Sentinelsat Documentation

Release 0.11

Marcel Wille, Kersten Clauss

Aug 16, 2017

Contents

Sentinelsat makes searching, downloading and retrieving the metadata of Sentinel satellite images from the Copernicus Open Access Hub easy.

It offers an easy-to-use command line interface

```
sentinel search --sentinel 2 --cloud 30 user password search_polygon.geojson
```

and a powerful Python API.

```
from sentinelsat import SentinelAPI, read_geojson, geojson_to_wkt

api = SentinelAPI('user', 'password')
footprint = geojson_to_wkt(read_geojson('search_polygon.geojson'))
products = api.query(footprint,
                     producttype='SLC',
                     orbitdirection='ASCENDING')
api.download_all(products)
```

CHAPTER 1

Contents

Installation

Install `sentinelsat` through pip:

```
pip install sentinelsat
```

Tests

To run the tests on `sentinelsat`:

```
git clone https://github.com/sentinelsat/sentinelsat.git
cd sentinelsat
pip install -e .[test]
py.test -v
```

By default, prerecorded responses to Copernicus Open Access Hub queries are used to not be affected by its downtime. To allow the tests to run actual queries against Copernicus Open Access Hub set the environment variables

and add `--vcr disable` to `py.test` arguments. To update the recordings use either `--vcr record_new` or `--vcr reset`.

Supported Python versions

Sentinelsat has been tested with Python versions 2.7 and 3.4+. Earlier Python 3 versions are expected to work as well as long as the dependencies are fulfilled.

Optional dependencies

The convenience functions `to_dataframe()` and `to_geodataframe()` require `pandas` and/or `geopandas` to be present.

Command Line Interface

Sentinelsat's CLI is divided into two commands:

- `sentinel search` to query and download a number of images over an area
- `sentinel download` to download individual images by their unique identifier

Quickstart

A basic search query consists of a search polygon as well as the username and password to access the Copernicus Open Access Hub.

```
sentinel search [OPTIONS] <user> <password> <geojson>
```

Search areas are provided as GeoJSON polygons, which can be created with QGIS or geojson.io. If you do not specify a start and end date only products published in the last 24 hours will be queried.

Start and end dates refer to the acquisition date given by the beginPosition of the products, i.e. the start of the acquisition time.

Sentinel-1

Search and download all Sentinel-1 scenes of type SLC, in descending orbit for the year 2015.

```
sentinel search -s 20150101 -e 20151231 -d \
--producttype SLC -q "orbitdirection=Descending" \
-u "https://scihub.copernicus.eu/dhus" <user> <password> poly.geojson
```

Download a single Sentinel-1 GRDH scene covering Santa Claus Village in Finland on Christmas Eve 2015.

```
sentinel download --md5 -u "https://scihub.copernicus.eu/dhus/" <user> <password>␣
↪a9048d1d-fea6-4df8-bedd-7bcb212be12e
```

Sentinel-2

Search and download Sentinel-2 scenes for January 2016 with a maximum cloud cover of 40%.

```
sentinel search -s 20160101 -e 20160131 --sentinel 2 --cloud 40 -d <user> <password>
↪<poly.geojson>
```

Download all Sentinel-2 scenes published in the last 24 hours.

```
sentinel search --sentinel 2 -d <user> <password> <poly.geojson>
```

sentinel search

```
sentinel search [OPTIONS] <user> <password> <geojson>
```

Options:

-s	--start	TEXT	Start date of the query in the format YYYYMMDD.
-e	--end	TEXT	End date of the query in the format YYYYMMDD.
-d	--download		Download all results of the query.
-f	--footprints		Create geojson file search_footprints.geojson with footprints of the query result.
-p	--path	PATH	Set the path where the files will be saved.
-q	--query	TEXT	Extra search keywords you want to use in the query. Separate keywords with comma. Example: 'producttype=GRD,polarisationmode=HH'.
-u	--url	TEXT	Define another API URL. Default URL is 'https://scihub.copernicus.eu/apihub/'.
	--md5		Verify the MD5 checksum and write corrupt product ids and filenames to corrupt_scenes.txt.
	--sentinel		Limit search to a Sentinel satellite (constellation).
	--instrument		Limit search to a specific instrument on a Sentinel satellite.
	--producttype		Limit search to a Sentinel product type.
-c	--cloud	INT	Maximum cloud cover in percent. (Automatically sets –sentinel2)
	--help		Show help message and exit.
	--version		Show version number and exit.

