ECget Documentation

Release 0.5

Doug Latornell

Jan 16, 2017

Contents

ECget is a tool to get observation data from Environment Canada weather and hydrometric web services and sites and store it in project-specific file formats. It is written in Python and is designed to be easily extensible for use in various projects and applications.

Contents:

CHAPTER 1

Installation

1.1 Python Versions

ECget is being developed under Python 3.4 and is tested with Python 2.7.

1.2 Source Code

The source code is hosted on Bitbucket: https://bitbucket.org/douglatornell/ecget

1.3 Reporting Bugs

Please report bugs through the Bitbucket project: https://bitbucket.org/douglatornell/ecget/issues

1.4 Installation

The steps to install ECget are:

1. Use Mercurial to clone the project repo from Bitbucket; i.e.

```
$ hg clone https://bitbucket.org/douglatornell/ecget
```

2. Install the Python packages that ECget depends on.
3. Install ECget iteself.

Several ways of accomplishing the above steps are described below. Please choose the one that best suits your working environment and your personal preferences regarding isolation of software installations.

1.4.1 Install in an Anaconda Python Default Environment

If you use the Anaconda Python distribution and want to install ECget in your default working environment (or you don't know about creating alternate environments with **`conda create`**), these are the instructions for you:

1. Use Mercurial to clone the project repo from Bitbucket; i.e.

```
$ hg clone https://bitbucket.org/douglatornell/ecget
```

2. Use **`conda`** to install the Python packages that ECget depends on that are part of the Anaconda distribution:

```
$ conda install pip requests beautiful-soup six
```

3. Use **`pip`** to install from PyPI the Python packages that ECget depends on that are *not* part of the Anaconda distribution:

```
$ pip install arrow cliff kombu
```

1. Use **`pip`** to install ECget in editable mode so that updates that you pull from the Bitbucket repo will take effect immediately:

```
$ pip install --editable ./ecget
```

1.4.2 Install in a New `conda` Environment

1. Use Mercurial to clone the project repo from Bitbucket; i.e.

```
$ hg clone https://bitbucket.org/douglatornell/ecget
```

2. Use **`conda`** to create a new Python 3.4 environment and install the Python packages that ECget depends on that are part of the Anaconda distribution:

```
$ conda create -n ecget python=3.4 pip requests beautiful-soup six
```

3. Activate the `ecget` environment:

```
$ source activate ecget
```

4. Use **`pip`** to install from PyPI the Python packages that ECget depends on that are *not* part of the Anaconda distribution:

```
(ecget)$ pip install arrow cliff kombu
```

5. Use **`pip`** to install ECget in editable mode so that updates that you pull from the Bitbucket repo will take effect immediately:

```
(ecget)$ pip install --editable ./ecget
```

When you are finished using ECget you can deactivate the environment with:

```
(ecget)$ source deactivate
```

1.4.3 Install in a Python 3.4 Virtual Environment

1. Use **`pyvenv-3.4`** to create a new Python 3.4 virtual environment and activate it:

```
$ pyvenv-3.4 ecget-venv
$ cd ecget-venv
$ source bin/activate
```

2. Use Mercurial to clone the project repo from Bitbucket; i.e.

```
(ecget-venv)$ hg clone https://bitbucket.org/douglatornell/ecget
```

3. **Use `pip` to install from PyPI the Python packages that ECget depends on,** and install ECget in editable mode so that updates that you pull from the Bitbucket repo will take effect immediately:

```
(ecget-venv)$ pip install --editable ./ecget
```

When you are finished using ECget you can deactivate the environment with:

```
(ecget-venv)$ deactivate
```

CHAPTER 2

For Developers

If you would like to contribute to ECget directly, these instructions should help you get started. Bug reports and feature requests are all welcome through the Bitbucket project.

Changes to ECget should be submitted as pull requests on Bitbucket project.

Bugs should be files under the Bitbucket project.

Note: Before contributing new features to ECget, please consider whether they should be implemented as an extension instead. The architecture is highly pluggable precisely to keep the core small.

2.1 Development Environment

ECget is developed under Python 3.3 and tested under Python 2.7 and Python 3.2. Setting up a Python 3.3 virtualenv via pyvenv is a little tricky because pyvenv doesn't install/include pip and setuptools. These commands should result in a viable, working Python 3.3 virtual environment:

```
pyvenv-3.3 ecget
cd ecget
(ecget)$ . bin/activate
(ecget)$ curl -O https://raw.github.com/pypa/pip/master/contrib/get-pip.py
(ecget)$ python3.3 get-pip.py
```

Thanks to Richard Jones for those commands.

After cloning the source code repo from the Bitbucket project, the Python packages at the versions used for development at tip can be installed with:

```
(ecget)$ pip install -r requirements.txt
```

Install the ECget package for development with:

```
(ecget)$ cd ecget
(ecget)$ pip install -e .
```

or

```
(ecget)$ cd ecget
(ecget)$ python setup.py develop
```

Note: Because ECget uses setuptools entry points for plug-in discovery it is necessary to install the package whenever entry points are changed or added in `setup.py`.

2.2 Building Documentation

The documentation for ECget is written in reStructuredText and converted to HTML using Sphinx. The build itself is driven by make. Installing the development packages via the `requirements.txt` file as described above will install Sphinx. Once that has been done use:

```
(ecget)$ (cd docs && make clean html)
rm -rf _build/*
sphinx-build -b html -d _build/doctrees   . _build/html
Making output directory...
Running Sphinx v1.2.1
loading pickled environment... done
building [html]: targets for 3 source files that are out of date
updating environment: 3 added, 0 changed, 0 removed
reading sources... [100%] install
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] install
writing additional files... (0 module code pages) genindex search
copying static files... done
copying extra files... done
dumping search index... done
dumping object inventory... done
build succeeded.

Build finished. The HTML pages are in _build/html.
```

to generate the HTML version of the documentation. The output ends up in `./docs/_build/html/` in your development directory.

2.3 Running Tests

The test suite for ECget uses pytest, coverage, and tox. Installing the development packages via the `requirements.txt` file as described in the *Development Environment* section above will install those packages and their dependencies.

Use the **`py.test`** command from the top level directory of the Mercurial repository to run the test suite in the development environment:

```
(ecget)$ py.test
====================================== test session starts
↪=======================================
platform darwin -- Python 3.3.2 -- pytest-2.5.1
collected 1 items

tests/test_SOG_formatters.py .

=================================== 1 passed in 0.13 seconds
↪====================================
```

To gather test coverage data use **coverage run -m py.test** and view the coverage report with **coverage report**:

```
(ecget)$ coverage report
Name                          Stmts   Miss Branch BrMiss  Cover   Missing
-------------------------------------------------------------------------
ecget/SOG_formatters              9      0      4      2    85%
ecget/__init__                    0      0      0      0   100%
ecget/main                       12     12      2      2     0%   22-45
ecget/river                     110    110     24     24     0%   18-229
tests/test_SOG_formatters         9      0      0      0   100%
-------------------------------------------------------------------------
TOTAL                           140    122     30     28    12%
```

The default **coverage** run and report option values are set in the `.coveragerc` file.

Use the **tox** command to run the tests under Python 3.3, 3.2, and 2.7.

Note: You must have all of those versions of Python installed on your system for **tox** to succeed.

To run the tests under a single version of Python, specify the appropriate environment when running **tox**:

```
(ecget)$ tox -e py27
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search