

王伟超

wangweichao@tedu.cn

DAY01

网络爬虫概述

定义

- 1 网络蜘蛛、网络机器人，抓取网络数据的程序
- 2 其实就是用Python程序模仿人点击浏览器并访问网站，而且模仿的越像越好，让Web站点无法发现你不是人

爬取数据目的

- 1 1、公司项目测试数据
- 2 2、公司业务部门及其他部门所需数据
- 3 3、数据分析

企业获取数据方式

- 1 1、公司自有数据
- 2 2、第三方数据平台购买(数据堂、贵阳大数据交易所)
- 3 3、爬虫爬取数据

Python做爬虫优势

- 1 Python：请求模块、解析模块丰富成熟,强大的Scrapy网络爬虫框架
- 2 PHP：对多线程、异步支持不太好
- 3 JAVA：代码笨重,代码量大
- 4 C/C++：虽然效率高,但是代码成型慢

爬虫分类

- 1 通用网络爬虫(搜索引擎使用,遵守robots协议)
robots协议：网站通过robots协议告诉搜索引擎哪些页面可以抓取,哪些页面不能抓取,通用网络爬虫需要遵守robots协议(君子协议)
`https://www.taobao.com/robots.txt`
- 2 聚焦网络爬虫：自己写的爬虫程序

爬虫爬取数据步骤

- 1 确定需要爬取的URL地址
- 2 由请求模块向URL地址发出请求,并得到网站的响应
- 3 从响应内容中提取所需数据
 - 1 所需数据,保存
 - 2 页面中有其他需要继续跟进的URL地址,继续第2步去发请求,如此循环

爬虫请求模块一

模块名及导入

- 1 模块名: `urllib.request`
- 2 导入方式:
 - 1 `import urllib.request`
 - 2 `from urllib import request`

常用方法详解

`urllib.request.urlopen`

作用

向网站发起请求并获取响应对象

参数

- 1 `url`: 需要爬取的URL地址
- 2 `timeout`: 设置等待超时时间,指定时间内未得到响应抛出超时异常

第一个爬虫程序 - 01_urlopen.py

打开浏览器，输入百度地址(<http://www.baidu.com/>)，得到百度的响应

```
1 import urllib.request
2
3 # urlopen() : 向URL发请求,返回响应对象
4 response=urllib.request.urlopen('http://httpbin.org/get')
5 # 提取响应内容
6 html = response.read().decode('utf-8')
7 # 打印响应内容
8 print(html)
```

■ 响应对象 (response) 方法

```
1 1、bytes = response.read() # read()得到结果为 bytes 数据类型
2 2、string = response.read().decode() # decode() 转为 string 数据类型
3 3、url = response.geturl() # 返回实际数据的URL地址
4 4、code = response.getcode() # 返回HTTP响应码
5 # 补充
6 5、string.encode() # bytes -> string
7 6、bytes.decode() # string -> bytes
```

思考：网站如何来判定是人类正常访问还是爬虫程序访问？？？

```
1 # 向测试网站: http://httpbin.org/get 发请求,查看自己请求头
2 # 代码如下
3 import urllib.request
4 response=urllib.request.urlopen('http://httpbin.org/get')
5 html = response.read().decode()
6 print(html)
7
8 # html中的请求头headers如下
9 "headers": {
10     "Accept-Encoding": "identity",
11     "Host": "httpbin.org",
12     "User-Agent": "Python-urllib/3.6"
13 },
14 发现请求头中User-Agent竟然是:Python-urllib/3.6!!!!!!!!!!!!!!!!!!!!!!
15 我们需要重构User-Agent,发请求时带着User-Agent过去,但是 urlopen()方法不支持重构User-Agent,那我们怎么办? 请看下面的方法!!!
```

urllib.request.Request

■ 作用

创建请求对象(包装请求，重构User-Agent，使程序更像正常人类请求)

■ 参数

```
1 1、url: 请求的URL地址
2 2、headers: 添加请求头 (爬虫和反爬虫斗争的第一步)
```

■ 使用流程

```

1 1、构造请求对象(重构User-Agent)
2   req = urllib.request.Request(
3       url = 'http://httpbin.org/get'
4       headers={'User-Agent': 'Mozilla/5.0'}
5   )
6 2、发请求获取响应对象(urlopen)
7   res = urllib.request.urlopen(req)
8 3、获取响应对象内容
9   html = res.read().decode('utf-8')

