# CentOS 7 + Apache + mod\_wsgi 部署Django项目

字数1359 阅读498 评论1 喜欢13

网上有很多这种方式部署的博客，可是细致讲的并没有发现，而且很多方式都已经比较陈旧，所以自己用比较新的环境配置了一番，中间遇到的一些问题都有详细描述，希望有同学受益。

系统环境

* 系统：CentOS 7 x64
* Python版本：2.7.5（系统自带）
* Apache版本：2.4.6（yum安装）
* Django版本：1.9.7（源码安装）
* pip版本：8.1.2（官网get-pip.py方式安装）

安装使用Apache作为应用代理服务器

* 安装Apache

# yum install httpd

* 设置Apache服务开机自启动（可选，不过设置自启动方便）

# /sbin/chkconfig httpd on

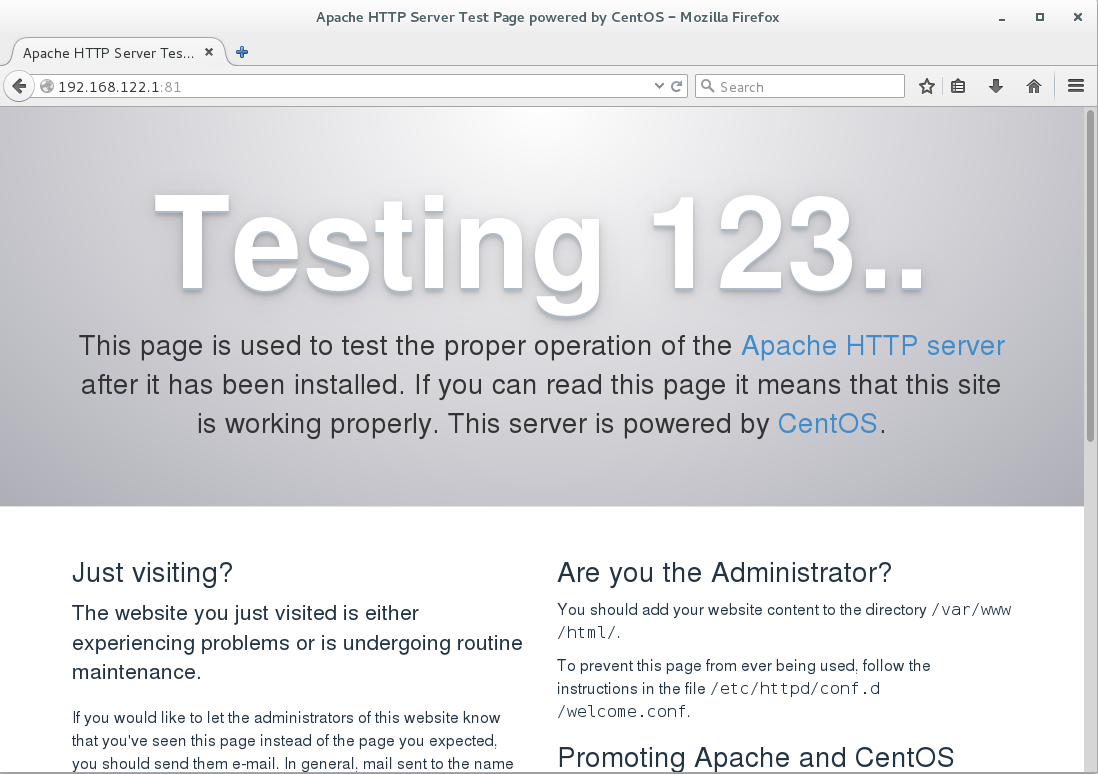
* 启动Apache服务

# /sbin/service httpd start

* 验证安装

访问本机的80端口：http://localhost

出现以下界面说明安装成功(apache端口被我更改过成81)



访问Apache服务

* Apache服务的配置说明

主配置文件是/etc/httpd/conf/httpd.conf  
其他配置文件存储在/etc/httpd/conf.d/目录

主配置文件内容为：

#服务器根目录

ServerRoot "/etc/httpd"

#端口#Listen 12.34.56.78:80

Listen 80

#域名+端口来标识服务器，没有域名用ip也可以#ServerName www.example.com:80

#不许访问根目录

<Directory />

AllowOverride none

Require all denied

</Directory>

# 文档目录

DocumentRoot "/var/www/html"

