python Linux 环境 （版本隔离工具）

首先新建用户，养成良好习惯useradd python

1、安装pyenv

GitHub官网：https://github.com/pyenv/pyenv-installer

pyenv installer

This tool installs pyenv and friends. It is inspired by rbenv-installer.

Prerequisites

In general, compiling your own Python interpreter requires the installation of the appropriate libraries and packages. The installation wiki provides a list of these for common operating systems.

Install:

```
$ curl https://pyenv.run | bash
```

`pyenv.run` redirects to the install script in this repository and the invocation above is equivalent to:

```
$ curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash
```

Restart your shell so the path changes take effect:

You can now begin using pyenv.

Update:

```
$ pyenv update
```

Uninstall: `pyenv` is installed within `$PYENV_ROOT` (default: `~/.pyenv`). To uninstall, just remove it:

```
$ rm -fr ~/.pyenv
```

and remove these three lines from `.bashrc`:

```
export PATH="$HOME/.pyenv/bin:$PATH"
eval "$(pyenv init -)"
eval "$(pyenv virtualenv-init -)"
```

If you need, export USE_GIT_URI to use git:// instead of https:// for git clone.

Travis itself uses pyenv and therefore PYENV_ROOT is set already. To make it work anyway the installation for pyenv-installer needs to look like this:

```
[...]
- unset PYENV_ROOT
- curl -L https://github.com/pyenv/pyenv-installer/raw/master/bin/pyenv-installer | bash
- export PATH="$HOME/.pyenv/bin:$PATH"
- pyenv install 3.5.2
```

The project on github contains a setup for vagrant to test the installer inside a vagrant managed virtual image.

If you don't know vagrant yet: just install the latest package, open a shell in this project directory and say

```
$ vagrant up
$ vagrant ssh
```

Now you are inside the vagrant container and your prompt should like something like `vagrant@vagrant-ubuntu-trusty-64:~$`

The project (this repository) is mapped into the vagrant image at /vagrant

```
$ cd /vagrant
$ python setup.py install
$ echo 'export PATH="$HOME/.pyenv/bin:$PATH"' >> ~/.bashrc
$ echo 'eval "$(pyenv init -)"' >> ~/.bashrc
$ echo 'eval "$(pyenv virtualenv-init -)"' >> ~/.bashrc
$ source ~/.bashrc
```

Pyenv should be installed and responding now.

20190111

- Remove experimental PyPi support and replace with a dummy package.

- Initial release on PyPi.

