DICOMweb Client Documentation

Release 0.59.1

Markus D. Herrmann

Feb 27, 2023

CONTENTS:

CHAPTER

ONE

INTRODUCTION

The dicomweb-client build distribution provides client intefaces for DICOMweb RESTful services QIDO-RS, WADO-RS and STOW-RS to search, retrieve and store DICOM objects over the web, respectively. For more information about DICOMweb please refer to the documentation of the DICOM standard, in particular PS3.18.

The dicomweb_client Python package exposes

- Application Programming Interface (API) (see *api* module)
- Command Line Interface (CLI) (see *cli* module)

CHAPTER

TWO

INSTALLATION GUIDE

2.1 Requirements

- Python (version 3.5 or higher)
- Python package manager pip

For support of image formats:

- JPEG (libjpeg or libjpeg-turbo)
- JPEG2000 (openjpeg)
- PNG (libpng)

2.2 Installation

Pre-build package available at PyPi:

```
pip install dicomweb-client
```

Additional dependencies required for extensions compatible with Google Cloud Platform (GCP) may be installed as:

```
pip install dicomweb-client[gcp]
```

Source code available at Github:

```
git clone https://github.com/ImagingDataCommons/dicomweb-client ~/dicomweb-client
pip install ~/dicomweb-client
```

CHAPTER

THREE

USER GUIDE

The client can be used with any DICOMweb server, such as dcm4che, orthanc or DICOMcloud.

3.1 Application Programming Interface (API)

3.1.1 Interacting with a remote DICOMweb server

To interact with a publicly accessible server, you only need to provide the `url` for the server address.

```
from dicomweb_client.api import DICOMwebClient

client = DICOMwebClient(url="https://mydicomwebserver.com")
```

Some servers expose the different DICOMweb RESTful services using different path prefixes. For example, the publicly accessible DICOMcloud server uses the prefixes `"qidors"`, `"wadors"`, and `"stowrs"` for QIDO-RS, WADO-RS, and STOW-RS, respectively. You can specify these prefixes using `qido_url_prefix`, `wado_url_prefix`, and `stow_url_prefix`.

```
from dicomweb_client.api import DICOMwebClient

client = DICOMwebClient(
    url="https://dicomcloud.azurewebsites.net",
    qido_url_prefix="qidors",
    wado_url_prefix="wadors",
    stow_url_prefix="stowrs"
)
```

3.1.2 Authentication and authorization

To interact with servers requiring authentication, `DICOMwebClient` accepts arbitrary authentication handlers derived from `requests.auth.AuthBase` (see here for details).

```
from requests.auth import HTTPBasicAuth
from dicomweb_client.api import DICOMwebClient
from dicomweb_client.session_utils import create_session_from_auth

auth = HTTPBasicAuth('myusername', 'mypassword')
session = create_session_from_auth(auth)
```

(continues on next page)

(continued from previous page)

```python
client = DICOMwebClient(
    url="https://mydicomwebserver.com",
    session=session
)
```

To simplify usage for basic HTTP authentication, you may also directly provide a username and password using the corresponding arguments.

```python
from dicomweb_client.api import DICOMwebClient
from dicomweb_client.session_utils import create_session_from_user_pass

session = create_session_from_user_pass(
    username='myusername',
    password='mypassword'
)

client = DICOMwebClient(
    url="https://mydicomwebserver.com",
    session=session
)
```

To interact with servers supporting token-based authorization, you can provide the access token using the `headers` argument (the header will be included in every client request message).

```python
from dicomweb_client.api import DICOMwebClient

access_token = "mytoken"
client = DICOMwebClient(
    url="https://mydicomwebserver.com",
    headers={"Authorization": "Bearer {}".format(access_token)}
)
```

To interact with servers requiring certificate-based authentication, you can provide the CA bundle and client certificate using the `ca_bundle` and `cert` arguments, respectively.

```python
from dicomweb_client.api import DICOMwebClient
from dicomweb_client.session_utils import (
    create_session,
    add_certs_to_session
)

session = create_session()
session = add_certs_to_session(
    session=session,
    ca_bundle="/path/to/ca.crt",
    cert="/path/to/cert.pem"
)

client = DICOMwebClient(url="https://mydicomwebserver.com")
```

To interact with a server of the Google Healthcare API requiring OpenID Connect based authentication and authorization, provide a session authenticated using the Google Cloud Platform (GCP) credentials. See GCP documentation for details.

The library provides the `gcp` extension, which facilitates interacting with the DICOMweb interface of the Google Healthcare API. Note that the `gcp` extension is optional and requires installation of the package distribution with the `gcp` extra requirements: `$ pip install dicomweb-client[gcp]`.

```
from dicomweb_client.api import DICOMwebClient
from dicomweb_client.ext.gcp.session_utils import create_session_from_gcp_credentials
from dicomweb_client.ext.gcp.uri import GoogleCloudHealthcareURL

session = create_session_from_gcp_credentials()

url = GoogleCloudHealthcareURL(
    project_id='my-project',
    location='us-east4',
    dataset_id='my-dataset',
    dicom_store_id='my-store'
)

client = DICOMwebClient(
    url=str(url),
    session=session
)
```

3.1.3 Accessing local DICOM Part10 files

The package provides the *dicomweb_client.api.DICOMfileClient* class for accessing data stored as DICOM Part10 files on a file system. The class exposes the same *dicomweb_client.api.DICOMClient* interface as the *dicomweb_client.api.DICOMwebClient* and can be used as a drop-in replacement.

To create a *dicomweb_client.api.DICOMfileClient* instance, you should pass the location of the desired data store on the file system. Note that the *file://* scheme designator is *required*.

```
from dicomweb_client.api import DICOMfileClient

client = DICOMfileClient("file:///path/to/directory")
```

3.1.4 STOW-RS StoreInstances

Store a single dataset obtained from a PS3.10 file:

```
import pydicom

filename = "/path/to/file.dcm"
dataset = pydicom.dcmread(filename)

client.store_instances(datasets=[dataset])
```

3.1.5 QIDO-RS SeachForStudies

Search for all studies (up to server-defined maximum set per call - see below to iteratively get all studies):

```
studies = client.search_for_studies()
```

Search for studies filtering by *PatientID*:

```
studies = client.search_for_studies(search_filters={'PatientID': 'ABC123'})
```

Note that attributes can be specified in `search_filters` using either the keyword or the tag:

```
studies = client.search_for_studies(search_filters={'00100020': 'ABC123'})
```

Search for all studies but limit the number of returned results using the `limit` parameter.

```
studies_subset = client.search_for_studies(limit=100)
```

A server may also automatically limit the number of results that it returns per search request. In this case, the method can be called repeatedly to request remaining results using the `offset` parameter.

```
studies = []
offset = 0
while True:
    subset = client.search_for_studies(offset=offset)
    if len(subset) == 0:
        break
    studies.extend(subset)
    offset += len(subset)
```

The same can be achieved more conveniently using the `get_remaining` parameter.

```
studies = client.search_for_studies(get_remaining=True)
```

3.1.6 QIDO-RS SeachForSeries

Search for all series:

```
series = client.search_for_series()
```

Search for series of a given study:

```
series = client.search_for_series('1.2.826.0.1.3680043.8.1055.1.20111103111148288.
↪98361414.79379639')
```

Search for series filtering by *AccessionNumber*:

```
series = client.search_for_series(search_filters={'AccessionNumber': '123456'})
```

Search for series filtering by *AccessionNumber* (using wildcard ? to match a range of numbers):

```
series = client.search_for_series(search_filters={'AccessionNumber': '12345?'})
```

Search for series filtering by *SeriesDescription*:

```
series = client.search_for_series(search_filters={'SeriesDescription': 'T2 AXIAL'})
```

Search for series filtering by *SeriesDescription* (using wildcard * to match a range of descriptions):

```
series = client.search_for_series(search_filters={'SeriesDescription': 'T2 AX*'})
```

Search for series filtering by *Modality*:

```
series = client.search_for_series(search_filters={'Modality': 'SM'})
```

3.1.7 QIDO-RS SeachForInstances

Search for all instances:

```
instances = client.search_for_instances()
```

Search for instances of a given study and series:

```
instances = client.search_for_instances(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪'
)
```

Search for instances filtering by *SOPClassUID*:

```
instances = client.search_for_instances(search_filters={'SOPClassUID': '1.2.840.10008.5.
↪1.4.1.1.2'})
```

3.1.8 WADO-RS RetrieveStudy

Retrieve instances of a given study:

```
instances = client.retrieve_study('1.2.826.0.1.3680043.8.1055.1.20111103111148288.
↪98361414.79379639')
```

3.1.9 WADO-RS RetrieveSeries

Retrieve instances of a given series:

```
instances = client.retrieve_series(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪'
)
```

Retrieve full instances of a given series using specific JPEG 2000 transfer syntax for encoding of bulk data:

```
instance = client.retrieve_instance(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    media_types=(('application/dicom', '1.2.840.10008.1.2.4.90', ), )
)
```

