virtualenv

Release 15.2.0.dev0

October 07, 2017

Contents

Mailing list | Issues | Github | PyPI | User IRC: #pypa Dev IRC: #pypa-dev

CHAPTER 1

Introduction

`virtualenv` is a tool to create isolated Python environments.

The basic problem being addressed is one of dependencies and versions, and indirectly permissions. Imagine you have an application that needs version 1 of LibFoo, but another application requires version 2. How can you use both these applications? If you install everything into `/usr/lib/python2.7/site-packages` (or whatever your platform's standard location is), it's easy to end up in a situation where you unintentionally upgrade an application that shouldn't be upgraded.

Or more generally, what if you want to install an application *and leave it be*? If an application works, any change in its libraries or the versions of those libraries can break the application.

Also, what if you can't install packages into the global `site-packages` directory? For instance, on a shared host.

In all these cases, `virtualenv` can help you. It creates an environment that has its own installation directories, that doesn't share libraries with other virtualenv environments (and optionally doesn't access the globally installed libraries either).

Installation

Warning: We advise installing virtualenv-1.9 or greater. Prior to version 1.9, the pip included in virtualenv did not download from PyPI over SSL.

Warning: When using pip to install virtualenv, we advise using pip 1.3 or greater. Prior to version 1.3, pip did not download from PyPI over SSL.

Warning: We advise against using easy_install to install virtualenv when using setuptools < 0.9.7, because easy_install didn't download from PyPI over SSL and was broken in some subtle ways.

To install globally with *pip* (if you have pip 1.3 or greater installed globally):

```
$ [sudo] pip install virtualenv
```

Or to get the latest unreleased dev version:

```
$ [sudo] pip install https://github.com/pypa/virtualenv/tarball/master
```

To install version X.X globally from source:

```
$ curl -O https://pypi.python.org/packages/source/v/virtualenv/virtualenv-X.X.tar.gz
$ tar xvfz virtualenv-X.X.tar.gz
$ cd virtualenv-X.X
$ [sudo] python setup.py install
```

To *use* locally from source:

```
$ curl -O https://pypi.python.org/packages/source/v/virtualenv/virtualenv-X.X.tar.gz
$ tar xvfz virtualenv-X.X.tar.gz
$ cd virtualenv-X.X
$ python virtualenv.py myVE
```

Note: The `virtualenv.py` script is *not* supported if run without the necessary pip/setuptools/virtualenv distributions available locally. All of the installation methods above include a `virtualenv_support` directory alongside `virtualenv.py` which contains a complete set of pip and setuptools distributions, and so are fully supported.

User Guide

Usage

Virtualenv has one basic command:

```
$ virtualenv ENV
```

Where `ENV` is a directory to place the new virtual environment. It has a number of usual effects (modifiable by many *Options*):

- `ENV/lib/` and `ENV/include/` are created, containing supporting library files for a new virtualenv python. Packages installed in this environment will live under `ENV/lib/pythonX.X/site-packages/`.
- `ENV/bin` is created, where executables live - noticeably a new **`python`**. Thus running a script with `#! /path/to/ENV/bin/python` would run that script under this virtualenv's python.
- The crucial packages pip and setuptools are installed, which allow other packages to be easily installed to the environment. This associated pip can be run from `ENV/bin/pip`.

The python in your new virtualenv is effectively isolated from the python that was used to create it.

activate script

In a newly created virtualenv there will also be a **`activate`** shell script. For Windows systems, activation scripts are provided for the Command Prompt and Powershell.

On Posix systems, this resides in `/ENV/bin/`, so you can run:

```
$ source bin/activate
```

For some shells (e.g. the original Bourne Shell) you may need to use the **.** command, when **source** does not exist. There are also separate activate files for some other shells, like csh and fish. `bin/activate` should work for bash/zsh/dash.

This will change your `$PATH` so its first entry is the virtualenv's `bin/` directory. (You have to use `source` because it changes your shell environment in-place.) This is all it does; it's purely a convenience. If you directly run a script or the python interpreter from the virtualenv's `bin/` directory (e.g. `path/to/ENV/bin/pip` or `/path/to/ENV/bin/python-script.py`) there's no need for activation.

The `activate` script will also modify your shell prompt to indicate which environment is currently active. To disable this behaviour, see *VIRTUAL_ENV_DISABLE_PROMPT*.

To undo these changes to your path (and prompt), just run:

```
$ deactivate
```

On Windows, the equivalent *activate* script is in the `Scripts` folder:

```
> \path\to\env\Scripts\activate
```

And type `deactivate` to undo the changes.

Based on your active shell (CMD.exe or Powershell.exe), Windows will use either activate.bat or activate.ps1 (as appropriate) to activate the virtual environment. If using Powershell, see the notes about code signing below.

Note: If using Powershell, the `activate` script is subject to the execution policies on the system. By default on Windows 7, the system's excution policy is set to `Restricted`, meaning no scripts like the `activate` script are allowed to be executed. But that can't stop us from changing that slightly to allow it to be executed.

In order to use the script, you can relax your system's execution policy to `AllSigned`, meaning all scripts on the system must be digitally signed to be executed. Since the virtualenv activation script is signed by one of the authors (Jannis Leidel) this level of the execution policy suffices. As an administrator run:

```
PS C:\> Set-ExecutionPolicy AllSigned
```

Then you'll be asked to trust the signer, when executing the script. You will be prompted with the following:

```
PS C:\> virtualenv .\foo
New python executable in C:\foo\Scripts\python.exe
Installing setuptools.............done.
Installing pip...................done.
PS C:\> .\foo\scripts\activate

Do you want to run software from this untrusted publisher?
File C:\foo\scripts\activate.ps1 is published by E=jannis@leidel.info,
CN=Jannis Leidel, L=Berlin, S=Berlin, C=DE, Description=581796-Gh7xfJxkxQSIO4E0
and is not trusted on your system. Only run scripts from trusted publishers.
[V] Never run  [D] Do not run  [R] Run once  [A] Always run  [?] Help
(default is "D"):A
(foo) PS C:\>
```

If you select `[A] Always Run`, the certificate will be added to the Trusted Publishers of your user account, and will be trusted in this user's context henceforth. If you select `[R] Run Once`, the script will be run, but you will be prompted on a subsequent invocation. Advanced users can add the signer's certificate to the Trusted Publishers of the Computer account to apply to all users (though this technique is out of scope of this document).

