

---

# SatNOGS Client

*Release 1.8.1+0.g14561ef.dirty*

SatNOGS

Sep 09, 2022



# CONTENTS

|          |                                           |           |
|----------|-------------------------------------------|-----------|
| <b>1</b> | <b>Table of Contents</b>                  | <b>3</b>  |
| 1.1      | User guide . . . . .                      | 3         |
| 1.1.1    | Requirements . . . . .                    | 3         |
| 1.1.2    | Installation . . . . .                    | 3         |
| 1.1.2.1  | Debian . . . . .                          | 3         |
| 1.1.2.2  | SatNOGS Client . . . . .                  | 3         |
| 1.1.3    | Configuration . . . . .                   | 4         |
| 1.1.3.1  | Environment variables . . . . .           | 4         |
| 1.1.4    | Usage . . . . .                           | 21        |
| 1.2      | Developer guide . . . . .                 | 21        |
| 1.2.1    | Installation . . . . .                    | 21        |
| 1.2.2    | Configuration . . . . .                   | 21        |
| 1.2.3    | Code Quality Assurance . . . . .          | 22        |
| 1.2.4    | Testing . . . . .                         | 22        |
| 1.2.4.1  | System testing . . . . .                  | 22        |
| 1.2.5    | Automation . . . . .                      | 22        |
| 1.2.5.1  | Environments . . . . .                    | 22        |
| 1.3      | Modules reference . . . . .               | 23        |
| 1.3.1    | satnogsclient . . . . .                   | 23        |
| 1.3.2    | satnogsclient.locator.locator . . . . .   | 23        |
| 1.3.3    | satnogsclient.observer.observer . . . . . | 23        |
| 1.3.4    | satnogsclient.observer.orbital . . . . .  | 24        |
| 1.3.5    | satnogsclient.artifacts . . . . .         | 24        |
| 1.3.6    | satnogsclient.waterfall . . . . .         | 24        |
| 1.3.7    | satnogsclient.observer.worker . . . . .   | 25        |
| 1.3.8    | satnogsclient.rig . . . . .               | 27        |
| 1.3.9    | satnogsclient.rotator . . . . .           | 27        |
| 1.3.10   | satnogsclient.scheduler.tasks . . . . .   | 29        |
| 1.3.11   | satnogsclient.settings . . . . .          | 29        |
| 1.3.12   | satnogsclient.radio.flowgraphs . . . . .  | 29        |
| <b>2</b> | <b>Indices and tables</b>                 | <b>31</b> |
|          | <b>Python Module Index</b>                | <b>33</b> |
|          | <b>Index</b>                              | <b>35</b> |



SatNOGS Client is the software responsible for automating satellite signal reception for SatNOGS. The main functions of SatNOGS Client is to pull observation jobs from SatNOGS Network and execute them on the a station host (usually a Raspberry Pi). These observation jobs create locally scheduled task which spawn SatNOGS Radio scripts for decoding and demodulating the signals. In addition to that, SatNOGS Client can optionally control a Hamlib compatible rotator to track the satellite. During the execution of an observation task, orbital calculations are performed to compensate for doppler shift and point the rotator to the correct direction. After the completion of an observation, the client gathers the observation artifacts and uploads them to SatNOGS Network.



## TABLE OF CONTENTS

### 1.1 User guide

#### 1.1.1 Requirements

- Python 3.6+
- Hamlib 4.0+ Python bindings
- GPSD 3.22+ Python bindings
- ATLAS 3.10+ library
- HDF5 1.10+ library

#### 1.1.2 Installation

##### 1.1.2.1 Debian

To install the required dependencies in Debian Bullseye run:

```
$ apt-get install libatlas3-base libhdf5-103-1 python3-gps python3-hamlib
```

##### 1.1.2.2 SatNOGS Client

To install SatNOGS Client run:

```
$ pip install satnogs-client
```

This will install a console script called `satnogs-client`.

### 1.1.3 Configuration

Configuration of SatNOGS Client is done through environment variables. The environment variables can also be defined in a file called `.env`, place on the project root directory. The format of each line in `.env` file is `VARIABLE=VALUE`.

#### 1.1.3.1 Environment variables

##### SATNOGS\_API\_TOKEN

**Type**

*string*

**Default**

*None*

**Required**

*Yes*

**Description**

SatNOGS Network API token associated with an account in SatNOGS Network. This token is secret. It can be found in SatNOGS Network user page.

##### SATNOGS\_PRE\_OBSERVATION\_SCRIPT

**Type**

*path*

**Default**

*None*

**Required**

*No*

**Description**

A path to an executable to be executed before an observation job is started. Execution of this script blocks the observation job.

##### SATNOGS\_POST\_OBSERVATION\_SCRIPT

**Type**

*path*

**Default**

*None*

**Required**

*No*

**Description**

A path to an executable to be executed after an observation job has finished. Execution of this script blocks the completion of an observation job.

**SATNOGS\_STATION\_ID****Type***integer***Default***None***Required***Yes***Description**

The ID of the SatNOGS Network station this client will act as. The station must be owned by the user with the defined API token.

