GarbageTruck Documentation

Release 0.1.0

Brad Robel-Forrest

September 27, 2016

Contents

Contents:

CHAPTER 1

GarbageTruck

A small tool to periodically move old files into the local file system trash.

- Free software: MIT license
- Full documentation (including how to install): https://garbagetruck.readthedocs.io.

1.1 Features

Use *GarbageTruck* to build and maintain scheduled cleanup of files in various directories that tend to collect files over time. *GarbageTruck* will send any files older than a relative period to the local file system trash using the current user's crontab to schedule checks for old files from cron. This makes the utility safe in that any files moved to the trash could be restored simply without worrying about immediate loss (until the trash is emptied, of course).

For example, let's say one never cleans out their downloaded files. Here's how to set a *GarbageTruck* job to periodically move files older than six months into the trash:

```
$ garbagetruck set --older-than '6 months' --check-every day 'Clean out old downloads' ~/Downloads/
```

This will set up a schedule (using cron) to look for files each day that are older than three montsh and have them moved into the correct trash (courtesy of send2trash). The details can be shown like this:

```
$ garbagetruck list
[2016-09-03T15:55:32-0700 #31693] INFO     garbagetruck Job 57d1db0a8b8427c3041ac1af89b0a348: name="C
[2016-09-03T15:55:32-0700 #31693] DEBUG    garbagetruck * 1 * * * /Users/brad/.virtualenvs/garbage_tr
```

Each call to the *set* command will replace the same named job. Alternatively, if the job is no longer useful, remove it like this:

```
$ garbagetruck remove 'Clean out old downloads'
```

To check on a job, any problems and results will be logged to one of the following locations:

- OS X will use `~/Library/Logs/garbagetruck.log`.
- Other systems will rely on what click_app_dir returns.

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 GarbageTruck, run this command in your terminal:

```
$ pip install garbagetruck
```

This is the preferred method to install GarbageTruck, 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 GarbageTruck can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/bradrf/garbagetruck
```

Or download the tarball:

```
$ curl  -OL https://github.com/bradrf/garbagetruck/tarball/master
```

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

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

CHAPTER 3

Usage

To use GarbageTruck in a project:

```
import garbagetruck
```

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/bradrf/garbagetruck/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

GarbageTruck could always use more documentation, whether as part of the official GarbageTruck 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/bradrf/garbagetruck/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 *garbagetruck* for local development.

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

```
$ git clone git@github.com:your_name_here/garbagetruck.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 garbagetruck
$ cd garbagetruck/
$ 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 garbagetruck 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/bradrf/garbagetruck/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_garbagetruck
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search