正则表达式

- 在线正则表达式：https://tool.oschina.net/regex#
- python中通过**re**库来实现

模式	描述
^	匹配字符串的开头
\s	匹配空格
\d	匹配数字
{n}	匹配前面表达式n次
\w	匹配字母、下划线、数字
[]	指定匹配范围
()	标记匹配组
.	匹配除换行符以外任意字符
*	匹配前面表达式0次或多次
$	匹配字符串的结尾
+	匹配前面表达式1次或多次
?	匹配前面表达式0次或1次（非贪婪）

match

- 通过**match**方法与字符串起始位置匹配正则表达式。如果匹配返回成功结果；如果不匹配，返回None。

```
import re
```

```
content = 'Hello 123 4567 World_This is a Regex Demo'
print(len(content))
# 匹配一个开头是"Hello"后跟
# 一个空格、三个数字、一个空格、四个数字、一个空格和10个字母、下划线或数字的字符串
result = re.match('^Hello\s\d\d\d\s\d{4}\s\w{10}', content)
print(result)
print(result.group())
print(result.span())
```

```
41
<_sre.SRE_Match object; span=(0, 25), match='Hello 123 4567 World_This'>
Hello 123 4567 World_This
(0, 25)
```

匹配目标

######

- 通过括号**()**标记子表达式的开始和结束位置，再调用**group**方法传入索引获取提取结果。

```
import re
```

```
content = 'Hello 1234567 World_This is a Regex Demo'
# 使用括号来创建一个捕获组，这里匹配数字序列，并将它们作为单独的组返回
result = re.match('^Hello\s(\d+)\sWorld', content)
print(result)
print(result.group())
print(result.group(1))
print(result.span())
```

```
<_sre.SRE_Match object; span=(0, 19), match='Hello 1234567 World'>
Hello 1234567 World
1234567
(0, 19)
```

通用匹配

- 通用匹配 `.*` 表示匹配任意字符。其中， `.` 表示可以匹配除换行符以外的任意字符， `*` 代表匹配前面字符无限次

```
import re
```

```
content = 'Hello 1234567 World_This is a Regex Demo'
# 使用.*进行贪婪匹配，匹配尽可能多的字符直到字符串末尾的"Demo"
result = re.match('^Hello.*Demo$', content)
print(result)
print(result.group())
print(result.span())
```

```
<_sre.SRE_Match object; span=(0, 40), match='Hello 1234567 World_This is a
Regex Demo'>
Hello 1234567 World_This is a Regex Demo
(0, 40)
```

贪婪与非贪婪

- 贪婪匹配： `.*` 会匹配尽可能多的字符。
- 非贪婪匹配： `.*?` 会匹配尽可能少的字符。

```
import re
```

```
"""贪婪匹配"""
content = 'Hello 1234567 World_This is a Regex Demo'
# 贪婪匹配：尽可能多地匹配数字，结果将是最后面的数字序列
result = re.match('^Hello.*(\d+).*Demo$', content)
print(result)
print(result.group(1))
```

```
<_sre.SRE_Match object; span=(0, 40), match='Hello 1234567 World_This is a
Regex Demo'>
7
```

```
"""非贪婪匹配"""
content = 'Hello dfdsfsdfs 1234567 World_This is a Regex Demo'
# 非贪婪匹配：尽可能少地匹配数字，结果将是第一个完整的数字序列
result = re.match('^Hello.*?(\d+).*Demo$', content)
print(result)
print(result.group(1))
```

```
<_sre.SRE_Match object; span=(0, 50), match='Hello dfdsfsdfs 1234567 World_This
is a Regex Dem>
1234567
```

```
"""
如果在字符串中间尽可能用非贪婪匹配。
如果匹配的结果在字符串结尾，非贪婪匹配可能匹配不到任何内容。
"""
content = 'http://weibo.com/comment/kEraCN'
result1 = re.match('^http.*?comment/(.*?)', content)  # 非贪婪
result2 = re.match('^http.*?comment/(.*)', content)  # 贪婪
print('result1', result1.group(1))
print('result2', result2.group(1))
```

```
result1
result2 kEraCN
```

修饰符

- 修饰符 `re.S` 作用是使匹配内容包括换行符在内的所有字符。

```
import re
```

```
content = '''Hello 1234567 World_This
is a Regex Demo'''
# 使用修饰符re.S使`.`匹配包括换行符在内的所有字符
result = re.match('^Hello.*?(\d+)(.*?)Demo$', content, re.S)
print(result)
print(result.group(1))
```

```
<_sre.SRE_Match object; span=(0, 41), match='Hello 1234567 World_This \nis a
Regex Demo'>
1234567
```