Retrieve bulk data of instances of a given series using specific JPEG 2000 transfer syntax:

```
instance = client.retrieve_instance(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    media_types=(('image/jp2', '1.2.840.10008.1.2.4.90', ), )
)
```

3.1.10 WADO-RS RetrieveInstance

Retrieve full instance using default Explicit VR Little Endian transfer syntax for encoding of bulk data:

```
instance = client.retrieve_instance(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪'
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534'
)
```

Retrieve full instance using specific JPEG 2000 transfer syntax for encoding of bulk data:

```
instance = client.retrieve_instance(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪'
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    media_types=(('application/dicom', '1.2.840.10008.1.2.4.90', ), )
)
```

Retrieve bulk data of instance using specific JPEG 2000 transfer syntax:

```
instance = client.retrieve_instance(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪'
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    media_types=(('image/jp2', '1.2.840.10008.1.2.4.90', ), )
)
```

3.1.11 WADO-RS RetrieveMetadata

Retrieve metadata for instances of a given study:

```
metadata = client.retrieve_study_metadata('1.2.826.0.1.3680043.8.1055.1.
↪20111103111148288.98361414.79379639')
```

Retrieve metadata for instances of a given series:

```
metadata = client.retrieve_series_metadata(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪'
)
```

Retrieve metadata for a particular instance:

```
metadata = client.retrieve_instance_metadata(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534'
)
```

Note: WADO-RS RetrieveMetadata always returns metadata at the instance-level, `retrieve_study_metadata()` and `retrieve_series_metadata()` return an array of metadata items for each instance belonging to a given study and series, respectively.

3.1.12 WADO-RS RetrieveFrames

Retrieve a set of frames with default transfer syntax ("application/octet-stream"):

```
frames = client.retrieve_instance_frames(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    frame_numbers=[1, 2]
)
```

Retrieve a set of frames of a given instances as JPEG compressed image:

```
frames = client.retrieve_instance_frames(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
```

(continues on next page)

(continued from previous page)

```
    frame_numbers=[1, 2],
    media_types=('image/jpeg', )
)
```

Retrieve a set of frames of a given instances as compressed image in any available format:

```
frames = client.retrieve_instance_frames(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    frame_numbers=[1, 2],
    media_types=('image/*', )
)
```

Retrieve a set of frames of a given instances as either JPEG 2000 or JPEG-LS compressed image:

```
frames = client.retrieve_instance_frames(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    frame_numbers=[1, 2],
    media_types=('image/jp2', 'image/jls', )
)
```

Retrieve a set of frames of a given instances as either JPEG, JPEG 2000 or JPEG-LS lossless compressed image using specific transfer syntaxes:

```
frames = client.retrieve_instance_frames(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    frame_numbers=[1, 2],
    media_types=(
        ('image/jpeg', '1.2.840.10008.1.2.4.57', ),
        ('image/jp2', '1.2.840.10008.1.2.4.90', ),
        ('image/jls', '1.2.840.10008.1.2.4.80', ),
    )
)
```

3.1.13 WADO-RS RetrieveBulkdata

Retrieve bulk data given a URL:

```
data = client.retrieve_bulkdata('https://mydicomwebserver.com/studies/...')
```

3.1.14 WADO-RS RetrieveRenderedTransaction

Retrieve a single-frame image instance rendered as a PNG compressed image:

```
frames = client.retrieve_instance_rendered(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    media_types=('image/png', )
)
```

Retrieve a single frame of a multi-frame image instance rendered as a high-quality JPEG compressed image that includes an ICC profile:

```
frames = client.retrieve_instance_frames_rendered(
    study_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639
↪',
    series_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034
↪',
    sop_instance_uid='1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534',
    frame_numbers=[1],
    media_types=('image/jpeg', ),
    params={'quality': 95, 'iccprofile': 'yes'}
)
```

When frames are retrieved in image format, they can be converted into a *NumPy* array using the *PIL* module:

```
from io import BytesIO

import numpy as np
from PIL import Image

image = Image.open(BytesIO(frames[0]))
array = np.array(image)
```

Warning: Retrieving images using lossy compression methods may lead to image recompression artifacts if the images have been stored lossy compressed.

3.1.15 Loading JSON Data To pydicom

Load metadata from JSON format into a `pydicom.dataset.Dataset` object. A common use for this is translating metadata received from a `RetrieveMetadata` or a `SearchFor`-style request:

```
from pydicom.dataset import Dataset

metadata = client.retrieve_study_metadata('1.2.826.0.1.3680043.8.1055.1.
↪20111103111148288.98361414.79379639')
metadata_datasets = [Dataset.from_json(ds) for ds in metadata]
```

Note that the metadata may include references to BulkData elements. By default, BulkData elements will not be handled and the values not be automatically retrieved. To handle BulkData elements and retrieve their values, one has to provide a `bulk_data_uri_handler` callable to the pydicom.dataset.Dataset.from_json() method.

3.2 Command Line Interface (CLI)

Search for studies:

```
dicomweb_client --url https://dicomcloud.azurewebsites.net/qidors search studies
```

Retrieve metadata for all instances of a given study:

```
dicomweb_client --url https://dicomcloud.azurewebsites.net/wadors \
    retrieve studies \
    --study 1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639 \
    metadata
```

The output can be *dicomized* for human interpretation:

```
dicomweb_client --url https://dicomcloud.azurewebsites.net/wadors \
    retrieve studies \
    --study 1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639 \
    metadata \
    --dicomize
```

Retrieve the full Part 3.10 files for all instances of a given study:

```
dicomweb_client --url https://dicomcloud.azurewebsites.net/wadors \
    retrieve studies \
    --study 1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639 \
    full
```

Retrieve a single frame of a given instances as JPEG compressed image:

```
dicomweb_client --url https://dicomcloud.azurewebsites.net/wadors \
    retrieve instances \
    --study 1.2.826.0.1.3680043.8.1055.1.20111103111148288.98361414.79379639 \
    --series 1.2.826.0.1.3680043.8.1055.1.20111103111208937.49685336.24517034 \
    --instance 1.2.826.0.1.3680043.8.1055.1.20111103111208937.40440871.13152534 \
    frames \
    --numbers 1 \
    --media-type image/jpeg
```

Store instances to a Google DICOMweb store:

```
dicomweb_client --url https://healthcare.googleapis.com/v1beta1/projects/MYPROJECT/
↪locations/us-central1/datasets/MYDATASET/dicomStores/MYDICOMSTORE/dicomWeb \
    --token $(gcloud auth print-access-token) \
    store instances \
    dicomfiles/*
```

CHAPTER

FOUR

DEVELOPER GUIDE

Source code is available at Github and can be cloned via git:

```
git clone https://github.com/ImagingDataCommons/dicomweb-client ~/dicomweb-client
```

The `dicomweb_client` package can be installed in *develop* mode for local development:

```
pip install -e ~/dicomweb-client
```

4.1 Pull requests

Don’t commit code changes to the `master` branch. New features should be implemented in a separate branch called `feature/*` and bug fixes should be applied in separate branch called `bugfix/*`.

Before creating a pull request on Github, read the coding style guideline, run the tests and check PEP8 compliance.

4.2 Coding style

Code must comply with PEP 8. The flake8 package is used to enforce compliance.

The project uses numpydoc for documenting code according to PEP 257 docstring conventions. Further information and examples for the NumPy style can be found at the NumPy Github repository and the website of the Napoleon sphinx extension.

All API classes, functions and modules must be documented (including “private” functions and methods). Each docstring must describe input parameters and return values. Types must be specified using type hints as specified by PEP 484 (see typing module) in both the function definition as well as the docstring.

4.3 Running tests

The project uses pytest to write and runs unit tests. Tests should be placed in a separate `tests` folder within the package root folder. Files containing actual test code should follow the pattern `test_*.py`.

Install requirements:

```
pip install -r ~/dicomweb-client/requirements_test.txt
```

Run tests (including checks for PEP8 compliance):

```
cd ~/dicomweb-client
pytest --flake8
```

4.4 Building documentation

Install requirements:

```
pip install -r ~/dicomweb-client/requirements_docs.txt
```

Build documentation in *HTML* format:

```
cd ~/dicomweb-client
sphinx-build -b html docs/ docs/build/
```

The built `index.html` file will be located in `docs/build`.

CHAPTER

FIVE

CONFORMANCE STATEMENT

5.1 QIDO-RS

Method	Resource	Implemented
GET	SearchForStudies	Y
GET	SearchForSeries	Y
GET	SearchForInstances	Y

5.2 WADO-RS

Method	Resource	Implemented
GET	RetrieveStudy	Y
GET	RetrieveSeries	Y
GET	RetrieveInstance	Y
GET	RetrieveMetadata	Y*
GET	RetrieveBulkdata	Y
GET	RetrieveFrames	Y
GET	RetrieveRenderedTransaction	Y

- *Metadata* resource representations are requested in JSON format according to the DICOM JSON model using `application/dicom+json` media type. Retrieval of metadata in XML form using `application/dicom+xml` is not supported.

