PyOpenDDS Documentation

Release 0.2-dev

Fred Hornsey

Mar 31, 2022

CONTENTS:

1 Getting Started **1**

2 IDL-to-Python Mapping Plan **3**
- 2.1 Primitive Types . . . 3
- 2.2 Composite Types . . . 4

3 API Reference **5**
- 3.1 pyopendds.constants . . . 5
- 3.2 pyopendds.DataReader . . . 6
- 3.3 pyopendds.DataWriter . . . 6
- 3.4 pyopendds.DomainParticipant . . . 6
- 3.5 pyopendds.exceptions . . . 7
- 3.6 pyopendds.init_opendds . . . 7
- 3.7 pyopendds.Publisher . . . 8
- 3.8 pyopendds.Subscriber . . . 8
- 3.9 pyopendds.Topic . . . 8
- 3.10 pyopendds.util . . . 9

4 Indices and tables **11**

Python Module Index **13**

Index **15**

CHAPTER

ONE

GETTING STARTED

Once `$DDS_ROOT/setenv.sh` has been sourced or the equivalent, run the commands below in this directory.

```
# Build and Install PyOpenDDS
pip install .

# Build Basic Test
cd tests/basic_test
mkdir build
cd build
cmake ..
make

# Build and Install Basic Test Python Type Support
itl2py -o basic_output basic_idl opendds_generated/basic.itl
# If using OpenDDS 3.19 or before, then just specify basic.itl
cd basic_output
pip install .

# Run Basic Test
cd ../..
bash run_test.sh
```

CHAPTER

TWO

IDL-TO-PYTHON MAPPING PLAN

2.1 Primitive Types

- IDL `boolean` maps to Python `bool`.
- All IDL integer types map to Python `int`.
  - During serialization, if the value it does not fit in the types range, raise a `ValueError`.
- IDL `float` and `double` map to Python `float`. IDL `long double` and `fixed` map to Python `decimal.Decimal`.

2.1.1 Character Types

All IDL characters and strings map to Python `str`.

Unlike C strings, Python `str` requires the encoding to be known. To help facilitate this, by default characters and strings will be assumed to be UTF-8 and wide characters and strings will be assumed to be UTF-16. The encoding will be able to be specified either using a global implementation option or manually using this IDL annotation:

```
@annotion encoding {
    string platform default "*";
    string value;
};
```

For Python, `platform` can be left to default or set to `python` and `value` should be a valid Python codec. Here is the list of codecs for Python 3.7. As an example, if you wanted to use ISO-8859-10 "on the wire", you could write something like this:

```
struct Data {
    @encoding(platform="python", value="latin6")
    string put_swedish_here;
};
```

During serialization and deserialization, encoding will be handled automatically, but will be subject to `UnicodeError` if there is a problem with the encoding.

Alternatively `value` can be set to `"none"` to represent that no automatic encoding and decoding should be done. This is probably the behavior many other IDL mappings and would probably be the default if the annotation was adopted in other implementation. For Python this will change the type from `str` to `bytes`, which better represents the idea of string of bytes of uncertain encoding.

During serialization, raise a `ValueError` if the size of the encoded data is larger than the limits of the IDL type. For example: assigning Python `"more than a byte"` to a IDL field of type `char`.

2.2 Composite Types

- IDL arrays and sequences map to Python `list`
  - During serialization, if the IDL type is an array or bounded sequence, raise `ValueError` if the element count of the list is out of the valid range.
- IDL structures map to Python dataclasses or equivalent.
- IDL `enum` map to Python enum.IntFlag

2.2.1 Unions

This IDL:

```
enum EnumType {
    A, B, C
};

union UnionType switch (EnumType) {
case A:
    long number;
case B:
case C:
    char character;
};
```

Will produce Python like:

```
class UnionType:
    def __init__(self):
        self._d = None

    @property
    def number(self):
        if self._d != EmunType.A:
            raise # TODO What kind of Error does this need to be?
        return self._number

    @number.setter
    def number(self, value: int):
        self._d = EnumType.A
        self._number = value
```

CHAPTER

THREE

API REFERENCE

Modules

pyopendds.constants	
pyopendds.DataReader(subscriber, topic[, ...])	
pyopendds.DataWriter()	
pyopendds.DomainParticipant(domain[, qos, ...])	
pyopendds.exceptions	
pyopendds.init_opendds(*args[, ...])	Initialize OpenDDS using the TheParticipantFactoryWithArgs macro while passing the positional arguments in.
pyopendds.Publisher(participant[, qos, listener])	
pyopendds.Subscriber(participant[, qos, ...])	
pyopendds.Topic(participant, name, topic_type)	
pyopendds.util	

3.1 pyopendds.constants

Classes

`InstanceState`(value)	An enumeration.
`ReturnCode`(value)	An enumeration.
`SampleState`(value)	An enumeration.
`StatusKind`(value)	An enumeration.
`ViewState`(value)	An enumeration.