Alternatively, you may relax the system execution policy to allow running of local scripts without verifying the code signature using the following:

```
PS C:\> Set-ExecutionPolicy RemoteSigned
```

Since the `activate.ps1` script is generated locally for each virtualenv, it is not considered a remote script and can then be executed.

Removing an Environment

Removing a virtual environment is simply done by deactivating it and deleting the environment folder with all its contents:

```
(ENV)$ deactivate
$ rm -r /path/to/ENV
```

The `--system-site-packages` Option

If you build with `virtualenv --system-site-packages ENV`, your virtual environment will inherit packages from `/usr/lib/python2.7/site-packages` (or wherever your global site-packages directory is).

This can be used if you have control over the global site-packages directory, and you want to depend on the packages there. If you want isolation from the global system, do not use this flag.

Windows Notes

Some paths within the virtualenv are slightly different on Windows: scripts and executables on Windows go in `ENV\Scripts\` instead of `ENV/bin/` and libraries go in `ENV\Lib\` rather than `ENV/lib/`.

To create a virtualenv under a path with spaces in it on Windows, you'll need the win32api library installed.

Using Virtualenv without `bin/python`

Sometimes you can't or don't want to use the Python interpreter created by the virtualenv. For instance, in a mod_python or mod_wsgi environment, there is only one interpreter.

Luckily, it's easy. You must use the custom Python interpreter to *install* libraries. But to *use* libraries, you just have to be sure the path is correct. A script is available to correct the path. You can setup the environment like:

```
activate_this = '/path/to/env/bin/activate_this.py'
execfile(activate_this, dict(__file__=activate_this))
```

This will change `sys.path` and even change `sys.prefix`, but also allow you to use an existing interpreter. Items in your environment will show up first on `sys.path`, before global items. However, global items will always be accessible (as if the *`--system-site-packages`* flag had been used in creating the environment, whether it was or not). Also, this cannot undo the activation of other environments, or modules that have been imported. You shouldn't try to, for instance, activate an environment before a web request; you should activate *one* environment as early as possible, and not do it again in that process.

Making Environments Relocatable

Note: this option is somewhat experimental, and there are probably caveats that have not yet been identified.

> **Warning:** The `--relocatable` option currently has a number of issues, and is not guaranteed to work in all circumstances. It is possible that the option will be deprecated in a future version of `virtualenv`.

Normally environments are tied to a specific path. That means that you cannot move an environment around or copy it to another computer. You can fix up an environment to make it relocatable with the command:

```
$ virtualenv --relocatable ENV
```

This will make some of the files created by setuptools use relative paths, and will change all the scripts to use `activate_this.py` instead of using the location of the Python interpreter to select the environment.

Note: scripts which have been made relocatable will only work if the virtualenv is activated, specifically the python executable from the virtualenv must be the first one on the system PATH. Also note that the activate scripts are not currently made relocatable by `virtualenv --relocatable`.

Note: you must run this after you've installed *any* packages into the environment. If you make an environment relocatable, then install a new package, you must run `virtualenv --relocatable` again.

Also, this **does not make your packages cross-platform**. You can move the directory around, but it can only be used on other similar computers. Some known environmental differences that can cause incompatibilities: a different version of Python, when one platform uses UCS2 for its internal unicode representation and another uses UCS4 (a compile-time option), obvious platform changes like Windows vs. Linux, or Intel vs. ARM, and if you have libraries that bind to C libraries on the system, if those C libraries are located somewhere different (either different versions, or a different filesystem layout).

If you use this flag to create an environment, currently, the *`--system-site-packages`* option will be implied.

The `--extra-search-dir` option

This option allows you to provide your own versions of setuptools and/or pip to use instead of the embedded versions that come with virtualenv.

To use this feature, pass one or more `--extra-search-dir` options to virtualenv like this:

```
$ virtualenv --extra-search-dir=/path/to/distributions ENV
```

The `/path/to/distributions` path should point to a directory that contains setuptools and/or pip wheels.

virtualenv will look for wheels in the specified directories, but will use pip's standard algorithm for selecting the wheel to install, which looks for the latest compatible wheel.

As well as the extra directories, the search order includes:

1. The `virtualenv_support` directory relative to virtualenv.py
2. The directory where virtualenv.py is located.
3. The current directory.

Reference Guide

`virtualenv` Command

Usage

`virtualenv [OPTIONS] ENV_DIR`

Where `ENV_DIR` is an absolute or relative path to a directory to create the virtual environment in.

Options

`--version`
show program's version number and exit

`-h, --help`
show this help message and exit

`-v, --verbose`
Increase verbosity.

`-q, --quiet`
Decrease verbosity.

`-p` `PYTHON_EXE`**`, --python`**`=PYTHON_EXE`
The Python interpreter to use, e.g., –python=python2.5 will use the python2.5 interpreter to create the new environment. The default is the interpreter that virtualenv was installed with (like `/usr/bin/python`)

`--clear`
Clear out the non-root install and start from scratch.

`--system-site-packages`
Give the virtual environment access to the global site-packages.

`--always-copy`
Always copy files rather than symlinking.

`--relocatable`
Make an EXISTING virtualenv environment relocatable. This fixes up scripts and makes all .pth files relative.