5.3 STOW-RS

Method	Resource	Implemented
POST	StoreInstances	Y

CHAPTER

SIX

LICENSE

DICOMweb Client is free and open source software licensed under the permissive MIT license.

CHAPTER

SEVEN

API DOCUMENTATION

7.1 dicomweb_client package

7.1.1 dicomweb_client.api module

Application Programming Interface (API)

class dicomweb_client.api.**DICOMClient**(**args*, ***kwargs*)

Bases: `Protocol`

Protocol for DICOM clients based on DICOMweb interface.

base_url: str

delete_instance(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*)

Delete specified instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID

Returns

HTTP response object returned.

Return type

requests.models.Response

Note: The Delete Instance resource is not part of the DICOM standard and may not be supported by all origin servers.

delete_series(*study_instance_uid*, *series_instance_uid*)

Delete all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID

Returns

HTTP response object returned.

Return type
requests.models.Response

Note: The Delete Series resource is not part of the DICOM standard and may not be supported by all origin servers.

`delete_study`(*study_instance_uid*)

Delete all instances of a study.

Parameters
`study_instance_uid` (*str*) – Study Instance UID

Returns
HTTP response object returned.

Return type
requests.models.Response

Note: The Delete Study resource is not part of the DICOM standard and may not be supported by all origin servers.

`delete_url_prefix: Optional[str] = None`

`iter_bulkdata`(*url*, *media_types=None*, *byte_range=None*)

Iterate over bulk data items at a given location.

Parameters

- `url` (*str*) – Location of the bulk data
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- `byte_range` (*Union[Tuple[int, int], None], optional*) – Start and end of byte range

Returns
Bulk data items

Return type
Iterator[bytes]

`iter_instance_frames`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*)

Iterate over frames of an image instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID
- `frame_numbers` (*Sequence[int]*) – One-based positional indices of the frames within the instance

- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Pixel data for each frame

Return type
Iterator[bytes]

iter_series(*study_instance_uid*, *series_instance_uid*, *media_types=None*)

Iterate over all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
Iterator[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

iter_study(*study_instance_uid*, *media_types=None*)

Iterate over all instances of a study.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
Iterator[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`protocol: Optional[str]`

`qido_url_prefix: Optional[str] = None`

`retrieve_bulkdata`(*url*, *media_types=None*, *byte_range=None*)

Retrieve bulk data at a given location.

Parameters

- `url` (*str*) – Location of the bulk data
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- `byte_range` (*Union[Tuple[int, int], None], optional*) – Start and end of byte range

Returns
Bulk data items

Return type
List[bytes]

`retrieve_instance`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *media_types=None*)

Retrieve an individual instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instance

Return type
pydicom.dataset.Dataset

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`retrieve_instance_frames`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*)

Retrieve one or more frames of an image instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID

- `frame_numbers` (*Sequence[int]*) – One-based positional indices of the frames within the instance
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Pixel data for each frame

Return type
List[bytes]

`retrieve_instance_frames_rendered`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*, *params=None*)

Retrieve one or more server-side rendered frames of an instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID
- `frame_numbers` (*Sequence[int]*) – One-based positional index of the frame within the instance
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media type (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`)
- `params` (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"` media type

Returns
Rendered representation of frames

Return type
bytes

Note: Not all media types are compatible with all SOP classes.

`retrieve_instance_metadata`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*)

Retrieve metadata of an individual instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID

Returns
Metadata of instance in DICOM JSON format

Return type
Dict[str, dict]

`retrieve_instance_rendered`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *media_types=None*, *params=None*)

Retrieve an individual, server-side rendered instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`, `"video/gif"`, `"video/mp4"`, `"video/h265"`, `"text/html"`, `"text/plain"`, `"text/xml"`, `"text/rtf"`, `"application/pdf"`)
- **params** (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"`

Returns

Rendered representation of instance

Return type

bytes

`retrieve_series`(*study_instance_uid*, *series_instance_uid*, *media_types=None*)

Retrieve all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns

Instances

Return type

List[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`retrieve_series_metadata`(*study_instance_uid*, *series_instance_uid*)

Retrieve metadata for all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID

Returns

Metadata of instances in DICOM JSON format

Return type
Dict[str, dict]

`retrieve_series_rendered`(*study_instance_uid*, *series_instance_uid*, *media_types=None*, *params=None*)

Retrieve rendered representation of a series.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`, `"video/gif"`, `"video/mp4"`, `"video/h265"`, `"text/html"`, `"text/plain"`, `"text/xml"`, `"text/rtf"`, `"application/pdf"`)
- `params` (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"`

Returns
Rendered representation of series

Return type
bytes

`retrieve_study`(*study_instance_uid*, *media_types=None*)

Retrieve all instances of a study.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
List[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`retrieve_study_metadata`(*study_instance_uid*)

Retrieve metadata of all instances of a study.

Parameters
`study_instance_uid` (*str*) – Study Instance UID

Returns
Metadata of instances in DICOM JSON format

Return type
List[Dict[str, dict]]

scheme: str

search_for_instances(*study_instance_uid=None*, *series_instance_uid=None*, *fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for instances.

Parameters

- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID
- **series_instance_uid** (*Union[str, None], optional*) – Series Instance UID
- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Union[list, tuple, set], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[Dict[str, Union[str, int, float]], None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included (this may repeatedly query the server for remaining results)

Returns

Instance representations (see Instance Result Attributes)

Return type

List[Dict[str, dict]]

Note: The server may only return a subset of search results. In this case, a warning will notify the client that there are remaining results. Remaining results can be requested via repeated calls using the *offset* parameter.

search_for_series(*study_instance_uid=None*, *fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for series.

Parameters

- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID
- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Union[list, tuple, set], None], optional*) – Names of fields (attributes) that should be included in results

- **search_filters** (*Union[Dict[str, Union[str, int, float]], None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included (this may repeatedly query the server for remaining results)

Returns

Series representations (see Series Result Attributes)

Return type

List[Dict[str, dict]]

Note: The server may only return a subset of search results. In this case, a warning will notify the client that there are remaining results. Remaining results can be requested via repeated calls using the *offset* parameter.

search_for_studies(*fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for studies.

Parameters

- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Sequence[str], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[dict, None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included (this may repeatedly query the server for remaining results)

Returns

Study representations (see Study Result Attributes)

Return type

List[Dict[str, dict]]

Note: The server may only return a subset of search results. In this case, a warning will notify the client that there are remaining results. Remaining results can be requested via repeated calls using the *offset* parameter.

store_instances(*datasets*, *study_instance_uid=None*)

Store instances.

Parameters

- **datasets** (*Sequence[pydicom.dataset.Dataset]*) – Instances that should be stored
- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID

Returns
Information about status of stored instances

Return type
pydicom.dataset.Dataset

`stow_url_prefix: Optional[str] = None`

`url_prefix: str`

`wado_url_prefix: Optional[str] = None`

`class dicomweb_client.api.DICOMfileClient`(*url*, *update_db=False*, *recreate_db=False*, *in_memory=False*, *db_dir=None*, *readonly=False*)

Bases: `object`

Client for managing DICOM Part10 files in a DICOMweb-like manner.

Facilitates serverless access to data stored locally on a file system as DICOM Part10 files.

Note: The class internally uses an in-memory database, which is persisted on disk to facilitate faster subsequent data access. However, the implementation details of the database and the structure of any database files stored on the file system may change at any time and should not be relied on.

Note: This is **not** an implementation of the DICOM File Service and does not depend on the presence of `DICOMDIR` files.

`base_url`

Unique resource locator of the DICOMweb service

Type
str

`scheme`

Name of the scheme (`file`)

Type
str

`protocol`

Name of the protocol (`None`)

Type
str

`url_prefix`

URL path prefix for DICOMweb services (part of *base_url*)

Type
str

`qido_url_prefix`

URL path prefix for QIDO-RS (not part of *base_url*)

Type
Union[str, None]

wado_url_prefix

URL path prefix for WADO-RS (not part of *base_url*)

Type

Union[str, None]

stow_url_prefix

URL path prefix for STOW-RS (not part of *base_url*)

Type

Union[str, None]

delete_url_prefix

URL path prefix for DELETE (not part of *base_url*)

Type

Union[str, None]

Instantiate client.

