










Python爬虫实战-爬取TIOBE TOP20语言排行榜

URL: [index](#) | TIOBE - The Software Quality Company

IDE: PyCharm Professional

要爬取的是排名前20的语言榜单，并将其存成文本文件和生成词云

Mar 2022	Mar 2021	Change	Programming Language		Ratings	Change
1	3	▲		Python	14.26%	+3.95%
2	1	▼		C	13.06%	-2.27%
3	2	▼		Java	11.19%	+0.74%
4	4			C++	8.66%	+2.14%
5	5			C#	5.92%	+0.95%
6	6			Visual Basic	5.77%	+0.91%
7	7			JavaScript	2.09%	-0.03%
8	8			PHP	1.92%	-0.15%
9	9			Assembly language	1.90%	-0.07%

这个榜单包括6列，分别是2022年3月的排名(Mar 2022)、2021年3月的排名(Mar 2021)、增减、程序语言(Programming Language)、占比(Ratings)、变化率(Change)

前置准备

导入要用的库

```
import requests
from bs4 import BeautifulSoup
import re
import wordcloud
```

requests是用来发起https请求，并获取结果的。BeautifulSoup用于解析网页html代码，re用于正则匹配，wordcloud用于生成词云。

检查网页源代码，可以发现整个榜单放在id为top20的table标签下












Got it!

programming skills are still up to date or to make a strategic decision about what programming language software system. The definition of the TIOBE index can be found [here](#).

[illegible]

We use cookies to analyse our traffic and to show ads. By using our website, you agree to our use of cookies.

The index can be used to check whether your programming skills are still up to date or to make a strategic decision about what programming language should be adopted when starting to build a new software system. The definition of the TIOBE index can be found [here](#).

	tbody 928 × 1000.5	Mar 2021	Change	Programming Language	Ratings	Change
1	3	▲	 Python	14.26%	+3.95%	
2	1	▼	 C	13.06%	-2.27%	
3	2	▼	 Java	11.19%	+0.74%	
4	4		 C++	8.66%	+2.14%	
5	5		 C#	5.92%	+0.95%	
6	6		 Visual Basic	5.77%	+0.91%	
7	7		 JavaScript	2.09%	-0.03%	
8	8		 PHP	1.92%	-0.15%	
9	9		 Assembly language	1.90%	-0.07%	
10	10		 SQL	1.85%	-0.02%	
11	13	▲	 R	1.37%	+0.12%	

The screenshot shows the Chrome DevTools interface. At the top, there are tabs for "DevTools is now available in Chinese!", "Always match Chrome's language", "Switch DevTools to Chinese", and "Don't show again". Below these are icons for Elements, Console, Sources, and a search bar.

The "Elements" pane displays the DOM tree. The selected element is a table with the following structure:








```
<h3>March</h3><div><p>STORY</p><p></p><p></p><table id="top20" class="table table-striped table-top20"><thead><tr></tr></thead><tbody></tbody></table><div align="center"><div><div id="container" style="width: 100%; height: 450px; overflow: hidden;"><div id="chart"></div></div></div></div>
```

The "Styles" pane shows the styles for the selected element. The "element.style" section is expanded, showing the following rules:

```
*.before, .after {
  -webkit-box-sizing: inherit;
  -moz-box-sizing: inherit;
  box-sizing: inherit;
}

tbody {
  display: table-row-group;
  vertical-align: middle;
  border-color: inherit;
}
```

The "Inherited from" section shows the inheritance chain: table#top20.table.table... > table.table { font-family: Arial, Verdana, sans-serif; } > tpcl.css:27

tr 928 × 50.5	Mar 2021	Change	Programming Language	Ratings	Change
1	3	▲	 Python	14.26%	+3.95%
2	1	▼	 C	13.06%	-2.27%
3	2	▼	 Java	11.19%	+0.74%
4	4		 C++	8.66%	+2.14%
5	5		 C#	5.92%	+0.95%
6	6		 Visual Basic	5.77%	+0.91%
7	7		 JavaScript	2.09%	-0.03%

