JavaScript中的String对象

# W3school学习网站

<http://www.w3school.com.cn/jsref/jsref_obj_string.asp>

# JavaScript中String对象

## String对象

字符串是 JavaScript 的一种基本的数据类型。

**String 对象**用于处理**文本（字符串）**。**String 类**定义了大量**操作字符串**的方法，例如从字符串中提取字符或子串，或者检索字符或子串。

**需要注意的是**，JavaScript 的字符串是**不可变的（immutable）**，String 类定义的方法都不能改变字符串的内容。像 String.toUpperCase() 这样的方法，返回的是全新的字符串，而不是修改原始字符串。

在较早的 Netscape 代码基的 JavaScript 实现中（例如 Firefox 实现中），字符串的行为就像**只读的字符数组**。例如，从字符串 s 中提取第三个字符，可以用 s[2] 代替更加标准的 **s.charAt(2)**。此外，对字符串应用 **for/in 循环**时，**它将枚举字符串中每个字符的数组下标**（但要注意，ECMAScript 标准规定，不能枚举 length 属性）。因为字符串的数组行为不标准，所以应该避免使用它。

## 创建字符串对象

new String(s);

String(s);

**参数 *s***是要存储在 String 对象中或转换成原始字符串的值。

通常， JavaScript 字符串是原始值，可以使用字符创建： **var firstName = "John"，**但我们也可以使用 new 关键字将字符串定义为一个对象： **var firstName = new String("John")。**

注意：不要创建 String 对象，它会拖慢执行速度，并可能产生其他副作用。

原始值可以使用 JavaScript 的属性和方法，因为 JavaScript 在执行方法和属性时可以把原始值当作对象。

### 当 String() 和运算符 new 一起作为构造函数使用时，它返回一个新创建的 String 对象，存放的是字符串 s 或 s 的字符串表示。

### 当不用 new 运算符调用 String() 时，它只把 s 转换成原始的字符串，并返回转换后的值。

如： **var str2 = new String("nihaoma");**

**var str3 = String("huiyizongxiangku");**

**var str4 = "jiyili";**

一个字符串可以使用**单引号或双引号**。

var str1 = "中国是一个伟大的国家";

## 注意单引号与双引号的使用

**单引号可以含有包含双引号，双引号中可以包含单引号，且不需要转义。**

**单引号中包含的单引号需要转义，双引号中包含的双引号也需要转义，转义符号"\"。**

var str2 = '"日本","中国"';

var str3 = "'美国','加拿大'";

var str4 = "\"日本\",\"中国\"";

var str5 = '\'美国\',\'加拿大\'';

var answer="It's alright";

var answer='It\'s alright';

## String对象与String对象的原始值

利用String(s)或直接引号括起来，都是原始的字符串。

利用String对象的**valueOf()方法**可以获取String对象的原始值。

# 字符串String对象的属性介绍

## JavaScript中String对象的属性

JavaScript中String对象具有三个属性：分别为

length、prototype、constructor。



## 获取字符串的长度--length属性

**字符串（String）**使用长度属性length来计算字符串的长度，length 属性可返回字符串中的字符数目。

**语法：stringObject.length**

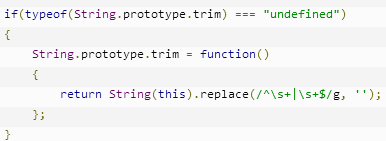
注意：Java中是利用length()方法获取字符串的长度。

## Prototype：允许您向对象添加属性和方法。

Object.prototype 属性表示 Object 的**原型对象**。

可以为字符串添加新的方法。

如添加trim方法。



## constructor属性：返回创建字符串属性的函数。

# 字符串String对象的方法介绍

## 常用方法

### charAt()、charCodeAt()

### concat()、fromCharCode()

### indexOf()、lastIndexOf()

### valueOf()

### match()、replace()、search()、split()

### slice()、substr()、substring()：获取子字符串

### toLowerCase()、toUpperCase()

### join()：字符串数组的方法

## 全部方法









# 获取单个字符或子字符串

## 获取单个字符：

根据**位置（索引）**获取字符串中任何的字符或者利用charAt(index)方法获取。

索引范围为：**0到str.length-1**

**字符串的索引从零开始, 所以字符串第一字符为 [0]**，最后一个字符为[str.length-1]。

如：var str1 = "中国是一个伟大的国家";

alert(str1[0]);//中

alert(str1[str1.length-1]);//家

**for(var i = 0;i < str1.length;i++){ //定义i时，var可有可无**

**alert(str1[i]);**

**}**

## 获取子字符串：字符串的截取

JavaScript中有三个常用函数用于字符串的截取：

### slice(start,[end])函数

slice(start,[end])：

①第一个参数代表**开始位置**,第二个参数代表结束位置的**下一个位置**；

②截取出来的字符串的长度为第二个参数与第一个参数之间的差;如slice(0,1)截取第一个字符。

③若参数值为负数,则将该值**加上字符串长度**后转为正值;

