Installing multiple versions of Python on Ubuntu 18.04 and running them within virtual environments

Jae-Yun Jun*

1 Installing multiple versions of Python on Ubuntu 18.04

Sources: https://hackersandslackers.com/multiple-versions-python-ubuntu/

Update Ubuntu's mirrors and packages to make sure we pull the latest packages when we install anything:

```
sudo apt update && apt upgrade -y
```

Install Python dependencies

```
sudo apt-get install build-essential checkinstall

sudo apt-get install libreadline-gplv2-dev libncursesw5-dev libssl-dev \
    libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev libffi-dev zlib1g-dev
```

Download the desired version of Python (say 3.7.13) from Python's official website (www.python.org). Then uncompress it from the directory `/opt`:

```
sudo cp ./Python-3.7.13.tgz /opt

sudo tar xzf Python-3.7.13.tgz
```

Now, install Python from Source in the alternative mode

```
cd Python-3.7.13

sudo ./configure --enable-optimizations

sudo make altinstall
```

Suppose that by default we have Python 3.6, and, therefore, we now have two Python versions (i.e., 3.6 and 3.7.13). We can manage alternative Python installations as follows:

*ECE Paris, 10 Rue Sextius Michel 75015 Paris, France; jaeyunjk@gmail.com

```
sudo update-alternatives --install /usr/bin/python python /usr/bin/python3.6 1
sudo update-alternatives --install /usr/bin/python python /opt/Python-3.7.13/python 2
```

We can list all the alternative installations of Python:

```
update-alternatives --list python
```

We can swap between versions of Python as follows:

```
update-alternatives --config python
```

Enter the number that corresponds to the alternative version that you want.

2 Code Python using virtual environments

Before you create a virtual environment for Python, make sure that you have chosen the right Python version

```
python --version
```

If the Python version is not the right one, then choose the right one with the following command

```
update-alternatives --config python
```

and enter the number that corresponds to the alternative version that you want. Once you have the right Python version, then create the virtual environment:

```
python -m venv my_venv
```

where `my_venv` is the name of the desired environment (which you can change at your will). Close the terminal and open it again. Then, you should see that the virtual environment will be activated. Or you can simply type the following command in the terminal:

```
source ~/code/my_venv/bin/activate
```

Now, let us see all the Python packages that are installed under the current virtual environment:

```
pip list
```

Now, whenever you install Python packages, these packages will be placed under the folder corresponding to the virtual environment in question. When we remove the virtual-environment folder, all the install Python packages will also be removed.
Now, launch Visual Studio from the terminal on which you have activated the Python virtual environment:

```
code .
```

Now, you can run any code from Visual Studio Code within the virtual environment that you just created.

3 Running a Python script from Visual Studio editor

If you notice that your virtual environment is not taken into account, then, from `Command Palette (Ctrl+Shift+P)`, type the command `Python:Select interpreter`, and then choose the right one among the suggested ones (e.g., Python 3.7.13 ('aieconomist_python3.7.13':venv)). Then, you should see that, when you run your code, it runs within the desired virtual environment.