TopShape Documentation

Release 1.0.0

Martin Chlumsky

May 29, 2018

Contents

Contents:

CHAPTER 1

TopShape

Library for easily creating text interfaces that look like Linux's top program.

It is built on top of urwid but requires no knowledge of urwid itself.

- Free software: MIT license
- Documentation: https://topshape.readthedocs.io.
- Python versions supported: 2.7, 3.3, 3.4, 3.5, 3.6

1.1 Quickstart

Here's an example of how to use TopShape:

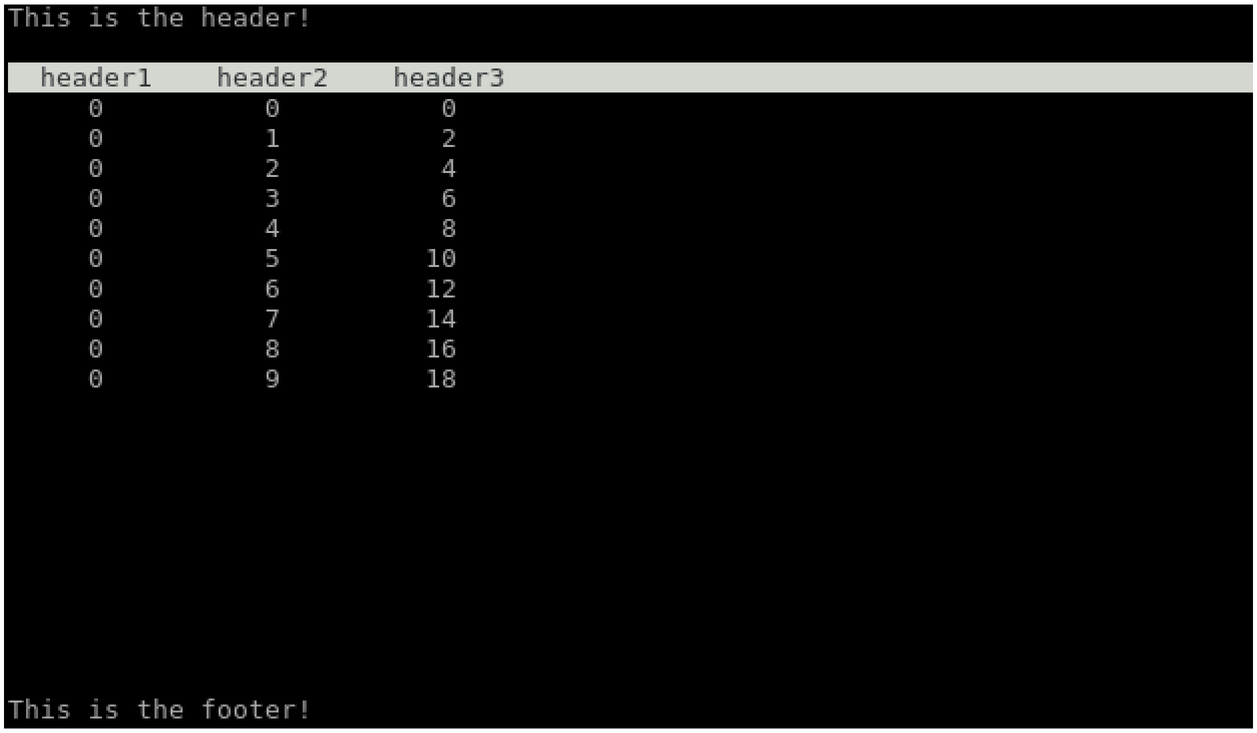
```python
from topshape import TopShape


# The columns are a list (or tuple) of dictionaries. Each
# dictionary defines a column in the body
columns = ({'label': 'header1'},
           {'label': 'header2'},
           {'label': 'header3'})


# The body function will be passed as a callback that must
# return a 2-dimensional array everytime it's called.
def body():
    return [[str(i*j) for i in range(3)] for j in range(10)]


# The header function will be passed as a callback that must
# return a string everytime it's called.
def header():
    return 'This is the header!'
```

(continues on next page)

(continued from previous page)