**SATNOGS\_STATION\_LAT****Type***float***Default***None***Required***Yes***Description**

Latitude of the station location. Higher precision of this value increases accuracy of Doppler correction while lower precision increases station location privacy.

**SATNOGS\_STATION\_LON****Type***float***Default***None***Required***Yes***Description**

Longitude of the station location. Higher precision of this value increases accuracy of Doppler correction while lower precision increases station location privacy.

**SATNOGS\_STATION\_ELEV****Type***integer***Default***None***Required***Yes*

**Description**

Elevation of the station location. Higher precision of this value increases accuracy of Doppler correction while lower precision increases station location privacy.

**SATNOGS\_GPSD\_CLIENT\_ENABLED**

**Type**

*boolean*

**Default**

*False*

**Required**

*No*

**Description**

Enable SatNOGS Client to connect to a GPSd daemon to pull positional information. The position is queried once, during SatNOGS Client startup.

**SATNOGS\_GPSD\_HOST**

**Type**

*host*

**Default**

*127.0.0.1*

**Required**

*No*

**Description**

Hostname or IP address of GPSd to connect to for pulling positional information.

**SATNOGS\_GPSD\_PORT**

**Type**

*port*

**Default**

*2947*

**Required**

*No*

**Description**

Port of GPSd to connect to for pulling positional information.

**SATNOGS\_GPSD\_TIMEOUT****Type***integer***Default***0***Required***No***Description**

Time to wait until GPSd returns positional information. A value of 0 means to wait indefinitely.

**SATNOGS\_APP\_PATH****Type***path***Default***/tmp/.satnogs***Required***No***Description**

Base path for storing output files.

**SATNOGS\_OUTPUT\_PATH****Type***path***Default***/tmp/.satnogs/data***Required***No***Description**

Path for storing output files.

**SATNOGS\_COMPLETE\_OUTPUT\_PATH****Type***path***Default****Required***No***Description**

Path to move output files once they are completed. Preserving output files is disabled if set to empty.

## SATNOGS\_INCOMPLETE\_OUTPUT\_PATH

**Type**

*path*

**Default**

*/tmp/.satnogs/data/incomplete*

**Required**

*No*

**Description**

Path for moving incomplete output files.

## SATNOGS\_REMOVE\_RAW\_FILES

**Type**

*boolean*

**Default**

*True*

**Required**

*No*

**Description**

Remove raw data files used for generating waterfalls.

## SATNOGS\_KEEP\_ARTIFACTS

**Type**

*boolean*

**Default**

*False*

**Required**

*No*

**Description**

Do not remove artifact files after successfully uploading them.

## SATNOGS\_ARTIFACTS\_OUTPUT\_PATH

**Type**

*path*

**Default**

*/tmp/.satnogs/artifacts*

**Required**

*No*

**Description**

Path for storing artifact files.

## SATNOGS\_VERIFY\_SSL

|                    |                                             |
|--------------------|---------------------------------------------|
| <b>Type</b>        | <i>boolean</i>                              |
| <b>Default</b>     | <i>True</i>                                 |
| <b>Required</b>    | <i>No</i>                                   |
| <b>Description</b> | Verify SSL certificates for HTTPS requests. |

## SATNOGS\_NETWORK\_API\_URL

|                    |                                               |
|--------------------|-----------------------------------------------|
| <b>Type</b>        | <i>url</i>                                    |
| <b>Default</b>     | <code>https://network.satnogs.org/api/</code> |
| <b>Required</b>    | <i>No</i>                                     |
| <b>Description</b> | URL pointing to API of SatNOGS Network.       |

## SATNOGS\_NETWORK\_API\_QUERY\_INTERVAL

|                    |                                                                  |
|--------------------|------------------------------------------------------------------|
| <b>Type</b>        | <i>integer</i>                                                   |
| <b>Default</b>     | <code>60</code>                                                  |
| <b>Required</b>    | <i>No</i>                                                        |
| <b>Description</b> | Interval (in seconds) for pulling jobs form SatNOGS Network API. |

## SATNOGS\_NETWORK\_API\_POST\_INTERVAL

|                    |                                                                            |
|--------------------|----------------------------------------------------------------------------|
| <b>Type</b>        | <i>integer</i>                                                             |
| <b>Default</b>     | <code>180</code>                                                           |
| <b>Required</b>    | <i>No</i>                                                                  |
| <b>Description</b> | Interval (in seconds) for posting observation data to SatNOGS Network API. |

## SATNOGS\_ROT\_ENABLED

**Type**

*boolean*

**Default**

*True*

**Required**

*No*

**Description**

Enable SatNOGS Client to connect to an antenna rotator using the Hamlib library.

## SATNOGS\_ROT\_MODEL

**Type**

*string*

**Default**

ROT\_MODEL\_DUMMY

**Required**

*No*

**Description**

Rotator model to control. This value must be the model string of a Hamlib rotator.

## SATNOGS\_ROT\_BAUD

**Type**

*integer*

**Default**

19200

**Required**

*No*

**Description**

Hamlib rotator serial interface baud rate.