```

■ 示例 - 02_Request.py

向测试网站 (<http://httpbin.org/get>) 发起请求, 构造请求头并从响应中确认请求头信息

```

1  from urllib import request
2
3  # 定义常用变量: URL 、 headers
4  url = 'http://httpbin.org/get'
5  headers = {
6      'User-Agent': 'Opera/9.80 (Windows NT 6.1; U; zh-cn) Presto/2.9.168 Version/11.50'
7  }
8  # 1. 创建请求对象 - 包装, 并没有真正发请求
9  req = request.Request(url=url, headers=headers)
10 # 2. 获取响应对象
11 res = request.urlopen(req)
12 # 3. 提取响应内容
13 html = res.read().decode('utf-8')
14
15 print(html)

```

URL地址编码模块

模块名及导入

■ 模块

```

1 # 模块名
2 urllib.parse
3
4 # 导入
5 import urllib.parse
6 from urllib import parse

```

■ 作用

给URL地址中查询参数进行编码

```

1 编码前: https://www.baidu.com/s?wd=美女
2 编码后: https://www.baidu.com/s?wd=%E7%BE%8E%E5%A5%B3

```

常用方法

urllib.parse.urlencode({dict})

■ URL地址中一个查询参数

```
1 # 查询参数: {'wd' : '美女'}
2 # urlencode编码后: 'wd=%e7%be%8e%e5%a5%b3'
3
4 # 示例代码
5 query_string = {'wd' : '美女'}
6 result = urllib.parse.urlencode(query_string)
7 # result: 'wd=%e7%be%8e%e5%a5%b3'
```

■ URL地址中多个查询参数

```
1 from urllib import parse
2 params = {
3     'wd' : '美女',
4     'pn' : '50'
5 }
6 params = parse.urlencode(query_string_dict)
7 url = 'http://www.baidu.com/s?{}'.format(params)
8 print(url)
```

■ 拼接URL地址的3种方式

```
1 # 1、字符串相加
2 baseurl = 'http://www.baidu.com/s?'
3 params = 'wd=%E7XXXX&pn=20'
4 url = baseurl + params
5
6 # 2、字符串格式化（占位符）
7 params = 'wd=%E7XXXX&pn=20'
8 url = 'http://www.baidu.com/s?%s'% params
9
10 # 3、format()方法
11 url = 'http://www.baidu.com/s?{'
12 params = 'wd=%E7XXXX&pn=20'
13 url = url.format(params)
```

■ 练习 在百度中输入要搜索的内容，把响应内容保存到本地文件

```
1 请输入搜索内容: 赵丽颖
2 # 最终保存到本地文件 - 赵丽颖.html
```

代码实现 - 03_parse_baidu.py

```
1 from urllib import request
2 from urllib import parse
3
4 # 拼接URL地址
```

```

5 def get_url(word):
6     url = 'http://www.baidu.com/s?{'
7     # params: wd=E7XXXXX
8     params = parse.urlencode({'wd':word})
9     url = url.format(params)
10
11     return url
12
13
14 # 发请求,保存本地文件
15 def request_url(url,filename):
16     headers = {'User-Agent':'Mozilla/5.0'}
17     # 请求对象 + 响应对象 + 提取内容
18     req = request.Request(url=url,headers=headers)
19     res = request.urlopen(req)
20     html = res.read().decode('utf-8')
21     # 保存数据
22     with open(filename,'w',encoding='utf-8') as f:
23         f.write(html)
24
25 # 主程序入口
26 if __name__ == '__main__':
27     word = input('请输入搜索内容:')
28     url = get_url(word)
29     filename = word + '.html'
30     request_url(url,filename)

```

quote(string)编码

■ 示例1

```

1 from urllib import parse
2
3 string = '美女'
4 print(parse.quote(string))
5 # 结果: %E7%BE%8E%E5%A5%B3

```

改写之前urlencode()代码, 使用quote()方法实现

```

1 from urllib import parse
2
3 url = 'http://www.baidu.com/s?wd={}'
4 word = input('请输入要搜索的内容:')
5 query_string = parse.quote(word)
6 print(url.format(query_string))

