

# DAY07

## Day06回顾

### 多线程爬虫

#### ■ 思路

```
1 1、将待爬取的URL地址存放到队列中
2 2、多个线程从队列中获取地址,进行数据抓取
3 3、注意获取地址过程中程序阻塞问题
4 while True:
5     if not q.empty():
6         url = q.get()
7         ... ..
8     else:
9         break
```

#### ■ 将抓取数据保存到同一文件

```
1 # 注意多线程写入的线程锁问题
2 from threading import Lock
3 lock = Lock()
4 lock.acquire()
5 python代码块
6 lock.release()
```

#### ■ 代码实现思路

```
1 # 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 self.f = open('xiaomi.csv','a',newline='')
3 self.writer = csv.writer(self.f)
4 self.lock = Lock()
5 # 2、每个线程抓取1页数据后将数据进行文件写入,写入文件时需要加锁
6 def parse_html(self):
7     app_list = []
8     for xxx in xxx:
9         app_list.append([name,link,typ])
10    self.lock.acquire()
11    self.wirter.writerow(app_list)
12    self.lock.release()
13 # 3、所有数据抓取完成关闭文件
14 def main(self):
```

```
15 | self.f.close()
```

## 解析模块汇总

re、lxml+xpath、json

```
1  # re
2  import re
3  pattern = re.compile(r'',re.S)
4  r_list = pattern.findall(html)
5
6  # lxml+xpath
7  from lxml import etree
8  parse_html = etree.HTML(html)
9  r_list = parse_html.xpath('')
10
11 # json
12 # 响应内容由json转为python
13 html = json.loads(res.text)
14 # 所抓数据保存到json文件
15 with open('xxx.json','a') as f:
16     json.dump(item_list,f,ensure_ascii=False)
17
18 # 或
19 f = open('xxx.json','a')
20 json.dump(item_list,f,ensure_ascii=False)
21 f.close()
```

## Day07笔记

### cookie模拟登录

适用网站及场景

```
1 | 抓取需要登录才能访问的页面
```

cookie和session机制

```
1  # http协议为无连接协议
2  cookie: 存放在客户端浏览器
3  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 1、先登录成功1次,获取到携带登录信息的Cookie
2   登录成功 - 个人主页 - F12抓包 - 刷新个人主页 - 找到主页的包(profile)
3 2、携带着cookie发请求
4   ** Cookie
5   ** User-Agent
```

```
1 |
```

## ■ 方法二 - requests模块处理Cookie

原理思路及实现

```
1 # 1. 思路
2 requests模块提供了session类,来实现客户端和服务端的会话保持
3
4 # 2. 原理
5 1、实例化session对象
6   session = requests.session()
7 2、让session对象发送get或者post请求
8   res = session.post(url=url,data=data,headers=headers)
9   res = session.get(url=url,headers=headers)
10
11 # 3. 思路梳理
12 浏览器原理: 访问需要登录的页面会带着之前登录过的cookie
13 程序原理: 同样带着之前登录的cookie去访问 - 由session对象完成
14 1、实例化session对象
15 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
16 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 1、寻找Form表单提交地址 - 寻找登录时POST的地址
2   查看网页源码,查看form表单,找action对应的地址: http://www.renren.com/PLogin.do
3
4 2、发送用户名和密码信息到POST的地址
5   * 用户名和密码信息以什么方式发送? -- 字典
6   键 : <input>标签中name的值(email,password)
7   值 : 真实的用户名和密码
8   post_data = {'email':'','password':''}
9
10 session = requests.session()
11 session.post(url=url,data=data)
```

程序实现

```
1 |
```

## ■ 方法三

原理

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

处理cookie为字典

```
1 |
```

代码实现

```
1 |
```

## json解析模块

json.loads(json)

### ▪ 作用

```
1 | 把json格式的字符串转为Python数据类型
```

### ▪ 示例

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

json.dumps(python)

