spacetrack Documentation

Release 0.13.6

Frazer McLean

Jun 21, 2020

Contents

`spacetrack` is a Python client for Space-Track, with methods for all request classes, keyword validation for request predicates, streaming download support, and automatic rate limiting.

Contents:

CHAPTER 1

Installation

The recommended installation method is using `pip`.

1.1 pip

```
$ pip install spacetrack
```

To install with `asyncio` support, install the async extra:

```
$ pip install spacetrack[async]
```

1.2 Git

```
$ git clone https://github.com/python-astrodynamics/spacetrack.git
Cloning into 'spacetrack'...
```

Check out a release tag

```
$ cd spacetrack
$ git checkout 0.13.6
```

CHAPTER 2

Usage

```
import spacetrack.operators as op
from spacetrack import SpaceTrackClient

st = SpaceTrackClient(identity='user@example.com', password='password')
```

Request classes are presented as methods on the *SpaceTrackClient* object. For example, `st.tle_publish()`. Each request class is part of a request controller. Since most request classes are only part of one request controller, `spacetrack` looks up the controller for you. It can be specified explicitly in several ways. All the following are equivalent:

```
st.tle_publish()
st.tle_publish(controller='basicspacedata')
st.basicspacedata.tle_publish()
st.generic_request('tle_publish')
st.generic_request('tle_publish', controller='basicspacedata')
```

Request predicates are passed as keyword arguments. Valid arguments can be checked using the *get_predicates()* method. The following are equivalent:

```
st.tle_publish.get_predicates()
st.tle_publish.get_predicates(controller='basicspacedata')
st.basicspacedata.tle_publish.get_predicates()
st.basicspacedata.get_predicates('tle_publish')
st.get_predicates('tle_publish')
st.get_predicates('tle_publish', controller='basicspacedata')
```

Returned object:

```
[Predicate(name='publish_epoch', type_='datetime', nullable=False),
 Predicate(name='tle_line1', type_='str', nullable=False),
 Predicate(name='tle_line2', type_='str', nullable=False)]
```

Internally, the client uses this mechanism to verify the keyword arguments. Types are not currently checked.

2.1 Streaming Downloads

It is possible to stream responses by passing `iter_content=True` (100 KiB chunks) or `iter_lines=True` to the request class methods.

2.1.1 Example

The same example is shown below synchronously and asynchronously.

```
import spacetrack.operators as op
from spacetrack import SpaceTrackClient

st = SpaceTrackClient(identity='user@example.com', password='password')

data = st.tle_latest(iter_lines=True, ordinal=1, epoch='>now-30',
                     mean_motion=op.inclusive_range(0.99, 1.01),
                     eccentricity=op.less_than(0.01), format='tle')

with open('tle_latest.txt', 'w') as fp:
    for line in data:
        fp.write(line + '\n')
```

```
import asyncio

import spacetrack.operators as op
from spacetrack.aio import AsyncSpaceTrackClient


async def download_latest_tles():
    st = AsyncSpaceTrackClient(identity='user@example.com',
                               password='password')

    with st:
        data = await st.tle_latest(
            iter_lines=True, ordinal=1, epoch='>now-30',
            mean_motion=op.inclusive_range(0.99, 1.01),
            eccentricity=op.less_than(0.01), format='tle')

        with open('tle_latest.txt', 'w') as fp:
            async for line in data:
                fp.write(line + '\n')

loop = asyncio.get_event_loop()
loop.run_until_complete(download_latest_tles())
```

2.2 File Uploads

To use the *upload* request class, pass a *file* keyword argument with the opened file:

```
from spacetrack import SpaceTrackClient

st = SpaceTrackClient(identity='user@example.com', password='password')

with open('somefile.txt', 'rb') as fp:
    st.upload(file=fp)
```

2.3 Rate Limiter

> "Space-track throttles API use in order to maintain consistent performance for all users. To avoid error messages, please limit your query frequency to less than 20 requests per minute."

