Day03回顾

目前反爬总结

- 基于User-Agent反爬

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

- 响应内容前端做处理反爬

```
1、html页面中可匹配出内容，程序中匹配结果为空
    * 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
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
```

- 基于IP反爬

```
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

```
# 编码
params = {
    '':'',
    '':''
}
params = urllib.parse.urlencode(params)
url = baseurl + params

# 请求
request = urllib.request.Request(url,headers=headers)
response = urllib.request.urlopen(request)
html = response.read().decode('utf-8')
```

- requests模块使用流程

```
# 方法一 ： 先使用urllib.parse编码，然后使用requests发请求
url = baseurl + urllib.parse.urlencode({dict})
html = requests.get(url,headers=headers).text

# 方法二 ： 利用params参数(自动对查询参数编码再拼接)
html = requests.get(baseurl,paramas=params,headers=headers).text
```

解析模块总结

- 正则解析re模块

```
import re

pattern = re.compile('正则表达式',re.S)
r_list = pattern.findall(html)
```

- lxml解析库

```
from lxml import etree

parse_html = etree.HTML(res.text)
r_list = parse_html.xpath('xpath表达式')
```

xpath表达式

- 匹配规则

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

- xpath高级

```
1、基准xpath表达式：得到节点对象列表
2、for r in [节点对象列表]:
        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

```
字典,字典中键值对作为查询参数
```

- 使用方法

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

- 示例

```
import requests

baseurl = 'http://tieba.baidu.com/f?'
params = {
  'kw' : '校花吧',
  'pn' : '50'
}
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)'}
# 自动对params进行编码,然后自动和url进行拼接,去发请求
res = requests.get(baseurl,params=params,headers=headers)
res.encoding = 'utf-8'
print(res.text)
```

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

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

```
import requests

url = 'https://httpbin.org/get'
headers = {
    'User-Agent':'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/74.0.3729.169 Safari/537.36'
}
# 定义代理,在代理IP网站中查找免费代理IP
proxies = {
    'http':'http://127.0.0.1:8888',
    'https':'https://127.0.0.1:8888'
}
html = requests.get(url,proxies=proxies,headers=headers,verify=False).text
print(html)
```

2、写一个获取开放代理的接口

```
# getip.py
# 获取开放代理的接口
import requests

# 提取代理IP
def get_ip_list():
  api_url = 'http://dev.kdlapi.com/api/getproxy/?orderid=996140620552954&num=100&protocol=2&method=2&an_an=1&an_ha=1&sep=1'
  res = requests.get(api_url)
  ip_port_list = res.text.split('\r\n')

```

```
  return ip_port_list

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

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

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

```
from getip import *
import time
import random

url = 'http://httpbin.org/get'
headers = {'User-Agent' : 'Mozilla/5.0'}
proxy_ip_list = get_ip_list()

while True:
    # 判断是否还有可用代理
    if not proxy_ip_list:
        proxy_ip_list = get_ip_list()

    proxy_ip = random.choice(proxy_ip_list)
    proxies = {
        'http' : 'http://{}'.format(proxy_ip),
        'https' : 'https://{}'.format(proxy_ip)
    }
    print(proxies)

    try:
        html = requests.get(url=url,proxies=proxies,headers=headers,timeout=5,verify=False).text
        print(html)
        break
    except:
        print('正在更换代理IP，请稍后... ...')
        # 及时把不可用的代理IP移除
        proxy_ip_list.remove(proxy_ip)
        continue
```

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

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


# 生成随机的User-Agent
def get_random_ua():
```

```python
    # 创建User-Agent对象
    ua = UserAgent()
    # 随机生成1个User-Agent
    return ua.random


# 请求头
headers = {
    'User-Agent': get_random_ua()
}
url = 'http://httpbin.org/get'


# 从西刺代理网站上获取随机的代理IP
def get_ip_list():
    # 访问西刺代理网站国内高匿代理，找到所有的tr节点对象
    html = requests.get('https://www.xicidaili.com/nn/', headers=headers).text
    parse_html = etree.HTML(html)
    # 基准xpath，匹配每个代理IP的节点对象列表
    ipobj_list = parse_html.xpath('//tr')
    # 定义空列表创建代理池，获取网页中所有代理IP地址及端口号
    ip_list = []
    # 从列表中第2个元素开始遍历，因为第1个为：字段名（国家、IP、... ...）
    for ip in ipobj_list[1:]:
        ip_info = ip.xpath('./td[2]/text()')[0]
        port_info = ip.xpath('./td[3]/text()')[0]
        ip_list.append(
            {
                'http': 'http://' + ip_info + ':' + port_info,
                'https': 'https://' + ip_info + ':' + port_info
            }
        )
    print(ip_list)
    # 随机选择一个代理
    proxies = random.choice(ip_list)
    print(proxies)
    # 返回代理IP及代理池（列表ip_list）
    return ip_list


# 主程序
def main_print():
    # 我的IP代理池
    ip_list = get_ip_list()
    while True:
        if not ip_list:
            ip_list = get_ip_list()
        try:
            # 设置超时时间，如果代理不能使用则切换下一个
            proxies = random.choice(ip_list)
            res = requests.get(url=url, headers=headers, proxies=proxies, timeout=5)
            res.encoding = 'utf-8'
            print(res.text)
            break

        except Exception as e:
            # 此代理IP不能使用，从代理池中移除
```

```
            ip_list.remove(proxies)
            print('%s不能用，已经移除' % proxies)
            # 继续循环获取下一个代理IP
            continue


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

- 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@122.114.67.136:16819',
    'https':'https://309435365:szayclhp@122.114.67.136:16819',
}
headers = {
    'User-Agent' : 'Mozilla/5.0',
}

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

Web客户端验证 参数-auth

- 作用及类型

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

- 达内code课程方向案例

