

DAY03

Day02回顾

爬取网站思路

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

数据持久化 - csv

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

数据持久化 - MySQL

```
1 import pymysql
2
3 # __init__(self):
4 self.db = pymysql.connect('IP',... ...)
5 self.cursor = self.db.cursor()
6 # write_data(self):
7 self.cursor.execute('sql',[data1])
8 self.cursor.executemany('sql',[(data1),(data2),(data3)])
9 self.db.commit()
10 # main(self):
11 self.cursor.close()
12 self.db.close()
```

数据持久化 - MongoDB

```
1 import pymongo
2
3 # __init__(self):
4 self.conn = pymongo.MongoClient('IP',27017)
5 self.db = self.conn['db_name']
6 self.myset = self.db['set_name']
7
8 # write_data(self):
9 self.myset.insert_one(dict)
10 self.myset.insert_many([{}},{},{})
11
12 # MongoDB - Command
13 >show dbs
14 >use db_name
15 >show collections
16 >db.collection_name.find().pretty()
17 >db.collection_name.count()
18 >db.collection_name.drop()
19 >db.dropDatabase()
```

多级页面数据抓取

```
1 # 整体思路
2 1、爬取一级页面,提取 所需数据+链接,继续跟进
3 2、爬取二级页面,提取 所需数据+链接,继续跟进
4 3、... ...
5 # 代码实现思路
6 1、所有数据最终都会在一级页面遍历每条数据时全部拿到
7 2、避免重复代码 - 请求、解析需定义函数
```

Day03笔记

电影天堂二级页面抓取案例

领取任务

```

1 # 地址
2 电影天堂 - 2019年新片精品 - 更多
3 # 目标
4 电影名称、下载链接
5
6 # 分析
7 *****一级页面需抓取*****
8     1、电影详情页链接
9
10 *****二级页面需抓取*****
11     1、电影名称
12     2、电影下载链接

```

实现步骤

- 1、确定响应内容中是否存在所需抓取数据
- 2、找URL规律

```

1 第1页 : https://www.dytt8.net/html/gndy/dyzz/list_23_1.html
2 第2页 : https://www.dytt8.net/html/gndy/dyzz/list_23_2.html
3 第n页 : https://www.dytt8.net/html/gndy/dyzz/list_23_n.html

```

- 3、写正则表达式

```

1 1、一级页面正则表达式
2 <table width="100%".*?<td width="5%".*?<a href="(.*?)"*.?u?link">.*?</table>
3 2、二级页面正则表达式
4 <div class="title_all"><h1><font color=#07519a>(.*?)</font></h1></div>.*?<td style="WORD-
  WRAP.*?>.*?>(.*?)</a>

```

- 4、把电影天堂数据存入MySQL数据库 - 增量爬取

```

1 # 思路
2 # 1、MySQL中新建表 request_finger,存储所有爬取过的链接的指纹
3 # 2、在爬取之前,先判断该指纹是否爬取过,如果爬取过,则不再继续爬取

```

5、代码实现

```

1 # 建库建表
2 create database filmskydb charset utf8;
3 use filmskydb;
4 create table request_finger(
5     finger char(32)
6 )charset=utf8;
7 create table filmtab(
8     name varchar(200),
9     download varchar(500)
10 )charset=utf8;

```

```
1
```

requests模块

安装

■ Linux

```
1 sudo pip3 install requests
```

■ Windows

```
1 # 方法一
2 进入cmd命令行 : python -m pip install requests
3 # 方法二
4 右键管理员进入cmd命令行 : pip install requests
```

requests.get()

■ 作用

```
1 # 向网站发起请求,并获取响应对象
2 res = requests.get(url,headers=headers)
```

■ 参数

```
1 1、url : 需要抓取的URL地址
2 2、headers : 请求头
3 3、timeout : 超时时间,超过时间会抛出异常
```

■ 响应对象(res)属性

```
1 1、encoding : 响应字符编码
2   res.encoding = 'utf-8'
3 2、text : 字符串
4 3、content : 字节流
5 4、status_code : HTTP响应码
6 5、url : 实际数据的URL地址
```

