A CLOUD GURU

Lesson Description - Virtualenv

We can only have one version of a package installed at a given time, and this can sometimes be a headache if we have multiple projects that require different versions of the same dependency. This is where `virtualenv` comes into play and allows us to create sandboxed Python environments.

Python Documentation for This Video

- `venv`

Virtualenv or Venv

Virtualenvs allow you to create sandboxed Python environments. In Python 2, you need to install the `virtualenv` package to do this, but with Python 3 it's been worked in under the module name of `venv`.

To create a virtualenv, we'll use the following command:

```
$ python3.6 -m venv [PATH FOR VIRTUALENV]
```

The `-m` flag loads a module as a script, so it looks a little weird, but "python3.6 -m venv" is a stand-alone tool. This tool can even handle its own flags.

Let's create a directory to store our virtualenvs called `venvs`. From here we create an `experiment` virtualenv to see how they work.

```
$ mkdir venvs
$ python3.6 -m venv venvs/experiment
```

Virtualenvs are local Python installations with their own site-packages, and they do absolutely nothing for us by default. To use a virtualenv, we need to `activate` it. We do this by sourcing an `activate` file in the virtualenv's `bin` directory:

```
$ source venvs/experiment/bin/activate
(experiment) ~ $
```

Notice that our prompt changed to indicate to us what virtualenv is active. This is part of what the `activate` script does. It also changes our `$PATH`:

```
(experiment) ~ $ echo $PATH
/home/user/venvs/experiment/bin:/home/user/bin:/usr/local/bin:/usr/
bin:/usr/local/sbin:/usr/sbin:/home/user/.local/bin:/home/user/bin
(experiment) ~ $ which python
~/venvs/experiment/bin/python
(experiment) ~ $ python --version
Python 3.6.4
(experiment) ~ $ pip list
Package    Version
---------- -------
pip        9.0.1
setuptools 28.8.0
(experiment) ~ $ deactivate
$ which python
/usr/bin/python
```

With the virtualenv activated, the `python` and `pip` binaries point to the local Python 3 variations, so we don't need to append the `3.6` to all of our commands. To remove the virtualenv's contents from our `$PATH`, we will utilize the `deactivate` script that the virtualenv provided.