

王伟超

wangweichao@tedu.cn

Spider-Day01笔记

网络爬虫概述

【1】定义

- 1.1) 网络蜘蛛、网络机器人, 抓取网络数据的程序
- 1.2) 其实就是用Python程序模仿人点击浏览器并访问网站, 而且模仿的越逼真越好

【2】爬取数据的目的

- 2.1) 公司项目的测试数据, 公司业务所需数据
- 2.2) 获取大量数据, 用来做数据分析

【3】企业获取数据方式

- 3.1) 公司自有数据
- 3.2) 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

【4】Python做爬虫优势

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

【5】爬虫分类

- 5.1) 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)

示例: <https://www.baidu.com/robots.txt>

- 5.2) 聚焦网络爬虫 : 自己写的爬虫程序

【6】爬取数据步骤

- 6.1) 确定需要爬取的URL地址
- 6.2) 由请求模块向URL地址发出请求, 并得到网站的响应
- 6.3) 从响应内容中提取所需数据
 - a> 所需数据, 保存
 - b> 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

requests模块

■ 安装

```
1  【1】 Linux
2      sudo pip3 install requests
3
4  【2】 Windows
5      方法1> cmd命令行 -> python -m pip install requests
6      方法2> 右键管理员进入cmd命令行 : pip install requests
```

常用方法

■ requests.get()

```
1  【1】 作用
2      向目标网站发起请求,并获取响应对象
3
4  【2】 参数
5      2.1> url : 需要抓取的URL地址
6      2.2> headers : 请求头
7      2.3> timeout : 超时时间, 超过时间会抛出异常
```

■ 此生第一个爬虫

```
1  """
2  打开浏览器, 输入京东地址(https://www.jd.com/), 得到响应内容
3  """
4  import requests
5
6  res = requests.get('https://www.jd.com/')
7  html = res.text
8  print(html)
```

■ 响应对象 (res) 属性

```
1  【1】 text      : 字符串
2  【2】 content   : 字节流
3  【3】 status_code : HTTP响应码
4  【4】 url       : 实际数据的URL地址
```

■ 重大问题思考

网站如何来判定是人类正常访问还是爬虫程序访问? --检查请求头!!!

```

1 # 请求头 (headers) 中的 User-Agent
2 # 测试案例: 向测试网站http://httpbin.org/get发请求, 查看请求头(User-Agent)
3 import requests
4
5 url = 'http://httpbin.org/get'
6 res = requests.get(url=url)
7 html = res.text
8 print(html)
9 # 请求头中:User-Agent为-> python-requests/2.22.0 那第一个被网站干掉的是谁??? 我们是不是需要发送
    请求时重构一下User-Agent??? 添加 headers 参数!!!

```

■ 重大问题解决

```

1 """
2 包装好请求头后,向测试网站发请求,并验证
3 养成好习惯, 发送请求携带请求头, 重构User-Agent
4 """
5 import requests
6
7 url = 'http://httpbin.org/get'
8 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'}
9 html = requests.get(url=url,headers=headers).text
10 print(html)

```

爬虫编码模块

■ urllib.parse模块

```

1 1、标准库模块: urllib.parse
2 2、导入方式:
3 import urllib.parse
4 from urllib import parse

```

■ 作用

```

1 给URL地址中查询参数进行编码
2
3 # 示例
4 编码前: https://www.baidu.com/s?wd=美女
5 编码后: https://www.baidu.com/s?wd=%E7%BE%8E%E5%A5%B3

```

常用方法

urlencode({ 参数为字典 })

■ 作用

```
1 | 给URL地址中查询参数进行编码，参数类型为字典
```

■ 使用方法

```
1 | # 1、URL地址中 一 个查询参数
2 | 编码前: params = {'wd': '美女'}
3 | 编码中: params = urllib.parse.urlencode(params)
4 | 编码后: params结果: 'wd=%E7%BE%8E%E5%A5%B3'
5 |
6 | # 2、URL地址中 多 个查询参数
7 | 编码前: params = {'wd': '美女', 'pn': '50'}
8 | 编码中: params = urllib.parse.urlencode(params)
9 | 编码后: params结果: 'wd=%E7%BE%8E%E5%A5%B3&pn=50'
10 | 发现编码后会自动对多个查询参数间添加 & 符号
```

■ 拼接URL地址的三种方式

```
1 | # url = 'http://www.baidu.com/s?'
2 | # params = {'wd': '赵丽颖'}
3 | # 问题: 请拼接出完整的URL地址
4 | *****
5 | params = urllib.parse.urlencode(params)
6 | 【1】字符串相加
7 | 【2】字符串格式化 (占位符 %s)
8 | 【3】format()方法
9 | 'http://www.baidu.com/s?{}'.format(params)
10 |
11 | 【练习】
12 | 进入瓜子二手车直卖网官网 - 我要买车 - 请使用3种方法拼接前20页的URL地址,从终端打印输出
13 | 官网地址: https://www.guazi.com/langfang/
```

