octokit.py

Release 0.1.0

Feb 03, 2019

Contents

CHAPTER 1

Overview

tests	
package	
docs	

Python client for GitHub API

1.1 Installation

requires python 3.5+

Yes that is opinionated. Python 2 is near the end of the life and this is a new project.

Note octokit and octokit.py were already taken in the cheese shop

```
pip install octokitpy
```

1.2 Documentation

https://octokitpy.readthedocs.io/en/latest/

1.2.1 Examples

REST API:

```
from octokit import Octokit
repos = Octokit().repos.get_for_org(org='octokit', type='public')
# Make an unauthenticated request for the public repositories of the octokit␣
↪organization
```

Webhooks:

```
from octokit import webhook
webhook.verify(headers, payload, secret, events=['push'])
```

`octokit.py` provides a function to verify webhooks sent to your application.

headers dictionary of request headers

payload string; payload of request

secret string; secret provided to GitHub to sign webhook

events list; events that you want to receive

verify_user_agent boolean; whether or not you want to verify the user agent string of the request

return_app_id boolean; whether or not you want to return the app id from the ping event for GitHub applications. This will only return the `id` if the event is the `ping` event. Otherwise the return value will be boolean.

1.2.2 Authentication

Instatiate a client with the authentication scheme and credentials that you want to use.

basic:

```
octokit = Octokit(auth='basic', username='myuser', password='mypassword')
octokit.repos.get_for_org(org='octokit', type='private')
```

token:

```
response = Octokit(auth='token', token='yak').authorization.get(id=100)
```

app:

```
octokit = Octokit(auth='app', app_id='42', private_key=private_key)
```

app installation:

```
octokit = Octokit(auth='installation', app_id='42', private_key=private_key)
```

For applications provide the application id either from the ping webhook or the application's page on GitHub. The `private_key` is a string of your private key provided for the application. The `app` scheme will use the application id and private key to get a token for the first installation id of the application.

1.2.3 API Schema/Routes/Specifications

One can instantiate the `Octokit` with `routes=specification` where the `specification` is one of `api.github.com`, `ghe-2.15`, etc.

1.3 TODOs

1.3.1 GitHub APIs

```
[-] REST (see best practices, integration tests, and errors)

[ ] GraphQL client

[x] GitHub Apps

[ ] OAuth Apps

[x] Webhooks
```

1.3.2 Data

The `octokit` client based on the available route data and webhook data

```
[x] Periodically, check if ``routes.json`` has changed and if so fetch and open a PR␣
↪for it to be merged

[ ] Periodically, check if ``webhook-names.json`` has changed and if so fetch and␣
↪open a PR for it to be merged
```

1.3.3 Tests

```
[x] unit tests

[ ] integration tests - need fixtures to assert against

[ ] coverage uploaded to code climate -- not sure why it is not working
```

1.3.4 Errors

```
[ ] Raise :code:`OctokitValidationError` for param validation error

[ ] Raise :code:`OctokitAuthenticationError` for auth error

[ ] Raise :code:`OctokitRateLimitError` for rate limiting errors
```

1.3.5 Best Practices

```
[ ] throttling

[ ] handles rate limiting

[x] pagination
```

1.3.6 Documentation

```
[ ] Auto generated documentation
```

1.3.7 Deployment

```
[x] Deploy wheels
[ ] Make GitHub releases work
```

Check box guide

```
[ ] Incomplete

[-] Partially completed

[x] Completed
```

1.4 Development

To run the all tests run:

```
tox
```

1.5 Contributing

Pull requests are very welcome!

Please see CONTRIBUTING.md for more information.

1.6 Credits

Package based on cookiecutter-pylibrary

1.7 License

MIT

CHAPTER 2

Installation

At the command line:

```
pip install octokitpy
```

CHAPTER 3

Usage

To use octokit.py in a project:

```
import octokit
```

3.1 Chaining requests

```
issue = Octokit().issues.edit(owner='testUser', repo='testRepo', number=1, state=
↪'closed')
# If the previous request had a required url attribute, the next request will use the␣
↪previous url attribute
# This does not apply attributes that are part of the body of the request on post,␣
↪patch, etc.
issue.pull_requests.create(head='branch', base='master', title='Title')
# Previous attributes can be overridden
issue.pull_requests.create(owner='differentOwner', head='branch', base='master',␣
↪title='Title')
```

3.2 Responses

Responses are the Octokit instance with state in `json` and `response`. `json` is the result of the Requests `response.json()`. `response` is the json as a python object.

3.3 octokit.json

```
issue = Octokit().issues.get(owner='testUser', repo='testRepo', number=1)
issue.json['title']  # Title of issue
```

3.4 octokit.response

```
issue = Octokit().issues.get(owner='testUser', repo='testRepo', number=1)
issue.response.title  # Title of issue
```

CHAPTER 4

Reference

4.1 octokit.py

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps.

5.1 Bug reports

When 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.

5.2 Documentation improvements

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

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/khornberg/octokit.py/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 code contributions are welcome :)

5.4 Development

To set up *octokit.py* for local development:

1. Fork octokit.py (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/octokit.py.git
```

3. Create a branch for local development:

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

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. Commit your changes using the commit message guide 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
```

6. Submit a pull request through the GitHub website.

5.4.1 Commit Message Guidelines

Prefix messages with one of the prefixes below followed by a colon:

Example message:

```
Style: yapf
```

Bug Fix A fix for a bug

Feature Something that did not previously exist

Enhancement Something that previously existed, but now works slightly differently in some way

Doc Documentation

Version A new (semver) version number

Dependency Updating the dependencies Updating 3rd party APIs ect

Refactor Improvements to code with no modification of external behavior Include Performance Enhancements

Test New tests or altering old tests without changing any production code Helper code intended to assist ONLY with test creation

Style Linting violations, code formatting, etc

Peripheral Updates to builds, deploys, etc

5.4.2 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add yourself to `AUTHORS.rst`.

5.4.3 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 6

Authors

- Kyle Hornberg - https://khornberg.github.io

CHAPTER 7

Changelog

7.1 0.1.0 (?)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search