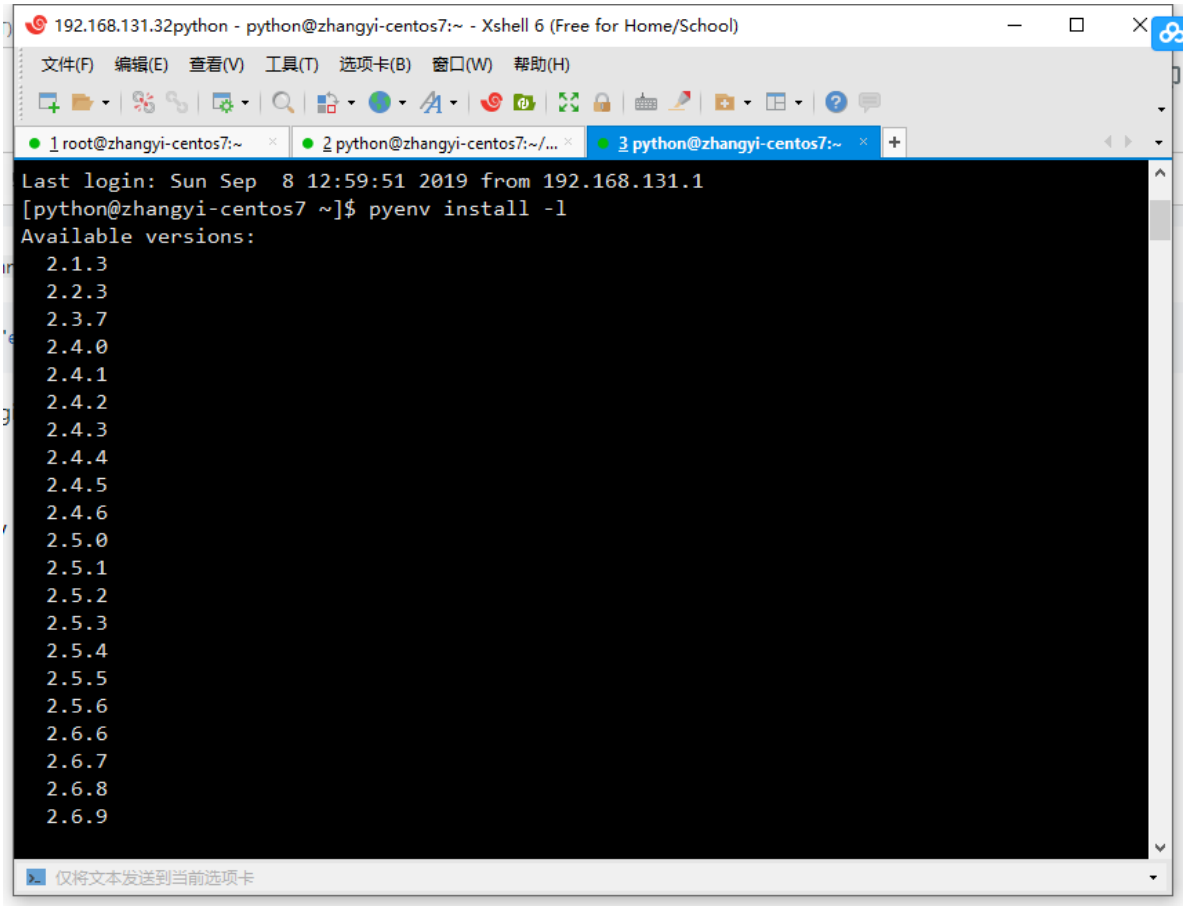
- Initial public release.

MIT - see License file.