■ 非结构化数据保存

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

▪ 示例

保存赵丽颖图片到本地

```
1 |
```

▪ 练习

- 1、将猫眼电影案例改写为 requests 模块实现
- 2、将电影天堂案例改写为 requests 模块实现
- 3、百度图片抓取：输入要抓取的图片内容，抓取首页的30张图片，保存到对应的文件夹，比如：
你想要谁的照片，请输入：赵丽颖
- 5 创建文件夹到指定目录：赵丽颖 并把首页30张图片保存到此文件夹下

百度图片练习代码实现

```
1 |
```

Chrome浏览器安装插件

安装方法

- 1、把下载的相关插件（对应操作系统浏览器）后缀改为 .zip
- 2、解压,打开Chrome浏览器 -> 右上角设置 -> 更多工具 -> 扩展程序 -> 点开开发者模式
- 3 #3、把相关插件文件夹拖拽到浏览器中,释放鼠标即可安装
- 4 #3、有的插件直接拖拽 .zip 文件释放即可

需要安装插件

- 1、Xpath Helper：轻松获取HTML元素的XPath路径
- 2 # 开启/关闭：Ctrl + Shift + x
- 2、Proxy SwitchyOmega：Chrome浏览器中的代理管理扩展程序
- 4 3、JsonView：格式化输出json格式数据

xpath解析

定义

- 1 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

```
1 <ul class="CarList">
2   <li class="bjd" id="car_001" href="http://www.bjd.com/">
3     <p class="name">布加迪</p>
4     <p class="model">威航</p>
5     <p class="price">2500万</p>
6     <p class="color">红色</p>
7   </li>
8
9   <li class="byd" id="car_002" href="http://www.byd.com/">
10    <p class="name">比亚迪</p>
11    <p class="model">秦</p>
12    <p class="price">15万</p>
13    <p class="color">白色</p>
14  </li>
15 </ul>
```

匹配演示

```
1 1、查找所有的li节点
2   //li
3 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）
4   //li/p[@class="name"]
5 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
6   //ul/li[2]
7 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
8   //ul/li/@href
9
10 # 只要涉及到条件,加 []
11 # 只要获取属性值,加 @
```

选取节点

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

匹配多路径（或）

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

常用函数

```
1 | 1、contains()：匹配属性值中包含某些字符串节点
2 |   # 查找id属性值中包含字符串 "car_" 的 li 节点
3 |
4 | 2、text()：获取节点的文本内容
5 |   # 查找所有汽车的价格
6 |
```

lxml解析库

安装

```
1 | sudo pip3 install lxml
```

使用流程

```
1 | 1、导模块
2 |   from lxml import etree
3 | 2、创建解析对象
4 |   parse_html = etree.HTML(html)
5 | 3、解析对象调用xpath
6 |   r_list = parse_html.xpath('xpath表达式')
```

html样本

```
1 | <div class="wrapper">
2 |   <a href="/" id="channel">新浪社会</a>
3 |   <ul id="nav">
4 |     <li><a href="http://domestic.sina.com/" title="国内">国内</a></li>
5 |     <li><a href="http://world.sina.com/" title="国际">国际</a></li>
6 |     <li><a href="http://mil.sina.com/" title="军事">军事</a></li>
7 |     <li><a href="http://photo.sina.com/" title="图片">图片</a></li>
8 |     <li><a href="http://society.sina.com/" title="社会">社会</a></li>
9 |     <li><a href="http://ent.sina.com/" title="娱乐">娱乐</a></li>
10 |    <li><a href="http://tech.sina.com/" title="科技">科技</a></li>
11 |    <li><a href="http://sports.sina.com/" title="体育">体育</a></li>
```

```

12     <li><a href="http://finance.sina.com/" title="财经">财经</a></li>
13     <li><a href="http://auto.sina.com/" title="汽车">汽车</a></li>
14 </ul>
15 </div>