ESA maintains a list of valid search keywords that can be used with `--query`.

The options `--sentinel`, `--instrument` and `--producttype` are mutually exclusive and follow a hierarchy from most specific to least specific, i.e. `--producttype` > `--instrument` > `--sentinel`. Only the most specific option will be included in the search when multiple ones are given.

sentinel download

```
sentinel download [OPTIONS] <user> <password> <productid>
```

Options:

-p	--path	PATH	Set the path where the files will be saved.
-u	--url	TEXT	Define another API URL. Default URL is 'https://scihub.copernicus.eu/apihub/'.
	--md5		Verify the MD5 checksum and write corrupt product ids and filenames to corrupt_scenes.txt.
	--version		Show version number and exit.

Python API

Quickstart

```
# connect to the API
from sentinelsat import SentinelAPI, read_geojson, geojson_to_wkt
api = SentinelAPI('user', 'password', 'https://scihub.copernicus.eu/dhus')

# download single scene by known product id
api.download(<product_id>)

# search by polygon, time, and SciHub query keywords
footprint = geojson_to_wkt(read_geojson('map.geojson'))
products = api.query(footprint,
                     '20151219', date(2015, 12, 29),
                     platformname = 'Sentinel-2',
                     cloudcoverpercentage = '[0 TO 30]')
```

```
# download all results from the search
api.download_all(products)

# GeoJSON FeatureCollection containing footprints and metadata of the scenes
api.to_geojson(products)

# GeoPandas GeoDataFrame with the metadata of the scenes and the footprints as␣
↪geometries
api.to_geopandas(products)

# Get basic information about the product: its title, file size, MD5 sum, date,␣
↪footprint and
# its download url
api.get_product_odata(<product_id>)

# Get the product's full metadata available on the server
api.get_product_odata(<product_id>, full=True)
```

Valid search query keywords can be found at the Copernicus Open Access Hub documentation.

Sorting & Filtering

In addition to the search query keywords sentinelsat allows filtering and sorting of search results before download. To simplify these operations sentinelsat offers the convenience functions `to_geojson()`,`to_dataframe()` and `to_geodataframe()` which return the search results as a GeoJSON object, Pandas DataFrame or a GeoPandas GeoDataFrame, respectively. `to_dataframe()` and `to_geodataframe()` require `pandas` and `geopandas` to be installed, respectively.

In this example we query Sentinel-2 scenes over a location and convert the query results to a Pandas DataFrame. The DataFrame is then sorted by cloud cover and ingestion date. We limit the query to first 5 results within our timespan and download them, starting with the least cloudy scene. Filtering can be done with all data types, as long as you pass the `id` to the download function.

```
# connect to the API
from sentinelsat import SentinelAPI, read_geojson, geojson_to_wkt

api = SentinelAPI('user', 'password', 'https://scihub.copernicus.eu/dhus')

# search by polygon, time, and SciHub query keywords
footprint = geojson_to_wkt(read_geojson('map.geojson'))
products = api.query(footprint,
                     '20151219', date(2015, 12, 29),
                     platformname = 'Sentinel-2')

# convert to Pandas DataFrame
products_df = api.to_dataframe(products)

# sort and limit to first 5 sorted products
products_df_sorted = products_df.sort_values(['cloudcoverpercentage', 'ingestiondate
↪'], ascending=[True, True])
products_df_sorted = products_df_sorted.head(5)

# download sorted and reduced products
api.download_all(products_df_sorted['id'])
```

Getting Product Metadata

Sentinelsat provides two methods for retrieving product metadata from the server, one for each API offered by the Copernicus Open Access Hub:

- `query()` for OpenSearch (Solr), which supports filtering products by their attributes and returns metadata for all matched products at once.
- `get_product_odata()` for OData, which can be queried one product at a time but provides the full metadata available for each product, as well as information about the product file such as the file size and checksum, which are not available from OpenSearch.

