第三周：网络爬虫之实战

第一节：Re（正则表达式）库入门

正则表达式是用来简洁表达一组字符串的表达式。它可以用来表达文本类型的特征。

正则表达式编译：将符合正则表达式语法的字符串转换成正则表达式特征。

1.正则表达式的语法

例如：

P(Y|YT|YTH|YTHO)?N

正则表达式语法由字符和操作符构成。正则表达式的常用操作符见表 1.1 所示。

表 1.1 正则表达式的常用操作符

操作符	说明	实例
.	表示任何单个字符	
[]	字符集，对单个字符给出取值范围	[abc]表示 a,b,c，[a-z]表示 a 到 z 单个字符
[^]	非字符集，对单个字符给出排除范围	[^abc]表示非 a 或 b 或 c 的单个字符
*	前一个字符 0 次或无限次扩展	abc*表示 ab,abc,abcc,abccc 等
+	前一个字符 1 次或无限次扩展	abc+表示 abc,abcc,abccc 等
?	前一个字符 0 次或 1 次扩展	abc?表示 ab，abc
|	左右表达式任意一个	abc|def 表示 abc 或 def
{m}	扩展前一个字符 m 次	ab{3}c 表示 abbbc
{m,n}	扩展前一个字符 m 至 n（含 n）次	ab{1,2}c 表示 abc,abbc
^	匹配字符串开头	^abc 表示 abc 且在第一个字符串的开头
$	匹配字符串结尾	abc$表示 abc 且在一个字符串的结尾
()	分组标记，内部只能使用|操作符	(abc)表示 abc，(abc|def)表示 abc、def
\d	数字，等价于[0-9]	
\w	单词字符，等价于[A-Za-z0-9_]	

表 1.2 经典正则表达式实例

^[A-Za-z]+$	由 26 个字母组成的字符串
^[A-Za-z0-9]+$	由 26 个字母和数字组成的字符串
^-?\d+$	整数形式的字符串
^[0-9]*[1-9][0-9]*$	正整数形式的字符串
[1-9]\d{5}	中国境内邮政编码，6 位
[\u4e00-\u9fa5]	匹配中文字符
\d{3}-\d{8}|\d{4}-\d{7}	国内电话号码（**3 位区号-8 位**或 **4 位区号-7 位**）

2.Re 库的基本使用

（1）.正则表达式的表示类型

①raw string 类型（原生字符串类型）

re 库采用 raw string 类型表示正则表达式，表示为：r'text'。例如中国境内邮政编码可以表示为：r'[1-9]\d{5}'。

raw string（原生字符串）是不包含转移符的字符串。Re 库的主要功能函数如表 1.3 所示。

表 1.3 Re 库主要功能函数

函数	说明
re.search()	在一个字符串中搜索匹配正则表达式的第一个位置，返回 match 对象。
re.match()	从第一个字符串的**开始位置**其匹配正则表达式，返回 match 对象。
re.findall()	搜索字符串，以列表类型返回全部能匹配的子串。
re.split()	将一个字符串按照正则表达式匹配结果进行分隔，返回列表类型，**匹配的部分去掉**。
re.finditer()	搜索字符串，返回一个匹配结果的迭代类型，每个元素都是 match 对象。
re.sub()	在一个字符串中替换所有匹配正则表达式的子串，返回替换后的字符串。

（2）功能函数的基本介绍

①re.search(pattern, string, flags=0)

在一个字符串中搜索匹配正则表达式的第一个位置，返回 match 对象。后面参数中，pattern 是正则表达式的字符串或原生字符串表示；string 是待匹配字符串，flags 是使用时的控制标记。标记包括 re.I（re.IGNORECASE）忽略正则表达式的大小写，[A-Z]能够匹配小鞋字符；re.M(re.MULTILINE)正则表达式中的^操作符能够将给定字符串的每行当作匹配开始；re.S(re.DOTALL)正则表达式中的.操作符能够匹配所有字符，默认匹配除换行外的所有字符。

②re.findall(pattern, string, flags=0)

例程：

```
import re
ls =re.findall(r'[1-9]\d{5}','BIT 100081 TSU100084 XDU710071')
print(ls)
```

结果：

```
['100081', '100084', '710071']
```

③re.split(pattern, string, maxsplit=0, flags=0)

maxsplit 表示最大分个数，剩余部分作为最后一个元素输出。

代码：

```
import re
ls =re.split(r'[1-9]\d{5}','BIT 100081 TSU100084 XDU710071',maxsplit =1)
#只匹配第一个
print(ls)
```

