程设第十三次作业
20377383 樊思涵

使用协程改写 week12 作业——设计思路

由于本次作业是修饰之前的代码，在学习本节课讲述的多种协程实现方法后，决定使用 gevent 库和 aiofiles 来实现爬虫的协程工作。其中，使用 gevent 将爬虫的部分改为协程，使用 aiofiles 将读写的内容改为协程。

准备部分

```
import gevent
from gevent import monkey
monkey.patch_all()
#将第三方库标记为IO非阻塞
import asyncio
import aiofiles
import os
import pandas as pd
import requests
import urllib.parse
from bs4 import BeautifulSoup
import time
```

gevent

在获取每个歌单 url 的过程中，可以将每一页的爬取工作改为协程工作。

```
for page in range(1,max_page+1):
    jobs.append(gevent.spawn(get_inf1, page))
gevent.joinall(jobs)
```

在爬取每个歌单 url 的过程中，也可以改为协程工作

```
jobs_new=[]
for id in url_lis:
    jobs_new.append(gevent.spawn(get_inf2,id))
gevent.joinall(jobs_new)
```

效果展示代码

```
def main():
    max_page = get_page(cat)
    jobs = []
    start_time = time.time()
    for page in range(1,max_page+1):
        jobs.append(gevent.spawn(get_inf1, page))
    gevent.joinall(jobs)
    first_time =time.time()-start_time
    pd_inf=pd.DataFrame(inf_list)
    url_lis=list(pd_inf[0])
    #img_lis=list(pd_inf[5])
```

```
    start_time = time.time()
    jobs_new=[]
    for id in url_lis:
        jobs_new.append(gevent.spawn(get_inf2,id))
    gevent.joinall(jobs_new)
    second_time = time.time() - start_time
    print(f'使用协程完成任务1耗时{first_time:.2f}')
    print(f'使用协程完成任务2耗时{second_time:.2f}')
```

结果展示

```
解析歌单成功
使用协程完成任务1耗时5.89
使用协程完成任务2耗时160.39
PS E:\code\py_code>
```

与多线程的结果比较

```
解析歌单成功
使用多线程完成任务1耗时4.95
使用多线程完成任务2耗时131.69
PS E:\code\py_code>
```

与单线程的结果比较

```
使用单线程完成任务1耗时20.84
使用单线程完成任务2耗时644.85
PS E:\code\py_code>
```

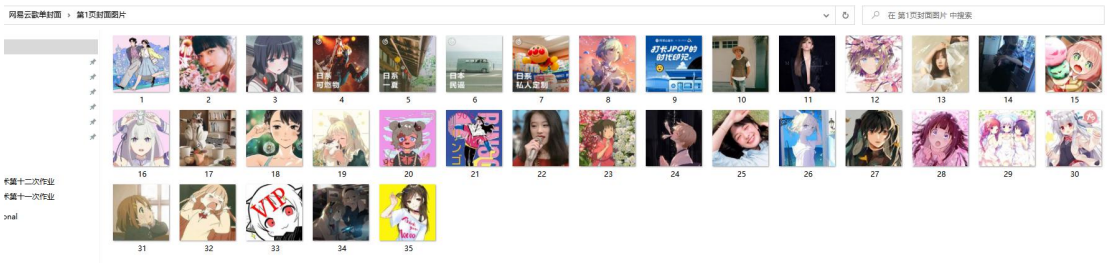
总结：使用协程或多线程相较单线程能节约大量时间

aiofiles

```
async def save_img(url, name):  # 保存图片
    async with aiofiles.open(name, mode='ab') as f:
        img = requests.get(url)
        await f.write(img.content)
        print(name + '文件保存成功')

async def main_save(img_lis):
    tasks=[]
    for i in range(len(img_lis)-1):
        tasks.append(save_img(img_lis[i],str(i)))
    await asyncio.gather(*tasks)
```

保存图片结果



在使用 aiofiles 过程中，容易出现如 loop 无法运行等错误，搜集相关资料后发现该库存在许多问题，使用时应当更为谨慎。