**正则表达式**

感觉：出BUG了，好乱，火星文，看不懂，什么鬼

是检索(模糊范围)字符串的一种规则

简写:

//

全写:

new RegExp(参数1：可以为字符串||正则,参数2：字符串)

**斜杠转义符 \**

a->/a/ ->/\/a\//

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***正则的常用方法-test以及 \d 一个数字转义符**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//console.dir(new RegExp)

/\*

//.test(string);

查看正则是否匹配字符串，如果是返回true否则false

**转义符:**

**\d 一个数字**

\*/

**//test小栗子**

//检测字符串中是否夹着"数字"

var s = 'sajdjahjahdjasjahsduad';

console.log(/\d/.test(s));UUUUUUUUUUUUUUU

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***正则的常用方法-search以及 i 修饰符**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

string.search()

查找字符串指定字符的位置。

找到范围位置，没找到返回-1;

**修饰符:**

**i - 忽略大小写**。

\*/

**//search小栗子**

//console.dir(new String());

//var s = 'dshjdsdXak0jkadak';

//console.log(s.indexOf('x')); //找不到返-1

//console.log(s.search('x')); //找不到返-1

//console.log(s.search(/\d/)); //10

//console.log(s.search(/x/i)); //忽略了大小写找到位置7

//console.log(s.search(new RegExp('x','i'))); //正则全写略了大小写找到位置7

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***正则的常用方法-match以及 g 修饰符 量词 +**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

**string.match(//);**

把正则匹配到的字符提取出来，放到数组中(返回值为数组)

能匹配到就放在数组中，匹配不到返回null

小细节:

如果只匹配到了一个，那么数组中不只一个东西，

但是\*\*length为1

**参数：**

第一个为匹配到的字符

第二个为匹配字符的索引

第三个整个字符串

**量词:**

**+**  最小一个最大不限

修饰前面的规则数量

**修饰符:**

**g** global 全局

特性:

贪婪:

一直找

懒惰:

找到一次之后就不再找了（直接返回）

\*/

**//match 小栗子**

//console.dir(new String());

//var str = 'dsd5637X2328236jdsdscxdxc37X83kljk54ls3x2xkd4rnej7';

//console.log(str.match(/\d+/g)); //找到全部连续数字

//console.log(str.match(/x+/ig)); //找到全部大小写的x

//console.log(str.match(new RegExp('x+','ig'))); //全写正则方法：找到全部大小写的x

//console.log(str.match(/z+/ig)); //null

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***正则的常用方法-replace**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

**string.replace(**字符串||正则,字符串||回调函数)

替换匹配的字符串

**第一个参数:**

匹配的字符串

**第二个参数:**

替换的字符串

返回值：

替换后的字符串。

|| 或 |

敏感词替换

**回调函数参数（默认情况无子项）：**

第一个参数：

找到匹配的字符

第二个参数:

索引

第三个参数:

整个字符串

第三个之后:

undefined

\*/

**//replace 小栗子**

//var s = 'miaov';

//console.log(s.replace('m','M')); //小写m替换成大写M

//var str = '中国共产党总书记习近平说:“法轮功是邪教”';

// var s = str.replace(/共产党|习近平|法轮|邪教/g,function($0,$1,$2,$3){

// // console.log($3);

// var temp = '';

// for(var i=0;i<$0.length;i++){

// temp += '\*';

// }

// return temp;

// });

// console.log(s); //替换了敏感词

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***转义符 /D 量词 ? 子项()**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

**转义符：**

**\D -** 一个非数字

**量词:**

**?** - 要么有1个要么1个都没有

**子项:**

**()**

从左往右去数

**() 提权**

(1+1)\*5

每有一个子项，就对应回调函数中的参数（从第二个参数起）

$0永远是本次匹配的整个字符

**转义符 /D 量词 ? 子项()小栗子：**

格式化日期:

2017-10-13

2017/10/13

2017年10月13日

2017.10.13

2017-.--\_10\_-13

把上面不规则的转成 2017年10月13日

var s = '2017p.--i10\_-13';

如果要匹配字符后面有 +号，那么小括号又包着匹配的字符，

结果为最后一个字符

2017

(\d)+ -> 7

(\d) -> 2

((\d+)(\D+)((\d)+)\D+)((\d)+) 7个子项

$1 2017-.--\_10\_-

$2 2017

$3 \_

$4 10

$5 0

$6 13

$7 3

2017p.--i10\_-13

(\d+)(\D)+((((\d+)(\D)+)(\d+)))\D?

$1 2017

$2 i

$3 10\_-13

$4 10\_-13

$5 10\_-

$6 10

$7 -

$8 13

(\d+(\D+((\d)+\D)+)\d+)

\*/

//var s = '2017-10-13';

// s.replace(/(\d+)(\D)+((((\d+)(\D)+)(\d+)))\D?/g,function($0,$1,$2,$3,$4,$5,$6,$7,$8){

// console.log($8);

// });

// s.replace(/((\d+)(\D+)((\d)+)\D+)((\d)+)/,function($0,$1,$2,$3,$4,$5,$6,$7){

// console.log($7);

// });

**//方法1：**

//s = s.replace(/(\d+)\D+(\d+)\D+(\d+)\D?/,function($0,$1,$2,$3,$4,$5,$6){

//return $1 + '年' + $2 + '月' + $3 + '日';

//});

//console.log(s);

**//方法2:**

// var arr = s.match(/\d+/g);

// s = arr[0] + '年' + arr[1] + '月' + arr[2] + '日';

// console.log(s);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***转义符 \w** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

**\w**  **一个数字|字母|下划线**

\* \*/

// **\w小栗子:**

// var obj = {

// name:'小小',

// age:'14',

// sex:'男'

// }

// var str = '我的名字叫{name},今年{age}岁,性别{sex}';

// var n = str.replace(/{(\w+)}/g,function($0,$1){

// return obj[$1];

// })

// console.log(n)

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* [] ^开头 $结尾 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**在中括号中，只选择一个**

**如果要找数字 0-9**

**字母 a-z**

18-108岁

[18-108]

dsadsaddsdsad sadsadsadsad jhh kjh jkdsa

**^ 开头**

**$ 结尾**

**// []小栗子：**

//var str = 'a1ca2ca3ca4ca23caacabcaccadcaAcaBcaZc';

//console.log(str.match(/a\dc/g)); //["a1c", "a2c", "a3c", "a4c"]

//console.log(str.match(/a(2|3)c/g)); //["a2c", "a3c"]

//console.log(str.match(/a[1-4]c/g)); //["a1c", "a2c", "a3c", "a4c"]

//console.log(str.match(/a[A-z]c/g)); //["aac", "abc", "acc", "adc", "aAc", "aBc", "aZc"]

//console.log(str.match(/a[A-Za-z]c/g)); //["aac", "abc", "acc", "adc", "aAc", "aBc", "aZc"]

//console.log(/[(18)-(108)][0-9]/.test('18')); //报错找不到

//console.log(/1[89]|[2-9][0-9]|10[0-8]/.test('108')); //true

**//^ 开头 方法：**

//console.log(/^1[89]$|^[2-9][0-9]$|^10[0-8]$/.test('108')); //true

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* **^排除法** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

**如果小尖尖在中括号中，就是排除的意思。**

\*/

var str = 'a1ca22ca3cabcafcayc';

//排除a2ca3c

//console.log(str.match(/a[^23y]c/g));

//console.log(str.match(/a[^1]2c/g));