Requirements-Builder Documentation

Release 0.2.1.dev20160913

Invenio Collaboration

September 27, 2016

Contents

CHAPTER 1

About

Build requirements from setup.py to test your package against minimum, latest and development versions of your package dependencies. Particularly useful when combined with your CI systems build matrix.

CHAPTER 2

Installation

Requirements-Builder is on PyPI so all you need is:

```
$ pip install requirements-builder
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv requirements-builder
$ pip install requirements-builder
```

CHAPTER 3

Testing

Running the test suite is as simple as:

```
$ ./run-tests.sh
```

CHAPTER 4

Usage

Build requirements files from setup.py requirements.

```
$ requirements-builder --help
Usage: requirements-builder [OPTIONS] SETUP

  Calculate requirements for different purposes.

Options:
  -l, --level [min|pypi|dev]  Specifies desired requirements level."min"
                               requests the minimal requirement that is
                               specified, "pypi" requests the maximum version
                               that satisfies the constrains and is available
                               in PyPi. "dev" includes experimental developer
                               versions for VCSs.
  -e, --extras TEXT            Comma separated list of extras.
  -r, --req PATH               Requirements file.
  --help                       Show this message and exit.
```

4.1 TravisCI

Following is an example of how to integrate Requirements-Builder with TravisCI:

```
env:
  - REQUIREMENTS=lowest
  - REQUIREMENTS=release
  - REQUIREMENTS=devel

python:
  - "2.7"
  - "3.3"
  - "3.4"
  - "3.5"

before_install:
  - "travis_retry pip install --upgrade pip"
  - "travis_retry pip install requirements-builder"
  - "requirements-builder --level=min setup.py
     > .travis-lowest-requirements.txt"
  - "requirements-builder --level=pypi setup.py
     > .travis-release-requirements.txt"
  - "requirements-builder --level=dev --req requirements-devel.txt setup.py
```

```
    > .travis-devel-requirements.txt"

install:
  - "travis_retry pip install -r .travis-$REQUIREMENTS-requirements.txt"
  - "pip install -e ."
```

CHAPTER 5

API Reference

Generate requirements from *setup.py* and *requirements-devel.txt*.

requirements_builder.requirements_builder.**iter_requirements**(*level*, *extras*, *pip_file*, *setup_fp*)
Iterate over requirements.

requirements_builder.requirements_builder.**minver_error**(*pkg_name*)
Report error about missing minimum version constraint and exit.

requirements_builder.requirements_builder.**parse_pip_file**(*path*)
Parse pip requirements file.

requirements_builder.requirements_builder.**parse_set**(*string*)
Parse set from comma separated string.

CHAPTER 6

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

6.1 Types of Contributions

6.1.1 Report Bugs

Report bugs at https://github.com/inveniosoftware/requirements-builder/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.

6.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

6.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

6.1.4 Write Documentation

Requirements-Builder could always use more documentation, whether as part of the official Requirements-Builder docs, in docstrings, or even on the web in blog posts, articles, and such.

6.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/inveniosoftware/requirements-builder/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 :)

6.2 Get Started!

Ready to contribute? Here's how to set up *requirements-builder* for local development.

1. Fork the *requirements-builder* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/requirements-builder.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 requirements-builder
$ cd requirements-builder/
$ 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 tests, including testing other Python versions with tox:

```
$ ./run-tests.sh
$ tox
```

   The tests will provide you with test coverage and also check PEP8 (code style), PEP257 (documentation), flake8 as well as build the Sphinx documentation and run doctests.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -s -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.

6.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests and must not decrease test coverage.
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, 3.5 and for PyPy. Check https://travis-ci.org/inveniosoftware/requirements-builder/pull_requests and make sure that the tests pass for all supported Python versions.

CHAPTER 7

Changes

7.1 Version 0.2.0 (released 2016-09-13)

7.1.1 New features

- Adds an output option which is useful in the tox context where one cannot redirect the output to a file. See more at https://bitbucket.org/hpk42/tox/issues/73/pipe-output-of-command-into-file

7.1.2 Bug fixes

- Fixes problem when the setup.py command try to import the package its about to install in order to get the information like the version. E.g. Django does that.
- Fixes problem when the setup.py command plays with *__file__* to read, exec, or whatever.

7.2 Version 0.1.0 (released 2015-10-05)

- Initial public release

CHAPTER 8

License

Requirements-Builder is free software; you can redistribute it and/or modify it under the terms of the Revised BSD License; see LICENSE file for more details.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

8.1 Authors

- Jiri Kuncar <jiri.kuncar@cern.ch>
- Lars Holm Nielsen <lars.holm.nielsen@cern.ch>
- Marco Neumann <marco@crepererum.net>
- Tibor Simko <tibor.simko@cern.ch>
- Yoan Blanc <yoan@dosimple.ch>

r

I

M

P

R