Day05回顾

增量爬取思路

```
1、将爬取过的地址存放到数据库中
2、程序爬取时先到数据库中查询比对，如果已经爬过则不会继续爬取
```

动态加载网站数据抓取

```
1、F12打开控制台，页面动作抓取网络数据包
2、抓取json文件URL地址
# 控制台中 XHR ： 异步加载的数据包
# XHR -> Query String(查询参数)
```

多线程爬虫

- **使用流程**

```
# 1、URL队列
q.put(url)
# 2、线程事件函数
while True:
    if not url_queue.empty():
        ...get()、请求、解析
    else:
        break
# 创建并启动线程
t_list = []
for i in range(5):
    t = Thread(target=parse_page)
    t_list.append(t)
    t.start()
# 阻塞等待回收线程
for i in t_list:
    i.join()
```

day06笔记

作业1 - 小米应用商店

将抓取数据保存到csv文件

```
注意多线程写入的线程锁问题
from threading import Lock
lock = Lock()
lock.acquire()
lock.release()
```

整体思路

```
1、在 __init__(self) 中创建文件对象，多线程操作此对象进行文件写入
2、每个线程抓取数据后将数据进行文件写入，写入文件时需要加锁
3、所有数据抓取完成关闭文件
```

代码实现

```

```

作业2 - 腾讯招聘数据抓取

确定URL地址及目标

```
1、URL：百度搜索腾讯招聘 - 查看工作岗位
2、目标：职位名称、工作职责、岗位要求
```

要求与分析

```
1、通过查看网页源码,得知所需数据均为 Ajax 动态加载
2、通过F12抓取网络数据包,进行分析
3、一级页面抓取数据：职位名称
4、二级页面抓取数据：工作职责、岗位要求
```

一级页面json地址(index在变,timestamp未检查)

```
https://careers.tencent.com/tencentcareer/api/post/Query?timestamp=1563912271089&countryId=&cityId=&bgIds=&productId=&categoryId=&parentCategoryId=&attrId=&keyword=&pageIndex={}&pageSize=10&language=zh-cn&area=cn
```

二级页面地址(postId在变,在一级页面中可拿到)

```
https://careers.tencent.com/tencentcareer/api/post/ByPostId?timestamp=1563912374645&postId={}&language=zh-cn
```

代码实现

```

```

多线程有什么思路?

```
把所有一级页面链接提交到队列,进行多线程数据抓取
```

代码实现

```

```

cookie模拟登录

适用网站及场景

```
抓取需要登录才能访问的页面
```

cookie和session机制

```
# http协议为无连接协议
cookie: 存放在客户端浏览器
session: 存放在Web服务器
```

人人网登录案例

- **方法一 - 登录网站手动抓取Cookie**

```
1、先登录成功1次,获取到携带登陆信息的Cookie
   登录成功 - 个人主页 - F12抓包 - 刷新个人主页 - 找到主页的包(profile)
2、携带着cookie发请求
   ** Cookie
   ** User-Agent
```

```

```

- **方法二 - requests模块处理Cookie**

原理思路及实现

```
# 1. 思路
```

```
requests模块提供了session类,来实现客户端和服务端的会话保持

# 2. 原理
1、实例化session对象
   session = requests.session()
2、让session对象发送get或者post请求
   res = session.post(url=url,data=data,headers=headers)
   res = session.get(url=url,headers=headers)

# 3. 思路梳理
浏览器原理: 访问需要登录的页面会带着之前登录过的cookie
程序原理: 同样带着之前登录的cookie去访问 - 由session对象完成
1、实例化session对象
2、登录网站: session对象发送请求,登录对应网站,把cookie保存在session对象中
3、访问页面: session对象请求需要登录才能访问的页面,session能够自动携带之前的这个cookie,进行请求
```

具体步骤

```
1、寻找Form表单提交地址 - 寻找登录时POST的地址
   查看网页源码,查看form表单,找action对应的地址: http://www.renren.com/PLogin.do

2、发送用户名和密码信息到POST的地址
   * 用户名和密码信息以什么方式发送? -- 字典
     键 : <input>标签中name的值(email,password)
     值 : 真实的用户名和密码
     post_data = {'email':'','password':''}
```

程序实现

```
整体思路
1、先POST: 把用户名和密码信息POST到某个地址中
2、再GET: 正常请求去获取页面信息
```

```

```

- **方法三**

原理

```
1、把抓取到的cookie处理为字典
2、使用requests.get()中的参数:cookies
```

代码实现

```

```

json解析模块

json.loads(json)

- 作用

```
把json格式的字符串转为Python数据类型
```

- 示例

```
html_json = json.loads(res.text)
```

json.dumps(python)

- **作用**

```
把 python 类型 转为 json 类型
```

- **示例**

```
import json

# json.dumps()之前
item = {'name':'QQ','app_id':1}
print('before dumps',type(item))
# json.dumps之后
item = json.dumps(item)
print('after dumps',type(item))
```

json.load(f)

作用

```
将json文件读取,并转为python类型
```

示例

```
import json

with open('D:\\spider_test\\xiaomi.json','r') as f:
    data = json.load(f)

print(data)
```

json.dump(python,f,ensure_ascii=False)

