DAY03

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)
  self.myset.insert_many([{},{},{}])

# 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遍历每条数据时全部拿到
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

```
# 地址
电影天堂 - 2019年新片精品 - 更多
# 目标
电影名称、下载链接

# 分析
*********一级页面需抓取**********
        1、电影详情页链接

*********二级页面需抓取**********
        1、电影名称
        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**

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

- **3、写正则表达式**

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

- **4、代码实现**

```
from urllib import request
import re
from useragents import ua_list
import time
import random

class FilmSkySpider(object):
  def __init__(self):
    # 一级页面url地址
    self.url = 'https://www.dytt8.net/html/gndy/dyzz/list_23_{}.html'

  # 获取html功能函数
  def get_html(self,url):
    headers = {
      'User-Agent':random.choice(ua_list)
    }
    req = request.Request(url=url,headers=headers)
    res = request.urlopen(req)
```

```
        # 通过网站查看网页源码,查看网站charset='gb2312'
        # 如果遇到解码错误,识别不了一些字符,则 ignore 忽略掉
        html = res.read().decode('gb2312','ignore')

        return html

    # 正则解析功能函数
    def re_func(self,re_bds,html):
        pattern = re.compile(re_bds,re.S)
        r_list = pattern.findall(html)

        return r_list

    # 获取数据函数 - html是一级页面响应内容
    def parse_page(self,one_url):
        html = self.get_html(one_url)
        re_bds = r'<table width="100%".*?<td width="5%".*?<a href="(.*?)".*?ulink">.*?</table>'
        # one_page_list: ['/html/xxx','/html/xxx','/html/xxx']
        one_page_list = self.re_func(re_bds,html)

        for href in one_page_list:
            two_url = 'https://www.dytt8.net' + href
            self.parse_two_page(two_url)
            # uniform: 浮点数,爬取1个电影信息后sleep
            time.sleep(random.uniform(1, 3))


    # 解析二级页面数据
    def parse_two_page(self,two_url):
        item = {}
        html = self.get_html(two_url)
        re_bds = r'<div class="title_all"><h1><font color=#07519a>(.*?)</font></h1></div>.*?<td style="WORD-WRAP.*?>.*?>(.*?)</a>'
        # two_page_list: [('名称1','ftp://xxxx.mkv')]
        two_page_list = self.re_func(re_bds,html)

        item['name'] = two_page_list[0][0].strip()
        item['download'] = two_page_list[0][1].strip()

        print(item)


    def main(self):
        for page in range(1,201):
            one_url = self.url.format(page)
            self.parse_page(one_url)
            # uniform: 浮点数
            time.sleep(random.uniform(1,3))

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

- **5、练习**

  把电影天堂数据存入MySQL数据库 - 增量爬取

```
# 思路
# 1、MySQL中新建表 request_finger,存储所有爬取过的链接的指纹
# 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
```

练习代码实现

```
# 建库建表
create database filmskydb charset utf8;
use filmskydb;
create table request_finger(
finger char(32)
)charset=utf8;
create table filmtab(
name varchar(200),
download varchar(500)
)charset=utf8;
```

```
from urllib import request
import re
from useragents import ua_list
import time
import random
import pymysql
from hashlib import md5


class FilmSkySpider(object):
    def __init__(self):
        # 一级页面url地址
        self.url = 'https://www.dytt8.net/html/gndy/dyzz/list_23_{}.html'
        self.db = pymysql.connect('192.168.153.151', 'tiger', '123456', 'filmskydb',
charset='utf8')
        self.cursor = self.db.cursor()

    # 获取html功能函数
    def get_html(self, url):
        headers = {
            'User-Agent': random.choice(ua_list)
        }
        req = request.Request(url=url, headers=headers)
        res = request.urlopen(req)
        # 通过网站查看网页源码,查看网站charset='gb2312'
        # 如果遇到解码错误,识别不了一些字符,则 ignore 忽略掉
        html = res.read().decode('gb2312', 'ignore')

        return html

    # 正则解析功能函数
    def re_func(self, re_bds, html):
        pattern = re.compile(re_bds, re.S)
        r_list = pattern.findall(html)

        return r_list

```

```
    # 获取数据函数
    def parse_page(self, one_url):
        html = self.get_html(one_url)
        re_bds = r'<table width="100%".*?<td width="5%".*?<a href="(.*?)".*?ulink">.*?</table>'
        # one_page_list: ['/html/xxx','/html/xxx','/html/xxx']
        one_page_list = self.re_func(re_bds, html)

        for href in one_page_list:
            two_url = 'https://www.dytt8.net' + href
            # 生成指纹 - md5加密
            s = md5()
            s.update(two_url.encode())
            two_url_md5 = s.hexdigest()
            # 判断链接是否需要抓取
            if self.is_go_on(two_url_md5):
                self.parse_two_page(two_url)
                # 爬取完成此链接后将指纹放到数据库表中
                ins = 'insert into request_finger values(%s)'
                self.cursor.execute(ins, [two_url_md5])
                self.db.commit()
                # uniform: 浮点数,爬取1个电影信息后sleep
                time.sleep(random.uniform(1, 3))


    def is_go_on(self, two_url_md5):
        # 爬取之前先到数据库中查询比对
        sel = 'select finger from request_finger where finger=%s'
        # 开始抓取之前,先来判断该链接之前是否抓取过
        result = self.cursor.execute(sel, [two_url_md5])
        if not result:
            return True


    # 解析二级页面数据
    def parse_two_page(self, two_url):
        item = {}
        html = self.get_html(two_url)
        re_bds = r'<div class="title_all"><h1><font color=#07519a>(.*?)</font></h1></div>.*?<td style="WORD-WRAP.*?>.*?>(.*?)</a>'
        # two_page_list: [('名称1','ftp://xxxx.mkv')]
        two_page_list = self.re_func(re_bds, html)

        item['name'] = two_page_list[0][0].strip()
        item['download'] = two_page_list[0][1].strip()

        ins = 'insert into filmtab values(%s,%s)'
        film_list = [
            item['name'], item['download']
        ]
        self.cursor.execute(ins, film_list)
        self.db.commit()
        print(film_list)


    def main(self):
        for page in range(1, 201):
```

```
            one_url = self.url.format(page)
            self.parse_page(one_url)
            # uniform: 浮点数
            time.sleep(random.uniform(1, 3))


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

requests模块

安装

- **Linux**

