# 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**

```python
import requests
from lxml import etree
import time
import random
from fake_useragent import UserAgent

class GetProxyIP(object):
  def __init__(self):
    self.url = 'https://www.xicidaili.com/nn/{}'

  # 随机生成1个User-Agent
  def get_headers(self):
    ua = UserAgent()
    headers = { 'User-Agent':ua.random }

    return headers

  # 获取可用代理IP文件
  def get_ip_file(self,url):
    html = requests.get(url=url,headers=self.get_headers(),timeout=5).text

    parse_html = etree.HTML(html)
    tr_list = parse_html.xpath('//tr')
    for tr in tr_list[1:]:
      ip = tr.xpath('./td[2]/text()')[0]
      port = tr.xpath('./td[3]/text()')[0]
      # 测试ip:port是否可用
      self.test_ip(ip,port)

  def test_ip(self,ip,port):
    proxies = {
      'http':'http://{}:{}'.format(ip,port),
      'https': 'https://{}:{}'.format(ip, port),
    }
    test_url = 'http://www.baidu.com/'
    try:
      res = requests.get(url = test_url,proxies = proxies,timeout = 8)
      if res.status_code == 200:
        print(ip,port,'Success')
        with open('proxies.txt','a') as f:
          f.write(ip + ':' + port + '\n')
    except Exception as e:
      print(ip,port,'Failed')

  # 主函数
  def main(self):
    for i in range(1,1001):
      url = self.url.format(i)
      self.get_ip_file(url)
      time.sleep(random.randint(5,10))

if __name__ == '__main__':
  spider = GetProxyIP()
  spider.main()
```

**写一个获取收费开放代理的接口**

```python
# 获取开放代理的接口
import requests

def test_ip(ip):
    url = 'http://www.baidu.com/'
    proxies = {
        'http':'http://{}'.format(ip),
        'https':'https://{}'.format(ip),
    }

    try:
      res = requests.get(url=url,proxies=proxies,timeout=8 )
      if res.status_code == 200:
          return True
    except Exception as e:
        return False

# 提取代理IP
def get_ip_list():
  api_url = 'http://dev.kdlapi.com/api/getproxy/?orderid=946562662041898&num=100&protocol=1&method=2&an_an=1&an_ha=1&sep=2'
  html = requests.get(api_url).content.decode('utf-8','ignore')
  # ip_port_list: ['IP:PORT','IP:PORT','']
  ip_port_list = html.split('\r\n')

  # 依次遍历代理IP,并进行测试
  with open('proxy_ip.txt','a') as f:
    for ip in ip_port_list:
      if test_ip(ip):
            f.write(ip + '\n')

if __name__ == '__main__':
    get_ip_list()
```

**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**

```python
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**

```python
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:  str(int(time.time()*1000))

# salt
js代码实现:  ts+parseInt(10 * Math.random(), 10);
python实现:  ts+str(random.randint(0,9))

# 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:
from hashlib import md5
string = "fanyideskweb" + word + salt + "n%A-rKaT5fb[Gy?;N5@Tj"
s = md5()
s.update(string.encode())
sign = s.hexdigest()
```

- **5、代码实现**

```
import requests
import time
import random
from hashlib import md5

class YdSpider(object):
  def __init__(self):
    # url一定为F12抓到的 headers -> General -> Request URL
    self.url = 'http://fanyi.youdao.com/translate_o?smartresult=dict&smartresult=rule'
    self.headers = {
      # 检查频率最高 - 3个
      "Cookie": "OUTFOX_SEARCH_USER_ID=970246104@10.169.0.83;
OUTFOX_SEARCH_USER_ID_NCOO=570559528.1224236;
_ntes_nnid=96bc13a2f5ce64962adfd6a278467214,1551873108952; JSESSIONID=aaae9i7plXPlKaJH_gkYw;
td_cookie=18446744072941336803; SESSION_FROM_COOKIE=unknown;
___rl__test__cookies=1565689460872",
      "Referer": "http://fanyi.youdao.com/",
      "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/76.0.3809.100 Safari/537.36",
    }

  # 获取salt,sign,ts
  def get_salt_sign_ts(self,word):
    # ts
    ts = str(int(time.time()*1000))
    # salt
```

```
    salt = ts + str(random.randint(0,9))
    # sign
    string = "fanyideskweb" + word + salt + "n%A-rKaT5fb[Gy?;N5@Tj"
    s = md5()
    s.update(string.encode())
    sign = s.hexdigest()

    return salt,sign,ts

  # 主函数
  def attack_yd(self,word):
    # 1. 先拿到salt,sign,ts
    salt,sign,ts = self.get_salt_sign_ts(word)
    # 2. 定义form表单数据为字典: data={}
    # 检查了salt sign
    data = {
      "i": word,
      "from": "AUTO",
      "to": "AUTO",
      "smartresult": "dict",
      "client": "fanyideskweb",
      "salt": salt,
      "sign": sign,
      "ts": ts,
      "bv": "7e3150ecbdf9de52dc355751b074cf60",
      "doctype": "json",
      "version": "2.1",
      "keyfrom": "fanyi.web",
      "action": "FY_BY_REALTlME",
    }
    # 3. 直接发请求:requests.post(url,data=data,headers=xxx)
    res = requests.post(
      url=self.url,
      data=data,
      headers=self.headers
    )
    # res.json() 将json格式的字符串转为python数据类型
    html = res.json()
    # html:{'translateResult': [[{'tgt': '你好', 'src': 'hello'}]], 'errorCode': 0, 'type':
'en2zh-CHS', 'smartResult': {'entries': ['', 'n. 表示问候, 惊奇或唤起注意时的用语\r\n', 'int.
喂; 哈罗\r\n', 'n. (Hello)人名; (法)埃洛\r\n'], 'type': 1}}
    result = html['translateResult'][0][0]['tgt']

    print(result)

