Targets Python Documentation

Release 0.1.1

targets-fs

March 25, 2016

Contents

Contents:

CHAPTER 1

Targets Python

Targets is the simplified universal file system

- Free software: ISC license
- Documentation: https://targets.readthedocs.org.

1.1 Description

Most of the initial code is contributed by luigi project Started as discussion: https://groups.google.com/forum/#!topic/luigi-user/ZLGJCj0dBJI

... As a Luigi user for the past couple of years, I have found out that I am using the concept of Luigi targets in most of the projects I was working on. Transparent file system, atomic write/reads, formats is a great functionality, and is highly missing from the global Python eco system. Moreover, going over multiple workflow management project implementations(mrjob,airflow) I have always been surprised to see yet another implementation of the abstraction over fs/ssh/s3/ftp/etc all over again.

As a developer, and especially as a Python developer, I want to focus on implementation of my business requirements, instead of working with low level filesystem APIs. The standard "open" function is simply not strong enough. ...

1.2 Features

- TODO

1.3 Credits

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

Initial version of Targets is based on Luigi project.

Luigi was built at Spotify, mainly by Erik Bernhardsson and Elias Freider. Many other people have contributed since open sourcing in late 2012. Arash Rouhani is currently the chief maintainer of Luigi.

CHAPTER 2

Installation

At the command line:

```
$ easy_install targets
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv targets
$ pip install targets
```

CHAPTER 3

Usage

To use Targets Python in a project:

```
import targets
```

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/targets-fs/targets-python/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

Targets Python could always use more documentation, whether as part of the official Targets Python 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/targets-fs/targets-python/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 *targets* for local development.

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

```
$ git clone git@github.com:your_name_here/targets.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 targets
$ cd targets/
$ 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 targets tests
$ make test
$ make test-all
```

   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 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/targets-fs/targets-python/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_targets
```

CHAPTER 5

Credits

5.1 Development Lead

- evgenyshulman <shulman.e@gmail.com>

5.2 Contributors

None yet. Why not be the first?

5.3 Initial Version

Most of the initial code is contributed by luigi project Started at discussion: https://groups.google.com/forum/#!topic/luigi-user/ZLGJCj0dBJI

Luigi was built at Spotify, mainly by Erik Bernhardsson and Elias Freider. Many other people have contributed since open sourcing in late 2012. Arash Rouhani is currently the chief maintainer of Luigi.

CHAPTER 6

History

6.1 0.1.0 (2016-1-22)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search