DataCite Documentation

Release 1.1.3

Invenio Software

Mar 20, 2023

CONTENTS

Python API wrapper for the DataCite Metadata Store API and DataCite XML generation.

CHAPTER

ONE

INSTALLATION

The datacite package is on PyPI so all you need is:

```
$ pip install datacite
```

CHAPTER

TWO

USAGE

The datacite package implements a Python client for DataCite MDS API and DataCite REST API. You can find below full usage example of the DataCite MDS client API wrapper. Please see the DataCite MDS API documentation for further information.

```
from datacite import DataCiteMDSClient, schema42

# If you want to generate XML for earlier versions, you need to use either the
# schema31, schema40 or schema41 instead.

data = {
    'identifiers': [{
        'identifierType': 'DOI',
        'identifier': '10.1234/foo.bar',
    }],
    'creators': [
        {'name': 'Smith, John'},
    ],
    'titles': [
        {'title': 'Minimal Test Case', }
    ],
    'publisher': 'Invenio Software',
    'publicationYear': '2015',
    'types': {
        'resourceType': 'Dataset',
        'resourceTypeGeneral': 'Dataset'
    },
    'schemaVersion': 'http://datacite.org/schema/kernel-4',
}

# Validate dictionary
assert schema42.validate(data)

# Generate DataCite XML from dictionary.
doc = schema42.tostring(data)

# Initialize the MDS client.
d = DataCiteMDSClient(
    username='MYDC.MYACCOUNT',
    password='mypassword',
    prefix='10.1234',
```

(continues on next page)

(continued from previous page)

```
    test_mode=True,
)

# Set metadata for DOI
d.metadata_post(doc)

# Mint new DOI
d.doi_post('10.1234/test-doi', 'http://example.org/test-doi')

# Get DOI location
location = d.doi_get("10.1234/test-doi")

# Set alternate URL for content type (available through content negotiation)
d.media_post(
    "10.1234/test-doi",
    {"application/json": "http://example.org/test-doi/json/",
     "application/xml": "http://example.org/test-doi/xml/"}
)

# Get alternate URLs
mapping = d.media_get("10.1234/test-doi")
assert mapping["application/json"] == "http://example.org/test-doi/json/"

# Get metadata for DOI
doc = d.metadata_get("10.1234/test-doi")

# Make DOI inactive
d.metadata_delete("10.1234/test-doi")
```

You can find below an usage example of the DataCite REST client API wrapper. Please see the DataCite REST API documentation for further information.

```
import os
from datacite import DataCiteRESTClient, schema42

# If you want to generate XML for earlier versions, you need to use either the
# schema31, schema40 or schema41 instead.

data = {
    'identifiers': [{
        'identifierType': 'DOI',
        'identifier': '10.1234/foo.bar',
    }],
    'creators': [
        {'name': 'Smith, John'},
    ],
    'titles': [
        {'title': 'Minimal Test Case', }
    ],
    'publisher': 'Invenio Software',
    'publicationYear': '2015',
    'types': {
```

(continues on next page)

(continued from previous page)

```
        'resourceType': 'Dataset',
        'resourceTypeGeneral': 'Dataset'
    },
    'schemaVersion': 'http://datacite.org/schema/kernel-4',
}

# Validate dictionary
assert schema42.validate(data)

# Generate DataCite XML from dictionary.
doc = schema42.tostring(data)

# Initialize the REST client.
d = DataCiteRESTClient(
    username="MYDC.MYACCOUNT",
    password="mypassword",
    prefix="10.1234",
    test_mode=True
)

# Reserve a draft DOI
doi = d.draft_doi()
```

Please see the DataCite Testing guide to know how to test this client with your test credentials.

CHAPTER

THREE

METADATA STORE API

Python API wrapper for the DataCite API.

class datacite.**DataCiteMDSClient**(*username*, *password*, *prefix*, *test_mode=False*, *url=None*, *timeout=None*)

DataCite MDS API client wrapper.