④若第一个参数**等于大于**第二个参数,则返回**空字符串**，非undefined；

⑤若只有一个参数，默认从参数索引至结束。

var str1 = "中国是一个伟大的国家";

alert(str1.slice(0,1));//中

alert(str1.slice(1,1));//空字符串，注意不是undefined

**alert(str1.slice(-5,-3));//伟大**

**alert(str1.slice(-5+str1.length,-3+str1.length));//伟大**

**注意：这里和python的语法相似，但是不如python的灵活。负数作为索引，可以获取倒数第几的字符，但是由于第二个索引为-1，获取不到最后一个，用0不行，所以只能获取倒数第二2个及之上。如alert(str1.slice(-3,-1));//的国**

**要求：第一个索引值必须小于第二个索引，且要为负数同为负数。**

### substring(start,[end])函数

1. 第一个参数代表**开始位置**,第二个参数代表**结束位置的下一个位置**;
2. **若参数值为负数,则将该值转为0;**
3. **两个参数中,取较小值作为开始位置**；

④截取出来的字符串的长度为较大值与较小值之间的差；

1. 若只有一个参数，默认从参数索引至结束。

alert(str1.substring(0,2));//的国

alert(str1.substring(-7,2));//的国

**alert(str1.substring(6,2));//是一个伟**

**alert(str1.substring(2,6));//是一个伟**

### substr(start,[length])

第一个参数代表**开始位置**,第二个参数代表**截取的长度。**

**第一个参数指代的索引可以为负数也可以为正数或0，第二个参数为正数。**

alert(str1.substr(-3,3));//的国家

alert(str1.substr(7,3));//的国家

### 三个函数的总结：

从这三个函数的特性来看，第三个函数substr(start,[length])最好使用。slice()与substring()区别在于负值索引。三个函数都可以使用一个参数，slice(start)与substr(start)作用一样，而**substring(start)**把**负值索引**默认为0。

# JavaScript中String对象常用方法

## indexOf()与lastIndexOf()方法

字符串使用 **indexOf()** 来**定位字符串**中某一个指定的字符**首次出现的位置**；

功能：返回字符串中匹配子串的第一个字符的下标。

如果没找到对应的字符函数返回**-1**；**lastIndexOf()** 方法在**字符串末尾**开始查找字符串出现的位置。

var str="Hello world, welcome to the universe.";  
var n=str.indexOf("welcome");

注意：中文汉字返回的索引不一定是汉字的索引，因为有些汉字占据多个字节。

## match()函数：内容匹配

**match()函数用来查找字符串中特定的字符**，并且如果找到的话，则返回这个字符。

如果找不到，则返回null。

## replace****(srcstr,replacestr):替换字符串或正则表达式****

**replace(srcstr,replacestr) 方法在字符串中用某些字符替换另一些字符，返回替换后的新的字符串。**

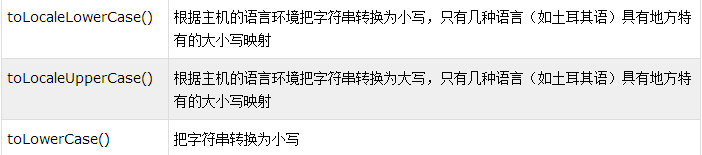
**示例：**

**var str1 = "China is a beautiful Country";**

**var str2 = str1.replace("China","America");**

## toUpperCase() / toLowerCase():字符串大小写转换

字符串大小写转换使用函数，返回转换之后的字符串。





## split(delimiter)方法:分割，返回数组。

功能：使用一个**指定的分隔符**把**一个字符串**分割存储到数组。

var str1 = "zhaohong,zhaoxiyuan,zhaoshuo,zhaoxiping";

**var strs = str1.split(",");**

for(i = 0;i<strs.length;i++){

alert(strs[i]);