```

unquote(string)解码

■ 示例

```
1 from urllib import parse
2
3 string = '%E7%BE%8E%E5%A5%B3'
4 result = parse.unquote(string)
5 print(result)
```

总结

```
1 # 1、urllib.request
2 req = urllib.request.Request(url,headers)
3 res = urllib.request.urlopen(req)
4 html = res.read().decode('utf-8')
5
6 # 2、响应对象res方法
7 res.read()
8 res.getcode()
9 res.geturl()
10
11 # 3、urllib.parse
12 urllib.parse.urlencode({})
13 urllib.parse.quote(string)
14 urllib.parse.unquote(string)
```

百度贴吧数据抓取案例

要求

```
1 1、输入贴吧名称:赵丽颖吧
2 2、输入起始页:1
3 3、输入终止页:3
4 4、保存到本地文件
5 赵丽颖吧-第1页.html、赵丽颖吧-第2页.html ...
```

实现步骤

■ 1、查看是否为静态页面

```
1 右键 - 查看网页源代码 - 搜索数据关键字
```

■ 2、找URL规律

```
1 第1页:http://tieba.baidu.com/f?kw=? ? &pn=0
2 第2页:http://tieba.baidu.com/f?kw=? ? &pn=50
3 第n页:pn=(n-1)*50
```

- 3、获取网页内容
- 4、提取所需数据
- 5、保存(本地文件、数据库)

代码实现 - 04_tieba_spider.py

```
1  from urllib import request,parse
2  import time
3  import random
4  from useragents import ua_list
5
6  class BaiduSpider(object):
7      def __init__(self):
8          self.url = 'http://tieba.baidu.com/f?kw={}&pn={}'
9
10     # 获取
11     def get_html(self,url):
12         headers = {
13             'User-Agent': random.choice(ua_list)
14         }
15         req = request.Request(url=url,headers=headers)
16         res = request.urlopen(req)
17         html = res.read().decode('utf-8')
18
19         return html
20
21     # 解析
22     def parse_html(self):
23         pass
24
25     # 保存
26     def write_html(self,filename,html):
27         with open(filename,'w',encoding='utf-8') as f:
28             f.write(html)
29
30     # 入口函数
31     def run(self):
32         name = input('请输入贴吧名:')
33         begin = int(input('请输入起始页:'))
34         end = int(input('请输入终止页:'))
35
36         # 查询参数编码
37
38         params = parse.quote(name)
39         for page in range(begin,end+1):
40             # URL拼接
41             pn = (page-1)*50
42             pn = (page-1)*50
43             url = self.url.format(params,pn)
44
45             # 调用类内函数进行页面抓取+保存
46             filename = '{}-第{}页.html'.format(name,page)
47             html = self.get_html(url)
48             self.write_html(filename,html)
49             # 控制爬取速度
50             time.sleep(random.randint(1,3))
```



```
51         print('第%d页爬取完成' % page)
52
53     if __name__ == '__main__':
54         start = time.time()
55         spider = BaiduSpider()
56         spider.run()
57         end = time.time()
58         print('执行时间:%.2f' % (end-start))
```

正则解析模块

re 模块使用流程

▪ 方法一

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

▪ 方法二

```
1 # 1、创建正则编译对象
2 pattern = re.compile(r'正则表达式',re.S)
3 r_list = pattern.findall(html)
```

正则表达式元字符

元字符	含义
.	任意一个字符（不包括\n）
\d	一个数字
\s	空白字符
\S	非空白字符
[]	包含[]内容
*	出现0次或多次
+	出现1次或多次

思考：请写出匹配任意一个字符的正则表达式？

```

1 import re
2 # 方法一
3 pattern = re.compile('.',re.S)
4 # 方法二
5 pattern = re.compile('[\s\S]')

