正则表达式

元字符

- .
  - 英文句号，表示除了换行符以外的1个字符
  - \. [.]都可以表示一个英文句号
- []
  - 方括号，表示一个种类或一堆列举的字符中的一个，顺序无关
  - 举例　[a-z]*表示连续小写字母组成的单词
- [^]　方括号加上扬抑符，表示一个种类或一堆列举的字符以外的一个

表示单个字符

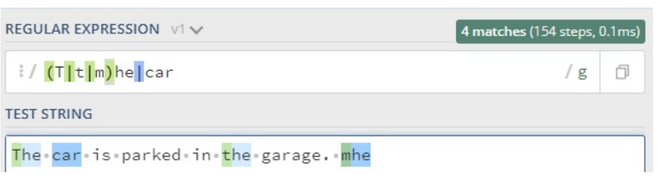
- ()
  - 圆括号
  - (xyz)　表示字符集，匹配和括号内字符集完全相等的文本
  - (表达式)　用来将圆括号内的表达式看作一个整体优先运算，就和数学公式中的圆括号一样

() 内的内容表示的是一个子表达式，() 本身不匹配任何东西，也不限制匹配任何东西，只是把括号内的内容作为同一个表达式来处理，例如 (ab){1,3}，就表示 ab 一起连续出现最少 1 次，最多 3 次。如果没有括号的话，ab{1,3} 就表示 a，后面紧跟的 b 出现最少 1 次，最多 3 次。

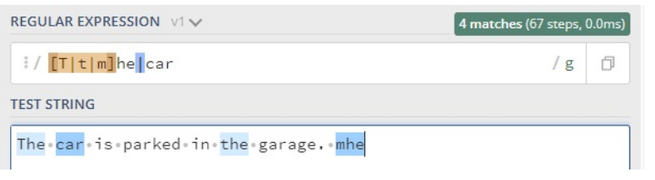
- []和()区别　[] 表示匹配的字符在 [] 中，并且只能出现一次，并且特殊字符写在 [] 会被当成普通字符来匹配。例如 [(a)]，会匹配 (、a、)、这三个字符。

- |　竖杠，表示或，匹配竖杠前的字符（集）或竖杠后的字符（集）

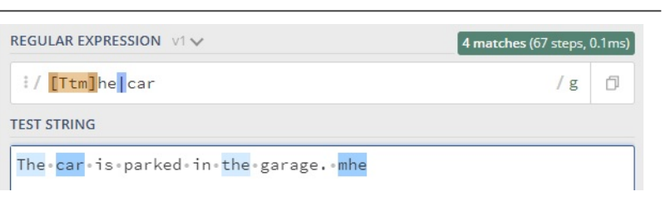
直接用表示什么？



搭配()使用



搭配[]没有意义，因为[]本来就有或的作用了



可以连续使用

- *　星号，匹配星号之前的字符（集）出现任意次（包括0）的情况。等于{0,}
  - 举例　.*表示一行中所有文本
- +　加号，和星号相比，要求字符（集）至少出现1次。
- ?　问号，问号之前的字符（集）是可有可无的（出现0次或1次）。
- {n,m}　花括号，n、m是两个整数，表示之前的字符（集）出现次数在[n,m]之间
  - 第二个参数可以省略，表示至少出现的次数：{n,}
  - 花括号内只写一个数字，表示一个具体的出现次数：{n}

表示重复次数

- \　反斜杠，用作转义符
- ^　扬抑符，表示一行的开头
  - [^]表示某些字符以外的
- $　美刀符，表示一行的末尾

锚点

简写字符集

- \w　所有字母+数字+下划线，等于[a-zA-z0-9_]
- \W　所有字母+数字+下划线以外的字符，等于[^\w]
- \d　所有数字，等于[0-9]
- \D　所有非数字，等于[^\d]
- \s　所有空格字符，等于[\t\n\f\r\v\p{Z}]

分隔符\p{Z}
\p{Zs}或\p{Space_Separator}：不可见的空格，但占据空间。
\p{Zl}或\p{Line_Separator}：分隔线字符U+2028。
\p{Zp}或\p{Paragraph_Separator}：分段字符U+2029。

- \S　所有非空格字符，等于[^\s]
- \f　换页符
- \n　换行符
- \r　回车符
- \p　匹配 CR/LF（等同于 \r\n），用来匹配 DOS 行终止符

在文本处理中, CR, LF, CR/LF 是不同操作系统上使用的换行符：
DOS 和 Windows 采用“回车+换行，CR/LF”表示下一行；
UNIX/Linux 采用“换行符，LF”表示下一行；
苹果机(MAC OS 系统)则采用“回车符，CR”表示下一行。

- \t　制表符
- \v　垂直制表符

零宽度断言（前后预查）

就是断言用于匹配，但是查询到的结果中，不会包含断言。

先行断言和后发断言（合称 lookaround）都属于非捕获组（用于匹配模式，但不包括在匹配列表中）。当我们需要一个模式的前面或后面有另一个特定的模式时，就可以使用它们。

- (?=...)
  - 正先行断言-存在
  - 放在待查找表达式后
  - 正先行断言，表示想要查找的表达式之后【必须跟着】什么表达式
- (?!...)
  - 负先行断言-排除
  - 放在待查找表达式后
  - 和正先行断言相反，表示想要查找的表达式后【没有跟着】什么表达式
- (?<=...)
  - 正后发断言-存在
  - 放在待查找表达式前
  - 表示想要查找的表达式【前面包含】什么表达式
- (?<!...)
  - 负后发断言-排除
  - 放在待查找表达式前
  - 表示想要查找的表达式【前面不包含】什么表达式

可能是出于性能考虑，大多数正则表达式引擎不支持后发断言的可变长度表达式。

- 把$放在断言中好像没用，比如(?=[-\+\s$])，应该用($|(?=[-\+\s]))

标识（模式修正符）

- i　忽略大小写
- g　全局搜索
- m　多行修饰符：锚点元字符 ^ $ 工作范围在每行的起始。

贪婪 / 懒惰匹配

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。
使用 ? 将贪婪匹配模式转化为惰性匹配模式。

```
"/(.*at)/" => The fat cat sat on the mat.
```

在线练习

```
"/(.*?at)/" => The fat cat sat on the mat.
```

在线练习

经过测试，在*后使用，其他情况待测试。

懒惰：从前往后进行处理，遇到了满足要求的情况，就马上停止
贪婪：从前往后进行处理，遇到了满足要求的情况，继续往下处理，说不定还能满足条件，直到没法满足条件位置。

运算符优先级

1. \　转义符
2. (), (?:), (?=), []　圆括号和方括号
3. *, +, ?, {n}, {n,}, {n,m}　表示次数的字符
4. ^, $, \任何元字符、任何字符　定位点和序列（即：位置和顺序）
5. |　替换，"或"操作
   字符具有高于替换运算符的优先级，使得"m|food"匹配"m"或"food"。若要匹配"mood"或"food"，请使用括号创建子表达式，从而产生"(m|f)ood"。