**reg正则表达式:总结**

**[]:**备选字列表里面写字符挨着写不要有空格

强调: 字和字之间不要带任何分隔符！

效果: 多个字每次选其一匹配

一位汉字: [\u4e00-\u9fa5]旁边不能挨着字母,不然不生效

\d 等效于 [0-9]

\w 等效于 [0-9A-Za-z]

\s 空格、制表符Tab等一切看不见的空字符

**数量词**

有明确数量边界:

字符集{5} 必须重复出现5次

字符集{4,6} 4次到6次 最少4次 最多6次 (采用贪婪模式)

字符集{6,} 6次以上，多了不限有明确边界的:

没哟明确边界的:

字符集\* 可有可无，多了不限

字符集? 可有可无，最多一次

字符集+ 至少一个，多了不限

**选择:**

"|"为或(**不采用贪婪模式,是短路逻辑**)

**分组:可包含所有以上的内容**

what: 将多个规则分为一组先匹配，再和分组外的规则联合匹配

"**()**":多个规则写法里面可以包含([]|()?)

example:

**([我卧]|wo)\s\*([草艹槽]|cao)**

**特殊位置:**

三个位置: **字符串开头^，字符串结尾$，单词边界\b**

**零宽断言:**

**?=[] 等于 符合正则就]匹配下一位内容 从后往前匹配**

**?! 不等于 符合正则就]匹配下一位内容**

**?<= 从前面往后匹配**

**正则在函数中的使用**

在正则第二个/后加后缀i, 意为ignore 忽略(大小写！) 加g可以匹配多个敏感词

**一.查找敏感词:**

在函数中应该在/正则表达式/

**indexOf**

没有找到敏感词返回-1 **if(reg!=-1)就找到** 只能写固定敏感词,不能写正则 返回关键词下标位置i

**var msg=/正则/i**

**search**

没有找到敏感词返回-1  **if(reg!=-1)就找到** 可以写正则表达式 返回关键词下标位置i

**var msg=/正则/i**

**match**

**1.没找到返回一个空数组,**  **if(编写正则的变量名!=null)**就找到可以写正则表达式 返回关键词以及数组下标的位置

**var msg=/正则/i**

example:

**数组名称即写正则的变量**=[ "**0**": "敏感词的内容", "**index"**: 敏感词的下标位置 ]

不能修改 0,和index; 信任 ex: **编写正则的变量名[0],编写正则的变量名.index;**

2.可以查找所有敏感词,但是不能查找敏感词的位置

**var msg=/正则/g**

**二.替换敏感词**

**replace**

**用法: var reg=变量名.replace(/正则/g,"替换的字符")**

**查找敏感词返回时要用变量接住 即**保存在变量中 在打印变量 ex: str=str.replace(/正则/g,"新值") console.log(str)

**高级:var reg=变量名.replace(/正则/g,回调函数ex:function(kw){return kw.Uppercase()})**

**查找敏感词返回时要用变量接住 即**保存在变量中 在打印变量 ex: str=str.replace(/正则/g,"回调函数") console.log(str)

**额外:repeat(数值/长度) example:function(kw){"\*".repeat(kw.length)}**

删除关键词: 其实就是将找到的关键词替换为""即删除开头和结尾的空字符 替换为""

用法: 看实例 (^\s+/g,"")

**三.分割split**

**用法:var str=str.split("@");//str为字符串**

**返回:保存在数组中返回,之后可以使用数组家函数api**

**打散字符串就得用到分割split**

**四.RegExp对象,正则对象(动态生成表达式/格式验证/)**

**RegExp正则表达的本质是:**

1.new RegExp("正则","ig")可以写js; 简写:"//或//ig"不能写js表达式 适合固定的写死法:电话,身份证,邮箱,车牌等;

强调:只是正则写法变,其他函数写法不变 ex:arr=input.match(new RegExp(kws为服务器端传回的数组.join("|")));实例参考day02 regexp对象

2**.验证字符串函数test():**"返回true or false 可以直接写在条件中ex:if(正则.test(str变量))

