

# Day03回顾

## 目前反爬总结

### ■ 基于User-Agent反爬

```
1 1、发送请求携带请求头: headers={'User-Agent' : 'Mozilla/5.0 xxxxxx'}
2 2、多个请求随机切换User-Agent
3 1、定义列表存放大量User-Agent, 使用random.choice()每次随机选择
4 2、定义py文件存放大量User-Agent, 使用random.choice()每次随机选择
5 3、使用fake_useragent每次访问随机生成User-Agent
6 * from fake_useragent import UserAgent
7 * ua = UserAgent()
8 * user_agent = ua.random
9 * print(user_agent)
```

### ■ 响应内容前端JS做处理反爬

```
1 1、html页面中可匹配出内容, 程序中匹配结果为空
2 * 响应内容中嵌入js, 对页面结构做了一定调整导致, 通过查看网页源代码, 格式化输出查看结构, 更改xpath或者正则测试
3 2、如果数据出不来可考虑更换 IE 的User-Agent尝试, 数据返回最标准
```

## 请求模块总结

### ■ urllib库使用流程

```
1 # 编码
2 params = {
3     '': '',
4     '': ''
5 }
6 params = urllib.parse.urlencode(params)
7 url = baseurl + params
8
9 # 请求
10 request = urllib.request.Request(url, headers=headers)
11 response = urllib.request.urlopen(request)
12 html = response.read().decode('utf-8')
```

### ■ requests模块使用流程

```
1 | baseurl = 'http://tieba.baidu.com/f?'
2 | html = requests.get(baseurl,params=params,headers=headers).content.decode('utf-8','ignore')
```

## 解析模块总结

### ■ 正则解析re模块

```
1 | import re
2 |
3 | pattern = re.compile('正则表达式',re.S)
4 | r_list = pattern.findall(html)
```

### ■ lxml解析库

```
1 | from lxml import etree
2 |
3 | parse_html = etree.HTML(res.text)
4 | r_list = parse_html.xpath('xpath表达式')
```

## xpath表达式

### ■ 匹配规则

```
1 | 1、节点对象列表
2 |   # xpath示例: //div、//div[@class="student"]、//div/a[@title="stu"]/span
3 | 2、字符串列表
4 |   # xpath表达式中末尾为: @src、@href、text()
```

### ■ xpath高级

```
1 | 1、基准xpath表达式: 得到节点对象列表
2 | 2、for r in [节点对象列表]:
3 |     username = r.xpath('./xxxxxx') # 此处注意遍历后继续xpath一定要以: . 开头, 代表当前节点
```

### 写程序注意

```
1 | # 最终目标: 不要使你的程序因为任何异常而终止
2 | 1、页面请求设置超时时间,并用try捕捉异常,超过指定次数则更换下一个URL地址
3 | 2、所抓取任何数据,获取具体数据前先判断是否存在该数据,可使用列表推到式
4 | # 多级页面数据抓取注意
5 | 1、主线函数: 解析一级页面函数(将所有数据从一级页面中解析并抓取)
```

# Day04笔记

## requests.get()参数

### *查询参数-params*

- 参数类型

```
1 字典,字典中键值对作为查询参数
```

- 使用方法

```
1 1、res = requests.get(url,params=params,headers=headers)
2 2、特点:
3   * url为基准的url地址, 不包含查询参数
4   * 该方法会自动对params字典编码,然后和url拼接
```

- 示例

```
1 import requests
2
3 baseurl = 'http://tieba.baidu.com/f?'
4 params = {
5     'kw' : '赵丽颖吧',
6     'pn' : '50'
7 }
8 headers = {'User-Agent' : 'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64;
9     Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center
10    PC 6.0; .NET4.0C; InfoPath.3)'}
11 # 自动对params进行编码,然后自动和url进行拼接,去发请求
12 res = requests.get(baseurl,params=params,headers=headers)
13 res.encoding = 'utf-8'
14 print(res.text)
```

