在 Ubuntu 上使用 uWSGI 和 Nginx 部署 Flask 项目

关于 uWSGI ，可以先看这篇文章。简单来说，**WSGI** 是一个 Python 协议，定义了**应用程序**（我们写的软件）如何与 **Web 服务器**（如 Nginx）通信，**WSGI** 只是一个接口。而 **uWSGI** 是一个支持多种语言的服务器容器，使用 **WSGI** 定义的标准实现与多种 **Web 服务器**的通信，并将 Web 服务器发来的请求"翻译"成**应用程序**所能理解形式。

安装

Python 2：

```
sudo apt-get update
sudo apt-get install python-pip python-dev nginx
```

Python 3：

```
sudo apt-get update
sudo apt-get install python3-pip python3-dev nginx
```

安装 Flask 和 uwsgi:

```
pip install uwsgi flask
```

创建一个简单的 Flask 项目 `~/myproject/myproject.py` ：

```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "<h1 style='color:blue'>Hello There!</h1>"

if __name__ == "__main__":
    app.run(host='0.0.0.0')
```

创建启动文件为 `~/myproject/run.py` ：

```
from myproject import app

if __name__ == "__main__":
    app.run()
```

运行 `python run.py` ，然后访问 `http://server_domain_or_IP:5000` 将会看到：

当然直接使用 `python run.py` 的方法只适合本地开发，线上的话速度太慢，我们需要使用 `uwsgi` 。

uwsgi

首先确保你安装了 `uwsgi` ，然后运行：

```
uwsgi --socket 0.0.0.0:5000 --protocol=http -w run:app
```

`protocol` 说明使用 `http` 协议， `-w` 指明了要启动的模块， `run` 就是项目启动文件 `run.py` 去掉扩展名， `app` 是 `run.py` 文件中的变量 `app` ，即 Falsk 实例。然后访问 `http://server_domain_or_IP:5000` ，同样会看到上图。说明 `uwsgi` 可以正常运行。

但是这样的话每次都从命令行启动太麻烦，可以在 `~/myproject/` 目录下创建一个配置文件 `myproject.ini` ：

```
[uwsgi]
module = run:app
master = true
processes = 3

chdir = /home/ubuntu/myproject
socket = /path/to/sock/myproject.sock
logto = /home/to/log/ishuhui.log
chmod-socket = 660
vacuum = true
```

- `processes = 5` 说明要启动5个子进程处理请求； `module = run:app` 和命令行使用的意义一样；
- `chdir = /home/ubuntu/myproject` 只想我们项目的根目录，即 `run.py` 所在文件夹；
- `socket = /path/to/sock/myproject.sock` 是 `uwsgi` 启动后所需要创建的文件，这个文件用来和 Nginx 通信，后面会在配置 Nginx 时用到，所以 `chmod-socket = 660` 是为了修改 `.sock` 文件权限来和 Nginx 通信；
- `logto = /home/to/log/ishuhui.log` 指明了 `uwsgi` 日志目录， `uwsgi` 会将请求历史写入该文件。

配置完成后运行：

```
uwsgi --ini myproject.ini
```

可以看到 `/path/to/sock/myproject.sock` 目录下多了 `myproject.sock` 文件，用来和 Nginx 通信。接下来配置 Nginx。

配置 Nginx

配置 Nginx 特别简单，找到 Nginx 配置文件（ `sudo vim /etc/nginx/sites-available/default` ），修改为如下格式：

```
server {
    listen 80;
    server_name server_domain_or_IP;

    location / {
        include uwsgi_params;
        uwsgi_pass unix:/path/to/sock/myproject.sock;
    }
}
```

其实重要的就两行：

```
include uwsgi_params;
uwsgi_pass unix:/path/to/sock/myproject.sock;
```

接下来不出意外，访问 80 端口，就可以看到你的程序了。

将 `uwsgi` 设置为系统服务

我们运行 `uwsgi --ini myproject.ini` 之后，按 `ctrl+c` 或者关闭 ssh 连接窗口，都会导致 `uwsgi` 进程关闭。 `uwsgi` 进程一关闭， `.sock` 文件就会消失，这时访问网站 Nginx 就会报错：

502 Bad Gateway

---

nginx/1.10.3 (Ubuntu)

这时，我们需要进程管理软件管理 `uwsgi` 进程的运行了。Ubuntu 自带的 systemd 是最简单的方法，可以将我们的项目变为系统服务。首先创建 `myproject.service` 文件 `sudo vim /etc/systemd/system/myproject.service` ：

```
[Unit]
Description=uWSGI instance to serve myproject
After=network.target

[Service]
User=lufficc
Group=www-data
WorkingDirectory=/home/ubuntu/myproject
Environment=FLASKR_SETTINGS=/home/ubuntu/myproject/env.cfg
ExecStart=/usr/local/bin/uwsgi --ini /home/ubuntu/myprojectenv/ishuhui.ini

[Install]
WantedBy=multi-user.target
```

- `WorkingDirectory` ： 你的项目目录。
- `Environment` ：需要的环境变量，比如指明你的项目的配置文件
- `ExecStart` ：服务启动的代码
- `WantedBy=multi-user.target` ：指明会跟随系统启动而启动该服务。

注意以上所有路径为 `绝对路径` **。**

接下来可以愉快的启动了(myproject 就是 `myproject.service` 文件名去掉扩展名)：

目录

```
sudo systemctl start myproject
sudo systemctl restart myproject
sudo systemctl stop myproject
```

参考：

1. https://www.digitalocean.com/community/tutorials/how-to-serve-flask-applications-with-uwsgi-and-nginx-on-ubuntu-16-04
2. https://www.digitalocean.com/community/tutorials/how-to-set-up-uwsgi-and-nginx-to-serve-python-apps-on-ubuntu-14-04

⬆

创建日期: 2017年08月27日

修改日期: 2017年08月28日

文章分类: Nginx

文章标签: Nginx Flask Linux

评论个数: 8

阅读次数: 6547

推荐阅读 :

你真的了解如何将 Nginx 配置为Web服务... 2016-11-27

一步一步教你部署自己的Laravel应用程... 2016-09-23

初识 Nginx 2016-11-23

Laragh #878
2017-08-28 00:39

不会吧 我是第一个啊。。。刚进你博客刚更新啊

回复

lufficc 博主 #879
2017-08-28 01:00

@Laragh 好快，你是第一个(●'◡'●)

回复

大佬 #1077
2017-12-12 20:49

66666666666666666666666

回复

lixinlong #1127
2018-01-19 11:28

如果是用virtualenv的话myproject.ini里面需要多一个home配置到你的虚拟环境
同时myproject.service里面的/usr/local/bin/uwsgi应该指向你的虚拟python库中的uwsgi

回复

leo #1231
2018-03-23 16:36

我跑到uwsgi --ini myproject.ini 就卡住了，會一直卡在那行指令

回复

lufficc 博主 #1233
2018-03-23 19:17

@leo 难能访问吗？有可能是服务器配置低？

回复

cen1234 #1400
2018-06-06 16:45

推荐使用supervisor管理uwsgi进程

回复

lufficc 博主 #1401
2018-06-06 18:40

@cen1234 这个用起来的确稳定方便

回复

姓名*

您的大名

邮箱*

邮箱不会公开

个人网站

可选，填写后点击头像可以直接进入

评论内容,支持 Markdown*

审核后才会显示

回复