  # 主函数
  def main(self):
    # 输入翻译单词
    word = input('请输入要翻译的单词:')
    self.attack_yd(word)

if __name__ == '__main__':
  spider = YdSpider()
  spider.main()
```

# 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**

```
import requests
from lxml import etree
import re

class GovementSpider(object):
  def __init__(self):
    self.url = 'http://www.mca.gov.cn/article/sj/xzqh/2019/'
    self.headers = {'User-Agent':'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; InfoPath.3)'}


  # 获取假链接
  def get_false_link(self):
```

```
    html = requests.get(url = self.url,headers = self.headers).text
    # 解析
    parse_html = etree.HTML(html)
    # a_list: [<element a at xx>,<element a at xxx>]
    a_list = parse_html.xpath('//a[@class="artitlelist"]')
    for a in a_list:
      # get()方法:获取某个属性的值
      title = a.get('title')
      if title.endswith('代码'):
        false_link = 'http://www.mca.gov.cn' + a.get('href')
        self.get_true_link(false_link)
        break


  # 获取真链接
  def get_true_link(self,false_link):
    # 先获取假链接的响应,然后根据响应获取真链接
    html = requests.get(url = false_link,headers = self.headers).text
    # 利用正则提取真实链接
    re_bds = r'window.location.href="(.*?)"'
    pattern = re.compile(re_bds,re.S)
    true_link = pattern.findall(html)[0]

    self.parse_html(true_link)

  # 数据提取
  def parse_html(self,true_link):
    html = requests.get( url = true_link,headers = self.headers).text

    # xpath提取数据
    parse_html = etree.HTML(html)
    tr_list = parse_html.xpath('//tr[@height="19"]')
    for tr in tr_list:
      code = tr.xpath('./td[2]/text()')[0].strip()
      name = tr.xpath('./td[3]/text()')[0].strip()

      print(name,code)


  # 主函数
  def main(self):
    self.get_false_link()

if __name__ == '__main__':
  spider = GovementSpider()
  spider.main()
```

**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**

```python
import requests
import time
import random
import re
from useragents import ua_list

class DoubanSpider(object):
    def __init__(self):
        self.url = 'https://movie.douban.com/j/chart/top_list?'
        self.i = 0

    # 获取随机headers
    def get_headers(self):
        headers = {'User-Agent':random.choice(ua_list)}

```

```
        return headers

    # 获取页面
    def get_page(self,params):
      # 返回 python 数据类型
        html = requests.get(url=self.url,params=params,headers=self.get_headers()).json()
        self.parse_page(html)

    # 解析并保存数据
    def parse_page(self,html):
        item = {}
        # html为大列表 [{电影1信息},{},{}]
        for one in html:
            # 名称 + 评分
            item['name'] = one['title'].strip()
            item['score'] = float(one['score'].strip())
            # 打印测试
            print(item)
            self.i += 1

    # 获取电影总数
    def total_number(self,type_number):
        # F12抓包抓到的地址
        url = 'https://movie.douban.com/j/chart/top_list_count?type=
{}&interval_id=100%3A90'.format(type_number)
        headers = self.get_headers()
        html = requests.get(url=url,headers=headers).json()
        total = int(html['total'])

        return total

    # 获取所有电影的名字和对应type值
    def get_all_type_films(self):
        # 获取 类型和类型码
        url = 'https://movie.douban.com/chart'
        headers = self.get_headers()
        html = requests.get(url=url,headers=headers).text
        re_bds = r'<a href=.*?type_name=(.*?)&type=(.*?)&.*?</a>'
        pattern = re.compile(re_bds,re.S)
        r_list = pattern.findall(html)
        # 存放所有类型和对应类型码大字典
        type_dict = {}
        menu = ''
        for r in r_list:
            type_dict[r[0].strip()] = r[1].strip()
            # 获取input的菜单，显示所有电影类型
            menu += r[0].strip() + '|'

        return type_dict,menu


    # 主函数
    def main(self):
        # 获取type的值
        type_dict,menu = self.get_all_type_films()
        menu = menu + '\n请做出你的选择:'
        name = input(menu)
```

```
        type_number = type_dict[name]
        # 获取电影总数
        total = self.total_number(type_number)
        for start in range(0,(total+1),20):
            params = {
                'type' : type_number,
                'interval_id' : '100:90',
                'action' : '',
                'start' : str(start),
                'limit' : '20'
            }
            # 调用函数,传递params参数
            self.get_page(params)
            # 随机休眠1-3秒
            time.sleep(random.randint(1,3))
        print('电影数量:',self.i)

if __name__ == '__main__':
    spider = DoubanSpider()
    spider.main()
```

## 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
```