Parameters

- **url** (*str*) – Unique resource locator of directory where data is stored (must have `file` scheme)
- **update_db** (*bool, optional*) – Whether the database should be updated (default: `False`). If `True`, the client will search *base_dir* recursively for new DICOM Part10 files and create database entries for each file. The client will further delete any database entries for files that no longer exist on the file system.
- **recreate_db** (*bool, optional*) – Whether the database should be recreated (default: `False`). If `True`, the client will search *base_dir* recursively for DICOM Part10 files and create database entries for each file.
- **in_memory** (*bool, optional*) – Whether the database should only be stored in memory (default: `False`).
- **db_dir** (*Union[pathlib.Path, str, None], optional*) – Path to directory where database files should be stored (defaults to *base_dir*)
- **readonly** (*bool, optional*) – Whether data should be considered read-only. Attempts to store or delete data will be denied.

delete_instance(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*)

Delete specified instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID

Return type

`None`

delete_series(*study_instance_uid*, *series_instance_uid*)

Delete all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID

- **series_instance_uid** (*str*) – Series Instance UID

Return type
None

`delete_study`(*study_instance_uid*)

Delete all instances of a study.

Parameters
study_instance_uid (*str*) – Study Instance UID

Return type
None

`iter_bulkdata`(*url*, *media_types=None*, *byte_range=None*)

Iterate over bulk data items at a given location.

Parameters

- **url** (*str*) – Location of the bulk data
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- **byte_range** (*Union[Tuple[int, int], None], optional*) – Start and end of byte range

Returns
Bulk data items

Return type
Iterator[bytes]

Raises
IOError – When requested resource is not found at *url*

`iter_instance_frames`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*)

Iterate over frames of an image instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **frame_numbers** (*List[int]*) – Frame numbers
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Frames

Return type
Iterator[bytes]

`iter_series`(*study_instance_uid*, *series_instance_uid*, *media_types=None*)

Iterate over all instances of a series.

Parameters

- **`study_instance_uid`** (*`str`*) – Study Instance UID
- **`series_instance_uid`** (*`str`*) – Series Instance UID
- **`media_types`** (*`Union[Tuple[Union[str, Tuple[str, str]], ...], None]`*, *`optional`*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Instances

Return type
Iterator[pydicom.dataset.Dataset]

`iter_study`(*study_instance_uid*, *media_types=None*)

Iterate over all instances of a study.

Parameters

- **`study_instance_uid`** (*`str`*) – Study Instance UID
- **`media_types`** (*`Union[Tuple[Union[str, Tuple[str, str]], ...], None]`*, *`optional`*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Instances

Return type
Iterator[pydicom.dataset.Dataset]

`static` **`lookup_keyword`**(*tag*)

Look up the keyword of a DICOM attribute.

Parameters
`tag` (*`Union[str, int, Tuple[int, int], pydicom.tag.BaseTag]`*) – Attribute tag (e.g. `"00080018"`)

Returns
Attribute keyword (e.g. `"SOPInstanceUID"`)

Return type
str

`static` **`lookup_tag`**(*keyword*)

Look up the tag of a DICOM attribute.

Parameters
`keyword` (*`str`*) – Attribute keyword (e.g. `"SOPInstanceUID"`)

Returns
Attribute tag as HEX string (e.g. `"00080018"`)

Return type
str

`retrieve_bulkdata`(*url*, *media_types=None*, *byte_range=None*)

Retrieve bulk data at a given location.

Parameters

- **`url`** (*`str`*) – Location of the bulk data

- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- **byte_range** (*Union[Tuple[int, int], None], optional*) – Start and end of byte range

Returns
Bulk data items

Return type
Iterator[bytes]

Raises
`IOError` – When requested resource is not found at *url*

`retrieve_instance`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *media_types=None*)

Retrieve metadata of a single instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Instance

Return type
pydicom.dataset.Dataset

`retrieve_instance_frames`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*)

Retrieve one or more frames of an image instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **frame_numbers** (*List[int]*) – Frame numbers
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Frames

Return type
List[bytes]

`retrieve_instance_frames_rendered`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*, *params=None*)

Retrieve server-side rendered frames of an image instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **frame_numbers** (*List[int]*) – Frame numbers
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- **params** (*Union[Dict[str, str], None], optional*) – Additional query parameters

Returns
Rendered representation of frames

Return type
bytes

retrieve_instance_metadata(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*)

Retrieve metadata of a single instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID

Returns
Metadata of instance

Return type
Dict[str, Any]

retrieve_instance_rendered(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *media_types=None*, *params=None*)

Retrieve an individual, server-side rendered instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`, `"video/gif"`, `"video/mp4"`, `"video/h265"`, `"text/html"`, `"text/plain"`, `"text/xml"`, `"text/rtf"`, `"application/pdf"`)
- **params** (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"`

Returns
Rendered representation of instance

Return type
bytes

Note: Only rendering of single-frame image instances is currently supported.

`retrieve_series`(*study_instance_uid*, *series_instance_uid*, *media_types=None*)

Retrieve all instances of a series.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns

Instances

Return type

Sequence[pydicom.dataset.Dataset]

`retrieve_series_metadata`(*study_instance_uid*, *series_instance_uid*)

Retrieve metadata of instances in a series.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID

Returns

Metadata of each instance in series

Return type

List[Dict[str, Any]]

`retrieve_series_rendered`(*study_instance_uid*, *series_instance_uid*, *media_types=None*, *params=None*)

Retrieve rendered representation of a series.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`, `"video/gif"`, `"video/mp4"`, `"video/h265"`, `"text/html"`, `"text/plain"`, `"text/xml"`, `"text/rtf"`, `"application/pdf"`)
- `params` (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"`

Returns

Rendered representation of series

Return type

bytes

`retrieve_study`(*study_instance_uid*, *media_types=None*)

Retrieve all instances of a study.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Instances

Return type
Sequence[pydicom.dataset.Dataset]

retrieve_study_metadata(*study_instance_uid*)

Retrieve metadata of instances in a study.

Parameters
study_instance_uid (*str*) – Study Instance UID

Returns
Metadata of each instance in study

Return type
List[Dict[str, Any]]

search_for_instances(*study_instance_uid=None*, *series_instance_uid=None*, *fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for instances.

Parameters

- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID
- **series_instance_uid** (*Union[str, None], optional*) – Series Instance UID
- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Sequence[str], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[dict, None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included

Returns

Instances (see Instance Result Attributes)

Return type
List[Dict[str, dict]]

Note: No additional *fields* are currently supported.

`search_for_series`(*study_instance_uid=None*, *fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for series.

Parameters

- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID
- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Sequence[str], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[dict, None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included

Returns

Series (see Series Result Attributes)

Return type

List[Dict[str, dict]]

`search_for_studies`(*fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for studies.

Parameters

- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Sequence[str], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[dict, None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included

Returns

Studies (see Study Result Attributes)

Return type

List[Dict[str, dict]]

Note: No additional *fields* are currently supported.

`store_instances`(*datasets*, *study_instance_uid=None*)

Store instances.

Parameters

- **datasets** (*Sequence[pydicom.dataset.Dataset]*) – Instances that should be stored
- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID

Returns

Information about status of stored instances

Return type

pydicom.dataset.Dataset

`class dicomweb_client.api.DICOMwebClient`(*url*, *session=None*, *qido_url_prefix=None*, *wado_url_prefix=None*, *stow_url_prefix=None*, *delete_url_prefix=None*, *proxies=None*, *headers=None*, *callback=None*, *chunk_size=1000000*)

Bases: `object`

Class for connecting to and interacting with a DICOMweb RESTful service.

`base_url`

Unique resource locator of the DICOMweb service

Type

str

`scheme`

Name of the scheme, e.g. `"https"`

Type

str

`protocol`

Name of the protocol, e.g. `"https"`

Type

str

`host`

IP address or DNS name of the machine that hosts the server

Type

str

`port`

Number of the port to which the server listens

Type

int

`url_prefix`

URL path prefix for DICOMweb services (part of *base_url*)

Type

str

`qido_url_prefix`

URL path prefix for QIDO-RS (not part of *base_url*)

Type
Union[str, None]

`wado_url_prefix`

URL path prefix for WADO-RS (not part of *base_url*)

Type
Union[str, None]

`stow_url_prefix`

URL path prefix for STOW-RS (not part of *base_url*)

Type
Union[str, None]

`delete_url_prefix`

URL path prefix for DELETE (not part of *base_url*)

Type
Union[str, None]

Instatiate client.

