目录
- Python教程
  - Python简介
  - 安装Python
  - 第一个Python程序
  - Python编辑器
  - 模块
  - 数据类型
  - 字符串和编码
  - 列表
  - 条件判断和循环
  - 使用dict和set数据类型
  - 函数，调用函数
  - 定义函数
  - 函数的参数
  - 递归函数
  - 切片
  - 迭代对象和迭代器
  - 列表生成式
  - 生成器
  - 迭代器
  - 小结
  - 练习题
关于作者
- 廖雪峰 V2-8893
- 关注作者

廖雪峰
在已有Java基础
Python商业黄金金解码
以天下没有易学的语言！
Python编程 | 数据分析
Python高级学习 | 深度学习
找摩雪峰老师
廖雪峰老师
自己的Java课程
Java高级架构师
更专业 更权威
源码分析专题 | 性能优化专题
架构设计与实践 | 性能优化专题
找摩雪峰老师
1 python免费公开课
投资理财、在线直播+课后答疑，从零基础到实战开发工程师
猿辅导 CEO
2 视频通话SDK
声网Agora，API 简单，4行代码集成，每月10分钟免费。
www.agora.io
1 python免费公开课
投资理财、在线直播+课后答疑，从零基础到实战开发工程师
猿辅导 CEO
2 视频通话SDK
声网Agora，API 简单，4行代码集成，每月10分钟免费。
www.agora.io

virtualenv

编辑: 廖雪峰 2017/3/3

在不同的Python应用程序中时候，系统的默认的Python版本一般不变，比如系统工具和系统模块（位于目录/Library/Python/2.7/site-packages）的某个版本。

如果要求某个应用程序必须使用一个特定版本，则应该使用virtualenv来创建一个Python，那表示该应用程序的Python 3，则用应用程序的virtualenv创建一个Python 3的工具？

这样做了，每个应用程序都有自己的Python，不管系统工具的Python版本怎么升级，也不影响应用程序为一个应用程序的virtualenv创建的Python。

例如，我已经安装了virtualenv：

```
$ pip install virtualenv
```

然后，假设我打算开始一个新的项目，需要一个新的Python运行环境，可以这么做：

```
$ cd your_project_root
$ virtualenv --no-site-packages venv
Running virtualenv with interpreter /usr/bin/python2.7
New python executable in venv/bin/python
Installing setuptools, pip, wsgiref, boto
```

命令virtualenv venv将创建一个纯净的Python运行环境，我们把这个目录叫做venv。这个目录里现在就类似系统的Python，也有自己的site-packages和一些工具：

```
$ ls venv
bin  include  lib  local  pyvenv.cfg
```

然后，在venv/bin目录下，有一个activate的脚本，用来让venv这个Python环境生效：

```
$ source venv/bin/activate
(venv)$ which python
venv/bin/python
(venv)$ deactivate
$ which python
/usr/bin/python
```

注意：在venv目录下，我们输入了命令source venv/bin/activate后，Python的运行环境被切换到了venv的Python，而且，命令提示符也变成了(venv)$的样子，提醒用户当前的Python是一个virtualenv。

在venv/bin目录下，还有一个easy_install，用来安装第三方的package，例如：

```
(venv)$ easy_install -U -i http://pypi.douban.com/simple/ jinja2
Searching for jinja2
Reading http://pypi.douban.com/simple/jinja2/
Reading http://jinja.pocoo.org/
Reading http://pypi.python.org/pypi/Jinja2
Best match: Jinja2 2.6b1
Downloading http://pypi.douban.com/packages/source/J/Jinja2/Jinja2-2.6b1.tar.gz#md5=ddb0d0a0d0d0d0d0d0d0d0d0d0d0d0
Processing Jinja2-2.6b1.tar.gz
Running setup.py -q bdist_egg --dist-dir /tmp/tmpwrqxzu
warning: Jinja2 2.6b1 is a development or incoming release
warning: Build of Jinja2 may not work
Adding Jinja2 2.6b1 to easy-install.pth
Installed /home/ubuntu/venv/lib/python2.7/site-packages/Jinja2-2.6b1-py2.7-linux-i686.egg
Finished processing dependencies for jinja2
```

现在，假设你要开始一个新的项目，需要一个纯净的Python运行环境，这个目录应该放到哪里呢？

我的习惯是，把virtualenv的venv放到项目根目录下，例如：/srv/python/myproject。这样做的好处是，venv目录随项目一起移动，比如，我在测试服下工作的目录是/srv/python/myproject，在生产环境下，我把这个目录移动到/srv/python/myproject，这样，只要在venv/bin目录下执行source activate，就可以在新的环境下工作了。

要退出virtualenv，只需输入命令deactivate，venv/bin目录下的activate脚本，用来在当前shell环境下把一个特定的virtualenv变成可用的Python环境。

在venv/bin目录下，还有一个pip，用来安装第三方的package，例如：

```
(venv)$ pip install -U -i http://pypi.douban.com/simple/ jinja2
Downloading/unzipping jinja2 (from -r requirements.txt)
  Running setup.py egg_info for package jinja2
    warning: Jinja2 2.6b1 is a development or incoming release
    warning: Build of Jinja2 may not work
Installing collected packages jinja2
  Running setup.py install for jinja2
    warning: Jinja2 2.6b1 is a development or incoming release
    warning: Build of Jinja2 may not work
    Adding Jinja2 2.6b1 to easy-install.pth
    Installed /home/ubuntu/venv/lib/python2.7/site-packages/Jinja2-2.6b1-py2.7-linux-i686.egg
Successfully installed jinja2 2.6b1
Cleaning up...
```

pip可以安装任何Python的package，并且，可以把当前venv/bin目录下的所有package的名称和版本号输出到一个文本文件：

```
(venv)$ pip freeze -r requirements.txt > requirements.txt
(venv)$ ls requirements.txt
requirements.txt
(venv)$ cat requirements.txt
Jinja2==2.6b1
wsgiref==0.1.2
wsgiref==0.1.2
wsgiref==0.1.2
```