1. 创建虚拟环境(virtualenv 和virtualenvwrapper)

1.1，virtualenv的概述

virtualenv是用来创建Python的虚拟环境的库，虚拟环境能够独立于真实环境存在，并且可以同时有多个互相独立的Python虚拟环境，每个虚拟环境都可以营造一个干净的开发环境，对于项目的依赖、版本的控制有着非常重要的作用。

虚拟环境有什么意义?

比如：我们要同时开发多个应用程序，应用A需要Django1.11，而应用B需要Django1.8怎么办，这种情况下，每个应用可能需要各自拥有一套独立的Python运行环境,virtualenv就可以用来为每一个应用创建一套'隔离'的Python运行环境。

1.2，安装pip

【请使用普通用户】

a. 查看pip版本

查看pip版本: pip -V

查看pip3版本: pip3 -V

b. 安装pip(如果存在则不需要安装)

安装pip3: apt install python3-pip

安装pip2: apt install python-pip

c. 更新pip

更新pip (如果pip版本高于9.0则不需要更新):

更新pip3: pip3 install --upgrade pip

更新pip: pip install --upgrade pip

注意: 更新后如出现以下错误 (这是pip 10.0.0版本的BUG) :

```
Traceback (most recent call last):
File “/usr/bin/pip”, line 9, in
from pip import main
```

解决方法: 修改对应pip文件中的代码(pip和pip3类似)

例如更新pip时报错则需要修改 /usr/bin/pip 文件中的代码,

使用: sudo vim /usr/bin/pip 打开pip文件

将:

```
from pip import main
if __name__ == '__main__':
    sys.exit(main())
```

改成:

```
from pip import __main__
if __name__ == '__main__':
    sys.exit(__main__._main())
```

d. 让pip默认使用python3，执行命令:

```
sudo update-alternatives --install /usr/bin/python python /usr/bin/python3 150
```

e. pip命令

pip install xxx:安装xxx依赖包

pip list:查看所有依赖包

pip freeze:查看新安装的包

pip uninstall xxx : 卸载xxx包

1.3，virtualenv和virtualenvwrapper 的安装和使用

【请使用普通用户】

a. 安装虚拟环境

```
sudo apt update
sudo pip3 install virtualenv virtualenvwrapper
```

安装后如果不能使用虚拟环境命令，则需要配置环境变量

1，进入家目录：cd ~

2，使用vim打开.bashrc，定位到最后:shift+g，并添加以下2行代码(注意修改自己Ubuntu的用户名)

```
export WORKON_HOME=/home/自己Ubuntu的用户名/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

3，在家目录创建.virtualenvs目录：mkdir .virtualenvs

4，加载修改后的设置，使之生效：source .bashrc

b. 创建虚拟环境：

```
mkvirtualenv env
mkvirtualenv env2 -p /usr/bin/python3  (指定python路径)
```

c. 退出虚拟环境

```
deactivate
```

d. 进入虚拟环境：

```
workon 虚拟环境名称
```

f. 删除虚拟环境

```
rmvirtualenv env
```