

# 正则表达式

xbZhong

2024-10-29

[本页PDF](#)

## 正则表达式

限定符	作用
?	表明前面的字符可以出现0或1次
*	可以匹配0或多个字符
+	会匹配出现一次以上的字符
{ }	在花括号里输入一个范围，会匹配字符出现的次数
( )	括号里可以填想匹配的字符
	左右可以填字符，要么匹配左边，要么匹配右边
^	匹配除了^后面列出的字符（后面跟的是[]）
[ ]	方括号里面的内容代表要求匹配的字符只能取自它们
a-z	代表所有小写英文字符
A-Z	代表所有大写英文字符
0-9	代表所有数字字符

元字符	作用
\d	代表数字字符
\w	代表单词字符（英文、数字及下划线）
\s	代表空白符，包含tab和换行符
\D	代表非数字字符
\W	代表非单词字符
\S	代表非空白字符
.	代表任意字符，不包含换行符

元字符	作用
<code>^</code>	匹配行首字符
<code>\$</code>	匹配行末字符

## match

从字符串开头进行匹配，匹配失败返回None

```
import re

re.match(pattern,string)
# pattern 要匹配的正则表达式
# string 要匹配的字符串

a=re.match(r'test','testasdtest')
(a) # 返回一个匹配对象
(a.group()) # 获取匹配结果
(a.span()) # 返回匹配结果的位置，左闭右开区间
(re.match(r'test','atestasdtest')) # 返回None
```

## search

匹配字符串中的任意位置

```
import re

# search和match
a=re.match(r'test','atestasdtest')
a=re.search(r'test','atestasdtest')
(a) # 返回None
(b) # 返回匹配对象
```

## findall

寻找所有能匹配的字符，并以列表的形式返回

- 使用`re.s`属性可以跨行匹配

t re

```
t = re.findall(r'test', '123test123test')
(result)          # 以列表形式返回匹配结果:['test', 'test']
```

行匹配

```
"aaatestaa
23"
(re.findall(r'test', a))          # 返回None
(re.findall(r'test', a, re.s))    # 返回匹配结果
```

## sub

查找字符串中所有相匹配的数据进行替换

t re

*b(要替换的数据, 替换成什么, 字符串)*

```
t = re.sub('php', 'python', 'php是世界上最好的语言--php')
(result)          # 输出"python是世界上最好的语言--python"
```

## spilt

对字符串进行分割, 返回一个列表

t re

```
itcast,java:php-php3;html"
(re.spilt(r',', s))          # 以,进行分割          返回['itcast', 'java:php-php3;html']
(re.spilt(r',|:|-|;', s))    # 以,或:或-或;进行分割      返回['itcast', 'java', 'php', 'php3', 'html']
(re.spilt(r',|:|-|%', s))    # 找不到的分隔符就忽略      返回['itcast', 'java', 'php', 'php3;html']
```

## 贪婪与非贪婪

- 贪婪: 尽可能匹配更多的字符
- 非贪婪: 尽可能匹配尽量少的字符, 在量词后面加上 `?` 实现非贪婪匹配

text

```
= 'abc123def456'
```

贪婪

```
= re.search(r'\d+.*',text)
```

```
(match.group())    # 输出: 123def456
```

贪婪

```
= re.search(r'\d+.*?',text)
```

```
(match.group())    # 输出: 123
```

贪婪

```
= "<div>content</div><p>paragraph</p>"
```

```
= re.search(r'<.*>', text)
```

```
(match.group())    # 输出: <div>content</div><p>paragraph</p>
```

贪婪

```
= re.search(r'<.*?>', text)
```

```
(match.group())    # 输出: <div>
```