Warning: The DataCite MDS API is being maintained but is no longer actively developed.

doi_get(*doi*)

Get the URL where the resource pointed by the DOI is located.

Parameters
doi – DOI name of the resource.

doi_post(*new_doi*, *location*)

Mint new DOI.

Parameters

- **new_doi** – DOI name for the new resource.
- **location** – URL where the resource is located.

Returns
"CREATED" or "HANDLE_ALREADY_EXISTS".

media_get(*doi*)

Get list of pairs of media type and URLs associated with a DOI.

Parameters
doi – DOI name of the resource.

media_post(*doi*, *media*)

Add/update media type/urls pairs to a DOI.

Standard domain restrictions check will be performed.

Parameters
media – Dictionary of (mime-type, URL) key/value pairs.

Returns
"OK"

metadata_delete(*doi*)

Mark as 'inactive' the metadata set of a DOI resource.

Parameters
doi – DOI name of the resource.

Returns
"OK"

`metadata_get`(*doi*)

Get the XML metadata associated to a DOI name.

Parameters
doi – DOI name of the resource.

`metadata_post`(*metadata*)

Set new metadata for an existing DOI.

Metadata should follow the DataCite Metadata Schema: http://schema.datacite.org/

Parameters
metadata – XML format of the metadata.

Returns
"CREATED" or "HANDLE_ALREADY_EXISTS"

`class datacite.DataCiteRESTClient`(*username*, *password*, *prefix*, *test_mode=False*, *url=None*, *timeout=None*)

DataCite REST API client wrapper.

`check_doi`(*doi*)

Check doi structure.

Check that the doi has a form 12.12345/123 with the prefix defined

`delete_doi`(*doi*)

Delete a doi.

This will only work for draft dois

Parameters
doi – DOI (e.g. 10.123/456)

Returns

`doi_get`(*doi*)

Get the URL where the resource pointed by the DOI is located.

Parameters
doi – DOI name of the resource.

`draft_doi`(*metadata=None*, *doi=None*)

Create a draft doi.

A draft DOI can be deleted

If doi is not provided, DataCite will automatically create a DOI with a random, recommended DOI suffix

Parameters

- **metadata** – metadata for the DOI
- **doi** – DOI (e.g. 10.123/456)

Returns

`get_doi`(*doi*)

Get the URL where the resource pointed by the DOI is located.

Parameters
doi – DOI name of the resource.

`get_media`(*doi*)

Get list of pairs of media type and URLs associated with a DOI.

Parameters
doi – DOI name of the resource.

`get_metadata`(*doi*)

Get the JSON metadata associated to a DOI name.

Parameters
doi – DOI name of the resource.

`hide_doi`(*doi*)

Hide a previously registered DOI.

This DOI will no longer be found in DataCite Search

Parameters
doi – DOI to hide e.g. 10.12345/1.

Returns

`media_get`(*doi*)

Get list of pairs of media type and URLs associated with a DOI.

Parameters
doi – DOI name of the resource.

`metadata_get`(*doi*)

Get the JSON metadata associated to a DOI name.

Parameters
doi – DOI name of the resource.

`post_doi`(*data*)

Post a new JSON payload to DataCite.

`private_doi`(*metadata*, *url*, *doi=None*)

Publish a doi in a registered state.

A DOI generated by this method will not be found in DataCite Search

This DOI cannot be deleted

If doi is not provided, DataCite will automatically create a DOI with a random, recommended DOI suffix

Metadata should follow the DataCite Metadata Schema: http://schema.datacite.org/

Parameters
metadata – JSON format of the metadata.

Returns

`public_doi`(*metadata*, *url*, *doi=None*)

Create a public doi.

This DOI will be public and cannot be deleted

If doi is not provided, DataCite will automatically create a DOI with a random, recommended DOI suffix

Metadata should follow the DataCite Metadata Schema: http://schema.datacite.org/

Parameters

- **metadata** – JSON format of the metadata.
- **doi** – DOI (e.g. 10.123/456)
- **url** – URL where the doi will resolve.

