链滴

# 浅议正则表达式

作者：vcjmhg
原文链接：https://ld246.com/article/1569728975838
来源网站：链滴

# 目录

# 基本匹配

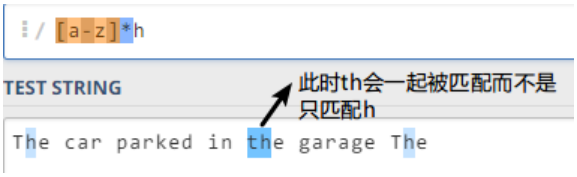
简单来说正则表达式就是对字符串匹配规则的描述，通过该规则我们可以匹配出自己的目标字符串。

Notice:

1. 正则表达式在匹配的时候，默认是 贪懒匹配属于尽力匹配即如果可以匹配满足条件的连续的多个符，就不会匹配满足条件的单个字符

2. 其匹配的过程中大小写敏感

如




# 2 元字符

在正则表达式匹配的过程中充当一些特殊功能的一些字符

| | **正则表达式元字符** |
| --- | --- |
| . | 句号匹配除了换行符之外的其他字符 |
| [ ] | 字符种类，匹配方括号内的任意字符 |
| [^] | 否定的字符种类，匹配除了方括号内的任意字符 |

原文链接：浅议正则表达式

| | |
|---|---|
| * | 匹配>=0个重复在*之前的字符 |
| + | 匹配>=1个重复的+号之前的字符 |
| ? | 标记?之前的字符为可选 |
| (n,m) | 匹配num个大括号之前的字符(n<=num<m) |
| (xyz) | 字符集，匹配与xyz完全相等的字符 |
| \| | 或运算，匹配符号前或后的字符 |
| \ | 转义字符，用于匹配一些保留字符，比如前边一些元字符 |
| ^ | 从开始行进行匹配 |
| $ | 从末端开始进行匹配 |

## 2.1 .运算

.字符用来匹配任意一个字符，比如.ar用来匹配除换行符外的任意字符

## 2.2 字符集

字符集又叫做字符类。通常用方括号来指定一个字符集。

**注意**

1. 方括号内通过字符来指定字符集的位置
2. 方括号内的字符不分前后顺序

eg:

[Tt]he·指定匹配范围为**T或者t+he**，同时它既会匹配The也会匹配the

### 2.2.1 否定字符集

一般来说^表示匹配一个字符串的开头，但是当其在中括号中的时候表示的是对当前匹配字符集的否。例如[^c]ar匹配的就是**除c之外的任意字符**+ar

## 2.3 重复次数

* 、+和?用来指定匹配子模式的个数,这些元字符在不同的情况下有不同的作用

### 2.3.1 *字符

*字符匹配*前边的零个或者一个字符，如a*,匹配以a开头的零个或者多个字符，其实也就是**全体字符**.如

FM:

整个匹配过程如下图所示：

### 2.3.2 +字符

+字符匹配+之前的字符出现>=1次。

eg:

### 2.3.3 ?字符

?前边所匹配的字符为可选，即出现**0次或者1次。**

eg:

Note:

此处的[]加上或者去掉都并不影响匹配结果

## 2.4 {}字符

在正则表达式中{}中相当于量词用来匹配一个或者一组重复的字符可以重复出现的次数，如表达式 [0-9{2,3} 匹配最少 2 位最多 3 位 09 的数字。

Note:

1.我们可以省略第二个参数. 例如~~, [0-9]{2,} 匹配至少两位~~09 的数字.

2.如果逗号也省略掉则表示重复固定的次数. 例如, [0-9]{3} 匹配3位数字

## 2.5 (...) 特征标群

特征目标群指一组写在(...)中的子模式

eg:

例如, 表达式 (ab)* 匹配连续出现 0 或更多个 ab

## 2.6 | 或运算符

或运算符用来判断条件，例如 (T|t)he|car 匹配 (T|t)he 或 car

## 2.7转码特殊字符

反斜线 \ 在表达式中用于转码紧跟其后的字符. 用于指定 { } [ ] / \ + * . $ ^ | ? 这些特殊字符. 如果想匹配这些特殊字符则要在其前面加上反斜线 .

eg:

## 2.8 锚点

在正则表达式中如果需要匹配开头或者末尾的字符需要使用到锚点。 ^ 指定开头, $ 指定结尾.

### 2.8.1 ^号

用来检查匹配的字符串是否在所匹配字符串的开头，例如, 在 abc 中使用表达式 ^a 会得到结果 a. 但果使用 ^b 将匹配不到任何结果. 因为在字符串 abc 中并不是以 b开头.