### ▪ 作用

```
1 | 把 python 类型 转为 json 类型
```

### ▪ 示例

```
1 | import json
2 |
3 | # json.dumps()之前
4 | item = {'name':'QQ','app_id':1}
5 | print('before dumps',type(item)) # dict
6 | # json.dumps之后
7 | item = json.dumps(item)
8 | print('after dumps',type(item)) # str
```

json.load(f)

### 作用

```
1 | 将json文件读取,并转为python类型
```

示例

```
1 import json
2
3 with open('D:\\spider_test\\xiaomi.json','r') as f:
4     data = json.load(f)
5
6 print(data)
```

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

#### ■ 作用

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

#### ■ 参数说明

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

#### ■ 示例1

```
1 import json
2
3 item = {'name':'QQ','app_id':1}
4 with open('小米.json','a') as f:
5     json.dump(item,f,ensure_ascii=False)
```

#### ■ 示例2

```
1 import json
2
3 item_list = []
4 for i in range(3):
5     item = {'name':'QQ','id':i}
6     item_list.append(item)
7
8 with open('xiaomi.json','a') as f:
9     json.dump(item_list,f,ensure_ascii=False)
```

#### json模块总结

```
1 # 爬虫最常用
2 1、数据抓取 - json.loads(html)
3   将响应内容由: json 转为 python
4 2、数据保存 - json.dump(item_list,f,ensure_ascii=False)
5   将抓取的数据保存到本地 json文件
6
7 # 抓取数据一般处理方式
8 1、txt文件
9 2、csv文件
10 3、json文件
11 4、MySQL数据库
12 5、MongoDB数据库
13 6、Redis数据库
```

## *selenium+phantomjs/Chrome/Firefox*

### selenium

#### ▪ 定义

- 1 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 2、只是工具，必须与第三方浏览器结合使用

#### ▪ 安装

- 1 Linux: sudo pip3 install selenium
- 2 Windows: python -m pip install selenium

### phantomjs浏览器

#### ▪ 定义

- 1 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 1、下载对应版本的phantomjs、chromedriver、geckodriver
2 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)
3   # 查看python安装路径: where python
4 3、验证
5   cmd命令行: chromedriver
6
7   # 下载地址
8   1、chromedriver : 下载对应版本
9   http://chromedriver.storage.googleapis.com/index.html
10  2、geckodriver
11   https://github.com/mozilla/geckodriver/releases
12  3、phantomjs
13   https://phantomjs.org/download.html
```

## Linux

```
1 1、下载后解压
2   tar -zxvf geckodriver.tar.gz
3 2、拷贝解压后文件到 /usr/bin/ (添加环境变量)
4   sudo cp geckodriver /usr/bin/
5 3、更改权限
6   sudo -i
7   cd /usr/bin/
8   chmod 777 geckodriver
```

## ■ 使用

示例代码一：使用 selenium+浏览器 打开百度

```
1 # 导入selenium的webdriver接口
2 from selenium import webdriver
3 import time
4
5 # 创建浏览器对象
6 browser = webdriver.PhantomJS()
7 browser.get('http://www.baidu.com/')
8
9 time.sleep(5)
10
11 # 关闭浏览器
12 browser.quit()
```

示例代码二：打开百度，搜索赵丽颖，点击搜索，查看

```

1 from selenium import webdriver
2 import time
3
4 # 1.创建浏览器对象 - 已经打开了浏览器
5 browser = webdriver.Chrome()
6 # 2.输入: http://www.baidu.com/
7 browser.get('http://www.baidu.com/')
8 # 3.找到搜索框,向这个节点发送文字: 赵丽颖
9 browser.find_element_by_xpath('//*[@id="kw"]').send_keys('赵丽颖')
10 # 4.找到 百度一下 按钮,点击一下
11 browser.find_element_by_xpath('//*[@id="su"]').click()

```

## ■ 浏览器对象(browser)方法