Both methods return a dictionary containing the metadata items. More specifically, `query()` returns a dictionary with an entry for each returned product with its ID as the key and the attributes' dictionary as the value.

All of the attributes returned by the OpenSearch API have a corresponding but differently named attribute in the OData's full metadata response. See the DataHubSystem's metadata definition files to find the exact mapping between them (OpenSearch attributes have a `<solrField>` tag added): - Sentinel-1 attributes - Sentinel-2 attributes - Sentinel-3 attributes

OpenSearch example

```
>>> api.query(initial_date='NOW-8HOURS', producttype='SLC')
OrderedDict([('04548172-c64a-418f-8e83-7a4d148adf1e',
              {'acquisitiontype': 'NOMINAL',
               'beginposition': datetime.datetime(2017, 4, 25, 15, 56, 12, 814000),
               'endposition': datetime.datetime(2017, 4, 25, 15, 56, 39, 758000),
               'filename': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_
↪01AF91_46FF.SAFE',
               'footprint': 'POLYGON ((34.322010 0.401648,36.540989 0.876987,36.
↪884121 -0.747357,34.664474 -1.227940,34.322010 0.401648))',
               'format': 'SAFE',
               'gmlfootprint': '<gml:Polygon srsName="http://www.opengis.net/gml/srs/
↪epsg.xml#4326" xmlns:gml="http://www.opengis.net/gml">\n   <gml:outerBoundaryIs>\n
↪    <gml:LinearRing>\n         <gml:coordinates>0.401648,34.322010 0.876987,36.
↪540989 -0.747357,36.884121 -1.227940,34.664474 0.401648,34.322010</gml:coordinates>
↪\n      </gml:LinearRing>\n   </gml:outerBoundaryIs>\n</gml:Polygon>',
               'identifier': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_
↪01AF91_46FF',
               'ingestiondate': datetime.datetime(2017, 4, 25, 19, 23, 45, 956000),
               'instrumentname': 'Synthetic Aperture Radar (C-band)',
               'instrumentshortname': 'SAR-C SAR',
               'lastorbitnumber': 16302,
               'lastrelativeorbitnumber': 130,
               'link': "https://scihub.copernicus.eu/apihub/odata/v1/Products(
↪'04548172-c64a-418f-8e83-7a4d148adf1e')/$value",
               'link_alternative': "https://scihub.copernicus.eu/apihub/odata/v1/
↪Products('04548172-c64a-418f-8e83-7a4d148adf1e')/",
               'link_icon': "https://scihub.copernicus.eu/apihub/odata/v1/Products(
↪'04548172-c64a-418f-8e83-7a4d148adf1e')/Products('Quicklook')/$value",
               'missiondatatakeid': 110481,
               'orbitdirection': 'ASCENDING',
               'orbitnumber': 16302,
               'platformidentifier': '2014-016A',
               'platformname': 'Sentinel-1',
               'polarisationmode': 'VV VH',
               'productclass': 'S',
               'producttype': 'SLC',
```

```
               'relativeorbitnumber': 130,
               'sensoroperationalmode': 'IW',
               'size': '7.1 GB',
               'slicenumber': 8,
               'status': 'ARCHIVED',
               'summary': 'Date: 2017-04-25T15:56:12.814Z, Instrument: SAR-C SAR,␣
↪Mode: VV VH, Satellite: Sentinel-1, Size: 7.1 GB',
               'swathidentifier': 'IW1 IW2 IW3',
               'title': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_
↪01AF91_46FF',
               'uuid': '04548172-c64a-418f-8e83-7a4d148adf1e'}),
...
```