### 2.8.2 $号

同理于 ^ 号, $ 号用来匹配字符是否是最后一个.

# 3 正则表达式常用的字符集

| 简写 | 描述 |
| --- | --- |
| . | 除换行符外的所有字符 |
| \w | 匹配所有字母数字, 等同于 [a-zA-Z0-9_] |
| \W | 匹配所有非字母数字, 即符号, 等同于: [^\w] |
| \d | 匹配数字: [0-9] |
| \D | 匹配非数字: [^\d] |
| \s | 匹配所有空格字符, 等同于: [\t\n\f\r\p{Z}] |
| \S | 匹配所有非空格字符: [^\s] |
| \f | 匹配一个换页符 |
| \n | 匹配一个换行符 |
| \r | 匹配一个回车符 |
| \t | 匹配一个制表符 |
| \v | 匹配一个垂直制表符 |
| \p | 匹配 CR/LF (等同于 \r\n)，用来匹配 DOS 行止符 |

# 4. 零宽度断言(前后预查)

先行断言和后发断言都属于非捕获簇(不捕获文本 ， 也不针对组合计进行计数). 先行断言用于判断所配的格式是否在另一个确定的格式之前, 匹配结果不包含该确定格式(仅作为约束).

例如, 我们想要获得所有跟在 符号后的数字, 我们可以使用正后发断言 (?<=\$)[0-9\.]*. 这个表达式配 `` 开头, 之后跟着 0,1,2,3,4,5,6,7,8,9,. 这些字符可以出现大于等于 0 次.

| 符号 | 描述 |
| --- | --- |
| ?= | 正线性断言 -存在 |
| ?= | 负线性断言 -排除 |
| ?<= | 正后发断言 -存在 |
| ?<! | 负后发断言 -排除 |

## 4.1 ?=... 正线性断言

?=...正线性断言，表示一部分表达式之后必须跟着?=...定义的表达式

返回结果只包括匹配条件的第一部分表达式，定义一个正线性断言要使用()。在括号内使用一个问号等号:(?=...)

正先行断言的内容写在括号中的等号后面. 例如, 表达式 (T|t)he(?=\sfat) 匹配 The 和 the, 在括号中

们又定义了正先行断言 (?=\sfat) ,即 The 和 the 后面紧跟着 (空格)fat

## 4.2 ?!... 负先行断言

负先行断言 ?! 用于筛选所有匹配结果, 筛选条件为 其后不跟随着断言中定义的格式. 正先行断言 定义和 负先行断言 一样, 区别就是 = 替换成 ! 也就是 (?!...)。

例如，表达式(T|t)he(?!\sfat)匹配The和the，且后边不跟(空格)fat。

## 4.3 ?<= ... 正后发断言

正后发断言 记作(?<=...) 用于筛选所有匹配结果, 筛选条件为**其前跟随着断言中定义的格式**. 例如, 表式 (?<=(T|t)he\s)(fat|mat) 匹配 fat 和 mat, 且其前跟着 The 或 the.

## 4.4 ?<!... 负后发断言

负后发断言 记作 (?<!...) 用于筛选所有匹配结果, 筛选条件为 其前不跟随着断言中定义的格式. 例如, 达式 (?<!(T|t)he\s)(cat) 匹配 cat, 且其前不跟着 The 或 the.

# 5 标志

标志又被称为模式修正符号（可以理解为在不同的匹配模式）

| 标志 | 描述 |
| --- | --- |
| i | 忽略大小写 |
| g | 全局搜索 |
| m | 多行的：锚点元字符^``$工作范围在每行的起始 |

## 5.1 忽略大小写

修饰语 i 用于忽略大小写. 例如, 表达式 /The/gi 表示在全局搜索 The, 在后面的 i 将其条件修改为忽大小写, 则变成搜索 the 和 The, g 表示全局搜索.

## 5.2 全局搜索

修饰符g常用于执行一个全局搜索匹配, 即(不仅仅返回第一个匹配的, 而是返回全部). 例如, 表达式 /.(at /g 表示搜索 任意字符(除了换行) + at, 并返回全部结果.

## 5.3多行搜索(Multiline)

多行修饰符m常用来执行一个多行匹配。

像之前介绍的 (^,$) 用于检查格式是否是在待检测字符串的开头或结尾. 但我们如果想要它在每行的开和结尾生效, 我们需要用到多行修饰符 m.

# 6. 贪婪匹配与惰性匹配 (Greedy vs lazy matchin)

正则表达式默认采用贪婪匹配模式，在该模式下意味着会匹配尽可能长的子串。我们可以使用？将贪匹配模式转化为惰性匹配模式。

# 7.参考材料】

1. github开源项目
2. 正则表达式测试工具