**安装uwsgi**

安装uwsgi

$ pip install uwsgi

**测试uwsgi**

在你的机器上写一个test.py

# test.py

def application(env, start\_response):

start\_response('200 OK', [('Content-Type','text/html')])

return "Hello World"

然后执行shell命令：

uwsgi --http :8000 --wsgi-file test.py

访问网页：

<http://127.0.0.1:8000/>

看在网页上是否有Hello World

**配置django**

请保证你的django项目是正常使用的。可以使用

[**Python**](http://lib.csdn.net/base/python) manage.py runserver 0.0.0.0:8000

来[**测试**](http://lib.csdn.net/base/softwaretest)一下你的django项目是否能正常跑起来。

请保证你的django程序已经关闭。

编写django\_wsgi.py文件，将其放在与文件manage.py同一个目录下。

注意： 编写文件时需要注意语句os.environ.setdefault。比如，如果你的项目为mysite，则你的语句应该是 os.environ.setdefault("DJANGO\_SETTINGS\_MODULE", "mysite.settings")

|  |  |
| --- | --- |
|  | *#!/usr/bin/env python*  *# coding: utf-8*  **import** **os**  **import** **sys**  *# 将系统的编码设置为UTF8*  reload(sys)  sys.setdefaultencoding('utf8')  os.environ.setdefault("DJANGO\_SETTINGS\_MODULE", "mysite.settings")  **from** **django.core.handlers.wsgi** **import** WSGIHandler  application = WSGIHandler() |

**连接django和uwsgi，实现简单的WEB服务器。**

我们假设你的Django项目的地址是/home/work/src/sites/testdjango1/testdjango/mysite，

然后，就可以执行以下命令：

uwsgi --http :8000 --chdir /home/work/src/sites/testdjango1/testdjango/mysite --module django\_wsgi

这样，你就可以在浏览器中访问你的Django程序了。所有的请求都是经过uwsgi传递给Django程序的。

当然，单单只有uWSGI是不够的，在实际的部署环境中，Nginx是必不可少的工具。

# Nginx + Django + Uwsgi

**环境介绍**

1. Centos7
2. django 1.8.18
3. nginx/1.10.2
4. uWSGI

**配置uWSGI**

之前我们是直接使用命令行来启动uWSGI，在实际部署环境中，我们常用的是配置文件的方式，而非命令行的方式。我的一般做法是用命令行来[**测试**](http://lib.csdn.net/base/softwaretest)是否uWSGI安装成功，然后用配置文件来真正部署。

另外，为了实现Nginx与uWSGI的连接，两者之间将采用soket来通讯方式。

在本节中，我们将使用uWSGI配置文件的方式来改进uWSGI的启动方式。

假定你的程序目录是 /home/work/src/sites/testdjango1/testdjango/mysite

我们将要让Nginx采用8088端口与uWSGI通讯，请确保此端口没有被其它程序采用。

新建一个soket.ini文件：将它放在 /home/work/src/sites/testdjango1/testdjango/mysite 目录下：

**[uwsgi]**

**socket = :8088**

**chdir = /root/projects/project1**

**module = django\_wsgi**

**processes = 4**

**daemonize = uwsgi.log**

在上面的配置中，我们使用 uwsgi.log 来记录日志，开启4个进程来处理请求。

这样，我们就配置好uWSGI了。

**配置Nginx**

我们假设你将会把Nginx程序日志放到你的目录/home/work/var/test/logs/下，请确保该目录存在。

我们假设你的Django的static目录是/home/work/src/sites/testdjango1/testdjango/collectedstatic/ ， media目录是/home/work/src/sites/testdjango1/testdjango/public/media/，请确保这些目录存在。

我们假设你的IP端口是 80

基于上面的假设，我们为conf/nginx.conf添加以下配置

**server** {

listen 80;

server\_name localhost;

access\_log /home/work/var/test/logs/access.log;

error\_log /home/work/var/test/logs/error.log;

#charset koi8-r;

#access\_log logs/host.access.log main;

location / {

include uwsgi\_params;

uwsgi\_pass 127.0.0.1:8088;

}

#error\_page **404** /**404.html**;

# **redirect** **server** **error** **pages** **to** **the** **static** **page** /**50x.html**

#

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

**location** = /**50x.html** {

root html;

}

**location** /**static**/ {

alias /home/work/src/sites/testdjango1/testdjango/collectedstatic/;

index index.html index.htm;

}

**location** /**media**/ {

alias /home/work/src/sites/testdjango1/testdjango/public/media/;

}

}

在上面的设置后，可以让Nginx来处理静态文件(/static/ 和 /media/ ）。非静态文件请求Nginx会发给 socket 8088，然后让uWSGI来进行处理。

**Nginx+uWSGI+Django的实现方式**

在完成上面配置后，需要按以下步骤来做：

1. 重启Nginx服务器，以使Nginx的配置生效。
2. nginx -s reload

重启后检查Nginx日志是否有异常。

1. 启动uWSGI服务器
2. cd /home/work/src/sites/testdjango1/testdjango/mysite
3. uwsgi -i socket.ini

检查日志 uwsgi.log 是否有异常发现。

1. 访问服务

基于上面的假设你的域名 localhost

因此，我们访问 localhost，如果发现程序与 单独使用Django启动的程序一模一样时，就说明成功啦！

1. 关闭服务的方法

将uWSGi进程杀死即可。

**一些建议**

1. uWSG配置文件的进程数，可以根据实际情况分配。不要开得太大，否则机器可能会内存耗用太高。一般来说，对于一个小社区来说，4个进程已经足够了。

## 注意：项目上传路径一般不要放在/root下，涉及到root权限问题，需要修改。

**参考、解释及其它**

* wsgi: WSGI是一种Web服务器网关接口。它是一个Web服务器（如nginx）与应用服务器（如uWSGI服务器）通信的一种规范。

关于WSGI协议看这里：[WSGI](http://wsgi.readthedocs.org/en/latest/)

* uWSGI: <http://uwsgi-docs.readthedocs.org/en/latest/index.html> uWSGI是一个Web服务器，它实现了WSGI协议、uwsgi、http等协议。 Nginx中HttpUwsgiModule的作用是与uWSGI服务器进行交换。
* uwsgi: uwsgi同WSGI一样是一种通信协议，而uWSGI是实现了uwsgi和WSGI两种协议的Web服务器。

uwsgi协议是一个uWSGI服务器自有的协议，它用于定义传输信息的类型（type of information），每一个uwsgi packet前4byte为传输信息类型描述，它与WSGI相比是两样东西。

关于uwsgi协议看这里：[The uwsgi protocol](http://projects.unbit.it/uwsgi/wiki/uwsgiProtocol)

* 有了uWSGI为什么还需要nginx？

nginx具备优秀的静态内容处理能力，然后将动态内容转发给uWSGI服务器，这样可以达到很好的客户端响应。