`--unzip-setuptools`
Unzip Setuptools when installing it.

`--no-setuptools`
Do not install setuptools in the new virtualenv.

`--no-pip`
Do not install pip in the new virtualenv.

`--no-wheel`
Do not install wheel in the new virtualenv.

`--extra-search-dir``=DIR`
Directory to look for setuptools/pip distributions in. This option can be specified multiple times.

`--prompt``=PROMPT`
Provides an alternative prompt prefix for this environment.

`--download`
Download preinstalled packages from PyPI.

--no-download
Do not download preinstalled packages from PyPI.

--no-site-packages
DEPRECATED. Retained only for backward compatibility. Not having access to global site-packages is now the default behavior.

--distribute

--setuptools
Legacy; now have no effect. Before version 1.10 these could be used to choose whether to install Distribute or Setuptools into the created virtualenv. Distribute has now been merged into Setuptools, and the latter is always installed.

Configuration

Environment Variables

Each command line option is automatically used to look for environment variables with the name format `VIRTUALENV_<UPPER_NAME>`. That means the name of the command line options are capitalized and have dashes (`'-'`) replaced with underscores (`'_'`).

For example, to automatically use a custom Python binary instead of the one virtualenv is run with you can also set an environment variable:

```
$ export VIRTUALENV_PYTHON=/opt/python-3.3/bin/python
$ virtualenv ENV
```

It's the same as passing the option to virtualenv directly:

```
$ virtualenv --python=/opt/python-3.3/bin/python ENV
```

This also works for appending command line options, like `--find-links`. Just leave an empty space between the passed values, e.g.:

```
$ export VIRTUALENV_EXTRA_SEARCH_DIR="/path/to/dists /path/to/other/dists"
$ virtualenv ENV
```

is the same as calling:

```
$ virtualenv --extra-search-dir=/path/to/dists --extra-search-dir=/path/to/other/
↪dists ENV
```

VIRTUAL_ENV_DISABLE_PROMPT
Any virtualenv created when this is set to a non-empty value will not have it's *activate script* modify the shell prompt.

Configuration File

virtualenv also looks for a standard ini config file. On Unix and Mac OS X that's `$HOME/.virtualenv/virtualenv.ini` and on Windows, it's `%APPDATA%\virtualenv\virtualenv.ini`.

The names of the settings are derived from the long command line option, e.g. the option *--python* would look like this:

```
[virtualenv]
python = /opt/python-3.3/bin/python
```

Appending options like *--extra-search-dir* can be written on multiple lines:

```
[virtualenv]
extra-search-dir =
    /path/to/dists
    /path/to/other/dists
```

Please have a look at the output of *--help* for a full list of supported options.

Extending Virtualenv

Creating Your Own Bootstrap Scripts

While this creates an environment, it doesn't put anything into the environment. Developers may find it useful to distribute a script that sets up a particular environment, for example a script that installs a particular web application.

To create a script like this, call `virtualenv.create_bootstrap_script()`, and write the result to your new bootstrapping script.

create_bootstrap_script(*extra_text*)
Creates a bootstrap script from `extra_text`, which is like this script but with extend_parser, adjust_options, and after_install hooks.

This returns a string that (written to disk of course) can be used as a bootstrap script with your own customizations. The script will be the standard virtualenv.py script, with your extra text added (your extra text should be Python code).

If you include these functions, they will be called:

extend_parser(*optparse_parser*)
You can add or remove options from the parser here.

adjust_options(*options*, *args*)
You can change options here, or change the args (if you accept different kinds of arguments, be sure you modify `args` so it is only `[DEST_DIR]`).

after_install(*options*, *home_dir*)
After everything is installed, this function is called. This is probably the function you are most likely to use. An example would be:

```
def after_install(options, home_dir):
    if sys.platform == 'win32':
        bin = 'Scripts'
    else:
        bin = 'bin'
    subprocess.call([join(home_dir, bin, 'easy_install'),
                     'MyPackage'])
    subprocess.call([join(home_dir, bin, 'my-package-script'),
                     'setup', home_dir])
```

This example immediately installs a package, and runs a setup script from that package.

Bootstrap Example

Here's a more concrete example of how you could use this:

```
import virtualenv, textwrap
output = virtualenv.create_bootstrap_script(textwrap.dedent("""
import os, subprocess
def after_install(options, home_dir):
    etc = join(home_dir, 'etc')
    if not os.path.exists(etc):
        os.makedirs(etc)
    subprocess.call([join(home_dir, 'bin', 'easy_install'),
                     'BlogApplication'])
    subprocess.call([join(home_dir, 'bin', 'paster'),
                     'make-config', 'BlogApplication',
                     join(etc, 'blog.ini')])
    subprocess.call([join(home_dir, 'bin', 'paster'),
                     'setup-app', join(etc, 'blog.ini')])
"""))
f = open('blog-bootstrap.py', 'w').write(output)
```

Another example is available here.

Development

Contributing

Refer to the pip development documentation - it applies equally to virtualenv, except that virtualenv issues should filed on the virtualenv repo at GitHub.

Virtualenv's release schedule is tied to pip's – each time there's a new pip release, there will be a new virtualenv release that bundles the new version of pip.

Files in the *virtualenv_embedded/* subdirectory are embedded into *virtualenv.py* itself as base64-encoded strings (in order to support single-file use of *virtualenv.py* without installing it). If your patch changes any file in *virtualenv_embedded/*, run *bin/rebuild-script.py* to update the embedded version of that file in *virtualenv.py*; commit that and submit it as part of your patch / pull request.

Running the tests

Virtualenv's test suite is small and not yet at all comprehensive, but we aim to grow it.