2、安装python

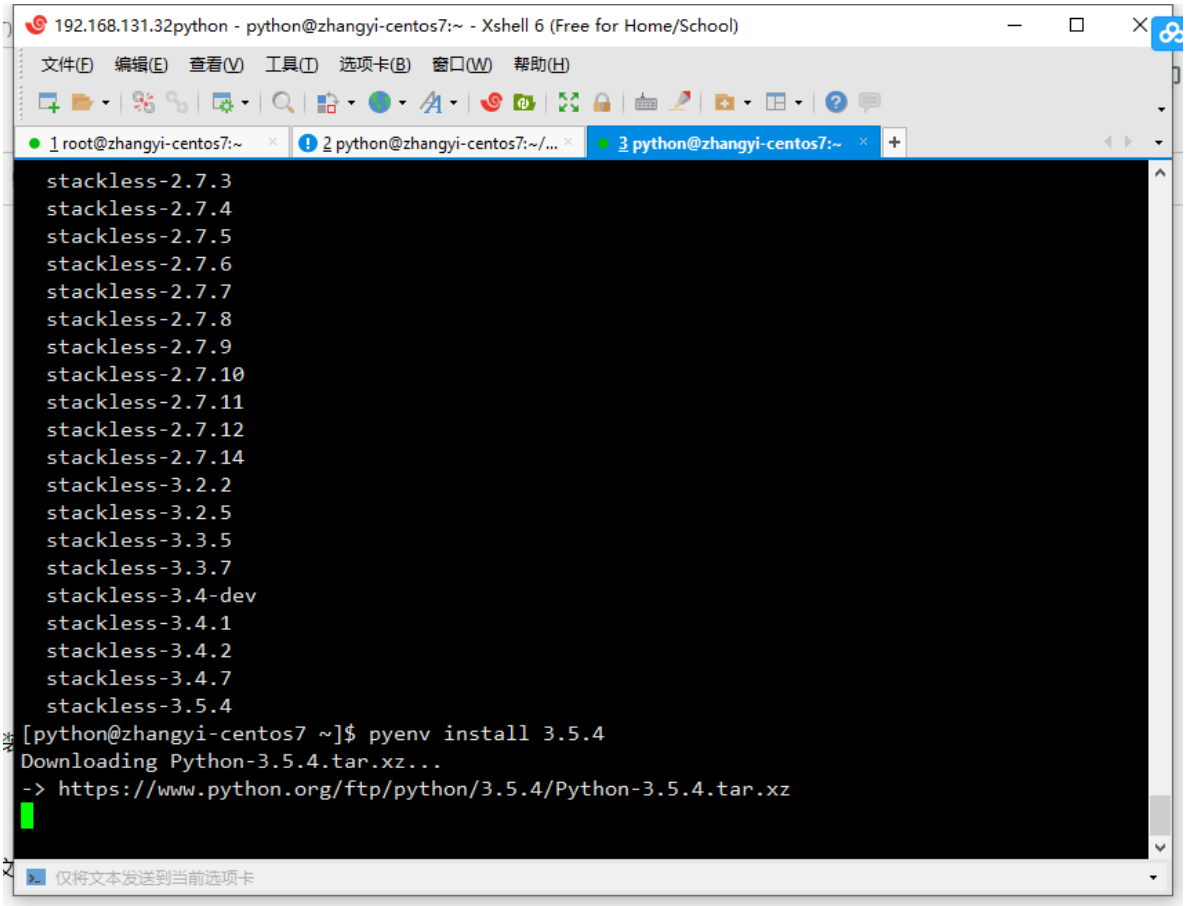
查看python可用版本

```
pyenv install -l
```



在线安装

```
[python@zhangyi-centos7 ~]$ pyenv install 3.5.4
Downloading Python-3.5.4.tar.xz...->
https://www.python.org/ftp/python/3.5.4/Python-3.5.4.tar.xz
```



