**String 内置对象 字符串**

**String**

**Var str=’’;**

**Var str=’hello’;**

**Var str1=’hello’;**

**Var str=new String(‘hello’);**

**属性：length**

**JS自带函数**concat  
将两个或多个字符的文本组合起来，返回一个新的字符串。  
var a = "hello";  
var b = ",world";  
var c = a.concat(b);  
alert(c);  
//c = "hello,world"  
indexOf  
返回字符串中一个子串第一处出现的索引（从左到右搜索）。如果没有匹配项，返回 -1 。  
var index1 = a.indexOf("l");  
//index1 = 2  
var index2 = a.indexOf("l",3); 代表开始的索引处查找指定字符   
//index2 = 3  
charAt  
返回指定位置的字符。  
var get\_char = a.charAt(0);  
//get\_char = "h"  
lastIndexOf  
返回字符串中一个子串最后一处出现的索引（从右到左搜索），如果没有匹配项，返回 -1 。  
var index1 = lastIndexOf('l');  
//index1 = 3  
var index2 = lastIndexOf('l',2)  
//index2 = 2  
match  
检查一个字符串匹配一个正则表达式内容，如果么有匹配返回 null。  
var re = new RegExp(/^\w+$/);  
var is\_alpha1 = a.match(re);  
//is\_alpha1 = "hello"  
var is\_alpha2 = b.match(re);  
//is\_alpha2 = null  
substring  
返回字符串的一个子串，传入参数是起始位置和结束位置。  
var sub\_string1 = a.substring(1);  
//sub\_string1 = "ello"  
var sub\_string2 = a.substring(1,4);  
//sub\_string2 = "ell"  
substr  
返回字符串的一个子串，传入参数是起始位置和长度  
var sub\_string1 = a.substr(1);  
//sub\_string1 = "ello"  
var sub\_string2 = a.substr(1,4);  
//sub\_string2 = "ello"  
replace  
用来查找匹配一个正则表达式的字符串，然后使用新字符串代替匹配的字符串。  
var result1 = a.replace(re,"Hello");  
//result1 = "Hello"  
var result2 = b.replace(re,"Hello");  
//result2 = ",world"  
search  
执行一个正则表达式匹配查找。如果查找成功，返回字符串中匹配的索引值。否则返回 -1 。  
var index1 = a.search(re);  
//index1 = 0  
var index2 = b.search(re);  
//index2 = -1  
slice  
提取字符串的一部分，并返回一个新字符串（与 substring 相同）。  
var sub\_string1 = a.slice(1);  
//sub\_string1 = "ello"  
var sub\_string2 = a.slice(1,4);  
//sub\_string2 = "ell"  
split  
通过将字符串划分成子串，将一个字符串做成一个字符串数组。  
var arr1 = a.split("");  
//arr1 = [h,e,l,l,o]  
length  
返回字符串的长度，所谓字符串的长度是指其包含的字符的个数。  
var len = a.length();  
//len = 5  
toLowerCase  
将整个字符串转成小写字母。  
var lower\_string = a.toLowerCase();  
//lower\_string = "hello"  
toUpperCase  
将整个字符串转成大写字母。  
var upper\_string = a.toUpperCase();  
//upper\_string = "HELLO"  
  
/\*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
字符串函数扩充                                  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
\*/  
  
/\*  
===========================================  
//去除左边的空格  
===========================================  
  
\*/  
String.prototype.LTrim = function()  
{  
return this.replace(/(^\s\*)/g, "");  
}  
  
  
/\*  
===========================================  
//去除右边的空格  
===========================================  
\*/  
String.prototype.Rtrim = function()  
{  
return this.replace(/(\s\*$)/g, "");  
}  
  
/\*  
===========================================  
//去除前后空格  
===========================================  
\*/  
String.prototype.Trim = function()  
{  
return this.replace(/(^\s\*)|(\s\*$)/g, "");  
}  
  
/\*  
===========================================  
//得到左边的字符串  
===========================================  
\*/  
String.prototype.Left = function(len)  
{  
  
if(isNaN(len)||len==null)  
{  
len = this.length;  
}  
else  
{  
if(parseInt(len)<0||parseInt(len)>this.length)  
{  
len = this.length;  
}  
}  
  
return this.substr(0,len);  
}  
  
  
/\*  
===========================================  
//得到右边的字符串  
===========================================  
\*/  
String.prototype.Right = function(len)  
{  
  
if(isNaN(len)||len==null)  
{  
len = this.length;  
}  
else  
{  
if(parseInt(len)<0||parseInt(len)>this.length)  
{  
len = this.length;  
}  
}  
  
return this.substring(this.length-len,this.length);  
}  
  
  
/\*  
===========================================  
//得到中间的字符串,注意从0开始  
===========================================  
\*/  
String.prototype.Mid = function(start,len)  
{  
return this.substr(start,len);  
}  
  
