Day01回顾

请求模块(urllib.request)

```
req = request.Request(url,headers=headers)
res = request.urlopen(req)
html = res.read().decode('utf-8','ignore')
```

编码模块(urllib.parse)

```
1、urlencode({dict})
   urlencode({'wd':'美女','pn':'20'})
   编码后 ： 'wd=%E8%D5XXX&pn=20'

2、quote(string)
   quote('织女')
   编码后 ： '%D3%F5XXX'

3、unquote('%D3%F5XXX')
```

解析模块(re)

使用流程

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

贪婪匹配和非贪婪匹配

```
贪婪匹配(默认) ：  .*
非贪婪匹配     ：  .*?
```

正则表达式分组

```
1、想要什么内容在正则表达式中加()
2、多个分组,先按整体正则匹配,然后再提取()中数据。结果：[(),(),(),(),()]
```

抓取步骤

```
1、确定所抓取数据在响应中是否存在（右键 - 查看网页源码 - 搜索关键字）
2、数据存在：查看URL地址规律
3、写正则表达式,来匹配数据
4、程序结构
  a>每爬取1个页面后随机休眠一段时间
```

```python
# 程序结构
class xxxSpider(object):
    def __init__(self):
        # 定义常用变量,url,headers及计数等

    def get_html(self):
        # 获取响应内容函数,使用随机User-Agent

    def parse_html(self):
        # 使用正则表达式来解析页面，提取数据

    def save_html(self):
        # 将提取的数据按要求保存，csv、MySQL数据库等

    def run(self):
        # 主函数，用来控制整体逻辑

if __name__ == '__main__':
    # 程序开始运行时间戳
    start = time.time()
    spider = xxxSpider()
    spider.main()
    # 程序运行结束时间戳
    end = time.time()
    print('执行时间:%.2f' % (end-start))
```

spider-day02笔记

猫眼电影top100抓取案例

- **爬虫需求**

```
# 1. 确定URL地址
百度搜索 - 猫眼电影 - 榜单 - top100榜

# 2. 爬取目标
电影名称、主演、上映时间
```

- **爬虫实现**

```
# 1. 查看网页源码，确认数据来源
响应内容中存在所需抓取数据 - 电影名称、主演、上映时间

# 2. 翻页寻找URL地址规律
第1页：https://maoyan.com/board/4?offset=0
第2页：https://maoyan.com/board/4?offset=10
第n页：offset=(n-1)*10

# 3. 编写正则表达式
<div class="movie-item-info">.*?title="(.*?)".*?class="star">(.*?)</p>.*?releasetime">(.*?)</p>
```

- **代码实现 - 初始代码**

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


class MaoyanSpider(object):
    def __init__(self):
        self.url = 'https://maoyan.com/board/4?offset={}'
        self.i = 0

    def get_html(self,url):
        headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.163 Safari/535.1',
            'Cookie':'__mta=219092646.1582029073016.1582029104477.1582029107258.4; uuid_n_v=v1; uuid=86F5E1A0524A11EA968A372EE6B0620493363A49A0D9439A84A14933EDB51143; _csrf=5029ad27fba276f31795f6351789d9c6752f33517e8a61236b65d2eca7885032; Hm_lvt_703e94591e87be68cc8da0da7cbd0be2=1582029064; _lx_utm=utm_source%3DBaidu%26utm_medium%3Dorganic; _lxsdk_cuid=1705847c8c1c8-00072237f76da3-3f6b4d00-100200-1705847c8c1c8; _lxsdk=86F5E1A0524A11EA968A372EE6B0620493363A49A0D9439A84A14933EDB51143; mojo-uuid=c8bce593970a7f39b7675f826ca3103b; mojo-session-id={"id":"62ea6d2e0b68f4a245fee15d84e88693","time":1582029073083}; __mta=219092646.1582029073016.1582029102826.1582029104477.3; mojo-trace-id=6; Hm_lpvt_703e94591e87be68cc8da0da7cbd0be2=1582029107; _lxsdk_s=1705847c8c4-e3c-f08-93d%7C%7C10'
        }
        req = request.Request(url=url,headers=headers)
        resp= request.urlopen(req)
        html= resp.read().decode()
        # 直接调用解析函数
        self.parse_html(html)

    def parse_html(self,html):
        """正则解析函数"""
        regex = '<div class="movie-item-info">.*?title="(.*?)".*?class="star">(.*?)</p>.*?releasetime">(.*?)</p>'
        pattern = re.compile(regex,re.S)
        # dd_list: [(),(),()]
        dd_list = pattern.findall(html)
```