离线安装

到官网下载 对应版本源码

https://www.python.org/downloads/source/

两个包都下载好

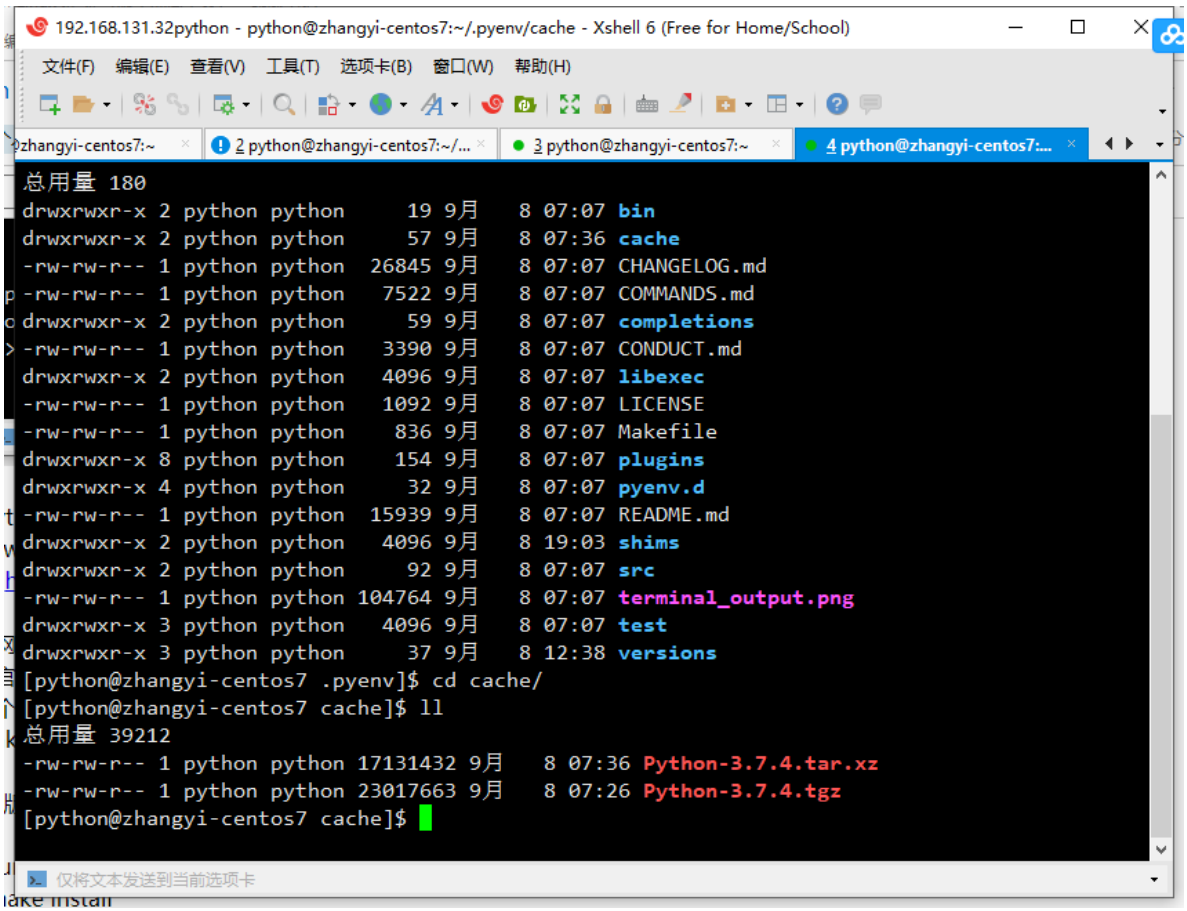
Python-x.x.x.tar.xz

Python-x.x.x.tgz

放入用户目录下的~/.pyenv/cache文件夹

新建文件夹

```
makedir -r ~/.pyenv/cache
```



3、3.7版本依赖问题:

3.7版本需要一个新的包libffi-devel，安装此包之后再次进行编译安装即可。

```
#yum install libffi-devel -y
#make install
```

若在安装前移除了/usr/bin下python的文件链接依赖，此时yum无法正常使用，需要自己下载相关软件包安装，为节省读者时间，放上链接

```
#wget http://mirror.centos.org/centos/7/os/x86_64/Packages/libffi-devel-3.0.13-18.el7.x86_64.rpm
#rpm -ivh libffi-devel-3.0.13-18.el7.x86_64.rpm
```

