第 12 关、协程实践

1、项目：薄荷网爬取食物热量

1-1、明确目标

（1）目标网站：http://www.boohee.com/food/；

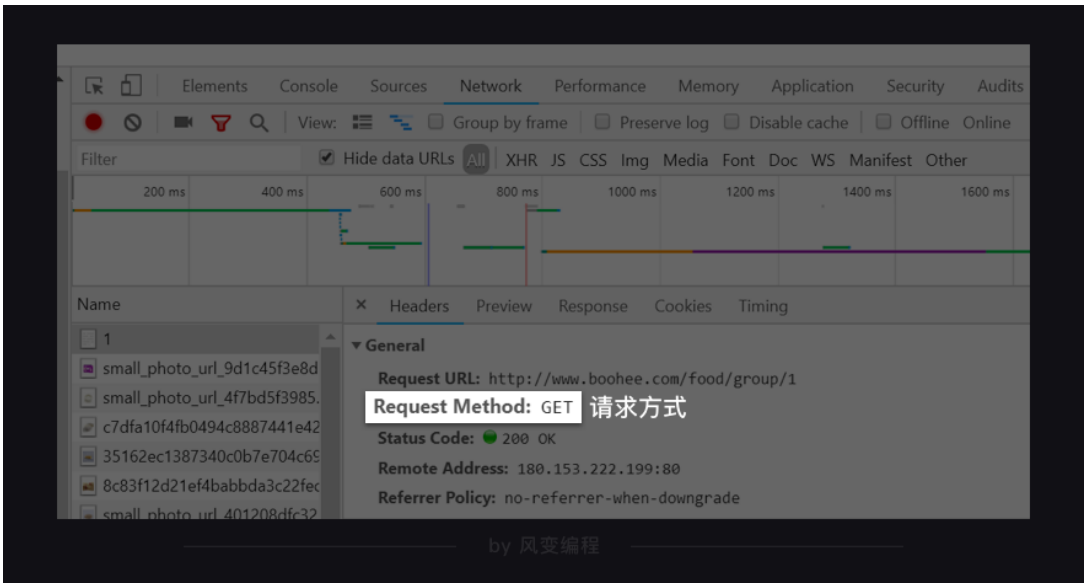
（2）网站协议：http://www.boohee.com/food/robots.txt（目标网站 + robots.txt 可查看目标网站的页面爬取许可）；

（3）项目目标：用多协程爬取薄荷网的食物热量信息（包含食物名、热量、食物详情页面链接）。

1-2、过程分析

（1）确定数据所在页面

- 右击打开“检查”工具，并点击 Network，然后刷新页面。点击第 0 个请求 1，看 Response 能在 Response 里找到食物的信息，说明我们想要的数据存在 HTML 里；
- 再看请求 1 的 Headers，可以发现薄荷网的网页请求方式是 get （即通过 requests.get() 获取数据）。



（2）确定数据所在位置

①、每个常见食物分类的网址

- 网址的group参数代表着常见食物分类，后面的数字代表着这是第几个类；

- 除常见食物分类【菜肴】的网址与其他不同，前10个常见食物分类的网址都是：http://www.boohee.com/food/group/+数字

②、每一页食物的网址

- 只要改变网址 http://www.boohee.com/food/group/1?page= 中 page 后面的数字，就能实现翻页

（3）数据获取思路

用 find_all/find 就能提取出 <li class="item clearfix">... 标签下的食物详情链接、名称和热量；

1-3、代码实现

1-3-1、导入库和模块

```
#导入所需的库和模块：

from gevent import monkey
monkey.patch_all()
#让程序变成异步模式。
import gevent,requests, bs4, csv
from gevent.queue import Queue
```

1-3-2、创建队列存储食物信息

```
work = Queue()
#创建队列对象，并赋值给work。

#前3个常见食物分类的前3页的食物记录的网址：
url_1 = 'http://www.boohee.com/food/group/{type}?page={page}'
for x in range(1, 4):
    for y in range(1, 4):
```

```
        real_url = url_1.format(type=x, page=y)
        work.put_nowait(real_url)
#通过两个for循环，能设置分类的数字和页数的数字。
#然后，把构造好的网址用put_nowait方法添加进队列里。

#第11个常见食物分类的前3页的食物记录的网址：
url_2 = 'http://www.boohee.com/food/view_menu?page={page}'
for x in range(1,4):
    real_url = url_2.format(page=x)
    work.put_nowait(real_url)
#通过for循环，能设置第11个常见食物分类的食物的页数。
#然后，把构造好的网址用put_nowait方法添加进队列里。

print(work)
#打印队列
```

