MAC下开发环境的搭建

相关账号：

MAC下虚拟机的ip地址：192.168.99.140（注意：这个是虚拟机网络桥接模式下的ip地址。非桥接下的ip地址10.0.2.15不能ping通，更不能ssh连接上）

Root密码和开机密码都是：End？

1. 安装虚拟机

下载Ubuntu14.04或者16.04版本

下载Virtual Box及其扩展Extension Pack。

新建虚拟机，名称按照Ubuntu 14.04（for Django）的形式。关于“文建位置和大小”，虚拟硬盘（.vdi）可以安装在任意位置。当前安装在zhangjinxiong/VirtualBox VMs/Ubuntu 16.04(for Django)下

当前虚拟机开机密码：End

1. 在虚拟机Linux终端中设置环境

进入终端的快捷键：control+option+T

当前终端密码：F12

更新系统信息：

Sudo apt-get update

Sudo apt-get –y upgrade

安装pip：

Sudo apt-get –y install python-pip

安装虚拟机环境virtualenv库（应该是在/home/Jason/VENV/bin/python中执行，如果jason是linux登录用户名的话）：

Sudo pip install virtualenv

查到本机IP地址（本机为192.168.99.140）：  
ip a

注意：需要虚拟机的网络连接是桥接模式。非桥接模式下的ip地址不能用

安装OpenSSH服务器（用来使用Windows的PUTTY连接或者Mac的终端连接）：

Sudo apt-get –y install openssh-server

安装FTP服务器：

Sudo apt-get –y install vsftpd

在本机环境直接操作虚拟机Linux环境：

Windows下通过PuTTY；Mac OS可直接在终端程序中使用ssh指令连接；也可通过IDE直接连接

1. 在Linux中使用Git

Github账号：zhangjinxiong F12

安装Git：

Sudo apt-get install –y git

Git全局设置（用户名称和账号需要和Github中一致）：

Git config - -global user.name “zhangjinxiong77”

Git config - -global user.email [876602469@qq.com](mailto:876602469@qq.com)

1. 创建新项目（myweb）并提交到Github

Sudo apt-get install –y python-pip （前面已执行过）

Sudo pip install virtualenv （前面已执行过）

Virtualenv VENV （在Home下新建了VENV文件夹。？VENV是否可更换其它名称？）

Source VENV/bin/activate （启动VENV，进入VENV环境。如果中途退出后。再从此环节进入）

（VENV）$ pip install django （安装Django，应该是安装在了VENV文建夹中。如果中途退出再进入，这一步应该不需要了）

（VENV）$ django-admin startproject myweb （创建Django项目myweb。新建了myweb文件夹，和VENV文件夹同一层级。也就是说，同一个虚拟环境可以创建多个项目。如果中途退出再进入，这一步应该不需要了）《Django架站的16堂课》

在《Python高效开发实战中》，VNEV是在项目所在目录创建的，这样就会作为该项目专用的虚环境。

虚环境复制了一份完整的当前系统Python环境。之后运行Python时可以直接运行该目录bin文件夹中的命令。比如 ./venv/bin/pip install tornado 或者 ./venv/bin/python xxxx.py。方法二是用activate命令启动虚环境，之后不必再显式地调用虚环境bin文件夹中的命令（如本例）。退出虚环境：deactivate命令

在Github中建立同名仓库myweb

进入myweb文件夹并初始化git

（VENV）$ cd myweb

（VENV）$ git init

（VENV）$ git remote add origin <https://github.com/zhangjinxiong77/myweb.git>

（origin为自定义，将成为远程仓库地址的别名）

（VENV）$ git add . （.点代表all，点号之前要加空格）

（VENV）$ git commit –m ‘first commit’

（VENV）$ git push origin master （将myweb中的内容提交到Github中origin的master分支）

五、第四步只同步了项目（myweb）文件夹，没有同步Python虚拟机环境（此例为VENV）和此项目中使用pip安装的套件（记录在虚拟机环境目录中，此例为VENV文件夹）

使用pip freeze建立套件列表，并将套件列表创建到项目的同一个目录中。此例为myweb。

指令如下（在myweb目录下操作）

（VENV）$ pip freeze > requirements.txt

（VENV）$ git add .