# 对 /var/www目录访问限制

<Directory "/var/www">

AllowOverride None

# Allow open access:

Require all granted

</Directory>

# 对/var/www/html目录访问限制

<Directory "/var/www/html">

　　 Options Indexes FollowSymLinks

　　 AllowOverride None

　　Require all granted

</Directory>

# 默认编码

AddDefaultCharset UTF-8

#EnableMMAP off

EnableSendfile on

# include进来其它配置文件

IncludeOptional conf.d/\*.conf

* 配置需要注意的点

Apache2.2.x版本和Apache2.4.x版本对于Directory标签中的配置方式并不一样，如果配置方式错误  
会导致 client denied by server configuration错误

Apache2.2.x的配置文件（IP地址以192.168.10.81为例）

<VirtualHost 192.168.10.81:80>

ServerAdmin rocdk890@gmail.com

directoryIndex index.html index.php index.htm index.shtml login.php

ServerName 192.168.10.81

DocumentRoot /var/www/vhosts/wwwroot

<Directory "/var/www/vhosts/wwwroot">

Options -Indexes

AllowOverride All

Order allow,deny

Allow from all

</Directory>

</VirtualHost>

Apache2.4.x的配置文件

<VirtualHost 192.168.10.81:80>

ServerAdmin rocdk890@gmail.com

directoryIndex index.html index.php index.htm index.shtml login.php

ServerName 192.168.10.81

DocumentRoot /var/www/vhosts/wwwroot

<Directory "/var/www/vhosts/wwwroot">

Options -Indexes

AllowOverride All

Require all granted

</Directory>

</VirtualHost>

变动地方主要有：

删除了 Order deny,allow 和 Order allow,deny

把 Deny from all 替换成了 Require all denied

把Allow from all 替换成了 Require all granted

然后还把 Allow from 192.168.10.21 这样的语句给替换成了 Require host 192.168.10.21

使用mod\_wsgi拓展作为Django的应用容器

* 首先安装httpd-devel

# yum install -y httpd-devel

* 安装mod\_\_wsgi

yum install mod\_wsgi-httpd

# yum install mod\_wsgi

安装完成之后，mod\_wsgi.so会在Apache的modules目录中

* 在httpd.conf文件中添加以下内容

LoadModule wsgi\_module modules/mod\_wsgi.so

该配置用来连接django.wsgi，使工程被Apache加载

项目根目录中新建wsgi文件夹，在wsgi文件夹中新建文件 django.wsgi

* django.wsgi文件内容如下：

import os

import sys

from django.core.wsgi import get\_wsgi\_application

# Add this file path to sys.path in order to import settings

sys.path.insert(0, os.path.join(os.path.dirname(os.path.realpath(\_\_file\_\_)), '..'))

os.environ['DJANGO\_SETTINGS\_MODULE'] = 'projectname.settings'

sys.stdout = sys.stderr

DEBUG = True

application = get\_wsgi\_application()

注意：对于1.3和1.4及之后的版本，此文件的代码内容是不同的，详情区别参考 [http://django.readthedocs.io/en/latest/releases/1.7.html#standalone-scripts](http://django.readthedocs.io/en/latest/releases/1.7.html" \l "standalone-scripts" \t "/home/yangmaoqiang/文档\\x/_blank)

配置wsgi时，

必须配置项目路径到系统路径中,因为要通过项目路径找到settings.py配置文件。也就是sys.path.insert(0, os.path.join(os.path......巴拉巴拉，DJANGO\_SETTINGS\_MODULE必须指向项目的settings.py文件，projectname指的就是项目的名称。  
修改了wsgi的配置后必须重启httpd服务。

在Apache中配置wsgi

* 在httpd.conf中添加一行

WSGIScriptAlias /python "/path/to/docRoot/django.wsgi"

如：WSGIScriptAlias /python /usr/local/testProject/wsgi/django.wsgi

设置项目目录访问权限

在httpd.conf中添加目录权限，否则会报错：Apache: access denied because search permissions are missing

<Directory "/usr/local/testProject">

AllowOverride All

Require all granted

</Directory>

安装Django

# wget https://www.djangoproject.com/m/releases/1.9/Django-1.9.7.tar.gz# cd Django-1.9.7/# python setup.py install

* 验证Django安装是否完成

# python>>> import django>>> django.VERSION

(1,9,7,'final',0)

