Day04

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模块每次访问随机生成User-Agent
    # sudo pip3 install fake_useraget

    * from fake_useragent import UserAgent
    * ua = UserAgent()
    * user_agent = ua.random
    * print(user_agent)
```

- **响应内容前端JS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
   * 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
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**

```
# 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**

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

- **响应对象res属性**

```
res.text ：字符串
res.content ：bytes
res.encoding：字符编码 res.encoding='utf-8'
res.status_code ：HTTP响应码
res.url ：实际数据URL地址
```

解析模块总结

- **正则解析re模块**

```
import re

pattern = re.compile(r'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
```

写程序注意

```
# 最终目标：不要使你的程序因为任何异常而终止
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
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,可使用列表推导式
# 多级页面数据抓取注意
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)
```

增量爬虫如何实现

```
1、数据库中创建指纹表,用来存储每个请求的指纹
2、在抓取之前,先到指纹表中确认是否之前抓取过
```

Chrome浏览器安装插件

- **安装方法**

```
# 在线安装
1、下载插件 - google访问助手
2、安装插件 - google访问助手：Chrome浏览器-设置-更多工具-扩展程序-开发者模式-拖拽(解压后的插件)
3、在线安装其他插件 - 打开google访问助手 - google应用商店 - 搜索插件 - 添加即可

# 离线安装
1、下载插件 - xxx.crx 重命名为 xxx.zip
2、输入地址：chrome://extensions/   打开- 开发者模式
3、拖拽 插件(或者解压后文件夹) 到浏览器中
4、重启浏览器，使插件生效
```

Day04笔记

链家二手房案例（xpath）

实现步骤

- 确定是否为静态

```
打开二手房页面 -> 查看网页源码 -> 搜索关键字
```

- xpath表达式

```
1、基准xpath表达式(匹配每个房源信息节点列表)
    此处滚动鼠标滑轮时,li节点的class属性值会发生变化,通过查看网页源码确定xpath表达式
  //ul[@class="sellListContent"]/li[@class="clear LOGVIEWDATA LOGCLICKDATA"]

2、依次遍历后每个房源信息xpath表达式
    * 名称: './/a[@data-el="region"]/text()'

    # 户型+面积+方位+是否精装
    info_list = './/div[@class="houseInfo"]/text()'  [0].strip().split('|')
    * 户型: info_list[1]
    * 面积: info_list[2]
    * 方位: info_list[3]
    * 精装: info_list[4]


    * 楼层: './/div[@class="positionInfo"]/text()'
    * 区域: './/div[@class="positionInfo"]/a/text()'
    * 总价: './/div[@class="totalPrice"]/span/text()'
    * 单价: './/div[@class="unitPrice"]/span/text()'
```

