python系列（五）centos6.x中部署多个python版本

原创 Mr大表哥 2017-05-06 15:11:57 评论(1) 230人阅读

博主QQ：819594300
博客地址：http://zpf666.blog.51cto.com/
有什么疑问的朋友可以联系博主，博主会帮你们解答，谢谢支持！

使用pyenv+virtualenv方式部署python多版本

pyenvvs virtualenv

pyenv 是针对 python 版本的管理，通过修改环境变量的方式实现；

virtualenv 是针对python的包的多版本管理，通过将python包安装到一个模块来作为python的包虚拟环境，通过切换目录来实现不同包环境间的切换。

pyenv原理

pyenv 的美好之处在于，它并没有使用将不同的 $PATH 植入不同的 shell 这种高耦合的工作方式，而是简单地在 $PATH 的最前面插入了一个垫片路径（shims）：~/.pyenv/shims:/usr/local/bin:/usr/bin:/bin。所有对Python 可执行文件的查找都会首先被这个 shims 路径截获，从而架空了后面的系统路径。

pyenv 安装使用 （安装使用网络yum源）

下面开始正式安装：

1）确认一下系统版本

```
[root@localhost ~]# cat /etc/redhat-release
CentOS release 6.5 (Final)
[root@localhost ~]#
```

2）安装依赖包

```
[root@localhost ~]# yum -y install gcc python-devel readline readline-devel read
line-static openssl openssl-devel openssl-static sqlite-devel bzip2-devel bzip2-
libs python-setuptools
```

截图中内容如下：

yum-y install gcc python-devel readline readline-devel readline-static opensslopenssl-devel openssl-static sqlite-devel bzip2-devel bzip2-libspython-setuptools

```
[root@localhost ~]# easy_install pip
```

3）安装pyenv

pyenv需要git工具，需要先安装git工具

```
[root@localhost ~]# yum -y install git
```

pyenv提供了自动安装的工具，执行命令安装即可：

```
[root@localhost ~]# curl -L https://raw.githubusercontent.com/yyuu/pyenv-install
er/master/bin/pyenv-installer | bash
```

截图中地址如下：

curl -L https://raw.githubusercontent.com/yyuu/pyenv-installer/master/bin/pyenv-installer| bash

还有一个网址，也是提供pyenv自动安装的工具的，这两个网站都一样，用哪个都行。

curl -Lhttps://raw.github.com/yyuu/pyenv-installer/master/bin/pyenv-installer | bash

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

安装结束后，在结尾会出现上面这三行。

看是否安装成功，执行下面的命令：

```
[root@localhost ~]# echo $?
0
[root@localhost ~]#
```

如果返回的值是0，就表示安装成功。

如果想手动安装，可以执行此命令：

将 pyenv 检出到你想安装的目录。建议路径为：$HOME/.pyenv

$ cd

$ git clone git://github.com/yyuu/pyenv.git.pyenv

4）添加环境变量

```
[root@localhost ~]# vim ~/.bashrc
```

该目录专用于当前用户bash shell的bash信息,当登录时以及每次打开新的shell时,该该文件被读取

在末尾新增加以下几行内容：

```
13 export PYENV_ROOT="${HOME}/.pyenv"
14
15 if [ -d "${PYENV_ROOT}" ]; then
16   export PATH="${PYENV_ROOT}/bin:${PATH}"
17   eval "$(pyenv init -)"
18 fi
```


截图中内容如下：

```
export PYENV_ROOT="${HOME}/.pyenv"

if [ -d "${PYENV_ROOT}" ]; then
  exportPATH="${PYENV_ROOT}/bin:${PATH}"
  eval "$(pyenv init-)"
fi
```