## SATNOGS\_ROT\_PORT

**Type**

*path*

**Default**

/dev/ttyUSB0

**Required**

*No*

**Description**

Path to Hamlib rotator serial port device. The device must be accessible to the user which SatNOGS Client is running.

## SATNOGS\_ROT\_THRESHOLD

**Type***integer***Default**

4

**Required***No***Description**

Azimuth/elevation threshold for moving the rotator. Position changes below this threshold will not cause the rotator to move.

## SATNOGS\_ROT\_FLIP

**Type***boolean***Default***False***Required***No***Description**

Enable rotator flipping during high elevation passes.

## SATNOGS\_ROT\_FLIP\_ANGLE

**Type***integer***Default**

75

**Required***No***Description**

Elevation angle above which the rotator will flip.

## SATNOGS\_RIG\_IP

**Type***host***Default**

127.0.0.1

**Required***No***Description**

Hostname or IP address of Hamlib rotctld.

## SATNOGS\_RIG\_PORT

**Type**  
*integer*

**Default**  
4532

**Required**  
*No*

**Description**  
Hamlib rigctld TCP port.

## SATNOGS\_SOAPY\_RX\_DEVICE

**Type**  
*string*

**Default**  
*None*

**Required**  
*Yes*

**Description**  
SoapySDR device driver to use for RX. This setting must be defined in the form `driver=<name>` where `<name>` is the name of the SoapySDR device driver to use.

It also allows setting different devices for specific frequency ranges.

The format for multiple devices is a space delimited list of frequency range – device pairs:

```
<min. freq.>-<max. freq>:driver=<name> <min. freq.>-<max. freq.
>:driver=<name> ...
```

Where frequencies are in MHz and `<name>` is the name of the SoapySDR device driver to use for the specified frequency range.

Example: two RTL-SDR receivers: one for VHF (135..148 MHz), another for UHF (430..470 MHz).

```
135-148:driver=rtlsdr,serial=1 430-470:driver=rtlsdr,serial=2
```

## SATNOGS\_RX\_SAMP\_RATE

**Type**  
*integer*

**Default**  
*None*

**Required**  
*Yes*

**Description**  
SoapySDR device sample rate. Valid sample rates for attached devices can be queried using `SoapySDRUtil --probe`.

**SATNOGS\_RX\_BANDWIDTH****Type***integer***Default***Flowgraph-defined***Required***No***Description**

SoapySDR device RF bandwidth. This setting configures the RF filter on devices that support it.

**SATNOGS\_DOPPLER\_CORR\_PER\_SEC****Type***integer***Default***Flowgraph-defined***Required***No***Description**

Number of Doppler corrections per second requested by SatNOGS Radio.

**SATNOGS\_LO\_OFFSET****Type***integer***Default***Flowgraph-defined***Required***No***Description**

SoapySDR device local oscillator offset to apply. This setting is used to shift the carrier away from the DC spike.

**SATNOGS\_PPM\_ERROR****Type***float***Default***Flowgraph-defined***Required***No***Description**

SoapySDR device oscillator frequency error correction to apply. This setting is defined in parts per million.

## SATNOGS\_GAIN\_MODE

**Type**

*string*

**Default**

Overall

**Required**

*No*

**Description**

SoapySDR device gain mode. Valid values are: Overall, Specific, Settings Field.

## SATNOGS\_RF\_GAIN

**Type**

*float*

**Default**

*Flowgraph-defined*

**Required**

*No*

**Description**

SoapySDR device overall gain, in dB. Device drivers set individual, device specific gains to approximate linearity on the overall gain.

## SATNOGS\_ANTENNA

**Type**

*string*

**Default**

*None*

**Required**

*Yes*

**Description**

SoapySDR device antenna to use for RX. Valid antennas for attached devices can be queried using `SoapySDRUtil --probe`.

## SATNOGS\_DEV\_ARGS

**Type**

*string*

**Default**

*Flowgraph-defined*

**Required**

*No*

**Description**

SoapySDR device arguments. Valid device arguments for attached devices can be queried using `SoapySDRUtil --probe`.

**SATNOGS\_STREAM\_ARGS****Type**

*string*

**Default**

*Flowgraph-defined*

**Required**

*No*

**Description**

SoapySDR stream arguments. Valid stream arguments for attached devices can be queried using `SoapySDRUtil --probe`.

**SATNOGS\_TUNE\_ARGS****Type**

*string*

**Default**

*Flowgraph-defined*

**Required**

*No*

**Description**

SoapySDR channel tune arguments.

**SATNOGS\_OTHER\_SETTINGS****Type**

*string*

**Default**

*Flowgraph-defined*

**Required**

*No*

**Description**

SoapySDR channel other settings.

## SATNOGS\_DC\_REMOVAL

**Type**  
*boolean*

**Default**  
*Flowgraph-defined*

**Required**  
*No*

**Description**  
SoapySDR device automatic DC offset suppression.

## SATNOGS\_BB\_FREQ

**Type**  
*string*

**Default**  
*Flowgraph-defined*

**Required**  
*No*

**Description**  
SoapySDR device baseband CORDIC frequency for devices that support it.

## ENABLE\_IQ\_DUMP

**Type**  
*boolean*

**Default**  
*False*

**Required**  
*No*

**Description**  
Create I/Q data dumps for every observation. Use this feature with caution. Enabling this setting will store large amount of data on the filesystem.

