← Python3 环境搭建 Python3 filter() 函数 →

Python uWSGI 安装配置

本文主要介绍如何部署简单的 WSGI 应用和常见的 Web 框架。

以 Ubuntu/Debian 为例，先安装依赖包：

```
apt-get install build-essential python-dev
```

Python 安装 uWSGI

1、通过 pip 命令：

```
pip install uwsgi
```

2、下载安装脚本：

```
curl http://uwsgi.it/install | bash -s default /tmp/uwsgi
```

将 uWSGI 二进制安装到 /tmp/uwsgi ，你可以修改它。

3、源代码安装：

```
wget http://projects.unbit.it/downloads/uwsgi-latest.tar.gz
tar zxvf uwsgi-latest.tar.gz
cd uwsgi-latest
make
```

安装完成后，在当前目录下，你会获得一个 uwsgi 二进制文件。

第一个 WSGI 应用

让我们从一个简单的 "Hello World" 开始，创建文件 foobar.py，代码如下：

```
def application(env, start_response):
    start_response('200 OK', [('Content-Type','text/html')])
    return [b"Hello World"]
```

uWSGI Python 加载器将会搜索的默认函数 `application` 。

接下来我们启动 uWSGI 来运行一个 HTTP 服务器，将程序部署在HTTP端口 9090 上：

```
uwsgi --http :9090 --wsgi-file foobar.py
```

添加并发和监控

默认情况下，uWSGI 启动一个单一的进程和一个单一的线程。

你可以用 `--processes` 选项添加更多的进程，或者使用 `--threads` 选项添加更多的线程 ，也可以两者同时使用。

```
uwsgi --http :9090 --wsgi-file foobar.py --master --processes 4 --threads 2
```

以上命令将会生成 4 个进程, 每个进程有 2 个线程。

如果你要执行监控任务，可以使用 stats 子系统，监控的数据格式是 JSON：

```
uwsgi --http :9090 --wsgi-file foobar.py --master --processes 4 --threads 2 --stats 127.0.0.1:9191
```

我们可以安装 uwsgitop（类似 Linux top 命令） 来查看监控数据：

```
pip install uwsgitop
```

结合 Web 服务器使用

我们可以将 uWSGI 和 Nginx Web 服务器结合使用，实现更高的并发性能。

一个常用的nginx配置如下：

```
location / {
    include uwsgi_params;
    uwsgi_pass 127.0.0.1:3031;
}
```

以上代码表示使用 nginx 接收的 Web 请求传递给端口为 3031 的 uWSGI 服务来处理。

现在，我们可以生成 uWSGI 来本地使用 uwsgi 协议：

```
uwsgi --socket 127.0.0.1:3031 --wsgi-file foobar.py --master --processes 4 --threads 2 --stats 127.0.0.1
:9191
```

如果你的 Web 服务器使用 HTTP，那么你必须告诉 uWSGI 本地使用 http 协议 (这与会自己生成一个代理的–http不同):

```
uwsgi --http-socket 127.0.0.1:3031 --wsgi-file foobar.py --master --processes 4 --threads 2 --stats 127.
0.0.1:9191
```

部署 Django

Django 是最常使用的 Python web 框架，假设 Django 项目位于 /home/foobar/myproject:

uwsgi --socket 127.0.0.1:3031 --chdir /home/foobar/myproject/ --wsgi-file myproject/wsgi.py --master --processes 4 --threads 2 --stats 127.0.0.1:9191

`--chdir` 用于指定项目路径。

我们可以把以上的命令弄成一个 yourfile.ini 配置文件:

```
[uwsgi]
socket = 127.0.0.1:3031
chdir = /home/foobar/myproject/
wsgi-file = myproject/wsgi.py
processes = 4
threads = 2
stats = 127.0.0.1:9191
```

接下来你只需要执行以下命令即可：

```
uwsgi yourfile.ini
```

部署 Flask

Flask 是一个流行的 Python web 框架。

创建文件 myflaskapp.py ，代码如下：

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def index():
    return "<span style='color:red'>I am app 1</span>"
```

执行以下命令：

```
uwsgi --socket 127.0.0.1:3031 --wsgi-file myflaskapp.py --callable app --processes 4 --threads 2 --stats
 127.0.0.1:9191
```

← Python3 环境搭建 Python3 filter() 函数 →