connextdds.CdrPaddingKind.CdrPaddingKind attribute*), 66
zero (*rti.connextdds.Duration attribute*), 192
zero (*rti.connextdds.SequenceNumber attribute*), 543
zero (*rti.connextdds.Time attribute*), 665
ZLIB (*rti.connextdds.CompressionIdMask attribute*), 75