- **作用**

```
把python数据类型 转为 json格式的字符串
# 一般让你把抓取的数据保存为json文件时使用
```

- **参数说明**

```
第1个参数：python类型的数据(字典，列表等)
第2个参数：文件对象
第3个参数：ensure_ascii=False # 序列化时编码
```

- **示例1**

```
import json

item = {'name':'QQ','app_id':1}
with open('小米.json','a') as f:
  json.dump(item,f,ensure_ascii=False)
```

- **示例2**

```
import json

item_list = []
for i in range(3):
  item = {'name':'QQ','id':i}
  item_list.append(item)

with open('xiaomi.json','a') as f:
    json.dump(item_list,f,ensure_ascii=False)
```

练习: 将腾讯招聘数据存入到json文件

```

```

json模块总结

```
# 爬虫最常用
1、数据抓取 - json.loads(html)
   将响应内容由: json 转为 python
2、数据保存 - json.dump(item_list,f,ensure_ascii=False)
   将抓取的数据保存到本地 json文件

# 抓取数据一般处理方式
1、txt文件
2、csv文件
3、json文件
4、MySQL数据库
5、MongoDB数据库
6、Redis数据库
```

selenium+phantomjs/Chrome/Firefox

selenium

- **定义**

```
1、Web自动化测试工具，可运行在浏览器,根据指令操作浏览器
2、只是工具，必须与第三方浏览器结合使用
```

- **安装**

```
Linux: sudo pip3 install selenium
Windows: python -m pip install selenium
```

phantomjs浏览器

- **定义**

```
无界面浏览器(又称无头浏览器)，在内存中进行页面加载,高效
```

- **安装(phantomjs、chromedriver、geckodriver)**

Windows

```
1、下载对应版本的phantomjs、chromedriver、geckodriver
2、把chromedriver.exe拷贝到python安装目录的Scripts目录下(添加到系统环境变量)
   # 查看python安装路径: where python
3、验证
   cmd命令行: chromedriver

# 下载地址
1、chromedriver : 下载对应版本
http://chromedriver.storage.googleapis.com/index.html
2、geckodriver
https://github.com/mozilla/geckodriver/releases
3、phantomjs
https://phantomjs.org/download.html
```

Linux

```
1、下载后解压
   tar -zxvf geckodriver.tar.gz
2、拷贝解压后文件到 /usr/bin/  (添加环境变量)
   sudo cp geckodriver /usr/bin/
3、更改权限
   sudo -i
   cd /usr/bin/
   chmod 777 geckodriver
```

- **使用**

示例代码一：使用 selenium+浏览器 打开百度

```

```

示例代码二：打开百度，搜索赵丽颖，查看

```

```

- 浏览器对象(browser)方法

```
1、browser = webdriver.Chrome(executable_path='path')
2、browser.get(url)
3、browser.page_source # 查看响应内容
4、browser.page_source.find('字符串')
   # 从html源码中搜索指定字符串,没有找到返回: -1
5、browser.quit() # 关闭浏览器
```

- 定位节点

单元素查找(1个节点对象)

```
1、browser.find_element_by_id('')
2、browser.find_element_by_name('')
3、browser.find_element_by_class_name('')
4、browser.find_element_by_xpath('')
... ...
```

多元素查找([节点对象列表])

```
1、browser.find_elements_by_id('')
2、browser.find_elements_by_name('')
3、browser.find_elements_by_class_name('')
4、browser.find_elements_by_xpath('')
... ...
```

- 节点对象操作

```
1、ele.send_keys('') # 搜索框发送内容
2、ele.click()
3、ele.text          # 获取文本内容
4、ele.get_attribute('src') # 获取属性值
```

京东爬虫案例

- 目标

```
1、目标网址 ： https://www.jd.com/
2、抓取目标 ： 商品名称、商品价格、评价数量、商品商家
```

- 思路提醒

```
1、打开京东，到商品搜索页
2、匹配所有商品节点对象列表
3、把节点对象的文本内容取出来，查看规律，是否有更好的处理办法?
4、提取完1页后，判断如果不是最后1页，则点击下一页
   # 如何判断是否为最后1页？？？
```

- 实现步骤

1. **找节点**

```
1、首页搜索框 ： //*[@id="key"]
2、首页搜索按钮   ://*[@id="search"]/div/div[2]/button
3、商品页的 商品信息节点对象列表 ://*[@id="J_goodsList"]/ul/li
```

2. **执行JS脚本，获取动态加载数据**

```
browser.execute_script(
    'window.scrollTo(0,document.body.scrollHeight)'
)
```

3. **代码实现**

```

```

chromedriver设置无界面模式

```
from selenium import webdriver

options = webdriver.ChromeOptions()
# 添加无界面参数
options.add_argument('--headless')
browser = webdriver.Chrome(options=options)
browser.get('http://www.baidu.com/')
browser.save_screenshot('baidu.png')
```

作业

```
1、使用selenium+浏览器抓取 民政部 数据
2、尝试去破解一下百度翻译
```