Returns

put_doi(*doi*, *data*)

Put a JSON payload to DataCite for an existing DOI.

show_doi(*doi*)

Show a previously registered DOI.

This DOI will be found in DataCite Search

Parameters
doi – DOI to hide e.g. 10.12345/1.

Returns

update_doi(*doi*, *metadata=None*, *url=None*)

Update the metadata or url for a DOI.

Parameters

- **url** – URL where the doi will resolve.
- **metadata** – JSON format of the metadata.

Returns

update_url(*doi*, *url*)

Update the url of a doi.

Parameters

- **url** – URL where the doi will resolve.
- **doi** – DOI (e.g. 10.123/456)

Returns

3.1 Errors

Errors for the DataCite API.

MDS error responses will be converted into an exception from this module. Connection issues raises *datacite.errors.HttpError* while DataCite MDS error responses raises a subclass of *datacite.errors.DataCiteError*.

exception datacite.errors.**DataCiteBadRequestError**

Bad request error.

Bad requests can include e.g. invalid XML, wrong domain, wrong prefix. Request body must be exactly two lines: DOI and URL One or more of the specified mime-types or urls are invalid (e.g. non supported mimetype, not allowed url domain, etc.)

exception datacite.errors.**DataCiteError**

Exception raised when the server returns a known HTTP error code.

Known HTTP error codes include:

- 204 No Content
- 400 Bad Request
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 410 Gone (deleted)

static factory(*err_code*, **args*)

Create exceptions through a Factory based on the HTTP error code.

exception datacite.errors.**DataCiteForbiddenError**

Login problem, dataset belongs to another party or quota exceeded.

exception datacite.errors.**DataCiteGoneError**

Requested dataset was marked inactive (using DELETE method).

exception datacite.errors.**DataCiteNoContentError**

DOI is known to MDS, but not resolvable.

This might be due to handle's latency.

exception datacite.errors.**DataCiteNotFoundError**

DOI does not exist in the database.

exception datacite.errors.**DataCitePreconditionError**

Metadata must be uploaded first.

exception datacite.errors.**DataCiteRequestError**

A DataCite request error. You made an invalid request.

Base class for all 4XX-related HTTP error codes as well as 204.

exception datacite.errors.**DataCiteServerError**

An internal server error happened on the DataCite end. Try later.

Base class for all 5XX-related HTTP error codes.

exception datacite.errors.**DataCiteUnauthorizedError**

Bad username or password.

exception datacite.errors.**HttpError**

Exception raised when a connection problem happens.

CHAPTER

FOUR

DATACITE V3.1 XML GENERATION

DataCite v3.1 JSON to XML transformations.

datacite.schema31.**dump_etree**(*data*)

Convert JSON dictionary to DataCite v3.1 XML as ElementTree.

datacite.schema31.**tostring**(*data*, ***kwargs*)

Convert JSON dictionary to DataCite v3.1 XML as string.

datacite.schema31.**validate**(*data*)

Validate DataCite v3.1 JSON dictionary.

CHAPTER

FIVE

DATACITE V4.0 XML GENERATION

DataCite v4.0 JSON to XML transformations.

datacite.schema40.**dump_etree**(*data*)

Convert JSON dictionary to DataCite v4.0 XML as ElementTree.

datacite.schema40.**tostring**(*data*, ***kwargs*)

Convert JSON dictionary to DataCite v4.0 XML as string.

datacite.schema40.**validate**(*data*)

Validate DataCite v4.0 JSON dictionary.

CHAPTER

SIX

DATACITE V4.1 XML GENERATION

DataCite v4.1 JSON to XML transformations.

datacite.schema41.**dump_etree**(*data*)

Convert JSON dictionary to DataCite v4.1 XML as ElementTree.

datacite.schema41.**tostring**(*data*, ***kwargs*)

Convert JSON dictionary to DataCite v4.1 XML as string.

datacite.schema41.**validate**(*data*)

Validate DataCite v4.1 JSON dictionary.

