王伟超

wangweichao@tedu.cn

DAY01

网络爬虫概述

- **定义**

```
网络蜘蛛、网络机器人，抓取网络数据的程序
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、公司项目测试数据
2、公司业务部门及其他部门所需数据
3、数据分析
```

- **企业获取数据方式**

```
1、公司自有数据
2、第三方数据平台购买(数据堂、贵阳大数据交易所)
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

```python
import urllib.request

# urlopen() ：  向URL发请求，返回响应对象
response=urllib.request.urlopen('http://www.baidu.com/')
# 提取响应内容
html = response.read().decode('utf-8')
# 打印响应内容
print(html)
```

- **响应对象（response）方法**

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

```
# 向测试网站： http://httpbin.org/get 发请求,查看自己请求头
# 代码如下
import urllib.request
response=urllib.request.urlopen('http://httpbin.org/get')
html = response.read().decode()
print(html)

# html中的请求头headers如下
"headers": {
    "Accept-Encoding": "identity",
    "Host": "httpbin.org",
    "User-Agent": "Python-urllib/3.6"
  },
发现请求头中User-Agent竟然是:Python-urllib/3.6!!!!!!!!!!!!!!!!!!!
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、url：请求的URL地址
2、headers：添加请求头（爬虫和反爬虫斗争的第一步）
```

- **使用流程**

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

```
from urllib import request

# 定义常用变量：URL 、headers
url = 'http://httpbin.org/get'
headers = {
  'User-Agent':'Opera/9.80 (Windows NT 6.1; U; zh-cn) Presto/2.9.168 Version/11.50'
```

```
}
# 1. 创建请求对象 - 包装,并没有真正发请求
req = request.Request(url=url,headers=headers)
# 2. 获取响应对象
res = request.urlopen(req)
# 3. 提取响应内容
html = res.read().decode('utf-8')

print(html)
```

URL地址编码模块

模块名及导入

- **模块**

```
# 模块名
urllib.parse

# 导入
import urllib.parse
from urllib import parse
```

- **作用**

给URL地址中查询参数进行编码

```
编码前: https://www.baidu.com/s?wd=美女
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**

```
# 查询参数: {'wd' : '美女'}
# urlencode编码后: 'wd=%e7%be%8e%e5%a5%b3'

# 示例代码
query_string = {'wd' : '美女'}
result = urllib.parse.urlencode(query_string)
# result: 'wd=%e7%be%8e%e5%a5%b3'
```

- **URL地址中多个查询参数**

```
from urllib import parse
params = {
  'wd' : '美女',
  'pn' : '50'
}
params = parse.urlencode(query_string_dict)
url = 'http://www.baidu.com/s?{}'.format(params)
print(url)
```

- **拼接URL地址的3种方式**

```
# 1、字符串相加
  baseurl = 'http://www.baidu.com/s?'
  params = 'wd=%E7XXXX&pn=20'
  url = baseurl + params

# 2、字符串格式化（占位符）
  params = 'wd=%E7XXXX&pn=20'
  url = 'http://www.baidu.com/s?%s'% params

# 3、format()方法
  url = 'http://www.baidu.com/s?{}'
  params = 'wd=#E7XXXX&pn=20'
  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

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

代码实现 - 03_parse_baidu.py

```
from urllib import request
from urllib import parse

# 拼接URL地址
def get_url(word):
  url = 'http://www.baidu.com/s?{}'
  # params: wd=%E7XXXXX
  params = parse.urlencode({'wd':word})
  url = url.format(params)

  return url


# 发请求,保存本地文件
def request_url(url,filename):
  headers = {'User-Agent':'Mozilla/5.0'}
  # 请求对象 + 响应对象 + 提取内容
  req = request.Request(url=url,headers=headers)
  res = request.urlopen(req)
```

```
    html = res.read().decode('utf-8')
    # 保存数据
    with open(filename,'w',encoding='utf-8') as f:
        f.write(html)

# 主程序入口
if __name__ == '__main__':
    word = input('请输入搜索内容:')
    url = get_url(word)
    filename = word + '.html'
    request_url(url,filename)
```

quote(string)编码

- **示例1**

```
from urllib import parse

string = '美女'
print(parse.quote(string))
# 结果: %E7%BE%8E%E5%A5%B3
```

改写之前urlencode()代码，使用quote()方法实现

```
from urllib import parse

url = 'http://www.baidu.com/s?wd={}'
word = input('请输入要搜索的内容:')
query_string = parse.quote(word)
print(url.format(query_string))
```

unquote(string)解码

- **示例**