```

贪婪匹配和非贪婪匹配

■ 贪婪匹配（默认）

- 1、在整个表达式匹配成功的前提下,尽可能多的匹配 *
- 2、表示方式: `.*`

■ 非贪婪匹配

- 1、在整个表达式匹配成功的前提下,尽可能少的匹配 *
- 2、表示方式: `.*`

示例代码 - 05_re_greed.py

```

1 import re
2
3 html = '''
4 <div><p>九霄龙吟惊天变</p></div>
5 <div><p>风云际汇潜水游</p></div>
6 '''
7 # 贪婪匹配 : .*
8 pattern = re.compile('<div><p>.*</p></div>',re.S)
9 r_list = pattern.findall(html)
10 print(r_list)
11
12 # 非贪婪匹配 : .*?
13 pattern = re.compile('<div><p>.*?</p></div>',re.S)
14 r_list = pattern.findall(html)
15 print(r_list)

```

正则表达式分组

■ 作用

在完整的模式中定义子模式，将每个圆括号中子模式匹配出来的结果提取出来

■ 示例

```

1 import re
2
3 s = 'A B C D'

```

```

4 p1 = re.compile('\w+\s+\w+')
5 print(p1.findall(s))
6 # 结果: ['A B','C D']
7
8 p2 = re.compile('(\w+)\s+\w+')
9 print(p2.findall(s))
10 # 结果: ['A','C']
11
12 p3 = re.compile('(\w+)\s+(\w+)')
13 print(p3.findall(s))
14 # 结果: [('A','B'),('C','D')]

```

■ 分组总结

- 1、在网页中,想要什么内容,就加()
- 2、先按整体正则匹配,然后再提取分组()中的内容
- 3 如果有2个及以上分组(),则结果中以元组形式显示 [('小区1','500万'),('小区2','600万'),()]

练习

页面结构如下:

```

1 # <div class="animal">.*?title="(.*?)".*?
2 <div class="animal">
3     <p class="name">
4         <a title="Tiger"></a>
5     </p>
6     <p class="content">
7         Two tigers two tigers run fast
8     </p>
9 </div>
10
11 <div class="animal">
12     <p class="name">
13         <a title="Rabbit"></a>
14     </p>
15
16     <p class="content">
17         Small white rabbit white and white
18     </p>
19 </div>

```

从以上html代码结构中完成如下内容信息的提取:

```

1 # 问题1
2 [('Tiger',' Two...'),('Rabbit','Small..')]
3 # 问题2
4 动物名称 : Tiger
5 动物描述 : Two tigers two tigers run fast
6 *****
7 动物名称 : Rabbit
8 动物描述 : Small white rabbit white and white

```

代码实现 - 06_re_exercise.py

```
1 import re
2
3 html = '''
4 <div class="animal">
5     <p class="name">
6         <a title="Tiger"></a>
7     </p>
8     <p class="content">
9         Two tigers two tigers run fast
10    </p>
11 </div>
12
13 <div class="animal">
14     <p class="name">
15         <a title="Rabbit"></a>
16     </p>
17
18     <p class="content">
19         Small white rabbit white and white
20     </p>
21 </div>
22 '''
23 pattern = re.compile(r'<div class="animal">.*?<a title="(.*?)".*?content">(.*?)</p>',re.S)
24 r_list = pattern.findall(html)
25 # 问题1
26 if r_list:
27     print(r_list)
28 # r_list: [('Tiger','\n\t\t Two tigers xxx'),()]
29 # 问题2
30 if r_list:
31     for r in r_list:
32         print('动物名称:',r[0].strip())
33         print('动物描述:',r[1].strip())
34         print('*' * 50)
35 else:
36     print('未匹配到数据')
```

今日作业

1、把百度贴吧案例重写一遍,不要参照课上代码 2、爬取猫眼电影信息：猫眼电影-榜单-top100榜

```
1 第1步完成：
2 猫眼电影-第1页.html
3 猫眼电影-第2页.html
4 ... ...
5
6 第2步完成：
7 1、提取数据：电影名称、主演、上映时间
8 2、先打印输出,然后再写入到本地文件
```

3、复习任务

- 1 | pymysql、MySQL基本命令
- 2 | MySQL : 建库建表普通查询等

DAY02

Day01回顾

请求模块(*urllib.request*)

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

编码模块(*urllib.parse*)

```
1 1、urlencode({dict})
2   urlencode({'wd':'美女','pn':'20'})
3   编码后 : 'wd=%E8%D5XXX&pn=20'
4
5 2、quote(string)
6   quote('织女')
7   编码后 : '%D3%F5XXX'
8
9 3、unquote('%D3%F5XXX')
```