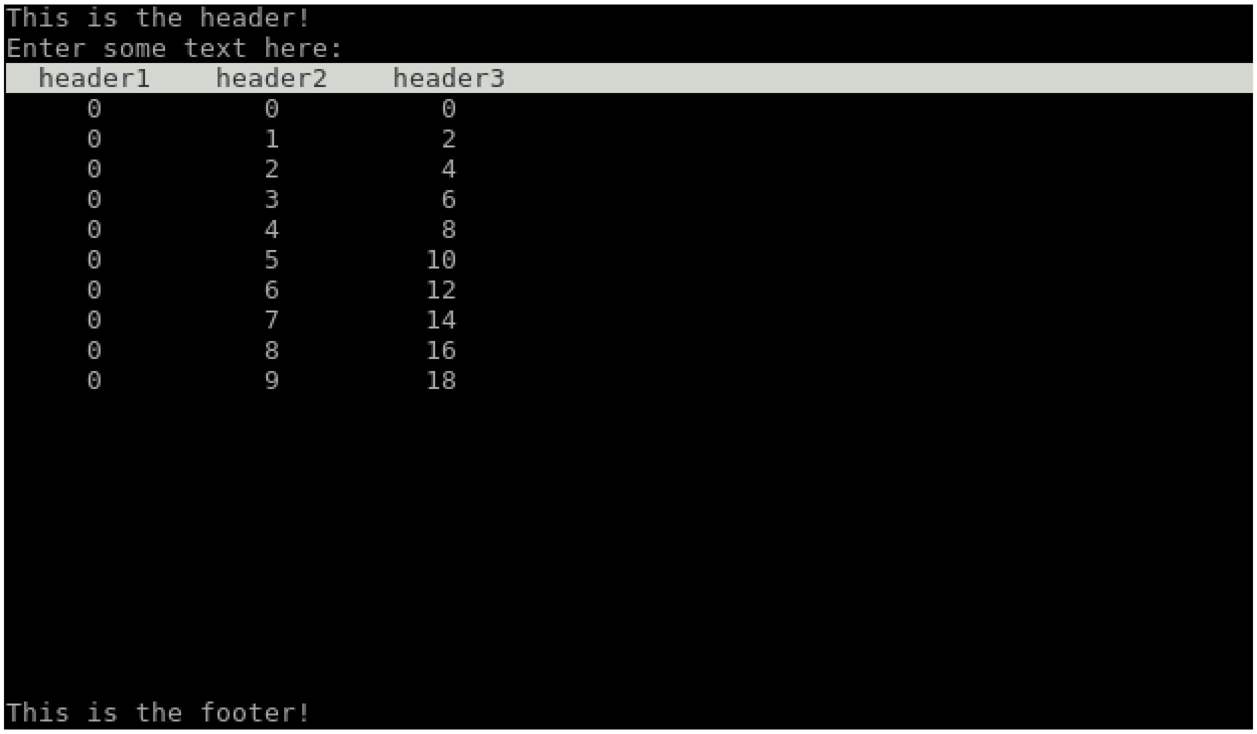
```python
# The footer function will be passed as a callback that must
# return a string everytime it's called.
def footer():
    return 'This is the footer!'


def handle_q(app):
    app.exit()

def handle_f(app, answer):
    # do something with the answer
    # ...

# key_map maps keys pressed to callbacks
key_map = {'q': handle_q,
           'f': (handle_f, 'Enter some text here:'}

app = TopShape.create_app(columns, body, header, footer,
                          key_mapping=key_map)
app.run()
```

Output:



```
This is the header!

  header1     header2     header3
     0           0           0
     0           1           2
     0           2           4
     0           3           6
     0           4           8
     0           5          10
     0           6          12
     0           7          14
     0           8          16
     0           9          18









This is the footer!
```

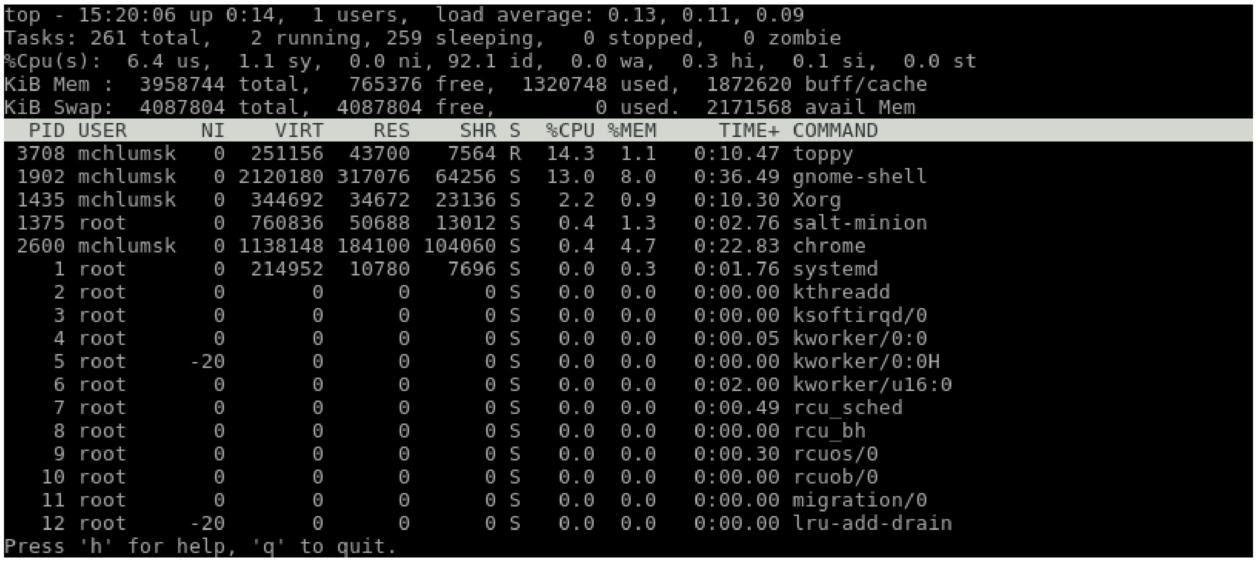
Output (waiting for input from user):



```
This is the header!
Enter some text here:
  header1     header2     header3
     0           0           0
     0           1           2
     0           2           4
     0           3           6
     0           4           8
     0           5          10
     0           6          12
     0           7          14
     0           8          16
     0           9          18









This is the footer!
```

There is also a more complete example here which is a clone of the linux top program.