```
        self.save_html(dd_list)

    def save_html(self,dd_list):
        """保存数据函数"""
        item = {}
        for dd in dd_list:
            item['name'] = dd[0].strip()
            item['star'] = dd[1].strip()[3:]
            item['time'] = dd[2].strip()[5:15]
            print(item)
            self.i += 1

    def run(self):
        for offset in range(0,91,10):
            url = self.url.format(offset)
            self.get_html(url)
            time.sleep(random.randint(1,3))
        print('电影数量: ',self.i)

if __name__ == '__main__':
    start_time = time.time()
    spider = MaoyanSpider()
    spider.run()
    end_time = time.time()
    print('执行时间:%.2f' % (end_time-start_time))
```

数据持久化 - csv

- **csv描述**

```
【1】作用
   将爬取的数据存放到本地的csv文件中

【2】使用流程
    2.1> 打开csv文件
    2.2> 初始化写入对象
    2.3> 写入数据(参数为列表)

【3】示例代码
    import csv
    with open('sky.csv','w') as f:
        writer = csv.writer(f)
        writer.writerow([])
```

- **示例**

```
【1】题目描述
    创建 test.csv 文件，在文件中写入数据

【2】单行写入 - writerow([])方法
    import csv
    with open('test.csv','w') as f:
```

```
      writer = csv.writer(f)
      writer.writerow(['步惊云','36'])
      writer.writerow(['超哥哥','25'])

【3】多行写入 - writerows([(),(),()]) 方法
   import csv
   with open('test.csv','w') as f:
     writer = csv.writer(f)
     writer.writerows([('聂风','36'),('秦霜','25'),('孔慈','30')])
```

- **练习 - 猫眼电影数据存入本地 maoyanfilm.csv 文件**

```
def save_html(self,r_list):
    with open('maoyanfilm.csv','a') as f:
        writer = csv.writer(f)
        for rt in r_list:
            one_film_list = [rt[0].strip(),rt[1].strip(),rt[2].strip()]
            writer.writerow(one_film_list)
```

数据持久化 - MySQL

- **pymysql回顾**

```
# 1. 单条插入表记录 - excute()方法
# 2. 多条插入表记录 - excutemany()方法

# 示例代码如下:
import pymysql

db = pymysql.connect('localhost','root','123456','maoyandb',charset='utf8')
cursor = db.cursor()

ins = 'insert into filmtab values(%s,%s,%s)'
# 1. 单条插入表记录之 excute() 方法
cursor.execute(ins,['霸王别姬','张国荣','1993'])
# 2. 多条插入表记录之 excutemany() 方法 - 高效且节省资源
cursor.executemany(ins,[('大话1','周','1993'),('大话2','周','1994')])

db.commit()
cursor.close()
db.close()
```

- **练习 - 将电影信息存入MySQL数据库**

```
【1】提前建库建表
mysql -h127.0.0.1 -uroot -p123456
create database maoyandb charset utf8;
use maoyandb;
create table maoyantab(
name varchar(100),
star varchar(300),
time varchar(100)
```

```
)charset=utf8;

【2】 使用excute()方法将数据存入数据库 - 在初始代码基础上做如下改动
import pymysql

def __init__(self):
    # 添加代码
    self.db = pymysql.connect('localhost','root','123456','maoyandb',charset='utf8')
    self.cursor = self.db.cursor()

def save_html(self,dd_list):
    # 覆盖原来代码
    ins = 'insert into maoyantab values(%s,%s,%s)'
    for dd in dd_list:
        # 将每个电影信息处理成列表
        dd_li = [dd[0].strip(),dd[1].strip(),dd[2].strip()]
        self.cursor.execute(ins,dd_li)
        self.db.commit()
        print(dd_li)
        self.i += 1

def run(self):
    # 添加代码
    self.cursor.close()
    self.db.close()

【3】使用excutemany()方法将数据存入数据库 - 在初始代码基础上做如下改动
import pymysql

def __init__(self):
    # 添加代码
    self.db = pymysql.connect('localhost','root','123456','maoyandb',charset='utf8')
    self.cursor = self.db.cursor()
    # 存放所有电影信息的大列表
    self.all_film_list = []

def save_html(self,dd_list):
    # 覆盖原来代码
    for dd in dd_list:
        dd_tuple = (dd[0].strip(),dd[1].strip(),dd[2].strip())
        self.all_film_list.append(dd_tuple)
        self.i += 1

def run(self):
    # 添加代码
    ins = 'insert into maoyantab values(%s,%s,%s)'
    self.cursor.executemany(ins,self.all_film_list)
    self.db.commit()
    self.cursor.close()
    self.db.close()
```

- **4、练习 - SQL查询**