添加完毕后，让其立即生效：

```
[root@localhost ~]# source ~/.bashrc
[root@localhost ~]#
```

至此，pyenv安装也就完成了，接下来看看都有哪些python版本可以安装。

```
[root@localhost ~]# pyenv install --list
Available versions:
  2.1.3
  2.2.3
  2.3.7
  2.4
  2.4.1
  2.4.2
  2.4.3
  2.4.4
  2.4.5
  2.4.6
  2.5
  2.5.1
  2.5.2
  2.5.3
  2.5.4
  2.5.5
  2.5.6
  2.6.6
  2.6.7
  2.6.8
  2.6.9
  2.7-dev
  2.7
  2.7.1
```

```
  2.7.2
  2.7.3
  2.7.4
  2.7.5
  2.7.6
  2.7.7
  2.7.8
  2.7.9
  2.7.10
  2.7.11
  2.7.12
  2.7.13
  3.0.1
  3.1
  3.1.1
  3.1.2
  3.1.3
  3.1.4
  3.1.5
  3.2-dev
  3.2
  3.2.1
  3.2.2
  3.2.3
  3.2.4
  3.2.5
  3.2.6
  3.3.0
  3.3-dev
  3.3.1
```

```
  3.3.2
  3.3.3
  3.3.4
  3.3.5
  3.3.6
  3.4.0
  3.4-dev
  3.4.1
  3.4.2
  3.4.3
  3.4.4
  3.4.5
  3.4.6
  3.5.0
  3.5-dev
  3.5.1
  3.5.2
  3.5.3
  3.6.0
  3.6-dev
  3.6.1
  3.7-dev
```


5）我们看一下，centos6.5默认安装了什么版本的python:

```
[root@localhost ~]# python -V
Python 2.6.6
[root@localhost ~]#
```

从上图可以看出，centos6.5默认安装的是2.6.6版本。

那现在我们安装一个3.6.1版本试一试：

```
[root@localhost ~]# pyenv install 3.6.1
Downloading Python-3.6.1.tar.xz...
-> https://www.python.org/ftp/python/3.6.1/Python-3.6.1.tar.xz
Installing Python-3.6.1...
Installed Python-3.6.1 to /root/.pyenv/versions/3.6.1
```

6）创建虚拟环境

```
[root@localhost ~]# pyenv virtualenv 3.6.1 my-virtual-env-3.6.1
Requirement already satisfied: setuptools in /root/.pyenv/versions/3.6.1/envs/my
-virtual-env-3.6.1/lib/python3.6/site-packages
Requirement already satisfied: pip in /root/.pyenv/versions/3.6.1/envs/my-virtua
l-env-3.6.1/lib/python3.6/site-packages
[root@localhost ~]#
```

其中my-virtual-env-3.6.1是自定义的名称，你也可以自己自定义。

7）列出当前虚拟环境

```
[root@localhost ~]# pyenv virtualenvs
  3.6.1/envs/my-virtual-env-3.6.1 (created from /root/.pyenv/versions/3.6.1)
  my-virtual-env-3.6.1 (created from /root/.pyenv/versions/3.6.1)
[root@localhost ~]#
```

8）激活虚拟环境

```
[root@localhost ~]# pyenv activate my-virtual-env-3.6.1
pyenv-virtualenv: prompt changing will be removed from future release. configure
 `export PYENV_VIRTUALENV_DISABLE_PROMPT=1' to simulate the behavior.
(my-virtual-env-3.6.1) [root@localhost ~]#
```

9）激活虚拟环境后，需要执行下面命令，更新一下数据库

```
(my-virtual-env-3.6.1) [root@localhost ~]# pyenv rehash
(my-virtual-env-3.6.1) [root@localhost ~]#
```

然后直接进入python查看：

```
(my-virtual-env-3.6.1) [root@localhost ~]# python
Python 3.6.1 (default, May  1 2017, 21:52:35)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-18)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

10）查看当前正在使用的版本

