
marvelator

Release 0.1.0

Sep 05, 2017

Contents

1	Overview	1
1.1	Installation	1
1.2	Documentation	1
2	Installation	3
3	Usage	5
4	Reference	7
4.1	marvelator	7
5	Contributing	9
5.1	Bug reports	9
5.2	Documentation improvements	9
5.3	Feature requests and feedback	9
5.4	Development	10
6	Authors	11
7	Changelog	13
7.1	0.1.0 (2017-09-03)	13
8	Indices and tables	15
	Python Module Index	17

CHAPTER 1

Overview

docs	
tests	
package	

Python abstraction for developer.marvel.com requests

- Free software: BSD license

Installation

```
pip install marvelator
```

Documentation

<https://marvelator.readthedocs.io/>

CHAPTER 2

Installation

At the command line:

```
pip install marvelator
```


CHAPTER 3

Usage

Marvelator is an abstraction to the marvel API. It is just a lib that you can use instead of issuing raw calls to the developer api.

To use marvelator in a project:

```
from marvelator.characters_resource import CharactersResource
from marvelator.comics_resource import ComicsResource
```

Those are, at this point, the available APIs for you to use.

CHAPTER 4

Reference

marvelator

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

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.

Documentation improvements

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

Feature requests and feedback

The best way to send feedback is to file an issue at <https://github.com/leonardok/marvelator/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 :)

Development

To set up *marvelator* for local development:

1. Fork [marvelator](#) (look for the “Fork” button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/marvelator.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 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.

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 a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

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

- Leonardo Korndorfer - leok.me

CHAPTER 7

Changelog

0.1.0 (2017-09-03)

- First release.

CHAPTER 8

Indices and tables

- `genindex`
- `modindex`
- `search`

m

marvelator, [7](#)

M

marvelator (module), [7](#)