```
1、查询20年以前的电影的名字和上映时间
   select name,time from film where time<=(now()-interval 20 year);
2、查询1990-2000年的电影名字和上映时间
   select name,time from film where time>='1990-01-01' and time<='2000-12-31';
```

数据持久化 - MongoDB

- **MongoDB特点**

```
1、非关系型数据库,数据以键值对方式存储
2、MongoDB基于磁盘存储
3、MongoDB数据类型单一,值为JSON文档,而Redis基于内存,数据类型丰富
4、MongoDB: 库 -> 集合 -> 文档
   MySQL  : 库 -> 表  ->  表记录
```

- **MongoDB常用命令**

```
Linux进入: mongo
>show dbs                    - 查看所有库
>use 库名                     - 切换库
>show collections            - 查看当前库中所有集合
>db.集合名.find().pretty()    - 查看集合中文档
>db.集合名.count()            - 统计文档条数
>db.集合名.drop()             - 删除集合
>db.dropDatabase()           - 删除当前库
```

- **pymongo回顾**

```
import pymongo

# 1.连接对象
conn = pymongo.MongoClient(host = 'localhost',port = 27017)
# 2.库对象
db = conn['maoyandb']
# 3.集合对象
myset = db['maoyanset']
# 4.插入数据库
myset.insert_one({'name':'赵敏'})
myset.insert_many([{'name':'小昭'},{'age':'30'}])
```

- **练习 - 将电影信息存入MongoDB数据库**

```
"""在初始代码基础上做如下更改"""
import pymongo

def __init__(self):
    # 添加
    self.conn = pymongo.MongoClient('localhost',27017)
    self.db = self.conn['maoyandb']
    self.myset = self.db['maoyanset']
```

```python

def save_html(self,r_list):
    # 覆盖
    # 将数据处理为字典,执行insert_one()
    item = {}
    for r in r_list:
        item['name'] = r[0].strip()
        item['star'] = r[1].strip()
        item['time'] = r[2].strip()
        self.myset.insert_one(item)
```

汽车之家数据抓取 - 二级页面

- **领取任务**

```
# 1、爬取地址
汽车之家 - 二手车 - 价格从低到高
https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp1exx0/


# 2、爬取目标
车的型号、行驶里程、上牌时间、档位、排量、车辆所在地、价格

# 3、爬取分析
*********一级页面需抓取**********
1、车辆详情页的链接

*********二级页面需抓取**********
1、名称
2、行驶里程
3、上牌时间
4、档位
5、排量
6、车辆所在地
7、价格
```

- **实现步骤**

```
【1】确定响应内容中是否存在所需抓取数据 - 存在

【2】找URL地址规律
    第1页: https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp1exx0/
    第2页: https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp2exx0/
    第n页: https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp{}exx0/

【3】 写正则表达式
    一级页面正则表达式:<li class="cards-li list-photo-li".*?<a href="(.*?)".*?</li>
    二级页面正则表达式:<div class="car-box">.*?<h3 class="car-brand-name">(.*?)</h3>.*?<ul class="brand-unit-item fn-clear">.*?<li>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<span class="price" id="overlayPrice">￥(.*?)<b

【4】代码实现
```

- **代码实现**

```python
from urllib import request
import re
import time
import random


class CarSpider(object):
    def __init__(self):
        self.url = 'https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp{}exx0/'
        self.headers = {'User-Agent':'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.163 Safari/535.1'}

    # 功能函数1 - 获取响应内容
    def get_html(self,url):
        req = request.Request(url=url,headers=self.headers)
        try:
            res = request.urlopen(req,timeout=3)
            html = res.read().decode('gb2312','ignore')
            return html
        except Exception as e:
            print(e)


    # 功能函数2 - 正则解析
    def re_func(self,regex,html):
        pattern = re.compile(regex,re.S)
        r_list = pattern.findall(html)

        return r_list

    # 爬虫函数开始
    def parse_html(self,one_url):
        one_html = self.get_html(one_url)
        one_regex = '<li class="cards-li list-photo-li".*?<a href="(.*?)".*?</li>'
        href_list = self.re_func(one_regex,one_html)
        for href in href_list:
            # 每便利一个汽车信息，必须要把此辆汽车所有数据提取完成后再提取下一辆汽车信息
            url = 'https://www.che168.com' + href