### *Web客户端验证 参数-auth*

- 作用及类型

```
1 1、针对于需要web客户端用户名密码认证的网站
2 2、auth = ('username','password')
```

- 达内code课程方向案例

```
1 import requests
2 from lxml import etree
```

```

3 import os
4
5 class NoteSpider(object):
6     def __init__(self):
7         self.url = 'http://code.tarena.com.cn/AIDCode/aid1904/14-redis/'
8         self.headers = {'User-Agent': 'Mozilla/5.0'}
9         self.auth = ('tarenacode', 'code_2013')
10
11     # 获取
12     def get_html(self):
13         html = requests.get(url=self.url, auth=self.auth, headers=self.headers).text
14         return html
15
16     # 解析提取数据 + 把笔记压缩包下载完成
17     def parse_page(self):
18         html = self.get_html()
19         xpath_bds = '//a/@href'
20         parse_html = etree.HTML(html)
21         # r_list : ['../', 'day01', 'day02', 'redis_day01.zip']
22         r_list = parse_html.xpath(xpath_bds)
23         for r in r_list:
24             if r.endswith('.zip') or r.endswith('.rar'):
25                 print(r)
26
27 if __name__ == '__main__':
28     spider = NoteSpider()
29     spider.parse_page()

```

思考：爬取具体的笔记文件？

```

1 import requests
2 from lxml import etree
3 import os
4
5 class NoteSpider(object):
6     def __init__(self):
7         self.url = 'http://code.tarena.com.cn/AIDCode/aid1904/14-redis/'
8         self.headers = {'User-Agent': 'Mozilla/5.0'}
9         self.auth = ('tarenacode', 'code_2013')
10
11     # 获取
12     def get_html(self):
13         html = requests.get(url=self.url, auth=self.auth, headers=self.headers).text
14         return html
15
16     # 解析提取数据 + 把笔记压缩包下载完成
17     def parse_page(self):
18         html = self.get_html()
19         xpath_bds = '//a/@href'
20         parse_html = etree.HTML(html)
21         # r_list : ['../', 'day01', 'day02', 'redis_day01.zip']
22         r_list = parse_html.xpath(xpath_bds)
23         for r in r_list:
24             if r.endswith('.zip') or r.endswith('.rar'):
25                 file_url = self.url + r
26                 self.save_files(file_url, r)

```

```

27
28     def save_files(self, file_url, r):
29         html_content = requests.get(file_url, headers=self.headers, auth=self.auth).content
30         # 判断保存路径是否存在
31         directory = '/home/tarena/AID1904/redis/'
32         filename = directory + r
33         if not os.path.exists(directory):
34             os.makedirs(directory)
35
36         with open(filename, 'wb') as f:
37             f.write(html_content)
38             print(r, '下载成功')
39
40 if __name__ == '__main__':
41     spider = NoteSpider()
42     spider.parse_page()

```

## SSL证书认证参数-verify

### ■ 适用网站及场景

- 1、适用网站：https类型网站但是没有经过 证书认证机构 认证的网站
- 2、适用场景：抛出 `SSLERROR` 异常则考虑使用此参数

### ■ 参数类型

```

1 1、verify=True(默认)    : 检查证书认证
2 2、verify=False (常用) : 忽略证书认证
3 # 示例
4 response = requests.get(
5     url=url,
6     params=params,
7     headers=headers,
8     verify=False
9 )

```

## 代理参数-proxies

### ■ 定义

- 1、定义：代替你原来的IP地址去对接网络的IP地址。
- 2、作用：隐藏自身真实IP,避免被封。

### ■ 普通代理

#### 获取代理IP网站

```
1 西刺代理、快代理、全网代理、代理精灵、... ..
```

## 参数类型

```
1 1、语法结构
2     proxies = {
3         '协议': '协议://IP:端口号'
4     }
5 2、示例
6     proxies = {
7         'http': 'http://IP:端口号',
8         'https': 'https://IP:端口号'
9     }
```

## 示例

### 1. 使用免费普通代理IP访问测试网站: <http://httpbin.org/get>

```
1 import requests
2
3 url = 'http://httpbin.org/get'
4 headers = {
5     'User-Agent': 'Mozilla/5.0'
6 }
7 # 定义代理,在代理IP网站中查找免费代理IP
8 proxies = {
9     'http': 'http://112.85.164.220:9999',
10    'https': 'https://112.85.164.220:9999'
11 }
12 html = requests.get(url,proxies=proxies,headers=headers,timeout=5).text
13 print(html)
```