**强调:使用此函数时必须添加表示前后的正则符^正则$**

3.**高级查找exec();**

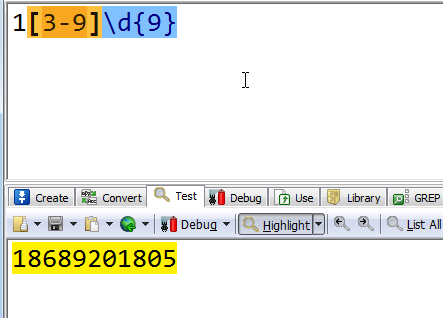
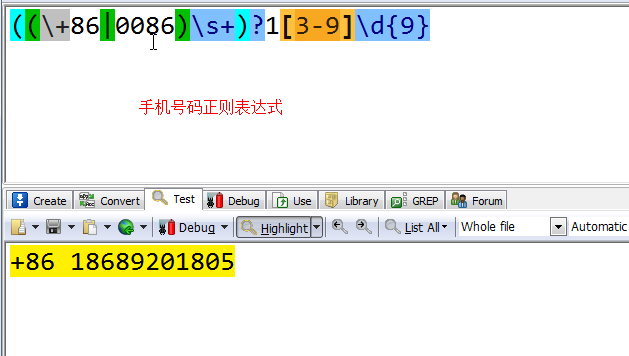
**用法:正则写在循环的外面 var reg=//g; do{var arr接收保存数据=reg.exec(str字符串变量)}**

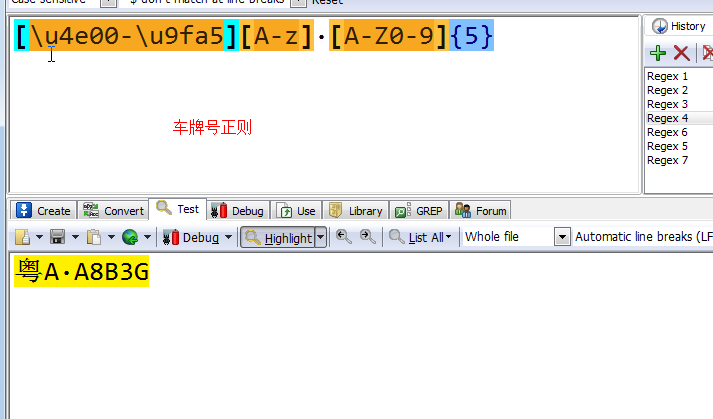
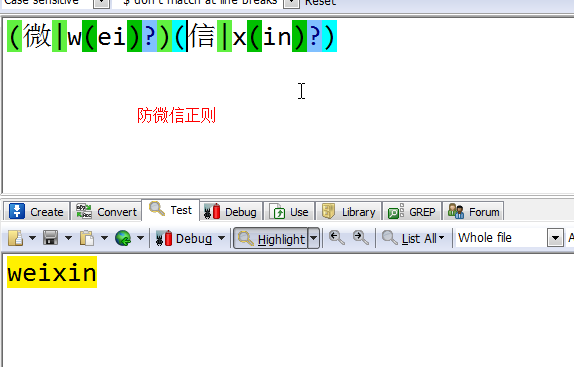
**返回 数组 找不到返回null if(reg!=null) reg.index reg[0] 使用do{}while()查找下一个;**

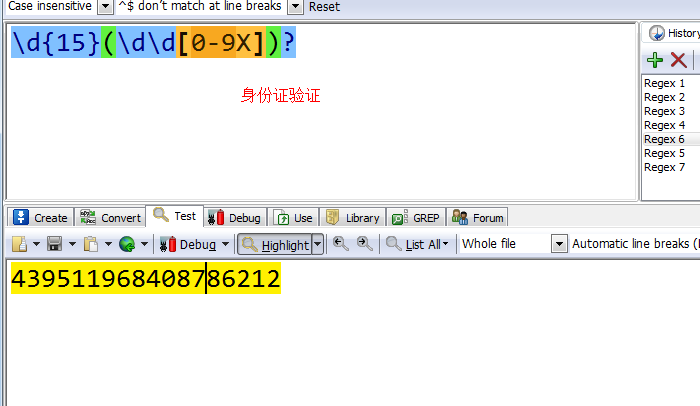
**案例:**

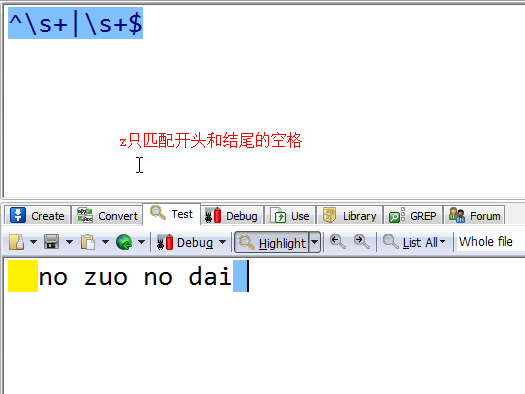
匹配4位或6位数字的验证码 采用贪婪模式

**\d{4}(\d{2})?**









笔试题