## IQ\_DUMP\_FILENAME

**Type**  
*path*

**Default**  
*None*

**Required**  
*No*

**Description**  
Path to file for storing I/Q data dumps.

## DISABLE\_DECODED\_DATA

**Type**  
*boolean*

**Default**  
*False*

**Required**  
*No*

**Description**  
Disable output of decoded data.

## UDP\_DUMP\_HOST

**Type**  
*host*

**Default**  
*Flowgraph-defined*

**Required**  
*No*

**Description**  
IP destination of UDP data with Doppler corrected I/Q.

## UDP\_DUMP\_PORT

**Type**  
*port*

**Default**  
57356

**Required**  
*No*

**Description**  
Port for UDP data with Doppler corrected I/Q.

## SATNOGS\_UPLOAD\_AUDIO\_FILES

**Type**  
*boolean*

**Default**  
*True*

**Required**  
*No*

**Description**  
Enable/Disable uploading audio files to SatNOGS network.

## SATNOGS\_UPLOAD\_WATERFALL\_FILES

|                    |                                                         |
|--------------------|---------------------------------------------------------|
| <b>Type</b>        | <i>boolean</i>                                          |
| <b>Default</b>     | <i>True</i>                                             |
| <b>Required</b>    | <i>No</i>                                               |
| <b>Description</b> | Enable/Disable uploading waterfalls to SatNOGS network. |

## SATNOGS\_WATERFALL\_AUTORANGE

|                    |                                                          |
|--------------------|----------------------------------------------------------|
| <b>Type</b>        | <i>boolean</i>                                           |
| <b>Default</b>     | <i>True</i>                                              |
| <b>Required</b>    | <i>No</i>                                                |
| <b>Description</b> | Automatically set power level range of waterfall images. |

## SATNOGS\_WATERFALL\_MIN\_VALUE

|                    |                                          |
|--------------------|------------------------------------------|
| <b>Type</b>        | <i>integer</i>                           |
| <b>Default</b>     | <i>-100</i>                              |
| <b>Required</b>    | <i>No</i>                                |
| <b>Description</b> | Minimum power level of waterfall images. |

## SATNOGS\_WATERFALL\_MAX\_VALUE

|                    |                                          |
|--------------------|------------------------------------------|
| <b>Type</b>        | <i>integer</i>                           |
| <b>Default</b>     | <i>-50</i>                               |
| <b>Required</b>    | <i>No</i>                                |
| <b>Description</b> | Maximum power level of waterfall images. |

## SATNOGS\_ARTIFACTS\_ENABLED

**Type**

*boolean*

**Default**

*False*

**Required**

*No*

**Description**

Enable generation and uploading of HDF5 artifacts files to SatNOGS DB.

## SATNOGS\_ARTIFACTS\_API\_URL

**Type**

*url*

**Default**

`https://db.satnogs.org/api/`

**Required**

*No*

**Description**

URL pointing to API of SatNOGS DB for uploading artifacts.

## SATNOGS\_ARTIFACTS\_API\_POST\_INTERVAL

**Type**

*integer*

**Default**

180

**Required**

*No*

**Description**

Interval (in seconds) for posting artifacts to SatNOGS DB.

## SATNOGS\_ARTIFACTS\_API\_TOKEN

**Type**

*string*

**Default**

*None*

**Required**

*No*

**Description**

SatNOGS DB API token associated with an account in SatNOGS DB. This token is secret. It is used to upload artifacts to SatNOGS DB. It can be found in SatNOGS DB user page.

## SATNOGS\_LOG\_LEVEL

**Type**

*string*

**Default**

WARNING

**Required**

*No*

**Description**

SatNOGS Client logging level. Valid values are:

- CRITICAL
- ERROR
- WARNING
- INFO
- DEBUG

## SATNOGS\_SCHEDULER\_LOG\_LEVEL

**Type**

*string*

**Default**

WARNING

**Required**

*No*

**Description**

SatNOGS Client scheduler logging level. Valid values are:

- CRITICAL
- ERROR
- WARNING
- INFO
- DEBUG

## SENTRY\_DSN

**Type**

*string*

**Default**

d50342fb75aa8f3945e2f846b77a0cdb7c7d2275

**Required**

*No*

**Description**

Sentry Data Source Name used for sending events to application monitoring and error tracking server.

## SENTRY\_ENABLED

**Type***boolean***Default***False***Required***No***Description**

Enable sending events to Sentry application monitoring and error tracking server.