The client will ensure that no more than 19 HTTP requests are sent per minute by sleeping if the rate exceeds this. This will be logged to the spacetrack module's logger. You can register a callback with the *SpaceTrackClient* or *AsyncSpaceTrackClient* classes. It will be passed the time that the module is sleeping until, in seconds since the epoch (as with `time.time()`).

```
import time

from spacetrack import SpaceTrackClient

def mycallback(until):
    duration = int(round(until - time.time()))
    print('Sleeping for {:d} seconds.'.format(duration))

st = SpaceTrackClient(identity='user@example.com', password='password')
st.callback = mycallback
```

2.4 Sample Queries

The Space-Track website lists some sample queries, which are shown here using the Python module.

```
output = st.boxscore(format='csv')
```

```
decay_epoch = op.inclusive_range(date(2012, 7, 2), date(2012, 7, 9))
st.decay(decay_epoch=decay_epoch, orderby=['norad_cat_id', 'precedence'], format=
↪'xml')
```

```
st.satcat(launch='>now-7', current='Y', orderby='launch desc', format='html')
```

```
st.satcat(period=op.inclusive_range(1430, 1450), current='Y',
          decay=None, orderby='norad_cat_id', format='html')
```

```
st.satcat(period=op.less_than(128), decay=None, current='Y')
```

```
st.tle_latest(ordinal=1, epoch='>now-30',
              mean_motion=op.inclusive_range(0.99, 1.01),
              eccentricity=op.less_than(0.01), format='tle')
```

```
st.tle_latest(ordinal=1, epoch='>now-30', mean_motion=op.greater_than(11.25),
              format='3le')
```

```
st.tle_latest(favorites='Amateur', ordinal=1, epoch='>now-30', format='3le')
```

```
st.tle_latest(
    ordinal=1,
    norad_cat_id=[
        36000,
        op.inclusive_range(36001, 36004),
        op.like(36005),
        op.startswith(3600),
        36010
    ],
```

(continues on next page)

(continued from previous page)

```
    orderby='norad_cat_id',
    format='html')
```

```
st.tle(norad_cat_id=25544, orderby='epoch desc', limit=22, format='tle')
```

```
st.omm(norad_cat_id=25544, orderby='epoch desc', limit=22, format='xml')
```

```
st.tip(norad_cat_id=[60, 38462, 38351], format='html')
```

```
st.cdm(constellation='iridium', limit=10, orderby='creation_date desc', format=
↪'html')
```

```
st.cdm(constellation='iridium', limit=10, orderby='creation_date desc', format='kvn
↪')
```

```
st.cdm(
    constellation='intelsat', tca='>now',
    predicates=['message_for', 'tca', 'miss_distance'],
    orderby='miss_distance', format='html', metadata=True)
```

```
st.cdm(
    constellation='intelsat', tca='>now',
    predicates=['message_for', 'tca', 'miss_distance'],
    orderby='miss_distance', format='kvn')
```

CHAPTER 3

Module Reference

3.1 SpaceTrack

exception spacetrack.base.**AuthenticationError**
Space-Track authentication error.

exception spacetrack.base.**UnknownPredicateTypeWarning**
Used to warn when a predicate type is unknown.

class spacetrack.base.**Predicate**(*name*, *type_*, *nullable=False*, *default=None*, *values=None*)
Hold Space-Track predicate information.

The current goal of this class is to print the repr for the user.

class spacetrack.base.**SpaceTrackClient**(*identity*, *password*, *base_url='https://www.space-track.org/'*)
SpaceTrack client class.

Parameters

- **identity** – Space-Track username.
- **password** – Space-Track password.
- **base_url** – May be overridden to use e.g. https://testing.space-track.org/

For more information, refer to the Space-Track documentation.

request_controllers
Ordered dictionary of request controllers and their request classes in the following order.

- *basicspacedata*
- *expandedspacedata*
- *fileshare*
- *spephemeris*

For example, if the `spacetrack.file` method is used without specifying which controller, the client will choose the *fileshare* controller (which comes before *spephemeris*).

Note: If new request classes and/or controllers are added to the Space-Track API but not yet to this library, you can safely subclass *SpaceTrackClient* with a copy of this ordered dictionary to add them.

That said, please open an issue on GitHub for me to add them to the library.

session = None
requests.Session instance. It can be mutated to configure e.g. proxies.

authenticate()
Authenticate with Space-Track.

Raises *spacetrack.base.AuthenticationError* – Incorrect login details.

Note: This method is called automatically when required.

generic_request(*class_*, *iter_lines=False*, *iter_content=False*, *controller=None*, *parse_types=False*, ***kwargs*)
Generic Space-Track query.