安装完成后重新进行make install，结束后再次配置相关文件的软连接即可。

4、使用 pyenv 进行版本隔离

查看已安装的python版本

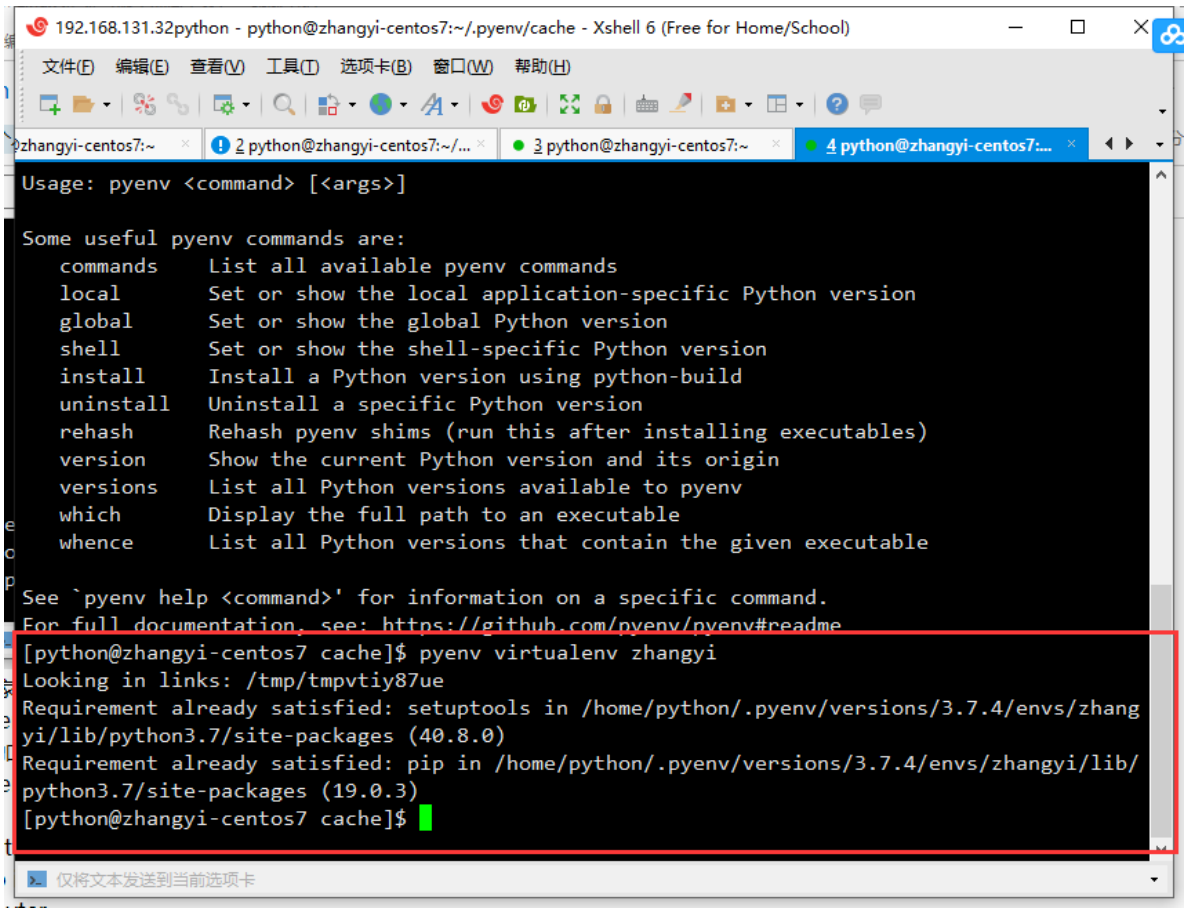
5、把用户目录下的环境设置成新安装的python版本

```
pyenv local 3.7.4
```

6、增加虚拟环境

增加名为zhangyi的虚拟环境

```
pyenv virtualenv zhangyi
```



查看虚拟环境

7、安装ipython

切换pip源

参考博客：https://blog.csdn.net/u011220960/article/details/81512435

Linux系统：

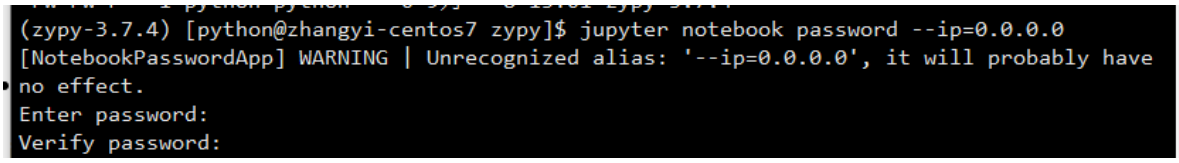
```
mkdir ~/.pip
cat > ~/.pip/pip.conf << EOF
[global]trusted-host=[mirrors.aliyun.com](http://mirrors.aliyun.com/)index-url=https://mirrors.aliyun.com/pypi/simple/
EOF
pip install ipython
```