OData example

Only the most basic information available from the OData API is returned by default, if `full=True` is not set. The full metadata query response is quite large and not always nrequired, so it is not requested by default.

```
>>> api.get_product_odata('04548172-c64a-418f-8e83-7a4d148adf1e')
{'date': datetime.datetime(2017, 4, 25, 15, 56, 12, 814000),
 'footprint': 'POLYGON((34.322010 0.401648,36.540989 0.876987,36.884121 -0.747357,34.
↪664474 -1.227940,34.322010 0.401648))',
 'id': '04548172-c64a-418f-8e83-7a4d148adf1e',
 'md5': 'E5855D1C974171D33EE4BC08B9D221AE',
 'size': 4633501134,
 'title': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_01AF91_46FF',
 'url': "https://scihub.copernicus.eu/apihub/odata/v1/Products('04548172-c64a-418f-
↪8e83-7a4d148adf1e')/$value"}
```

With `full=True` we receive the full metadata available for the product.

```
>>> api.get_product_odata('04548172-c64a-418f-8e83-7a4d148adf1e', full=True)
{'Acquisition Type': 'NOMINAL',
 'Carrier rocket': 'Soyuz',
 'Cycle number': 107,
 'Date': datetime.datetime(2017, 4, 25, 15, 56, 12, 814000),
 'Filename': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_01AF91_46FF.SAFE
↪',
 'Footprint': '<gml:Polygon srsName="http://www.opengis.net/gml/srs/epsg.xml#4326"␣
↪xmlns:gml="http://www.opengis.net/gml">\n   <gml:outerBoundaryIs>\n
↪<gml:LinearRing>\n         <gml:coordinates>0.401648,34.322010 0.876987,36.540989 -
↪0.747357,36.884121 -1.227940,34.664474 0.401648,34.322010</gml:coordinates>\n
↪</gml:LinearRing>\n   </gml:outerBoundaryIs>\n</gml:Polygon>',
 'Format': 'SAFE',
 'Identifier': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_01AF91_46FF',
 'Ingestion Date': datetime.datetime(2017, 4, 25, 19, 23, 45, 956000),
 'Instrument': 'SAR-C',
 'Instrument abbreviation': 'SAR-C SAR',
 'Instrument description': '<a target="_blank" href="https://sentinel.esa.int/web/
↪sentinel/missions/sentinel-1">https://sentinel.esa.int/web/sentinel/missions/
↪sentinel-1</a>',
 'Instrument description text': 'The SAR Antenna Subsystem (SAS) is developed and␣
↪build by AstriumGmbH. It is a large foldable planar phased array antenna, which␣
↪isformed by a centre panel and two antenna side wings. In deployedconfiguration the␣
↪antenna has an overall aperture of 12.3 x 0.84 m.The antenna provides a fast␣
↪electronic scanning capability inazimuth and elevation and is based on low loss and␣
↪highly stablewaveguide radiators build in carbon fibre technology, which arealready␣