The easy way to run tests (handles test dependencies automatically):

```
$ python setup.py test
```

If you want to run only a selection of the tests, you'll need to run them directly with pytest instead. Create a virtualenv, and install required packages:

```
$ pip install pytest mock
```

Run pytest:

```
$ pytest
```

Or select just a single test file to run:

```
$ pytest tests/test_virtualenv
```

Status and License

`virtualenv` is a successor to workingenv, and an extension of virtual-python.

It was written by Ian Bicking, sponsored by the Open Planning Project and is now maintained by a group of developers. It is licensed under an MIT-style permissive license.

Release History

15.2.0 (unreleased)

15.1.0 (2016-11-15)

- Support Python 3.6.
- Upgrade setuptools to 28.0.0.
- Upgrade pip to 9.0.1.
- Don't install pre-release versions of pip, setuptools, or wheel from PyPI.

15.0.3 (2016-08-05)

- Test for given python path actually being an executable *file*, #939
- Only search for copy actual existing Tcl/Tk directories (PR #937)
- Generically search for correct Tcl/Tk version (PR #926, PR #933)
- Upgrade setuptools to 22.0.5

15.0.2 (2016-05-28)

- Copy Tcl/Tk libs on Windows to allow them to run, fixes #93 (PR #888)
- Upgrade setuptools to 21.2.1.
- Upgrade pip to 8.1.2.

15.0.1 (2016-03-17)

- Print error message when DEST_DIR exists and is a file
- Upgrade setuptools to 20.3
- Upgrade pip to 8.1.1.

15.0.0 (2016-03-05)

- Remove the *virtualenv-N.N* script from the package; this can no longer be correctly created from a wheel installation. Resolves #851, #692
- Remove accidental runtime dependency on pip by extracting certificate in the subprocess.
- Upgrade setuptools 20.2.2.
- Upgrade pip to 8.1.0.

14.0.6 (2016-02-07)

- Upgrade setuptools to 20.0
- Upgrade wheel to 0.29.0
- Fix an error where virtualenv didn't pass in a working ssl certificate for pip, causing "weird" errors related to ssl.

14.0.5 (2016-02-01)

- Homogenize drive letter casing for both prefixes and filenames. #858

14.0.4 (2016-01-31)

- Upgrade setuptools to 19.6.2
- Revert ac4ea65; only correct drive letter case. Fixes #856, #815

14.0.3 (2016-01-28)

- Upgrade setuptools to 19.6.1

14.0.2 (2016-01-28)

- Upgrade setuptools to 19.6
- Supress any errors from *unset* on different shells (PR #843)
- Normalize letter case for prefix path checking. Fixes #837

14.0.1 (2016-01-21)

- Upgrade from pip 8.0.0 to 8.0.2.
- Fix the default of `--(no-)download` to default to downloading.

14.0.0 (2016-01-19)

- **BACKWARDS INCOMPATIBLE** Drop support for Python 3.2.
- Upgrade setuptools to 19.4
- Upgrade wheel to 0.26.0
- Upgrade pip to 8.0.0
- Upgrade argparse to 1.4.0
- Added support for `python-config` script (PR #798)
- Updated activate.fish (PR #589) (PR #799)
- Account for a `site.pyo` correctly in some python implementations (PR #759)
- Properly restore an empty PS1 (#407)
- Properly remove `pydoc` when deactivating
- Remove workaround for very old Mageia / Mandriva linuxes (PR #472)
- Added a space after virtualenv name in the prompt: `(env) $PS1`
- Make sure not to run a –user install when creating the virtualenv (PR #803)
- Remove virtualenv.py's path from sys.path when executing with a new python. Fixes issue #779, #763 (PR #805)
- Remove use of () in .bat files so `Program Files (x86)` works #35
- Download new releases of the preinstalled software from PyPI when there are new releases available. This behavior can be disabled using `--no-download`.
- Make `--no-setuptools`, `--no-pip`, and `--no-wheel` independent of each other.

13.1.2 (2015-08-23)

- Upgrade pip to 7.1.2.

13.1.1 (2015-08-20)

- Upgrade pip to 7.1.1.
- Upgrade setuptools to 18.2.
- Make the activate script safe to use when bash is running with `-u`.

13.1.0 (2015-06-30)

- Upgrade pip to 7.1.0
- Upgrade setuptools to 18.0.1

13.0.3 (2015-06-01)

- Upgrade pip to 7.0.3

13.0.2 (2015-06-01)

- Upgrade pip to 7.0.2
- Upgrade setuptools to 17.0

13.0.1 (2015-05-22)

- Upgrade pip to 7.0.1

13.0.0 (2015-05-21)

- Automatically install wheel when creating a new virutalenv. This can be disabled by using the `--no-wheel` option.
- Don't trust the current directory as a location to discover files to install packages from.
- Upgrade setuptools to 16.0.
- Upgrade pip to 7.0.0.

12.1.1 (2015-04-07)

- Upgrade pip to 6.1.1

12.1.0 (2015-04-07)

- Upgrade setuptools to 15.0
- Upgrade pip to 6.1.0

12.0.7 (2015-02-04)

- Upgrade pip to 6.0.8

12.0.6 (2015-01-28)

- Upgrade pip to 6.0.7
- Upgrade setuptools to 12.0.5

12.0.5 (2015-01-03)

- Upgrade pip to 6.0.6
- Upgrade setuptools to 11.0

12.0.4 (2014-12-23)

- Revert the fix to `-p` on Debian based pythons as it was broken in other situations.
- Revert several sys.path changes new in 12.0 which were breaking virtualenv.