### 2. 思考: 建立一个自己的代理IP池, 随时更新用来抓取网站数据

```
1 import requests
2 from lxml import etree
3 from fake_useragent import UserAgent
4 import random
5 import time
6
7 class GetProxyIP(object):
8     def __init__(self):
9         self.url = 'https://www.xicidaili.com/n/{ }'
10        self.test_url = 'http://www.baidu.com/'
11
12
13        # 生成随机的User-Agent
14        def get_random_header(self):
15            # 创建User-Agent对象
16            ua = UserAgent()
17            headers = {'User-Agent': ua.random}
18            return headers
19
20
```

```

21 # 从西刺代理网站上获取随机的代理IP
22 def get_ip_file(self,url):
23     headers = self.get_random_header()
24     # 访问西刺代理网站国内高匿代理, 找到所有的tr节点对象
25     html = requests.get(url=url, headers=headers).content.decode('utf-8', 'ignore')
26     parse_html = etree.HTML(html)
27     # 基准xpath, 匹配每个代理IP的节点对象列表
28     ipobj_list = parse_html.xpath('//tr')
29
30     # 保存代理
31     self.save_proxies(ipobj_list)
32
33 # 保存代理功能函数
34 def save_proxies(self, ipobj_list):
35     # 从列表第2个元素开始遍历, 因为第1个为: 字段名 (国家、IP、... ...)
36     with open('ip_port.txt', 'a') as f:
37         for ip in ipobj_list[1:]:
38             ip_info = ip.xpath('./td[2]/text()')[0].strip()
39             port_info = ip.xpath('./td[3]/text()')[0].strip()
40             ip_port = ip_info + ':' + port_info + '\n'
41
42             proxies = {
43                 'http': 'http://{}:{ {}'.format(ip_info, port_info),
44                 'https': 'https://{}:{ {}'.format(ip_info, port_info)
45             }
46             print(proxies)
47
48             if self.test_proxy_ip(proxies):
49                 f.write(ip_port)
50                 print(ip_port, 'success')
51             else:
52                 print(ip_port, 'failed')
53
54 # 测试抓取的代理IP是否可用
55 def test_proxy_ip(self, proxies):
56     try:
57         res =
requests.get(self.test_url, headers=self.get_random_header(), proxies=proxies, timeout=5)
58         if res.status_code == 200:
59             return True
60         except Exception as e:
61             return False
62
63 # 主函数
64 def main(self):
65     for i in range(1,1000):
66         url = self.url.format(i)
67         self.get_ip_file(url)
68         time.sleep(random.randint(5,10))
69
70 if __name__ == '__main__':
71     spider = GetProxyIP()
72     spider.main()

```

## 2、写一个获取收费开放代理的接口

```

1 # 获取开放代理的接口
2 import requests
3
4 def test_ip(ip):
5     url = 'http://www.baidu.com/'
6     ip_port = ip.split(':')
7     proxies = {
8         'http': 'http://{}:{}'.format(ip_port[0], ip_port[1]),
9         'https': 'https://{}:{}'.format(ip_port[0], ip_port[1]),
10    }
11    res = requests.get(url=url)
12    if res.status_code == 200:
13        return True
14    else:
15        return False
16
17 # 提取代理IP
18 def get_ip_list():
19     api_url = 'http://dev.kdlapi.com/api/getproxy/?
20     orderid=946562662041898&num=100&protocol=1&method=2&an_an=1&an_ha=1&sep=2'
21     html = requests.get(api_url).content.decode('utf-8', 'ignore')
22     ip_port_list = html.split('\n')
23
24     for ip in ip_port_list:
25         with open('proxy_ip.txt', 'a') as f:
26             if test_ip(ip):
27                 f.write(ip + '\n')
28
29 if __name__ == '__main__':
30     get_ip_list()

```

### 3、使用随机收费开放代理IP写爬虫

```

1 import random
2 import requests
3
4 class BaiduSpider(object):
5     def __init__(self):
6         self.url = 'http://www.baidu.com/'
7         self.headers = {'User-Agent': 'Mozilla/5.0'}
8         self.blag = 1
9
10    def get_proxies(self):
11        with open('proxy_ip.txt', 'r') as f:
12            result = f.readlines()
13            proxy_ip = random.choice(result)[-1]
14            L = proxy_ip.split(':')
15            proxy_ip = {
16                'http': 'http://{}:{}'.format(L[0], L[1]),
17                'https': 'https://{}:{}'.format(L[0], L[1])
18            }
19            return proxy_ip
20
21    def get_html(self):
22        proxies = self.get_proxies()
23        if self.blag <= 3:

```



```

24         try:
25             html =
requests.get(url=self.url,proxies=proxies,headers=self.headers,timeout=5).text
26             print(html)
27         except Exception as e:
28             print('Retry')
29             self.blag += 1
30             self.get_html()
31
32 if __name__ == '__main__':
33     spider = BaiduSpider()
34     spider.get_html()

```

## ■ 私密代理

### 语法格式

```

1 1、语法结构
2 proxies = {
3     '协议': '协议://用户名:密码@IP:端口号'
4 }
5
6 2、示例
7 proxies = {
8     'http': 'http://用户名:密码@IP:端口号',
9     'https': 'https://用户名:密码@IP:端口号'
10 }

```

### 示例代码

```

1 import requests
2 url = 'http://httpbin.org/get'
3 proxies = {
4     'http': 'http://309435365:szayclhp@106.75.71.140:16816',
5     'https': 'https://309435365:szayclhp@106.75.71.140:16816',
6 }
7 headers = {
8     'User-Agent' : 'Mozilla/5.0',
9 }
10
11 html = requests.get(url,proxies=proxies,headers=headers,timeout=5).text
12 print(html)

```

# 控制台抓包

## ■ 打开方式及常用选项

- 1 1、打开浏览器，F12打开控制台，找到Network选项卡
- 2 2、控制台常用选项
  - 3 1、Network：抓取网络数据包
    - 4 1、ALL：抓取所有的网络数据包
    - 5 2、XHR：抓取异步加载的网络数据包

```
6         3、JS：抓取所有的JS文件
7     2、Sources：格式化输出并打断点调试JavaScript代码，助于分析爬虫中一些参数
8     3、Console：交互模式，可对JavaScript中的代码进行测试
9 3、抓取具体网络数据包后
10     1、单击左侧网络数据包地址，进入数据包详情，查看右侧
11     2、右侧：
12         1、Headers：整个请求信息
13             General、Response Headers、Request Headers、Query String、Form Data
14     2、Preview：对响应内容进行预览
15     3、Response：响应内容
```

## requests.post()

### ■ 适用场景

```
1 Post类型请求的网站
```

### ■ 参数-data

```
1 response = requests.post(url,data=data,headers=headers)
2 # data : post数据 (Form表单数据-字典格式)
```

### ■ 请求方式的特点

```
1 # 一般
2 GET请求：参数在URL地址中有显示
3 POST请求：Form表单提交数据
```

## 有道翻译破解案例(post)

### 1. 目标

```
1 破解有道翻译接口，抓取翻译结果
2 # 结果展示
3 请输入要翻译的词语：elephant
4 翻译结果：大象
5 *****
6 请输入要翻译的词语：喵喵叫
7 翻译结果：mews
```

### 2. 实现步骤

```
1 1、浏览器F12开启网络抓包,Network-All,页面翻译单词后找Form表单数据
2 2、在页面中多翻译几个单词，观察Form表单数据变化（有数据是加密字符串）
3 3、刷新有道翻译页面，抓取并分析JS代码（本地JS加密）
4 4、找到JS加密算法，用Python按同样方式加密生成加密数据
5 5、将Form表单数据处理为字典，通过requests.post()的data参数发送
```

## 具体实现

- 1、开启F12抓包，找到Form表单数据如下：

```
1 i: 喵喵叫
2 from: AUTO
3 to: AUTO
4 smartresult: dict
5 client: fanyideskweb
6 salt: 15614112641250
7 sign: 94008208919faa19bd531acde36aac5d
8 ts: 1561411264125
9 bv: f4d62a2579ebb44874d7ef93ba47e822
10 doctype: json
11 version: 2.1
12 keyfrom: fanyi.web
13 action: FY_BY_REALTIME
```

- 2、在页面中多翻译几个单词，观察Form表单数据变化

```
1 salt: 15614112641250
2 sign: 94008208919faa19bd531acde36aac5d
3 ts: 1561411264125
4 bv: f4d62a2579ebb44874d7ef93ba47e822
5 # 但是bv的值不变
```