Screenshot:



```
top - 15:20:06 up 0:14,  1 users,  load average: 0.13, 0.11, 0.09
Tasks: 261 total,   2 running, 259 sleeping,   0 stopped,   0 zombie
%Cpu(s):  6.4 us,  1.1 sy,  0.0 ni, 92.1 id,  0.0 wa,  0.3 hi,  0.1 si,  0.0 st
KiB Mem :  3958744 total,   765376 free,  1320748 used,  1872620 buff/cache
KiB Swap:  4087804 total,  4087804 free,        0 used.  2171568 avail Mem
  PID USER       NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 3708 mchlumsk   0  251156  43700   7564 R  14.3  1.1   0:10.47 toppy
 1902 mchlumsk   0 2120180 317076  64256 S  13.0  8.0   0:36.49 gnome-shell
 1435 mchlumsk   0  344692  34672  23136 S   2.2  0.9   0:10.30 Xorg
 1375 root       0  760836  50688  13012 S   0.4  1.3   0:02.76 salt-minion
 2600 mchlumsk   0 1138148 184100 104060 S   0.4  4.7   0:22.83 chrome
    1 root       0  214952  10780   7696 S   0.0  0.3   0:01.76 systemd
    2 root       0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    3 root       0       0      0      0 S   0.0  0.0   0:00.00 ksoftirqd/0
    4 root       0       0      0      0 S   0.0  0.0   0:00.05 kworker/0:0
    5 root     -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    6 root       0       0      0      0 S   0.0  0.0   0:02.00 kworker/u16:0
    7 root       0       0      0      0 S   0.0  0.0   0:00.49 rcu_sched
    8 root       0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root       0       0      0      0 S   0.0  0.0   0:00.30 rcuos/0
   10 root       0       0      0      0 S   0.0  0.0   0:00.00 rcuob/0
   11 root       0       0      0      0 S   0.0  0.0   0:00.00 migration/0
   12 root     -20       0      0      0 S   0.0  0.0   0:00.00 lru-add-drain
Press 'h' for help, 'q' to quit.
```

1.2 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

CHAPTER 2

Installation

2.1 Stable release

To install TopShape, run this command in your terminal:

```
$ pip install topshape
```

This is the preferred method to install TopShape, 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.

2.2 From sources

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

You can either clone the public repository:

```
$ git clone git://github.com/mchlumsky/topshape
```

Or download the tarball:

```
$ curl  -OL https://github.com/mchlumsky/topshape/tarball/master
```

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

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

CHAPTER 3

Usage

A topshape program consists of creating a TopShape object by calling TopShape.create_app() and then calling run().

```
from topshape import TopShape

# define arguments for create_app here
# ...

app = TopShape.create_app(...)
app.run()
```

Checkout here for the arguments to pass to create_app().

3.1 Exiting the application

To exit the application, simply call exit() on the TopShape object.

3.2 Sorting column

The rows in the body of the topshape application are sorted by a sorting column (defaults to the leftmost column and can be overridden by passing an arg to create_app()).

While in the main loop, the current column used for sorting can be moved left or right by calling the TopShape object's move_sort_left() and move_sort_right() methods.

3.3 Keypress handling

You can define what topshape does when certain keys are pressed by passing a dict as the arg key_mapping to create_app().

`key_mapping`'s keys are the physical keys that get pressed and the values are the functions that get called when the keys get pressed. The values can also be tuples (or lists) where each tuple is `(handler_function, question)`. The question will be displayed as the bottom line in the header while waiting for input from the user. Once the enter key is pressed, the `handler_function` is called and passed the TopShape app object and the answer to the question typed in the bottom line of the header.

The key *h* is not overridable. It always displays the help output. Any override for this key in `key_mapping` is ignored.

The key *q* defaults to causing topshape to exit however it can be overriden.

3.4 Displaying help to the user

While the application is running, pressing `h` will show the help screen. The help text is the string that was passed as the `help_text` argument to `create_app()`.

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/mchlumsky/topshape/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" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

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

4.1.4 Write Documentation

TopShape could always use more documentation, whether as part of the official TopShape 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/mchlumsky/topshape/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 *topshape* for local development.

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

   ```
   $ git clone git@github.com:your_name_here/topshape.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 topshape
   $ cd topshape/
   $ 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 topshape tests
   $ python setup.py test or 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, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/mchlumsky/topshape/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_topshape
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search