■ 练习

```
1 | """
2 | 问题: 在百度中输入要搜索的内容, 把响应内容保存到本地文件
3 | 编码方法使用 urlencode()
4 | """
5 | import requests
6 | from urllib import parse
7 |
8 | # 1.拼接URL地址
9 | word = input('请输入搜索内容:')
10 | params = parse.urlencode({'wd': word})
11 |
12 | url = 'http://www.baidu.com/s?{}'.format(params)
13 | url = url.format(params)
14 |
15 | # 2.发请求获取响应内容
16 | headers = {'User-Agent': 'Mozilla/5.0'}
```

```

17 | html = requests.get(url=url,headers=headers).content.decode('utf-8')
18 |
19 | # 3.保存到本地文件
20 | filename = word + '.html'
21 | with open(filename,'w',encoding='utf-8') as f:
22 |     f.write(html)

```

quote('参数为字符串')

■ 使用方法

```

1 | # 对单独的字符串进行编码 - URL地址中的中文字符
2 | word = '美女'
3 | result = urllib.parse.quote(word)
4 | result结果: '%E7%BE%8E%E5%A5%B3'

```

■ 练习

```

1 | """
2 | 问题：在百度中输入要搜索的内容，把响应内容保存到本地文件
3 | 编码方法使用 quote()
4 | """
5 | import requests
6 | from urllib import parse
7 |
8 | # 1.拼接URL地址
9 | word = input('请输入搜索内容:')
10 | params = parse.quote(word)
11 |
12 | url = 'http://www.baidu.com/s?wd={}'
13 | url = url.format(params)
14 |
15 | # 2.发请求获取响应内容
16 | headers = {'User-Agent':'Mozilla/5.0'}
17 | html = requests.get(url=url,headers=headers).content.decode('utf-8')
18 |
19 | # 3.保存到本地文件
20 | filename = word + '.html'
21 | with open(filename,'w',encoding='utf-8') as f:
22 |     f.write(html)

```

■ unquote(string)解码

```

1 | # 将编码后的字符串转为普通的Unicode字符串
2 | from urllib import parse
3 |
4 | params = '%E7%BE%8E%E5%A5%B3'
5 | result = parse.unquote(params)
6 |
7 | result结果: 美女

```

■ 小总结

```
1  【1】 什么是robots协议
2
3  【2】 requests模块使用
4      res = requests.get(url=url,headers={'User-Agent':'xxx'})
5      响应对象res属性:
6          a> res.text
7          b> res.content
8          c> res.status_code
9          d> res.url
10
11 【3】 urllib.parse
12     a> urlencode({'key1':'xxx','key2':'xxx'})
13     b> quote('xxx')
14     c> unquote('xxx')
15
16 【4】 URL地址拼接 - urlencode()
17     a> 'http://www.baidu.com/s?' + params
18     b> 'http://www.baidu.com/s?%s' % params
19     c> 'http://www.baidu.com/s?{}'.format(params)
```

案例 - 百度贴吧数据抓取

■ 需求

```
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: 2
4  4、保存到本地文件: 赵丽颖吧_第1页.html、赵丽颖吧_第2页.html
```

■ 实现步骤

```
1  【1】 查看所抓数据在响应内容中是否存在
2      右键 - 查看网页源码 - 搜索关键字
3
4  【2】 查找并分析URL地址规律
5      第1页: http://tieba.baidu.com/f?kw=???&pn=0
6      第2页: http://tieba.baidu.com/f?kw=???&pn=50
7      第n页: pn=(n-1)*50
8
9  【3】 发请求获取响应内容
10
11 【4】 保存到本地文件
```

■ 代码实现

```
1  import requests
2  from urllib import parse
3  import time
4  import random
```

```

5
6 class BaiduSpider(object):
7     def __init__(self):
8         self.url='http://tieba.baidu.com/f?kw={}&pn={}'
9         self.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)' }
10
11     def get_html(self,url):
12         """获取响应内容html"""
13         html = requests.get(url=url,headers=self.headers).content.decode('utf-8')
14
15         return html
16
17     def parse_html(self):
18         """解析提取数据"""
19         pass
20
21     def save_html(self,filename,html):
22         """处理数据"""
23         with open(filename,'w',encoding='utf-8') as f:
24             f.write(html)
25
26     def run(self):
27         """入口函数"""
28         name = input('请输入贴吧名:')
29         beign_page = int(input('请输入起始页:'))
30         end_page = int(input('请输入终止页:'))
31         # 对name进行编码
32         params = parse.quote(name)
33         for page in range(beign_page,end_page+1):
34             # 拼接URL地址
35             pn = (page-1)*50
36             url = self.url.format(params,pn)
37             # 请求+保存
38             html = self.get_html(url)
39             filename = '{}_第{}页.html'.format(name,page)
40             self.save_html(filename,html)
41             # 控制爬取频率:uniform(0,1)生成0-1之间浮点数
42             time.sleep(random.randint(1,2))
43             # time.sleep(random.uniform(0,1))
44             print('第%d页抓取完成' % page)
45
46 if __name__ == '__main__':
47     spider = BaiduSpider()
48     spider.run()

```

正则解析模块re

re模块使用流程

```

1 # 方法一
2 r_list=re.findall('正则表达式',html,re.S)
3
4 # 方法二
5 pattern = re.compile('正则表达式',re.S)
6 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('[\s\S]')
4 result = pattern.findall(html)
5
6 # 方法二
7 pattern = re.compile('.',re.S)
8 result = pattern.findall(html)

```

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: .*? .+? .??