```
from urllib import parse

string = '%E7%BE%8E%E5%A5%B3'
result = parse.unquote(string)
print(result)
```

总结

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

# 2、响应对象res方法
res.read()
res.getcode()
```

```
res.geturl()

# 3、urllib.parse
urllib.parse.urlencode({})
urllib.parse.quote(string)
urllib.parse.unquote(string)
```

百度贴吧数据抓取案例

要求

```
1、输入贴吧名称:赵丽颖吧
2、输入起始页:1
3、输入终止页:3
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、查看是否为静态页面**

```
右键 - 查看网页源代码 - 搜索数据关键字
```

- **2、找URL规律**

```
第1页:http://tieba.baidu.com/f?kw=? ? &pn=0
第2页:http://tieba.baidu.com/f?kw=? ? &pn=50
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

```
from urllib import request,parse
import time
import random
from useragents import ua_list

class BaiduSpider(object):
  def __init__(self):
    self.url = 'http://tieba.baidu.com/f?kw={}&pn={}'

# 获取
  def get_html(self,url):
    headers = {
      'User-Agent': random.choice(ua_list)
    }
```

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

        return html

    # 解析
    def parse_html(self):
        pass

    # 保存
    def write_html(self,filename,html):
        with open(filename,'w',encoding='utf-8') as f:
            f.write(html)

# 入口函数
    def run(self):
        name = input('请输入贴吧名:')
        begin = int(input('请输入起始页:'))
        end = int(input('请输入终止页:'))

        # 查询参数编码

        params = parse.quote(name)
        for page in range(begin,end+1):
            # URL拼接
            pn = (page-1)*50
            pn = (page-1)*50
            url = self.url.format(params,pn)

            # 调用类内函数进行页面抓取+保存
            filename = '{}-第{}页.html'.format(name,page)
            html = self.get_html(url)
            self.write_html(filename,html)
            # 控制爬取速度
            time.sleep(random.randint(1,3))
            print('第%d页爬取完成' % page)

if __name__ == '__main__':
    start = time.time()
    spider = BaiduSpider()
    spider.run()
    end = time.time()
    print('执行时间:%.2f' % (end-start))
```

正则解析模块

re模块使用流程

- **方法一**

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

- **方法二**

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

```
import re
# 方法一
pattern = re.compile('.',re.S)
# 方法二
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

```
import re

html = '''
<div><p>九霄龙吟惊天变</p></div>
```

```
<div><p>风云际汇潜水游</p></div>
'''
# 贪婪匹配 ：  .*
pattern = re.compile('<div><p>.*</p></div>',re.S)
r_list = pattern.findall(html)
print(r_list)

# 非贪婪匹配 ：.*?
pattern = re.compile('<div><p>.*?</p></div>',re.S)
r_list = pattern.findall(html)
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**

```
import re

s = 'A B C D'
p1 = re.compile('\w+\s+\w+')
print(p1.findall(s))
# 结果: ['A B','C D']

p2 = re.compile('(\w+)\s+\w+')
print(p2.findall(s))
# 结果: ['A','C']

p3 = re.compile('(\w+)\s+(\w+)')
print(p3.findall(s))
# 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
   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：

```
# <div class="animal">.*?title="(.*?)".*?
<div class="animal">
    <p class="name">
    <a title="Tiger"></a>
    </p>
    <p class="content">
    Two tigers two tigers run fast
```

```
    </p>
</div>

<div class="animal">
    <p class="name">
    <a title="Rabbit"></a>
    </p>

    <p class="content">
    Small white rabbit white and white
    </p>
</div>
```

从以上html代码结构中完成如下内容信息的提取：

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

代码实现 - 06_re_exercise.py

```
import re

html = '''
<div class="animal">
    <p class="name">
      <a title="Tiger"></a>
    </p>
    <p class="content">
      Two tigers two tigers run fast
    </p>
</div>

<div class="animal">
    <p class="name">
      <a title="Rabbit"></a>
    </p>

    <p class="content">
      Small white rabbit white and white
    </p>
</div>
'''
pattern = re.compile(r'<div class="animal">.*?<a title="(.*?)".*?content">(.*?)
</p>',re.S)
r_list = pattern.findall(html)
# 问题1
```

```
if r_list:
  print(r_list)
# r_list: [('Tiger','\n\t\t Two tigers xxx'),()]
# 问题2
if r_list:
    for r in r_list:
      print('动物名称:',r[0].strip())
      print('动物描述:',r[1].strip())
      print('*' * 50)
else:
    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

```
第1步完成:
  猫眼电影-第1页.html
  猫眼电影-第2页.html
  ... ...

第2步完成:
  1、提取数据 ： 电影名称、主演、上映时间
  2、先打印输出,然后再写入到本地文件
```

3、复习任务

```
pymysql、MySQL基本命令
MySQL  ： 建库建表普通查询等
```