```

1 # from selenium import webdriver
2 1、 browser = webdriver.Chrome(executable_path='path')
3 2、 browser.get(url)
4 3、 browser.page_source # HTML结构源码
5 4、 browser.page_source.find('字符串')
6 # 从html源码中搜索指定字符串,没有找到返回: -1
7 5、 browser.quit() # 关闭浏览器

```

## ■ 定位节点

单元素查找(1个节点对象)

```

1 1、 browser.find_element_by_id('')
2 2、 browser.find_element_by_name('')
3 3、 browser.find_element_by_class_name('')
4 4、 browser.find_element_by_xpath('')
5 ... ..

```

多元素查找([节点对象列表])

```

1 1、 browser.find_elements_by_id('')
2 2、 browser.find_elements_by_name('')
3 3、 browser.find_elements_by_class_name('')
4 4、 browser.find_elements_by_xpath('')
5 ... ..

```

## ■ 节点对象操作

```

1 1、 ele.send_keys('') # 搜索框发送内容
2 2、 ele.click()
3 3、 ele.text # 获取文本内容, 包含子节点和后代节点的文本内容
4 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，是否有更好的处理方法？
- 4、提取完1页后，判断如果不是最后1页，则点击下一页
- 5、# 如何判断是否为最后1页？？？

## ■ 实现步骤

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`
- 4、for循环遍历后
- 5、名称：`./div[@class="p-name"]/a/em`
- 6、价格：`./div[@class="p-price"]`
- 7、评论：`./div[@class="p-commit"]/strong`
- 8、商家：`./div[@class="p-shopnum"]`

执行JS脚本，获取动态加载数据

```
1 browser.execute_script(  
2     'window.scrollTo(0,document.body.scrollHeight)'  
3 )
```

代码实现

```
1 |
```

## *chromedriver*设置无界面模式

```
1 from selenium import webdriver  
2  
3 options = webdriver.ChromeOptions()  
4 # 添加无界面参数  
5 options.add_argument('--headless')  
6 browser = webdriver.Chrome(options=options)  
7 browser.get('http://www.baidu.com/')  
8 browser.save_screenshot('baidu.png')
```

## selenium - 键盘操作

```
1 from selenium.webdriver.common.keys import Keys
2
3 browser = webdriver.Chrome()
4 browser.get('http://www.baidu.com/')
5 # 1、在搜索框中输入"selenium"
6 browser.find_element_by_id('kw').send_keys('赵丽颖')
7 # 2、输入空格
8 browser.find_element_by_id('kw').send_keys(Keys.SPACE)
9 # 3、Ctrl+a 模拟全选
10 browser.find_element_by_id('kw').send_keys(Keys.CONTROL, 'a')
11 # 4、Ctrl+c 模拟复制
12 browser.find_element_by_id('kw').send_keys(Keys.CONTROL, 'c')
13 # 5、Ctrl+v 模拟粘贴
14 browser.find_element_by_id('kw').send_keys(Keys.CONTROL, 'v')
15 # 6、输入回车,代替 搜索 按钮
16 browser.find_element_by_id('kw').send_keys(Keys.ENTER)
```

## selenium - 鼠标操作

```
1 from selenium import webdriver
2 # 导入鼠标事件类
3 from selenium.webdriver import ActionChains
4
5 driver = webdriver.Chrome()
6 driver.get('http://www.baidu.com/')
7
8 #移动到 设置, perform()是真正执行操作, 必须有
9 element = driver.find_element_by_xpath('//*[@id="u1"]/a[8]')
10 ActionChains(driver).move_to_element(element).perform()
11
12 #单击, 弹出的Ajax元素, 根据链接节点的文本内容查找
13 driver.find_element_by_link_text('高级搜索').click()
```

## selenium - 切换页面

### ■ 适用网站

1 页面中点开链接出现新的页面, 但是浏览器对象browser还是之前页面的对象

### ■ 应对方案

```
1 # 获取当前所有句柄（窗口）
2 all_handles = browser.window_handles
3 # 切换browser到新的窗口，获取新窗口的对象
4 browser.switch_to.window(all_handles[1])
```