代码实现

```python
import requests
from lxml import etree
import time
import random
from useragents import ua_list

class LianjiaSpider(object):
  def __init__(self):
    self.url='https://bj.lianjia.com/ershoufang/pg{}/'
    self.blog = 1

  def get_html(self,url):
    headers = {'User-Agent':random.choice(ua_list)}
    # 尝试3次,否则换下一页地址
    if self.blog <= 3:
      try:
        res = requests.get(url=url,headers=headers,timeout=5)
```

```
        res.encoding = 'utf-8'
        html = res.text
        # 直接调用解析函数
        self.parse_page(html)
      except Exception as e:
        print('Retry')
        self.blog += 1
        self.get_html(url)


  def parse_page(self,html):
    parse_html = etree.HTML(html)
    # li_list: [<element li at xxx>,<element li at xxx>]
    li_list = parse_html.xpath('//ul[@class="sellListContent"]/li[@class="clear LOGVIEWDATA LOGCLICKDATA"]')
    item = {}
    for li in li_list:
      # 名称
      xpath_name = './/a[@data-el="region"]/text()'
      name_list = li.xpath(xpath_name)
      item['name'] = [
        name_list[0].strip() if name_list else None
      ][0]
      # 户型+面积+方位+是否精装
      info_xpath = './/div[@class="houseInfo"]/text()'
      info_list = li.xpath(info_xpath)
      if info_list:
        info_list = info_list[0].strip().split('|')
        if len(info_list) == 5:
          item['model'] = info_list[1].strip()
          item['area'] = info_list[2].strip()
          item['direction'] = info_list[3].strip()
          item['perfect'] = info_list[4].strip()
        else:
          item['model']=item['area']=item['direction']=item['perfect']=None
      else:
        item['model'] = item['area'] = item['direction'] = item['perfect'] = None

      # 楼层
      xpath_floor = './/div[@class="positionInfo"]/text()'
      floor_list = li.xpath(xpath_floor)
      item['floor'] = [
        floor_list[0].strip().split()[0] if floor_list else None
      ][0]

      # 地区
      xpath_address = './/div[@class="positionInfo"]/a/text()'
      address_list = li.xpath(xpath_address)
      item['address'] = [
        address_list[0].strip() if address_list else None
      ][0]
      # 总价
      xpath_total = './/div[@class="totalPrice"]/span/text()'
      total_list = li.xpath(xpath_total)
      item['total_price'] = [
        total_list[0].strip() if total_list else None
      ][0]
```

```
        # 单价
        xpath_unit = './/div[@class="unitPrice"]/span/text()'
        unit_list = li.xpath(xpath_unit)
        item['unit_price'] = [
          unit_list[0].strip() if unit_list else None
        ][0]

        print(item)

    def main(self):
      for pg in range(1,101):
        url = self.url.format(pg)
        self.get_html(url)
        time.sleep(random.randint(1,3))
        # 对self.blog进行一下初始化
        self.blog = 1


if __name__ == '__main__':
    start = time.time()
    spider = LianjiaSpider()
    spider.main()
    end = time.time()
    print('执行时间:%.2f' % (end-start))
```

百度贴吧图片抓取

目标思路

- 目标

```
抓取指定贴吧所有图片
```

- 思路

```
1、获取贴吧主页URL,下一页,找到不同页的URL规律
2、获取1页中所有帖子URL地址: [帖子链接1,帖子链接2,...]
3、对每个帖子链接发请求,获取图片URL
4、向图片的URL发请求,以wb方式写入本地文件
```

实现步骤

- 贴吧URL规律

```
http://tieba.baidu.com/f?kw=??&pn=50
```

- xpath表达式

```
1、帖子链接xpath
   //div[@class="t_con cleafix"]/div/div/div/a/@href

2、图片链接xpath
   //div[@class="d_post_content j_d_post_content  clearfix"]/img[@class="BDE_Image"]/@src

3、视频链接xpath
   //div[@class="video_src_wrapper"]/embed/@data-video
   # 注意：此处视频链接前端对响应内容做了处理,需要查看网页源代码来查看，复制HTML代码在线格式化
```