[illegible]

获取数据

```
url = "https://www.tiobe.com/tiobe-index/"
res = requests.get(url)
soup = BeautifulSoup(res.text, "html.parser")
table = soup.find("table", id="top20").find("tbody").find_all("tr")
```

上述代码获取了网页html源码并解析，返回一个soup对象，使用find和find_all函数根据标签查找，获取tr标签下数据组成的列表

接下来将其打印出来看看

```
/usr/bin/python3.7 /home/hwx/PycharmProjects/work/main.py
<tr><td>1</td><td>3</td><td></td><td class="td-top20"></td><tr><td>2</td><td>1</td><td></td><td class="td-top20"></td><td>C</td><tr><td>3</td><td>2</td><td></td><td class="td-top20"></td><td>C</td><tr><td>4</td><td>4</td><td></td><td class="td-top20"></td><td>C++</td><td>8.66%</td><td>+2.14%</td></tr><tr><td>5</td><td>5</td><td></td><td class="td-top20"></td><td>C#</td><td>5.92%</td><td>+0.95%</td></tr><tr><td>6</td><td>6</td><td></td><td class="td-top20"></td><td>Visual Basic</td><td>5.7%</td><td></td></tr><tr><td>7</td><td>7</td><td></td><td class="td-top20"></td><td>JavaScript</td><td>2.09%</td><td></td></tr><tr><td>8</td><td>8</td><td></td><td class="td-top20"></td><td>PHP</td><td>1.92%</td><td>-0.15%</td></tr><tr><td>9</td><td>9</td><td></td><td class="td-top20"></td><td>Assembly langu</td><td></td><td></td></tr><tr><td>10</td><td>10</td><td></td><td class="td-top20"></td><td>SQL</td><td>1.85%</td><td>-0.02%</td></tr><tr><td>11</td><td>13</td><td></td><td class="td-top20"></td><td>R</td><td></td><td></td></tr><tr><td>12</td><td>14</td><td></td><td class="td-top20"></td><td>Delphi/Object Pascal</td><td></td><td></td></tr><tr><td>13</td><td>11</td><td></td><td class="td-top20"></td><td>Go</td><td></td><td></td></tr><tr><td>14</td><td>19</td><td></td><td class="td-top20"></td><td>Swift</td><td></td><td></td></tr><tr><td>15</td><td>18</td><td></td><td class="td-top20"></td><td>MATLAB</td><td></td><td></td></tr><tr><td>16</td><td>16</td><td></td><td class="td-top20"></td><td>Ruby</td><td>0.66%</td><td>-0.52%</td></tr><tr><td>17</td><td>12</td><td></td><td class="td-top20"></td><td>Classic Visual Basic</td><td></td><td></td></tr><tr><td>18</td><td>20</td><td></td><td class="td-top20"></td><td>Objective-C</td><td></td><td></td></tr><tr><td>19</td><td>17</td><td></td><td class="td-top20"></td><td>Perl</td><td></td><td></td></tr><tr><td>20</td><td>38</td><td></td><td class="td-top20"></td><td>Lua</td><td></td><td></td></tr></table>
```

打印出了每一个tr标签下的内容，要获取的数据就在其中

```
for item in table:
    print(item.text)
```

那直接取出文本

```
/usr/bin/python3.7 /home/hwx/PycharmProjects/work/main.py
13Python14.26%+3.95%
21C13.06%-2.27%
32Java11.19%+0.74%
44C++8.66%+2.14%
55C#5.92%+0.95%
66Visual Basic5.77%+0.91%
77JavaScript2.09%-0.03%
88PHP1.92%-0.15%
99Assembly language1.90%-0.07%
1010SQL1.85%-0.02%
1113R1.37%+0.12%
1214Delphi/Object Pascal1.12%-0.07%
1311Go0.98%-0.33%
1419Swift0.90%-0.05%
1518MATLAB0.80%-0.23%
1616Ruby0.66%-0.52%
1712Classic Visual Basic0.60%-0.66%
1820Objective-C0.59%-0.31%
1917Perl0.57%-0.58%
2038Lua0.56%+0.23%

Process finished with exit code 0
```