CHAPTER

SEVEN

DATACITE V4.2 XML GENERATION

DataCite v4.2 JSON to XML transformations.

datacite.schema42.**dump_etree**(*data*)

Convert JSON dictionary to DataCite v4.2 XML as ElementTree.

datacite.schema42.**tostring**(*data*, ***kwargs*)

Convert JSON dictionary to DataCite v4.2 XML as string.

datacite.schema42.**validate**(*data*)

Validate DataCite v4.2 JSON dictionary.

CHAPTER

EIGHT

CHANGES

Version v1.1.3 (released 2023-03-20):

- Updates dependency versions and adds python 3.9 support
- Changes internal definition name for affiliation in 4.3 schema

Version v1.1.2 (released 2021-06-22):

- Standardizes function names in DataCiteRESTClient. Old functions will be depreciated in a future release

Version v1.1.1 (released 2021-04-20):

- Fixes DataCiteRESTClient attributes' type. Prefix, username and password are always cast to string.

Version v1.1.0 (released 2021-04-15):

- Adds full support for DataCite Metadata Schema v4.2 and v4.3 XML generation.
- Uses Official DataCite JSON Schema, which has the following notable changes from the previous schema:
  - Uses "identifiers" which is a combination of the XML "identifier" and "alternativeIdentifiers" elements
  - "creatorName" is now "name"
  - "contributorName" is now "name"
  - "affiliations" is now "affiliation" (is still an array)
  - "affilition" is now "name"
  - There is no longer a funder identifier object (the identifier and type are just elements)
- Removes Python 2 support
- Removes the old way of testing with DataCite: test mode for the MDS APIs and the test DOI 10.5072

Version v1.0.1 (released 2018-03-08):

- Fixes schema location url for DataCite v4.1

Version v1.0.0 (released 2018-02-28):

- Adds full support for DataCite Metadata Schema v4.1 XML generation.

Version v0.3.0 (released 2016-11-18):

- Adds full support for DataCite Metadata Schema v4.0 XML generation.
- Adds the message from the server in the error exceptions.

Version v0.2.2 (released 2016-09-23):

- Fixes issue with generated order of nameIdentifier and affiliation tags.

Version v0.2.1 (released 2016-03-29):

- Fixes issue with JSON schemas not being included when installing from PyPI.

Version v0.2.0 (released 2016-03-21):

- Adds DataCite XML generation support.

Version 0.1 (released 2015-02-25):

- Initial public release.

CHAPTER

NINE

CONTRIBUTING

Bug reports, feature requests, and other contributions are welcome. If you find a demonstrable problem that is caused by the code of this library, please:

1. Search for already reported problems.
2. Check if the issue has been fixed or is still reproducible on the latest *master* branch.
3. Create an issue with **a test case**.

If you create a feature branch, you can run the tests to ensure everything is operating correctly:

```
$ python setup.py test
```

CHAPTER

TEN

LICENSE

DataCite is free software; you can redistribute it and/or modify it under the terms of the Revised BSD License quoted below.

Copyright (C) 2015-2018 CERN. Copyright (C) 2018 Center for Open Science. Copyright (C) 2019 Caltech.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Note: In applying this license, CERN does not waive the privileges and immunities granted to it by virtue of its status as an Intergovernmental Organization or submit itself to any jurisdiction.

10.1 Authors

DataCite is developed for use in Invenio digital library software.

Contact us at info@inveniosoftware.org

10.1.1 Contributors

- Lars Holm Nielsen <lars.holm.nielsen@cern.ch>
- Laura Rueda Garcia <laura.rueda@cern.ch>
- Jiri Kuncar <jiri.kuncar@cern.ch>
- Sami Hiltunen <sami.mikael.hiltunen@cern.ch>
- Tibor Simko <tibor.simko@cern.ch>
- Rémi Ducceschi <remi.ducceschi@gmail.com>

PYTHON MODULE INDEX

d

INDEX

S

T

U

V