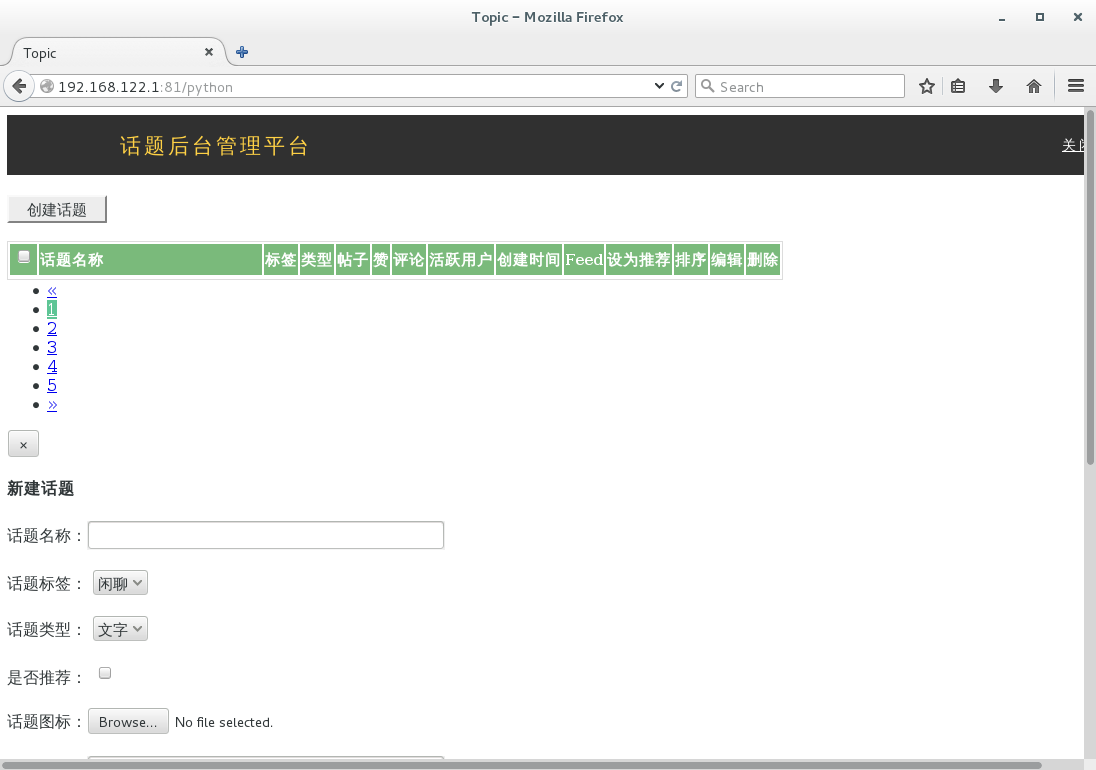
安装pip

# wget https://bootstrap.pypa.io/get-pip.py# python get-pip.py

安装MySQLdb模块（常用模块安装）

# yum install python-devel# yum install mysql-devel# pip install MySQL-python

最后访问项目的效果（之前做的一个小项目哈，表介意界面)



部署完成之后访问项目

下一步改进方法

使用Docker技术，把Django部署环境制作成镜像，然后在生产服务器上安装Docker环境，项目的更新直接更新镜像即可，了解Docker技术到[https://www.gitbook.com/book/yeasy/docker\_practice/details](https://www.gitbook.com/book/yeasy/docker_practice/details" \t "/home/yangmaoqiang/文档\\x/_blank)，生产环境使用镜像的部署方式，就可以避免环境重复部署的问题，而且很多服务器使用的CentOS版本还是6.x，然后自带的python版本较低，升级python版本又比较麻烦，Docker在生产环境已经在广泛应用了。

### apache 找不到django的第三方包

在setting.py里设置

**sys.path.append('/usr/local/python2.7/lib/python2.7/site-packages')，**

/usr/local/python2.7/lib/python2.7/site-packages为第三方包的安装路径

查找第三方包的安装路径：pip show --files simplejson

# Django.wsgi的内容：

"""

WSGI config for HMWeb project.

This module contains the WSGI application used by Django's development server

and any production WSGI deployments. It should expose a module-level variable

named ``application``. Django's ``runserver`` and ``runfcgi`` commands discover

this application via the ``WSGI\_APPLICATION`` setting.

Usually you will have the standard Django WSGI application here, but it also

might make sense to replace the whole Django WSGI application with a custom one

that later delegates to the Django one. For example, you could introduce WSGI

middleware here, or combine a Django application with an application of another

framework.

"""

import os

import sys

sys.path = sys.path + ["/home/hmamis/www/ams/"]

os.environ['DJANGO\_SETTINGS\_MODULE'] = 'ams.settings'

#os.environ['PYTHON\_EGG\_CACHE'] = '/tmp/.python-eggs'

##current\_dir = os.path.dirname(\_\_file\_\_)

current\_dir = '/home/hmamis/www/ams/wsgi/'

sys.stdout = sys.stderr

DEBUG = True

if current\_dir not in sys.path:

sys.path.append(current\_dir)

import django.core.handlers.wsgi

application = django.core.handlers.wsgi.WSGIHandler()

#

## Apply WSGI middleware here.

# from helloworld.wsgi import HelloWorldApplication

# application = HelloWorldApplication(application)