StreamStats Documentation

Release 0.1.5

Maxwell B. Joseph

Mar 16, 2022

CONTENTS:

1 StreamStats 1
- 1.1 Features 1
- 1.2 Credits 1

2 Installation 3
- 2.1 Stable release 3
- 2.2 From sources 3

3 Usage 5
- 3.1 Finding the watershed that contains a point 5

4 streamstats 7
- 4.1 streamstats package 7

5 Contributing 9
- 5.1 Types of Contributions 9
- 5.2 Get Started! 10
- 5.3 Pull Request Guidelines 11
- 5.4 Tips 11
- 5.5 Deploying 11

6 Credits 13
- 6.1 Development Lead 13
- 6.2 Contributors 13

7 History 15
- 7.1 0.1.3 15
- 7.2 0.1.0 (2018-10-22) 15

8 Indices and tables 17

Python Module Index 19

Index 21

CHAPTER

ONE

STREAMSTATS

Python package for interfacing with the USGS StreamStats API.

- Free software: MIT license
- Documentation: https://streamstats-python.readthedocs.io/en/latest/

1.1 Features

- Get the GeoJSON of the watershed containing a spatial point in the U.S.

1.2 Credits

This package was created with Cookiecutter.

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install StreamStats, run this command in your terminal:

```
$ pip install streamstats
```

This is the preferred method to install StreamStats, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for StreamStats can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/earthlab/streamstats
```

Or download the tarball:

```
$ curl  -OL https://codeload.github.com/earthlab/streamstats/legacy.tar.gz/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER

THREE

USAGE

3.1 Finding the watershed that contains a point

Given a spatial point in the U.S. defined by a lat/lon location and a state, the *Watershed* class can be used to find the watershed that contains that point using the USGS StreamStats API.

```
>>> from streamstats import Watershed
>>> Watershed(lat=43.939, lon=-74.524)
Watershed object with HUC8: 04150305, containing lat/lon: (43.939, -74.524)
```

CHAPTER

FOUR

STREAMSTATS

4.1 streamstats package

4.1.1 Submodules

4.1.2 streamstats.utils module

Utility functions for streamstats.

`streamstats.utils.`**`find_address`**(*lat*, *lon*)

Find the address associated with a lat/lon pair.

Parameters

- **`lat`** (*`float`*) – Latitude of point in decimal degrees
- **`lon`** (*`float`*) – Longitude of point in decimal degrees

Return type dictionary containing address data

`streamstats.utils.`**`find_state`**(*address*)

Find the U.S. state that contains an address

Parameters **`address`** (*`dict`*) – An address found by `find_address`

Return type string of the state code (e.g., "CO" for Colorado)

`streamstats.utils.`**`requests_retry_session`**(*retries=3*, *backoff=0.3*, *status_forcelist=(500, 502, 504)*)

Make a session that backs off automatically.

Parameters

- **`retries`** (*`int`*) – Number of times to retry a request
- **`backoff`** (*`float`*) – Interval: {backoff} * (2^({number of total retries} - 1))
- **`status_forcelist`** (*`tuple of ints`*) – Status codes that prompt a retry

4.1.3 streamstats.watershed module

Functionality for finding watershed information for specific locations.

`class` `streamstats.watershed.`**`Watershed`**(*lat*, *lon*)

Bases: `object`

Watershed covering a spatial region, with associated information.

The USGS StreamStats API is built around watersheds as organizational units. Watersheds in the 50 U.S. states can be found using lat/lon lookups, along with information about the watershed including its HUC code and a GeoJSON representation of the polygon of a watershed. Basin characteristics can also be extracted from watersheds.

`base_url = 'https://streamstats.usgs.gov/streamstatsservices/'`

`property boundary`
Return the full watershed GeoJSON as a dictionary.

:rtype dict containing GeoJSON watershed boundary

`property characteristics`
List the available watershed characteristics.

Details about these characteristics can be found in the StreamStats docs: https://streamstatsags.cr.usgs.gov/ss_defs/basin_char_defs.aspx

:rtype OrderedDict with characteristic codes and descriptions

`get_characteristic`(*code=None*)
Retrieve a specified watershed characteristic

Parameters **`code`** (*`string`*) – Watershed characteristic code to extract.

get_characteristic() requires a characteristic code as an argument. Valid codes can be seen as keys in the dictionary returned by the characteristics() method.

:rtype dict containing specified characteristic’s data and metadata

`property huc`
Find the Hydrologic Unit Code (HUC) of the watershed.

4.1.4 Module contents

Top-level package for StreamStats.

CHAPTER

FIVE

CONTRIBUTING

We welcome and greatly appreciate contributions to streamstats! Every bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/earthlab/streamstats/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

StreamStats could always use more documentation, whether as part of the official StreamStats docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/earthlab/streamstats/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up StreamStats for local development.

1. Fork the *streamstats* repo on GitHub.
2. Clone your fork locally:

   ```
   $ git clone git@github.com:your_name_here/streamstats.git
   ```

3. Install your local copy into a new environment.

If you have virtualenvwrapper installed:

```
$ mkvirtualenv streamstats
```

If you are a conda user:

```
$ conda create -n streamstats python=3
$ conda activate streamstats
```

Then install StreamStats:

```
$ cd streamstats/
$ pip install -r requirements.txt
$ pip install -r requirements_dev.txt
$ pip install -e .
```

4. Create a branch for local development:

   ```
   $ git checkout -b name-of-your-bugfix-or-feature
   ```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

   ```
   $ pytest
   $ tox
   ```

6. Commit your changes and push your branch to GitHub:

   ```
   $ git add .
   $ git commit -m "Your detailed description of your changes."
   $ git push origin name-of-your-bugfix-or-feature
   ```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/earthlab/streamstats/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ py.test tests.test_streamstats
```

5.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER SIX

CREDITS

6.1 Development Lead

- Maxwell B. Joseph <maxwell.b.joseph@colorado.edu>

6.2 Contributors

- Scott Eilerman
- Leah Wasser
- Jeremy Diaz
- Nate Mietkiewicz

CHAPTER SEVEN

HISTORY

7.1 0.1.3

- remove get_flow_stats() which is down server side

7.2 0.1.0 (2018-10-22)

- First release on PyPI.

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

S

streamstats, 8
streamstats.utils, 7
streamstats.watershed, 7

INDEX

B

base_url (*streamstats.watershed.Watershed attribute*), 8

boundary() (*streamstats.watershed.Watershed property*), 8

C

characteristics() (*streamstats.watershed.Watershed property*), 8

F

find_address() (*in module streamstats.utils*), 7

find_state() (*in module streamstats.utils*), 7

G

get_characteristic() (*streamstats.watershed.Watershed method*), 8

H

huc() (*streamstats.watershed.Watershed property*), 8

R

requests_retry_session() (*in module streamstats.utils*), 7

S

streamstats (*module*), 8

streamstats.utils (*module*), 7

streamstats.watershed (*module*), 7

W

Watershed (*class in streamstats.watershed*), 7