（VENV）$ git commit –m ‘add requirements.txt’

（VENV）$ git push origin master

注意事项：随着项目的进展，如果有新安装的python套件，需要使用 pip freeze更新requirements.txt

六、在另一台计算机中同步项目

七、创建第一个正式项目mblog

新建myDjango文件夹，作为Django所有项目的专用文件夹

cd myDjango 进入myDjango文件夹

Virtualenv VENV 在myDjango中建立VENV文件夹（虚环境）

Source VENV/bin/activate 进入虚环境

(VENV) ~/myDjango $ pip install django在虚环境中安装Django（注意：上例的Django是在上例的虚环境装的，和本例的虚环境没有关系）

(VENV) ~/myDjango $ django-admin startproject mblog 在myDjango目录下利用本虚环境创建项目。执行命令后，在myDjango目录下新建了mblog文件夹（理解：在本虚环境下，使用的命令都是该虚环境的命令；如果不在虚环境下，使用的命令则是系统的Python环境命令）。在mblog目录下，还有一个mblog同名文件夹。内层mblog文件夹用于存放urls》py等本项目的实际文件。\_init\_.py用来告诉Python该目录是一个Python包。

(VENV) ~/myDjango $ cd mblog

(VENV) ~/myDjango/mblog $ python manage.py startapp mainsite 执行mblog（指的是外层的mblog目录）中manage.py 中的startapp，在mblog中创建应用mainsite及同名文件夹。此时在settings.py的INSTALLED\_APPS中会有mainsite（or 手动加入？）

(VENV) ~/myDjango/mblog $ cd .. 返回myDjango目录

(VENV) ~/myDjango $ tree mblog 查看mblog目录的结构（如果在myDjango目录中创建其它项目，则还会有和外层mblog、VNEV同级的文件夹）——注意：前提是已经安装了tree命令（安装方法：）在系统目录下 sudo apt-get install –y tree

下一步，测试网站

在myDjango目录下，进入虚环境。然后进入mblog目录

(VENV) ~/myDjango/mblog $ python manage.py runserver 192.168.99.140:8000 启动服务器

在浏览器中输入网址：http:// 192.168.99.140:8000 进行测试。（可在虚拟机中linux的浏览器，也可在外层系统的浏览器）

——测试时，可能会报错DisallowedHost : Invalid HTTP\_HOST header。原因在于在项目目录（内层mblog文件夹）下的settting.py有个字段：ALLOWED\_HOSTS = []，表示允许访问的主机，对应的值为空，则表示不允许所有的主机访问该服务器。需要修改该字段的值为：ALLOWED\_HOSTS = ['\*']，设置允许所有主机都可以访问。修改方法：先关闭服务器（ctrl+C），修改好文件之后，重启服务器。——类似于白名单？

# 最后，git同步

进入mblog文件夹并初始化git

（VENV）~/myDjango/mblog $ cd mblog

（VENV）~/myDjango/mblog $ git init

（VENV）~/myDjango/mblog $ git remote add origin

<https://github.com/zhangjinxiong77/mblog.git>

（VENV）~/myDjango/mblog $ git add . （.点代表all，点号之前要加空格）

（VENV）~/myDjango/mblog $ git commit –m ‘first commit’

（VENV）~/myDjango/mblog $ git push origin master

复制环境并git同步

（VENV）~/myDjango/mblog $ pip freeze > requirements.txt

（VENV）~/myDjango/mblog $ git add .

（VENV）~/myDjango/mblog $ git commit –m ‘add requirement.txt’

（VENV）~/myDjango/mblog $ git push origin master

网站的初步设置：

将创建的APP模块（本例为mainsite），加入settings.py的INSTALL\_APPS中，并在settings.py中修改语言和时区配置：

LANGUAGE\_CODE = ‘zh-Hans’

TIME\_ZONE = ‘Asia/Shanghai’

运行migrate，产生db.sqlite3文件（？外层mblog文件夹下不是有该文件么？）：

（VENV）~/myDjango/mblog $ python manage.py migrate 使数据库状态与当前模型集和迁移集同步。说白了，就是将对数据库的更改，主要是数据表设计的更改，在数据库中真实执行。例如，新建、修改、删除数据表，新增、修改、删除某数据表内的字段等等。