现在拿到的数据就很间接了，程序将单个标签下所有的文本都连接到了一起

现在已经获取到了数据，接下来要做的是数据处理

数据处理

现在获取的仅仅是文本，接下来要对其进行分割

这时派上用场的是正则表达式

这里使用re模块中的search函数

```
re.search(pattern, string, flags=0) # 扫描整个字符串并返回第一个成功的匹配。
```

参数	描述
pattern	匹配的正则表达式
string	要匹配的字符串
flags	标志位，用于控制正则表达式的匹配方式，如：是否区分大小写，多行匹配等等

不难发现，在上面爬到的文本中，每一行的第一个数字就是语言的当前排名(1,2,3...)，因此这是固定的，后面紧跟的数字是去年排名，再跟着的若干单词是语言的名字，随后两个百分数就是占比和变化率。

那么每一次循环中，都是对一行的文本中数据的提取。

首先剔除每一行的第一个数字即当前排名，那么到字母之前剩下的数字就是去年排名，可以轻易的写出正则表达式"[0-9]+", 来匹配第一个出现的纯数字。

随后就是语言名字了，直接匹配第一个非数字字符串即可，即"\D+"。

接下来要匹配占比，显然这个百分数后面都会跟着一个正号或负号，因此表达式可以写成

"(\d+(\.[0-9]?)%+)|(\d+(\.[0-9]?)%-)"，最后剩下的是变化率，这个百分数之前都有正负号，所以可以直接写出"-(.?)%|+(.?)%"。最后得到的结果用.group()函数取出即可

每一行提取出的数据放到一个元组里

```
current_rand = 1
for item in table:
    if current_rand < 10:
        target = item.text[1:]
    else:
        target = item.text[2:]
    previous_rand = re.search(r"[0-9]+", target).group()
    language = re.search(r"(\D)+", target).group()
    ratings = re.search(r"(\d+\.\([0-9]*\)%+)|(\d+\.\([0-9]*\)%-)",
target).group()[:-1]
    change = re.search(r"-(..*?)%|+(..*?)%", target).group()
    current_rand = current_rand + 1
```

再循环中将结果打印

```
/usr/bin/python3.7 /home/hwx/PycharmProjects/work/main.py
('1', '3', 'Python', '14.26%', '+3.95%')
('2', '1', 'C', '13.06%', '-2.27%')
('3', '2', 'Java', '11.19%', '+0.74%')
('4', '4', 'C++', '8.66%', '++8.66%')
('5', '5', 'C#', '5.92%', '+0.95%')
('6', '6', 'Visual Basic', '5.77%', '+0.91%')
('7', '7', 'JavaScript', '2.09%', '-0.03%')
('8', '8', 'PHP', '1.92%', '-0.15%')
('9', '9', 'Assembly language', '1.90%', '-0.07%')
('10', '10', 'SQL', '1.85%', '-0.02%')
('11', '13', 'R', '1.37%', '+0.12%')
('12', '14', 'Delphi/Object Pascal', '1.12%', '-0.07%')
('13', '11', 'Go', '0.98%', '-0.33%')
('14', '19', 'Swift', '0.90%', '-0.05%')
('15', '18', 'MATLAB', '0.80%', '-0.23%')
('16', '16', 'Ruby', '0.66%', '-0.52%')
('17', '12', 'Classic Visual Basic', '0.60%', '-0.66%')
('18', '20', 'Objective-C', '0.59%', '-C0.59%')
('19', '17', 'Perl', '0.57%', '-0.58%')
('20', '38', 'Lua', '0.56%', '+0.23%')
```

