How To Set Up a Virtual Python Environment (Linux)

`virtualenv` is a tool to create isolated Python environments. You can read more about it in the Virtualenv documentation. This article provides a quick summary to help you set up and use a virtual environment.

A Note About Python 3.6 and Ubuntu 16.04 LTS

If you're running Ubuntu 16.04 LTS (or and earlier version), Python 3.5 is likely installed by default. *Don't remove it!* To get Python 3.6, follow the instructions in this section.

Add the PPA

Run the following command to add the Python 3.6 PPA.

```
sudo add-apt-repository ppa:jonathonf/python-3.6
```

Check for Updates and Install

Check for updates and install Python 3.6 via the following commands.

```
sudo apt-get update
sudo apt-get install python3.6
```

Now you have three Python version, use `python` to run version 2.7, `python3` for version 3.5, and `python3.6` for version 3.6.

For more information on this subject, check out Ji m's article How to Install Python 3.6.1 in Ubuntu 16.04 LTS.

Create a Virtual Python Environment

`cd` to your project directory and run `virtualenv` to create the new virtual environment.

The following commands will create a new virtual environment under my-project/my-venv.

```
cd my-project
virtualenv --python python3.6 venv
```

Activate the Environment

Now that we have a virtual environment, we need to activate it.

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

After you activate the environment, your command prompt will be modified to reflect the change.

Add Libraries and Create a *requirements.txt* File

After you activate the virtual environment, you can add packages to it using `pip`. You can also create a description of your dependencies using `pip`.

The following command creates a file called `requirements.txt` that enumerates the installed packages.

```
pip freeze > requirements.txt
```

This file can then be used by collaborators to update virtual environments using the following command.

```
pip install -r requirements.txt
```

Deactivate the Environment

To return to normal system settings, use the `deactivate` command.

```
deactivate
```

After you issue this command, you'll notice that the command prompt returns to normal.

Acknowledgments

Much of this article is taken from The Hitchhiker's Guide to Python. Go buy a copy right now.

Previous Next