### 1.1.4 Usage

To execute the script, run it on the command line:

```
$ satnogs-client
```

## 1.2 Developer guide

### 1.2.1 Installation

To install the required dependencies in Debian Bullseye run:

```
$ apt-get install libatlas3-base libhdf5-103-1 python3-gps python3-hamlib
```

It is recommended to install the client in a virtualenv. The virtualenv needs to have access to system Python bindings. To create the virtualenv, you can use `virtualenvwrapper`. On the first time, create the virtualenv by running:

```
$ mkvirtualenv --system-site-packages -a . satnogs-client
```

To activate the virtualenv after it is created run:

```
$ workon satnogs-client
```

To install SatNOGS Client for development run in the project root directory:

```
$ pip install -e .[dev]
```

### 1.2.2 Configuration

This project uses `python-dotenv`. Configuration of `satnogsclient/settings.py` can be overridden by setting the respective environment variables or an `.env` file placed on the project root directory. Check [Configuration](#) for a list of all configuration variables.

### 1.2.3 Code Quality Assurance

The following code quality assurance tools are used in this project:

- flake8
- isort
- yapf
- pylint
- robotframework

### 1.2.4 Testing

#### 1.2.4.1 System testing

Robot Framework is used for system testing. `robot/testsuites` contain Robot test cases and suites.

### 1.2.5 Automation

tox is used to automate development tasks. To execute the default list of tasks run:

```
$ tox
```

#### 1.2.5.1 Environments

The following tox environments are available:

- flake8 - Check code for common errors, coding style and complexity
- isort - Check code for correct imports order
- isort-apply - Sort imports
- yapf - Check code for correct formatting
- yapf-apply - Reformat source code
- pylint - Execute static code analysis
- build - Build source and binary distributions
- upload - Upload source and binary distributions to PyPI
- docs - Build documentation
- deps - Check for broken dependencies
- pytest - Execute application tests
- robot-lint - Lint system test cases and suites
- robot-tidy - Reformat system test cases and suites
- robot - Execute system testing

To execute a single environment run:

```
$ tox -e <environment>
```

## 1.3 Modules reference

### 1.3.1 satnogsclient

SatNOGS Client module initialization

`satnogsclient.main()`

Main function

### 1.3.2 satnogsclient.locator.locator

`class satnogsclient.locator.locator.Locator`

`static show_location(gpsd)`

`update_location()`

### 1.3.3 satnogsclient.observer.observer

`class satnogsclient.observer.observer.Observer`

`observe()`

Starts threads for rotrcl and rigctl.

`plot_waterfall(waterfall)`

`poll_gnu_proc_status(flowgraph)`

`remove_waterfall_file()`

`rename_data_file()`

`rename_ogg_file()`

`run_rig()`

`run_rot()`

`setup(observation_id, tle, observation_end, frequency, mode, baud)`

Sets up required internal variables. \* returns True if setup is ok \* returns False if issue is encountered

`satnogsclient.observer.observer.post_and_save_artifacts(artifacts_file, observation_id)`

Accepts a TemporaryFile *artifacts\_file* opened at the beginning of the file, saves it to disk if enabled, then tries to upload it to satnogs-db. Finally closes this file.

#### Parameters

- **artifacts\_file** (*file*) – The Artifact file (opened at position 0).
- **observation\_id** (*str*) – The observation id of the associated observation (for enhanced logging).

`satnogsclient.observer.observer.rx_device_for_frequency(rx_device_spec, frequency)`

`satnogsclient.observer.observer.save_artifacts(artifacts_file, filename)`

Accepts a TemporaryFile *artifacts\_file* opened at the beginning of the file and writes this file to disk. Finally seeks back to the beginning of the file.

NOTE: Does NOT close this file.

**Parameters**

- **artifacts\_file** (*file*) – TemporaryFile, open at position 0
- **filename** (*str*) – Filename of artifacts file

### 1.3.4 `satnogsclient.observer.orbital`

`satnogsclient.observer.orbital.pinpoint(observer_dict, satellite_dict, timestamp=None)`

Provides azimuth and altitude of tracked object.

**args:**

*observer\_dict*: dictionary with details of observation point. *satellite\_dict*: dictionary with details of satellite. *time*: timestamp we want to use for pinpointing the observed object.

**returns:**

Dictionary containing azimuth, altitude and “ok” for error detection.

### 1.3.5 `satnogsclient.artifacts`

`class satnogsclient.artifacts.Artifacts(waterfall, metadata)`

`create()`

### 1.3.6 `satnogsclient.waterfall`

**exception** `satnogsclient.waterfall.EmptyArrayError`

Empty data array exception

**class** `satnogsclient.waterfall.Waterfall(datafile_path)`

Parse waterfall data file

**Parameters**

**datafile\_path** (*str\_array*) – Path to data file

**plot**(*figure\_path*, *vmin=None*, *vmax=None*)

Plot waterfall into a figure

**Parameters**

- **figure\_path** (*str*) – Path of figure file to save
- **vmin** (*int*) – Minimum value range
- **vmax** (*int*) – Maximum value range

`satnogsclient.waterfall._compress_waterfall(waterfall)`

Compress spectra of waterfall

**Parameters**

**waterfall** (*dict*) – Waterfall data

**Returns**

Compressed spectra

**Return type**

dict

`satnogsclient.waterfall._get_waterfall(datafile_path)`

Get waterfall data

**Parameters**

**datafile\_path** (*str\_array*) – Path to data file

**Returns**

Waterfall data including compressed data

**Return type**

dict

`satnogsclient.waterfall._read_waterfall(datafile_path)`

Read waterfall data file

**Parameters**

**datafile\_path** (*str*) – Path to data file

**Raises**

[\*EmptyArrayError\*](#) – Empty waterfall data

**Returns**

Waterfall data

**Return type**

dict

### 1.3.7 satnogsclient.observer.worker

`class satnogsclient.observer.worker.Worker(time_to_stop=None, sleep_time=None)`

Class to facilitate as a worker for rotctl/rigctl.

