webhooks Documentation

Release 0.4.1

Daniel Greenfeld

Mar 05, 2018

Contents

Contents:

CHAPTER 1

webhooks

Python + Webhooks Made Easy

- Free software: BSD license
- Documentation: http://webhooks.rtfd.org.

WARNING This project is in a beta state. It's still undergoing some changes and documentation is in-progress.

1.1 Python Versions

Currently works in:

- Python 2.7
- Python 3.3

1.2 Existing Features

- Easy to integrate into any package or project
- Comes with several built-in senders for synchronous webhooks.
- Comes with a RedisQ-powered asynchronous webhook.
- Extendable functionality through the use of custom senders and hash functions.

1.3 Planned Features

- Comes with many built-in senders for synchronous and asynchronous webhooks.
- Special functions for combining multiple sends of identical payloads going to one target into one.

- Follows http://resthooks.org patterns
- Great documentation
- Compatibility with PyPy

1.4 Usage

Follow these easy steps:

1. Import the `webhook` decorator.
2. Define a function that returns a JSON-serializable dictionary or iterable.
3. Add the `webhook` decorator and pass in a `sender_callable`.
4. Define timeout, any custom headers such as authentication, signing_secret, and encoding (application/json|application/x-www-form-urlencoded)
5. Call the function!

Synchronous example (async examples to come soon):

```
>>> from webhooks import webhook
>>> from webhooks.senders import targeted

>>> @webhook(sender_callable=targeted.sender)
>>> def basic(wife, husband, url, encoding, timeout, custom_headers, signing_secret):
>>>     return {"husband": husband, "wife": wife}

>>> r = basic("Audrey Roy Greenfeld", "Daniel Roy Greenfeld", url="http://httpbin.org/
↪post", encoding="application/json", \
>>>     timeout=10, custom_headers = {"Basic" : "dXNlcjpzdXBlcnNlY3JldA=="}, signing_
↪secret="secret1")
>>> import pprint
>>> pprint.pprint(r)
{'attempt': 1,
 'error': None,
 'hash': '9d930cc754004d5790869fdfb6064f62',
 'husband': 'Daniel Roy Greenfeld',
 'post_attributes': {'headers': {'Basic': 'dXNlcjpzdXBlcnNlY3JldA==',
                                 'x-hub-signature':
↪'sha256=e67a669f944fe752f9d9da15c5bcb4d332fceb4940ab512090e124c52c44cfa5'},
                     'json': '{"hash": "9d930cc754004d5790869fdfb6064f62", "husband":
↪"Daniel Roy Greenfeld", "wife": "Audrey Roy Greenfeld"}',
                     'timeout': 10},
 'response': '{\n  "args": {}, \n  "data": "\\"{\\\\\\"hash\\\\\\": \\\\\\
↪"9d930cc754004d5790869fdfb6064f62\\\\\\", \\\\\\"husband\\\\\\": \\\\\\"Daniel Roy␣
↪Greenfeld\\\\\\", \\\\\\"wife\\\\\\": \\\\\\"Audrey Roy Greenfeld\\\\\\"}\\"", \n
↪"files": {}, \n  "form": {}, \n  "headers": {\n    "Accept": "*/*", \n    "Accept-
↪Encoding": "gzip, deflate", \n    "Basic": "dXNlcjpzdXBlcnNlY3JldA==", \n
↪"Connection": "close", \n    "Content-Length": "125", \n    "Content-Type":
↪"application/json", \n    "Host": "httpbin.org", \n    "User-Agent": "python-
↪requests/2.18.4", \n    "X-Hub-Signature":
↪"sha256=e67a669f944fe752f9d9da15c5bcb4d332fceb4940ab512090e124c52c44cfa5"\n  }, \n
↪"json": "{\\"hash\\": \\"9d930cc754004d5790869fdfb6064f62\\", \\"husband\\": \\
↪"Daniel Roy Greenfeld\\", \\"wife\\": \\"Audrey Roy Greenfeld\\"}", \n  "origin":
↪"38.104.237.126", \n  "url": "http://httpbin.org/post"\n}\n',
 'status_code': 200,
```

```
'success': True,
'wife': 'Audrey Roy Greenfeld'}
```

1.5 Projects Powered by Webhooks

- https://github.com/pydanny/dj-webhooks

CHAPTER 2

Installation

At the command line:

```
$ easy_install webhooks
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv webhooks
$ pip install webhooks
```

CHAPTER 3

Usage

To use webhooks in a project:

```
import webhooks
```

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/pydanny/webhooks/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" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

webhooks could always use more documentation, whether as part of the official webhooks 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/pydanny/webhooks/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 *webhooks* for local development.

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

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

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

```
$ mkvirtualenv webhooks
$ cd webhooks/
$ python setup.py develop
$ pip install -r dev-requirements.txt
```

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 webhooks tests
$ python setup.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.rst.
3. The pull request should work for Python 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/pydanny/webhooks/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_webhooks
```

CHAPTER 5

Credits

5.1 Development Lead

- Daniel Greenfeld / @pydanny

5.2 Contributors

- Audrey Roy / @audreyr
- Noah Haibach / @noahhai

CHAPTER 6

History

6.1 0.5.0 (2018-02-XY)

- Added encoding, header, and signature
- Improved tests
- Better RQ support

6.2 0.4.2 (2014-05-22)

- Convert python-requests bytes to string when using Python 3

6.3 0.4.1 (2014-05-22)

- Replaced *json262* with *standardjson* package.

6.4 0.4.0 (2014-05-20)

- Replaced *utils.encoders* with *json262* package.
- utf-8 encoding everywhere
- Add *from '__future__ import absolute_import* everywhere.

6.5 0.3.2 (2014-05-17)

- Brought in simplified *cached_property* decorator

6.6 0.3.1 (2014-05-15)

- Added more Senderable attributes to make it easier to track what's going on.
- Added the missing webhooks.sender package to setup.py.

6.7 0.3.0 (2014-05-14)

- Added extensible Senderable class to expedite creating new senders.
- Added async_redis sender.
- Added travis-ci.

6.8 0.2.0 (2014-05-13)

- Added functioning hook decorator.
- Ramped up test coverage.
- Hash functions placed in their own module.
- Cleaned up JSON encoder thanks to Audrey Roy Greenfeld!

6.9 0.1.0 (2014-05-07)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search