因为最后的结果要保存，所以要定义一个列表，将元组放到列表中

考虑到要生成词云，所以同样要定义一个字典，与程序语言名称为key，以占比为值，这里要把百分比的%去掉，然后转化为浮点数

```
words = {}
result = []
current_rand = 1
for item in table:
    if current_rand < 10:
        target = item.text[1:]
    else:
        target = item.text[2:]
    previous_rand = re.search(r"[0-9]+", target).group()
    language = re.search(r"(\D)+", target).group()
    ratings = re.search(r"(\d+\.[0-9]*?)%+)|(\d+\.[0-9]*?)%-",
target).group()[:-1]
    change = re.search(r"-(.*?)%|(.*?)%", target).group()
    result.append((str(current_rand), previous_rand, language, ratings,
change))
    words[language] = float(ratings.strip("%"))
    current_rand = current_rand + 1
```

得到列表后，保存文本文件

```
f = open("result.txt", "w+")
f.write("Mar 2022      Mar 2021      Programming Language      Ratings\nChange\n")
for item in result:
    line = "{:<10}\t{:<10}\t{:<20}\t{:<10}\t{:<10}".format(item[0], item[1],
item[2], item[3], item[4])
    f.write(line + "\n")
f.close()
```

上述代码中使用了格式化字符串，利用了format函数，利用open函数打开一个文件，指定标志为"w+"，使用write函数向其中写入数据，使用close函数最后关闭文件

如下是生成词云的代码，width和height指定长宽，background_color制定背景色_

```
w = wordcloud.WordCloud(width=1000, height=700, background_color="white")
w.generate_from_frequencies(words)
w.to_file("result.png")
```

generate_from_frequencies接收一个字典

to_file生成最后的图片文件

完整代码如下

```
import requests
from bs4 import BeautifulSoup
import re
import wordcloud

url = "https://www.tiobe.com/tiobe-index/"
res = requests.get(url)
soup = BeautifulSoup(res.text, "html.parser")
table = soup.find("table", id="top20").find("tbody").find_all("tr")

words = {}
result = []
current_rand = 1
for item in table:
    if current_rand < 10:
        target = item.text[1:]
    else:
        target = item.text[2:]
    previous_rand = re.search(r"[0-9]+", target).group()
    language = re.search(r"(\D)+", target).group()
    ratings = re.search(r"(\d+\.[0-9]*?)%+)|(\d+\.[0-9]*?)%-",
target).group()[:-1]
    change = re.search(r"-(.*?)%|\+(.*?)%", target).group()
    result.append((str(current_rand), previous_rand, language, ratings,
change))
    words[language] = float(ratings.strip("%"))
    current_rand = current_rand + 1

f = open("result.txt", "w+")
f.write("Mar 2022      Mar 2021      Programming Language      Ratings
Change\n")
for item in result:
    line = "{:<10}\t{:<10}\t{:<20}\t{:<10}\t{:<10}".format(item[0], item[1],
item[2], item[3], item[4])
    f.write(line + "\n")
f.close()

w = wordcloud.WordCloud(width=1000, height=700, background_color="white")
w.generate_from_frequencies(words)
w.to_file("result.png")
```

最后生成了如下的文本文件

	Mar 2022	Mar 2021	Programming Language	Ratings	Change
1	1	3	Python	14.26%	+3.95%
2	2	1	C	13.06%	-2.27%
3	3	2	Java	11.19%	+0.74%
4	4	4	C++	8.66%	++8.66%
5	5	5	C#	5.92%	+0.95%
6	6	6	Visual Basic	5.77%	+0.91%
7	7	7	JavaScript	2.09%	-0.03%
8	8	8	PHP	1.92%	-0.15%
9	9	9	Assembly language	1.90%	-0.07%
10	10	10	SQL	1.85%	-0.02%
11	11	13	R	1.37%	+0.12%
12	12	14	Delphi/Object Pascal	1.12%	-0.07%
13	13	11	Go	0.98%	-0.33%
14	14	19	Swift	0.90%	-0.05%
15	15	18	MATLAB	0.80%	-0.23%
16	16	16	Ruby	0.66%	-0.52%
17	17	12	Classic Visual Basic	0.60%	-0.66%
18	18	20	Objective-C	0.59%	-C0.59%
19	19	17	Perl	0.57%	-0.58%
20	20	38	Lua	0.56%	+0.23%
21					

以及词云图片

