
Asciinemaster Documentation

Release 0.2.0

Dinu Gherman

May 01, 2018

Contents:

1	Asciinemaster	1
1.1	Features	1
1.2	Examples	1
1.3	Credits	1
2	Installation	3
2.1	Stable release	3
2.2	From sources	3
3	Usage	5
4	Contributing	7
4.1	Types of Contributions	7
4.2	Get Started!	8
4.3	Pull Request Guidelines	9
4.4	Tips	9
4.5	Deploying	9
5	Credits	11
5.1	Development Lead	11
5.2	Contributors	11
6	History	13
6.1	0.1.1 (2018-04-xx)	13
6.2	0.1.0 (2018-04-26)	13
7	Indices and tables	15

Asciinemaster contains some experiments to add features to drive Asciinema.

- Free software: GNU General Public License v3
- Documentation: <https://asciinemaster.readthedocs.io>.

This will take a shell-like script, iterate over its commands, execute them, and record the commands and their respective output as an Asciinema.org (v2) screencast.

Later, it will also take a shell-like script and the generated screencast from it, re-execute the script and compare the new screencast to the old one for testing purposes.

1.1 Features

- TODO

1.2 Examples

- TODO

1.3 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

2.1 Stable release

To install Asciiemaster, run this command in your terminal:

```
$ pip install asciinemaster
```

This is the preferred method to install Asciiemaster, 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 Asciiemaster can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/deeplook/asciinemaster
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/deeplook/asciinemaster/tarball/master
```

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

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


CHAPTER 3

Usage

To use Asciiinemaster in a project:

```
import asciiinemaster
```


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/deeplook/asciinemaster/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

Asciinemaster could always use more documentation, whether as part of the official Asciinemaster 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/deeplook/asciinemaster/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 *asciinemaster* for local development.

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

```
$ git clone git@github.com:your_name_here/asciinemaster.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 asciinemaster
$ cd asciinemaster/
$ 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 asciinemaster 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 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/deeplook/asciinemaster/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests.test_asciinemaster
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

5.1 Development Lead

- Dinu Gherman <gberman@darwin.in-berlin.de>

5.2 Contributors

None yet. Why not be the first?

6.1 0.1.1 (2018-04-xx)

6.1.1 Features added

- Added initial aciinemaster code, developed before as a single script.
- Replaced click with argparse.
- Added CHANGES.rst (inspired by that of lxml).

6.2 0.1.0 (2018-04-26)

6.2.1 Features added

- Created project skeleton using audreyr/cookiecutter-pypackage template.

CHAPTER 7

Indices and tables

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