12.0.3 (2014-12-23)

- Fix an issue where Debian based Pythons would fail when using -p with the host Python.
- Upgrade pip to 6.0.3

12.0.2 (2014-12-23)

- Upgraded pip to 6.0.2

12.0.1 (2014-12-22)

- Upgraded pip to 6.0.1

12.0 (2014-12-22)

- **PROCESS** Version numbers are now simply `X.Y` where the leading `1` has been dropped.
- Split up documentation into structured pages
- Now using pytest framework
- Correct sys.path ordering for debian, issue #461
- Correctly throws error on older Pythons, issue #619
- Allow for empty $PATH, pull #601
- Don't set prompt if $env:VIRTUAL_ENV_DISABLE_PROMPT is set for Powershell
- Updated setuptools to 7.0

1.11.6 (2014-05-16)

- Updated setuptools to 3.6
- Updated pip to 1.5.6

1.11.5 (2014-05-03)

- Updated setuptools to 3.4.4
- Updated documentation to use https://virtualenv.pypa.io/
- Updated pip to 1.5.5

1.11.4 (2014-02-21)

- Updated pip to 1.5.4

1.11.3 (2014-02-20)

- Updated setuptools to 2.2
- Updated pip to 1.5.3

1.11.2 (2014-01-26)

- Fixed easy_install installed virtualenvs by updated pip to 1.5.2

1.11.1 (2014-01-20)

- Fixed an issue where pip and setuptools were not getting installed when using the `--system-site-packages` flag.
- Updated setuptools to fix an issue when installed with easy_install
- Fixed an issue with Python 3.4 and sys.stdout encoding being set to ascii
- Upgraded pip to v1.5.1
- Upgraded setuptools to v2.1

1.11 (2014-01-02)

- **BACKWARDS INCOMPATIBLE** Switched to using wheels for the bundled copies of setuptools and pip. Using sdists is no longer supported - users supplying their own versions of pip/setuptools will need to provide wheels.
- **BACKWARDS INCOMPATIBLE** Modified the handling of `--extra-search-dirs`. This option now works like pip's `--find-links` option, in that it adds extra directories to search for compatible wheels for pip and setuptools. The actual wheel selected is chosen based on version and compatibility, using the same algorithm as `pip install setuptools`.
- Fixed #495, –always-copy was failing (#PR 511)
- Upgraded pip to v1.5
- Upgraded setuptools to v1.4

1.10.1 (2013-08-07)

- **New Signing Key** Release 1.10.1 is using a different key than normal with fingerprint: 7C6B 7C5D 5E2B 6356 A926 F04F 6E3C BCE9 3372 DCFA
- Upgraded pip to v1.4.1
- Upgraded setuptools to v0.9.8

1.10 (2013-07-23)

- **BACKWARDS INCOMPATIBLE** Dropped support for Python 2.5. The minimum supported Python version is now Python 2.6.
- **BACKWARDS INCOMPATIBLE** Using `virtualenv.py` as an isolated script (i.e. without an associated `virtualenv_support` directory) is no longer supported for security reasons and will fail with an error.

  Along with this, `--never-download` is now always pinned to `True`, and is only being maintained in the short term for backward compatibility (Pull #412).
- **IMPORTANT** Switched to the new setuptools (v0.9.7) which has been merged with Distribute again and works for Python 2 and 3 with one codebase. The `--distribute` and `--setuptools` options are now no-op.
- Updated to pip 1.4.
- Added support for PyPy3k
- Added the option to use a version number with the `-p` option to get the system copy of that Python version (Windows only)
- Removed embedded `ez_setup.py`, `distribute_setup.py` and `distribute_from_egg.py` files as part of switching to merged setuptools.
- Fixed `--relocatable` to work better on Windows.
- Fixed issue with readline on Windows.

1.9.1 (2013-03-08)

- Updated to pip 1.3.1 that fixed a major backward incompatible change of parsing URLs to externally hosted packages that got accidentily included in pip 1.3.

1.9 (2013-03-07)

- Unset VIRTUAL_ENV environment variable in deactivate.bat (Pull #364)
- Upgraded distribute to 0.6.34.
- Added `--no-setuptools` and `--no-pip` options (Pull #336).
- Fixed Issue #373. virtualenv-1.8.4 was failing in cygwin (Pull #382).
- Fixed Issue #378. virtualenv is now "multiarch" aware on debian/ubuntu (Pull #379).
- Fixed issue with readline module path on pypy and OSX (Pull #374).
- Made 64bit detection compatible with Python 2.5 (Pull #393).

1.8.4 (2012-11-25)

- Updated distribute to 0.6.31. This fixes #359 (numpy install regression) on UTF-8 platforms, and provides a workaround on other platforms: `PYTHONIOENCODING=utf8 pip install numpy`.
- When installing virtualenv via curl, don't forget to filter out arguments the distribute setup script won't understand. Fixes #358.
- Added some more integration tests.
- Removed the unsupported embedded setuptools egg for Python 2.4 to reduce file size.

1.8.3 (2012-11-21)

- Fixed readline on OS X. Thanks minrk
- Updated distribute to 0.6.30 (improves our error reporting, plus new distribute features and fixes). Thanks Gabriel (g2p)
- Added compatibility with multiarch Python (Python 3.3 for example). Added an integration test. Thanks Gabriel (g2p)
- Added ability to install distribute from a user-provided egg, rather than the bundled sdist, for better speed. Thanks Paul Moore.
- Make the creation of lib64 symlink smarter about already-existing symlink, and more explicit about full paths. Fixes #334 and #330. Thanks Jeremy Orem.
- Give lib64 site-dir preference over lib on 64-bit systems, to avoid wrong 32-bit compiles in the venv. Fixes #328. Thanks Damien Nozay.
- Fix a bug with prompt-handling in `activate.csh` in non-interactive csh shells. Fixes #332. Thanks Benjamin Root for report and patch.
- Make it possible to create a virtualenv from within a Python 3.3. pyvenv. Thanks Chris McDonough for the report.
- Add optional –setuptools option to be able to switch to it in case distribute is the default (like in Debian).