8、安装jupyter

```
pip install jupyter
```

启动jupyter初始化密码

```
jupyter notebook passwd
```

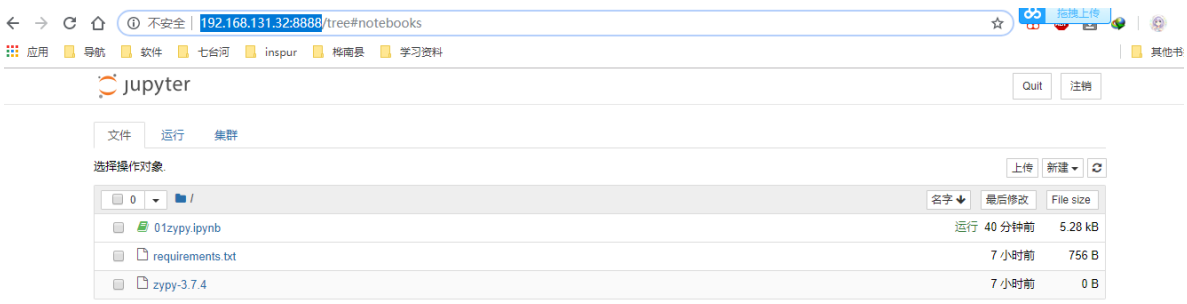


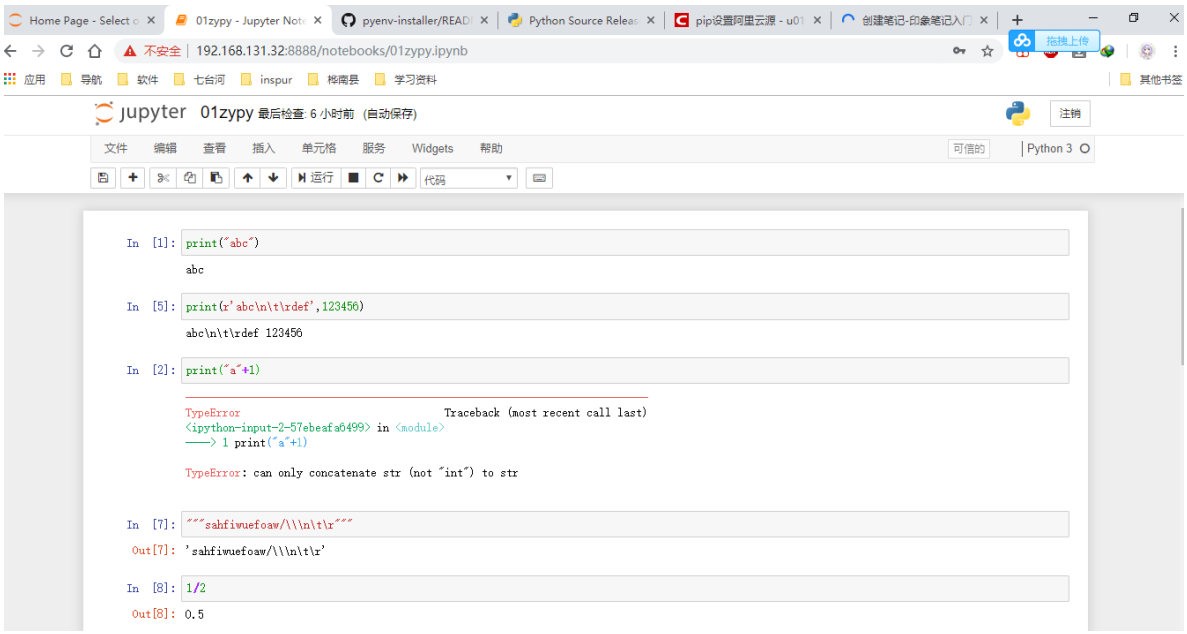
指定jupyter 启动绑定的ip

```
jupyter notebook --ip=0.0.0.0
```

浏览器访问jupyter

http://192.168.131.32:8888





9、Python 虚拟环境包导出

导出包配置文件

pip freeze > requirement

导入包配置文件

pip -r requirement