解析模块(*re*)

■ 使用流程

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

■ 贪婪匹配和非贪婪匹配

```
1 贪婪匹配(默认) : .*
2 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
 - 1、使用随机User-Agent
 - 2、每爬取1个页面后随机休眠一段时间

```
1 # 程序结构
2 class xxxSpider(object):
3     def __init__(self):
4         # 定义常用变量,url,headers及计数等
5
6     def get_html(self):
7         # 获取响应内容函数,使用随机User-Agent
8
9     def parse_html(self):
10        # 使用正则表达式来解析页面,提取数据
11
12    def write_html(self):
13        # 将提取的数据按要求保存, csv、MySQL数据库等
14
15    def main(self):
16        # 主函数,用来控制整体逻辑
17
18 if __name__ == '__main__':
19     # 程序开始运行时间戳
20     start = time.time()
21     spider = xxxSpider()
22     spider.main()
23     # 程序运行结束时间戳
24     end = time.time()
25     print('执行时间:%.2f' % (end-start))
```

Day02笔记

作业讲解

作业1 - 正则分组练习

页面结构如下：

```
1 <div class="animal">
2   <p class="name">
3     <a title="Tiger"></a>
4   </p>
5   <p class="content">
6     Two tigers two tigers run fast
7   </p>
8 </div>
9
10 <div class="animal">
11   <p class="name">
12     <a title="Rabbit"></a>
13   </p>
14
15   <p class="content">
16     Small white rabbit white and white
17   </p>
18 </div>
```

从以上html代码结构中完成如下内容信息的提取：

```
1 # 问题1
2 [('Tiger', ' Two...'), ('Rabbit', 'Small..')]
3 # 问题2
4 动物名称 : Tiger
5 动物描述 : Two tigers two tigers run fast
6 *****
7 动物名称 : Rabbit
8 动物描述 : Small white rabbit white and white
```

代码实现

```
1 |
```

猫眼电影top100抓取案例

```
1 猫眼电影 - 榜单 - top100榜
2 电影名称、主演、上映时间
```

数据抓取实现

■ 1、确定响应内容中是否存在所需数据

```
1 右键 - 查看网页源代码 - 搜索关键字 - 存在！！
```

▪ 2、找URL规律

```
1 第1页: https://maoyan.com/board/4?offset=0
2 第2页: https://maoyan.com/board/4?offset=10
3 第n页: offset=(n-1)*10
```

▪ 3、正则表达式

```
1 <div class="movie-item-info">.*?title="(.*?)".*?class="star">(.*?)</p>.*?releasetime">(.*?)</p>
```

▪ 4、编写程序框架，完善程序

```
1 |
```

数据持久化存储

数据持久化存储 - csv文件

▪ 作用

```
1 将爬取的数据存放到本地的csv文件中
```

▪ 使用流程

```
1 1、导入模块
2 2、打开csv文件
3 3、初始化写入对象
4 4、写入数据(参数为列表)
5 import csv
6
7 with open('film.csv','w') as f:
8     writer = csv.writer(f)
9     writer.writerow([])
```

▪ 示例代码

创建 test.csv 文件，在文件中写入数据


```

1 # 单行写入 (writerow([]))
2 import csv
3 with open('test.csv','w',newline='') as f:
4     writer = csv.writer(f)
5     writer.writerow(['步惊云','36'])
6     writer.writerow(['超哥哥','25'])
7
8 # 多行写入(writerows([()],(),[]))
9 import csv
10 with open('test.csv','w',newline='') as f:
11     writer = csv.writer(f)
12     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 - 使用writerow()方法实现

```

1 # 存入csv文件 - writerow()
2 def write_html(self,film_list):
3     with open('film.csv','a') as f:
4         # 初始化写入对象,注意参数f别忘了
5         writer = csv.writer(f)
6         for film in film_list:
7             L = [
8                 film[0].strip(),
9                 film[1].strip(),
10                film[2].strip()[5:15]
11            ]
12            # writerow()参数为列表
13            writer.writerow(L)