Parameters

- **url** (*str*) – Unique resource locator of the DICOMweb service consisting of protocol, hostname (IP address or DNS name) of the machine that hosts the service and optionally port number and path prefix
- **session** (*Union[requests.Session, None], optional*) – Session required to make connections to the DICOMweb service (see `dicomweb_client.session_utils` module to create a valid session if necessary)
- **qido_url_prefix** (*Union[str, None], optional*) – URL path prefix for QIDO RESTful services
- **wado_url_prefix** (*Union[str, None], optional*) – URL path prefix for WADO RESTful services
- **stow_url_prefix** (*Union[str, None], optional*) – URL path prefix for STOW RESTful services
- **delete_url_prefix** (*Union[str, None], optional*) – URL path prefix for DELETE RESTful services
- **proxies** (*Union[Dict[str, str], None], optional*) – Mapping of protocol or protocol + host to the URL of a proxy server
- **headers** (*Union[Dict[str, str], None], optional*) – Custom headers that should be included in request messages, e.g., authentication tokens
- **callback** (*Union[Callable[[requests.Response, ...], requests.Response], None], optional*) – Callback function to manipulate responses generated from requests (see requests event hooks)
- **chunk_size** (*int, optional*) – Maximum number of bytes that should be transferred per data chunk when streaming data from the server using chunked transfer encoding (used by `iter_*()` methods as well as the `store_instances()` method); defaults to `10**6` bytes (1MB)

> **Warning:** Modifies the passed *session* (in particular header fields), so be careful when reusing the session outside the scope of an instance.

> **Warning:** Choose the value of *chunk_size* carefully. A small value may cause significant network communication and message parsing overhead.

`delete_instance`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*)

Delete specified instance.

Parameters

- `study_instance_uid` (`str`) – Study Instance UID
- `series_instance_uid` (`str`) – Series Instance UID
- `sop_instance_uid` (`str`) – SOP Instance UID

Note: The Delete Instance resource is not part of the DICOM standard and may not be supported by all origin servers.

> **Warning:** This method performs a DELETE and should be used with caution.

Return type

`None`

`delete_series`(*study_instance_uid*, *series_instance_uid*)

Delete all instances of a series.

Parameters

- `study_instance_uid` (`str`) – Study Instance UID
- `series_instance_uid` (`str`) – Series Instance UID

Note: The Delete Series resource is not part of the DICOM standard and may not be supported by all origin servers.

> **Warning:** This method performs a DELETE and should be used with caution.

Return type

`None`

`delete_study`(*study_instance_uid*)

Delete all instances of a study.

Parameters

`study_instance_uid` (`str`) – Study Instance UID

Note: The Delete Study resource is not part of the DICOM standard and may not be supported by all origin servers.

Warning: This method performs a DELETE and should be used with caution.

Return type
`None`

`iter_bulkdata`(*url*, *media_types=None*, *byte_range=None*)

Iterate over bulk data items at a given location.

Parameters

- **url** (*str*) – Location of the bulk data
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- **byte_range** (*Union[Tuple[int, int], None], optional*) – Start and end of byte range

Returns
Bulk data items

Return type
Iterator[bytes]

Note: Data is streamed from the DICOMweb server.

`iter_instance_frames`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*)

Iterate over frames of an image instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **frame_numbers** (*Sequence[int]*) – One-based positional indices of the frames within the instance
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Pixel data for each frame

Return type
Iterator[bytes]

Note: Data is streamed from the DICOMweb server.

`iter_series`(*study_instance_uid*, *series_instance_uid*, *media_types=None*)

Iterate over all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
Iterator[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

Note: Data is streamed from the DICOMweb server.

`iter_study`(*study_instance_uid*, *media_types=None*)

Iterate over all instances of a study.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
Iterator[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

Note: Data is streamed from the DICOMweb server.

`static lookup_keyword`(*tag*)

Look up the keyword of a DICOM attribute.

Parameters
`tag` (*Union[str, int, Tuple[str, str], pydicom.tag.BaseTag]*) – Attribute tag (e.g. `"00080018"`)

Returns
Attribute keyword (e.g. `"SOPInstanceUID"`)

Return type
str

`static lookup_tag`(*keyword*)

Look up the tag of a DICOM attribute.

Parameters
`keyword` (*str*) – Attribute keyword (e.g. `"SOPInstanceUID"`)

Returns
Attribute tag as HEX string (e.g. `"00080018"`)

Return type
str

`retrieve_bulkdata`(*url*, *media_types=None*, *byte_range=None*)

Retrieve bulk data at a given location.

Parameters

- `url` (*str*) – Location of the bulk data
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes
- `byte_range` (*Union[Tuple[int, int], None], optional*) – Start and end of byte range

Returns
Bulk data items

Return type
Iterator[bytes]

`retrieve_instance`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *media_types=None*)

Retrieve an individual instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instance

Return type
pydicom.dataset.Dataset

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`retrieve_instance_frames`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*)

Retrieve one or more frames of an image instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **frame_numbers** (*Sequence[int]*) – One-based positional indices of the frames within the instance
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the corresponding transfer syntaxes

Returns
Pixel data for each frame

Return type
List[bytes]

`retrieve_instance_frames_rendered`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *frame_numbers*, *media_types=None*, *params=None*)

Retrieve one or more server-side rendered frames of an instance.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **sop_instance_uid** (*str*) – SOP Instance UID
- **frame_numbers** (*Sequence[int]*) – One-based positional index of the frame within the instance
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media type (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`)
- **params** (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"` media type

Returns
Rendered representation of frames

Return type
bytes

Note: Not all media types are compatible with all SOP classes.

`retrieve_instance_metadata`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*)

Retrieve metadata of an individual instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID

Returns
Metadata of instance in DICOM JSON format

Return type
Dict[str, dict]

`retrieve_instance_rendered`(*study_instance_uid*, *series_instance_uid*, *sop_instance_uid*, *media_types=None*, *params=None*)

Retrieve an individual, server-side rendered instance.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `sop_instance_uid` (*str*) – SOP Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`, `"video/gif"`, `"video/mp4"`, `"video/h265"`, `"text/html"`, `"text/plain"`, `"text/xml"`, `"text/rtf"`, `"application/pdf"`)
- `params` (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"`

Returns
Rendered representation of instance

Return type
bytes

`retrieve_series`(*study_instance_uid*, *series_instance_uid*, *media_types=None*)

Retrieve all instances of a series.

Parameters

- `study_instance_uid` (*str*) – Study Instance UID
- `series_instance_uid` (*str*) – Series Instance UID
- `media_types` (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
List[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`retrieve_series_metadata`(*study_instance_uid*, *series_instance_uid*)

Retrieve metadata for all instances of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID

Returns
Metadata of instances in DICOM JSON format

Return type
Dict[str, dict]

`retrieve_series_rendered`(*study_instance_uid*, *series_instance_uid*, *media_types=None*, *params=None*)

Retrieve rendered representation of a series.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **series_instance_uid** (*str*) – Series Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types (choices: `"image/jpeg"`, `"image/jp2"`, `"image/gif"`, `"image/png"`, `"video/gif"`, `"video/mp4"`, `"video/h265"`, `"text/html"`, `"text/plain"`, `"text/xml"`, `"text/rtf"`, `"application/pdf"`)
- **params** (*Union[Dict[str, Any], None], optional*) – Additional parameters relevant for given *media_type*, e.g., `{"quality": 95}` for `"image/jpeg"`

Returns
Rendered representation of series

Return type
bytes

`retrieve_study`(*study_instance_uid*, *media_types=None*)

Retrieve all instances of a study.

Parameters

- **study_instance_uid** (*str*) – Study Instance UID
- **media_types** (*Union[Tuple[Union[str, Tuple[str, str]], ...], None], optional*) – Acceptable media types and optionally the UIDs of the acceptable transfer syntaxes

Returns
Instances

Return type
List[pydicom.dataset.Dataset]

Note: Instances are by default retrieved using Implicit VR Little Endian transfer syntax (Transfer Syntax UID `"1.2.840.10008.1.2"`). This means that Pixel Data of Image instances will be retrieved uncompressed. To retrieve instances in any available transfer syntax (typically the one in which instances were originally stored), specify acceptable transfer syntaxes using the wildcard `("application/dicom", "*")`.

`retrieve_study_metadata`(*study_instance_uid*)

Retrieve metadata of all instances of a study.

Parameters
`study_instance_uid` (*`str`*) – Study Instance UID

Returns
Metadata of instances in DICOM JSON format

Return type
List[Dict[str, dict]]

`search_for_instances`(*study_instance_uid=None*, *series_instance_uid=None*, *fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for instances.

Parameters

- **`study_instance_uid`** (*`Union[str, None], optional`*) – Study Instance UID
- **`series_instance_uid`** (*`Union[str, None], optional`*) – Series Instance UID
- **`fuzzymatching`** (*`Union[bool, None], optional`*) – Whether fuzzy semantic matching should be performed
- **`limit`** (*`Union[int, None], optional`*) – Maximum number of results that should be returned
- **`offset`** (*`Union[int, None], optional`*) – Number of results that should be skipped
- **`fields`** (*`Union[Union[list, tuple, set], None], optional`*) – Names of fields (attributes) that should be included in results
- **`search_filters`** (*`Union[Dict[str, Union[str, int, float]], None], optional`*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **`get_remaining`** (*`bool, optional`*) – Whether remaining results should be included (this may repeatedly query the server for remaining results)

Returns

Instance representations (see Instance Result Attributes)

Return type
List[Dict[str, dict]]

Note: The server may only return a subset of search results. In this case, a warning will notify the client that there are remaining results. Remaining results can be requested via repeated calls using the *offset* parameter.