            # 获取一辆汽车的信息
            self.get_data(url)
            time.sleep(random.randint(1,2))

    # 获取一辆汽车信息
    def get_data(self,url):
        two_html = self.get_html(url)
        two_regex = '<div class="car-box">.*?<h3 class="car-brand-name">(.*?)</h3>.*?<ul class="brand-unit-item fn-clear">.*?<li>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<span class="price" id="overlayPrice">￥(.*?)<b'
        item = {}
        car_info_list = self.re_func(two_regex,two_html)
        item['name'] = car_info_list[0][0]
        item['km'] = car_info_list[0][1]
        item['year'] = car_info_list[0][2]
```

```
        item['type'] = car_info_list[0][3].split('/')[0]
        item['displacement'] = car_info_list[0][3].split('/')[1]
        item['city'] = car_info_list[0][4]
        item['price'] = car_info_list[0][5]
        print(item)

    def run(self):
        for p in range(1,11):
            url = self.url.format(p)
            self.parse_html(url)

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

- **扩展 - 增量爬取**

```
# 将数据存入MySQL数据库 - 增量爬取
【1】思路
     1.1 MySQL中新建表 urltab,存储所有爬取过的链接的指纹
     1.2 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

  【2】建库建表
  create database cardb charset utf8;
  use cardb;
  create table request_finger(
  finger char(32)
  )charset=utf8;
  create table cartab(
  name varchar(100),
  km varchar(50),
  years varchar(50),
  type varchar(50),
  displacement varchar(50),
  city varchar(50),
  price varchar(50)
  )charset=utf8;
```

- **增量爬取 - MySQL**

```
from urllib import request
import re
import time
import random
import pymysql
from hashlib import md5
import sys


class CarSpider(object):
    def __init__(self):
        self.url = 'https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp{}exx0/'
        self.headers = {'User-Agent':'Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.163 Safari/535.1'}
        self.db = pymysql.connect('localhost','root','123456','cardb',charset='utf8')
```

```
        self.cursor = self.db.cursor()

    # 功能函数1 - 获取响应内容
    def get_html(self,url):
      req = request.Request(url=url,headers=self.headers)
      try:
        res = request.urlopen(req,timeout=3)
        html = res.read().decode('gb2312','ignore')
        return html
      except Exception as e:
        print(e)

    # 功能函数2 - 正则解析
    def re_func(self,regex,html):
        pattern = re.compile(regex,re.S)
        r_list = pattern.findall(html)

        return r_list

    # 爬虫函数开始
    def parse_html(self,one_url):
        one_html = self.get_html(one_url)
        one_regex = '<li class="cards-li list-photo-li".*?<a href="(.*?)".*?</li>'
        href_list = self.re_func(one_regex,one_html)
        for href in href_list:
            # 加密指纹
            s = md5()
            s.update(href.encode())
            finger = s.hexdigest()
            # 如果指纹表中不存在
            if self.go_spider(finger):
                # 每便利一个汽车信息，必须要把此辆汽车所有数据提取完成后再提取下一辆汽车信息
                url = 'https://www.che168.com' + href

                # 获取一辆汽车的信息
                self.get_data(url)
                ins = 'insert into request_finger values(%s)'
                self.cursor.execute(ins,[finger])
                self.db.commit()
                time.sleep(random.randint(1,2))
            else:
                sys.exit('抓取结束')

    # 判断是否存在：存在返回False，不存在返回True
    def go_spider(self,finger):
        sel = 'select * from request_finger where finger=%s'
        result = self.cursor.execute(sel,[finger])
        if result:
            return False
        return True

    # 获取一辆汽车信息
    def get_data(self,url):
        two_html = self.get_html(url)
        two_regex = '<div class="car-box">.*?<h3 class="car-brand-name">(.*?)</h3>.*?<ul
class="brand-unit-item fn-clear">.*?<li>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)
</h4>.*?<h4>(.*?)</h4>.*?<span class="price" id="overlayPrice">￥(.*?)<b'
```

```
        item = {}
        car_info_list = self.re_func(two_regex,two_html)
        item['name'] = car_info_list[0][0]
        item['km'] = car_info_list[0][1]
        item['year'] = car_info_list[0][2]
        item['type'] = car_info_list[0][3].split('/')[0]
        item['displacement'] = car_info_list[0][3].split('/')[1]
        item['city'] = car_info_list[0][4]
        item['price'] = car_info_list[0][5]
        print(item)

        one_car_list = [
            item['name'],
            item['km'],
            item['year'],
            item['type'],
            item['displacement'],
            item['city'],
            item['price']
        ]
        ins = 'insert into cartab values(%s,%s,%s,%s,%s,%s,%s)'
        self.cursor.execute(ins,one_car_list)
        self.db.commit()

    def run(self):
        for p in range(1,2):
            url = self.url.format(p)
            self.parse_html(url)

        # 断开数据库链接
        self.cursor.close()
        self.db.close()

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

- **能不能使用redis来实现增量**