```
sudo pip3 install requests
```

- **Windows**

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

requests.get()

- **作用**

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

- **参数**

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

- **响应对象(res)属性**

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

- **非结构化数据保存**

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

- **示例**

保存赵丽颖图片到本地

```
import requests

url = 'https://timgsa.baidu.com/timg?image&quality=80&size=b9999_10000&sec=1567090051520&di=77e8b97b3280f999cf51340af4315b4b&imgtype=jpg&src=http%3A%2F%2F5b0988e595225.cdn.sohucs.com%2Fimages%2F20171121%2F4e6759d153d04c6badbb0a5262ec103d.jpeg'
headers = {'User-Agent':'Mozilla/5.0'}

html = requests.get(url=url,headers=headers).content
with open('花千骨.jpg','wb') as f:
    f.write(html)
```

- **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
   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

```
import requests
import re
from urllib import parse
import os

class BaiduImageSpider(object):
    def __init__(self):
        self.url = 'https://image.baidu.com/search/index?tn=baiduimage&word={}'
        self.headers = {'User-Agent':'Mozilla/5.0'}

    # 获取图片
    def get_image(self,url,word):
        html = requests.get(url,headers=self.headers).text
        pattern = re.compile('"hoverURL":"(.*?)"',re.S)
        img_link_list = pattern.findall(html)
```

```
        # 创建目录，准备保存图片
        directory = 'E:\\{}\\'.format(word)
        if not os.path.exists(directory):
            os.makedirs(directory)

        i = 1
        for img_link in img_link_list:
            filename = '{}{}_{}.jpg'.format(directory, word, i)
            self.save_image(img_link,filename)
            i += 1

    def save_image(self,img_link,filename):
        html = requests.get(url=img_link,headers=self.headers).content
        with open(filename,'wb') as f:
            f.write(html)
        print(filename,'下载成功')

    def run(self):
        word = input('你要谁的照片: ')
        word_parse = parse.quote(word)
        url = self.url.format(word)
        self.get_image(url,word)

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

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、把相关插件文件夹拖拽到浏览器中,释放鼠标即可安装
#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
  # 开启/关闭: Ctrl + Shift + x
2、Proxy SwitchyOmega: Chrome浏览器中的代理管理扩展程序
3、JsonView: 格式化输出json格式数据
```

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

```
<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>
</ul>
```

匹配演示

```
1、查找所有的li节点
  //li
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）
  //li/p[@class="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
  //ul/li[2]
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
  //ul/li/@href

# 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
   //li[contains(@id,"car_")]
2、text() ： 获取节点的文本内容
   # 查找所有汽车的价格
   //li/p[@class="price"]/text()
```

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
a_list = parse_html.xpath('//a/text()')
print(a_list)

# 提取所有的href的属性值
href_list = parse_html.xpath('//a/@href')
print(href)
# 提取所有href的值,不包括 /
href_list = parse_html.xpath('//ul[@id="nav"]/li/a/@href')
print(href_list)
# 获取 图片、军事、...,不包括新浪社会
```

```
a_list = parse_html.xpath('//ul[@id="nav"]/li/a/text()')
for a in a_list:
  print(a)
```

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

- 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
```

```
54
55        # 楼层
56        xpath_floor = './/div[@class="positionInfo"]/text()'
57        floor_list = li.xpath(xpath_floor)
58        item['floor'] = [
59          floor_list[0].strip().split()[0] if floor_list else None
60        ][0]
61
62        # 地区
63        xpath_address = './/div[@class="positionInfo"]/a/text()'
64        address_list = li.xpath(xpath_address)
65        item['address'] = [
66          address_list[0].strip() if address_list else None
67        ][0]
68        # 总价
69        xpath_total = './/div[@class="totalPrice"]/span/text()'
70        total_list = li.xpath(xpath_total)
71        item['total_price'] = [
72          total_list[0].strip() if total_list else None
73        ][0]
74        # 单价
75        xpath_unit = './/div[@class="unitPrice"]/span/text()'
76        unit_list = li.xpath(xpath_unit)
77        item['unit_price'] = [
78          unit_list[0].strip() if unit_list else None
79        ][0]
80
81        print(item)
82
83    def main(self):
84      for pg in range(1,101):
85        url = self.url.format(pg)
86        self.get_html(url)
87        time.sleep(random.randint(1,3))
88        # 对self.blog进行一下初始化
89        self.blog = 1
90
91
92  if __name__ == '__main__':
93    start = time.time()
94    spider = LianjiaSpider()
95    spider.main()
96    end = time.time()
97    print('执行时间:%.2f' % (end-start))
```

作业1 - 猫眼电影数据抓取

实现分析

```
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 ：
```

作业2 - 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
```

作业3 - 电影天堂（xpath）

作业4 - 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
```