Pyenv安装

操作系统准备

准备Linux最小系统即可。

如果在虚拟机中克隆，MAC地址会变。 这里使用CentOS 6.5+

pyenv安装方式

git 安装

1、安装git

```
# yum install git -y
```

2、安装Python编译依赖

```
# yum -y install gcc make patch gdbm-devel openssl-devel sqlite-devel
readline-devel zlib-devel bzip2-devel
```

3、创建用户python

```
# useradd python
```

4、使用python用户登录后安装Pyenv

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

下载的 pyenv-installer 是一个shell脚本。

注意：

1. 在 https://github.com/pyenv/pyenv-installer 有安装文档
2. 如果curl出现 curl: (35) SSL connect error ，是nss版本低的问题，更新它。 可能需要配置一个有较新包的yum源

```
[updates]
name=CentOS-Updates
baseurl=https://mirrors.aliyun.com/centos/6.9/os/x86_64
gpgcheck=0
```

然后更新nss # yum update nss

5、在python用户的~/.bash_profile中追加

```
# Load pyenv automatically by adding
# the following to ~/.bash_profile:

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

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

$ source ~/.bash_profile
```

这样当用户启动的时候，会执行用户的.bash_profile中的脚本，就会启动pyenv。 安装好的pyenv就在~/.pyenv中

Pyenv的使用

python 版本及path路径

```
$ python --version
$ python -V
$ echo $PATH
```

可以看到当前系统Python路径

pyenv 命令

```
[python@node ~]$ pyenv
pyenv 1.2.2
Usage: pyenv <command> [<args>]

Some useful pyenv commands are:
   commands    List all available pyenv commands
   local       Set or show the local application-specific Python version
   global      Set or show the global Python version
   shell       Set or show the shell-specific Python version
   install     Install a Python version using python-build
   uninstall   Uninstall a specific Python version
   rehash      Rehash pyenv shims (run this after installing executables)
   version     Show the current Python version and its origin
   versions    List all Python versions available to pyenv
   which       Display the full path to an executable
   whence      List all Python versions that contain the given executable
```

install

$ pyenv help install 列出所有可用版本 $ pyenv install --list

在线安装指定版本 $ pyenv install 3.5.3 $ pyenv versions 这样的安装可能较慢，为了提速，可是选用cache方法。

使用缓存方式安装 在~/.pyenv目录下，新建cache目录，放入下载好的待安装版本的文件。 不确定要哪一个文件，把下载的3个文件都放进去。 $ pyenv install 3.5.3 -v

为了方便演示，请用客户端再打开两个会话窗口。 提前安装备用 $ pyenv install 3.6.4

pyenv的python版本控制

version 显示当前的python版本 versions 显示所有可用的python版本，和当前版本

global 全局设置 $ pyenv global 3.5.3 可以看到所有受pyenv控制的窗口中都是3.5.3的python版本了。 这里用global是作用于非root用户python用户上，如果是root用户安装，请不要使用global，否则影响太大。比如，这里使用的CentOS6.5就是Python2.6，使用了global就成了3.x，会带来很不好的影响。 $ pyenv global system

shell 会话设置 影响只作用于当前会话 $ pyenv shell 3.5.3

local 本地设置 使用pyenv local设置从当前工作目录开始向下递归都继承这个设置。 $ pyenv local 3.5.3

Virtualenv 虚拟环境设置

为什么要使用虚拟环境？ 因为刚才使用的Python环境都是一个公共的空间，如果多个项目使用不同Python版本开发，或者使用不同的Python版本部署运行，或者使用同样的版本开发的但不同项目使用了不同版本的库，等等这些问题都会带来冲突。最好的解决办法就是每一个项目独立运行自己的“独立小环境”中。

使用插件，在plugins/pyenv-virtualenv中 $ pyenv virtualenv 3.5.3 mag353 使用python 3.5.3版本创建出一个独立的虚拟空间。

```
$ pyenv versions
* system (set by /home/python/.pyenv/version)
  3.5.3
  3.5.3/envs/mag353
  mag353
```

可以在版本列表中存在，就和3.5.3是一样的，就是一个版本了。 真实目录在~/.pyenv/versions/下，以后只要使用这个虚拟版本，包就会按照到这些对应的目录下去，而不是使用3.5.3。

```
[python@node ~]$ mkdir -p magedu/projects/web
[python@node ~]$ cd magedu/projects/web/
[python@node web]$ pyenv local mag353
(mag353) [python@node web]$ cd ..
[python@node projects]$ cd web/
(mag353) [python@node web]$
```

pip 通用配置

pip 是Python的包管理工具，3.x的版本直接带了，可以直接使用。 和yun一样为了使用国内镜像，如下配置。

Linux系统 $ mkdir ~/.pip 配置文件在~/.pip/pip.conf

```
[global]
index-url=https://mirrors.aliyun.com/pypi/simple/
trusted-host=mirrors.aliyun.com
```

在不同的虚拟环境中，安装redis包，使用pip list看看效果。 $ pip -V

pip install pkgname 命令，是以后经常要使用的安装python包的命令

windows系统 windows下pip的配置文件在~/pip/pip.ini，内容同上

安装ipython

ipython 是增强的交互式Python命令行工具

```
$ pip install ipython
$ ipython
```

Jupyter 是基于WEB的交互式笔记本，其中可以非常方便的使用Python。 安装Jupyter，也会安装ipython的

```
$ pip install jupyter
$ jupyter notebook help
$ jupyter notebook --ip=0.0.0.0 --no-browser
$ ss -tanl
```

导出包

虚拟环境的好处就在于和其他项目运行环境隔离。每一个独立的环境都可以使用pip命令导出已经安装的包，在另一个环境中安装这些包。

```
(mag353) [python@node web]$ pip freeze > requirement
(mag353) [python@node web]$ mkdir ~/magedu/projects/pro1
(mag353) [python@node web]$ cd ~/magedu/projects/pro1
[python@node pro1]$ pyenv install --list
[python@node pro1]$ pyenv install 3.6.4
[python@node pro1]$ pyenv virtualenv 3.6.4 mag364
[python@node pro1]$ pyenv local mag364
(mag364) [python@node pro1]$ mv ../web/requirement ./
(mag364) [python@node pro1]$ pip install -r requirement
```

python的windows安装

下载 Windows x86-64 executable installer ，按照提示安装即可。 注意，勾选增加PATH路径。

剩下就可以使用pip安装IPython或者其它包和工具了。

pyenv离线安装

首先从github上克隆项目

```
$ git clone https://github.com/pyenv/pyenv.git ~/.pyenv
$ git clone https://github.com/pyenv/pyenv-virtualenv.git ~/.pyenv/plugins
/pyenv-virtualenv
$ git clone https://github.com/pyenv/pyenv-update.git ~/.pyenv/plugins/pyenv-
update
$ git clone https://github.com/pyenv/pyenv-which-ext.git ~/.pyenv/plugins
/pyenv-which-ext
```

可以把克隆的目录打包，方便以后离线使用。

```
$ vim ~/.bash_profile
export PATH="/home/python/.pyenv/bin:$PATH"
eval "$(pyenv init -)"
eval "$(pyenv virtualenv-init -)"

$ source ~/.bash_profile
```