HOW TO INSTALL A VIRTUAL ENVIRONMENT

MARC WILLIAMSON

1. What is a Virtual Environment?

A virtual environment simulates having administrator priviledges on your machine. In general, we will conduct research on the NYU machines, either through ssh or on the computers in the physics buildings. These computers have access to the NYU supercomputers, which most of us will have to use. If you've ever tried to install a new python module, like scikit-learn or scipy, you will notice that you are not allowed to install new packages because you don't have admin priviledges. This can be extremely restrictive. There is also a DANGER that if you write code that works on the NYU machines, that their software will be updated, and your code will no longer work. For these reasons, a virtual environment is the best, safest, most robust method for writing code.

2. Install Instructions

There are two main steps: the virtual environment that we install comes with Python 2.6, which is too old. The first step is to install an updated version of python. We will do this manually in a directory that we have full write access to. Only then will we install the virtual environment.

(1) Create a /src directory that you can install python into. Go to your home directory (cd ~), and run...

```
bash-4.2$ mkdir src
```

(2) cd into your /src directory, and download the version of Python that you want to install using wget (Replace the X.X.X with the version you want.)

```
bash-4.2$ wget http://www.python.org/ftp/python/X.X.X/Python-X.X.X.tgz
```

(3) Unpack the .tgz file that you just downloaded with the tar command...

```
bash-4.2$ tar -zxvf Python-X.X.X.tgz
```

(4) cd into the /src/Python-X.X.X directory.

(5) Create a directory to store install files for Python.

```
bash-4.2$ mkdir ~/.localpython
```

(6) Configure your Python download for installation.

```
bash-4.2$ ./configure --prefix=$HOME/.localpython
```

The $HOME path should be the absolute path to your home directory by default.

(7) Now we can install Python. First type `make`, then type `make install`.

Congratulations! You have now installed your own version of python. When you're not using your virtual environment, you will still be using the system's default Python. If you always want to use the Python version that you just installed, you will need to change your $PYTHONPATH. Now we can install the virtual environment.

(1) cd into your ~/src directory.

(2) Download the virtual environment version that you want. I recommend version 15.0.0.

```
bash-4.2$ curl -O https://pypi.python.org/packages/source/v/virtualenv/virtualenv-15.0.0.tar.gz
```

(3) Unpack the virtual environment.

```
bash-4.2$ tar -zxvf virtualenv-15.0.0.tar.gz
```

(4) cd into the virtualenv directory.
(5) Run the setup for installing the virtualenv with your version of Python.

```
bash-4.2$ ~/.localpython/bin/python setup.py install
```

(6) Create a directory in your home to store different versions of your virtual environments. cd to your home, `bash$ cd $HOME`, and then make a new directory, `bash$ mkdir virtualenvs`
(7) cd into your ~/virtualenvs directory and install your virtual environment with the version of Python that you wan to use...

```
bash-4.2$ python $HOME/virtualenv-15.0.0/virtualenv.py -p $HOME/.localpython/bin/pythonX.X <name of your virtual env>
```

(8) ls in your ~/virtualenvs directory, and you should now see a new directory with the name of your virtual environment! All that is left is to learn how to activate and deactivate your virtual environment. I call my virtual environments mypyX.X, where X.X is the version of Python that I have installed. Inside your ~/virtualenvs/mypyX.X/bin/ directory, there is an executable called activate. You can turn on your virtual environment by running `bash$ source /PATH TO mypyX.X/bin/activate`. Your terminal should now look like this...

```
(mypy3.5) bash-4.2$
```

You can run python now, and check to see that it is indeed the version X.X that you installed!
(9) When you are finished using your virtual environment, you can simply type `bash$ deactivate` and you will return to the regular system on the machine you are using.

3. Using your Virtual Environment

You don't want to have type so much just to turn on your virtual environment, do you? No. So let's create our own bash command to do it!

(1) cd to your home directory, cd ~
(2) create, or open, the .bashrc file using your editor of choice.
(3) Add the following line:

```
alias mypyX.X="source $HOME/virtualenvs/mypyX.X/bin/activate"
```

Now when you want to turn on your virtual environment, all you have to type is `bash$ mypyX.X`, and it will turn on!

The .bashrc file is run everytime a new terminal window is opened. There is a separate file for when you open a new terminal through logging on via ssh. This file should be called .bash_profile

4. Installing Things!

Now that you have a virtual environment, life is easy! If you want to install a python module, you can just use pip to install it! Our virtual environment automatically has pip installed in it, so getting new packages and upgrading existing packages is extremely easy.