- 第 7 行用 Queue() 创建了空的队列；
- 第 12 – 23 行通过两个 for 循环，构造了前 3 个常见食物分类的前 3 页的食物记录的网址（其中由于第 11 个常见食物分类的网址比较特殊，要分开构造。然后把构造好的网址用 put_nowait 方法，都放进队列里。）；

1-3-3、多协程爬取数据

```
def crawler():
#定义crawler函数
    headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
    }
    #添加请求头
    while not work.empty():
    #当队列不是空的时候，就执行下面的程序。
        url = work.get_nowait()
        #用get_nowait()方法从队列里把刚刚放入的网址提取出来。
        res = requests.get(url, headers=headers)
        #用requests.get获取网页源代码。
        bs_res = bs4.BeautifulSoup(res.text, 'html.parser')
        #用BeautifulSoup解析网页源代码。
        foods = bs_res.find_all('li', class_='item clearfix')
        #用find_all提取出<li class="item clearfix">标签的内容。
        for food in foods:
        #遍历foods
            food_name = food.find_all('a')[1]['title']
            #用find_all在<li class="item clearfix">标签下，提取出第2个<a>元素title属性的值，也就是食物名称。
            food_url = 'http://www.boohee.com' + food.find_all('a')[1]['href']
            #用find_all在<li class="item clearfix">标签下，提取出第2个<a>元素href属性的值，跟'http://www.boohee.com'组合在一起，就是食物详情页的链接。
            food_calorie = food.find('p').text
```

```
            #用find在<li class="item clearfix">标签下，提取<p>元素，再用text方法留下纯文本，就提取出了食物的热量。
            print(food_name)
            #打印食物的名称。

tasks_list = []
#创建空的任务列表
for x in range(5):
#相当于创建了5个爬虫
    task = gevent.spawn(crawler)
    #用gevent.spawn()函数创建执行crawler()函数的任务。
    tasks_list.append(task)
    #往任务列表添加任务。
gevent.joinall(tasks_list)
```

1-3-4、csv 存储数据

```
from gevent import monkey
monkey.patch_all()
import gevent,requests, bs4, csv
from gevent.queue import Queue

work = Queue()
url_1 = 'http://www.boohee.com/food/group/{type}?page={page}'
for x in range(1, 4):
    for y in range(1, 4):
        real_url = url_1.format(type=x, page=y)
        work.put_nowait(real_url)

url_2 = 'http://www.boohee.com/food/view_menu?page={page}'
for x in range(1,4):
    real_url = url_2.format(page=x)
    work.put_nowait(real_url)

def crawler():
    headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110 Safari/537.36'
    }
    while not work.empty():
        url = work.get_nowait()
        res = requests.get(url, headers=headers)
        bs_res = bs4.BeautifulSoup(res.text, 'html.parser')
        foods = bs_res.find_all('li', class_='item clearfix')
        for food in foods:
            food_name = food.find_all('a')[1]['title']
            food_url = 'http://www.boohee.com' + food.find_all('a')[1]['href']
            food_calorie = food.find('p').text
            writer.writerow([food_name, food_calorie, food_url])
```

```
            #借助writerow()函数，把提取到的数据：食物名称、食物热量、食物详情链接，写入csv文件。
            print(food_name)

csv_file= open('boohee.csv', 'w', newline='')
#调用open()函数打开csv文件，传入参数：文件名“boohee.csv”、写入模式“w”、newline=''。
writer = csv.writer(csv_file)
# 用csv.writer()函数创建一个writer对象。
writer.writerow(['食物', '热量', '链接'])
#借助writerow()函数往csv文件里写入文字：食物、热量、链接

tasks_list = []
for x in range(5):
    task = gevent.spawn(crawler)
    tasks_list.append(task)
gevent.joinall(tasks_list)
```