1.8.2 (2012-09-06)

- Updated the included pip version to 1.2.1 to fix regressions introduced there in 1.2.

1.8.1 (2012-09-03)

- Fixed distribute version used with *–never-download*. Thanks michr for report and patch.
- Fix creating Python 3.3 based virtualenvs by unsetting the `__PYVENV_LAUNCHER__` environment variable in subprocesses.

1.8 (2012-09-01)

- **Dropped support for Python 2.4** The minimum supported Python version is now Python 2.5.
- Fix *–relocatable* on systems that use lib64. Fixes #78. Thanks Branden Rolston.
- Symlink some additional modules under Python 3. Fixes #194. Thanks Vinay Sajip, Ian Clelland, and Stefan Holek for the report.
- Fix `--relocatable` when a script uses `__future__` imports. Thanks Branden Rolston.
- Fix a bug in the config option parser that prevented setting negative options with environment variables. Thanks Ralf Schmitt.
- Allow setting `--no-site-packages` from the config file.
- Use `/usr/bin/multiarch-platform` if available to figure out the include directory. Thanks for the patch, Mika Laitio.
- Fix `install_name_tool` replacement to work on Python 3.X.
- Handle paths of users' site-packages on Mac OS X correctly when changing the prefix.

- Updated the embedded version of distribute to 0.6.28 and pip to 1.2.

1.7.2 (2012-06-22)

- Updated to distribute 0.6.27.
- Fix activate.fish on OS X. Fixes #8. Thanks David Schoonover.
- Create a virtualenv-x.x script with the Python version when installing, so virtualenv for multiple Python versions can be installed to the same script location. Thanks Miki Tebeka.
- Restored ability to create a virtualenv with a path longer than 78 characters, without breaking creation of virtualenvs with non-ASCII paths. Thanks, Bradley Ayers.
- Added ability to create virtualenvs without having installed Apple's developers tools (using an own implementation of `install_name_tool`). Thanks Mike Hommey.
- Fixed PyPy and Jython support on Windows. Thanks Konstantin Zemlyak.
- Added pydoc script to ease use. Thanks Marc Abramowitz. Fixes #149.
- Fixed creating a bootstrap script on Python 3. Thanks Raul Leal. Fixes #280.
- Fixed inconsistency when having set the `PYTHONDONTWRITEBYTECODE` env var with the –distribute option or the `VIRTUALENV_USE_DISTRIBUTE` env var. `VIRTUALENV_USE_DISTRIBUTE` is now considered again as a legacy alias.

1.7.1.2 (2012-02-17)

- Fixed minor issue in *–relocatable*. Thanks, Cap Petschulat.

1.7.1.1 (2012-02-16)

- Bumped the version string in `virtualenv.py` up, too.
- Fixed rST rendering bug of long description.

1.7.1 (2012-02-16)

- Update embedded pip to version 1.1.
- Fix *–relocatable* under Python 3. Thanks Doug Hellmann.
- Added environ PATH modification to activate_this.py. Thanks Doug Napoleone. Fixes #14.
- Support creating virtualenvs directly from a Python build directory on Windows. Thanks CBWhiz. Fixes #139.
- Use non-recursive symlinks to fix things up for posix_local install scheme. Thanks michr.
- Made activate script available for use with msys and cygwin on Windows. Thanks Greg Haskins, Cliff Xuan, Jonathan Griffin and Doug Napoleone. Fixes #176.
- Fixed creation of virtualenvs on Windows when Python is not installed for all users. Thanks Anatoly Techtonik for report and patch and Doug Napoleone for testing and confirmation. Fixes #87.
- Fixed creation of virtualenvs using -p in installs where some modules that ought to be in the standard library (e.g. *readline*) are actually installed in *site-packages* next to *virtualenv.py*. Thanks Greg Haskins for report and fix. Fixes #167.

- Added activation script for Powershell (signed by Jannis Leidel). Many thanks to Jason R. Coombs.

1.7 (2011-11-30)

- Gave user-provided `--extra-search-dir` priority over default dirs for finding setuptools/distribute (it already had priority for finding pip). Thanks Ethan Jucovy.
- Updated embedded Distribute release to 0.6.24. Thanks Alex Gronholm.
- Made `--no-site-packages` behavior the default behavior. The `--no-site-packages` flag is still permitted, but displays a warning when used. Thanks Chris McDonough.
- New flag: `--system-site-packages`; this flag should be passed to get the previous default global-site-package-including behavior back.
- Added ability to set command options as environment variables and options in a `virtualenv.ini` file.
- Fixed various encoding related issues with paths. Thanks Gunnlaugur Thor Briem.
- Made `virtualenv.py` script executable.

1.6.4 (2011-07-21)

- Restored ability to run on Python 2.4, too.

1.6.3 (2011-07-16)

- Restored ability to run on Python < 2.7.

1.6.2 (2011-07-16)

- Updated embedded distribute release to 0.6.19.
- Updated embedded pip release to 1.0.2.
- Fixed #141 - Be smarter about finding pkg_resources when using the non-default Python interpreter (by using the `-p` option).
- Fixed #112 - Fixed path in docs.
- Fixed #109 - Corrected doctests of a Logger method.
- Fixed #118 - Fixed creating virtualenvs on platforms that use the "posix_local" install scheme, such as Ubuntu with Python 2.7.
- Add missing library to Python 3 virtualenvs (`_dummy_thread`).

1.6.1 (2011-04-30)

- Start to use git-flow.
- Added support for PyPy 1.5
- Fixed #121 – added sanity-checking of the -p argument. Thanks Paul Nasrat.
- Added progress meter for pip installation as well as setuptools. Thanks Ethan Jucovy.
- Added –never-download and –search-dir options. Thanks Ethan Jucovy.

