Pelican Documentation

Release 0.1.0

Carsten Ehbrecht

Jul 09, 2019

CONTENTS:

Pelican (the bird) *The Brown Pelican is a common bird of San Franscisco Bay Area.*

A Web Processing Service for ESGF compute.

- Free software: Apache Software License 2.0
- Documentation: https://birdhouse-pelican.readthedocs.io.

CHAPTER

ONE

CREDITS

This package was created with Cookiecutter and the bird-house/cookiecutter-birdhouse project template.

1.1 Installation

1.1.1 Install from Conda

Warning: TODO: Prepare Conda package.

1.1.2 Install from GitHub

Check out code from the Pelican GitHub repo and start the installation:

```
$ git clone https://github.com/bird-house/pelican.git
$ cd pelican
$ conda env create -f environment.yml
$ source activate pelican
$ python setup.py develop
```

. . . or do it the lazy way

The previous installation instructions assume you have Anaconda installed. We provide also a `Makefile` to run this installation without additional steps:

```
$ git clone https://github.com/bird-house/pelican.git
$ cd pelican
$ make clean    # cleans up a previous Conda environment
$ make install  # installs Conda if necessary and runs the above installation steps
```

1.1.3 Start Pelican PyWPS service

After successful installation you can start the service using the `pelican` command-line.

```
$ pelican --help # show help
$ pelican start  # start service with default configuration

OR

$ pelican start --daemon # start service as daemon
loading configuration
forked process id: 42
```

The deployed WPS service is by default available on:

http://localhost:5000/wps?service=WPS&version=1.0.0&request=GetCapabilities.

Note: Remember the process ID (PID) so you can stop the service with `kill PID`.

You can find which process uses a given port using the following command (here for port 5000):

```
$ netstat -nlp | grep :5000
```

Check the log files for errors:

```
$ tail -f  pywps.log
```

. . . or do it the lazy way

You can also use the `Makefile` to start and stop the service:

```
$ make start
$ make status
$ tail -f pywps.log
$ make stop
```

1.1.4 Run Pelican as Docker container

You can also run Pelican as a Docker container.

Warning: TODO: Describe Docker container support.

1.1.5 Use Ansible to deploy Pelican on your System

Use the Ansible playbook for PyWPS to deploy Pelican on your system.

1.2 Configuration

1.2.1 Command-line options

You can overwrite the default PyWPS configuration by using command-line options. See the Pelican help which options are available:

```
$ pelican start --help
--hostname HOSTNAME        hostname in PyWPS configuration.
--port PORT                port in PyWPS configuration.
```

Start service with different hostname and port:

```
$ pelican start --hostname localhost --port 5001
```

1.2.2 Use a custom configuration file

You can overwrite the default PyWPS configuration by providing your own PyWPS configuration file (just modifiy the options you want to change). Use one of the existing `sample-*.cfg` files as example and copy them to `etc/custom.cfg`.

For example change the hostname (*demo.org*) and logging level:

```
$ cd pelican
$ vim etc/custom.cfg
$ cat etc/custom.cfg
[server]
url = http://demo.org:5000/wps
outputurl = http://demo.org:5000/outputs

[logging]
level = DEBUG
```

Start the service with your custom configuration:

```
# start the service with this configuration
$ pelican start -c etc/custom.cfg
```

1.3 Developer Guide

Warning: To create new processes look at examples in Emu.

1.3.1 Building the docs

First install dependencies for the documentation:

```
$ make bootstrap_dev
$ make docs
```

1.3.2 Running tests

Run tests using pytest.

First activate the `pelican` Conda environment and install `pytest`.

```
$ source activate pelican
$ conda install pytest flake8  # if not already installed
```

Run quick tests (skip slow and online):

```
$ pytest -m 'not slow and not online'"
```

Run all tests:

```
$ pytest
```

Check pep8:

```
$ flake8
```

1.3.3 Run tests the lazy way

Do the same as above using the `Makefile`.

```
$ make test
$ make testall
$ make pep8
```

1.3.4 Prepare a release

Update the Conda specification file to build identical environments on a specific OS.

Note: You should run this on your target OS, in our case Linux.

```
$ make clean
$ make install
$ make spec
```

1.3.5 Bump a new version

Make a new version of Pelican in the following steps:

- Make sure everything is commit to GitHub.
- Update `CHANGES.rst` with the next version.
- Dry Run: `bumpversion --dry-run --verbose --new-version 0.8.1 patch`
- Do it: `bumpversion --new-version 0.8.1 patch`
- ... or: `bumpversion --new-version 0.9.0 minor`
- Push it: `git push`
- Push tag: `git push --tags`

See the bumpversion documentation for details.

1.4 Processes

1.4.1 Say Hello

1.5 Changes

1.5.1 0.1.0 (2019-02-13)

- First release.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search