  
/\*  
===========================================  
//在字符串里查找另一字符串:位置从0开始  
===========================================  
\*/  
String.prototype.InStr = function(str)  
{  
  
if(str==null)  
{  
str = "";  
}  
  
return this.indexOf(str);  
}  
  
/\*  
===========================================  
//在字符串里反向查找另一字符串:位置0开始  
===========================================  
\*/  
String.prototype.InStrRev = function(str)  
{  
  
if(str==null)  
{  
str = "";  
}  
  
return this.lastIndexOf(str);  
}  
  
/\*  
===========================================  
//计算字符串打印长度  
===========================================  
\*/  
String.prototype.LengthW = function()  
{  
return this.replace(/[^\x00-\xff]/g,"\*\*").length;  
}  
  
/\*  
===========================================  
//是否是正确的IP地址  
===========================================  
\*/  
String.prototype.isIP = function()  
{  
  
var reSpaceCheck = /^(\d+)\.(\d+)\.(\d+)\.(\d+)$/;  
  
if (reSpaceCheck.test(this))  
{  
this.match(reSpaceCheck);  
if (RegExp.$1 <= 255 && RegExp.$1 >= 0  
&& RegExp.$2 <= 255 && RegExp.$2 >= 0  
&& RegExp.$3 <= 255 && RegExp.$3 >= 0  
&& RegExp.$4 <= 255 && RegExp.$4 >= 0)  
{  
return true;      
}  
else  
{  
return false;  
}  
}  
else  
{  
return false;  
}  
  
}  
  
  
/\*  
===========================================  
//是否是正确的长日期  
===========================================  
\*/  
String.prototype.isLongDate = function()  
{  
var r = this.replace(/(^\s\*)|(\s\*$)/g, "").match(/^(\d{1,4})(-|\/)(\d{1,2})\2(\d{1,2}) (\d{1,2}):(\d{1,2}):(\d{1,2})$/);  
if(r==null)  
{  
return false;  
}  
var d = new Date(r[1], r[3]-1,r[4],r[5],r[6],r[7]);  
return (d.getFullYear()==r[1]&&(d.getMonth()+1)==r[3]&&d.getDate()==r[4]&&d.getHours()==r[5]&&d.getMinutes()==r[6]&&d.getSeconds()==r[7]);  
  
}  
  
/\*  
===========================================  
//是否是正确的短日期  
===========================================  
\*/  
String.prototype.isShortDate = function()  
{  
var r = this.replace(/(^\s\*)|(\s\*$)/g, "").match(/^(\d{1,4})(-|\/)(\d{1,2})\2(\d{1,2})$/);  
if(r==null)  
{  
return false;  
}  
var d = new Date(r[1], r[3]-1, r[4]);  
return (d.getFullYear()==r[1]&&(d.getMonth()+1)==r[3]&&d.getDate()==r[4]);  
}  
  
/\*  
===========================================  
//是否是正确的日期  
===========================================  
\*/  
String.prototype.isDate = function()  
{  
return this.isLongDate()||this.isShortDate();  
}  
  
/\*  
===========================================  
//是否是手机  
===========================================  
\*/  
String.prototype.isMobile = function()  
{  
return /^0{0,1}13[0-9]{9}$/.test(this);  
}  
  
/\*  
===========================================  
//是否是邮件  
===========================================  
\*/  
String.prototype.isEmail = function()  
{  
return /^\w+((-\w+)|(\.\w+))\*\@[A-Za-z0-9]+((\.|-)[A-Za-z0-9]+)\*\.[A-Za-z0-9]+$/.test(this);  
}  
  
/\*  
===========================================  
//是否是邮编(中国)  
===========================================  
\*/  
  
String.prototype.isZipCode = function()  
{  
return /^[\\d]{6}$/.test(this);  
}  
  
/\*  
===========================================  
//是否是有汉字  
===========================================  
\*/  
String.prototype.existChinese = function()  
{  
//[\u4E00-\u9FA5]為漢字﹐[\uFE30-\uFFA0]為全角符號  
return /^[\x00-\xff]\*$/.test(this);  
}  
  
/\*  
===========================================  
//是否是合法的文件名/目录名  
===========================================  
\*/  
String.prototype.isFileName = function()  
{  
return !/[\\\/\\*\?\|:"<>]/g.test(this);  
}  
  
/\*  
===========================================  
//是否是有效链接  
===========================================  
\*/  
String.prototype.isUrl = function()  
{  
return /^http[s]?:\/\/([\w-]+\.)+[\w-]+([\w-./?%&=]\*)?$/i.test(this);  
}  
  
