pyline Documentation

Release 0.3.4

Wes Turner

April 25, 2015

Contents

1 pyline **3**
- 1.1 Features . . . 3
- 1.2 Why . . . 3
- 1.3 What . . . 3
- 1.4 Installing . . . 4
- 1.5 Usage . . . 4
- 1.6 Documentation . . . 5
- 1.7 License . . . 5

2 Installation **7**

3 Usage **9**

4 Contributing **11**
- 4.1 Types of Contributions . . . 11
- 4.2 Get Started! . . . 12
- 4.3 Pull Request Guidelines . . . 12
- 4.4 Tips . . . 13

5 Credits **15**

6 History **17**
- 6.1 0.3.4 (2015-04-25) . . . 17
- 6.2 0.3.3 (2015-04-25) . . . 17
- 6.3 0.3.2 (2014-11-30) . . . 17
- 6.4 0.2.0 (2014-08-24) . . . 17
- 6.5 0.1.5 (2014-05-12) . . . 17
- 6.6 0.1.4 (2014-05-12) . . . 17
- 6.7 0.1.3 (2014-05-12) . . . 18
- 6.8 0.1.2 (2014-05-12) . . . 18
- 6.9 0.1.1 (2014-05-12) . . . 18
- 6.10 0.1.0 (2014-05-12) . . . 18
- 6.11 0.0.1 (Unreleased) . . . 18

7 License **19**
- 7.1 PYTHON SOFTWARE FOUNDATION LICENSE VERSION 2 . . . 19

8 Indices and tables **21**

Contents:

CHAPTER 1

pyline

GitHub | PyPi | Warehouse | ReadTheDocs | Travis-CI Pyline: a grep-like, sed-like command-line tool (Python package)

1.1 Features

- Compatibility with the original pyline recipe
- Python str.split by an optional delimiter str (`-F`, `--input-delim`)
- Python regex (`-r`, `--regex`, `-R`, `--regex-options`)
- Output as *txt*, *csv*, *tsv*, *json*, *html* (`-O csv`, `--output-filetype=csv`)
- Output as Markdown/ReStructuredText checkboxes (`-O checkbox`, `--output-filetype=checkbox`)
- Lazy sorting (`-s`, `--sort-asc`; `-S`, `--sort-desc`)
- Create path.py (or pathlib) objects from each line (`-p`, `--path-tools`)
- Functional namedtuples, iterators `yield` -ing generators
- optparse argument parsing (`-h`, `--help`)
- cookiecutter-pypackage project templating

1.2 Why

Somewhat unsurprisingly, I found the original pyline recipe while searching for "python grep sed" (see `AUTHORS.rst` and `LICENSE.psf`).

I added an option for setting `p = Path(line)` in the eval/compile command context and added it to my dotfiles ; where it grew tests and an `optparse.OptionParser`; and is now promoted to a GitHub project with ReadTheDocs documentation, tests with tox and Travis-CI, and a setup.py for PyPi.

1.3 What

Pyline is an ordered MapReduce tool:

Input Readers:

- stdin (default)
- file (`codecs.open(file, 'r', encoding='utf-8')`)

Map Functions:

- Python module imports (`-m os`)
- Python regex pattern (`-r '\(.*\)'`)
- path library (`p` from `--pathpy` OR `--pathlib`)
- Python codeobj **eval** output transform:

```
ls | pyline -m os 'line and os.path.abspath(line.strip())'
ls | pyline -r '\(.*\)' 'rgx and (rgx.group(0), rgx.group(1)) or line'
ls | pyline -p 'p and p.abspath() or ("# ".format(line))'

# With an extra outer loop to bind variables in
# (because (_p = p.abspath(); <codeobj>) does not work)
find $PWD | pyline --pathpy -m os -m collections --input-delim='/' \
    'p and [collections.OrderedDict((
            ("p", p),
            ("_p", _p),
            ("_p.split()", str(_p).split(os.path.sep)),
            ("line.rstrip().split()", line.rstrip().split(os.path.sep)),
            ("l.split()", l.split(os.path.sep)),
            ("words", words),
            ("w", w)))
        for _p in [p.abspath()]][0]' \
       -O json
```

Partition Function: None

Compare Function: `Result(collections.namedtuple).__cmp__`

Reduce Functions: `bool()`, `sorted()`

Output Writers: `ResultWriter` classes

```
pyline -O csv
pyline -O tsv
pyline -O json
```

1.4 Installing

Install from PyPi:

```
pip install pyline
```

Install from GitHub as editable (add a `pyline.pth` in `site-packages`):

```
pip install -e git+https://github.com/westurner/pyline#egg=pyline
```

