

bs4模块安装和使用

- bs4模块安装

在python中我一般只推荐用pip进行安装. 原因: 简单!!!!

```
1 pip install bs4
```

如果安装的速度慢, 建议更换国内源(推荐阿里源或者清华源)

```
1 pip install -i  
  https://pypi.tuna.tsinghua.edu.cn/simple bs4
```

- 如何使用bs4

bs4在使用的时候就需要参照一些html的基本语法来进行使用了. 我们直接上案例哈. 案例是最能直观的展现出bs4的便捷效果的.

我们来尝试抓取北京新发地市场的农产品价格. <http://www.xinfadi.com.cn/marketanalysis/0/list/1.shtml>

每日价格行情 - 北京新发地市场

2020年12月23日 农历十一月初九 星期三 北京 晴 -1°C ~ -5°C

品名	最低价	平均价	最高价	规格	单位	发布日期
羊腰子 (挂油)	17.00	17.00	17.00	普通	个	2020-12-23
荸荠	1.80	1.90	2.00	统	斤	2020-12-23
油菜菜	2.40	2.50	2.60	统	斤	2020-12-23
玉米	2.80	3.00	3.20	黄	斤	2020-12-23
彩椒	5.00	5.50	6.00	红/黄	斤	2020-12-23
青蒜	2.00	2.20	2.40	统	斤	2020-12-23
西洋芹	1.80	2.10	2.40	普通	斤	2020-12-23
茭白	3.80	3.90	4.00	京浙	斤	2020-12-23
藕子	1.50	2.00	2.50	普通	斤	2020-12-23
苤蓝	2.50	2.65	2.80	普通	斤	2020-12-23
凉薯	1.70	1.75	1.80	普通	斤	2020-12-23
佛手瓜	1.80	1.90	2.00	普通	斤	2020-12-23
黄心菜	0.80	0.90	1.00	普通	斤	2020-12-23
水萝卜	1.70	1.75	1.80	普通	斤	2020-12-23
兔子姜	0.70	0.80	0.90	普通	斤	2020-12-23
紫菜苔	5.60	5.80	6.00	普通	斤	2020-12-23
雪里蕻	0.40	0.50	0.60	普通	斤	2020-12-23
大白菜	0.40	0.50	0.60	/黄心	斤	2020-12-23
娃娃菜	0.60	0.70	0.80	大小	斤	2020-12-23
芹菜	1.40	1.60	1.80	统	斤	2020-12-23

老规矩, 先获取页面源代码. 并且确定数据就在页面源代码中~

```

1 import requests
2 from bs4 import BeautifulSoup
3
4 resp =
    requests.get("http://www.xinfadi.com.cn/market
    analysis/0/list/1.shtml")
5 print(resp.text)

```

将页面源代码丢给BeautifulSoup, 然后我们就可以通过bs对象去检索页面源代码中的html标签了

```
1 page = BeautifulSoup(resp.text)
```

BeautifulSoup对象获取html中的内容主要通过两个方法来完成

- find()

▪ find_all()

```
m find(self, name, attrs, recursive, text, kwargs) Tag
m find_all(self, name, attrs, recursive, text, limit, kwargs) Tag
m find_next(self, name, attrs, text, kwargs) PageElement
f findAll Tag
f findChild Tag
m find_all_next(self, name, attrs, text, limit, kwargs) PageElement
m find_all_previous(self, name, attrs, text, limit, kw... PageElement
m find_next_sibling(self, name, attrs, text, kwargs) PageElement
m find_next_siblings(self, name, attrs, text, limit, k... PageElement
m find_parent(self, name, attrs, kwargs) PageElement
m find_parents(self, name, attrs, limit, kwargs) PageElement
m find_previous(self, name, attrs, text, kwargs) PageElement
m find_previous_sibling(self, name, attrs, text, kwarg... PageElement
```

Press ^ to choose the selected (or first) suggestion and insert a dot afterwards Next Tip

基本上有这两个方法就够用了. 其他的可以自行进行英文翻译就知道啥意思了.

不论是find还是find_all 参数几乎是一致的.

语法:

find(标签, 属性=值)

意思是在页面中查找 xxx标签, 并且标签的xxx属性必须是xxx值

例:

find('div', age=18) 含义: 在页面中查找div标签, 并且属性age必须是18的这个标签.

find_all()的用法和find()几乎一致. find()查找1个. find_all()查找页面中所有的.