↪successfully used by the TerraSAR-X radar imaging mission.The SAR Electronic␣
↪Subsystem (SES) is developed and build byAstrium Ltd. It provides all radar control,
↪ IF/ RF signalgeneration and receive data handling functions for the antenna panels␣
↪The fully redundant SES is based on a channelisedarchitecture with one transmit and␣
↪two receive chains, providing amodular approach to the generation and reception of␣
↪wide-bandsignals and the handling of multi-polarisation modes. One keyfeature is␣
↪the implementation of the Flexible Dynamic BlockAdaptive Quantisation (FDBAQ) data␣
```

```
 'Instrument mode': 'IW',
 'Instrument name': 'Synthetic Aperture Radar (C-band)',
 'Instrument swath': 'IW1 IW2 IW3',
 'JTS footprint': 'POLYGON ((34.322010 0.401648,36.540989 0.876987,36.884121 -0.
↪747357,34.664474 -1.227940,34.322010 0.401648))',
 'Launch date': 'April 3rd, 2014',
 'Mission datatake id': 110481,
 'Mission type': 'Earth observation',
 'Mode': 'IW',
 'NSSDC identifier': '2014-016A',
 'Operator': 'European Space Agency',
 'Orbit number (start)': 16302,
 'Orbit number (stop)': 16302,
 'Pass direction': 'ASCENDING',
 'Phase identifier': 1,
 'Polarisation': 'VV VH',
 'Product class': 'S',
 'Product class description': 'SAR Standard L1 Product',
 'Product composition': 'Slice',
 'Product level': 'L1',
 'Product type': 'SLC',
 'Relative orbit (start)': 130,
 'Relative orbit (stop)': 130,
 'Satellite': 'Sentinel-1',
 'Satellite description': '<a target="_blank" href="https://sentinel.esa.int/web/
↪sentinel/missions/sentinel-1">https://sentinel.esa.int/web/sentinel/missions/
↪sentinel-1</a>',
 'Satellite name': 'Sentinel-1',
 'Satellite number': 'A',
 'Sensing start': datetime.datetime(2017, 4, 25, 15, 56, 12, 814000),
 'Sensing stop': datetime.datetime(2017, 4, 25, 15, 56, 39, 758000),
 'Size': '7.1 GB',
 'Slice number': 8,
 'Start relative orbit number': 130,
 'Status': 'ARCHIVED',
 'Stop relative orbit number': 130,
 'Timeliness Category': 'Fast-24h',
 'date': datetime.datetime(2017, 4, 25, 15, 56, 12, 814000),
 'footprint': 'POLYGON((34.322010 0.401648,36.540989 0.876987,36.884121 -0.747357,34.
↪664474 -1.227940,34.322010 0.401648))',
 'id': '04548172-c64a-418f-8e83-7a4d148adf1e',
 'md5': 'E5855D1C974171D33EE4BC08B9D221AE',
 'size': 4633501134,
 'title': 'S1A_IW_SLC__1SDV_20170425T155612_20170425T155639_016302_01AF91_46FF',
 'url': "https://scihub.copernicus.eu/apihub/odata/v1/Products('04548172-c64a-418f-
↪8e83-7a4d148adf1e')/$value"}
```