The request class methods use this method internally; the public API is as follows:

```
st.tle_publish(*args, **kw)
st.basicspacedata.tle_publish(*args, **kw)
st.file(*args, **kw)
st.fileshare.file(*args, **kw)
st.spephemeris.file(*args, **kw)
```

They resolve to the following calls respectively:

```
st.generic_request('tle_publish', *args, **kw)
st.generic_request('tle_publish', *args, controller='basicspacedata', **kw)
st.generic_request('file', *args, **kw)
st.generic_request('file', *args, controller='fileshare', **kw)
st.generic_request('file', *args, controller='spephemeris', **kw)
```

Parameters

- **class_** – Space-Track request class name
- **iter_lines** – Yield result line by line
- **iter_content** – Yield result in 100 KiB chunks.
- **controller** – Optionally specify request controller to use.
- **parse_types** – Parse string values in response according to type given in predicate information, e.g. `'2017-01-01'` -> `datetime.date(2017, 1, 1)`.
- ****kwargs** – These keywords must match the predicate fields on Space-Track. You may check valid keywords with the following snippet:

```
spacetrack = SpaceTrackClient(...)
spacetrack.tle.get_predicates()
# or
spacetrack.get_predicates('tle')
```

See *_stringify_predicate_value()* for which Python objects are converted appropriately.

Yields Lines—stripped of newline characters—if `iter_lines=True`

Yields 100 KiB chunks if `iter_content=True`

Returns

Parsed JSON object, unless `format` keyword argument is passed.

> **Warning:** Passing `format='json'` will return the JSON **unparsed**. Do not set `format` if you want the parsed JSON object returned!

get_predicates(*class_*, *controller=None*)
Get full predicate information for given request class, and cache for subsequent calls.

3.2 Experimental `asyncio` API

> **Warning:** The `asyncio` API is not thoroughly unit tested, use with caution. Please report issues on GitHub.

class spacetrack.aio.**AsyncSpaceTrackClient**(*identity*, *password*, *base_url='https://www.space-track.org/'*)
Bases: *spacetrack.base.SpaceTrackClient*

Asynchronous SpaceTrack client class.

This class should be considered experimental.

It must be closed by calling *close()*. Alternatively, instances of this class can be used as a context manager.

Parameters

- **identity** – Space-Track username.
- **password** – Space-Track password.
- **base_url** – May be overridden to use e.g. https://testing.space-track.org/

For more information, refer to the Space-Track documentation.

session
`aiohttp.ClientSession` instance.

close()
Close aiohttp session.

authenticate()
Authenticate with Space-Track.

Raises *spacetrack.base.AuthenticationError* – Incorrect login details.

> **Note:** This method is called automatically when required.

generic_request(*class_*, *iter_lines=False*, *iter_content=False*, *controller=None*, *parse_types=False*, ***kwargs*)
Generic Space-Track query coroutine.

The request class methods use this method internally; the public API is as follows:

```
st.tle_publish(*args, **st)
st.basicspacedata.tle_publish(*args, **st)
st.file(*args, **st)
st.fileshare.file(*args, **st)
st.spephemeris.file(*args, **st)
```

They resolve to the following calls respectively:

```
st.generic_request('tle_publish', *args, **st)
st.generic_request('tle_publish', *args, controller='basicspacedata', **st)
st.generic_request('file', *args, **st)
st.generic_request('file', *args, controller='fileshare', **st)
st.generic_request('file', *args, controller='spephemeris', **st)
```

Parameters

- **class** – Space-Track request class name
- **iter_lines** – Yield result line by line
- **iter_content** – Yield result in 100 KiB chunks.
- **controller** – Optionally specify request controller to use.
- **parse_types** – Parse string values in response according to type given in predicate information, e.g. `'2017-01-01'` -> `datetime.date(2017, 1, 1)`.
- ****kwargs** – These keywords must match the predicate fields on Space-Track. You may check valid keywords with the following snippet:

```
spacetrack = AsyncSpaceTrackClient(...)
await spacetrack.tle.get_predicates()
# or
await spacetrack.get_predicates('tle')
```

See *_stringify_predicate_value()* for which Python objects are converted appropriately.

Yields Lines—stripped of newline characters—if `iter_lines=True`

Yields 100 KiB chunks if `iter_content=True`

Returns

Parsed JSON object, unless `format` keyword argument is passed.

> **Warning:** Passing `format='json'` will return the JSON **unparsed**. Do not set `format` if you want the parsed JSON object returned!

get_predicates(*class_*, *controller=None*)

Get full predicate information for given request class, and cache for subsequent calls.