1.6

- Added Python 3 support! Huge thanks to Vinay Sajip and Vitaly Babiy.
- Fixed creation of virtualenvs on Mac OS X when standard library modules (readline) are installed outside the standard library.
- Updated bundled pip to 1.0.

1.5.2

- Moved main repository to Github: https://github.com/pypa/virtualenv
- Transferred primary maintenance from Ian to Jannis Leidel, Carl Meyer and Brian Rosner
- Fixed a few more pypy related bugs.
- Updated bundled pip to 0.8.2.
- Handed project over to new team of maintainers.
- Moved virtualenv to Github at https://github.com/pypa/virtualenv

1.5.1

- Added `_weakrefset` requirement for Python 2.7.1.
- Fixed Windows regression in 1.5

1.5

- Include pip 0.8.1.
- Add support for PyPy.
- Uses a proper temporary dir when installing environment requirements.
- Add `--prompt` option to be able to override the default prompt prefix.
- Fix an issue with `--relocatable` on Windows.
- Fix issue with installing the wrong version of distribute.
- Add fish and csh activate scripts.

1.4.9

- Include pip 0.7.2

1.4.8

- Fix for Mac OS X Framework builds that use `--universal-archs=intel`
- Fix `activate_this.py` on Windows.
- Allow `$PYTHONHOME` to be set, so long as you use `source bin/activate` it will get unset; if you leave it set and do not activate the environment it will still break the environment.

- Include pip 0.7.1

1.4.7

- Include pip 0.7

1.4.6

- Allow `activate.sh` to skip updating the prompt (by setting `$VIRTUAL_ENV_DISABLE_PROMPT`).

1.4.5

- Include pip 0.6.3
- Fix `activate.bat` and `deactivate.bat` under Windows when `PATH` contained a parenthesis

1.4.4

- Include pip 0.6.2 and Distribute 0.6.10
- Create the `virtualenv` script even when Setuptools isn't installed
- Fix problem with `virtualenv --relocate` when `bin/` has subdirectories (e.g., `bin/.svn/`); from Alan Franzoni.
- If you set `$VIRTUALENV_DISTRIBUTE` then virtualenv will use Distribute by default (so you don't have to remember to use `--distribute`).

1.4.3

- Include pip 0.6.1

1.4.2

- Fix pip installation on Windows
- Fix use of stand-alone `virtualenv.py` (and boot scripts)
- Exclude ~/.local (user site-packages) from environments when using `--no-site-packages`

1.4.1

- Include pip 0.6

1.4

- Updated setuptools to 0.6c11
- Added the –distribute option
- Fixed packaging problem of support-files

1.3.4

- Virtualenv now copies the actual embedded Python binary on Mac OS X to fix a hang on Snow Leopard (10.6).
- Fail more gracefully on Windows when `win32api` is not installed.
- Fix site-packages taking precedent over Jython's `__classpath__` and also specially handle the new `__pyclasspath__` entry in `sys.path`.
- Now copies Jython's `registry` file to the virtualenv if it exists.
- Better find libraries when compiling extensions on Windows.
- Create `Scripts\pythonw.exe` on Windows.
- Added support for the Debian/Ubuntu `/usr/lib/pythonX.Y/dist-packages` directory.
- Set `distutils.sysconfig.get_config_vars()['LIBDIR']` (based on `sys.real_prefix`) which is reported to help building on Windows.
- Make `deactivate` work on ksh
- Fixes for `--python`: make it work with `--relocatable` and the symlink created to the exact Python version.

1.3.3

- Use Windows newlines in `activate.bat`, which has been reported to help when using non-ASCII directory names.
- Fixed compatibility with Jython 2.5b1.
- Added a function `virtualenv.install_python` for more fine-grained access to what `virtualenv.create_environment` does.
- Fix a problem with Windows and paths that contain spaces.
- If `/path/to/env/.pydistutils.cfg` exists (or `/path/to/env/pydistutils.cfg` on Windows systems) then ignore `~/.pydistutils.cfg` and use that other file instead.
- Fix ‘ a problem <https://bugs.launchpad.net/virtualenv/+bug/340050>‘_ picking up some `.so` libraries in `/usr/local`.

1.3.2

- Remove the `[install] prefix = ...` setting from the virtualenv `distutils.cfg` – this has been causing problems for a lot of people, in rather obscure ways.
- If you use a boot script it will attempt to import `virtualenv` and find a pre-downloaded Setuptools egg using that.
- Added platform-specific paths, like `/usr/lib/pythonX.Y/plat-linux2`

1.3.1

- Real Python 2.6 compatibility. Backported the Python 2.6 updates to `site.py`, including user directories (this means older versions of Python will support user directories, whether intended or not).

- Always set `[install] prefix` in `distutils.cfg` – previously on some platforms where a system-wide `distutils.cfg` was present with a `prefix` setting, packages would be installed globally (usually in `/usr/local/lib/pythonX.Y/site-packages`).
- Sometimes Cygwin seems to leave `.exe` off `sys.executable`; a workaround is added.
- Fix `--python` option.
- Fixed handling of Jython environments that use a jython-complete.jar.

1.3

- Update to Setuptools 0.6c9
- Added an option `virtualenv --relocatable EXISTING_ENV`, which will make an existing environment "relocatable" – the paths will not be absolute in scripts, `.egg-info` and `.pth` files. This may assist in building environments that can be moved and copied. You have to run this *after* any new packages installed.
- Added `bin/activate_this.py`, a file you can use like `execfile("path_to/activate_this.py", dict(__file__="path_to/activate_this.py"))` – this will activate the environment in place, similar to what the mod_wsgi example does.
- For Mac framework builds of Python, the site-packages directory `/Library/Python/X.Y/site-packages` is added to `sys.path`, from Andrea Rech.
- Some platform-specific modules in Macs are added to the path now (`plat-darwin/`, `plat-mac/`, `plat-mac/lib-scriptpackages`), from Andrea Rech.
- Fixed a small Bashism in the `bin/activate` shell script.
- Added `__future__` to the list of required modules, for Python 2.3. You'll still need to backport your own `subprocess` module.
- Fixed the `__classpath__` entry in Jython's `sys.path` taking precedent over virtualenv's libs.

