

DAY08

Day07回顾

cookie模拟登陆

```
1 1、适用网站类型：爬取网站页面时需要登录后才能访问，否则获取不到页面的实际响应数据
2 2、方法1（利用cookie）
3   1、先登录成功1次,获取到携带登陆信息的Cookie（处理headers）
4   2、利用处理的headers向URL地址发请求
5 3、方法2(利用requests.get()中cookies参数)
6   1、先登录成功1次,获取到cookie,处理为字典
7   2、res=requests.get(xxx,cookies=cookies)
8 4、方法3（利用session会话保持）
9   1、实例化session对象
10      session = requests.session()
11   2、先post : session.post(post_url,data=post_data,headers=headers)
12      1、登陆，找到POST地址：form -> action对应地址
13      2、定义字典，创建session实例发送请求
14          # 字典key : <input>标签中name的值(email,password)
15          # post_data = {'email':'','password':''}
16 3、再get : session.get(url,headers=headers)
```

三个池子

```
1 1、User-Agent池
2 2、代理IP池
3 3、cookie池
```

selenium+phantomjs/chrome/firefox

■ 特点

```
1 1、简单，无需去详细抓取分析网络数据包，使用真实浏览器
2 2、需要等待页面元素加载，需要时间，效率低
```

■ 安装

```
1 1、下载、解压
2 2、添加到系统环境变量
3   # windows: 拷贝到Python安装目录的Scripts目录中
4   # Linux : 拷贝到/usr/bin目录中
5 3、Linux中修改权限
6   # sudo -i
7   # cd /usr/bin/
8   # chmod +x phantomjs
9   改权限前: rwxr--r--
10  改权限后: rwxr-xr-x
```

■ 使用流程

```
1 from selenium import webdriver
2
3 # 1、创建浏览器对象
4 browser = webdriver.Firefox(executable_path='/xxx/geckodriver')
5 # 2、输入网址
6 browser.get('URL')
7 # 3、查找节点
8 browser.find_xxxx
9 # 4、做对应操作
10 element.send_keys('')
11 element.click()
12 # 5、关闭浏览器
13 browser.quit()
```

■ 重要知识点

```
1 1、 browser.page_source
2 2、 browser.page_source.find('')
3 3、 node.send_keys('')
4 4、 node.click()
5 5、 find_element AND find_elements
6 6、 browser.execute_script('javascript')
7 7、 browser.quit()
```

Day08笔记

*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 from selenium import webdriver
2 import pymysql
3
4 class GovSpider(object):
5     def __init__(self):
6         # 设置无界面
7         options = webdriver.ChromeOptions()
8         options.add_argument('--headless')
9         # 添加参数
10        self.browser = webdriver.Chrome(options=options)
```

```

11     self.one_url = 'http://www.mca.gov.cn/article/sj/xzqh/2019/'
12     self.db = pymysql.connect(
13         'localhost', 'root', '123456', 'govdb', charset='utf8'
14     )
15     self.cursor = self.db.cursor()
16     # 创建3个列表,用来executemany()往3张表中插入记录
17     self.province_list = []
18     self.city_list = []
19     self.county_list = []
20
21     def get_incr_url(self):
22         self.browser.get(self.one_url)
23         # 提取最新链接,判断是否需要增量爬
24         td = self.browser.find_element_by_xpath(
25             '//td[@class="arlisttd"]/a[contains(@title,"代码")]'
26         )
27         # 提取链接 和 数据库中做比对,确定是否需要抓取
28         # get_attribute()会自动补全提取的链接
29         two_url = td.get_attribute('href')
30         sel = 'select url from version where url=%s'
31         # result为返回的受影响的条数
32         result = self.cursor.execute(sel, [two_url])
33         if result:
34             print('无须爬取')
35         else:
36             td.click()
37             # 切换句柄
38             all_handlers = self.browser.window_handles
39             self.browser.switch_to.window(all_handlers[1])
40             self.get_data()
41             # 把URL地址存入version表
42             dele = 'delete from version'
43             ins = 'insert into version values(%s)'
44             self.cursor.execute(dele)
45             self.cursor.execute(ins, [two_url])
46             self.db.commit()
47
48     def get_data(self):
49         tr_list = self.browser.find_elements_by_xpath(
50             '//tr[@height="19"]'
51         )
52         for tr in tr_list:
53             code = tr.find_element_by_xpath('./td[2]').text.strip()
54             name = tr.find_element_by_xpath('./td[3]').text.strip()
55             print(name, code)
56             # 把数据添加到对应的表中
57             if code[-4:] == '0000':
58                 self.province_list.append([name, code])
59                 if name in ['北京市', '天津市', '上海市', '重庆市']:
60                     self.city_list.append([name, code, code])
61
62             elif code[-2:] == '00':
63                 self.city_list.append([name, code, (code[:2]+'0000')])
64
65             else:
66                 if code[:2] in ['11', '12', '31', '50']:
67                     self.county_list.append([name, code, (code[:2]+'0000')])

```

