## **环境配置**

环境ubuntu16.04   
python3.6 、 Django1.8.2  
安装python3 sudo apt-get install python3  
安装虚拟环境参考博客(<https://blog.csdn.net/da953824/article/details/87946504>)  
下面博客的内容：  
安装虚拟环境(须在联网状态下)  
$ sudo pip install virtualenv  
$ sudo pip install virtualenvwrapper  
安装完虚拟环境后，如果提示找不到mkvirtualenv命令，须配置环境变量：

1、创建目录用来存放虚拟环境  
mkdir $HOME/.virtualenvs

2、打开~/.bashrc文件，并添加如下：  
export WORKON\_HOME=$HOME/.virtualenvs   
source /usr/local/bin/virtualenvwrapper.sh

3、运行  
source ~/.bashrc

4 创建虚拟环境(ubuntu里须在联网状态下)  
$ mkvirtualenv django\_py3 -p python3  
进入虚拟环境

$ workon django\_py3 进入项目目录 req.txt所在目录  
安装相应python包 pip install -r req.txt 如果自己安装python包，版本请一致，因为有些PY包会导致django版本的变动 然后将项目目录中的ChineseAnalyzer.py和whoosh\_cn\_backend.py复制到/home/python/.virtualenvs/bjll\_py3/lib/python3.5/site-packages/haystack/backends，  
前面的部分是/home/python/是家目录，.virtualenvs包含所有你创建的虚拟环境，然后一直往下就是你安装haystack的目录，  
你需要加这两个文件，才能支持中文分词搜素，runserver时才不会出现没有whoosh\_cn\_backend.py的错误

## **数据库安装**

### **mysql**

安装服务端   
sudo apt-get install mysql-server   
安装客户端   
sudo apt-get install mysql-client   
安装libmysqlclient，这个在使用开发工具连接数据库使用会用到   
sudo apt-get install libmysqlclient-dev

### **redis**

Ubuntu上直接sudo apt-get install redis-server   
ps -aux|grep redis  
让redis服务器可远程登陆，修改redis配置文件，sudo vim /etc/redis/redis.conf  
将bind 127.0.0.1注释掉  
这里我没注释，因为我的redis并没有远程连接的需求 重启redis服务 sudo /etc/init.d/redis-server restart  
如果过段时间服务器报错，可能是服务器内存小，也可能是你的redis内存被别人的程序利用的情况，  
如果是第一种可以通过设置配置文件的maxmemory参数设置最大占用内存大小,默认是不设的，然后设置maxmemory-policy删除策略为 allkeys-lru最近最少使用的删除，以避免redis内存不足导致网站报错。  
如果是第二种要不设置密码，要不将redis的绑定ip为本地ip

## **测试**

在manage.py所在目录下 输入python manage.py runserver 8085 如果看到下面信息，表示正确 。

Starting development server at http://127.0.0.1:8080/

Quit the server with CONTROL-C

此时还不能用浏览器访问服务器，还需安装fastdfs

## **fastdfs与nginx安装**

参考这个博客<https://blog.csdn.net/MissEel/article/details/80856194>

主要内容如下：

### **FastDFS安装**

一、安装fastdfs依赖包

1. 解压缩libfastcommon-master.zip

2. 进入到libfastcommon-master的目录中

3. 执行 ./make.sh

4. 执行 sudo ./make.sh install

二、安装fastdfs

1. 解压缩fastdfs-master.zip

2. 进入到 fastdfs-master目录中

3. 执行 ./make.sh

4. 执行 sudo ./make.sh install

三、配置跟踪服务器tracker

1. sudo cp /etc/fdfs/tracker.conf.sample /etc/fdfs/tracker.conf

2. 在/home/python/目录中创建目录 fastdfs/tracker

mkdir –p /home/python/fastdfs/tracker

3. 编辑/etc/fdfs/tracker.conf配置文件 sudo vim /etc/fdfs/tracker.conf

修改 base\_path=/home/python/fastdfs/tracker

四、配置存储服务器storage

1. sudo cp /etc/fdfs/storage.conf.sample /etc/fdfs/storage.conf

2. 在/home/python/fastdfs/ 目录中创建目录 storage

mkdir –p /home/python/fastdfs/storage

3. 编辑/etc/fdfs/storage.conf配置文件 sudo vim /etc/fdfs/storage.conf

修改内容：

base\_path=/home/python/fastdfs/storage

store\_path0=/home/python/fastdfs/storage

tracker\_server=自己本地的ip地址/服务器ip地址:22122

修改完成保存并退出 vim ，这时候我们可以使用

/usr/bin/fdfs\_trackerd /etc/fdfs/tracker.conf start

1. **启动tracker 和 storage**

**在自己虚拟机上启动**storage 和tracker

sudo service fdfs\_trackerd start

sudo service fdfs\_storaged start

腾讯云启动 storage 和tracker

/usr/bin/fdfs\_storaged /etc/fdfs/storage.conf start

/usr/bin/fdfs\_trackerd /etc/fdfs/tracker.conf start

五、测试是否安装成功

1. sudo cp /etc/fdfs/client.conf.sample /etc/fdfs/client.conf

2. 编辑/etc/fdfs/client.conf配置文件 sudo vim /etc/fdfs/client.conf

修改内容：

base\_path=/home/python/fastdfs/tracker

tracker\_server=自己ubuntu虚拟机的ip地址:22122

3. 上传文件测试：

fdfs\_upload\_file /etc/fdfs/client.conf 要上传的图片文件

如果返回类似group1/M00/00/00/rBIK6VcaP0aARXXvAAHrUgHEviQ394.jpg的文件id则说明文件上传成功

六、安装nginx及fastdfs-nginx-module web服务器 epoll

1. 解压缩 nginx-1.8.1.tar.gz

2. 解压缩 fastdfs-nginx-module-master.zip

3. 进入nginx-1.8.1目录中

4. 执行

sudo ./configure --prefix=/usr/local/nginx/ --add-module=fastdfs-nginx-module-master解压后的目录的绝对路径/src