## 民政部网站案例

### 目标

```
1 将民政区划代码爬取到数据库中，按照层级关系（分表 -- 省表、市表、县表）
```

### 数据库中建表

```
1 # 建库
2 create database govdb charset utf8;
3 use govdb;
4 # 建表
5 create table province(
6 p_name varchar(20),
7 p_code varchar(20)
8 )charset=utf8;
9 create table city(
10 c_name varchar(20),
11 c_code varchar(20),
12 c_father_code varchar(20)
13 )charset=utf8;
14 create table county(
15 x_name varchar(20),
16 x_code varchar(20),
17 x_father_code varchar(20)
18 )charset=utf8;
```

### 思路

```
1 1、selenium+Chrome打开一级页面，并提取二级页面最新链接
2 2、增量爬取：和数据库version表中进行比对，确定之前是否爬过（是否有更新）
3 3、如果没有更新，直接提示用户，无须继续爬取
4 4、如果有更新，则删除之前表中数据，重新爬取并插入数据库表
5 5、最终完成后：断开数据库连接，关闭浏览器
```

### 代码实现

```
1 |
```

### SQL命令练习

```
1 # 1. 查询所有省市县信息（多表查询实现）
2
3 # 2. 查询所有省市县信息（连接查询实现）
4
```

## selenium - Web客户端验证

弹窗中的用户名和密码如何输入？

- 1 不用输入，在URL地址中填入就可以

示例: 爬取某一天笔记

```
1 from selenium import webdriver
2
3 url = 'http://tarenacode:code_2013@code.tarena.com.cn/AIDCode/aid1904/15-
  spider/spider_day06_note.zip'
4 browser = webdriver.Chrome()
5 browser.get(url)
```

## selenium - iframe子框架

特点

- 1 网页中嵌套了网页，先切换到iframe子框架，然后再执行其他操作

方法

- 1 browser.switch\_to.iframe(iframe\_element)

示例 - 登录qq邮箱

```
1 from selenium import webdriver
2 import time
3
4 driver = webdriver.Chrome()
5 driver.get('https://mail.qq.com/')
6
7 # 切换到iframe子框架
8 login_frame = driver.find_element_by_id('login_frame')
9 driver.switch_to.frame(login_frame)
10
11 # 用户名+密码+登录
12 driver.find_element_by_id('u').send_keys('qq号')
13 driver.find_element_by_id('p').send_keys('qq密码')
14 driver.find_element_by_id('login_button').click()
15
16 # 预留页面记载时间
17 time.sleep(5)
18
19 # 提取数据
20 ele = driver.find_element_by_id('useralias')
21 print(ele.text)
```

# 百度翻译破解案例

## 目标

- 1 破解百度翻译接口，抓取翻译结果数据

## 实现步骤

### 1、F12抓包,找到json的地址,观察查询参数

- 1、POST地址: `https://fanyi.baidu.com/v2transapi`
- 2、Form表单数据 (多次抓取在变的字段)
- 3 `from: zh`
- 4 `to: en`
- 5 `sign: 54706.276099` #这个是如何生成的?
- 6 `token: a927248ae7146c842bb4a94457ca35ee` # 基本固定,但也想办法获取

### 2、抓取相关JS文件

- 1 右上角 - 搜索 - `sign` - 找到具体JS文件(`index_c8a141d.js`) - 格式化输出

### 3、在JS中寻找sign的生成代码

- 1、在格式化输出的JS代码中搜索: `sign`: 找到如下JS代码: `sign: m(a)`,
- 2、通过设置断点, 找到`m(a)`函数的位置, 即生成`sign`的具体函数
- 3 # 1. `a` 为要翻译的单词
- 4 # 2. 鼠标移动到 `m(a)` 位置处, 点击可进入具体`m(a)`函数代码块

