Python Multiline Log Formatter

Release 0.1.8

January 15, 2016

Contents

1 Overview **1**
- 1.1 Benchmarking . . . 1
- 1.2 Installation . . . 1
- 1.3 Usage . . . 1
- 1.4 Documentation . . . 2
- 1.5 Development . . . 2
- 1.6 Other . . . 2

2 Installation **3**

3 Usage **5**

4 Reference **7**
- 4.1 multiline_formatter . . . 7

5 Contributing **9**
- 5.1 Bug reports . . . 9
- 5.2 Documentation improvements . . . 9
- 5.3 Feature requests and feedback . . . 9
- 5.4 Development . . . 9

6 Authors **11**

7 Changelog **13**
- 7.1 0.1.0 (2016-01-14) . . . 13

8 Indices and tables **15**

Python Module Index **17**

CHAPTER 1

Overview

docs	
tests	
package	

Python logging formatter that prefix multiline log message and trackebacks. Makes the logs more readable, both for tracebacks and for multiline log messages.

- Free software: BSD license

1.1 Benchmarking

Benchmark against pythons built in formatter **logging.Formatter**. The benchmark is done using **pytest-benchmark**, and the below results is from a run on a faily new MacBook Pro, using Python 2.7.

One should note that we are using StringIO as stream output, so we should expect the results to be even closer with more production alike setup, for example writing to disc.

1.2 Installation

```
pip install multiline-log-formatter
```

1.3 Usage

Add this to dictConfig:

```
'formatters': {
    'default': {
        '()': 'multiline_formatter.formatter.MultilineMessagesFormatter',
        'format': '[%(levelname)s] %(message)s'
    },
},
```

And log messages will look like this:

```
[ERROR] LOGGING_MESSAGE ... (49564:MainThread)
... (49564:MainThread) : Traceback (most recent call last):
... (49564:MainThread) :   File "/Users/plauri/work/opensource/python-multiline-log-formatter/tests/t
... (49564:MainThread) :     raise Exception('EXCEPTION_MESSAGE')
... (49564:MainThread) : Exception: EXCEPTION_MESSAGE
```

And if you don't like the default, you can customize it by extending **MultilineMessagesFormatter** and set **multiline_marker**. You can also change **multiline_fmt**, but assure you include **%(message)s** in the formating string.

1.4 Documentation

https://python-multiline-log-formatter.readthedocs.org/

1.5 Development

To run the all tests run:

```
tox
```

1.6 Other

This project sceleton is generated by ionelmc's pylibrary cookiecutter.

CHAPTER 2

Installation

At the command line:

```
pip install multiline-log-formatter
```

CHAPTER 3

Usage

To use Python Multiline Log Formatter in a project:

```
import multiline_formatter
```

Add this to dictConfig:

```
'formatters': {
    'default': {
        '()': 'multiline_formatter.formatter.MultilineMessagesFormatter',
        'format': '[%(levelname)s] %(message)s'
    },
},
```

And log messages will look like this:

```
[ERROR] LOGGING_MESSAGE ... (49564:MainThread)
... (49564:MainThread) : Traceback (most recent call last):
... (49564:MainThread) :   File "/Users/plauri/work/opensource/python-multiline-log-formatter/tests/t
... (49564:MainThread) :     raise Exception('EXCEPTION_MESSAGE')
... (49564:MainThread) : Exception: EXCEPTION_MESSAGE
```

And if you don't like the default, you can customize it by extending **MultilineMessagesFormatter** and set **multiline_marker**. You can also change **multiline_fmt**, but assure you include **%(message)s** in the formating string.

CHAPTER 4

Reference

4.1 multiline_formatter

CHAPTER 5

Contributing

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

5.1 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.

5.2 Documentation improvements

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

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/peterlauri/python-multiline-log-formatter/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 :)

5.4 Development

To set up *python-multiline-log-formatter* for local development:

1. Fork python-multiline-log-formatter on GitHub.
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/python-multiline-log-formatter.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.

5.4.1 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`.

5.4.2 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

- Peter Lauri - https://github.com/peterlauri

CHAPTER 7

Changelog

7.1 0.1.0 (2016-01-14)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

m

multiline_formatter, 7

M

multiline_formatter (module), 7