结果：

```
['BIT ', ' TSU100084 XDU710071']
```

④re.finditer(pattern, string, flags=0)

代码：

```
import re
for m in re.finditer(r'[1-9]\d{5}','BIT 100081 TSU100084 XDU710071'):
    if m:
        print(m.group(0))
```

结果：

```
100081
100084
710071
```

⑤re.sub(pattern,rep,string, count=0,flags=0)

rep 是替换匹配字符串的字符串，count 代表匹配的最大次数。

例程：

```
import re
print(re.sub(r'[1-9]\d{5}', 'xiaoao','BIT 100081 TSU100084 XDU710071',count=1))
```

结果：

```
BIT xiaoao TSU100084 XDU710071
```

（3）Re 库的另一种等价用法

例如，原来我们使用这种函数式用法，这是一种一次性操作。

```
rst = re.search(r'[1-9]\d{5}','BIT 100081 TSU100084 XDU710071')
```

现在我们可以采用面向对象用法，可以在编译后多次操作。

```
pat = re.compile(r' [1-9]\d{5}')
rst = pat.search('BIT 100081 TSU100084 XDU710071')
```

regex = re.compile(pattern, flags=0)

将正则表达式的字符串形式编译成正则表达式对象。

3.Re 库的 match 对象

match 对象的属性如表 1.4 所示。

表 1.4 Match 对象的属性

属性	说明
.string	待匹配的文本
.re	匹配使用的 pattern 对象（正则表达式）
.pos	正则表达式搜索文本的开始位置
.endpos	正则表达式搜索文本的结束位置
.group(0)	返回匹配后的字符串
.start()	匹配字符串在原始字符串的开始位置
.end()	匹配字符串在原始字符串的结束位置
.span()	返回(.start(),.end())，是**元组类型**。

4.Re 库的贪婪匹配和最小匹配

（1）贪婪匹配

Re 库默认采用贪婪匹配，即输出匹配最长的子串。例如输入：

```
import re
match = re.search(r'PY.*N', 'PYANBNCNDN')
print(match.group(0))
```

结果为：

```
PYANBNCNDN
```

（2）最小匹配

代码如下：

```
import re
match = re.search(r'PY.*?N', 'PYANBNCNDN')
print(match.group(0))
```

结果：

```
PYAN
```

最小匹配操作符如表 1.5 所示。

表 1.5 最小匹配操作符

操作符	说明
*?	前一个字符 0 次或无限次扩展，最小匹配。
+?	前一个字符 1 次或无限次扩展，最小匹配。
??	前一个字符 0 次或 1 次扩展，最小匹配。
{m,n,}?	扩展前一个字符 m 至 n 次（含 n），最小匹配。

第二节：实例 2：淘宝商品比价定向爬虫

1.实例介绍

功能描述：

目标：获取淘宝搜索页面的信息，提取其中的商品名称和价格。

理解：淘宝的搜索接口；翻页的处理。

技术路线：requests 库+re 库。

程序结构设计

步骤 1：提交商品搜索请求，循环获取页面。

步骤 2：对于每个页面，提取商品名称和价格信息。

步骤 3：将信息输出到屏幕上。

2.实例编写

代码：

```
import requests
import re

def getHTMLText(url):
    try:
        r = requests.get(url, timeout=30)
        r.raise_for_status()
        r.encoding = r.apparent_encoding
        return r.text
    except:
        return ""
def parsePage(ilt, html):
    try:
        plt = re.findall(r'\"view_price\"\:\"[\d\.]*\"',html)
        tlt = re.findall(r'\"raw_title\"\:\".*?\"',html)
        for i in range(len(plt)):
            price = eval(plt[i].split(':')[1])
            title = eval(tlt[i].split(':')[1])
            ilt.append([price , title])
    except:
        print("")
def printGoodsList(ilt):
```

```
    tplt = "{:4}\t{:8}\t{:16}"
    print(tplt.format("序号", "价格", "商品名称"))
    count = 0
    for g in ilt:
        count = count + 1
        print(tplt.format(count, g[0], g[1]))

def main():
    goods='书包'
    depth = 1
    start_url = 'https://s.taobao.com/search?q=' + goods
    infoList = []
    for i in range(depth):
        try:
            url = start_url + '&s=' + str(44*i)
            html = getHTMLText(url)
            parsePage(infoList, html)
        except:
            continue
    printGoodsList(infoList)

main()
```