```

68         else:
69             self.county_list.append([name,code,(code[:4]+'00')])
70
71     # 执行数据库插入语句
72     self.insert_mysql()
73
74     def insert_mysql(self):
75         # 1. 先删除原有数据
76         del_province = 'delete from province'
77         del_city = 'delete from city'
78         del_county = 'delete from county'
79         self.cursor.execute(del_province)
80         self.cursor.execute(del_city)
81         self.cursor.execute(del_county)
82         # 2. 插入新数据
83         ins_province = 'insert into province values(%s,%s)'
84         ins_city = 'insert into city values(%s,%s,%s)'
85         ins_county = 'insert into county values(%s,%s,%s)'
86         self.cursor.executemany(ins_province,self.province_list)
87         self.cursor.executemany(ins_city,self.city_list)
88         self.cursor.executemany(ins_county,self.county_list)
89         # 3.提交到数据库执行
90         self.db.commit()
91
92     def main(self):
93         self.get_incr_url()
94         self.cursor.close()
95         self.db.close()
96         self.browser.quit()
97
98 if __name__ == '__main__':
99     spider = GovSpider()
100    spider.main()

```

SQL命令练习

```

1  # 1. 查询所有省市县信息（多表查询实现）
2  select province.p_name,city.c_name,county.x_name from province,city,county where
   province.p_code=city.c_father_code and city.c_code=county.x_father_code;
3  # 2. 查询所有省市县信息（连接查询实现）
4  select province.p_name,city.c_name,county.x_name from province inner join city on
   province.p_code=city.c_father_code inner join county on city.c_code=county.x_father_code;

```

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 =
4     'http://tarenacode:code_2013@code.tarena.com.cn/AIDCode/aid1905/14_spider/spider_day06.zip'
5 browser = webdriver.Chrome()
6 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邮箱账号')
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 1、POST地址: https://fanyi.baidu.com/v2transapi
2 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 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 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) +
23 r.substr(-10, 10))
24     } else {
25         for (var e = r.split(/[\uD800-\uDBFF][\uDC00-\uDFFF]/), C = 0, h = e.length, f = []; h
26 > C; C++)
27             "" !== e[C] && f.push.apply(f, a(e[C].split(""))),
28             C !== h - 1 && f.push(o[C]);
29         var g = f.length;
```



```

28         g > 30 && (r = f.slice(0, 10).join("") + f.slice(Math.floor(g / 2) - 5, Math.floor(g /
29         2) + 5).join("") + f.slice(-10).join(""))
30     }
31     //     var u = void 0
32     //     , l = "" + String.fromCharCode(103) + String.fromCharCode(116) +
33     String.fromCharCode(107);
34     //     u = null !== i ? i : (i = window[l] || "") || "";
35     // 断点调试,然后从网页源码中找到 window.gtk的值
36     var u = '320305.131321201'
37
38     for (var d = u.split("."), m = Number(d[0]) || 0, s = Number(d[1]) || 0, S = [], c = 0, v
39     = 0; v < r.length; v++) {
40         var A = r.charCodeAt(v);
41         128 > A ? S[c++] = A : (2048 > A ? S[c++] = A >> 6 | 192 : (55296 === (64512 & A) && v
42         + 1 < r.length && 56320 === (64512 & r.charCodeAt(v + 1)) ? (A = 65536 + ((1023 & A) << 10) +
43         (1023 & r.charCodeAt(++v)),
44         S[c++] = A >> 18 | 240,
45         S[c++] = A >> 12 & 63 | 128) : S[c++] = A >> 12 | 224,
46         S[c++] = A >> 6 & 63 |
47         128),
48         S[c++] = 63 & A | 128)
49     }
50     for (var p = m, F = "" + String.fromCharCode(43) + String.fromCharCode(45) +
51     String.fromCharCode(97) + ("" + String.fromCharCode(94) + String.fromCharCode(43) +
52     String.fromCharCode(54)), D = "" + String.fromCharCode(43) + String.fromCharCode(45) +
53     String.fromCharCode(51) + ("" + String.fromCharCode(94) + String.fromCharCode(43) +
54     String.fromCharCode(98)) + ("" + String.fromCharCode(43) + String.fromCharCode(45) +
55     String.fromCharCode(102)), b = 0; b < S.length; b++)
56         p += S[b],
57         p = n(p, F);
58     return p = n(p, D),
59     p ^= s,
60     0 > p && (p = (2147483647 & p) + 2147483648),
61     p %= 1e6,
62     p.toString() + "." + (p ^ m)
63 }

