
Deployer

Release 0.1.0

Sep 20, 2018

Contents

| | | |
|----------|---|-----------|
| 1 | Overview | 1 |
| 1.1 | Features | 1 |
| 1.2 | Installation | 2 |
| 1.3 | Documentation | 2 |
| 1.4 | Development | 2 |
| 2 | Installation | 3 |
| 3 | Usage | 5 |
| 4 | Reference | 7 |
| 4.1 | deployer | 7 |
| 5 | Contributing | 9 |
| 5.1 | Bug reports | 9 |
| 5.2 | Documentation improvements | 9 |
| 5.3 | Feature requests and feedback | 9 |
| 5.4 | Development | 10 |
| 6 | Authors | 11 |
| 7 | Changelog | 13 |
| 7.1 | 0.1.0 (2018-08-21) | 13 |
| 8 | Indices and tables | 15 |
| | Python Module Index | 17 |

CHAPTER 1

Overview

| | |
|-------|--|
| docs | |
| tests | |

Local+CI+CD Made Easy!

- Free software: Apache Software License 2.0

1.1 Features

- Project strives for pipelines being similar in spirit to Ansible playbooks/roles.
- Supports for both POSIX and Windows operating systems.
- A wide range of Python versions are supported: 2.7, 3.4, 3.5, 3.6, 3.7, PyPy, and PyPy3.
- Automatic schema validation; however, Jinja templating is not validated until run-time of the pipeline.
- Pipeline stages (named group of tasks), including arbitrarily nested ones.
- Matrix driven pipelines, including arbitrarily nested ones.
- Command and Shell script execution; supporting inline, templated inline, and/or external tools.
 - Support for Bourne, Bash, CMD.EXE, and PowerShell 2.x or newer.
 - Support for automatic retry, on failures.
 - Support for silent output, unless Command/Shell fails.
- Support for conditional task execution.
- Environment variables.

- Template variables.
- Filter execution of tags and/or Matrix tags.
- Support for recursive Jinj2 templating.
- All execution times are tracked and reported; at all possible nested levels of the running pipeline.
- Support for various logging levels

1.2 Installation

```
pip install deployer
```

1.3 Documentation

<https://py-deployer.readthedocs.io/en/latest/>

1.4 Development

To run the all tests run:

```
tox
```

Note, to combine the coverage data from all the tox environments run:

| | |
|---------|--|
| Windows | <pre>set PYTEST_ADDOPTS=--cov-append tox</pre> |
| Other | <pre>PYTEST_ADDOPTS=--cov-append tox</pre> |

CHAPTER 2

Installation

At the command line:

```
pip install deployer
```


CHAPTER 3

Usage

To use Deployer in a project:

```
import deployer
```


4.1 `deployer`

Deployer versioning information.

`deployer.cli.initialize (level=10)`
Perform basic initialization of program.

`deployer.cli.setup_logging (level=10)`
Initialize the logging infrastructure.

`deployer.loader.ordered_load (stream, loader=<class 'yaml.loader.SafeLoader'>, object_pairs_hook=<class 'collections.OrderedDict'>)`
Load YAML, preserving the ordering of all data.

class `deployer.registry.Registry`
A central data repository. Responsible for alleviating global variable usage.

plugins ()
All available plugins.

register_plugin (*name*, *cls*)
Register the *cls* for handling pipeline nodes, utilizing *name*.

class `deployer.singleton.Singleton`
Wrap classes to create Singleton objects.

The public plug-in API of the PyDeployer project.

copyright

3. Copyright 2018 by Joseph Benden.

license Apache License 2.0, see LICENSE.txt for full details.

`deployer.plugins.api.hookimpl = <pluggy.hooks.HookimplMarker object>`
All PyDeployer plug-ins *MUST* decorate their hooks with *hookimpl*. This decorator is used by pluggy to au-

tomatically find all functions which implement a hook, per the available hooks specified inside of *deployer.plugins.hookspec*.

```
deployer.plugins.hookspec = <module 'deployer.plugins.hookspec' from '/home/docs/checkouts/  
Hook specifications for PyDeployer`.
```

Copyright

3. Copyright 2018 by Joseph Benden.

License Apache License 2.0, see LICENSE for full details.

Hook specifications for PyDeployer`.

copyright

3. Copyright 2018 by Joseph Benden.

license Apache License 2.0, see LICENSE for full details.

```
deployer.plugins.hookspec.deployer_register(registry)  
A hook available for a plug-in to perform its' registration.
```

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

Deployer could always use more documentation, whether as part of the official Deployer 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/jbenden/deployer/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 *deployer* for local development:

1. Fork [deployer](#) (look for the “Fork” button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/deployer.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 -- pytest -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

- Joseph Benden - <https://benden.us>

CHAPTER 7

Changelog

7.1 0.1.0 (2018-08-21)

- First release on PyPI.

CHAPTER 8

Indices and tables

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

d

`deployer`, [7](#)
`deployer.plugins.api`, [7](#)
`deployer.plugins.hookspec`, [8](#)

D

deployer (module), 7
deployer.plugins.api (module), 7
deployer.plugins.hookspec (module), 8
deployer_register() (in module deployer.plugins.hookspec), 8

H

hookimpl (in module deployer.plugins.api), 7
hookspec (in module deployer.plugins), 8

I

initialize() (in module deployer.cli), 7

O

ordered_load() (in module deployer.loader), 7

P

plugins() (deployer.registry.Registry method), 7

R

register_plugin() (deployer.registry.Registry method), 7
Registry (class in deployer.registry), 7

S

setup_logging() (in module deployer.cli), 7
Singleton (class in deployer.singleton), 7