changelogs Documentation

Release 0.15.1-dev

Jannis Gebauer

Jan 16, 2023

Contents

Contents:

A changelog finder and parser with command line interface for packages available on pypi, npm, rubygems and launchpad.net. Originally developed for pyup.io.

CHAPTER 1

Installation

To install changelogs, run this command in your terminal:

```
$ pip install changelogs
```

CHAPTER 2

Usage

To use changelogs in a Python project:

```
import changelogs

logs = changelogs.get("flask")
logs = changelogs.get("babel", vendor="npm")
logs = changelogs.get("bundler", vendor="npm")
```

Or, from the command line:

```
changelogs flask
changelogs babel npm
changelogs bundler gem
```

If you are on macOS, you can chain the *open* command:

```
changelogs babel npm >> babel.log && open babel.log
```

CHAPTER 3

About

When trying to get a changelog for a given package, there are a bunch of problems:

- There is no central place to store a changelog. If a project has a changelog, it's most likely somewhere in the git repo at all kinds of different places. This makes it hard to find.
- The package index meta data often has no direct link to the git repo. This makes the repo hard to find.
- There is no changelog standard. Everyone uses a different approach. This makes it hard to parse.

This project is trying to solve this by:

- first querying the package vendor for package meta data like the homepage or docs URL.
- if the meta data doesn't contain a valid URL to a repo, visit all available URLs and scrape them to find one.
- if there is a valid repo URL, visit the repo and look for possible changelogs like *Changes.txt*, *NEWS.md* or *history.rst*.
- fetch the content and somewhat try to parse it.

CHAPTER 4

Installation

4.1 Stable release

To install changelogs, run this command in your terminal:

```
$ pip install changelogs
```

This is the preferred method to install changelogs, 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.

4.2 From sources

The sources for changelogs can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/pyupio/changelogs
```

Or download the tarball:

```
$ curl  -OL https://github.com/pyupio/changelogs/tarball/master
```

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

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

CHAPTER 5

Usage

To use changelogs in a project:

```python
import changelogs

changelogs.get("flask")
```

CHAPTER 6

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/pyupio/changelogs/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" and "help wanted" is open to whoever wants to implement it.

6.1.3 Implement Features

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

6.1.4 Write Documentation

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

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

   ```
   $ git clone git@github.com:your_name_here/changelogs.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 changelogs
   $ cd changelogs/
   $ pip install -e .[dev]
   ```

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
   $ tox
   ```

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 all recent Python versions.

6.4 Tips

To run a subset of tests:

```
$ py.test tests.test_changelogs
```

CHAPTER 7

Indices and tables

- genindex
- modindex
- search