```

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 import requests
2 import re
3 import execjs
4
5 class BaiduTranslateSpider(object):
6     def __init__(self):
7         self.token_url = 'https://fanyi.baidu.com/?aldtype=16047'
8         self.post_url = 'https://fanyi.baidu.com/v2transapi'
9         self.headers = {
10             'accept':
11             'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3',
12             'accept-language': 'zh-CN,zh;q=0.9',
13             'cache-control': 'no-cache',
14             'cookie': 'BAIDUID=52920E829C1F64EE98183B703F4E37A9:FG=1;
15             BIDUPSID=52920E829C1F64EE98183B703F4E37A9; PSTM=1562657403;
16             to_lang_often=%5B%7B%22value%22%3A%22en%22%2C%22text%22%3A%22u82F1%u8BED%22%7D%2C%7B%22value%
17             %22%3A%22zh%22%2C%22text%22%3A%22u4E2D%u6587%22%7D%5D; REALTIME_TRANS_SWITCH=1;
18             FANYI_WORD_SWITCH=1; HISTORY_SWITCH=1; SOUND_SPD_SWITCH=1; SOUND_PREFER_SWITCH=1; delPer=0;
19             BDORZ=B490B5EBF6F3CD402E515D22BCDA1598; BCLID=6890774803653935935;
20             BDSFRCVID=4XAsJeCCxG3DLCbwbJrKDGwjNA0UN_I3KhXZ3J;
21             H_BDCLCKID_SF=trK8oIDaJCvSe6r1MtQ_M4F_qxby26nUQ5neaJ5n0-
22             nnhnL4W46bqJKFLtozKM0I3C7fotJJ5no1o1IRy6CKjJb-jaDqJ5n3bTnjstcS2RREHJrg-
23             trSMDCSHGRGW109WDTm_D_KfxnkOnc6qJj0-jjXqo8K5Ljaa5n-
24             pPKKRAaqD04bPbZL4DdMa7HLtA03mkjbnczfn020P5P1J_e-4syPRG2xRnWIvrKfA-
25             b4ncjRcTehom3xI8LNj4050Tt2LEoDPMJKIBMI_rMbbfhK3CjHqJfaI62aKDs_RCMbhqEIL4eJOIb6_w5gcq0T_HttjtXR
26             0atn7Z5MbSj4Qo5pK95p38bxnDK2rQLb5zah5nhMJS3j7JDMP0-4rJhxy523i5J6vQpnJ8hQ3DRoWXPiQbN7P-
27             p5Z5mAqKl0MLIOkbC_6j5DWDtVLeU7J-n8XbI60XRj85-
28             ohHJrFmTq_q4tehHRMBUo9WDTm_DoTttt5fUj6qJj855jXqo8KMtHJaFf-pPKKRAashnzWjrKqQ0Q5pj-
29             WnQr3mkjbn5yfn020pjPX6joht4syPRG2xRnWIvrKfA-
30             b4ncjRcTehom3xI8LNj4050Tt2LEoC0XtIDhMDvPMCTSMt_HMxrKetJyaR0JhpbWJ5TEPNjDUOdLPDW-
31             46HBM3xbKQw5CJGBf7zhpvdWhC5y6ISKx-_J68DtF5; ZD_ENTRY=baidu; PSINO=2;
32             H_PS_PSSID=26525_1444_21095_29578_29521_28518_29098_29568_28830_29221_26350_29459; locale=zh;
33             Hm_lvt_64ecd82404c51e03dc91cb9e8c025574=1563426293,1563996067;
34             from_lang_often=%5B%7B%22value%22%3A%22zh%22%2C%22text%22%3A%22u4E2D%u6587%22%7D%2C%7B%22valu
35             e%22%3A%22en%22%2C%22text%22%3A%22u82F1%u8BED%22%7D%5D;
36             Hm_lpvT_64ecd82404c51e03dc91cb9e8c025574=1563999768;
37             yjs_js_security_passport=2706b5b03983b8fa12fe756b8e4a08b98fb43022_1563999769_js',
38             'pragma': 'no-cache',
39             'upgrade-insecure-requests': '1',
40             'user-agent': 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML,
41             like Gecko) Chrome/75.0.3770.142 Safari/537.36',
42         }
43
44     # 获取token和gtk
45     def get_token(self):
46         token_url = 'https://fanyi.baidu.com/?aldtype=16047'

```

