Python3开发环境安装

```
一. 安装Python

1. 下载Python3源码包(此处我们下载Python3.6.1)
https://www.python.org/

2. 在Linux系统中安装必要的依赖包
   1)yum -y install gcc* openssl-devel  libtermcap-devel ncurses-devel libevent-devel readline-devel python-devel  zlib zlib-devel

   2)yum -y groupinstall  development tools

3. 开始编译安装Python3
[root@k8s-master1 Python-3.6.1]# ./configure --prefix=/usr/local/python36 && make && make install

4. 将python3加入到系统的PATH变量中, 即可通过python3命令进入交互式开发界面

   1)[root@k8s-master1 Python-3.6.1]# echo $PATH  ##查看操作系统PATH环境变量
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/usr/local/sersync:/root/bin

   2)[root@k8s-master1 Python-3.6.1]# ln -s /usr/local/python3/bin/python3   /usr/bin/python3

5. Python3安装完成
[root@k8s-master1 /]# python3
Python 3.6.1 (default, Jan 24 2019, 12:46:50)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>



二.Python包管理工具 setuptools 与 pip

在Python中安装第三方包时(如 paramiko flask), 需要使用easy_install命令, 这个命令是Setuptools包中的一个命令, pip也是一个包管理工具,它依赖于setuptools, 所以在安装pip之前需要安装Setuptools.

2.1 setuptools安装
    2.1.1 下载Setuptools包
          https://pypi.org/project/setuptools/40.6.3/
    2.1.2 安装setuptools包
          [root@k8s-master1 setuptools-40.6.3]# unzip setuptools-40.6.3.zip
          [root@k8s-master1 src]#  cd setuptools-40.6.3
          [root@k8s-master1 setuptools-40.6.3]# python setup.py  install

2.2 pip安装
    2.2.1 下载pip包
          https://pypi.org/project/pip/19.0.1/
    2.2.2 安装pip包
          [root@k8s-master1 src]# tar -zxvf pip-19.0.1.tar.gz
          [root@k8s-master1 src]#  cd pip-19.0.1
          [root@k8s-master1 pip-19.0.1]# python setup.py  install

2.3 安装第三方包如paramiko
[root@k8s-master1 pip-19.0.1]# pip install paramiko

2.3.1
安装指定版本的包
例如:
ip install paramiko==2.4.1(“==之后为指定安装的版本”)

2.3.2 从指定的pip源安装包
例如:
pip install jupyter -i http://pypi.douban.com/simple  --trusted-host pypi.douban.com

参数:
-i 指定pip源网站
--trusted-host 当在安装包时存在https错误时, 可以通过该参数信任https网站

2.3.3 查看当前python环境中已安装的包
[root@k8s-master1 pip-19.0.1]# pip freeze
ansible==2.6.2
apache-libcloud==2.0.0
Babel==0.9.6
backports.ssl-match-hostname==3.5.0.1
cffi==1.6.0
chardet==2.2.1
CherryPy==5.6.0


三. 沙盒工具virtualenv

通过virtualenv, 我们可以系统中的不同Python项目所应用的Python环境进行隔离, 这样就可以不同项目之间的安装包相互影响与干扰

3.1 安装virtualenv
3.1.1 pip安装
      [root@k8s-master1 /]# pip install virtualenv
3.1.2 源码安装
      [root@k8s-master1 src]# tar -zxvf virtualenv-16.2.0.tar.gz
      [root@k8s-master1 virtualenv-16.2.0]#  cd virtualenv-16.2.0
      [root@k8s-master1 virtualenv-16.2.0]# python setup.py  install

3.2 使用virtualenv创建沙盒环境
[root@k8s-node1 /]#  cd /app_shell/
[root@k8s-node1 app_shell]# virtualenv webproj  ##创建项目隔离沙盒环境webproj
New python executable in /app_shell/webproj/bin/python  #在创建沙盒环境的同时安装pip setuptools等包管理工具
Installing setuptools, pip, wheel...
done.

常用参数:
-p  指定沙盒中的python环境
[root@nfs /]#  virtualenv  -p /usr/local/python3/bin/python3  /app_shell/webproj


3.3 使用沙盒并查看Python环境
3.3.1 Python环境变量

1)未应用沙盒时 Python执行文件的路径为
[root@k8s-node1 app_shell]# which python
/usr/bin/python

2)应用沙盒
[root@k8s-node1 app_shell]# source  /app_shell/webproj/bin/activate
(webproj) [root@k8s-node1 app_shell]#   ##此时系统的Shell将会发生变化

3)继续再看Python变量
(webproj) [root@k8s-node1 app_shell]# which python
/app_shell/webproj/bin/python


3.4关闭沙盒
(webproj) [root@k8s-node1 app_shell]# deactivate
[root@k8s-node1 app_shell]#
```

四.交互式Python开发工具ipython安装

4.1 什么是ipython
它是一个交互式的Python shell环境，设计目的是在交互式计算和软件开发的过程中，大大的提高生产力． 工作模式是 编码-->执行，而不是传统的 编码-->编译-->执行． 在一定的情况下，ipython可以对我们的编码过程做出一些提示，提升编码效率．

4.2 ipython安装

4.2.1 pip方式安装 pip install ipython
(webproj) [root@k8s-node1 app_shell]# pip install ipython
如上面例子：我们安装的ipython是在 webproj沙盒中安装的，此时的ipython仅在该沙盒中可以使用，其他的环境中不能使用，这就是通过沙盒可以将项目进行隔离的目的．

4.2.2 源码方式安装：
方法如上，细节略...