

# Introduction To virtual-Environment (Python setup in Linux)

Python, Python3..... Pip, Pip3..... It seems like we have faced a great disaster in Linux OS just because of having pre-installed versions of them. For so, I want to introduce you with a new technique. Using this I can ensure you there will be no effect on system (pre-installed modules). In this way you will create a virtual environment (virtualenv) of a version of python and introduce your project that this python is its default python. For so, project will never interfere with the system. But of course, you have to install the necessary libraries for projects in another directories, as like the libraries are installed using virtualenv, can only be accessed by the projects which holds the local position of it (means the virtualenv directory and the project directory is same). Let's stop discussion and dig into for practical experience. We are going to use Python3 as an example.

Steps:-

1. **Install desire python version** (not necessary if you already have that as pre-installed). For us, it's Python3.6.7 (pre-installed). If you want to check about yours then type "python3" in terminal. It will show your python-version if there is any. If not then install your desire.
2. **Install virtualenv using pip** (any version of pip) by the command : **sudo pip install virtualenv**
3. Open a directory where you want to create the virtual environment for local python. Note that the directory must be in the same directory where Linux is installed and which must be freed from root monitoring (Doesn't require root access to enter). I suggest you to create a folder in Desktop. It's the best way.

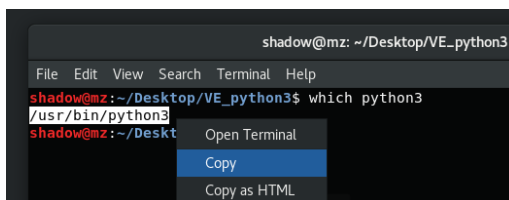
4. Open terminal in the directory as like :

5. Type the command in terminal as like:

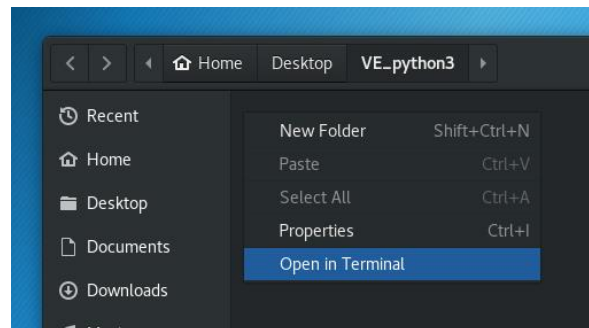
For Python v3: **Which python3**

For Python v2: **Which python**

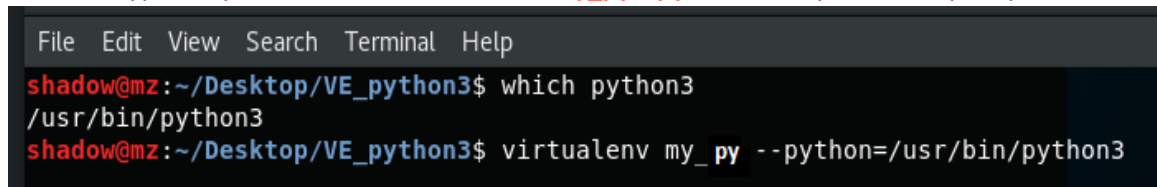
It will give you a path. Copy that as like:



```
shadow@mz: ~/Desktop/VE_python3
File Edit View Search Terminal Help
shadow@mz:~/Desktop/VE_python3$ which python3
/usr/bin/python3
shadow@mz:~/Desktop/
```



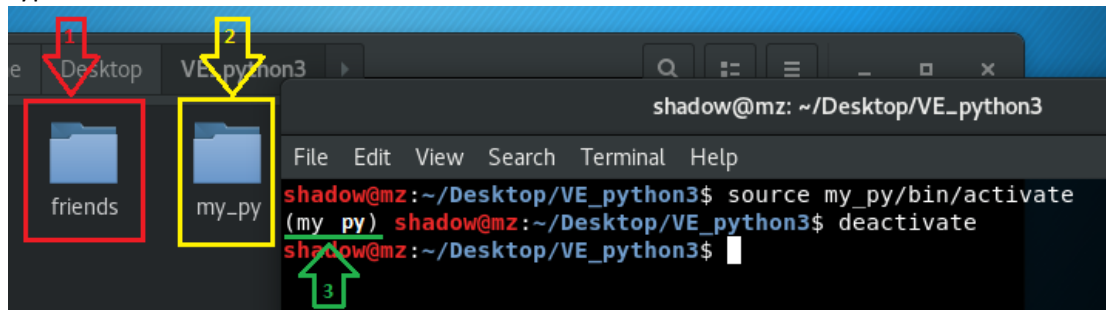
6. Now type in opened terminal: **virtualenv my\_py --python=** and pest the copied path as like:



```
File Edit View Search Terminal Help
shadow@mz:~/Desktop/VE_python3$ which python3
/usr/bin/python3
shadow@mz:~/Desktop/VE_python3$ virtualenv my_py --python=/usr/bin/python3
```

Here, my\_py is the environment name. You can change the name if you wish. But then the activation code of it also be different. Here "**source my\_py/bin/active**" my\_py will be the name what you give it.

- Done. This directory becomes the virtual environment for python3 (for my case). Now copy any project directory in the location and if you run that project from any ide (like: vs-code) then this virtual environment will be active by default.
- If you want to activate this from the terminal then type **source my\_py/bin/active** & to disable type **deactivate**. Note: when the environment is activate then its name also be shown as:



Here,

- My project directory (a git repository too)
- My virtualenv directory ( Local Python)
- When I have activated my local python then its name has been appeared to indicate that the local is activated. After activate the local the terminal acts like the OS has only one python at the local directory location. So there is no pip/pip3 or python/python3. Just use pip & python to control the local.
- Note that: The local is activated for only those terminals, which have called it. Must do any command after active it which you want to work on the local. Like if you want to install libraries which are necessary for our project in the local, then do as:
  - Active the local virtual environment.
  - pip install numpy**
  - pip install opencv-python**
  - pip install pandas**
  - pip install sklearn**
  - pip install keras**
  - pip install tensorflow**

## Conclusion:

### Why this?..... Advantage

- It can secure the system from being interrupted.
- When a project uses many libraries which are only for the project and there is no need of them after the project finished then those can be a burden for OS to carry the unnecessary items. Using it if anything is installed then those are staying in the local directory (**my-py** for us). If we want to unload the packages then just simply can delete the local directory (**my-py**).
- While activate it then the OS acts like it has only one python for the activated terminal. So there is no tangly of using pip, pip3, pip3.8 and so on. Simple like windows OS.

4. No interrupt with multi-versions of same library used for differ projects.
5. It can make project even more platform free by allowing a project to run in any OS which even never have the libraries without installing any. (Make project portable)

**Disadvantage:**

1. Library packages are need to be installed for every different virtualenv directory(means every **my\_py** directory of differ locations)

For any problem or confusion you are welcome to ask [mz.minhaz5683@gmail.com](mailto:mz.minhaz5683@gmail.com)