```
"""
  提示: 使用redis中的集合,sadd()方法,添加成功返回1,否则返回0
  请各位大佬忽略掉下面代码,自己独立实现
"""

from urllib import request
import re
import time
import random
import pymysql
from hashlib import md5
import sys
import redis


class CarSpider(object):
    def __init__(self):
```

```
        self.url = 'https://www.che168.com/beijing/a0_0msdgscncgpi1lto1csp{}exx0/'
        self.headers = {'User-Agent':'Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.163 Safari/535.1'}
        self.db = pymysql.connect('localhost','root','attack','cardb',charset='utf8')
        self.cursor = self.db.cursor()
        # 连接redis去重
        self.r = redis.Redis(host='localhost',port=6379,db=0)

    # 功能函数1 - 获取响应内容
    def get_html(self,url):
        req = request.Request(url=url,headers=self.headers)
        res = request.urlopen(req)
        html = res.read().decode('gb2312','ignore')

        return html

    # 功能函数2 - 正则解析
    def re_func(self,regex,html):
        pattern = re.compile(regex,re.S)
        r_list = pattern.findall(html)

        return r_list

    # 爬虫函数开始
    def parse_html(self,one_url):
        one_html = self.get_html(one_url)
        one_regex = '<li class="cards-li list-photo-li".*?<a href="(.*?)".*?</li>'
        href_list = self.re_func(one_regex,one_html)
        for href in href_list:
            # 加密指纹
            s = md5()
            s.update(href.encode())
            finger = s.hexdigest()
            # 如果指纹表中不存在
            if self.r.sadd('car:urls',finger):
                # 每便利一个汽车信息，必须要把此辆汽车所有数据提取完成后再提取下一辆汽车信息
                url = 'https://www.che168.com' + href

                # 获取一辆汽车的信息
                self.get_data(url)
                time.sleep(random.randint(1,2))
            else:
                sys.exit('抓取结束')

    # 判断是否存在：存在返回False，不存在返回True
    def go_spider(self,finger):
        sel = 'select * from request_finger where finger=%s'
        result = self.cursor.execute(sel,[finger])
        if result:
            return False
        return True

    # 获取一辆汽车信息
    def get_data(self,url):
        two_html = self.get_html(url)
```

```
        two_regex = '<div class="car-box">.*?<h3 class="car-brand-name">(.*?)</h3>.*?<ul class="brand-unit-item fn-clear">.*?<li>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<h4>(.*?)</h4>.*?<span class="price" id="overlayPrice">￥(.*?)<b'
        item = {}
        car_info_list = self.re_func(two_regex,two_html)
        item['name'] = car_info_list[0][0]
        item['km'] = car_info_list[0][1]
        item['year'] = car_info_list[0][2]
        item['type'] = car_info_list[0][3].split('/')[0]
        item['displacement'] = car_info_list[0][3].split('/')[1]
        item['city'] = car_info_list[0][4]
        item['price'] = car_info_list[0][5]
        print(item)

        one_car_list = [
            item['name'],
            item['km'],
            item['year'],
            item['type'],
            item['displacement'],
            item['city'],
            item['price']
        ]
        ins = 'insert into cartab values(%s,%s,%s,%s,%s,%s,%s)'
        self.cursor.execute(ins,one_car_list)
        self.db.commit()

    def run(self):
        for p in range(1,2):
            url = self.url.format(p)
            self.parse_html(url)

        # 断开数据库链接
        self.cursor.close()
        self.db.close()

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

requests模块

- **安装**

```
# 1. Linux
sudo pip3 install requests

# 2. Windows
方法1:  cmd命令行 -> python -m pip install requests
方法2:  右键管理员进入cmd命令行 : pip install requests
```

- **requests.get()**

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

# 2. 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

# 3. 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)
```

- **示例代码 - 图片抓取**

```
# 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)
```