`_communicate_tracking_info()`

`_observation_end = None`

`_sleep_time = 0.1`

`_stay_alive = False`

`property is_alive`

Returns if tracking loop is alive or not.

`observer_dict = {}`

`satellite_dict = {}`

**send\_to\_socket**(*pin, sock*)

**trackobject**(*observer\_dict, satellite\_dict*)

Sets tracking object. Can also be called while tracking to manipulate observation.

**Parameters**

- **observer\_dict** (*dict*) – Location of the Observer, example: {'lon': 0.0, 'lat': 0.0, 'elev':0.0}
- **satellite\_dict** (*dict*) – TLE of the satellite, example: {'tle0': '', 'tle1': '', 'tle2': ''}

**trackstart**()

Starts the thread that communicates tracking info to remote socket. Stops by calling trackstop()

**trackstop**()

Sets object flag to false and stops the tracking thread.

**class** satnogsclient.observer.worker.**WorkerFreq**(*ip, port, \*\*kwargs*)

**\_communicate\_tracking\_info**()

Runs as a daemon thread, communicating tracking info to remote socket. Uses observer and satellite objects set by trackobject(). Will exit when observation\_end timestamp is reached.

**\_frequency** = None

**send\_to\_socket**(*pin, sock*)

**trackobject**(*frequency, observer\_dict, satellite\_dict*)

Track object

**Parameters**

- **frequency** (*int*) – Frequency of original signal in Hz
- **observer\_dict** (*dict*) – Location of the Observer example: {'lon': 0.0, 'lat': 0.0, 'elev':0.0}
- **satellite\_dict** (*dict*) – TLE of the satellite example: {'tle0': '', 'tle1': '', 'tle2': ''}

**class** satnogsclient.observer.worker.**WorkerTrack**(*port, \*\*kwargs*)

**\_altitude** = None

**\_azimuth** = None

**\_communicate\_tracking\_info**()

Runs as a daemon thread, communicating tracking info to remote socket. Uses observer and satellite objects set by trackobject(). Will exit when observation\_end timestamp is reached.

**\_flip** = False

**\_midpoint** = None

**static find\_midpoint**(*observer\_dict, satellite\_dict, start*)

**static flip\_coordinates**(*azi, alt, timestamp, midpoint*)

**static normalize\_angle**(*num, lower=0, upper=360*)

**send\_to\_socket**(*pin, sock*)

**trackobject(\*args)**

Track object

**Parameters**

**args** – Positional Arguments of parent class Worker

### 1.3.8 satnogsclient.rig

**class** satnogsclient.rig.**Rig**(*model=Hamlib.RIG\_MODEL\_DUMMY, path="", debug=Hamlib.RIG\_DEBUG\_WARN*)

Communicate and interface with rigs

**Parameters**

- **model** – Model of Hamlib rig
- **path** (*str*, *optional*) – Path or address to Hamlib rig device
- **debug** (*int*, *optional*) – Hamlib rig debug level

**close()**

Close Hamlib rig device

**property frequency**

Get rig frequency

**Returns**

Rig frequency

**Return type**

float

**open()**

Open Hamlib rig device

**property vfo**

Get active VFO

**Returns**

Active VFO

**Return type**

int

### 1.3.9 satnogsclient.rotator

**class** satnogsclient.rotator.**Rotator**(*model, baud, port*)

Communicate and interface with rotators

**Parameters**

- **model** (*str*) – Model of rotator e.g. “ROT\_MODEL\_EASYCOMM3” or “ROT\_MODEL\_DUMMY”
- **baud** (*int*, *optional*) – The baud rate of serial communication, e.g. 19200
- **port** (*str*) – The port of the rotator, e.g. “/dev/ttyUSB0”

**close()**

End the communication with rotator

**get\_conf(cmd)**

Return the configuration of a register

**Parameters**

**cmd** (*int*) – Number of the register

**Returns**

Value of register

**Return type**

str

**get\_info()**

Return information about the rotator

**Returns**

Information about the rotator

**Return type**

str

**move(direction, speed)**

Move the rotator with speed (mdeg/s) to specific direction

**Parameters**

- **direction** (*str*) – The direction of movement, e.g. ROT\_MOVE\_UP, ROT\_MOVE\_DOWN, ROT\_MOVE\_LEFT, ROT\_MOVE\_RIGHT
- **speed** (*int*) – The velocity set point in mdeg/s

**open()**

Start the communication with rotator

**park()**

Move the rotator to park position and return the current position

**Returns**

Current position in degrees

**Return type**

None or list[float, float]

**property position**

Return the position in degrees of azimuth and elevation

**Returns**

Position in degrees

**Return type**

list[float, float]

**reset()**

Move the rotator to home position and return the current position

**Returns**

Current position in degrees

**Return type**

None or list[float, float]

**stop()**

Stop the rotator and return the current position

**Returns**

Current position in degrees

**Return type**

None or list[float, float]

**1.3.10 satnogsclient.scheduler.tasks****satnogsclient.scheduler.tasks.get\_jobs()**

Query SatNOGS Network API to GET jobs.