结果（2018 年 5 月 7 日 22:08 结果）:

```
序号      价格          商品名称
   1      79.00         小学生大空间容量女生公主书包双肩背包减负
   2      69.00         艾奔 双肩包男背包女韩版潮 商务男士电脑包高中学生书包休闲旅行
   3      39.90         迪卡侬双肩包男女学生书包健身包运动新款校园 20 升休闲 KIPSTA
   4      59.80         商务背包男士双肩包韩版潮流旅行包休闲女学生书包简约时尚电脑包
   5      49.00         书包小学生 1-2-3-6 年级男女生减负双肩儿童书包男孩防水护脊背包
   6      69.00         双肩包男士背包时尚潮流初中高中学生书包大学生女商务旅行电脑包
   7      59.00         男背包旅游休闲商务电脑韩版时尚潮流大高中学生书包旅行双肩包
   8      59.00         背包男韩版时尚潮流双肩包休闲商务电脑旅行大容量高中大学生书包
   9      129.00        七匹狼商务双肩包男书包中学生女电脑包旅行包休闲男士背包大容量
  10      99.00         米熙休闲运动背包双肩包女书包中学生男时尚潮流大容量旅游旅行包
  11      138.00        双肩包女背包 2018 新款韩版潮牛津布帆布百搭防盗女士旅行休
```

闲包包
12 69.00 2018 新款双肩包女牛津布韩版潮百搭背包时尚休闲书包女包包旅行包
13 119.00 巴朗新款商务双肩包休闲时尚潮流大学生书包 15.6 寸电脑包男士背包
14 55.00 双肩包女包包 2018 新款韩版潮时尚个性百搭 2017 牛津帆布小书包背包
15 129.00 七匹狼双肩包男 韩版女休闲潮包 学生书包电脑包旅行包双肩背包男
16 69.00 双肩包女 2018 新款包包韩版百搭时尚背包女双肩软皮旅行包书包女包
17 49.00 背包男双肩包高中初中学生韩版书包男时尚潮流青年电脑旅游旅行包
18 149.00 小米双肩包简约休闲多功能书包男女笔记本电脑包时尚潮流旅行背包
19 69.00 双肩包女包包 2018 新款韩版潮个性百搭时尚迷你小 2017 书包女士背包
20 139.00 施维茨男士双肩包大容量背包女商务电脑旅行包休闲潮高中学生书包
21 139.00 瑞士军刀双肩包男瑞士电脑旅行背包女大容量高中大学初中学生书包
22 139.00 七匹狼背包男商务双肩包男士旅行休闲时尚潮流电脑青年书包大容量
23 139.00 赫登尔双肩包男士潮流背包男韩版旅行包时尚休闲学生书包电脑包潮
24 125.00 瑞士军刀双肩包瑞士中学生书包女休闲男士商务旅行大容量电脑背包
25 208.00 【香港直邮】anello 双肩包男女离家出走包学生书包旅行包休闲背包
26 79.00 书包小学生男 1-3-4-6 年级儿童减负护脊女 6-12 周岁初中学生双肩包
27 119.00 韩国 kk 树书包小学生男 1-3-4-6 年级儿童书包女 6-12 周岁双肩包护脊
28 79.00 书包女韩版原宿 ulzzang 高中学生 2018 新款初中生校园甜甜圈双肩包
29 118.00 瑞士军刀双肩包男士背包大容量休闲商务电脑包韩版高中书包旅行包
30 149.00 双肩包男士时尚潮流背包韩版个性书包男大学生运动休闲电脑旅行包
31 89.00 双肩包男帆布背包电脑包休闲防水运动后背包男旅行包青年学生书包
32 208.00 日本 anello 双肩包女防水牛津布乐天背包学生书包离家出走包alleno
33 158.00 专柜正品 JanSport 杰斯伯经典双肩背包男女同款学生书包 T501

纯色
34 69.00 休闲双肩包韩版简约旅行背包电脑初中高中学生校园书包男时尚潮流
35 129.00 男士双肩包男韩版时尚背包男大学生书包潮流旅行包电脑包男包皮包
36 189.00 商务双肩包男大容量青年短途出差旅行包 17 寸电脑包多功能防盗背包