转义匹配

- 当在目标字符串中遇到用作正则匹配模式的特殊字符时，在此字符前面加反斜线 \ 转义一下即可。

```
import re
```

```
content = '(百度)www.baidu.com'
# 对包含特殊字符的字符串进行转义匹配
result = re.match('\(百度\)www\.baidu\.com', content)
print(result.group())
```

```
(百度)www.baidu.com
```

search

- **match**方法是从字符串开头开始匹配的，意味着一旦开头不匹配，整个匹配就失败了。

```
import re
```

```
content = 'Extra stings Hello 1234567 World_This is a Regex Demo Extra stings'
result = re.match('Hello.*?(\d+).*?Demo', content)
print(result)
```

```
None
```

- **search**在匹配时会扫描整个字符串，然后返回第一个匹配成功的结果。

```
content = 'Hello Extra stings Hello 1234567 World_This is a Regex Demo Extra
stings'
# 使用search而不是match，因为match只从字符串开始处进行匹配
result = re.search('Hello.*?(\d+).*?Demo', content)
print(result)
print(result.group(1))
```

```
<_sre.SRE_Match object; span=(0, 59), match='Hello Extra stings Hello 1234567
World_This is a >
1234567
```

练习

- 获取：齐秦　往事随风
- 因此，为了匹配方便，尽量使用**search**方法。如下实例：

```
html = '''<div id="songs-list">
<h2 class="title"> 经典老歌 </h2>
<p class="introduction">
经典老歌列表
</p>
<ul id="list" class="list-group">
<li data-view="2"> 一路上有你 </li>
<li data-view="7">
```

```
<a href="/2.mp3" singer="任贤齐"> 沧海一声笑 </a>
</li>
<li data-view="4" class="active">
<a href="/3.mp3" singer="齐秦"> 往事随风 </a>
</li>
<li data-view="6"><a href="/4.mp3" singer="beyond"> 光辉岁月 </a></li>
<li data-view="5"><a href="/5.mp3" singer="陈慧琳"> 记事本 </a></li>
<li data-view="5">
<a href="/6.mp3" singer="邓丽君"> 但愿人长久 </a>
</li>
</ul>
</div>'''
result = re.search('<li.*?active.*?singer="(.*?)">(.*?)</a>', html, re.S)
# result=re.search('<li.*?singer="(.*?)">(.*?)</a>',html,re.S) # 不加active
# result=re.search('<li.*?singer="(.*?)">(.*?)</a>',html) # 不加re.S
print(result.group(1), result.group(2))
```

```
齐秦  往事随风
```

findall

- **findall**获取与正则表达式相匹配的所有字符串

```
import re
```

```
html = '''<div id="songs-list">
<h2 class="title"> 经典老歌 </h2>
<p class="introduction">
经典老歌列表
</p>
<ul id="list" class="list-group">
<li data-view="2"> 一路上有你 </li>
<li data-view="7">
<a href="/2.mp3" singer="任贤齐"> 沧海一声笑 </a>
</li>
<li data-view="4" class="active">
<a href="/3.mp3" singer="齐秦"> 往事随风 </a>
</li>
<li data-view="6"><a href="/4.mp3" singer="beyond"> 光辉岁月 </a></li>
<li data-view="5"><a href="/5.mp3" singer="陈慧琳"> 记事本 </a></li>
<li data-view="5">
<a href="/6.mp3" singer="邓丽君"> 但愿人长久 </a>
</li>
</ul>
</div>'''
# 使用findall来获取所有匹配的歌曲信息，包括链接、歌手名和歌曲名
results = re.findall(
    '<li.*?href="(.*?)".*?singer="(.*?)">(.*?)</a>', html, re.S)
print(results)
print(type(results))
print('-' * 50)
for result in results:
    print(result)
```

```
    print(result[0], result[1], result[2])
```

```
[('/2.mp3', '任贤齐', ' 沧海一声笑 '), ('/3.mp3', '齐秦', ' 往事随风 '), ('/4.mp3',
'beyond', ' 光辉岁月 '), ('/5.mp3', '陈慧琳', ' 记事本 '), ('/6.mp3', '邓丽君', ' 但
愿人长久 ')]
<class 'list'>
----------------------------------------------------
('/2.mp3', '任贤齐', ' 沧海一声笑 ')
/2.mp3 任贤齐  沧海一声笑
('/3.mp3', '齐秦', ' 往事随风 ')
/3.mp3 齐秦  往事随风
('/4.mp3', 'beyond', ' 光辉岁月 ')
/4.mp3 beyond  光辉岁月
('/5.mp3', '陈慧琳', ' 记事本 ')
/5.mp3 陈慧琳  记事本
('/6.mp3', '邓丽君', ' 但愿人长久 ')
/6.mp3 邓丽君  但愿人长久
```