**satnogsclient.scheduler.tasks.keep\_or\_remove\_file(filename)****satnogsclient.scheduler.tasks.post\_data()**

PUT observation data back to Network API.

**satnogsclient.scheduler.tasks.spawn\_observer(\*\*kwargs)****satnogsclient.scheduler.tasks.status\_listener()****satnogsclient.scheduler.tasks.upload\_observation\_data(observation\_id, observation, fil)**

Upload observation data to SatNOGS Network API.

**1.3.11 satnogsclient.settings**

SatNOGS Client settings file

**satnogsclient.settings.\_cast\_or\_none(func, value)****satnogsclient.settings.validate(logger)**

Validate the provided settings: - Check for the existence of all required variables - Validate format of the provided value for some required variables

Since this module has to be loaded before the logger has been initialized, this method requires a configured logger to be passed.

Arguments: logger – the output logger

**1.3.12 satnogsclient.radio.flowgraphs****class satnogsclient.radio.flowgraphs.Flowgraph(device, sampling\_rate, frequency, mode, baud, output\_data)**

Execute SatNOGS Flowgraphs

**Parameters**

- **device** (*str*) – SoapySDR device
- **sampling\_rate** (*int*) – Sampling rate
- **frequency** (*int*) – RX frequency
- **mode** (*str*) – Mode of operation

- **baud** (*int*) – Baud rate or WPM
- **output\_data** (*dict*) – Dictionary of output data

**property enabled**

Get flowgraph running status

**Returns**

Flowgraph running status

**Return type**

bool

**property info**

Get information and parameters of flowgraph and radio

**Returns**

Information about flowgraph and radio

**Return type**

dict

## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### S

- `satnogsclient`, [23](#)
- `satnogsclient.artifacts`, [24](#)
- `satnogsclient.locator.locator`, [23](#)
- `satnogsclient.observer.observer`, [23](#)
- `satnogsclient.observer.orbital`, [24](#)
- `satnogsclient.observer.worker`, [25](#)
- `satnogsclient.radio.flowgraphs`, [29](#)
- `satnogsclient.rig`, [27](#)
- `satnogsclient.rotator`, [27](#)
- `satnogsclient.scheduler.tasks`, [29](#)
- `satnogsclient.settings`, [29](#)
- `satnogsclient.waterfall`, [24](#)



## Symbols

`_altitude` (*satnogsclient.observer.worker.WorkerTrack attribute*), 26

`_azimuth` (*satnogsclient.observer.worker.WorkerTrack attribute*), 26

`_cast_or_none()` (in module *satnogsclient.settings*), 29

`_communicate_tracking_info()` (*satnogsclient.observer.worker.Worker method*), 25

`_communicate_tracking_info()` (*satnogsclient.observer.worker.WorkerFreq method*), 26

`_communicate_tracking_info()` (*satnogsclient.observer.worker.WorkerTrack method*), 26

`_compress_waterfall()` (in module *satnogsclient.waterfall*), 24

`_flip` (*satnogsclient.observer.worker.WorkerTrack attribute*), 26

`_frequency` (*satnogsclient.observer.worker.WorkerFreq attribute*), 26

`_get_waterfall()` (in module *satnogsclient.waterfall*), 25

`_midpoint` (*satnogsclient.observer.worker.WorkerTrack attribute*), 26

`_observation_end` (*satnogsclient.observer.worker.WorkerTrack attribute*), 25

`_read_waterfall()` (in module *satnogsclient.waterfall*), 25

`_sleep_time` (*satnogsclient.observer.worker.Worker attribute*), 25

`_stay_alive` (*satnogsclient.observer.worker.Worker attribute*), 25

## A

Artifacts (class in *satnogsclient.artifacts*), 24

## C

`close()` (*satnogsclient.rig.Rig method*), 27

`close()` (*satnogsclient.rotator.Rotator method*), 27

`create()` (*satnogsclient.artifacts.Artifacts method*), 24

## E

EmptyArrayError, 24

`enabled` (*satnogsclient.radio.flowgraphs.Flowgraph property*), 30

## F

`find_midpoint()` (*satnogsclient.observer.worker.WorkerTrack static method*), 26

`flip_coordinates()` (*satnogsclient.observer.worker.WorkerTrack static method*), 26

