Table of Contents

Introduction

虚拟环境

虚拟环境 virtual environment

介绍

使用虚拟环境安装开发环境，可以避免包的混乱和版本的冲突。 虚拟环境是Python解释器的副本，在虚拟环境中你可以安装扩展包，为每个程序单独创建的虚拟环境，可以保证程序只能访问虚拟环境中的包。 不会影响系统中安装的Python解释器，从而保证解释器的整洁。

单独说明

本文档主要说明以Ubuntu为例的Liunx系统以及Mac OS系统下Python虚拟开发环境的配置。 Windows下命令略有不同，故单独说明。

学习要求

掌握Ubuntu、Mac系统中的Python虚拟开发环境配置使用即可，原因如下：

- 实际开发中，用远程调试，服务器为Linux系统。
- 课程学习中
  - 可以在Ubuntu、Mac系统中进行学习。
  - 可以在Windows系统中：
    - 通过Xshell连接Ubuntu虚拟机进行终端操作；
    - 通过Pycharm连接Ubuntu虚拟机进行远程调试。

复习

1.pip

安装**Python**的包

```
pip install 包名称
pip3 install 包名称
```

查看当前环境下安装的**Python**包

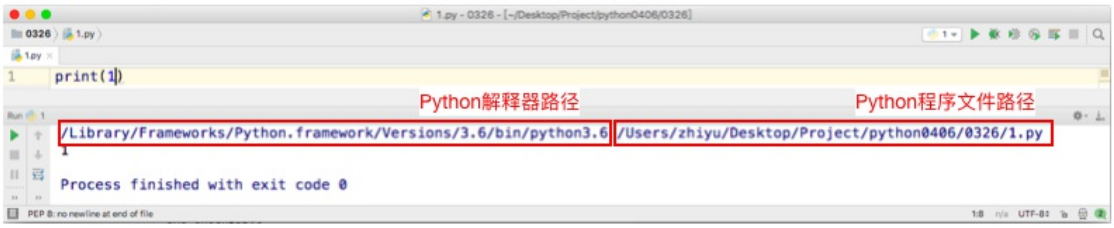
`pip list` 或者 `pip freeze`

2.查看当前使用**Python**解释器的路径

Pycharm中

在运行结果上方可以看到

- 当前使用的Python解释器的路径
- 当前运行的Python程序的路径



终端/命令行/控制台中

- 可以在终端直接输入 `which python` 查看
- 在Python环境中执行如下命令查看

```
import sys
sys.executable
```

此处要求区分以下两条命令：

- `sys.executable` 查看解释器路径
- `sys.path` 查看包路径

3.重定向

- `>` 将输出结果写在目标文件中，新建或覆盖目标文件
- `>>` 将输出结果追加在目标文件末尾

虚拟环境包安装配置

虚拟环境包安装

```
sudo pip install virtualenv
sudo pip install virtualenvwrapper
sudo easy_install virtualenvwrapper  # Mac（用上一条不成功时用这一条）
```

虚拟环境包配置

1、创建目录用来存放虚拟环境

```
mkdir $HOME/.virtualenvs
```

2、打开~/.bashrc(Mac下为.bash_profile)文件，并添加如下：

```
export WORKON_HOME=$HOME/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

注意： Mac的 `virtualenvwrapper.sh` 不在上面代码对应的目录下时：

- 用 `which virtualenvwrapper.sh` 找到 `virtualenvwrapper.sh` 的路径；
- 在上面的代码中写在相应的位置。

3、加载配置

```
 source ~/.bashrc
 source ~/.bash_profile  # Mac
```

创建、使用虚拟环境

创建虚拟环境

此处注意创建虚拟环境的版本

此处以web开发中Flask、Django对应的虚拟环境为例。 如需创建web开发中的Tornado、爬虫开发中的Scrapy等环境，可以类推。

小白特别注意此处：虚拟环境名仅是一个名字，需要使用的模块还要后续配置，创建虚拟环境时重要的是**python**版本选择,虚拟环境名能够见名知意并方便选择即可。

```
# 创建Python2虚拟环境
mkvirtualenv flask_py2
     # 以Flask为例，虚拟环境名称自取，此处为了给Flask使用起名为flask_py2
# 创建Python2虚拟环境
mkvirtualenv -p python3 django_py3
     # 以Django为例,起名原因同上
```

此处虚拟环境名仅作举例，**Flask**、**Django**使用不用版本**python**时注意**Python**版本

如新经资讯项目使用Python3的Flask：

```
mkvirtualenv -p python3 f3pyinfo
```

虚拟环境创建成功后，会自动工作在这个环境上

命令行前面有`(虚拟环境名)`表示在这个虚拟环境中

```
deactivate  # 退出虚拟环境
```

使用虚拟环境

没有工作在虚拟环境上的时候

```
# 命令
workon
```

`workon 虚拟环境名称 # 工作到对应的虚拟环境上`

`workon + 两次tab # 显示可以选用的虚拟环境`

在虚拟环境中配置开发环境

获取配置需要的文件

在配置好的虚拟环境中执行命令
注意目录（可自选，此处放在了桌面）