}

## search(regexp)：检索字符串或正则表达式，返回匹配的起始位置，或-1。

search(regexp) 方法用于**检索字符串中指定的子字符串**，或**检索与正则表达式相匹配的子字符串**。

语法：stringObject.search(regexp)

regexp 该参数可以是需要在 stringObject 中检索的子串，也可以是需要检索的 RegExp 对象。注释：要执行忽略大小写的检索，请追加标志 i。

**返回值：**stringObject 中第一个与 regexp 相匹配的子串的**起始位置**。

注释：如果没有找到任何匹配的子串，则返回 -1。

**说明：search() 方法**不执行全局匹配，它将忽略标志 g。它同时忽略 regexp 的 lastIndex 属性，并且总是从字符串的开始进行检索，这意味着它总是返回 stringObject 的第一个匹配的位置。

## concat()：连接两个或多个字符串。

concat() 方法用于连接两个或多个字符串。

语法：stringObject.concat(stringX,stringX,...,stringX)

**concat() 方法**将把它的所有参数转换成字符串，然后按顺序连接到字符串 stringObject 的尾部，**并返回连接后的字符串**。请注意，stringObject 本身并没有被更改。

stringObject.concat() 与 Array.concat() 很相似。

提示和注释

提示：请注意，使用 " + " 运算符来进行字符串的连接运算通常会更简便一些。一般利用“+”实现字符串的连接。

## valueOf()方法：返回某个字符串对象的原始值

## 字符串数组的join(delimiter)方法

使用**您选择的分隔符**将一个数组合并为一个字符串

**var delimitedString=myArray.join(delimiter);**

例子：var myList=new Array(”jpg”,”bmp”,”gif”,”ico”,”png”);

var portableList=myList.join(”|”);//结果是jpg|bmp|gif|ico|png

# 字符串数组

## 字符串数组的定义

利用方括号[]定义一个字符串数组。

var strArray = ["zhaohong","zhaoxiyuan","zhaoshuo","zhaoxiping"];

## 字符串数组与字符串之间的转换

### 字符串转换为字符串数组

利用字符串对象的**split(delimiter)**方法。

### 字符串数组转换为字符串

利用**字符串数组**的**join方法**。

函数：John(separator)

功能：使用**您选择的分隔符**将一个数组合并为一个字符串

**var delimitedString=myArray.join(delimiter);**

# JavaScript中的特殊字符

**反斜杠**是一个转义字符；转义字符将特殊字符转换为字符串字符：可以用于转义**撇号，换行，引号，等其他特殊字符**。

下表中列举了在字符串中可以使用转义字符转义的特殊字符：

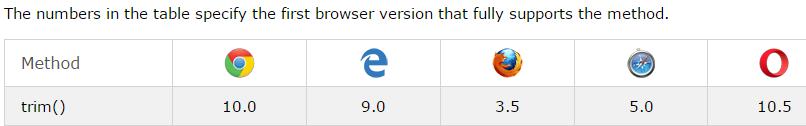


# 关于 JavaScript 里的 trim()/strip() 方法

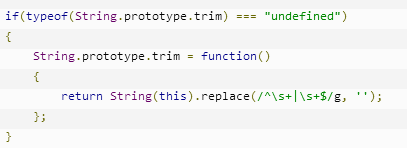
trim() **移除字符串首尾空白**。

在其他语言中常常用 **trim()/strip() 方法**脱去不必要的**空格**等元素。

JS trim()方法使用上有**浏览器限制**: 浏览器版本限制：JavaScript Version 1.8



**JavaScript 本身并不提供 trim() 方法**，不过可以用正则表达式，通过给 **String 对象增加 trim 方法**的方式实现。实现方法：



或者自定义工具方法，对字符串进行处理。

function myTrim(x) {

return **x.replace(/^\s+|\s+$/gm,'')**;

}//最后的m可有可无？

# JavaScript中字符串数组的遍历

对字符串应用 **for/in 循环**时，它将**枚举字符串中每个字符的数组下标**。

var strArray = ["zhaohong","zhaoxiyuan","zhaoshuo","zhaoxiping"];

for(var **strIndex** in **strArray**){

alert(strArray[**strIndex**]);

}

# 关于汉字replace的问题

在网上看到另一种非常简单的方法，代码如下：

　　function func(s, n) {

　　　　return s.replace(/([^x00-xff])/g, “$1a”).slice(0, n).replace(/([^x00-xff])a/g, “$1″);

　　}

这个方法非常巧妙，而且基本上是正确的。说“基本上”是因为它在取“123汉字测试”左边长度为 6 的子串时，它返回的是“123汉字”，而不是“123汉”。当然，这也并不一定就是问题，某些情况下需求可能就是这样。这个方法还可以再改进一下，如下：

　function func(s, n) {

return s.slice(0, n).replace(/([^x00-xff])/g, “$1a”).slice(0, n).replace(/([^x00-xff])a/g, “$1″);

　　}