
profile-middleware Documentation

Release 0.2.6

Vaibhav Mishra

Oct 05, 2017

Contents

1	profile-middleware	3
1.1	Documentation	3
1.2	Quickstart	3
1.3	Features	3
2	Installation	5
3	Usage	7
4	Contributing	9
4.1	Types of Contributions	9
4.2	Get Started!	10
4.3	Pull Request Guidelines	11
4.4	Tips	11
5	Credits	13
5.1	Development Lead	13
5.2	Contributors	13
6	History	15
6.1	0.2.6 (2014-05-14)	15
6.2	0.2.5 (2014-05-13)	15
6.3	0.2.1 (2014-05-09)	15
6.4	0.2.0 (2014-05-09)	15
6.5	0.1.0 (2014-05-09)	15

Contents:

Profiler for django views

Documentation

The full documentation is at <https://django-profile-view.readthedocs.org/>.

Quickstart

Install profile-middleware:

```
pip install profile-middleware
```

Then use it in a project:

Usage

Features

- uses cProfile module

CHAPTER 2

Installation

At the command line:

```
$ easy_install profile-middleware
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv profiler  
$ pip install profile-middleware
```


CHAPTER 3

Usage

To use profile-middleware in a project:

```
pip install profile-middleware
```

now add it to `INSTALLED_APPS`

```
INSTALLED_APPS += ('profiler',)
```

Add the middleware class

```
MIDDLEWARE_CLASSES += ('profiler.middleware.ProfileMiddleware',)
```

To see profiling results, just visit a url of your django app and add

```
<url>?prof=time
```

this will show you a view with profiling details

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

You can contribute in many ways:

Types of Contributions

Currently a good way to contribute is editing `middleware.py` module in `profiler` directory, UI changes are also welcome to make output more useful

Report Bugs

Report bugs at <https://github.com/vinu76jsr/django-profiler/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.

Fix Bugs

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

Implement Features

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

Write Documentation

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

Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/vinu76jsr/profiler/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 :)

Get Started!

Ready to contribute? Here's how to set up *profile-middleware* for local development.

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

```
$ git clone git@github.com:your_name_here/django_profiler.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 django_profiler
$ cd django_profiler/
$ 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 profiler 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.

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/vinu76jsr/profiler/pull_requests and make sure that the tests pass for all supported Python versions.

Tips

To run a subset of tests:

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


CHAPTER 5

Credits

Development Lead

- Vaibhav Mishra <vinu76jsr@gmail.com>

Contributors

None yet. Why not be the first?

0.2.6 (2014-05-14)

- Updated project link in setup file - to be reflected on PYPI

0.2.5 (2014-05-13)

- Shipping with versions of jquery and datatables and images.
- profiler changed from hotshot to cProfile for performance

0.2.1 (2014-05-09)

- Version bump for README changes to be reflected on PyPI

0.2.0 (2014-05-09)

- Documentation update for contributors.

0.1.0 (2014-05-09)

- First release on PyPI.