- 3、一般为本地js文件加密，刷新页面，找到js文件并分析JS代码

```
1 # 方法1
2 Network - JS选项 - 搜索关键词salt
3 # 方法2
4 控制台右上角 - Search - 搜索salt - 查看文件 - 格式化输出
5
6 # 最终找到相关JS文件 : fanyi.min.js
```

- 4、打开JS文件，分析加密算法，用Python实现

```
1 # ts : 经过分析为13位的时间戳，字符串类型
2 js代码实现: "" + (new Date).getTime()
3 python实现: str(int(time.time()*1000))
4
5 # salt
6 js代码实现: r+parseInt(10 * Math.random(), 10);
7 python实现: ts + str(random.randint(0,9))
8
9 # sign (设置断点调试，来查看 e 的值，发现 e 为要翻译的单词)
10 js代码实现: n.md5("fanyideskweb" + e + salt + "n%A-rKaT5fb[Gy?;N5@Tj")
11 python实现:
12 from hashlib import md5
13 s = md5()
14 s.update("fanyideskweb" + e + salt + "n%A-rKaT5fb[Gy?;N5@Tj".encode())
15 sign = s.hexdigest()
```

- 5、代码实现

```

1 import requests
2 import time
3 from hashlib import md5
4 import random
5
6 # 获取相关加密算法的结果
7 def get_salt_sign_ts(word):
8     # salt
9     salt = str(int(time.time()*1000)) + str(random.randint(0,9))
10    # sign
11    string = "fanyideskweb" + word + salt + "n%A-rKaT5fb[Gy?;N5@Tj"
12    s = md5()
13    s.update(string.encode())
14    sign = s.hexdigest()
15    # ts
16    ts = str(int(time.time()*1000))
17    return salt,sign,ts
18
19 # 攻克有道
20 def attack_yd(word):
21     salt,sign,ts = get_salt_sign_ts(word)
22     # url为抓包抓到的地址 F12 -> translate_o -> post
23     url = 'http://fanyi.youdao.com/translate_o?smartresult=dict&smartresult=rule'
24     headers = {
25         "Accept": "application/json, text/javascript, */*; q=0.01",
26         "Accept-Encoding": "gzip, deflate",
27         "Accept-Language": "zh-CN,zh;q=0.9",
28         "Connection": "keep-alive",
29         "Content-Length": "238",
30         "Content-Type": "application/x-www-form-urlencoded; charset=UTF-8",
31         "Cookie": "OUTFOX_SEARCH_USER_ID=-1449945727@10.169.0.82;
OUTFOX_SEARCH_USER_ID_NCOO=1492587933.976261; JSESSIONID=aaa5_Lj5jzfQZ_IPPuaSw;
__rl_test_cookies=1559193524685",
32         "Host": "fanyi.youdao.com",
33         "Origin": "http://fanyi.youdao.com",
34         "Referer": "http://fanyi.youdao.com/",
35         "User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/74.0.3729.169 Safari/537.36",
36         "X-Requested-With": "XMLHttpRequest",
37     }
38     # Form表单数据
39     data = {
40         'i': word,
41         'from': 'AUTO',
42         'to': 'AUTO',
43         'smartresult': 'dict',
44         'client': 'fanyideskweb',
45         'salt': salt,
46         'sign': sign,
47         'ts': ts,
48         'bv': 'cf156b581152bd0b259b90070b1120e6',
49         'doctype': 'json',
50         'version': '2.1',
51         'keyfrom': 'fanyi.web',
52         'action': 'FY_BY_REALTIME'
53     }
54

```

```
55     json_html = requests.post(url,data=data,headers=headers).json()
56     result = json_html['translateResult'][0][0]['tgt']
57     return result
58
59 if __name__ == '__main__':
60     word = input('请输入要翻译的单词: ')
61     result = attack_yd(word)
62     print(result)
```

## 今日作业

- 1、Web客户端验证 - 如何抓取文件及保存文件
- 2、代理参数 - 如何建立自己的IP代理池,并使用随机代理IP访问网站
- 3、仔细复习并总结有道翻译案例,抓包流程,代码实现