```
>>> quit()
(my-virtual-env-3.6.1) [root@localhost ~]# pyenv versions
  system
  3.6.1
  3.6.1/envs/my-virtual-env-3.6.1
* my-virtual-env-3.6.1 (set by PYENV_VERSION environment variable)
(my-virtual-env-3.6.1) [root@localhost ~]#
```

注：其中的星号表示当前正在使用的版本

11）退出虚拟环境

```
(my-virtual-env-3.6.1) [root@localhost ~]# pyenv deactivate
[root@localhost ~]# python -V
Python 2.6.6
[root@localhost ~]#
```

从上图可以看出，我们真实环境依然是2.6.6版本，而虚拟环境里面是3.6.1版本。

12）删除虚拟环境

```
[root@localhost ~]# pyenv virtualenvs
  3.6.1/envs/my-virtual-env-3.6.1 (created from /root/.pyenv/versions/3.6.1)
* my-virtual-env-3.6.1 (created from /root/.pyenv/versions/3.6.1)
[root@localhost ~]# pyenv uninstall my-virtual-env-3.6.1
```

出现一个？，输入一个y，继续执行即可。

如果想在两个版本之间切换，在每次想使用3.6.1版本的时候就输入：

pyenv activate my-virtual-env-3.6.1

再输入python，就可以进3.6.1版本了。

不想用3.6.1版本了，就退出来，使用：

pyenvdeactivate命令就退出来了，可以使用我们的2.6.6版本。

如果不想那么麻烦怎么办？

好办，那就再创建一个虚拟环境，用来运行2.6.6版本。

```
[root@localhost ~]# pyenv virtualenv my-virtual-env-2.6.6
```

说明：若不指定python 版本，则默认使用当前环境python版本。

再看一下当前所有的虚拟环境：

```
[root@localhost ~]# pyenv virtualenvs
  3.6.1/envs/my-virtual-env-3.6.1 (created from /root/.pyenv/versions/3.6.1)
  my-virtual-env-2.6.6 (created from /usr)
  my-virtual-env-3.6.1 (created from /root/.pyenv/versions/3.6.1)
[root@localhost ~]#
```

激活2.6.6虚拟环境：

```
[root@localhost ~]# pyenv activate my-virtual-env-2.6.6
pyenv-virtualenv: prompt changing will be removed from future release. configure
  `export PYENV_VIRTUALENV_DISABLE_PROMPT=1' to simulate the behavior.
(my-virtual-env-2.6.6) [root@localhost ~]#
```

更新一下数据库：

```
(my-virtual-env-2.6.6) [root@localhost ~]# pyenv rehash
(my-virtual-env-2.6.6) [root@localhost ~]#
```

至此，2.6.6和3.6.1的两个虚拟环境都有了，现在开始更方便的切换：

```
[root@localhost ~]# python -V
Python 2.6.6
[root@localhost ~]# pyenv local my-virtual-env-3.6.1
[root@localhost ~]# python -V
Python 3.6.1
[root@localhost ~]# pyenv local my-virtual-env-2.6.6
[root@localhost ~]# python -V
Python 2.6.6
[root@localhost ~]#
```


就只需要这两条命令就可以在当前真实环境下来回的切换两个版本了，是不是很方便呢。

额外增加一个知识点说明：

如果你只有3.6.1虚拟环境，而没有创建2.6.6虚拟环境，当你执行：

pyenvlocal my-virtual-env-3.6.1

即：在当前环境下，切换了3.6.1版本，切换过来是没有问题，主要是你想过没有，当你想用2.6.6版本怎么办？切不回去了，因为没有2.6.6虚拟环境，你执行不了：pyenvlocal my-virtual-env-2.6.6这个命令，这时候你再去新创建一个2.6.6虚拟环境，已经来不及了，具体的解决办法就是去~/.bashrc里面把新增加那几行删除或注释，然后删除3.6.1虚拟环境，最后重启系统即可真实环境恢复python2.6.6版本，然后你再把上面的实验再来一遍，记住要创建两个虚拟环境，这样来回切换就很方便了。