正则表达式

一、什么是正则表达式:是对字符串操作的一种逻辑公式，事先定义好的一些特定的字符、及这些特定字符的组合，组成一个“规则字符串”,这些特定字符串是对字符串一种过滤逻辑

二、正则表达式的作用:

给定一个正则表达式和一个字符串，我们可以做到以下目的

1.给定的字符串是否符合正则表达式的过滤逻辑

2.可以通过正则表达式，从字符串获取我们想要的特定的部分

三、正则表达式的特点:

1.灵活性，逻辑性和功能性非常强

2.可以迅速的用极简单的方式达到对字符串的复杂的控制

3.对于没有基础的人来说有点难理解

四、创建正则表达式方法

1.构造函数方式 var reg=new RegExp("表达式","附加的参数")

2.普通方式 var reg=/表达式/附加参数

五、正则表达式的使用表示方式

1.test() 语法: reg.test(str) 如果匹配成功，返回true 否则返回false

2.exec() 语法 reg.exec(str) 如果匹配成功，返回的内容,否则返回null

3.replace() 语法 str.replace(reg,"替换成字符") 返回替换后的内容

六、正则表达式的元素

[0-9] 数字

[a-z] 字母

[A-z] 大小写字母 [a-zA-Z]

[\u4e00-\u9fa5] 中文汉字

[0-9A-z] 数字或字母

[0-9A-z\_]数字或字母或下划线

\w 相当于 [A-z0-9\_]

\W 和\w 相反，不包含[A-z0-9\_]

\d 相当 [0-9]

\D 和 \d 不包含 [0-9]

量词

+ 至少包含一个前导字符

? 包含0个或1个前导字符

\* 包含0个或多个前导字符

. 匹配任意字符 有几个点，就匹配几个

{x} 匹配包含x个前导字符

{x,y} 匹配包含x到y个前导字符

{x,} 匹配至少包含x个前导字符

$匹配字符串的行尾，$

^ 字符串开头

正则表达式的应用

1.只能是中文 长度为2-7位

/^([\u4e00-\u9fa5]){2,7}$/

2.长度为8到18位的小写大写字母或数字或下划线

/^[a-zA-Z0-9\_]{6,10}$/

3.正确的邮箱格式

/^[A-Za-z0-9\u4e00-\u9fa5\_]+@[a-zA-Z0-9\_-]+(\.[a-zA-Z0-9\_-]+)+$/

4.正确的手机号

/^1(3|4|5|7|8)\d{9}$/

5.正确的身份证号:

/^\d{15}|\d{17}([0-9]|x)/

/^[1-9][0-9]{16}[0-9x]$/

6.正确的qq号

/[1-9][0-9]{4,}/

7.正确的电话号码格式

/^0[0-9]{2}-[0-9]{8}|0[0-9]{3}-[0-9]{7}$/