dh-virtualenv Documentation

Release 0.7

Spotify AB

July 21, 2015

Contents

Contents:

CHAPTER 1

What is dh-virtualenv

`dh-virtualenv` is a tool that aims to combine Debian packaging with self-contained virtualenv based Python deployments. To do this, the package extends debhelper's sequence by providing a new command in sequence, `dh_virtualenv`, which effectively replaces following commands from the sequence:

- `dh_auto_install`
- `dh_python2`
- `dh_pycentral`
- `dh_pysupport`

In the sequence the `dh_virtualenv` is inserted right after `dh_perl`.

CHAPTER 2

Changelog

Following list contains most notable changes by version. For full list consult the git history of the project.

2.1 0.7 (unreleased)

- **Backwards incompatible** Support running tests. This change breaks builds that use distutils. For those cases a flag `--no-test` needs to be passed.
- Add tutorial to documentation
- Don't crash on debbuild parameters `-i` and `-a`
- Support custom source directory (debhelper's flag `-D`)

2.2 0.6

First public release of *dh-virtualenv*

CHAPTER 3

Tutorial

This tutorial will guide you through setting up your first project using *dh-virtualenv*. Having some knowledge on how Debian packages work won't hurt, but it is not necessary a mandatory requirement. You also need some basic build tools, so it is recommended to install *build-essential* and *devscripts* packages.

3.1 Step 1: Install dh-virtualenv

In order to use it, you need to install the *dh-virtualenv*. If you run Debian Jessie (testing), Debian Sid (unstable) or Ubuntu 14.04 LTS (Trusty), you can install *dh-virtualenv* simply with *apt-get*:

```
sudo apt-get install dh-virtualenv
```

For other systems the only way is to build and install it yourself. Steps to do that, after you have cloned the repository are:

```
sudo apt-get install devscripts python-virtualenv git equivs # Install needed packages
git clone https://github.com/spotify/dh-virtualenv.git       # Clone Git repository
cd dh-virtualenv                                              # Move into the repository
sudo mk-build-deps -ri                                        # This will install build dependencies
dpkg-buildpackage -us -uc -b                                  # Build the *dh-virtualenv* package

# and finally, install it (you might have to solve some
# dependencies when doing this):
sudo dpkg -i ../dh-virtualenv_<version>.deb
```

3.2 Step 2: Setup the Debian packaging

Grab your favourite Python project you want to use *dh-virtualenv* with and set it up. Only requirement is that your project has a somewhat sane `setup.py` and requirements listed in a `requirements.txt` file. Note however that the defining requirements is not mandatory.

Next you need to define the Debian packaging for your software. To do this, create a directory called `debian` in the project root.

To be able to build a debian package, a few files are needed. First, we need to define the compatibility level of the project. For this, do:

```
echo "9" > debian/compat
```

The 9 is a magic number for latest compatibility level, but we don't need to worry about that. Next we need a file that tells what our project is about, a file called `control`. Enter a following `debian/control` file:

```
Source: my-awesome-python-software
Section: python
Priority: extra
Maintainer: Matt Maintainer <matt@example.com>
Build-Depends: debhelper (>= 9), python, dh-virtualenv
Standards-Version: 3.9.5

Package: my-awesome-python-software
Architecture: any
Depends: ${python:Depends}, ${misc:Depends}
Description: really neat package!
 second line can contain extra information about it.
```

The `control` file is used to define the build dependencies, so if you are building a package that requires for example `lxml`, make sure you define `libxml2-dev` in *Build-Depends* etc.

Depends in the lower section is used to define run-time dependencies. Following the example above, in case of lxml you would add `libxml2` in to the *Depends* field.

Next, we need a changelog file. It is basically a documentation of changes in your package plus the source for version number for Debian package builder. Here's a short sample changelog to be entered in `debian/changelog`:

```
my-awesome-python-software (0.1-1) unstable; urgency=low

  * Initial public release

 -- Matt Maintainer <matt@example.com>  Fri, 01 Nov 2013 17:00:00 +0200
```

You don't need to create this file by hand, a handy tool called `dch` exists for entering new changelog entries.

Now, last bit is left, which is the `debian/rules` file. This file is basically a Makefile that Debian uses to build the package. Content for that is fairly straightforward:

```
#!/usr/bin/make -f

%:
	dh $@ --with python-virtualenv
```

And there we go, debianization of your new package is ready!

3.3 Step 3: Build your project

Now you can just build your project by running `dpkg-buildpackage -us -uc`. Enjoy your newly baked *dh-virtualenv* backed project! :)

CHAPTER 4

Building packages with dh-virtualenv

Building packages with *dh-virtualenv* is relatively easy to start with but it also supports lot of customization to fit in your general needs.

By default, *dh-virtualenv* installs your packages under `/usr/share/python/<packagename>`. The package name is provided by the `debian/control` file.

4.1 Simple usecase

To signal debhelper to use *dh-virtualenv* for building your package, you need to pass `--with python-virtualenv` to debhelper sequencer.

In a nutshell, the simplest `debian/rules` file to build using *dh-virtualenv* looks like this:

```
#!/usr/bin/make -f

%:
	dh $@ --with python-virtualenv
```

However, the tool makes a few assumptions of your project's structure:

- For installing requirements, you need to have a file called `requirements.txt` in the root directory of your project. The requirements file is not mandatory.
- The project must have a `setup.py` file in the root of the project. Sequencer will run `setup.py install` to install the package inside the virtualenv.

After these are place, you can just build the package with your favorite tool!

4.2 Command line options

To change the default behavior the `dh_virtualenv` command accepts a few command line options:

`-p` `<package>`**`, --package`** `<package>`
: Act on the package named *<package>*

`-N` `<package>`**`, --no-package`** `<package>`
: Do not act on the specified package

`-v, --verbose`
: Turn on verbose mode. This has a few effects: it sets root logger level to `DEBUG` and passes verbose flag to `pip` when installing packages. This can also be provided using the standard `DH_VERBOSE` environment variable.

--extra-index-url <url>
: Use extra index url **<*url*>** when running `pip` to install packages. This can be provided multiple times to pass multiple URLs to `pip`. This is useful if you for example have a private Python Package Index.

--preinstall <package>
: Package to install before processing the requirements. This flag can be used to provide a package that is installed by `pip` before processing requirements file. This is handy if you need to install for example a custom setup script or other packages needed to parse `setup.py`. This flag can be provided multiple times to pass multiple packages for pre-install.

--pypi-url <URL>
: Base URL of the PyPI server. This flag can be used to pass in a custom URL to a PyPI mirror. It's useful if you for example have an internal mirror of the PyPI or you run a special instance that only exposes selected packages of PyPI. If this is not provided, the default will be whatever `pip` uses as default (usually `http://pypi.python.org/simple`).

--setuptools
: Use setuptools instead of distribute in the virtualenv

--no-test
: Skip running `python setup.py test` after dependencies and the package is installed. This is useful if the PYthon code is packaged using distutils and not setuptools.

4.3 Advanced usage

To provide command line options to `dh_virtualenv` sequence the override mechanism of the debhelper is the best tool.

Following `debian/rules` will provide *http://example.com* as additional Python Package Index URI:

```
#!/usr/bin/make -f

%:
	dh $@ --with python-virtualenv

override_dh_virtualenv:
	dh_virtualenv --extra-index-url http://example.com
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Symbols

C