

[原创](#)

# 使用Gunicorn Ngnx Supervisor部署Django项目

[蓝色\\_风暴](#)[关注](#)

2018-05-24 09:40:02 393人阅读 0人评论

Django作为最受欢迎基于Python的Web框架之一，为用户提供了一个轻量级的测试Server，但这个Server最好不要用于生产环境。那么如果我们需要在生产环境上面部署Django项目，那么我们使用什么环境呢？最早Django推荐使用Apache+mod\_wsgi，但现在已经Django的部署方法已对有很多，越来越好，也相对更复杂了，本文将介绍其中一种Django在生产环境的部署方案：Gunicorn+Nginx+Supervisor

实验环境：

- 操作系统：CentOS 7
- Python版本：3.4.8
- Django版本：2.0.5

准备实验环境：

因为CentOS 7默认的Python版本为2.7.5，所以我们需要再安装一个3.4.8的版本。另外我们使用的Django版本为2.0.5，Django 2.x版本只支持Python 3.x的版本，所以我们第一步先安装3.x版本的Python

```
yum install python34 python34-pip
```

接下来再安装Django

```
# 此命令表示使用3.4版本的Python来安装Django，如果直接使用pip install django，默认是使用2.7.5的Python3.4 -m pip install django
```

安装Gunicorn

```
python3.4 -m pip install gunicorn
```

安装Nginx

```
yum install nginx
```

安装Supervisor

```
yum install supervisor
```

创建Django项目：

```
# 注意替换ProjectName为自己的项目名，AppName为自己的App名
django-admin startproject ProjectName
cd ProjectName
django-admin startapp AppName
```

使用Gunicorn测试启动Django项目：



0

[分享](#)[蓝色\\_风暴](#)[关注](#)

```
gunicorn ProJectName.wsgi
```

启动成功后输入大概如下

```
[2018-05-24 09:50:33 +0800] [22849] [INFO] Starting gunicorn 19.8.1
[2018-05-24 09:50:33 +0800] [22849] [INFO] Listening at: http://127.0.0.1:8000 (22849)
[2018-05-24 09:50:33 +0800] [22849] [INFO] Using worker: sync
[2018-05-24 09:50:33 +0800] [22852] [INFO] Booting worker with pid: 22852
```

从以上信息我们可以看出项目已经启动成功，gunicorn版本为19.8.1，监听127.0.0.1:8000端口并且工作在22849号进程，启动的worker方式为sync，并启动了一个worker进程，进程号为22852。更多关于workder相关的资料自行查略更多的信息

由上面我们监听在127.0.0.1，我们客户端无法访问，所以我们需要指定监听端口启动，把上面的直接Ctrl+c结束，然后执行下面的命令启动

```
# 使用-b指定监听的地址和端口
gunicorn Gogenius.wsgi -b 0.0.0.0:8000
```

配置Supervisor:

Supervisor主要是用来管理进程的工具，默认安装好了以后有一个配置文件/etc/supervisord.conf，这是主配置文件。还有一个目录/etc/supervisord.d，这个目录一般用来存放我们管理的进程，每个进程一个配置文件，以.ini结尾

创建一个管理Gunicorn的配置文件

```
touch /etc/supervisord.d/gunicorn.ini
```

gunicorn.conf的内容为

```
[program:gunicorn]
directory= /usr/local/ProJectName
command = /usr/bin/gunicorn ProJectName.wsgi -b 127.0.0.1:8000
user = root
autostart= true
autorestart= true
redirect_stderr = true
stdout_logfile = /var/log/gunicorn.log
```

说明:

- [program:gunicorn]: 其中gunicorn为要管理的子进程名称，此名称可以自定义，但最好不要乱写，与子进程有点关系为好
- directory: Django项目的根目
- command: 要执行的命令，这里为启动子进程的命令，子进程为gunicorn
- user: 指定启动子进程的用户
- autostart: 自动启动，也就是当父亲进程启动的时候，子进程也跟随启动。supervisor为父进程
- autorestart: 自动重启，也就是当子进程挂掉的时候，父进程将会尝试自动去重启了进程
- redirect\_stderr: 当此选项为true的时候，错误日志也会写进stdout\_logfile中
- stdout\_logfile: 定义stdout\_logfile路径

关于更多资料可以参考[此链接](#)

接下来就是启动Supervisor

```
systemctl start supervisord.service
systemctl enable supervisord.service
```

如果更改过子进程的配置文件，这时候我们只需要reload下我们的Supervisor服务即可，不需要重启整个服务，这样会影响其它的服务



配置Nginx转发：

我们知道我们Django本身就可以作用一个轻量级的Web Server，Django里面为我们定义了我们html js img css等页面的存放位置，并且为自动的去找到需要的页面。但现在我们没有使用Django本身的Web Server，这时候就需要通过我们Nginx配置代理来让客户端请求访问到对应的页面，Nginx配置如下：根据自己需求

```
server {  
  
    listen 5000;  
  
    location / {  
        # proxy_pass指定地址为访问gunicorn地址和端口  
        proxy_pass http://127.0.0.1:8000;  
        proxy_next_upstream http_500 http_502 http_503 error timeout invalid_header;  
        proxy_set_header Host $host;  
        proxy_set_header X-Forwarded-For $remote_addr;  
    }  
  
    location ~.*\.(html|css|js|jpg|json|png|map|\tttf*|\woff2*|\woff*|eot|otf|ttf|json|cur|woff|svg|woff2|  
        # root指定Django项目的根目录  
        root /usr/local/Gogenius;  
        proxy_next_upstream http_500 http_502 http_503 error timeout invalid_header;  
        proxy_set_header Host $host;  
        proxy_set_header X-Forwarded-For $remote_addr;  
    }  
}
```

启动Nginx

```
systemctl start nginx  
systemctl enable nginx
```

至此，我们就可以通过Nginx的5000端口访问我们的Django项目了

©著作权归作者所有：来自51CTO博客作者蓝色\_风暴的原创作品，如需转载，请注明出处，否则将追究法律责任

django

gunicorn

nginx

Python


0

收藏

分享

上一篇：Redis 3.0原生集群部署

下一篇：jQuery判断radio属性为...



蓝色\_风暴

60篇文章，16W+人气，6粉丝

关注




提问和评论都可以，用心的回复会被更多人看到和认可



0

分享



蓝色\_风暴

关注