```
import requests
import re
```

```python

class NoteSpider(object):
    def __init__(self):
        self.url = 'http://code.tarena.com.cn/'
        self.headers = {'User-Agent':'Mozilla/5.0'}
        self.auth = ('tarenacode','code_2013')

    # 获取+解析
    def get_parse_page(self):
        res = requests.get(
            url=self.url,
            auth=self.auth,
            headers=self.headers
        )
        res.encoding = 'utf-8'
        html = res.text
        # 解析
        p = re.compile('<a href=.*?>(.*?)/</a>',re.S)
        r_list = p.findall(html)
        # r_list : ['..','AIDCode','ACCCode']
        for r in r_list:
            if r != '..':
                print({ '课程方向' : r })

if __name__ == '__main__':
    spider = NoteSpider()
    spider.get_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、verify=True(默认)   ：检查证书认证
2、verify=False (常用) ：忽略证书认证
# 示例
response = requests.get(
    url=url,
    params=params,
    proxies=proxies,
    headers=headers,
    verify=False
)
```

requests.post()

- 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)

1. 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
```

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:

```
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:  r+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 + "@6f#X3=cCuncYssPsuRUE")
python实现:
from hashlib import md5
s = md5()
s.update("fanyideskweb" + e + salt + "@6f#X3=cCuncYssPsuRUE".encode())
sign = s.hexdigest()
```

- 5、代码实现

```
import requests
import time
```

```python
from hashlib import md5
import random

# 获取相关加密算法的结果
def get_salt_sign_ts(word):
    # salt
    salt = str(int(time.time()*1000)) + str(random.randint(0,9))
    # sign
    string = "fanyideskweb" + word + salt + "@6f#X3=cCuncYssPsuRUE"
    s = md5()
    s.update(string.encode())
    sign = s.hexdigest()
    # ts
    ts = str(int(time.time()*1000))
    return salt,sign,ts

# 攻克有道
def attack_yd(word):
    salt,sign,ts = get_salt_sign_ts(word)
    # url为抓包抓到的地址 F12 -> translate_o -> post
    url = 'http://fanyi.youdao.com/translate_o?smartresult=dict&smartresult=rule'
    headers = {
        "Accept": "application/json, text/javascript, */*; q=0.01",
        "Accept-Encoding": "gzip, deflate",
        "Accept-Language": "zh-CN,zh;q=0.9",
        "Connection": "keep-alive",
        "Content-Length": "238",
        "Content-Type": "application/x-www-form-urlencoded; charset=UTF-8",
        "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",
        "Host": "fanyi.youdao.com",
        "Origin": "http://fanyi.youdao.com",
        "Referer": "http://fanyi.youdao.com/",
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/74.0.3729.169 Safari/537.36",
        "X-Requested-With": "XMLHttpRequest",
    }
    # Form表单数据
    data = {
        'i': word,
        'from': 'AUTO',
        'to': 'AUTO',
        'smartresult': 'dict',
        'client': 'fanyideskweb',
        'salt': salt,
        'sign': sign,
        'ts': ts,
        'bv': 'cf156b581152bd0b259b90070b1120e6',
        'doctype': 'json',
        'version': '2.1',
        'keyfrom': 'fanyi.web',
        'action': 'FY_BY_REALTlME'
    }

    json_html = requests.post(url,data=data,headers=headers).json()
    result = json_html['translateResult'][0][0]['tgt']
```

```
    return result

if __name__ == '__main__':
    word = input('请输入要翻译的单词: ')
    result = attack_yd(word)
    print(result)
```

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
# 控制台中 XHR : 异步加载的数据包
# 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)
# 抓取的查询参数如下:
type: 13
interval_id: 100:90
action: ''
start: 0
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
# 示例
html = json.loads(res.text)
print(type(html))
```

- 代码实现

```
import requests
```

```python
import json
import pymysql

class DoubanSpider(object):
    def __init__(self):
        self.url = 'https://movie.douban.com/j/chart/top_list?'
        self.headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.119 Safari/537.36'}

    # 获取页面
    def get_page(self,params):
        res = requests.get(
            url=self.url,
            params=params,
            headers=self.headers,
            verify=True
        )
        res.encoding = 'utf-8'
        # json.loads() josn格式->Python格式
        html = res.json()
        self.parse_page(html)

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

    # 主函数
    def main(self):
        limit = input('请输入电影数量:')
        params = {
            'type' : '24',
            'interval_id' : '100:90',
            'action' : '',
            'start' : '0',
            'limit' : limit
        }
        # 调用函数,传递params参数
        self.get_page(params)

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

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