1.2

- Added a `--python` option to select the Python interpreter.
- Add `warnings` to the modules copied over, for Python 2.6 support.
- Add `sets` to the module copied over for Python 2.3 (though Python 2.3 still probably doesn't work).

1.1.1

- Added support for Jython 2.5.

1.1

- Added support for Python 2.6.
- Fix a problem with missing `DLLs/zlib.pyd` on Windows. Create
- `bin/python` (or `bin/python.exe`) even when you run virtualenv with an interpreter named, e.g., `python2.4`
- Fix MacPorts Python

- Added –unzip-setuptools option
- Update to Setuptools 0.6c8
- If the current directory is not writable, run ez_setup.py in `/tmp`
- Copy or symlink over the `include` directory so that packages will more consistently compile.

1.0

- Fix build on systems that use `/usr/lib64`, distinct from `/usr/lib` (specifically CentOS x64).
- Fixed bug in `--clear`.
- Fixed typos in `deactivate.bat`.
- Preserve `$PYTHONPATH` when calling subprocesses.

0.9.2

- Fix include dir copying on Windows (makes compiling possible).
- Include the main `lib-tk` in the path.
- Patch `distutils.sysconfig`: `get_python_inc` and `get_python_lib` to point to the global locations.
- Install `distutils.cfg` before Setuptools, so that system customizations of `distutils.cfg` won't effect the installation.
- Add `bin/pythonX.Y` to the virtualenv (in addition to `bin/python`).
- Fixed an issue with Mac Framework Python builds, and absolute paths (from Ronald Oussoren).

0.9.1

- Improve ability to create a virtualenv from inside a virtualenv.
- Fix a little bug in `bin/activate`.
- Actually get `distutils.cfg` to work reliably.

0.9

- Added `lib-dynload` and `config` to things that need to be copied over in an environment.
- Copy over or symlink the `include` directory, so that you can build packages that need the C headers.
- Include a `distutils` package, so you can locally update `distutils.cfg` (in `lib/pythonX.Y/distutils/distutils.cfg`).
- Better avoid downloading Setuptools, and hitting PyPI on environment creation.
- Fix a problem creating a `lib64/` directory.
- Should work on MacOSX Framework builds (the default Python installations on Mac). Thanks to Ronald Oussoren.

0.8.4

- Windows installs would sometimes give errors about `sys.prefix` that were inaccurate.
- Slightly prettier output.

0.8.3

- Added support for Windows.

0.8.2

- Give a better warning if you are on an unsupported platform (Mac Framework Pythons, and Windows).
- Give error about running while inside a workingenv.
- Give better error message about Python 2.3.

0.8.1

Fixed packaging of the library.

0.8

Initial release. Everything is changed and new!

Warning: Python bugfix releases 2.6.8, 2.7.3, 3.1.5 and 3.2.3 include a change that will cause "import random" to fail with "cannot import name urandom" on any virtualenv created on a Unix host with an earlier release of Python 2.6/2.7/3.1/3.2, if the underlying system Python is upgraded. This is due to the fact that a virtualenv uses the system Python's standard library but contains its own copy of the Python interpreter, so an upgrade to the system Python results in a mismatch between the version of the Python interpreter and the version of the standard library. It can be fixed by removing `$ENV/bin/python` and re-running virtualenv on the same target directory with the upgraded Python.

CHAPTER 2

Other Documentation and Links

- Blog announcement of virtualenv.
- James Gardner has written a tutorial on using virtualenv with Pylons.
- Chris Perkins created a showmedo video including virtualenv.
- Doug Hellmann's virtualenvwrapper is a useful set of scripts to make your workflow with many virtualenvs even easier. His initial blog post on it. He also wrote an example of using virtualenv to try IPython.
- Pew is another wrapper for virtualenv that makes use of a different activation technique.
- Using virtualenv with mod_wsgi.
- virtualenv commands for some more workflow-related tools around virtualenv.
- PyCon US 2011 talk: Reverse-engineering Ian Bicking's brain: inside pip and virtualenv. By the end of the talk, you'll have a good idea exactly how pip and virtualenv do their magic, and where to go looking in the source for particular behaviors or bug fixes.

CHAPTER 3

Compare & Contrast with Alternatives

There are several alternatives that create isolated environments:

- `workingenv` (which I do not suggest you use anymore) is the predecessor to this library. It used the main Python interpreter, but relied on setting `$PYTHONPATH` to activate the environment. This causes problems when running Python scripts that aren't part of the environment (e.g., a globally installed `hg` or `bzr`). It also conflicted a lot with Setuptools.
- virtual-python is also a predecessor to this library. It uses only symlinks, so it couldn't work on Windows. It also symlinks over the *entire* standard library and global `site-packages`. As a result, it won't see new additions to the global `site-packages`.

  This script only symlinks a small portion of the standard library into the environment, and so on Windows it is feasible to simply copy these files over. Also, it creates a new/empty `site-packages` and also adds the global `site-packages` to the path, so updates are tracked separately. This script also installs Setuptools automatically, saving a step and avoiding the need for network access.
- zc.buildout doesn't create an isolated Python environment in the same style, but achieves similar results through a declarative config file that sets up scripts with very particular packages. As a declarative system, it is somewhat easier to repeat and manage, but more difficult to experiment with. `zc.buildout` includes the ability to setup non-Python systems (e.g., a database server or an Apache instance).

I *strongly* recommend anyone doing application development or deployment use one of these tools.

Index

Symbols

A

C

E

V