Logging

Sentinelsat logs to `sentinelsat` and the API to `sentinelsat.SentinelAPI`.

There is no predefined logging handler, so in order to have your script print the log messages, either use `logging.baseConfig`

```
import logging

logging.basicConfig(format='%(message)s', level='INFO')
```

or add a custom handler for `sentinelsat` (as implemented in `cli.py`)

```
import logging

logger = logging.getLogger('sentinelsat')
logger.setLevel('INFO')

h = logging.StreamHandler()
h.setLevel('INFO')
fmt = logging.Formatter('%(message)s')
h.setFormatter(fmt)
logger.addHandler(h)
```

API

Change Log

All notable changes to `sentinelsat` will be listed here.

[0.11] – 2017-06-01

Changed

- Replace `pycurl` dependency with `requests`. This makes installation significantly easier. (#117)
- An exception is raised in `download_all()` if all downloads failed.
- Change 'Sentinels Scientific Datahub' to 'Copernicus Open Access Hub' (#100)
- Renamed `py.test` option `--vcr reset_all` to `--vcr reset` to better reflect its true behavior.

[0.10] – 2017-05-30

Added

- GeoJSON footprints are allowed to contain just a single geometry instead of a feature collection. Any geometry type that has a WKT equivalent is supported (rather than only Polygons).
- `get_product_odata()` can be used to get the full metadata information available for a product if `full=True` is set.
- Added `query_raw()` that takes full text search string as input and returns a parsed dictionary just like the updated `query()` method.
- CLI: `--sentinel=<int>` option to select satellite (constellation)

Changed

- `SentinelAPI`, etc. can be directly imported from `sentinelsat` rather than `sentinelsat.sentinel`.
- `query()` changes:

  - The `area` argument expects a WKT string as input instead of a coordinate string. (Issue #101)
  - Date arguments can be disabled by setting them to `None` and their values are validated on the client side. (Issue #101)
  - The return value has been changed to a dict of dicts of parsed metadata values. One entry per product with the product ID as the key.
- `download_all()` expects a list of product IDs as input. This is compatible with the output of `query()`.
- `get_coordinates()` has been replaced with functions `read_geojson()` and `geojson_to_wkt()`. (Issue #101)
- Use more compact and descriptive error messages from the response headers, if available.

Deprecated

- CLI: `--sentinel1` and `--sentinel2` will be removed with the next major release

Removed

- `to_dict()` has been removed since it is no longer required.
- `load_query()` has been made private (renamed to `_load_query()`).

Fixed

- Fixed invalid GeoJSON output in both the CLI and API. (Issue #104)
- Fixed broken reporting of failed downloads in the CLI. (Issue #88)
- Attempting to download a product with an invalid ID no longer creates an infinite loop and a more informative error message is displayed in the CLI.

[0.9.1] – 2017-03-06

Added

- `--version` option to command line utilities
- install requirements for building the documentation
- documentation of sorting with `to_*` convenience functions

[0.9] – 2017-02-26

Added

- Added `to_dict`, `to_dataframe` and `to_geodataframe` which convert the response content to respective types. The pandas, geopandas and shapely dependencies are not installed by default.

Changed

- `--footprints` now includes all returned product properties in the output.
- `KeyError('No results returned.')` is no longer returned for zero returned products in a response.
- Renamed `get_footprint` to `to_geojson` and `get_product_info` to `get_product_odata`.
- Added underscore to methods and functions that are not expected to be used outside the package.
- Instance variables `url` and `content` have been removed, `last_query` and `last_status_code` have been made private.

[0.8.1] – 2017-02-05

Added

- added a changelog

Changed

- use logging instead of print

Fixed

- docs represent new `query` and `download_all` behaviour

[0.8] – 2017-01-27

Added

- options to create new, reset or ignore vcr cassettes for testing

Changed

- `query` now returns a list of search results
- `download_all` requires the list of search results as an argument

Removed

- `SentinelAPI` does not save query results as class attributes

[0.7.4] – 2017-01-14

Added

- Travis tests for Python 3.6

[0.7.3] – 2016-12-09

Changed

- changed `SentinelAPI` `max_rows` attribute to `page_size` to better reflect pagination
- tests use `vcrpy` cassettes

Fixed

- support GeoJSON polygons with optional (third) z-coordinate

[0.7.1] – 2016-10-28

Added

- pagination support for query results

Changed

- number of query results per page set to 100

[0.6.5] – 2016-06-22

Added

- support for large queries

Changed

- Removed redundant information from Readme that is also present on Readthedocs

[0.6.4] – 2016-04-06-03

Changed

- `initial_date` / `--start` changed from ingestion to acquisition date

[0.6.1] – 2016-04-22

Added

- Sphinx documentation setup with autodoc and numpydoc
- Redthedocs.org integration

[0.5.5] – 2016-01-13

Added

- Sentinel-2 support

[0.5.1] – 2015-12-18

Added

- Travis added as continuous integration service for automated testing

[0.5] – 2015-12-09

Added

- validate downloaded products with their MD5 checksums

[0.4.3] – 2015-11-23

Added

- option to select a different dhus api `--url`

Changed

- `https://scihub.esa.int/apihub/` as standard url

[0.4] – 2015-09-28

Added

- method to manually select the CA certificate bundle
- function to return footprints of the queried Sentinel scenes

Fixed

- CA-certificate SSL errors

[0.3] – 2015-06-10

Added

- `--query` parameter to use extra search keywords in the cli

[0.1] – 2015-06-05

- first release

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

S