`search_for_series`(*study_instance_uid=None*, *fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for series.

Parameters

- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID
- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Union[list, tuple, set], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[Dict[str, Union[str, int, float]], None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match
- **get_remaining** (*bool, optional*) – Whether remaining results should be included (this may repeatedly query the server for remaining results)

Returns

Series representations (see Series Result Attributes)

Return type

List[Dict[str, dict]]

Note: The server may only return a subset of search results. In this case, a warning will notify the client that there are remaining results. Remaining results can be requested via repeated calls using the *offset* parameter.

`search_for_studies`(*fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*, *get_remaining=False*)

Search for studies.

Parameters

- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Sequence[str], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[dict, None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match

- **get_remaining** (*bool, optional*) – Whether remaining results should be included (this may repeatedly query the server for remaining results)

Returns

Study representations (see Study Result Attributes)

Return type

List[Dict[str, dict]]

Note: The server may only return a subset of search results. In this case, a warning will notify the client that there are remaining results. Remaining results can be requested via repeated calls using the *offset* parameter.

`set_chunk_size`(*chunk_size*)

Set value of *chunk_size* attribute.

Parameters

chunk_size (*int*) – Maximum number of bytes that should be transferred per data chunk when streaming data from the server using chunked transfer encoding (used by `iter_*()` methods as well as the `store_instances()` method)

Return type

`None`

`set_http_retry_params`(*retry=True*, *max_attempts=5*, *wait_exponential_multiplier=1000*, *retriable_error_codes=(<HTTPStatus.TOO_MANY_REQUESTS: 429>, <HTTPStatus.REQUEST_TIMEOUT: 408>, <HTTPStatus.SERVICE_UNAVAILABLE: 503>, <HTTPStatus.GATEWAY_TIMEOUT: 504>)*)

Set parameters for HTTP retrying logic.

These parameters determine whether and how individual HTTP requests will be retried in case the origin server responds with an error code defined in *retriable_error_codes*. The retrying method uses exponential back off using the multiplier *wait_exponential_multiplier* for a max attempts defined by *max_attempts*.

Parameters

- **retry** (*bool, optional*) – Whether HTTP retrying should be performed, if it is set to `False`, the rest of the parameters are ignored.
- **max_attempts** (*int, optional*) – The maximum number of request attempts.
- **wait_exponential_multiplier** (*float, optional*) – Exponential multiplier applied to delay between attempts in ms.
- **retriable_error_codes** (*tuple, optional*) – Tuple of HTTP error codes to retry if raised.

Return type

`None`

`store_instances`(*datasets*, *study_instance_uid=None*)

Store instances.

Parameters

- **datasets** (*Sequence[pydicom.dataset.Dataset]*) – Instances that should be stored
- **study_instance_uid** (*Union[str, None], optional*) – Study Instance UID

Returns
Information about status of stored instances

Return type
pydicom.dataset.Dataset

7.1.2 dicomweb_client.cli module

Command Line Interface (CLI)

`dicomweb_client.cli.`**`main`**(*args*)

Main entry point for the `dicomweb_client` command line program.

dicomweb_client

Client for DICOMweb RESTful services.

```
usage: dicomweb_client [-h] [-v] [-u NAME] [-p PASSWORD] [--ca CERT-FILE]
                       [--cert CERT-FILE] [--bearer-token TOKEN] [--url URL]
                       [--chunk-size NUM]
                       {search,retrieve,store} ...
```

`-h``, `**`--help`**

show this help message and exit

`-v``, `**`--verbosity`**

logging verbosity that maps to a logging level (default: error, -v: warning, -vv: info, -vvv: debug, -vvvv: debug + traceback); all log messages are written to standard error

`-u` `<name>`, **`--user`** `<name>`

username for authentication with the DICOMweb service

`-p` `<password>`, **`--password`** `<password>`

password for authentication with the DICOMweb service

`--ca` `<cert-file>`

path to a CA bundle file

`--cert` `<cert-file>`

path to a client certificate file in PEM format

`--bearer-token` `<token>`

bearer token for authentication with the DICOMweb service

`--url` `<url>`

uniform resource locator of the DICOMweb service

`--chunk-size` `<num>`

maximum size of a network transfer chunk in bytes

dicomweb_client retrieve

WADO-RS: Web Access to DICOM Objects by RESTful Services.

```
usage: dicomweb_client retrieve [-h] INFORMATION ENTITIES ...
```

-h, --help
  show this help message and exit

dicomweb_client retrieve bulkdata

Retrieve bulk data of a DICOM object from a known location.

```
usage: dicomweb_client retrieve bulkdata [-h]
                                         [--media-type MEDIATYPE [MEDIATYPE ...]]
                                         --uri URI
```

-h, --help
  show this help message and exit

--media-type <mediatype>
  acceptable media type and the optionally the UID of a corresponding tranfer syntax separted by a whitespace(e.g., “image/jpeg” or “image/jpeg 1.2.840.10008.1.2.4.50”)

--uri <uri>
  unique resource identifier of bulk data element

dicomweb_client retrieve instances

Retrieve data for an individual DICOM instance.

```
usage: dicomweb_client retrieve instances [-h] --study UID --series UID
                                          --instance UID
                                          {metadata,full,frames} ...
```

-h, --help
  show this help message and exit

--study <uid>
  unique study identifier (StudyInstanceUID)

--series <uid>
  unique series identifier (SeriesInstanceUID)

--instance <uid>
  unique instance identifier (SOPInstanceUID)

dicomweb_client retrieve instances frames

Retrieve one or more frames of the pixel data element of an invidividual DICOM instance.

```
usage: dicomweb_client retrieve instances frames [-h] [--save]
                                                 [--output-dir PATH]
                                                 [--media-type MEDIATYPE [MEDIATYPE ...]]
                                                 [--numbers NUM [NUM ...]]
                                                 [--show]
```

-h, --help

show this help message and exit

--save

whether downloaded data should be saved

--output-dir <path>

path to directory where downloaded data should be saved

--media-type <mediatype>

acceptable media type and the optionally the UID of a corresponding tranfer syntax separted by a whitespace(e.g., “image/jpeg” or “image/jpeg 1.2.840.10008.1.2.4.50”)

--numbers <num>

frame numbers

--show

display retrieved images

dicomweb_client retrieve instances full

Retrieve a DICOM instance.

```
usage: dicomweb_client retrieve instances full [-h] [--save]
                                               [--output-dir PATH]
                                               [--media-type MEDIATYPE [MEDIATYPE ...]]
```

-h, --help

show this help message and exit

--save

whether downloaded data should be saved

--output-dir <path>

path to directory where downloaded data should be saved

--media-type <mediatype>

acceptable media type and the optionally the UID of a corresponding tranfer syntax separted by a whitespace(e.g., “image/jpeg” or “image/jpeg 1.2.840.10008.1.2.4.50”)

dicomweb_client retrieve instances metadata

Retrieve metadata of an invidividual DICOM instance.

```
usage: dicomweb_client retrieve instances metadata [-h]
                                                   [--prettify | --dicomize]
                                                   [--save]
                                                   [--output-dir PATH]
```

-h, --help
  show this help message and exit

--prettify
  pretty print JSON response message

--dicomize
  convert JSON response message to DICOM data set

--save
  whether downloaded data should be saved

--output-dir <path>
  path to directory where downloaded data should be saved

dicomweb_client retrieve series

Retrieve data for all DICOM instances of a given DICOM series.

```
usage: dicomweb_client retrieve series [-h] --study UID --series UID
                                       {metadata,full} ...
```

-h, --help
  show this help message and exit

--study <uid>
  unique study identifier (StudyInstanceUID)

--series <uid>
  unique series identifier (SeriesInstanceUID)

dicomweb_client retrieve series full

Retrieve DICOM instances of a given DICOM series.

```
usage: dicomweb_client retrieve series full [-h] [--save] [--output-dir PATH]
                                            [--media-type MEDIATYPE [MEDIATYPE ...]]
```

-h, --help
  show this help message and exit

--save
  whether downloaded data should be saved

--output-dir <path>

path to directory where downloaded data should be saved

--media-type <mediatype>

acceptable media type and the optionally the UID of a corresponding tranfer syntax separted by a whitespace(e.g., “image/jpeg” or “image/jpeg 1.2.840.10008.1.2.4.50”)

dicomweb_client retrieve series metadata

Retrieve metadata of DICOM instances of a given DICOM series.

```
usage: dicomweb_client retrieve series metadata [-h] [--prettify | --dicomize]
                                                [--save] [--output-dir PATH]
```

-h, --help

show this help message and exit

--prettify

pretty print JSON response message

--dicomize

convert JSON response message to DICOM data set

--save

whether downloaded data should be saved

--output-dir <path>

path to directory where downloaded data should be saved

dicomweb_client retrieve studies

Retrieve data for all DICOM instances of a given DICOM study.

```
usage: dicomweb_client retrieve studies [-h] --study UID {metadata,full} ...
```

