Spider_Day02回顾

爬取网站思路

```
1、先确定是否为动态加载网站
2、找URL规律
3、正则表达式
4、定义程序框架，补全并测试代码
```

数据持久化 - csv

```
import csv
with open('xxx.csv','w') as f:
    writer = csv.writer(f)
    writer.writerow([])
    writer.writerows([(),(),()])
```

数据持久化 - MySQL

```
import pymysql

# __init__(self):
    self.db = pymysql.connect('IP',... ...)
    self.cursor = self.db.cursor()
# write_data(self):
    self.cursor.execute('sql',[data1])
    self.cursor.executemany('sql',[(data1),(data2),(data3)])
    self.db.commit()
# main(self):
    self.cursor.close()
    self.db.close()
```

数据持久化 - MongoDB

```
import pymongo

```

```
# __init__(self):
    self.conn = pymongo.MongoClient('IP',27017)
    self.db = self.conn['db_name']
    self.myset = self.db['set_name']
# write_data(self):
    self.myset.insert_one(dict)

# MongoDB - Commmand
>show dbs
>use db_name
>show collections
>db.collection_name.find().pretty()
>db.collection_name.count()
>db.collection_name.drop()
>db.dropDatabase()
```

多级页面数据抓取

```
# 整体思路
1、爬取一级页面,提取 所需数据+链接,继续跟进
2、爬取二级页面,提取 所需数据+链接,继续跟进
3、... ...
# 代码实现思路
1、所有数据最终都会在一级页面解析函数中拿到
2、避免重复代码 - 请求、解析需定义函数
```

requests模块

- get()

```
1、发请求并获取响应对象
2、res = requests.get(url,headers=headers)
```

- 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
```

- 非结构化数据保存

```
with open('xxx.jpg','wb') as f:
    f.write(res.content)
```

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
```

- 爬虫常用插件

```
1、google-access-helper : 谷歌访问助手,可访问 谷歌应用商店
2、Xpath Helper: 轻松获取HTML元素的xPath路径
   开启/关闭: Ctrl + Shift + x
3、JsonView: 格式化输出json格式数据
4、Proxy SwitchyOmega: Chrome浏览器中的代理管理扩展程序
```

Spider_Day03笔记

xpath解析

- 定义

```
XPath即为XML路径语言，它是一种用来确定XML文档中某部分位置的语言，同样适用于HTML文档的检索
```

- 示例HTML代码

```
<ul class="CarList">
    <li class="bjd" id="car_001" href="http://www.bjd.com/">
        <p class="name">布加迪</p>
        <p class="model">威航</p>
        <p class="price">2500万</p>
        <p class="color">红色</p>
    </li>

    <li class="byd" id="car_002" href="http://www.byd.com/">
        <p class="name">比亚迪</p>
        <p class="model">秦</p>
        <p class="price">15万</p>
        <p class="color">白色</p>
    </li>
```

```
15 </ul>
```

- 匹配演示

```
1、查找所有的li节点

2、获取所有汽车的名称：所有li节点下的子节点p的值（class属性值为name）

3、找比亚迪车的信息：获取ul节点下第2个li节点的汽车信息

4、获取所有汽车的链接：ul节点下所有li子节点的href属性的值


# 只要涉及到条件,加 []
# 只要获取属性值,加 @
```

- 选取节点

```
1、// ： 从所有节点中查找（包括子节点和后代节点）
2、@  ： 获取属性值
   # 使用场景1（属性值作为条件）
     //div[@class="movie-item-info"]
   # 使用场景2（直接获取属性值）
     //div[@class="movie-item-info"]/a/img/@src
```

- 匹配多路径（或）

```
xpath表达式1 | xpath表达式2 | xpath表达式3
```

- 常用函数

```
1、contains() ： 匹配属性值中包含某些字符串节点
   # 查找id属性值中包含字符串 "car_" 的 li 节点

2、text() ： 获取节点的文本内容
   # 查找所有汽车的价格

