JavaScript 正则表达式

JavaScript 正则表达式

发表于：方凳会

获取一个实例

```
var regexp = /\w+/igm
var regexp = RegExp('\\w+','igm')
```

两种方式基本没有区别。

\w、\d 这样的特殊字符有很多，详见 [MDN参考]。

实例属性和方法

- global，对应g，是否全局匹配
- ignoreCase，对应i，是否忽略大小写
- multiline，对应m，是否跨行匹配
- lastIndex，下次匹配时的起始位置
- source，被搜索的字符串
- compile()，弃用
- test()
- exec()

regexp.test(string) : bool

如果 regex 和 string 匹配，则返回 true。

它和 string.search(regexp) 类似。不同点是返回值，如果 string.search 匹配成功，则返回第一个匹配的 index；匹配失败则返回 -1。

regexp.exec(string) : array?

如果匹配成功，则返回的数组为

1. 匹配到的字符串
2. 第 1 个捕获（如果有）

3. 第 2 个捕获（如果有）

同时，该数组还有两个属性

- index，匹配到的字符串在原字符串中的 index
- input，原字符串

同时 regexp 的 lastIndex 会被更新。

如果匹配失败，则返回 null。

如果 regexp 有 g 标志，那么下次执行 regexp.exec(string) 时，搜索会从 regexp.lastIndex 开始。

例子如下：（注意，如果没有 g 标志，下面的程序会陷入死循环）

```
 myRe = /ab*/g;
 str = 'ababbabbbabbbb';
 myArray;
while ((myArray = myRe.exec(str)) !== ) {
   msg = 'Found ' + myArray[] + ;
  msg += 'Next match starts at ' + myRe.lastIndex;
  console.log(msg);
}
```

Found . match starts at
Found abb. match starts at
Found abbb. match starts at
Found abbbb. match starts at

该方法与 string.match(regexp):array? 类似。

- 在没有 g 标志的情况下，两者返回值一样，即包含匹配和捕获。
- 如果有 g 标志，string.match 会返回所有的匹配，但不返回捕获。

string.replace(regexp, newString | function)

newString 可以包含特殊字符：

- $n – 第n个捕获，从1开始
- $` – 匹配到的字符的前面的所有字符
- $' – 匹配到的字符的后面的所有字符
- $& – 匹配到的字符
- $$ – 表示一个$

function 的参数为：

1. 匹配到的字符串
2. 第 1 个捕获（如果有）
3. 第 2 个捕获（如果有）
4. 最后一个捕获（如果有）
5. 匹配到的字符串的 index

function 的返回值将插入到原字符串中对应的位置。

如果有 g 标志，那么每个匹配都会执行一次 function。
如果一次匹配都没有，那么 function 则不执行。

RegExp 的属性

lastIndex

下次搜索开始的地方。

弃用的属性

- input
- lastMatch
- lastParen
- leftContext
- rightContext