Virtualenv

Virtualenv is a tool that lets you create an isolated Python environment for your project. It creates an environment that has its own installation directories, that doesn't share dependencies with other `virtualenv` environments (and optionally doesn't access the globally installed dependencies either). You can even configure what version of Python you want to use for each individual environment. It's very much recommended to use `virtualenv` when dealing with Python applications.

Installation

To install `virtualenv` run:

```
$ pip install virtualenv
```

Usage

If you have a project in a directory called `my-project` you can set up `virtualenv` for that project by running:

```
$ cd my-project/
$ virtualenv venv
```

If you want your `virtualenv` to also inherit globally installed packages run:

```
$ virtualenv venv --system-site-packages
```

These commands create a `venv/` directory in your project where all dependencies are installed. You need to **activate** it first though (in every terminal instance where you are working on your project):

```
$ source venv/bin/activate
```

You should see a `(venv)` appear at the beginning of your terminal prompt indicating that you are working inside the `virtualenv`. Now when you install something like this:

```
$ pip install <package>
```

It will get installed in the `venv/` folder, and not conflict with other projects.

To leave the virtual environment run:

```
$ deactivate
```

Important: Remember to add `venv` to your project's `.gitignore` file so you don't include all of that in your source code.

It is preferable to install big packages (like Numpy), or packages you always use (like IPython) globally. All the rest can be installed in a `virtualenv`.

Virtualenvwrapper

To make it easier to work on multiple projects that has separate environments you can install `virtualenvwrapper`. It's an extension to `virtualenv` and makes it easier to create and delete virtual environments without creating dependency conflicts.

To install `virtualenvwrapper` run:

```
$ pip install virtualenvwrapper
```

Depending on your setup you might need to install it using `sudo`. Read the installation documentation for more information.

Note: `virtualenvwrapper` keeps all the virtual environments in `~/.virtualenv` while `virtualenv` keeps them in the project directory.