代码实现

```
import requests
from lxml import etree
import random
import time
from useragents import ua_list
from urllib import parse
import os

class BaiduImageSpider(object):
  def __init__(self):
    self.url = 'http://tieba.baidu.com/f?kw={}&pn={}'

  # 获取html功能函数
  def get_html(self,url):
    html = requests.get(
      url=url,
      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)'}
    ).content.decode('utf-8','ignore')
    return html

  # 解析html功能函数
  def xpath_func(self,html,xpath_bds):
    parse_html = etree.HTML(html)
    r_list = parse_html.xpath(xpath_bds)
    return r_list

  # 解析函数 - 实现最终图片抓取
  def parse_html(self,one_url):
    html = self.get_html(one_url)
    # 准备提取帖子链接:xpath_list ['/p/32323','','']
    xpath_bds = '//div[@class="t_con cleafix"]/div/div/div/a/@href'
    r_list = self.xpath_func(html,xpath_bds)
    for r in r_list:
      # 拼接帖子的URL地址
      t_url = 'http://tieba.baidu.com' + r
      # 把帖子中所有图片保存到本地
      self.get_image(t_url)
      # 爬完1个帖子中所有图片,休眠0-2秒钟
      time.sleep(random.uniform(0,2))

  # 功能:给定1个帖子URL,把帖子中所有图片保存到本地
```

```
  def get_image(self,t_url):
    html = self.get_html(t_url)
    # 图片链接的xpath表达式:img_list ['http://xxx.jpg','']
    # 使用xpath表达式的或| : 图片链接 + 视频链接
    xpath_bds = '//div[@class="d_post_content j_d_post_content clearfix"]/img[@class="BDE_Image"]/@src | //div[@class="video_src_wrapper"]/embed/@data-video'
    img_list = self.xpath_func(html,xpath_bds)
    for img in img_list:
      html_bytes = requests.get(
        url=img,
        headers={'User-Agent':random.choice(ua_list)}
      ).content
      self.save_img(html_bytes,img)

  # 保存图片函数
  def save_img(self,html_bytes,img):
    filename = img[-10:]
    with open(filename,'wb') as f:
      f.write(html_bytes)
      print('%s下载成功' % filename)

  # 主函数
  def main(self):
    name = input('请输入贴吧名:')
    begin = int(input('请输入起始页:'))
    end = int(input('请输入终止页:'))

    # 对贴吧名进行编码
    kw = parse.quote(name)
    for page in range(begin,end+1):
      pn = (page-1)*50
      url = self.url.format(kw,pn)
      # 调用主线函数
      self.parse_html(url)

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

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' : '赵丽颖吧',
  'pn' : '50'
}
headers = {'User-Agent' : 'Mozilla/4.0'}
# 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(url=baseurl,params=params,headers=headers)
res.encoding = 'utf-8'
print(res.text)
```

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**

```
# xpath表达式
//a/@href
# url
http://code.tarena.com.cn/AIDCode/aid1904/14-redis/
```

思考：爬取具体的笔记文件?

```
import os

# 保存在: /home/tarena/redis
# 先判断 /home/tarena/redis 是否存在
  1、不存在: 先创建目录,然后再保存 .zip
  2、存在:  直接保存 .zip

# 使用频率很高
if not os.path.exists('路径'):
      os.makedirs('路径')
```

代码实现

```
import requests
from lxml import etree
import random
```

```python
from useragents import ua_list
import os

class CodeSpider(object):
  def __init__(self):
    self.url = 'http://code.tarena.com.cn/AIDCode/aid1904/14-redis/'
    self.auth = ('tarenacode','code_2013')

  def get_headers(self):
      headers = {'User-Agent':random.choice(ua_list)}
      return headers

  def get_html(self,url):
      res = requests.get(url,headers=self.get_headers(),auth=self.auth)
      html = res.content
      return html

  def parse_html(self):
    # 获取响应内容
    html = self.get_html(self.url).decode()
    # 解析
    parse_html = etree.HTML(html)
    # r_list: ['../','day01/','redis-xxx.zip']
    r_list = parse_html.xpath('//a/@href')

    directory = '/home/tarena/myredis/'
    if not os.path.exists(directory):
      os.makedirs(directory)

    for r in r_list:
      if r.endswith('.zip') or r.endswith('.rar'):
        self.save_files(r,directory)

  def save_files(self,r,directory):
    # 拼接地址,把zip文件保存到指定目录
    url = self.url + r
    # filename: /home/tarena/AID/redis/xxx.zip
    filename = directory + r
    html = self.get_html(url)

    with open(filename,'wb') as f:
      f.write(html)
      print('%s下载成功' % r)


if __name__ == '__main__':
  spider = CodeSpider()
  spider.parse_html()
```

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,
  headers=headers,
  verify=False
)
```

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

```python
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()
    useragent = ua.random
    headers = {'User-Agent':useragent}
    return headers

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

```
# 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
  for ip in ip_port_list:
    with open('proxy_ip.txt','a') as f:
      if test_ip(ip):
            f.write(ip + '\n')
```

```
30
31 if __name__ == '__main__':
32     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

```
import requests
url = 'http://httpbin.org/get'
proxies = {
    'http': 'http://309435365:szayclhp@106.75.71.140:16816',
    'https':'https://309435365:szayclhp@106.75.71.140:16816',
}
headers = {
    'User-Agent' : 'Mozilla/5.0',
}

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

今日作业

```
1、总结前几天内容,理顺知识点
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、Web客户端验证 - 如何下载、保存
```