练习

- 通过findall方法获取所有li中的歌名

```
"""通过findall方法获取所有li中的歌名"""
import re
```

```
html = '''<div id="songs-list">
<h2 class="title"> 经典老歌 </h2>
<p class="introduction">
经典老歌列表
</p>
<ul id="list" class="list-group">
<li data-view="2"> 一路上有你 </li>
<li data-view="7">
<a href="/2.mp3" singer="任贤齐"> 沧海一声笑 </a>
</li>
<li data-view="4" class="active">
<a href="/3.mp3" singer="齐秦"> 往事随风 </a>
</li>
<li data-view="6"><a href="/4.mp3" singer="beyond"> 光辉岁月 </a></li>
<li data-view="5"><a href="/5.mp3" singer="陈慧琳"> 记事本 </a></li>
<li data-view="5">
<a href="/6.mp3" singer="邓丽君"> 但愿人长久 </a>
</li>
</ul>
</div>'''
# results = re.findall('<li.*?>\s*?(<a.*?>)?(\w+)(</a>)?\s*?</li>', html, re.S)
# 使用findall方法获取所有li中的歌名
results = re.findall(
    '<li.*?>(\n?<a.*?>)?\s(.*?)\s(</a>\n?)?</li>', html, re.S)
print(results,len(results))
# 将获取的结果写入文件
with open('./re.txt','w',encoding='utf-8') as f:
    for result in results:
        print(result[1])
```

```
        f.write(result[1]+'\n')
```

```
[('', '一路上有你', ''), ('\n<a href="/2.mp3" singer="任贤齐">', '沧海一声笑',
'</a>\n'), ('\n<a href="/3.mp3" singer="齐秦">', '往事随风', '</a>\n'), ('<a
href="/4.mp3" singer="beyond">', '光辉岁月', '</a>'), ('<a href="/5.mp3"
singer="陈慧琳">', '记事本', '</a>'), ('\n<a href="/6.mp3" singer="邓丽君">', '但愿
人长久', '</a>\n')] 6
一路上有你
沧海一声笑
往事随风
光辉岁月
记事本
但愿人长久
```

sub

- **sub**用来修改相匹配的文本

```
"""通过sub方法替换所有数字"""
import re
```

```
content = '54aK54yr5oiR54ix5L2g'
# 使用sub方法替换字符串中的数字
content = re.sub('\d+', '', content)
print(content)
```

```
aKyroiRixLg
```

练习

- 通过sub方法将a节点去掉，再使用findall方法获取所有li中的歌名

```
"""通过sub方法将a节点去掉"""
import re
```

```
html = '''<div id="songs-list">
<h2 class="title"> 经典老歌 </h2>
<p class="introduction">
经典老歌列表
</p>
<ul id="list" class="list-group">
<li data-view="2"> 一路上有你 </li>
<li data-view="7">
<a href="/2.mp3" singer="任贤齐"> 沧海一声笑 </a>
</li>
<li data-view="4" class="active">
<a href="/3.mp3" singer="齐秦"> 往事随风 </a>
</li>
<li data-view="6"><a href="/4.mp3" singer="beyond"> 光辉岁月 </a></li>
<li data-view="5"><a href="/5.mp3" singer="陈慧琳"> 记事本 </a></li>
<li data-view="5">
```

```
<a href="/6.mp3" singer="邓丽君"> 但愿人长久 </a>
</li>
</ul>
</div>'''
# 使用sub方法去除所有a标签
html = re.sub('<a.*?>|</a>', '', html)
print(html)
# 使用findall方法获取所有li中的歌名
results = re.findall('<li.*?>(.*?)</li>', html, re.S)
for result in results:
    print(result.strip())
```

```
<div id="songs-list">
<h2 class="title"> 经典老歌 </h2>
<p class="introduction">
经典老歌列表
</p>
<ul id="list" class="list-group">
<li data-view="2"> 一路上有你 </li>
<li data-view="7">
 沧海一声笑
</li>
<li data-view="4" class="active">
 往事随风
</li>
<li data-view="6"> 光辉岁月 </li>
<li data-view="5"> 记事本 </li>
<li data-view="5">
 但愿人长久
</li>
</ul>
</div>
一路上有你
沧海一声笑
往事随风
光辉岁月
记事本
但愿人长久
```

compile

- **compile**可以将正在字符串编译成正则表达式对象，便于后面匹配中复用

```
import re
```

```
content1 = '2022-7-15 12:00'
content2 = '2022-7-17 12:55'
content3 = '2022-7-22 13:21'
# 使用compile创建正则表达式对象，用于多次匹配，获取年月日
pattern = re.compile('\d{2}:\d{2}')
result1 = re.sub(pattern, '', content1)
result2 = re.sub(pattern, '', content2)
result3 = re.sub(pattern, '', content3)
print(result1, result2, result3)
```

```
2022-7-15  2022-7-17  2022-7-22
```

作业

oj平台：www.cqvist.club

- 提取日期信息
- 验证电话号码格式
- 提取首个标签内的文本