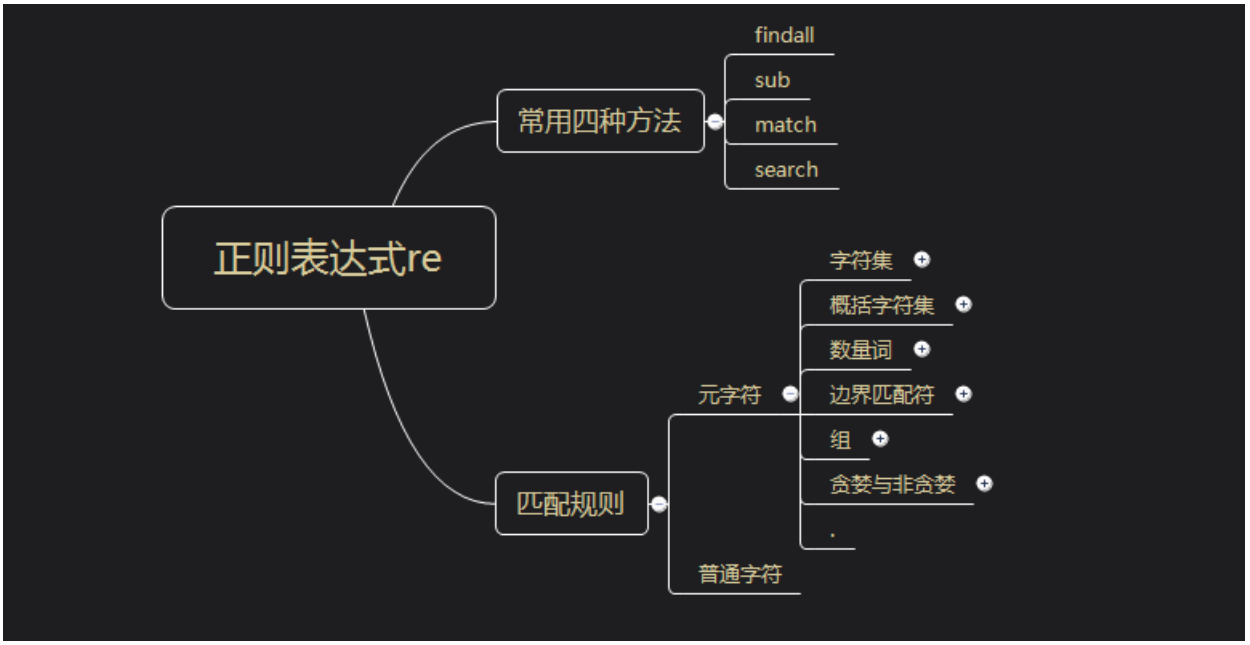
爬虫06-闲谈正则表达式（一）

官方解释：正则表达式的概念是使用单个字符串来描述、匹配一系列匹配某个句法规则的字符串。

简单来说，正则表达式就是通过一定的匹配规则，从一个字符串中提取出我们想要的数据，虽然有时候会比较复杂，但无疑它是非常强大的。

在python中要使用正则表达式，首先需要先导入python内置的re模块。

接下来的五天我们要分享的主要内容如下图：



正则表达式的灵魂在于匹配规则的灵活使用，而匹配规则，我简单将之分为两大类，一类是普通字符，这类匹配意义往往不大，另一类是元字符，是我们需要重点掌握的内容。

```
我们先来用代码示例来看看普通字符的匹配：

import re
target = 'life is short, i learn python.'
result = re.findall('python', target)
```

```
# findall是re库的一个重要方法，第一个参数是匹配规则，第二个参数是要匹配的目标字符串，还有第三个参数，我们之后讲，findall返回的结果是一个列表。
# 这行代码的意思是从target中匹配'python',如果匹配到就返回，没有匹配到就返回空列表。
result1 = re.findall('java', target)
print(result)
 # 得到的结果是['python']
 print(result1)
 # 得到的结果是[]

```

如果匹配规则是一个普通字符串的话，意义并不大，试想一下，一个网页上的内容都是变化的，处处可能都不一样，我们想用一个固定的普通字符串去匹配到内容，显然是不太合适的。

就比如简书网站，我们想要获取每篇文章的发布时间，每篇的文章的发布时间都是不一样的，用一个固定的字符串显然匹配不出来我们想要的内容。

但是，通过上述的匹配，我们能够联想到我们之前学过的什么内容吗？有关于字符串的。

if in，是不是？判断一个字符串是否存在于另一个字符串中。我们就可以用上述代码来自己实现这个功能，有兴趣的童鞋可以自己尝试封装成一个函数。

接下来，就是我们的重头戏了，**元字符**，我粗略地将他们分为了7类，我们先来看第一类，字符集，用[]表示，中括号内可以写任意字符，各字符间是或的关系（不理解没关系，后面会有代码解释）：

现在我们得到了一个这样的字符串 target = ‘abc acc aec agc adc aic’ ，我们有这样一个需求，需要找出这个字符串中中间是d或者e的单词，我们该怎么做呢？

很多童鞋第一反应就是for循环遍历，for循环当然可以写出来，for循环是非常厉害的，有兴趣的童鞋可以去尝试一下，但是用正则会简单很多，我们来看看怎么做：

```
import re
target = 'abc acc aec agc adc aic'
result = re.findall('a[de]c', target)
# 这一行中的[de]表示这个位置上的字符是d或者是e都可以匹配出来
print(result)
```

```
# 得到的结果是['aec', 'adc']
```

这只是字符集[]的一个最简单的应用，现在我们又有一个需求，需要找出这个字符串中中间是b-z之间的任意一个字符的单词，就可以这样写了，而不需要把b-z之间的字符都写出来。

```

import re
target = 'abc acc aec agc adc aic'
result = re.findall('a[b-z]c', target)
# 这一行中的[b-z]表示这个位置上的字符在b-z范围内都可以匹配出来
print(result)
```

得到的结果是['abc', 'acc', 'aec', 'agc', 'adc', 'aic']

还有另一种用法需要了解，我们同样还是通过代码来理解比较好。

```
import re
target = 'abc acc aec agc adc aic'
result = re.findall('a[^c-z]c', target)
# 这一行中的[^c-z]表示这个位置上的字符不在c-z范围内都可以匹配出来，注意是不在
print(result)
# 得到的结果是['abc']

```

好了，字符集[]的匹配规则就讨论完了，我们来总结下：

匹配规则（举例说明）	释义
[abf]	表示该位置上的字符为a或者b或者f，即匹配成功
[a-z]	表示该位置上的字符在a-z之间，即匹配成功
[^a-z]	表示该位置上的字符不在a-z之间，即匹配成功