-h, --help

show this help message and exit

--study <uid>

unique study identifier (StudyInstanceUID)

dicomweb_client retrieve studies full

Retrieve DICOM instances of a given DICOM study.

```
usage: dicomweb_client retrieve studies full [-h] [--save] [--output-dir PATH]
                                             [--media-type MEDIATYPE [MEDIATYPE ...]]
```

-h, --help

show this help message and exit

--save
: whether downloaded data should be saved

--output-dir <path>
: path to directory where downloaded data should be saved

--media-type <mediatype>
: acceptable media type and the optionally the UID of a corresponding tranfer syntax separted by a whitespace(e.g., “image/jpeg” or “image/jpeg 1.2.840.10008.1.2.4.50”)

dicomweb_client retrieve studies metadata

Retrieve metadata of DICOM instances of a given DICOM study.

```
usage: dicomweb_client retrieve studies metadata [-h]
                                                 [--prettify | --dicomize]
                                                 [--save] [--output-dir PATH]
```

-h, --help
: show this help message and exit

--prettify
: pretty print JSON response message

--dicomize
: convert JSON response message to DICOM data set

--save
: whether downloaded data should be saved

--output-dir <path>
: path to directory where downloaded data should be saved

dicomweb_client search

QIDO-RS: Query based on ID for DICOM Objects by RESTful Serices.

```
usage: dicomweb_client search [-h] INFORMATION ENTITIES ...
```

-h, --help
: show this help message and exit

dicomweb_client search instances

Search for DICOM instances.

```
usage: dicomweb_client search instances [-h] [--prettify | --dicomize]
                                        [--filter KEY=VALUE] [--field NAME]
                                        [--limit NUM] [--offset NUM] [--fuzzy]
                                        [--study UID] [--series UID]
```

-h, --help

show this help message and exit

--prettify

pretty print JSON response message

--dicomize

convert JSON response message to DICOM data set

--filter <key=value>

query filter criterion

--field <name>

field that should be included in response

--limit <num>

number of items that should be maximally retrieved

--offset <num>

number of items that should be skipped

--fuzzy

perform fuzzy matching

--study <uid>

unique study identifer (StudyInstanceUID)

--series <uid>

unique series identifier (SeriesInstanceUID)

dicomweb_client search series

Search for DICOM series.

```
usage: dicomweb_client search series [-h] [--filter KEY=VALUE] [--field NAME]
                                     [--limit NUM] [--offset NUM] [--fuzzy]
                                     [--prettify | --dicomize] [--study UID]
```

-h, --help

show this help message and exit

--filter <key=value>

query filter criterion

--field <name>

field that should be included in response

--limit <num>

number of items that should be maximally retrieved

--offset <num>

number of items that should be skipped

--fuzzy

perform fuzzy matching

--prettify
pretty print JSON response message

--dicomize
convert JSON response message to DICOM data set

--study <uid>
unique study identifer (StudyInstanceUID)

dicomweb_client search studies

Search for DICOM studies.

```
usage: dicomweb_client search studies [-h] [--filter KEY=VALUE] [--field NAME]
                                      [--limit NUM] [--offset NUM] [--fuzzy]
                                      [--prettify | --dicomize]
```

-h, --help
show this help message and exit

--filter <key=value>
query filter criterion

--field <name>
field that should be included in response

--limit <num>
number of items that should be maximally retrieved

--offset <num>
number of items that should be skipped

--fuzzy
perform fuzzy matching

--prettify
pretty print JSON response message

--dicomize
convert JSON response message to DICOM data set

dicomweb_client store

STOW-RS: Store Over the Web by RESTful Services.

```
usage: dicomweb_client store [-h] INFORMATION ENTITIES ...
```

-h, --help
show this help message and exit

dicomweb_client store instances

Store DICOM instances.

```
usage: dicomweb_client store instances [-h] [--study UID] PATH [PATH ...]
```

path

paths to DICOM files that should be loaded

-h, --help

show this help message and exit

--study <uid>

unique study identifer (StudyInstanceUID)

7.1.3 dicomweb_client.log module

Utility functions for logging configuration

dicomweb_client.log.**configure_logging**(*verbosity*)

Configures the root logger with a "stderr" stream handler that directs logging messages to standard error (to allow capturing program standard output, e.g. in order to redirect it to a file).

Logging verbosity maps to levels as follows:

```
0 -> no messages
1 -> CRITICAL, ERROR & WARN/WARNING messages
2 -> CRITICAL, ERROR, WARN/WARNING, & INFO messages
3 -> CRITICAL, ERROR, WARN/WARNING, INFO & DEBUG messages
4 -> all messages
```

Parameters

verbosity (*int*) – logging verbosity

Returns

package root logger

Return type

logging.Logger

7.1.4 dicomweb_client.session_utils module

dicomweb_client.session_utils.**add_certs_to_session**(*session*, *ca_bundle=None*, *cert=None*)

Adds CA bundle and certificate to an existing session.

Parameters

- **session** (*requests.Session*) – input session
- **ca_bundle** (*str, optional*) – path to CA bundle file
- **cert** (*str, optional*) – path to client certificate file in Privacy Enhanced Mail (PEM) format

Returns

verified session

Return type
requests.Session

`dicomweb_client.session_utils.`**`create_session`**`()`

Creates an unauthorized session.

Returns
unauthorized session

Return type
requests.Session

`dicomweb_client.session_utils.`**`create_session_from_auth`**`(`*auth*`)`

Creates a session from a gicen AuthBase object.

Parameters
`auth` (*`requests.auth.AuthBase`*) – an implementation of *requests.auth.AuthBase* to be used for authentication with services

Returns
authorized session

Return type
requests.Session

`dicomweb_client.session_utils.`**`create_session_from_gcp_credentials`**`(`*google_credentials=None*`)`

Creates an authorized session for Google Cloud Platform.

Parameters
`google_credentials` (*`Any`*) – Google cloud credentials. (see https://cloud.google.com/docs/authentication/production for more information on Google cloud authentication). If not set, will be initialized to `google.auth.default()`

Returns
Google cloud authorized session

Return type
requests.Session

`dicomweb_client.session_utils.`**`create_session_from_user_pass`**`(`*username*, *password*`)`

Creates a session from a given username and password.

Parameters

- **`username`** (*`str`*) – username for authentication with services
- **`password`** (*`str`*) – password for authentication with services

Returns
authorized session

Return type
requests.Session

7.1.5 dicomweb_client.uri module

Utilities for DICOMweb URI manipulation.

class dicomweb_client.uri.**URI**(*base_url*, *study_instance_uid=None*, *series_instance_uid=None*, *sop_instance_uid=None*, *frames=None*, *suffix=None*, *permissive=False*)

Bases: `object`

Class to represent a fully qualified HTTP[S] URI to a DICOMweb resource.

http://dicom.nema.org/medical/dicom/current/output/html/part18.html

This is an immutable class. Use *URI.update()* to create copies of an instance with updated (new) values for its attributes.

Given an HTTP[S] *base_url*, a valid DICOMweb-compatible URI would be:

- `<base_url>` (no DICOMweb suffix)
- `<base_url>/studies/<study_instance_uid>`
- `<base_url>/studies/<study_instance_uid>/metadata`
- `<base_url>/studies/<study_instance_uid>/rendered`
- `<base_url>/studies/<study_instance_uid>/thumbnail`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/metadata`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/rendered`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/thumbnail`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>/metadata`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>/rendered`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>/thumbnail`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>/frames/<frames>`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>/frames/<frames>/rendered`
- `<base_url>/studies/<study_instance_uid>/series/<series_instance_uid>/instances/<sop_instance_uid>/frames/<frames>/thumbnail`

Instantiates an object.

As per the DICOM Standard, the Study, Series, and Instance UIDs must be a series of numeric components (`0-9`) separated by the period `.` character, with a maximum length of 64 characters. If the `permissive` flag is set to `True`, any alpha-numeric or special characters (except for `/` and `@`) may be used.

Parameters

- **base_url** (*str*) – DICOMweb service HTTP[S] URL. Trailing forward slashes are not permitted.

- **study_instance_uid** (*str, optional*) – DICOM Study Instance UID.
- **series_instance_uid** (*str, optional*) – DICOM Series Instance UID.
- **sop_instance_uid** (*str, optional*) – DICOM SOP Instance UID.
- **frames** (*Sequence[int], optional*) – A non-empty sequence of positive frame numbers in ascending order.
- **suffix** (URISuffix, *optional*) – Suffix attached to the DICOM resource URI. This could refer to a metadata, rendered, or thumbnail resource.
- **permissive** (*bool*) – If `True`, relaxes the DICOM Standard validation for UIDs (see main docstring for details). This option is made available since users may be occasionally forced to work with DICOMs or services that may be in violation of the standard. Unless required, use of this flag is **not** recommended, since non-conformant UIDs may lead to unexpected errors downstream, e.g., rejection by a DICOMweb server, etc.