八、在Mac终端中连接虚拟机Ubuntu的方法

上面已经安装好了openSSH

虚拟机的ip地址：192.168.99.140

直接在Mac终端中 ssh [jason@192.168.99.140](mailto:jason@192.168.99.140) 即可。Jason是虚拟机的用户名

九、在windows中安装虚拟机

下载Vmware workstation Player。

全名：Jason

用户名：jason

密码：F12

虚拟机名称：Ubuntu14.04

更新系统信息：

Sudo apt-get update

Sudo apt-get –y upgrade

查到本机IP地址（Windows中虚拟机为192.168.99.130）：  
ip a

注意：需要虚拟机的网络连接是桥接模式。非桥接模式下的ip地址不能用

安装OpenSSH服务器（用来使用Windows的PUTTY连接或者Mac的终端连接）：

Sudo apt-get –y install openssh-server

安装FTP服务器：

Sudo apt-get –y install vsftpd

安装Git：

Sudo apt-get install –y git

Git全局设置（用户名称和账号需要和Github中一致）：

Git config - -global user.name “zhangjinxiong77”

Git config - -global user.email [876602469@qq.com](mailto:876602469@qq.com)

安装pip：

Sudo apt-get –y install python-pip

安装虚拟机环境virtualenv库（应该是在/home/Jason/VENV/bin/python中执行，如果jason是linux登录用户名的话）：

Sudo pip install virtualenv

新建myDjango文件夹，作为Django所有项目的专用文件夹

cd myDjango 进入myDjango文件夹

Virtualenv venv 在myDjango中建立venv文件夹（虚环境）

source venv/bin/activate 进入虚环境。因为venv在myDjango目录，所以从myDjango目录进入虚环境

(venv) ~/myDjango $ pip install django在虚环境中安装Django（注意：上例的Django是在上例的虚环境装的，和本例的虚环境没有关系）

(venv) ~/myDjango $ git clone <https://github.com/zhangjinxiong77/mblog.git> 在myDjango下进行git，将mblog项目同步过来

(venv) ~/myDjango $ cd mblog

(venv) ~/myDjango/mblog $ git remote 可以看到mblog的远程分支已经是origin。因此不需要在mblog文件夹下git init或者git remote add了

(venv) ~/myDjango/mblog $ git pull origin master 在项目文件夹下进行git，将最新的分支拉过来（如果有最新分支，才需要拉）

(venv) ~/myDjango/mblog $ pip install –r ‘requirements.txt’ 安装所有使用到的套件

测试是否都同步完成

(VENV) ~/myDjango/mblog $ python manage.py runserver 192.168.99.130:8000 启动服务器

在浏览器中输入网址：192.168.99.130:8000 进行测试。

十、windows安装PUTTY并连接虚拟机

设置Host Name、Session连接的名称、字体、定时发送数据封包（60秒），然后保存。保存后，进行连接（open）

测试方法：

进入myDjango->进入虚环境->进入mblog->启动服务器->浏览器测试

十一、使用Pycharm编辑虚拟机中的文件

Tools -> deployment –> configuration。点击左侧的加号按钮，新增一个连接。协议选择sftp。填写完成之后，测试是否配置正确，点击Test SFTP connection。完成之后点击OK即可

重新回到Pycharm主界面，Tools -> deployment –>browse remote host。成功连接之后，虚拟机中的所有文件会显示在右侧，就可以进行编辑了。

（讨论：正确的团队合作开发方式是否应当是：① 在服务器Ubuntu中新建项目代码，然后上传到Git ② Mac或者Windows的虚拟机Ubuntu从Git中拉取代码分支 ③ Pycharm连上虚拟机中的代码进行开发 ④ 在本地虚拟机Ubuntu中测试无误后，再提交到Ubuntu中）

十二、启动Admin管理界面

创建管理员账号和密码：

(venv) ~/myDjango/mblog $ python manage.py createsuperuser

账号：admin

邮箱：[876602469@qq.com](mailto:876602469@qq.com)

密码：F12

**设置Apache和配置域名**

P82