

正则表达式 Cheatsheet

Zhengyaing制作
教学视频请BiliBili搜索Zhengyaing

JavaScript方法

`String.match()`
`String.replace()`
`String.search()`
`String.split()`
`RegExp.exec()`
`RegExp.test()`



```
'abc&mno&xyz'.match(/[a-z]+/g);  
// ["abc", "mno", "xyz"]  
'abc-xyz-abc'.replace(/abc/g, 'biu');  
// "biu-xyz-biu"  
'abc-xyz-abc'.search(/xyz/g);  
// 4  
'abc-def_mno+xyz'.split(/[_+]/g);  
// ["abc", "def", "mno", "xyz"]  
/abc/.exec('abc-xyz-abc');  
// ["abc", index: 0, input: "abc-xyz-  
//abc", groups: undefined]  
/abc/.test('abc-xyz-abc');  
// true
```

优先级

转义符 \
括号和方括号 ()[]
量词限定符 {m,n}
位置和序列 ^、\$、字符
管道符 |

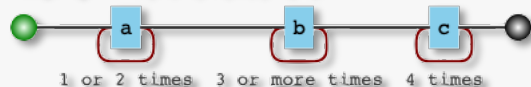
字符匹配-纵向-字符组

a		普通字符
[abc]		字符集合
[^abc]		字符集合取反
[a-zA-Z0-9]		范围表示法
.....		
\d	[0-9]	digit一位数字
\D	[^0-9]	非数字
\w	[0-9a-zA-Z]	数字、字母、下划线
\W	[^0-9a-zA-Z]	非数字、字母、下划线
\s	[\t\v\n\r\f]	Space 空白符号
\S	[^\t\v\n\r\f]	非空白符号
.		通配符

字符匹配-横向-量词

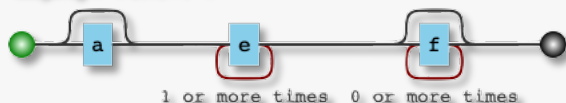
$x\{m, n\}$ x匹配m到n次
 $x\{m, \}$ x匹配至少m次
 $x\{m \}$ x匹配m次

RegExp: /a{1,2}b{3,}c{4}/



x^* x匹配任意次
 x^+ x匹配至少一次
 $x^?$ x匹配零次或一次

RegExp: /a?e+f*/



位置匹配



^ 匹配开头位置
\$ 匹配结束位置

RegExp: /\^b\$/



\b 匹配单词边界

\B 匹配非单词边界

(?=x) 该位置之后的字符要匹配x

(?!x) 该位置之后的字符不匹配x

RegExp: /(?!a)/

Followed by:



其他 | 修饰符

$x | y$ 多选分支匹配x或者y

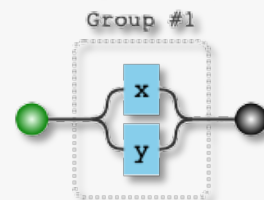
g 全局匹配

() 分组

i 忽略大小写

RegExp: /(x|y)/

m 多行匹配



y 粘性匹配

s 让.匹配任意单个字符