Raises

ValueError – In the following cases:

- *base_url* has a trailing slash. - *base_url* does not use the HTTP[S] addressing scheme. - *base_url* is incompatible with the DICOMweb standard. - *series_instance_uid* is supplied without *study_instance_uid*. - *sop_instance_uid* is supplied without *study_instance_uid* or *series_instance_uid*. - *frames* is supplied without *study_instance_uid*, *series_instance_uid*, or *sop_instance_uid*. - *frames* is empty. - A frame number in *frames* is not positive. - The frame numbers in *frames* are not in ascending order. - *suffix* is *URISuffix.METADATA* with *frames* or without *study_instance_uid*. - *suffix* is *URISuffix.RENDERED* without *study_instance_uid*. - *suffix* is *URISuffix.THUMBNAIL* without *study_instance_uid*. - Any one of *study_instance_uid*, *series_instance_uid*, or *sop_instance_uid* does not meet the DICOM Standard UID spec in the docstring.

base_uri()

Returns *URI* for the DICOM Service within this object.

Return type

dicomweb_client.uri.URI

property base_url: str

Returns the Base (DICOMweb Service) URL.

Return type

`str`

frame_uri()

Returns *URI* for the DICOM frames within this object.

Return type

dicomweb_client.uri.URI

property frames: Optional[Tuple[int, ...]]

Returns the sequence of frame numbers, if available.

Return type

`typing.Optional[typing.Tuple[int, …]]`

classmethod from_string(*dicomweb_uri*, *uri_type=None*, *permissive=False*)

Parses the string to return the URI.

Any valid DICOMweb compatible HTTP[S] URI is permitted, e.g., `<SERVICE>/studies/<StudyInstanceUID>/series/<SeriesInstanceUID>`.

Parameters

- `dicomweb_uri` (*str*) – An HTTP[S] DICOMweb-compatible URI.
- `uri_type` (`URIType`, *optional*) – The expected DICOM resource type referenced by the object. If set, it validates that the resource-scope of the *dicomweb_uri* matches the expected type.
- `permissive` (*bool*) – Set if permissive handling of UIDs (if any) in `dicomweb_uri` is required. See the class initializer docstring for details.

Returns
The newly constructed *URI* object.

Return type
URI

Raises
`ValueError` – If the URI cannot be parsed or the actual URI type doesn't match the specified expected *uri_type*.

`instance_uri()`

Returns *URI* for the DICOM Instances within this object.

Return type
dicomweb_client.uri.URI

`property parent: URI`

Returns a URI to the "parent" resource.

Depending on the *type* of the current *URI*, the *URI* of the parent resource is defined as:

Current	Suffixed	Parent
Service	N/A	Service
Study	No	Service
Study	Yes	Study
Series	No	Study
Series	Yes	Series
Instance	No	Series
Instance	Yes	Instance
Frame	No	Instance
Frame	Yes	Frame

Returns
An instance pointing to the parent resource.

Return type
URI

`property permissive: bool`

Returns the `permissive` parameter value in the initializer.

Return type
bool

`property series_instance_uid: Optional[str]`

Returns the Series UID, if available.

Return type
`typing.Optional[str]`

series_uri()

Returns *URI* for the DICOM Series within this object.

Return type
dicomweb_client.uri.URI

property sop_instance_uid: Optional[str]

Returns the Instance UID, if available.

Return type
`typing.Optional[str]`

property study_instance_uid: Optional[str]

Returns the Study UID, if available.

Return type
`typing.Optional[str]`

study_uri()

Returns *URI* for the DICOM Study within this object.

Return type
dicomweb_client.uri.URI

property suffix: Optional[*URISuffix*]

Returns the DICOM resource suffix, if available.

Return type
`typing.Optional[`*dicomweb_client.uri.URISuffix*`]`

property type: *URIType*

The *URIType* of DICOM resource referenced by the object.

Return type
dicomweb_client.uri.URIType

update(*base_url=None*, *study_instance_uid=None*, *series_instance_uid=None*, *sop_instance_uid=None*, *frames=None*, *suffix=None*, *permissive=False*)

Creates a new *URI* object based on the current one.

Replaces the specified *URI* components in the current *URI* to create the new one.

Parameters

- **base_url** (*str, optional*) – DICOMweb service HTTP[S] URL to use in the new *URI* or *None* if the *base_url* from the current *URI* should be used.
- **study_instance_uid** (*str, optional*) – Study Instance UID to use in the new *URI* or *None* if the *study_instance_uid* from the current *URI* should be used.
- **series_instance_uid** (*str, optional*) – Series Instance UID to use in the new *URI* or *None* if the *series_instance_uid* from the current *URI* should be used.
- **sop_instance_uid** (*str, optional*) – SOP Instance UID to use in the new *URI* or *None* if the *sop_instance_uid* from the current *URI* should be used.
- **frames** (*Sequence[int], optional*) – Frame numbers to use in the new *URI* or *None* if the *frames* from the current *URI* should be used.

- **suffix** (`URISuffix`, *optional*) – Suffix to use in the new *URI* or *None* if the *suffix* from the current *URI* should be used.
- **permissive** (*bool, optional*) – Set if permissive handling of UIDs (if any) in the updated `URI` is required. See the class initializer docstring for details.

Returns
The newly constructed *URI* object.

Return type
URI

Raises
ValueError – If the new *URI* is invalid (e.g., if only the SOP Instance UID is specified, but the Series Instance UID is missing in the current *URI*).

class dicomweb_client.uri.**URISuffix**(*value*)

Bases: `Enum`

Optional suffixes for a DICOM resource.

METADATA = 'metadata'

RENDERED = 'rendered'

THUMBNAIL = 'thumbnail'

class dicomweb_client.uri.**URIType**(*value*)

Bases: `Enum`

Type of DICOM resource the URI points to.

FRAME = 'frame'

INSTANCE = 'instance'

SERIES = 'series'

SERVICE = 'service'

STUDY = 'study'

dicomweb_client.uri.**build_query_string**(*params=None*)

Build query string for a request message.

Parameters
params (*Union[Dict[str, Any], None], optional*) – Query parameters as mapping of key-value pairs; in case a key should be included more than once with different values, values need to be provided in form of an iterable (e.g., `{"key": ["value1", "value2"]}` will result in "?key=value1&key=value2")

Returns
Query string

Return type
str

dicomweb_client.uri.**parse_query_parameters**(*fuzzymatching=None*, *limit=None*, *offset=None*, *fields=None*, *search_filters=None*)

Parse query parameters for inclusion into a query string.

Parameters

- **fuzzymatching** (*Union[bool, None], optional*) – Whether fuzzy semantic matching should be performed
- **limit** (*Union[int, None], optional*) – Maximum number of results that should be returned
- **offset** (*Union[int, None], optional*) – Number of results that should be skipped
- **fields** (*Union[Sequence[str], None], optional*) – Names of fields (attributes) that should be included in results
- **search_filters** (*Union[Dict[str, Any], None], optional*) – Search filter criteria as key-value pairs, where *key* is a keyword or a tag of the attribute and *value* is the expected value that should match

Returns
Sanitized and sorted query parameters

Return type
collections.OrderedDict

7.1.6 dicomweb_client.ext.gcp.session_utils module

7.1.7 dicomweb_client.ext.gcp.uri module

Utilities for Google Cloud Healthcare DICOMweb API URI manipulation.

For details, visit: https://cloud.google.com/healthcare

`class dicomweb_client.ext.gcp.uri.GoogleCloudHealthcareURL(`*project_id*, *location*, *dataset_id*, *dicom_store_id*`)`

Bases: `object`

URL container for DICOM Stores under the Google Cloud Healthcare API.

This class facilitates the parsing and creation of `URI.base_url` corresponding to DICOMweb API Service URLs under the v1 API. The URLs are of the form: `https://healthcare.googleapis.com/v1/projects/{project_id}/locations/{location}/datasets/{dataset_id}/dicomStores/{dicom_store_id}/dicomWeb`

project_id

The ID of the GCP Project that contains the DICOM Store.

Type
str

location

The Region name of the geographic location configured for the Dataset that contains the DICOM Store.

Type
str

dataset_id

The ID of the Dataset that contains the DICOM Store.

Type
str

`dicom_store_id`

The ID of the DICOM Store.

Type

str

`dataset_id: str`

`dicom_store_id: str`

`classmethod from_string(`*base_url*`)`

Create an instance from *base_url*.

Parameters

`base_url` (*str*) – The URL for the DICOMweb API Service endpoint corresponding to a CHC API DICOM Store. See class docstring for supported formats.

Raises

`ValueError` – If `base_url` does not match the specifications in the class docstring.

Return type

dicomweb_client.ext.gcp.uri.GoogleCloudHealthcareURL

`location: str`

`project_id: str`

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

Symbols