■ 代码示例

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

正则表达式分组

作用

- 1 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

示例代码

```
1 import re
2
3 s = 'A B C D'
4 p1 = re.compile('\w+\s+\w+')
5 print(p1.findall(s))
6 # 分析结果是什么? ??
7 # ['A B','C D']
8
9 p2 = re.compile('(\w+)\s+\w+')
10 print(p2.findall(s))
11 # 第1步: ['A B','C D']
12 # 第2步: ['A','C']
13
14 p3 = re.compile('(\w+)\s+(\w+)')
15 print(p3.findall(s))
16 # 第1步: ['A B','C D']
17 # 第2步: [('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 [(,),(),()]

课堂练习

```

1 # 从如下html代码结构中完成如下内容信息的提取:
2 问题1 : [('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

```

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>

```

练习答案

```

1 import re
2
3 html = '''<div class="animal">
4     <p class="name">
5         <a title="Tiger"></a>
6     </p>
7
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 p = re.compile('<div class="animal">.*?title="(.*?)".*?content">(.*?)</p>.*?</div>', re.S)
24 r_list = p.findall(html)
25
26 for rt in r_list:
27     print('动物名称:', rt[0].strip())
28     print('动物描述:', rt[1].strip())
29     print('*' * 50)

```

猫眼电影top100抓取案例

爬虫需求

```

1  【1】 确定URL地址
2      百度搜索 - 猫眼电影 - 榜单 - top100榜
3
4  【2】 爬取目标
5      所有电影的 电影名称、主演、上映时间

```

爬虫实现

```

1  【1】 查看网页源码，确认数据来源
2      响应内容中存在所需抓取数据 - 电影名称、主演、上映时间
3
4  【2】 翻页寻找URL地址规律
5      第1页: https://maoyan.com/board/4?offset=0
6      第2页: https://maoyan.com/board/4?offset=10
7      第n页: offset=(n-1)*10
8
9  【3】 编写正则表达式
10     <div class="movie-item-info">.*?title="(.*?)".*?class="star">(.*?)</p>.*?releasetime">
11     (.*?)</p>
12
13 【4】 开干吧兄弟

```

代码实现

```

1
2 import requests
3 import re
4 import time
5 import random
6
7
8 class MaoyanSpider(object):
9     def __init__(self):
10         self.url = 'https://maoyan.com/board/4?offset={}'
11         self.headers = {
12             'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML,
13             like Gecko) Chrome/14.0.835.163 Safari/535.1'
14         self.i = 0

```

```

15     def get_html(self, url):
16         html = requests.get(url=url, headers=self.headers).text
17         # 直接调用解析函数
18         self.parse_html(html)
19
20     def parse_html(self, html):
21         """正则解析函数"""
22         regex = '<div class="movie-item-info">.*?title="(.*?)".*?class="star">(.*?)</p>.*?
release_time">(.*?)</p>'
23         pattern = re.compile(regex, re.S)
24         # dd_list: [(),(),()]
25         dd_list = pattern.findall(html)
26         self.save_html(dd_list)
27
28     def save_html(self, dd_list):
29         """保存数据函数"""
30         item = {}
31         for dd in dd_list:
32             item['name'] = dd[0].strip()
33             item['star'] = dd[1].strip()[3:]
34             item['time'] = dd[2].strip()[5:15]
35             print(item)
36             self.i += 1
37
38     def run(self):
39         for offset in range(0, 91, 10):
40             url = self.url.format(offset)
41             self.get_html(url)
42             time.sleep(random.randint(1, 3))
43             print('电影数量: ', self.i)
44
45
46 if __name__ == '__main__':
47     start_time = time.time()
48     spider = MaoyanSpider()
49     spider.run()
50     end_time = time.time()
51     print('执行时间:%.2f' % (end_time - start_time))

```

今日作业

- 把百度贴吧案例重写一遍,不要参照课上代码
- 猫眼电影案例重写一遍,不要参照课上代码
- 复习任务

- 1 pymysql、MySQL基本命令
- 2 MySQL : 建库建表普通查询、插入、删除等
- 3 Redis : python和redis交互,集合基本操作

■ 建库建表

```
1  【1】使用SQL命令完成
2      1.1) 创建库 maoyandb ,字符编码为 utf8
3      1.2) 在maoyandb库中创建表 maoyantab, 字段要求如下:
4          name 变长,宽度为100
5          star 变长,宽度为500
6          time 变长,宽度为100
7
8  【2】使用pymysql模块完成
9      在表maoyantab中插入1条表记录 (电影名称、主演、上映时间三个字段)
10
11 【3】将猫眼电影案例中所抓取的电影信息存入到MySQL数据库的maoyantab表中
```