第三节：实例 3：股票数据定向爬虫

1.实例介绍

功能描述：

目标：获取上交所和深交所所有股票的名称和交易信息。

输出：保存在文件中。

技术路线：requests 库++bs4 库+re 库。

候选数据网站选择：

选取原则：信息静态存在于 HTML 页面中，非 js 代码生成，没有 Robots 协议限制。

选取方法：查看源代码等。

选取心态：不要纠结于某个网站，多找信息源尝试。

程序结构设计

步骤 1：从东方财富网获取股票列表。

步骤 2：根据股票列表逐个到百度股票获取个股信息。。

步骤 3：将结果存储到文件中。

2.实例编写

代码：

```
import requests
from bs4 import BeautifulSoup
import traceback
import re

def getHTMLText(url):
    try:
        r = requests.get(url)
        r.raise_for_status()
        r.encoding = r.apparent_encoding
        return r.text
    except:
        return ""

def getStockList(lst, stockURL):
    html = getHTMLText(stockURL)
    soup = BeautifulSoup(html, 'html.parser')
    a = soup.find_all('a')
    for i in a:
```

```
            try:
                href = i.attrs['href']
                lst.append(re.findall(r"[s][hz]\d{6}", href)[0])
            except:
                continue

def getStockInfo(lst, stockURL, fpath):
    for stock in lst:
        url = stockURL + stock + ".html"
        html = getHTMLText(url)
        try:
            if html=="":
                continue
            infoDict = {}
            soup = BeautifulSoup(html, 'html.parser')
            stockInfo = soup.find('div',attrs={'class':'stock-bets'})

            name = stockInfo.find_all(attrs={'class':'bets-name'})[0]
            infoDict.update({'股票名称': name.text.split()[0]})

            keyList = stockInfo.find_all('dt')
            valueList = stockInfo.find_all('dd')
            for i in range(len(keyList)):
                key = keyList[i].text
                val = valueList[i].text
                infoDict[key] = val

            with open(fpath, 'a', encoding='utf-8') as f:
                f.write( str(infoDict) + '\n' )
        except:
            traceback.print_exc()
            continue

def main():
    stock_list_url = 'http://quote.eastmoney.com/stocklist.html'
    stock_info_url = 'http://gupiao.baidu.com/stock/'
    output_file = 'D:/BaiduStockInfo.txt'
    slist=[]
    getStockList(slist, stock_list_url)
    getStockInfo(slist, stock_info_url, output_file)

main()
```

3.编写优化

代码:

```
import requests
from bs4 import BeautifulSoup
import traceback
import re

def getHTMLText(url, code="utf-8"):
    try:
        r = requests.get(url)
        r.raise_for_status()
        r.encoding = code
        return r.text
    except:
        return ""

def getStockList(lst, stockURL):
    html = getHTMLText(stockURL, "GB2312")
    soup = BeautifulSoup(html, 'html.parser')
    a = soup.find_all('a')
    for i in a:
        try:
            href = i.attrs['href']
            lst.append(re.findall(r"[s][hz]\d{6}", href)[0])
        except:
            continue

def getStockInfo(lst, stockURL, fpath):
    count = 0
    for stock in lst:
        url = stockURL + stock + ".html"
        html = getHTMLText(url)
        try:
            if html=="":
                continue
            infoDict = {}
            soup = BeautifulSoup(html, 'html.parser')
            stockInfo = soup.find('div',attrs={'class':'stock-bets'})

            name = stockInfo.find_all(attrs={'class':'bets-name'})[0]
            infoDict.update({'股票名称': name.text.split()[0]})

            keyList = stockInfo.find_all('dt')
            valueList = stockInfo.find_all('dd')
            for i in range(len(keyList)):
                key = keyList[i].text
```

```
                    val = valueList[i].text
                    infoDict[key] = val

                with open(fpath, 'a', encoding='utf-8') as f:
                    f.write( str(infoDict) + '\n' )
                    count = count + 1
                    print("\r 当前进度: {:.2f}%".format(count*100/len(lst)),end="")
            except:
                count = count + 1
                print("\r 当前进度: {:.2f}%".format(count*100/len(lst)),end="")
                continue

def main():
    stock_list_url = 'http://quote.eastmoney.com/stocklist.html'
    stock_info_url = 'http://gupiao.baidu.com/stock/'
    output_file = 'BaiduStockInfo.txt'
    slist=[]
    getStockList(slist, stock_list_url)
    getStockInfo(slist, stock_info_url, output_file)

main()
```