ubuntu安装nginx时提示error: the HTTP rewrite module requires the PCRE library

需要安装pcre包。

sudo apt-get update

sudo apt-get install libpcre3 libpcre3-dev

*sudo* apt-get *install openssl* libssl-dev

sudo make

sudo make install

5. sudo cp fastdfs-nginx-module-master解压后的目录中src下的mod\_fastdfs.conf /etc/fdfs/mod\_fastdfs.conf

6. sudo vim /etc/fdfs/mod\_fastdfs.conf

修改内容：

connect\_timeout=10

tracker\_server=自己ubuntu虚拟机的ip地址:22122

url\_have\_group\_name=true

store\_path0=/home/ubuntu/fastdfs/storage

7. sudo cp 解压缩的fastdfs-master目录conf目录中的http.conf /etc/fdfs/http.conf

8. sudo cp 解压缩的fastdfs-master目录conf目录中的mime.types /etc/fdfs/mime.types

9.sudo vim /usr/local/nginx/conf/nginx.conf

在http部分中添加配置信息如下：

server {

listen 8888;

server\_name localhost;

location ~/group[0-9]/ {

ngx\_fastdfs\_module;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root html;

}

}

七、启动nginx

sudo /usr/local/nginx/sbin/nginx

## **服务器部署--uwsgi与nginx**

uwsgi.ini和uwsgi2.ini文件,一般改三个socket=127.0.0.1:8000,

第二uwsgi的端口要跟第一个不一样。用于与nginx进程间通信；  
目录chdir=/home/ubuntu/你的项目，一般我会放在家目录下，  
设置虚拟环境的路径virtualenv=/home/ubuntu/.virtualenvs/django\_py3   
nginx还是参考我附在项目目录下的nginx.conf，有几个要注意的地方：

1. server 80要放最前，其他server放在后面，  
   2.upstream里面就uwsgi对应两个socket， 正则匹配 / 就是所有url都有这个来处理，通过设置多个location来实现负载均衡。

具体参考下面

### 1.uwsgi的配置

项目部署时，需要把settings.py文件夹下的：

DEBUG=FALSE

ALLOWED\_HOSTS=[‘\*’]

[uwsgi]

#使用nginx连接时使用

#socket=127.0.0.1:8080

#直接做web服务器使用

http=127.0.0.1:8080

#项目目录

chdir=/Users/smart/Desktop/dj/bj17/dailyfresh

#项目中wsgi.py文件的目录，相对于项目目录

wsgi-file=dailyfresh/wsgi.py

processes=4

threads=2

master=True

pidfile=uwsgi.pid

daemonize=uwsgi.log

virtualenv=/Users/smart/.virtualenvs/dailyfresh

### 2.uwsgi的启动和停止

启动:uwsgi –-ini 配置文件路径 例如:uwsgi –-ini uwsgi.ini

停止:uwsgi --stop uwsgi.pid路径 例如:uwsgi –-stop uwsgi.pid

## 二、 nginx

### 1 nginx 配置转发请求给uwsgi

location / {

include uwsgi\_params;

uwsgi\_pass uwsgi服务器的ip:port;

}

### 2. nginx配置处理静态文件

django settings.py中配置收集静态文件路径:

STATIC\_ROOT=收集的静态文件路径 例如:/var/www/dailyfresh/static;

django 收集静态文件的命令:

python manage.py collectstatic

执行上面的命令会把项目中所使用的静态文件收集到STATIC\_ROOT指定的目录下。

收集完静态文件之后,让nginx提供静态文件，需要在nginx配置文件中增加如下配置:

location /static {

alias /var/www/dailyfresh/static/;

}

### 3 nginx转发请求给另外地址

在location 对应的配置项中增加 proxy\_pass 转发的服务器地址。

如当用户访问127.0.0.1时，在nginx中配置把这个请求转发给172.16.179.131:80(nginx)服务器，让这台服务器提供静态首页。

配置如下:

location = /{

proxy\_pass http://172.16.179.131;

}

4. [nginx配置upstream实现负载均衡](http://www.baidu.com/link?url=tRw5xlP0ktNnJtBzk3YU6DNaWgD6zFGKamwpc8k4qastl1_eDyHAJH4sj0FSRsvtsd7k_wrl8PX7S_YGd7XpY_" \t "_blank)

ngnix 配置负载均衡时，在server配置的前面增加upstream配置项。

upstream dailyfresh {

server 127.0.0.1:8080;

server 127.0.0.1:8081;

}

*# celery守护进程启动*

celery multi start w1 -A celery\_tasks.main -l info --logfile=./celerylog.log

*# 停止和重启 分别将 start 改为 stop / restart*