```

22     # 定义请求头
23     r = requests.get(self.token_url,headers=self.headers)
24     token = re.findall(r"token: '(.*?)'",r.text)
25     window_gtk = re.findall(r"window.*?gtk = '(.*?)';</script>",r.text)
26     if token:
27         return token[0],window_gtk[0]
28
29     # 获取sign
30     def get_sign(self,word,gtk):
31         with open('translate.js','r') as f:
32             js_data = f.read()
33
34             exec_object = execjs.compile(js_data)
35             sign = exec_object.eval('e("{}","{}").format(word,gtk)')
36
37             return sign
38
39     # 主函数
40     def main(self,word,fro,to):
41         token,gtk = self.get_token()
42         sign = self.get_sign(word,gtk)
43         # 找到form表单数据如下,sign和token需要想办法获取
44         form_data = {
45             'from': fro,
46             'to': to,
47             'query': word,
48             'transtype': 'realtime',
49             'simple_means_flag': '3',
50             'sign': sign,
51             'token': token
52         }
53         r = requests.post(self.post_url,data=form_data,headers=self.headers)
54         print(r.json()[ 'trans_result' ][ 'data' ][ 0 ][ 'dst' ])
55
56 if __name__ == '__main__':
57     spider = BaiduTranslateSpider()
58     choice = input('1. 英译汉 2. 汉译英 : ')
59     word = input('请输入要翻译的单词:')
60     if choice == '1':
61         fro,to = 'en','zh'
62     elif choice == '2':
63         fro,to = 'zh','en'
64
65     spider.main(word,fro,to)

```

scrapy框架

■ 定义

1 异步处理框架,可配置和可扩展程度非常高,Python中使用最广泛的爬虫框架

■ 安装

```

1  # Ubuntu安装
2  1、安装依赖包
3      1、sudo apt-get install libffi-dev
4      2、sudo apt-get install libssl-dev
5      3、sudo apt-get install libxml2-dev
6      4、sudo apt-get install python3-dev
7      5、sudo apt-get install libxslt1-dev
8      6、sudo apt-get install zlib1g-dev
9      7、sudo pip3 install -I -U service_identity
10 2、安装scrapy框架
11     1、sudo pip3 install Scrapy

```

```

1  # Windows安装
2  cmd命令行(管理员): python -m pip install Scrapy
3  # Error: Microsoft Visual C++ 14.0 is required xxx

```

■ Scrapy框架五大组件

```

1  1、引擎(Engine)      : 整个框架核心
2  2、调度器(Scheduler) : 维护请求队列
3  3、下载器(Downloader): 获取响应对象
4  4、爬虫文件(Spider)  : 数据解析提取
5  5、项目管道(Pipeline): 数据入库处理
6  *****
7  # 下载器中间件(Downloader Middlewares) : 引擎->下载器,包装请求(随机代理等)
8  # 蜘蛛中间件(Spider Middlewares) : 引擎->爬虫文件,可修改响应对象属性

```

■ scrapy爬虫工作流程

```

1  # 爬虫项目启动
2  1、由引擎向爬虫程序索要第一个要爬取的URL,交给调度器去入队列
3  2、调度器处理请求后出队列,通过下载器中间件交给下载器去下载
4  3、下载器得到响应对象后,通过蜘蛛中间件交给爬虫程序
5  4、爬虫程序进行数据提取:
6      1、数据交给管道文件去入库处理
7      2、对于需要继续跟进的URL,再次交给调度器入队列,依次循环

```

今日作业

```

1  1、将链家二手房案例、小米应用商店案例、猫眼电影案例等使用selenium+Chrome实现
2  2、熟记scrapy框架的组件及工作流程 - 要求能口头描述清楚

```