```

思考：使用 writerows()方法实现？

```

1 # 存入csv文件 - writerows()
2 def write_html(self,film_list):
3     L = []
4     with open('film.csv','a') as f:
5         # 初始化写入对象,注意参数f别忘了
6         writer = csv.writer(f)
7         for film in film_list:
8             t = (
9                 film[0].strip(),
10                film[1].strip(),
11                film[2].strip()[5:15]
12            )
13            L.append(t)
14            # writerows()参数为列表
15            writer.writerows(L)

```

1、在数据库中建库建表

```
1 # 连接到mysql数据库
2 mysql -h127.0.0.1 -uroot -p123456
3 # 建库建表
4 create database maoyandb charset utf8;
5 use maoyandb;
6 create table filmtab(
7 name varchar(100),
8 star varchar(300),
9 time varchar(50)
10 )charset=utf8;
```

■ 2、回顾pymysql基本使用

```
1 import pymysql
2
3 # 创建2个对象
4 db = pymysql.connect('localhost','root','123456','maoyandb',charset='utf8')
5 cursor = db.cursor()
6
7 # 执行SQL命令并提交到数据库执行
8 # execute()方法第二个参数为列表传参补位
9 ins = 'insert into filmtab values(%s,%s,%s)'
10 cursor.execute(ins,['霸王别姬','张国荣','1993'])
11 db.commit()
12
13 # 关闭
14 cursor.close()
15 db.close()
```

■ 来试试高效的executemany()方法?

```
1 import pymysql
2
3 # 创建2个对象
4 db = pymysql.connect('192.168.153.137','tiger','123456','maoyandb',charset='utf8')
5 cursor = db.cursor()
6
7 # 抓取的数据
8 film_list = [('月光宝盒','周星驰','1994'),('大圣娶亲','周星驰','1994')]
9
10 # 执行SQL命令并提交到数据库执行
11 # execute()方法第二个参数为列表传参补位
12 cursor.executemany('insert into filmtab values(%s,%s,%s)',film_list)
13 db.commit()
14
15 # 关闭
16 cursor.close()
17 db.close()
```

■ 3、将电影信息存入MySQL数据库（尽量使用executemany方法）

```
1 # mysql - executemany([()],(),())
```

```

2 def write_html(self, film_list):
3     L = []
4     ins = 'insert into filmtab values(%s,%s,%s)'
5     for film in film_list:
6         t = (
7             film[0].strip(),
8             film[1].strip(),
9             film[2].strip()[5:15]
10        )
11        L.append(t)
12
13        self.cursor.executemany(ins, L)
14        # 千万别忘了提交到数据库执行
15        self.db.commit()

```

▪ 4、做个SQL查询

- 1 1、查询20年以前的电影的名字和上映时间
- 2
- 3 2、查询1990-2000年的电影名字和上映时间
- 4

数据持久化存储 - MongoDB数据库

pymongo操作mongodb数据库

```

1 import pymongo
2
3 # 1.数据库连接对象
4 conn=pymongo.MongoClient('localhost',27017)
5 # 2.库对象
6 db = conn['库名']
7 # 3.集合对象
8 myset = db['集合名']
9 # 4.插入数据
10 myset.insert_one({字典})

```

思考

- 1 1、能否到电影详情页把评论抓取下来？
- 2 2、能否到电影详情页把电影图片抓取下来？ - 并按照电影名称分别创建文件夹

代码实现

```

1 |

```

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="(.*?)"*.?<ulink">.*?</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、代码实现

```
1 |
```

- 5、练习

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 urltab,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
```

练习代码实现

```
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 1、将猫眼电影案例改写为 requests 模块实现
2 2、将电影天堂案例改写为 requests 模块实现
```

Chrome浏览器安装插件

安装方法

```
1 1、把下载的相关插件（对应操作系统浏览器）后缀改为 .zip
2 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 1、Xpath Helper: 轻松获取HTML元素的XPath路径
2   # 开启/关闭: Ctrl + Shift + x
3 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
```

今日作业

- 1 1、把之前所有代码改为 requests 模块
- 2 2、电影天堂数据,存入MySQL、MongoDB、CSV文件
- 3 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:
- 4 你想要谁的图片, 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
- 6 4、抓取链家二手房房源信息 (房源名称、总价) ,把结果存入到MySQL数据库,MongoDB数据库,CSV文件
- 7 # 小区名 、总价 、单价