3.2 pyopendds.DataReader

class pyopendds.**DataReader**(*subscriber:* Subscriber, *topic:* Topic, *qos=None*, *listener=None*)

__init__(*subscriber:* Subscriber, *topic:* Topic, *qos=None*, *listener=None*)

Methods

`__init__`(subscriber, topic[, qos, listener])
`take_next_sample`()
`wait_for`(status, timeout)

3.3 pyopendds.DataWriter

class pyopendds.**DataWriter**

__init__()

Methods

`__init__`()

3.4 pyopendds.DomainParticipant

class pyopendds.**DomainParticipant**(*domain:* *int*, *qos=None*, *listener=None*)

__init__(*domain:* *int*, *qos=None*, *listener=None*)

Methods

`__init__`(domain[, qos, listener])
`create_publisher`([qos, listener])
`create_subscriber`([qos, listener])
`create_topic`(name, topic_type[, qos, listener])

3.5 pyopendds.exceptions

Exceptions

`AlreadyDeletedReturnCodeError`([unknown_code])	
`BadParameterReturnCodeError`([unknown_code])	
`ErrorReturnCodeError`([unknown_code])	
`IllegalOperationReturnCodeError`([unknown_code])	
`ImmutablePolicyReturnCodeError`([unknown_code])	
`InconsistentPolicyReturnCodeError`([unknown_code])	
`NoDataReturnCodeError`([unknown_code])	
`NotEnabledReturnCodeError`([unknown_code])	
`OutOfResourcesReturnCodeError`([unknown_code])	
`PreconditionNotMetReturnCodeError`([unknown_code])	
`PyOpenDDS_Error`	Base for all errors in PyOpenDDS
`ReturnCodeError`([unknown_code])	Raised when a ReturnCode_t other than RETURN-CODE_OK was returned from a OpenDDS function that returns ReturnCode_t.
`TimeoutReturnCodeError`([unknown_code])	
`UnsupportedReturnCodeError`([unknown_code])	

3.6 pyopendds.init_opendds

`pyopendds.`**`init_opendds`**(**args: str, default_rtps=True, opendds_debug_level=0*) → None

Initialize OpenDDS using the TheParticipantFactoryWithArgs macro while passing the positional arguments in.

default_rtps In PyOpenDDS the default discovery and transport is RTPS. Pass False to this to stop PyOpenDDS from setting up RTPS and let OpenDDS default to In OpenDDS the default discovery is InfoRepo and the default transport is TCP.

opendds_debug_level Debug logging level in OpenDDS which goes from 0 (off) to 10 (most verbose). It is printed to stdout.

3.7 pyopendds.Publisher

class pyopendds.**Publisher**(*participant:* DomainParticipant, *qos=None*, *listener=None*)

__init__(*participant:* DomainParticipant, *qos=None*, *listener=None*)

Methods

`__init__`(participant[, qos, listener])
`create_datawriter`(topic[, qos, listener])

3.8 pyopendds.Subscriber

class pyopendds.**Subscriber**(*participant:* DomainParticipant, *qos=None*, *listener=None*)

__init__(*participant:* DomainParticipant, *qos=None*, *listener=None*)

Methods

`__init__`(participant[, qos, listener])
`create_datareader`(topic[, qos, listener])

3.9 pyopendds.Topic

class pyopendds.**Topic**(*participant:* DomainParticipant, *name:* *str*, *topic_type:* *Any*, *qos=None*, *listener=None*)

__init__(*participant:* DomainParticipant, *name:* *str*, *topic_type:* *Any*, *qos=None*, *listener=None*)

Methods

`__init__`(participant, name, topic_type[, ...])

3.10 pyopendds.util

Functions

`normalize_time_duration`(duration)

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

pyopendds.constants, 5
pyopendds.exceptions, 7
pyopendds.util, 9

INDEX

Symbols

`__init__()` (*pyopendds.DataReader method*), 6
`__init__()` (*pyopendds.DataWriter method*), 6
`__init__()` (*pyopendds.DomainParticipant method*), 6
`__init__()` (*pyopendds.Publisher method*), 8
`__init__()` (*pyopendds.Subscriber method*), 8
`__init__()` (*pyopendds.Topic method*), 8

D

`DataReader` (*class in pyopendds*), 6
`DataWriter` (*class in pyopendds*), 6
`DomainParticipant` (*class in pyopendds*), 6

I

`init_opendds()` (*in module pyopendds*), 7

M

`module`
    `pyopendds.constants`, 5
    `pyopendds.exceptions`, 7
    `pyopendds.util`, 9

P

`Publisher` (*class in pyopendds*), 8
`pyopendds.constants`
    `module`, 5
`pyopendds.exceptions`
    `module`, 7
`pyopendds.util`
    `module`, 9

S

`Subscriber` (*class in pyopendds*), 8

T

`Topic` (*class in pyopendds*), 8