第71天： Python Scrapy 项目实战

原创：戴景波 Python技术 4天前

爬虫编写流程

首先明确 Python 爬虫代码编写的流程：先直接打开网页，找到你想要的数据，就是走一遍流程。比如这个项目我要爬取历史某一天所有比赛的赔率数据、每场比赛的比赛结果等。

那么我就先打开这个网址：https://live.leisu.com/wanchang?date=20190606 然后点击“竞彩”，再点击“指数”，跳转到另一个网址：https://live.leisu.com/3in1-2674547，然后就看到了想要的数据：各公司主队获胜赔率1.61、1.65等。

到此为止，开始动手通过代码实现这个过程。

解析“爬虫主程序.py”：（主程序包括四个函数）

start_requests

向 https://live.leisu.com/wanchang?date=20190606 发送请求。（你可以打开这个网址，里边是爬虫程序爬取数据的最外层网站） scrapy.http.FormRequest 方法：第一个参数是请求的具体网址；第二个参数是下一步调用的函数；第三个参数 meta 是向调用函数传递的参数。

parseLs（parseWl 同理，不再重复讲解）

主要用于解析次外层网页数据。这里用 XPath 解析，也是比较容易掌握的解析方式。网页结构如下：（通过 Google 浏览器打开https://live.leisu.com/wanchang?date=20190606 然后右键点击网页空白处点击“查看网页源代码”，找到你需要爬取的核心数据部分，这里我要找每场比赛的信息，那么拷贝下来，然后以易于查看的规整方式列出，如下：）

```
<li class="list-item list-item-2674547 list-day-6-6 finished " data-id="2674547" data-status="8" (
<div class="find-table layout-grid-tbody hide">
<div class="clearfix-row">
...
<span class="lab-round"> 0</span>
<span class="lab-lottery">
<span class="text-jc">周三001</span>
<span class="text-bd">北单018</span>
<span class="text-zc"></span>
</span>
......
```

parseLS函数里的下边代码，结合上表中 xml 中的元素：获取了比赛场次，存储到item['cc']。

```
sel_div=sel('//li[@data-id='+str(raceid[0])+']/div[@class="find-table layout-grid-tbody hide"]/div
if str(sel_div.xpath('span[@class="lab-lottery"]/span[@class="text-jc"]/text()').extract()) == "[]"
    item['cc']=""
else:
    item['cc']=str(d2) + str(sel_div.xpath('span[@class="lab-lottery"]/span[@class="text-jc"]/text
```

此外，还要获取比赛的赔率信息，但并不在当前这个网页，而在更内层的网页中，需要从当前网页跳转。存储赔率的内层网页为 https://live.leisu.com/3in1-2674547，不同场次的比赛只有-后边的数字是变化的，那么程序中只要循环构造对应的数字2674547就好了。发现这个数字刚好是 data-id。通过以下代码实现获取：

```
racelist=[e5.split("'") for e5 in sel('//li[@data-status="8"]/@data-id').extract()]
for raceid in racelist:
    plurl='https://live.leisu.com/3in1-'+raceid[0]
    request = scrapy.http.FormRequest(plurl,callback=self.parse,meta={'item':item})
    yield request
```

再提交该网页请求到下一个函数parse。

parse

网页结构如下：（通过Google浏览器打开https://live.leisu.com/3in1-2674547 然后右键点击网页空白处点击“查看网页源代码”，拷贝需要赔率的部分到文本文档，换行操作后如下：

```
<tr class="td-data td-pd-8 f-s-12 color-666 bd-top " data-id="4">
<td>
......
<td class="bd-left">
<div class="begin float-left w-bar-100 bd-bottom p-b-8 color-999 m-b-8">
<span class="float-left col-3"> 1.620 </span>
<span class="float-left col-3"> 3.600 </span>
<span class="float-left col-3"> 5.250 </span>
</div>
......
```

通过以下代码获取赔率：

```
pl_str = '/td[@class="bd-left"]/div[@class="begin float-left w-bar-100 bd-bottom p-b-8 color-999 m
if str(pv('//*[@data-id="5"]'+pl_str).extract())=="[]":
     item['li'] =  ''
else:
     item['li']=pv('//*[@data-id="5"]' + pl_str).extract()[0]
```

总结

以上我们实现了一个爬虫实战项目，通过分析网页结构，借助 Scrapy 框架获取数据，为今后的数据分析做准备。

代码地址

本篇的全部源码（可执行）：(github.com.cn/acredjb/FBP有完整项目爬虫源码)

> 示例代码：https://github.com/JustDoPython/python-100-day/tree/master/day-071

PS：公号内回复 ：Python，即可进入Python 新手学习交流群，一起**100天计划！**

-END-

Python 技术

关于 Python 都在这里