DAY05

Day04回顾

requests.get()参数

```
1、url
2、params -> {} : 查询参数 Query String
3、proxies -> {}
   proxies = {
    'http':'http://1.1.1.1:8888',
    'https':'https://1.1.1.1:8888'
   }
4、auth -> ('tarenacode','code_2013')
5、verify -> True/False
6、timeout
```

常见的反爬机制及处理方式

```
1、Headers反爬虫 : Cookie、Referer、User-Agent
   解决方案：通过F12获取headers,传给requests.get()方法

2、IP限制 : 网站根据IP地址访问频率进行反爬,短时间内限制IP访问
   解决方案:
        1、构造自己IP代理池,每次访问随机选择代理,经常更新代理池
        2、购买开放代理或私密代理IP
        3、降低爬取的速度

3、User-Agent限制 : 类似于IP限制
   解决方案：构造自己的User-Agent池,每次访问随机选择

5、对查询参数或Form表单数据认证(salt、sign)
   解决方案：找到JS文件,分析JS处理方法,用Python按同样方式处理

6、对响应内容做处理
   解决方案：打印并查看响应内容,用xpath或正则做处理
```

Day05笔记

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**

```
西刺代理、快代理、全网代理、代理精灵、... ...
```

- **参数类型**

```
1、语法结构
    proxies = {
        '协议':'协议://IP:端口号'
    }
2、示例
    proxies = {
      'http':'http://IP:端口号',
      'https':'https://IP:端口号'
  }
```

- 示例

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

```
import requests

url = 'http://httpbin.org/get'
headers = {
    'User-Agent':'Mozilla/5.0'
}
# 定义代理,在代理IP网站中查找免费代理IP
proxies = {
    'http':'http://112.85.164.220:9999',
    'https':'https://112.85.164.220:9999'
}
html = requests.get(url,proxies=proxies,headers=headers,timeout=5).text
print(html)
```

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、从西刺代理IP网站上，抓取免费代理IP
2、测试抓取的IP，可用的保存在文件中
```

思考 - 代码实现

```
1
```

写一个获取收费开放代理的接口

```
1
```

私密代理

- **语法格式**

```
1、语法结构
proxies = {
    '协议':'协议://用户名:密码@IP:端口号'
}

2、示例
proxies = {
  'http':'http://用户名:密码@IP:端口号',
  'https':'https://用户名:密码@IP:端口号'
}
```

示例代码

```
import requests
url = 'http://httpbin.org/get'
proxies = {
    'http':'http://309435365:szayclhp@1.195.160.232:17509',
    'https':'https://309435365:szayclhp@1.195.160.232:17509'
}
headers = {
    'User-Agent' : 'Mozilla/5.0',
}

html = requests.get(url,proxies=proxies,headers=headers,timeout=5).text
print(html)
```

控制台抓包

- **打开方式及常用选项**

```
1、打开浏览器，F12打开控制台，找到Network选项卡
2、控制台常用选项
   1、Network：抓取网络数据包
        1、ALL：抓取所有的网络数据包
        2、XHR：抓取异步加载的网络数据包
        3、JS ：抓取所有的JS文件
   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
   3、Console：交互模式，可对JavaScript中的代码进行测试
3、抓取具体网络数据包后
   1、单击左侧网络数据包地址，进入数据包详情，查看右侧
```

```
    2、右侧：
        1、Headers：整个请求信息
           General、Response Headers、Request Headers、Query String、Form Data
        2、Preview：对响应内容进行预览
        3、Response：响应内容
```

requests.post()参数

- **适用场景**

```
Post类型请求的网站
```

- **参数-data**

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

- **请求方式的特点**

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

有道翻译破解案例(post)

- **目标**

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

- **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：**

```
i: 喵喵叫
from: AUTO
to: AUTO
smartresult: dict
client: fanyideskweb
salt: 15614112641250
sign: 94008208919faa19bd531acde36aac5d
ts: 1561411264125
bv: f4d62a2579ebb44874d7ef93ba47e822
doctype: json
version: 2.1
keyfrom: fanyi.web
action: FY_BY_REALTlME
```

- **2、在页面中多翻译几个单词，观察Form表单数据变化**

```
salt: 15614112641250
sign: 94008208919faa19bd531acde36aac5d
ts: 1561411264125
bv: f4d62a2579ebb44874d7ef93ba47e822
# 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**

```
# 方法1
Network - JS选项 - 搜索关键词salt
# 方法2
控制台右上角 - Search - 搜索salt - 查看文件 - 格式化输出
# 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**

```
# ts : 经过分析为13位的时间戳，字符串类型
js代码实现:  "" + (new Date).getTime()
python实现:

# salt
js代码实现:  ts+parseInt(10 * Math.random(), 10);
python实现:

# sign (设置断点调试，来查看 e 的值，发现 e 为要翻译的单词)
js代码实现: n.md5("fanyideskweb" + e + salt + "n%A-rKaT5fb[Gy?;N5@Tj")
python实现:
```

- **5、代码实现**

```

```

python中正则处理headers和formdata

```
1、pycharm进入方法 ： Ctrl + r ，选中 Regex
2、处理headers和formdata
  (.*): (.*)
  "$1": "$2",
3、点击 Replace All
```

民政部网站数据抓取

目标

```
1、URL: http://www.mca.gov.cn/ - 民政数据 - 行政区划代码
   即: http://www.mca.gov.cn/article/sj/xzqh/2019/
2、目标: 抓取最新中华人民共和国县以上行政区划代码
```

实现步骤

- **1、从民政数据网站中提取最新行政区划代码链接**

```
# 特点
1、最新的在上面
2、命名格式: 2019年X月中华人民共和国县以上行政区划代码
```

2、从二级页面链接中提取真实链接（反爬-响应内容中嵌入JS，指向新的链接）

```
1、向二级页面链接发请求得到响应内容，并查看嵌入的JS代码
2、正则提取真实的二级页面链接
```

3、真实链接中提取所需数据

4、代码实现

```

```

扩展

```
1、建立增量爬虫 - 网站有更新时抓取，否则不抓
2、所抓数据存到数据库，按照层级关系分表存储 - 省、市、县表
```

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,或者页面局部刷新
```

- **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 -> QueryStringParameters(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)
# 抓取的查询参数如下：
type: 13 # 电影类型
interval_id: 100:90
action: ''
start: 0  # 每次加载电影的起始索引值 0 20 40 60
limit: 20 # 每次加载的电影数量
```

- **代码实现 - 全站抓取 - 完美接口 - 指定类型所有电影信息**

```

```

今日作业

```
1、有道翻译案例复写一遍
2、抓取腾讯招聘数据(两级页面 - 职位名称、岗位职责、工作要求)
3、抓取百度图片 - 所有,而不是30张
4、民政部数据抓取案例完善
    # 1、将抓取的数据存入数据库，最好分表按照层级关系去存
    # 2、增量爬取时表中数据也要更新
```