1.5 Usage

Print help:

```
pyline --help
```

Process:

```
# Print every line (null transform)
cat ~/.bashrc | pyline line
cat ~/.bashrc | pyline l

# Number every line
cat ~/.bashrc | pyline -n l

# Print every word (str.split(input-delim=None))
cat ~/.bashrc | pyline words
cat ~/.bashrc | pyline w

# Split into words and print (default: tab separated)
cat ~/.bashrc | pyline 'len(w) >= 2 and w[1] or "?"'

# Select the last word, dropping lines with no words
pyline -f ~/.bashrc 'w[-1:]'

# Regex matching with groups
cat ~/.bashrc | pyline -n -r '^#(.*)' 'rgx and rgx.group()'
cat ~/.bashrc | pyline -n -r '^#(.*)'

## Original Examples
# Print out the first 20 characters of every line
tail access_log | pyline "line[:20]"

# Print just the URLs in the access log (seventh "word" in the line)
tail access_log | pyline "words[6]"
```

Work with paths and files:

```
# List current directory files larger than 1 Kb
ls | pyline -m os \
  "os.path.isfile(line) and os.stat(line).st_size > 1024 and line"

# List current directory files larger than 1 Kb
#pip install path.py
ls | pyline -p 'p and p.size > 1024 and line'
```

1.6 Documentation

https://pyline.readthedocs.org/en/latest/

1.7 License

Python Software License

CHAPTER 2

Installation

At the command line:

```
$ easy_install pyline
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv pyline
$ pip install pyline
```

CHAPTER 3

Usage

To use pyline in a project:

```
import pyline
```

To use pyline as a shell command:

```
$ pyline --help
/bin/sh: 1: pyline: not found
```

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/westurner/pyline/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

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

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

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

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 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/westurner/pyline/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_pyline
```

CHAPTER 5

Credits

- Graham Fawcett
- Jacob Oscarson
- Mark Eichin
- Wes Turner – https://github.com/westurner

CHAPTER 6

History

6.1 0.3.4 (2015-04-25)

- DOC: HISTORY.txt

6.2 0.3.3 (2015-04-25)

- TST, BUG, CLN, DOC

6.3 0.3.2 (2014-11-30)

- DOC: pyline/pyline.py: docstrings, import path as pathpy
- BUG: pyline/__init__.py: Set pyline.pyline.__main__ correctly (so that `python -m pyline.pyline --help` works)
- DOC: usage.rst: add :shell: option to 'pyline –help' output

6.4 0.2.0 (2014-08-24)

- BUG: add NullHandler to logger (closes #6)
- ENH: Add checkbox output formatter (closes #5)

6.5 0.1.5 (2014-05-12)

- DOC: Updated HISTORY.rst
- DOC: setup.py keywords, classifiers

6.6 0.1.4 (2014-05-12)

- DOC: setup.py version, download_url, license

6.7 0.1.3 (2014-05-12)

- DOC: setup.py description spans newline

6.8 0.1.2 (2014-05-12)

- DOC: Setup.py long_description

6.9 0.1.1 (2014-05-12)

- CLN: factor functions out of main and pyline
- BUG: `-p` path.py option

6.10 0.1.0 (2014-05-12)

- First release on PyPI.

6.11 0.0.1 (Unreleased)

Source: http://code.activestate.com/recipes/437932-pyline-a-grep-like-sed-like-command-line-tool/

- Updated 2012.11.17, Wes Turner
- Updated 2005.07.21, thanks to Jacob Oscarson
- Updated 2006.03.30, thanks to Mark Eichin

CHAPTER 7

License

7.1 PYTHON SOFTWARE FOUNDATION LICENSE VERSION 2

1. This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using this software ("Python") in source or binary form and its associated documentation.

2. Subject to the terms and conditions of this License Agreement, PSF hereby grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use Python alone or in any derivative version, provided, however, that PSF's License Agreement and PSF's notice of copyright, i.e., "Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014 Python Software Foundation; All Rights Reserved" are retained in Python alone or in any derivative version prepared by Licensee.

3. In the event Licensee prepares a derivative work that is based on or incorporates Python or any part thereof, and wants to make the derivative work available to others as provided herein, then Licensee hereby agrees to include in any such work a brief summary of the changes made to Python.

4. PSF is making Python available to Licensee on an "AS IS" basis. PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material breach of its terms and conditions.

7. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between PSF and Licensee. This License Agreement does not grant permission to use PSF trademarks or trade name in a trademark sense to endorse or promote products or services of Licensee, or any third party.

8. By copying, installing or otherwise using Python, Licensee agrees to be bound by the terms and conditions of this License Agreement.

CHAPTER 8

Indices and tables

- *genindex*
- *modindex*
- *search*