```
pip freeze > ~/Desktop/requirements.txt
```

在当前虚拟环境中进行配置

```
pip install -r ~/Desktop/requirements.txt
```

配置成功,其他常见虚拟环境命令

常用
列举虚拟环境用 `workon+两次tab`
查看虚拟环境中的包，可在虚拟环境下 `pip list` 或者 `pip freeze`

不常用

```
lsvirtualenv    # 列举所有的环境。
cdvirtualenv    # 导航到当前激活的虚拟环境的目录中，比如说这样您就能够浏览它的 site-packages。
cdsitepackages   # 和上面的类似，但是是直接进入到 site-packages 目录中。
lssitepackages     # 显示 site-packages 目录中的内容。
```

退出、删除虚拟环境

退出虚拟环境

```
deactivate  # 退出虚拟环境
```

删除虚拟环境

```
deactivate  # 退出虚拟环境
rmvirtualenv 虚拟环境名称  # 删除虚拟环境
```

拓展复习

根据端口号杀死进程

-> 根据程序端口号查看进程号

-> 根据进程号杀死进程

Ubuntu&Mac

```
lsof -i:portid  # 根据程序端口号查看进程号
kill -9 pid  # 根据进程号杀死进程
```

Windows

```
netstat -aon|findstr "portid"  # 根据程序端口号查看进程号
taskkill /pid "pid" -f  # 根据进程号杀死进程
```

Linux命令之rm、cp、mv

rm

可通过rm删除文件或目录。

- 使用rm命令要小心，因为文件删除后不能恢复。
- 为了防止文件误删，可以在rm后使用-i参数以逐个确认要删除的文件。

常用参数及含义如下表所示：

参数	含义
-i	以进行交互式方式执行
-f	强制删除，忽略不存在的文件，无需提示
-r	递归地删除目录下的内容，删除文件夹时必须加此参数

cp

cp命令的功能是将给出的文件或目录复制到另一个文件或目录中，

- 相当于DOS下的copy命令。

常用选项说明：

选项	含义
-a	该选项通常在复制目录时使用，它保留链接、文件属性，并递归地复制目录，简单而言，保持文件原有属性。
-f	已经存在的目标文件而不提示
-i	交互式复制，在覆盖目标文件之前将给出提示要求用户确认
-v	显示拷贝进度

mv

- 用户可以使用mv命令来移动文件或目录，也可以给文件或目录重命名。

常用选项说明：

选项	含义
-f	禁止交互式操作，如有覆盖也不会给出提示
-i	确认交互方式操作，如果mv操作将导致对已存在的目标文件的覆盖，系统会询问是否重写，要求用户回答以避免误覆盖文件
-v	显示移动进度

Windows配置虚拟环境

将文件拖拽到命令行窗口，快速得到文件路径（在windows系统中更常用）。

虚拟环境包安装配置

配置内部命令**pip2**、**pip3**

- 安装Python解释器时自动配置环境变量，pip即是内部命令；
- 如果提示pip不是内部命令，把python文件下的Scripts的文件路径加入计算机环境变量。

既有Python2又有Python3时为pip对应最后安装的python解释器。

安装虚拟环境包

```
pip2 install virtualenv
pip3 install virtualenv
```

区分**Python2**和**Python3**的**virtualenv**

将python2目录下的Scripts目录里的virtualenv.exe改为virtualenv2.exe，在保证python的环境变量都加到了计算机环境变量的情况下。 我们就可以 使用'virtualenv2 虚拟环境名' 来创建py2的虚拟环境， 用'virtualenv 虚拟环境名' 创建py3的虚拟环境了。

建议将**python3**目录下的**Scripts**目录里的**virtualenv.exe**改为**virtualenv3.exe** 以严格区分防止误操作。

创建存放虚拟环境的文件夹并切换到该目录

- `md [盘符:\][路径\]新目录名` 或者 `mkdir [盘符:\][路径\]新目录名`
- `cd [盘符:\][路径\]新目录名`

  可以在窗口中完成上面的创建及打开目录的操作，并在该目录下 `shift+右键` 选择进入power shell进行命令操作。

创建虚拟环境

```
virtualenv2 虚拟环境名  # 创建Python2解释器对应的虚拟环境
virtualenv3 虚拟环境名  # 创建Python3解释器对应的虚拟环境
```

- Windows中创建虚拟环境之后没有自动工作在这个虚拟环境中。
- 在虚拟环境中工作前需要激活。

激活虚拟环境

- cd进入虚拟环境下的Scripts文件夹

- activate 激活虚拟环境
  - 命令行前面有`(虚拟环境名)`表示在这个虚拟环境中

在虚拟环境中**pip**安装要使用的模块

```
pip install -r requirements.txt
```

退出虚拟环境

`deactivate`退出虚拟环境