```

lxml解析库

- **安装**

```
sudo pip3 install lxml
```

- **使用流程**

```
1、导模块
   from lxml import etree
2、创建解析对象
   parse_html = etree.HTML(html)
3、解析对象调用xpath
   r_list = parse_html.xpath('xpath表达式')
```

- **html样本**

```
<div class="wrapper">
    <a href="/" id="channel">新浪社会</a>
    <ul id="nav">
        <li><a href="http://domestic.sina.com/" title="国内">国内</a></li>
        <li><a href="http://world.sina.com/" title="国际">国际</a></li>
        <li><a href="http://mil.sina.com/" title="军事">军事</a></li>
        <li><a href="http://photo.sina.com/" title="图片">图片</a></li>
        <li><a href="http://society.sina.com/" title="社会">社会</a></li>
        <li><a href="http://ent.sina.com/" title="娱乐">娱乐</a></li>
        <li><a href="http://tech.sina.com/" title="科技">科技</a></li>
        <li><a href="http://sports.sina.com/" title="体育">体育</a></li>
        <li><a href="http://finance.sina.com/" title="财经">财经</a></li>
        <li><a href="http://auto.sina.com/" title="汽车">汽车</a></li>
    </ul>
</div>
```

- **示例+练习**

```
from lxml import etree

html = '''
<div class="wrapper">
    <a href="/" id="channel">新浪社会</a>
    <ul id="nav">
        <li><a href="http://domestic.sina.com/" title="国内">国内</a></li>
        <li><a href="http://world.sina.com/" title="国际">国际</a></li>
        <li><a href="http://mil.sina.com/" title="军事">军事</a></li>
        <li><a href="http://photo.sina.com/" title="图片">图片</a></li>
        <li><a href="http://society.sina.com/" title="社会">社会</a></li>
        <li><a href="http://ent.sina.com/" title="娱乐">娱乐</a></li>
        <li><a href="http://tech.sina.com/" title="科技">科技</a></li>
        <li><a href="http://sports.sina.com/" title="体育">体育</a></li>
        <li><a href="http://finance.sina.com/" title="财经">财经</a></li>
        <li><a href="http://auto.sina.com/" title="汽车">汽车</a></li>
    </ul>
</div>'''
# 创建解析对象
parse_html = etree.HTML(html)
# 调用xpath返回结束,text()为文本内容
r_list = parse_html.xpath('//a/text()')
#print(r_list)

# 提取所有的href的属性值

# 提取所有href的值,不包括 /
```

```
28
29  # 获取 图片、军事、...,不包括新浪社会
30
```

xpath最常使用方法

```
1、先匹配节点对象列表
  # r_list: ['节点对象1','节点对象2']
  r_list = parse_html.xpath('基准xpath表达式')
2、遍历每个节点对象,利用节点对象继续调用 xpath
  for r in r_list:
        name = r.xpath('./xxxxxx')
        star = r.xpath('.//xxxxx')
        time = r.xpath('.//xxxxx')
```

链家二手房案例（xpath）

- 实现步骤

1. 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
```

2. xpath表达式

```
1、基准xpath表达式(匹配每个房源信息节点列表)
  //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()'
```

3. 代码实现

```

```

练习: 将猫眼电影用xpath去实现

- **1、xpath表达式**

```
1、基准xpath：匹配所有电影信息的节点对象列表


2、遍历对象列表，依次获取每个电影信息
   for dd in dd_list:
       电影名称 ：
       电影主演 ：
       上映时间 ：
```

- **2、代码实现**

```

```

3、完整代码实现

```

```

百度贴吧图片抓取

- 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

1. **贴吧URL规律**

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

2. **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
```

3. **代码实现**

```

```

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 (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)
```

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

```

```

思考：爬取具体的笔记文件?

```

```

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

1. 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)
```

2. 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

```

```

今日作业

电影天堂（xpath）

糗事百科（xpath）

```
1、URL地址: https://www.qiushibaike.com/text/
2、目标 ： 用户昵称、段子内容、好笑数量、评论数量
```