想提高爬虫效率？aiohttp 了解下

极客猴

题图：by cfunk44 from Instagram

阅读文本大概需要 8 分钟。

对于爬虫程序，我们往往会很关注其爬虫效率。影响爬虫效率有几个因素有，是否使用多线程，I/O 操作，是否同步执行等。其中 I/O 操作、同步执行是最影响爬虫效率的。

众所周知，Requests 库一个优秀的 HTTP 库，通过它可以非常简单地发起 HTTP 请求。不过，这个库所执行的网络请求都是同步。当爬虫程序进程获得 CPU 的时间片时，如果程序在进行 I/O 操作（例下载图片），在这段 IO 执行的时间里，CPU 处于空闲中，这样会造成 CPU 的计算能力就被浪费了。

如果 CPU 能将等待时间利用起来，那么爬虫效率就提高了。那就需要对程序进行改造，将 I/O 同步操作变成异步操作。本文内容是介绍一个强大的异步 I/O 操作的库 —— aiohttp。

1 aiohttp 介绍

说到 aiohttp ，不得不说下 asyncio 。asyncio 是 Python 3.4 版本引入的标准库。它工作模式是单线程并发，使用协同执行 I/O 操作。asyncio 的编程模型就是一个消息循环。我们从 asyncio 模块中直接获取一个 EventLoop 的引用，然后把需要执行的协程扔到 EventLoop 中执行，就实现了异步 IO。

使用 asyncio 实现一个异步函数 hello() 的例子:

```
import asyncio

@asyncio.coroutine # 修饰符，等同于 asyncio.coroutine(hello())

def hello():

        print("Hello world!")

        # 异步调用asyncio.sleep(1):

        r = yield from asyncio.sleep(1)

        print("Hello again!")

# 获取EventLoop:

loop = asyncio.get_event_loop()

# 执行coroutine

loop.run_until_complete(hello())

loop.close()
```

而 aiohttp 则是基于 asyncio 实现的 HTTP 框架。 aiohttp 全称是 Async http client/server framework。翻译成中文是异步 HTTP 的客户端/服务器框架。从名字中，我们可知 aiohttp 是分为服务器端和客户端，专门异步处理 HTTP 的请求。

2 aiohttp 安装

安装 aiohttp 可以通过 pip 方式安装，在终端中执行安装命令即可。

```
pip install aiohttp
```

3 async/await 语法

前面我们讲到异步 I/O 的用法，但是声明异步函数比较繁琐，还需要依赖 yield 语法。在 Python 3.5 中，引入了 async/await 关键字，使得异步回调的写法更加直观和人性化。

在函数 def 之前增加关键字async,表示这个函数是异步函数。相当于替代语法@asyncio.coroutine。具体例子例如：

```
async def hello():
    print("Hello World!")
```

另外使用 await 替换了 yield from, 表示这部分操作为异步操作。

```
async def hello():
    print("Hello World!")
    r = await asyncio.sleep(1)
    print("Hello again!")
```

最后执行异步函数，还是需要用到 EventLoop 引用，然后利用协程执行异步函数。最终的代码如下：

```
import asyncio
async def hello():
    print("Hello world!")
    r = await asyncio.sleep(1)
    print("Hello again!")
if __name__ == '__main__':
    loop = asyncio.get_event_loop()
    tasks = [hello(), ]
    loop.run_until_complete(asyncio.wait(tasks))
    loop.close()
```

运行结果如下：

```
Hello world!
>> 会暂停一秒钟
Hello again!
```

4 aiohttp 基本用法

我们使用 aiohttp 以 GET 方式向httpbin.org网站发起一个 HTTP 请求。因为是 aiohttp 是异步处理 HTTP 请求。所以还必须遵循 Python 的异步函数语法，即需使用 async/await 语法。

使用 aiohttp 发起一个 HTTP 请求，具体编写可以分为以下几步：

1）使用 async 定义异步函数

2）通过 aiohttp.ClientSession 获取一个 session 对象

3）用该 session 对象以 GET、POST、PUT 等方式去请求网页

4）最后获取 EventLoop 引用，执行异步函数。

```
import asyncio
import aiohttp
# 定义异步函数 main()
async def main():
    # 获取 session 对象
        async with aiohttp.ClientSession() as session:
            # get 方式请求 httbin  async with session.get('http://httpbin.org/get') as response:
                print(response.status)
                print(await response.text())
loop = asyncio.get_event_loop()
loop.run_until_complete(main())
```

aiohttp 支持自定义 headers、设置超时时间、设置代理、自定义 cookie 等。

```
import asyncio
import aiohttp
url = 'http://httpbin.org/post'
headers = {
    'User-
agent': "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2228.0 Safari/537.36",
}
data = {
```

```
        'data': 'person data',
}
# 定义异步函数 main()
async def main():
        # 获取 session 对象
        async with aiohttp.ClientSession() as session:
                # post 方式请求 httbin
                async with session.post(url=url, headers=headers, data=data) as response:
                        print(response.status)
                        print(await response.text())
loop = asyncio.get_event_loop()
loop.run_until_complete(main())
```

关于 aiohttp 更多用法，可以执行阅读官网文档。说句实话，aiohttp 跟 Requests 的用法大同小异。如果你已经学会了 Requests 库，很快就能掌握 aiohttp 的用法。

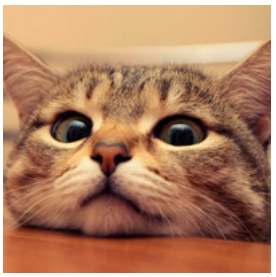
推荐阅读：

我的增长黑客之旅

用Python告诉你深圳房租有多高

人必有痴，而后有成

文章转载自公众号



极客猴

极客猴