3.3 SpaceTrack operators

Refer to REST Operators in the Space-Track documentation.

spacetrack.operators.**greater_than**(*value*)
    '>value'.

spacetrack.operators.**less_than**(*value*)
    '<value'.

spacetrack.operators.**not_equal**(*value*)
    '<>value'.

spacetrack.operators.**inclusive_range**(*left*, *right*)
    'left--right'.

spacetrack.operators.**like**(*value*)
'~~value'.

spacetrack.operators.**startswith**(*value*)
'^value'.

spacetrack.operators.**_stringify_predicate_value**(*value*)
Convert Python objects to Space-Track compatible strings

- Booleans (`True` -> `'true'`)
- Sequences (`[25544, 34602]` -> `'25544,34602'`)
- dates/datetimes (`date(2015, 12, 23)` -> `'2015-12-23'`)
- `None` -> `'null-val'`

CHAPTER 4

Change Log

4.1 Unreleased

N/A

4.2 0.13.6

4.2.1 Fixed

- Regression in 0.13 that prevented `spephemeris/download` from working by trying to load a model definition which it doesn't have.

4.3 0.13.5

4.3.1 Fixed

- The 'text' predicate type is now understood.
- Unknown predicate types issue a warning instead of raising an exception.

4.4 0.13.4

4.4.1 Added

- `SpaceTrackClient` gained a `base_url` parameter to allow the use of an alternate Space-Track server.

4.5 0.13.3

4.5.1 Fixed

- The deprecation warning about importing `Sequence` or `Mapping` from `collections` instead of `collections.abc`.

4.6 0.13.2

4.6.1 Fixed

- The `async` extra installs aiohttp 2 because `spacetrack` is not yet aiohttp 3 compatible.
- Deprecation warnings about invalid escape sequences.

4.7 0.13.1

4.7.1 Fixed

- `spacetrack` can be installed with setuptools v38.0+, which requires `install_requires` in `setup.py` to be ordered.

4.8 0.13.0

4.8.1 Added

- `parse_types` flag to optionally parse types as described by the `modeldef` API.
- Compatibility with `maneuver` and `maneuver_history` request classes for `expandedspacedata` request controller.
- Compatibility with `upload` and `delete` request classes for `fileshare` request controller. ### Fixed
- Predicates with the enum type are parsed correctly. Previously, single-valued enums had `None` as a second value, and enums with more than two values only had the first and last value due to the regex match not capturing repeated groups. The values aren't used by `spacetrack`, so the bug went unnoticed.
- Exception on Python 3.5+ in threads without an `asyncio` event loop (even using the normal `SpaceTrackClient`). Fixed by requiring `ratelimiter` >= 1.2.0 ### Changed
- Require aiohttp >= 2.0 for the `async` extra.

4.9 0.12.0

4.9.1 Added

- Request controller can be passed explicitly to methods that take a request class, because some request classes are present in more than one controller.
- Request controller proxy attribute, e.g. `SpaceTrackClient.fileshare.file()`, which is equivalent to `SpaceTrackClient.generic_request('file', controller='fileshare')`.
- `dir(SpaceTrackClient(...))` now includes the request controllers and request classes so it's easier to see what can be called.

4.9.2 Fixed

- `/modeldef` API not queried if no predicates are passed. This allows `spephemeris/download` to be used, which doesn't have a model definition.

4.9.3 Changed

- Calling request class methods uses first request controller that matches. The order is stored in the keys of the `SpaceTrackClient.request_controllers` ordered dict, currently `basicspacedata`, `expandedspacedata`, `fileshare`, `spephemeris`. Any new request controllers will be added to the end, to preserve lookup order. New request classes that would change the order will accompany a major version bump.
- `AsyncSpaceTrackClient` uses requests' CA file for same experience with both clients.

4.10 0.11.1

4.10.1 Fixed

- Bump ratelimiter version to improve rate limiting for `AsyncSpaceTrackClient`

4.10.2 Changed

- Documentation included in source distribution.

4.11 0.11.0

4.11.1 Added

- Some unit tests added for `AsyncSpaceTrackClient`.

4.11.2 Fixed

- `\r\n` to `\n` newline conversion for async chunk iterator.

4.11.3 Changed

- `AsyncSpaceTrackClient` can no longer be imported from the top level `spacetrack` module, since this would cause an error if optional dependency `aiohttp` was not installed. It must be imported from `spacetrack.aio`.

4.12 0.10.0 - 2016-02-04

4.12.1 Fixed

- Compatibility with `file` and `download` request classes for `fileshare` request controller. `upload` request class removed, unable to test.
- Rate limit violation HTTP status code 500 handled during predicate information request.

4.12.2 Changed

- `iter_lines=True` now raises `ValueError` if receiving binary data (currently only possible with `download` request class).
- Removed internal method `_get_predicate_fields`, set comprehension used inline instead.
- `Predicate` class now has a `default` attribute.

4.13 0.9.0 - 2016-01-28

First release.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

Symbols

A

C

G

I

L

N

P

S

U