python-goodreads Documentation

Release 0.1.3

Paul Shannon

October 20, 2015

Contents

Contents:

No Longer Maintained

Please see: https://github.com/sefakilic/goodreads for a much more fully implemented wrapper.

If you are interested in *this* code base, significant work was put into an improved implementation but is still unfinished: https://github.com/paulshannon/python-goodreads/tree/feature/rewrite

Goodreads

A Python wrapper for the goodreads.com api

- Documentation: http://python-goodreads.rtfd.org/
- Code: https://github.com/paulshannon/python-goodreads
- Free software: BSD license

2.1 Features

- Coming soon

Installation

At the command line:

```
$ easy_install goodreads
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv goodreads
$ pip install goodreads
```

Usage

To use the Goodreads API you need to acquire a Developer Key and Developer Secret from Goodreads.

To use Goodreads in a project:

```python
import goodreads
```

Examples

User Authorization:

```python
from config import DEVELOPER_KEY, DEVELOPER_SECRET
from goodreads import Goodreads

g = Goodreads(DEVELOPER_KEY, DEVELOPER_SECRET)

authorize_url = g.oauth_authorize_url()

print 'Please authorize at: %s' % authorize_url
accepted = 'n'
while accepted.lower() == 'n':
    # you need to access the authorize_link via a browser,
    # and proceed to manually authorize the consumer
    accepted = raw_input('Have you authorized me? (y/n) ')

token = g.oauth_retrieve_token()
print 'You need to save key: \'%s\' and secret: \'%s\'' % (token.key, token.secret)
```

Author:

```python
from config import DEVELOPER_KEY, DEVELOPER_SECRET, AUTHORS
from goodreads import Goodreads
import pprint

author_id = AUTHORS[0]['id']

g = Goodreads(DEVELOPER_KEY, DEVELOPER_SECRET)

g.author_show(author_id)

for book in g.author_books(author_id):
    pprint.pprint(book)
```

User:

```python
from config import DEVELOPER_KEY, DEVELOPER_SECRET, AUTH_USER, NON_AUTH_USER, SECONDARY_AUTH_USER
from goodreads import Goodreads

g = Goodreads(DEVELOPER_KEY, DEVELOPER_SECRET, SECONDARY_AUTH_USER['token'], SECONDARY_AUTH_USER['tok
c = g.user_followers(SECONDARY_AUTH_USER['id'])
```

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:

6.1 Types of Contributions

6.1.1 Report Bugs

Report bugs at https://github.com/paulshannon/python-goodreads/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.

6.1.2 Fix Bugs

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

6.1.3 Implement Features

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

6.1.4 Write Documentation

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

6.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/paulshannon/python-goodreads/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 :)

6.2 Get Started!

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

1. Fork the *goodreads* repo on GitHub.

2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/python-goodreads.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 goodreads
$ cd goodreads/
$ 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 goodreads 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.

6.3 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/paulshannon/python-goodreads/pull_requests and make sure that the tests pass for all supported Python versions.

6.4 Tips

To run a subset of tests:

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

Credits

7.1 Development Lead

- Paul Shannon <paul@paulshannon.ca>

7.2 Contributors

OAuth code adapted from: https://gist.github.com/gpiancastelli/537923

History

8.1 0.1.2 (2014-02-19)

- **made all methods visible**
    - raises NotImplementedError

8.2 0.1.1 (2014-02-18)

- rename external references to python-goodreads

8.3 0.1.0 (2014-02-18)

- First release on PyPI.

Indices and tables

- genindex
- modindex
- search