

# 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
```

### ■ 响应内容前端做处理反爬

```
1 1、html页面中可匹配出内容, 程序中匹配结果为空
2 * 前端对响应内容做了一些结构上的调整导致, 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、如果数据不出来可考虑更换 IE 的User-Agent尝试, 数据返回最标准
```

### ■ 基于IP反爬

```
1 控制爬取速度, 每爬取页面后随机休眠一定时间, 再继续爬取下一个页面
```

## 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  # 方法一：先使用urllib.parse编码，然后使用requests发请求
2  url = baseurl + urllib.parse.urlencode({dict})
3  html = requests.get(url, headers=headers).text
4
5  # 方法二：利用params参数(自动对查询参数编码再拼接)
6  html = requests.get(baseurl, params=params, headers=headers).text

```

## 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
```

# 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' : '校花吧',
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)
```

### *代理参数-proxies*

- 定义

- 1 1、定义：代替你原来的IP地址去对接网络的IP地址。
- 2 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 |
```

- 2、写一个获取开放代理的接口

```
1 | # getip.py
```

- 3、使用收费开放代理IP访问测试网站: <http://httpbin.org/get>

- 1 1、从代理网站上获取购买的普通代理的api链接
- 2 2、从api链接中提取出IP
- 3 3、随机选择代理IP访问网站进行数据抓取

```
1 |
```

- 4、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 |
```

## ■ 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 |
```

## 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 |
```

## SSL 证书认证参数-verify

#### ■ 适用网站及场景

```
1 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 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     proxies=proxies,
8     headers=headers,
9     verify=False
10 )
```

# 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、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、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、刷新有道翻译页面,抓取并分析JS代码(本地JS加密)
- 4、找到JS加密算法,用Python按同样方式加密生成加密数据
- 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:
4
5 # salt
6 js代码实现: r+parseInt(10 * Math.random(), 10);
7 python实现:
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 + "@6f#X3=cUncYssPsuRUE")
11 python实现:

```

- 5、代码实现

```

1 |

```

# 动态加载数据抓取-Ajax

## ■ 特点

- 1、右键 -> 查看网页源码中没有具体数据
- 2、滚动鼠标滑轮或其他动作时加载

## ■ 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
- 3、# 控制台中 XHR : 异步加载的数据包
- 4、# XHR -> Query String(查询参数)

## 豆瓣电影数据抓取案例

## ■ 目标

- 1、地址：豆瓣电影 - 排行榜 - 剧情
- 2、目标：电影名称、电影评分

## ■ F12抓包 (XHR)

- 1、Request URL(基准URL地址) : `https://movie.douban.com/j/chart/top_list?`
- 2、Query String(查询参数)
- 3、# 抓取的查询参数如下：
- 4、`type: 13`
- 5、`interval_id: 100:90`
- 6、`action: ''`
- 7、`start: 0`
- 8、`limit: 用户输入的电影数量`

## ■ json模块的使用

- 1、`json.loads(json格式的字符串)`：把json格式的字符串转为python数据类型
- 2、# 示例
- 3、`html = json.loads(res.text)`
- 4、`print(type(html))`

## ■ 代码实现

1 |

思考: 实现用户在终端输入电影类型和电影数量，将对应电影信息抓取到数据库



# 今日作业

- 1 1、仔细复习有道翻译案例，抓包流程，代码实现
- 2 2、豆瓣电影升级（输入电影类型、抓取数量）
- 3 3、抓取腾讯招聘职位信息
- 4 4、抓取腾讯招聘职位详情
- 5 5、哔哩哔哩小视频下载
- 6 # 1、url：<http://vc.bilibili.com/p/eden/rank#/?tab=全部>
- 7 # 2、抓取目标：所有异步加载的小视频