### 4、生成sign的`m(a)`函数具体代码如下(在一个大的define中)

```
1 function a(r) {
2     if (Array.isArray(r)) {
3         for (var o = 0, t = Array(r.length); o < r.length; o++)
4             t[o] = r[o];
5         return t
6     }
7     return Array.from(r)
8 }
9 function n(r, o) {
10     for (var t = 0; t < o.length - 2; t += 3) {
11         var a = o.charAt(t + 2);
12         a = a >= "a" ? a.charCodeAt(0) - 87 : Number(a),
13         a = "+" === o.charAt(t + 1) ? r >>> a : r <<< a,
14         r = "+" === o.charAt(t) ? r + a & 4294967295 : r ^ a
15     }
16     return r
17 }
18 function e(r) {
19     var o = r.match(/[\uD800-\uDBFF][\uDC00-\uDFFF]/g);
20     if (null === o) {
21         var t = r.length;
```

```

22         t > 30 && (r = "" + r.substr(0, 10) + r.substr(Math.floor(t / 2) - 5, 10) +
r.substr(-10, 10))
23     } else {
24         for (var e = r.split(/[\uD800-\uDBFF][\uDC00-\uDFFF]/), C = 0, h = e.length, f = []; h
> C; C++)
25             "" !== e[C] && f.push.apply(f, a(e[C].split(""))),
26             C !== h - 1 && f.push(o[C]);
27         var g = f.length;
28         g > 30 && (r = f.slice(0, 10).join("") + f.slice(Math.floor(g / 2) - 5, Math.floor(g /
2) + 5).join("") + f.slice(-10).join(""))
29     }
30     // var u = void 0
31     // , l = "" + String.fromCharCode(103) + String.fromCharCode(116) +
String.fromCharCode(107);
32     // u = null !== i ? i : (i = window[l] || "") || "";
33     // 断点调试,然后从网页源码中找到 window.gtk的值
34     var u = '320305.131321201'
35
36     for (var d = u.split("."), m = Number(d[0]) || 0, s = Number(d[1]) || 0, S = [], c = 0, v
= 0; v < r.length; v++) {
37         var A = r.charCodeAt(v);
38         128 > A ? S[c++] = A : (2048 > A ? S[c++] = A >> 6 | 192 : (55296 === (64512 & A) && v
+ 1 < r.length && 56320 === (64512 & r.charCodeAt(v + 1))) ? (A = 65536 + ((1023 & A) << 10) +
(1023 & r.charCodeAt(++v)),
39             S[c++] = A >> 18 | 240,
40             S[c++] = A >> 12 & 63 | 128) : S[c++] = A >> 12 | 224,
41             S[c++] = A >> 6 & 63 |
128),
42             S[c++] = 63 & A | 128)
43     }
44     for (var p = m, F = "" + String.fromCharCode(43) + String.fromCharCode(45) +
String.fromCharCode(97) + ("" + String.fromCharCode(94) + String.fromCharCode(43) +
String.fromCharCode(54)), D = "" + String.fromCharCode(43) + String.fromCharCode(45) +
String.fromCharCode(51) + ("" + String.fromCharCode(94) + String.fromCharCode(43) +
String.fromCharCode(98)) + ("" + String.fromCharCode(43) + String.fromCharCode(45) +
String.fromCharCode(102)), b = 0; b < S.length; b++)
45         p += S[b],
46         p = n(p, F);
47     return p = n(p, D),
48         p ^= s,
49         0 > p && (p = (2147483647 & p) + 2147483648),
50         p %= 1e6,
51         p.toString() + "." + (p ^ m)
52 }

```

5、直接将代码写入本地js文件,利用pyexecjs模块执行js代码进行调试

```
1 # 安装pyexecjs模块
2 sudo pip3 install pyexecjs
3
4 # 使用
5 import execjs
6
7 with open('translate.js','r') as f:
8     js_data = f.read()
9
10 # 创建对象
11 exec_object = execjs.compile(js_data)
12 sign = exec_object.eval('e("hello")')
13 print(sign)
```

获取token

```
1 # 在js中
2 token: window.common.token
3 # 在响应中想办法获取此值
4 token_url = 'https://fanyi.baidu.com/?aldtype=16047'
5 regex: "token: '(.*?)'"
```

具体代码实现

```
1 |
```