  
/\*  
===========================================  
//是否是有效的身份证(中国)  
===========================================  
\*/  
String.prototype.isIDCard = function()  
{  
var iSum=0;  
var info="";  
var sId = this;  
  
 var aCity={11:"北京",12:"天津",13:"河北",14:"山西",15:"内蒙古",21:"辽宁",22:"吉林",23:"黑龙 江",31:"上海",32:"江苏",33:"浙江",34:"安徽",35:"福建",36:"江西",37:"山东",41:"河南",42:"湖 北",43:"湖南",44:"广东",45:"广西",46:"海南",50:"重庆",51:"四川",52:"贵州",53:"云南",54:"西藏",61:"陕西",62:"甘肃",63:"青海",64:"宁夏",65:"新疆",71:"台湾",81:"香港",82:"澳门",91:"国外"};  
  
if(!/^\d{17}(\d|x)$/i.test(sId))  
{  
return false;  
}  
sId=sId.replace(/x$/i,"a");  
//非法地区  
if(aCity[parseInt(sId.substr(0,2))]==null)  
{  
return false;  
}  
  
var sBirthday=sId.substr(6,4)+"-"+Number(sId.substr(10,2))+"-"+Number(sId.substr(12,2));  
  
var d=new Date(sBirthday.replace(/-/g,"/"))  
  
//非法生日  
if(sBirthday!=(d.getFullYear()+"-"+ (d.getMonth()+1) + "-" + d.getDate()))  
{  
return false;  
}  
for(var i = 17;i>=0;i--)  
{  
iSum += (Math.pow(2,i) % 11) \* parseInt(sId.charAt(17 - i),11);  
}  
  
if(iSum%11!=1)  
{  
return false;  
}  
return true;  
  
}  
  
/\*  
===========================================  
//是否是有效的电话号码(中国)  
===========================================  
\*/  
String.prototype.isPhoneCall = function()  
{  
return /(^[0-9]{3,4}\-[0-9]{3,8}$)|(^[0-9]{3,8}$)|(^\([0-9]{3,4}\)[0-9]{3,8}$)|(^0{0,1}13[0-9]{9}$)/.test(this);  
}  
  
  
/\*  
===========================================  
//是否是数字  
===========================================  
\*/  
String.prototype.isNumeric = function(flag)  
{  
//验证是否是数字  
if(isNaN(this))  
{  
  
return false;  
}  
  
switch(flag)  
{  
  
case null:        //数字  
case "":  
return true;  
case "+":        //正数  
return                /(^\+?|^\d?)\d\*\.?\d+$/.test(this);  
case "-":        //负数  
return                /^-\d\*\.?\d+$/.test(this);  
case "i":        //整数  
return                /(^-?|^\+?|\d)\d+$/.test(this);  
case "+i":        //正整数  
return                /(^\d+$)|(^\+?\d+$)/.test(this);                         
case "-i":        //负整数  
return                /^[-]\d+$/.test(this);  
case "f":        //浮点数  
return                /(^-?|^\+?|^\d?)\d\*\.\d+$/.test(this);  
case "+f":        //正浮点数  
return                /(^\+?|^\d?)\d\*\.\d+$/.test(this);                         
case "-f":        //负浮点数  
return                /^[-]\d\*\.\d$/.test(this);                 
default:        //缺省  
return true;                         
}  
}  
  
/\*  
===========================================  
//是否是颜色(#FFFFFF形式)  
===========================================  
\*/  
String.prototype.IsColor = function()  
{  
var temp        = this;  
if (temp=="") return true;  
if (temp.length!=7) return false;  
return (temp.search(/\#[a-fA-F0-9]{6}/) != -1);  
}  
  
/\*  
===========================================  
//转换成全角  
===========================================  
\*/  
String.prototype.toCase = function()  
{  
var tmp = "";  
for(var i=0;i<this.length;i++)  
{  
if(this.charCodeAt(i)>0&&this.charCodeAt(i)<255)  
{  
tmp += String.fromCharCode(this.charCodeAt(i)+65248);  
}  
else  
{  
tmp += String.fromCharCode(this.charCodeAt(i));  
}  
}  
return tmp  
}  
  
/\*  
===========================================  
//对字符串进行Html编码  
===========================================  
\*/  
String.prototype.toHtmlEncode = function()  
{  
var str = this;  
  
str=str.replace(/&/g,"&amp;");  
str=str.replace(/</g,"&lt;");  
str=str.replace(/>/g,"&gt;");  
str=str.replace(/\'/g,"&apos;");  
str=str.replace(/\"/g,"&quot;");  
str=str.replace(/\n/g,"<br>");  
str=str.replace(/\ /g,"&nbsp;");  
str=str.replace(/\t/g,"&nbsp;&nbsp;&nbsp;&nbsp;");  
  
return str;  
}  
  
/\*  
===========================================  
//转换成日期  
===========================================  
\*/  
String.prototype.toDate = function()  
{  
try  
{  
return new Date(this.replace(/-/g, "\/"));  
}  
catch(e)  
{  
return null;  
}  
}