```

示例+练习

```

1 from lxml import etree
2
3 html = '''
4 <div class="wrapper">
5     <a href="/" id="channel">新浪社会</a>
6     <ul id="nav">
7         <li><a href="http://domestic.sina.com/" title="国内">国内</a></li>
8         <li><a href="http://world.sina.com/" title="国际">国际</a></li>
9         <li><a href="http://mil.sina.com/" title="军事">军事</a></li>
10        <li><a href="http://photo.sina.com/" title="图片">图片</a></li>
11        <li><a href="http://society.sina.com/" title="社会">社会</a></li>
12        <li><a href="http://ent.sina.com/" title="娱乐">娱乐</a></li>
13        <li><a href="http://tech.sina.com/" title="科技">科技</a></li>
14        <li><a href="http://sports.sina.com/" title="体育">体育</a></li>
15        <li><a href="http://finance.sina.com/" title="财经">财经</a></li>
16        <li><a href="http://auto.sina.com/" title="汽车">汽车</a></li>
17    </ul>
18 </div>'''
19 # 创建解析对象
20 parse_html = etree.HTML(html)
21 # 调用xpath返回结果,text()为文本内容
22 a_list = parse_html.xpath('//a/text()')
23 print(a_list)
24
25 # 提取所有的href的属性值
26
27 # 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 1、先匹配节点对象列表
2     # r_list: ['节点对象1','节点对象2']
3     r_list = parse_html.xpath('基准xpath表达式')
4 2、遍历每个节点对象,利用节点对象继续调用 xpath
5     for r in r_list:
6         name = r.xpath('./xxxxxx')
7         star = r.xpath('./xxxxxx')
8         time = r.xpath('./xxxxxx')

```


链家二手房案例 (xpath)

实现步骤

- 确定是否为静态

```
1 | 打开二手房页面 -> 查看网页源码 -> 搜索关键字
```

- xpath表达式

```
1 | 1、基准xpath表达式(匹配每个房源信息节点列表)
2 | 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
3 | //ul[@class="sellListContent"]/li[@class="clear LOGVIEWDATA LOGCLICKDATA"]
4 |
5 | 2、依次遍历后每个房源信息xpath表达式
6 | * 名称: '//*[@data-el="region"]/text()'
7 |
8 | # 户型+面积+方位+是否精装
9 | info_list = '//*[@class="houseInfo"]/text()' [0].strip().split('|')
10 | * 户型: info_list[1]
11 | * 面积: info_list[2]
12 | * 方位: info_list[3]
13 | * 精装: info_list[4]
14 |
15 |
16 | * 楼层: '//*[@class="positionInfo"]/text()'
17 | * 区域: '//*[@class="positionInfo"]/a/text()'
18 | * 总价: '//*[@class="totalPrice"]/span/text()'
19 | * 单价: '//*[@class="unitPrice"]/span/text()'
```

代码实现

```
1 |
```

作业1 - 猫眼电影数据抓取

实现分析

```
1 | 1、基准xpath: 匹配所有电影信息的节点对象列表
2 |
3 |
4 | 2、遍历对象列表, 依次获取每个电影信息
5 | for dd in dd_list:
6 |     电影名称 :
7 |     电影主演 :
8 |     上映时间 :
```

作业2 - 百度贴吧图片抓取

目标思路

■ 目标

```
1  抓取指定贴吧所有图片
```

■ 思路

```
1  1、获取贴吧主页URL, 下一页, 找到不同页的URL规律
2  2、获取1页中所有帖子URL地址: [帖子链接1, 帖子链接2, ...]
3  3、对每个帖子链接发请求, 获取图片URL
4  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

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

■ xpath表达式

```
1  1、帖子链接xpath
2  //div[@class="t_con clearfix"]/div/div/div/a/@href
3
4  2、图片链接xpath
5  //div[@class="d_post_content j_d_post_content clearfix"]/img[@class="BDE_Image"]/@src
6
7  3、视频链接xpath
8  //div[@class="video_src_wrapper"]/embed/@data-video
9  # 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
```

作业3 - 电影天堂 (xpath)