requests-ratelimit-adapter Documentation

Release 0.2.0

Max Dymond

Jan 06, 2019

Contents:

1 requests-ratelimit-adapter **1**
- 1.1 Example . . . 1
- 1.2 Credits . . . 2

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

3 Usage **5**

4 Contributing **7**
- 4.1 Types of Contributions . . . 7
- 4.2 Get Started! . . . 8
- 4.3 Pull Request Guidelines . . . 9
- 4.4 Tips . . . 9
- 4.5 Deploying . . . 9

5 Credits **11**
- 5.1 Development Lead . . . 11
- 5.2 Contributors . . . 11

6 Changelog **13**
- 6.1 Unreleased . . . 13
- 6.2 0.2.0 - 2019-01-06 . . . 13
- 6.3 0.1.2 - 2018-12-24 . . . 13
- 6.4 0.1.1 - 2018-12-24 . . . 13
- 6.5 0.1.0 - 2018-12-24 . . . 14

7 Indices and tables **15**

CHAPTER 1

requests-ratelimit-adapter

A ratelimiting Session adapter for requests.

- Free software: MIT license
- Documentation: https://requests-ratelimit-adapter.readthedocs.io.

1.1 Example

```
>>> from requests_ratelimit_adapter import HTTPRateLimitAdapter
>>> import requests
>>> import time

>>> # Create a rate limiting adapter
>>> rate_limiter = HTTPRateLimitAdapter(calls=1, period=1)

>>> s = requests.Session()
>>> s.mount("https://", rate_limiter)

>>> # This first request will start the period.
>>> r = s.get("https://httpbin.org/get")
>>> r.raise_for_status()
>>> time1 = time.time()

>>> # This second request will wait 1 second before executing.
>>> r2 = s.get("https://httpbin.org/get")
>>> r2.raise_for_status()
>>> time2 = time.time()

>>> # For this example, verify the timestamps are more than a second apart.
>>> assert time2 >= time1 + 1
```

1.2 Credits

This package was created with Cookiecutter and the cmeister2/cookiecutter-pypackage project template.

CHAPTER 2

Installation

2.1 Stable release

To install requests-ratelimit-adapter, run this command in your terminal:

```
$ pip install requests_ratelimit_adapter
```

This is the preferred method to install requests-ratelimit-adapter, 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 requests-ratelimit-adapter can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/cmeister2/requests_ratelimit_adapter
```

Or download the tarball:

```
$ curl  -OL https://github.com/cmeister2/requests_ratelimit_adapter/tarball/master
```

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

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

CHAPTER 3

Usage

To use requests-ratelimit-adapter in a project:

```
import requests_ratelimit_adapter
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/cmeister2/requests_ratelimit_adapter/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.

4.1.2 Fix Bugs

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

4.1.3 Implement Features

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

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/cmeister2/requests_ratelimit_adapter/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 :)

4.2 Get Started!

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

1. Fork the `requests_ratelimit_adapter` repo on GitHub.
2. Clone your fork locally:

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

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv requests_ratelimit_adapter
$ cd requests_ratelimit_adapter/
$ python setup.py develop
```

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:

```
$ flake8 requests_ratelimit_adapter tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

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.

4.3 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.md.
3. The pull request should work for Python 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/cmeister2/requests_ratelimit_adapter/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests.test_requests_ratelimit_adapter
```

4.5 Deploying

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

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

Travis will then deploy to PyPI if tests pass.

CHAPTER 5

Credits

5.1 Development Lead

- Max Dymond <cmeister2@gmail.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

Changelog

All notable changes to this project will be documented in this file.

The format is based on Keep a Changelog, and this project adheres to Semantic Versioning.

6.1 Unreleased

6.1.1 Changed

6.2 0.2.0 - 2019-01-06

6.2.1 Added

- Wrap the HTTPAdapter code from `requests` so there's less boilerplate.

6.3 0.1.2 - 2018-12-24

6.3.1 Changed

- Use the *correct* travis password for deploying to PyPI.

6.4 0.1.1 - 2018-12-24

6.4.1 Added

- Add travis password for deploying to PyPI.

6.5 0.1.0 - 2018-12-24

6.5.1 Added

- Initial commit of code.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search