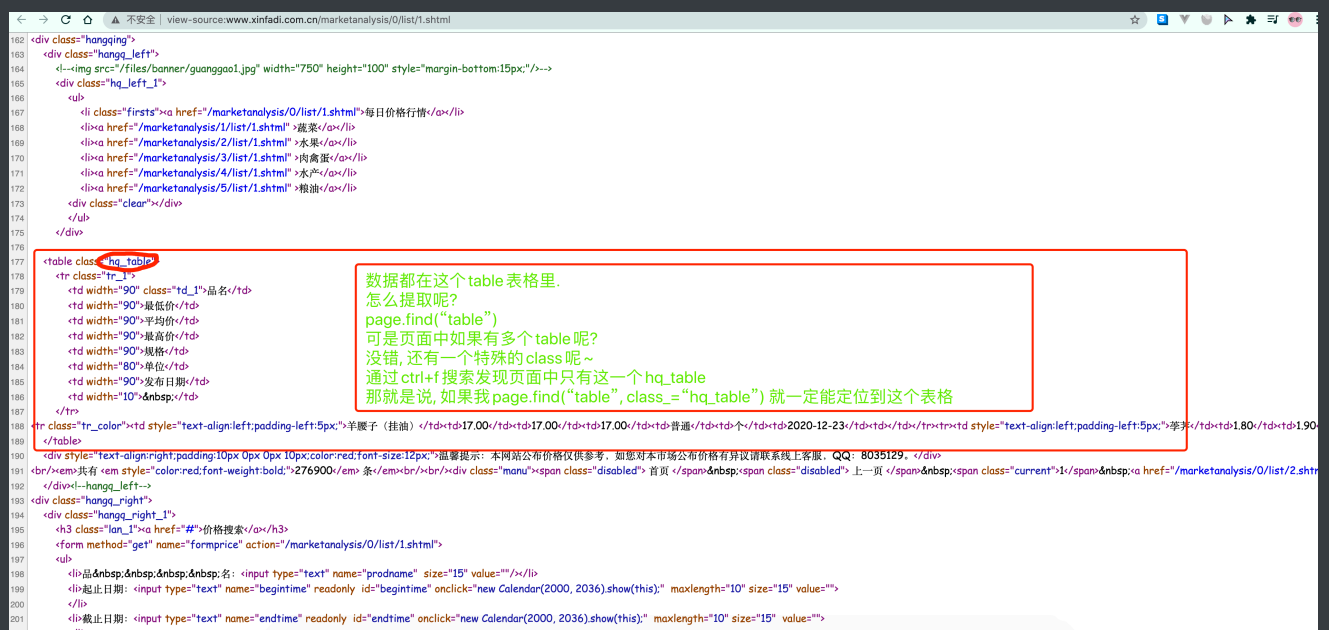
但是这种写法会有些问题. 比如html标签中的class属性.

```
1 <div class="honor">
2
3 page.find("div", class="honor")
4 注意，python中class是关键字．会报错的．怎么办呢？可
  以在class后面加个下划线
5 page.find("div", class_="honor")
```

我们可以使用第二种写法来避免这类问题出现

```
1 page.find("div", attrs={"class": "honor"})
```

好了, 用法说完了. 接下来就回来看怎么抓取新发地的价格吧



数据都在这个table表格里.
怎么提取呢?
page.find("table")
可是页面中如果有多个table呢?
没错, 还有一个特殊的class呢~
通过ctrl+f搜索发现页面中只有这一个hq_table
那就是说, 如果我page.find("table", class_="hq_table")就一定定位到这个表格

```
1 table = page.find("table", class_="hq_table")
2 print(table)
```

```
06_bs4模块使用.py x
1 import requests
2 from bs4 import BeautifulSoup
3
4 resp = requests.get("http://www.xinfadi.com.cn/marketanalysis/0/list/1.shtml")
5
6 page = BeautifulSoup(resp.text)
7
8 table = page.find("table", class_="hq_table")
9 print(table)
10

Run: 06_bs4模块使用 x
<table class="hq_table">
<tr class="tr_1">
<td class="td_1" width="90">品名</td>
<td width="90">最低价</td>
<td width="90">平均价</td>
<td width="90">最高价</td>
<td width="90">规格</td>
<td width="80">单位</td>
<td width="90">发布日期</td>
<td width="10"> </td>
</tr>
<tr class="tr_color"><td style="text-align:left;padding-left:5px;">羊腰子（挂油）</td><td>17.00<
</table>
```

完美~~

接下来就可以进一步去提取数据了. 后面的直接给出完整代码. 因为逻辑都是一样的. 并没有多么的复杂, 过程就省略了. 如果顺不下来, 去看视频吧. 视频会讲解的详细一些

```
1 import requests
2 from bs4 import BeautifulSoup
3 import csv
4
5 resp =
    requests.get("http://www.xinfadi.com.cn/marke
tanalysis/0/list/1.shtml")
6
7 page = BeautifulSoup(resp.text)
8
```

```
9 table = page.find("table", class_="hq_table")
10
11 f = open("新发地.csv", mode="w",
12         encoding="utf-8")
13 cv_writer = csv.writer(f)
14 # 提取到所有tr
15 tr_list = table.find_all("tr")[1:] # 注意,第一
16                                     # 行并不是我想要的数据。(第一行是表头)
17 for tr in tr_list:
18     td_list = tr.find_all("td")
19     name = td_list[0].text # 获取文本内容
20     low = td_list[1].text
21     avg = td_list[2].text
22     high = td_list[3].text
23     gui = td_list[4].text
24     dan = td_list[5].text
25     day = td_list[6].text
26     cv_writer.writerow([name, low, avg, high,
27                          gui, dan, day])
28
29 f.close()
30 print("搞定")
```

有人可能要问了. 为什么只有第一页数据. 你观察一下第二页, 第三页的url就明白了了

- 1 <http://www.xinfadi.com.cn/marketanalysis/0/list/1.shtml>
- 2 <http://www.xinfadi.com.cn/marketanalysis/0/list/2.shtml>
- 3 <http://www.xinfadi.com.cn/marketanalysis/0/list/3.shtml>

此处省略一万个字~