Flowgraph (class in *satnogsclient.radio.flowgraphs*), 29

`frequency` (*satnogsclient.rig.Rig property*), 27

## G

`get_conf()` (*satnogsclient.rotator.Rotator method*), 28

`get_info()` (*satnogsclient.rotator.Rotator method*), 28

`get_jobs()` (in module *satnogsclient.scheduler.tasks*), 29

## I

`info` (*satnogsclient.radio.flowgraphs.Flowgraph property*), 30

`is_alive` (*satnogsclient.observer.worker.Worker property*), 25

## K

`keep_or_remove_file()` (in module *satnogsclient.scheduler.tasks*), 29

## L

Locator (class in *satnogsclient.locator.locator*), 23

## M

`main()` (in module *satnogsclient*), 23

module

- satnogsclient*, 23
- satnogsclient.artifacts*, 24
- satnogsclient.locator.locator*, 23
- satnogsclient.observer.observer*, 23
- satnogsclient.observer.orbital*, 24
- satnogsclient.observer.worker*, 25

satnogsclient.radio.flowgraphs, 29  
satnogsclient.rig, 27  
satnogsclient.rotator, 27  
satnogsclient.scheduler.tasks, 29  
satnogsclient.settings, 29  
satnogsclient.waterfall, 24  
move() (satnogsclient.rotator.Rotator method), 28

## N

normalize\_angle() (satnogsclient.observer.worker.WorkerTrack static method), 26

## O

observe() (satnogsclient.observer.observer.Observer method), 23  
Observer (class in satnogsclient.observer.observer), 23  
observer\_dict (satnogsclient.observer.worker.Worker attribute), 25  
open() (satnogsclient.rig.Rig method), 27  
open() (satnogsclient.rotator.Rotator method), 28

## P

park() (satnogsclient.rotator.Rotator method), 28  
pinpoint() (in module satnogsclient.observer.orbital), 24  
plot() (satnogsclient.waterfall.Waterfall method), 24  
plot\_waterfall() (satnogsclient.observer.observer.Observer method), 23  
poll\_gnu\_proc\_status() (satnogsclient.observer.observer.Observer method), 23  
position (satnogsclient.rotator.Rotator property), 28  
post\_and\_save\_artifacts() (in module satnogsclient.observer.observer), 23  
post\_data() (in module satnogsclient.scheduler.tasks), 29

## R

remove\_waterfall\_file() (satnogsclient.observer.observer.Observer method), 23  
rename\_data\_file() (satnogsclient.observer.observer.Observer method), 23  
rename\_ogg\_file() (satnogsclient.observer.observer.Observer method), 23  
reset() (satnogsclient.rotator.Rotator method), 28  
Rig (class in satnogsclient.rig), 27  
Rotator (class in satnogsclient.rotator), 27  
run\_rig() (satnogsclient.observer.observer.Observer method), 23  
run\_rot() (satnogsclient.observer.observer.Observer method), 23  
rx\_device\_for\_frequency() (in module satnogsclient.observer.observer), 23

## S

satellite\_dict (satnogsclient.observer.worker.Worker attribute), 25  
satnogsclient module, 23  
satnogsclient.artifacts module, 24  
satnogsclient.locator.locator module, 23  
satnogsclient.observer.observer module, 23  
satnogsclient.observer.orbital module, 24  
satnogsclient.observer.worker module, 25  
satnogsclient.radio.flowgraphs module, 29  
satnogsclient.rig module, 27  
satnogsclient.rotator module, 27  
satnogsclient.scheduler.tasks module, 29  
satnogsclient.settings module, 29  
satnogsclient.waterfall module, 24  
save\_artifacts() (in module satnogsclient.observer.observer), 24  
send\_to\_socket() (satnogsclient.observer.worker.Worker method), 25  
send\_to\_socket() (satnogsclient.observer.worker.WorkerFreq method), 26  
send\_to\_socket() (satnogsclient.observer.worker.WorkerTrack method), 26  
setup() (satnogsclient.observer.observer.Observer method), 23  
show\_location() (satnogsclient.locator.locator.Locator static method), 23  
spawn\_observer() (in module satnogsclient.scheduler.tasks), 29  
status\_listener() (in module satnogsclient.scheduler.tasks), 29  
stop() (satnogsclient.rotator.Rotator method), 29

## T

trackobject() (satnogsclient.observer.worker.Worker method), 26  
trackobject() (satnogsclient.observer.worker.WorkerFreq method), 26  
trackobject() (satnogsclient.observer.worker.WorkerTrack method), 26  
trackstart() (satnogsclient.observer.worker.Worker method), 26

`trackstop()` (*satnogsclient.observer.worker.Worker*  
*method*), [26](#)

## U

`update_location()` (*satnogsclient.locator.locator.Locator*  
*method*), [23](#)

`upload_observation_data()` (*in module*  
*satnogsclient.scheduler.tasks*), [29](#)

## V

`validate()` (*in module satnogsclient.settings*), [29](#)

`vfo` (*satnogsclient.rig.Rig* property), [27](#)

## W

`Waterfall` (*class in satnogsclient.waterfall*), [24](#)

`Worker` (*class in satnogsclient.observer.worker*), [25](#)

`WorkerFreq` (*class in satnogsclient.observer.worker*), [26](#)

`WorkerTrack` (*class in satnogsclient.observer.worker*),  
[26](#)