Javascript

2017年8月2日

17:11

* + **JavaScript是一种脚本语言**
    - **JavaScript是一种解释型语言（这意味着脚本的执行没有初步编译)**
  + **JavaScript被设计为HTML页面添加交互性**
  + **JavaScript通常是直接嵌入到HTML页面，但是写成单独的**[**js**](https://baike.baidu.com/item/js/10687961)**文件有利于结构和行为的**[**分离**](https://baike.baidu.com/item/%E5%88%86%E7%A6%BB)

Javascript能做什么？

**JavaScript是无数的网页应用的改进设计，为HTML页面添加动态内容，读写HTML元素，对事件作出反应，验证表单，检测浏览器，创建cookies，和更多.**

**JavaScript是互联网上最流行的脚本语言，并可以运行在所有主要的浏览器**

**插入Javascript的方法：**

* + **使用script标记对将脚本嵌入到网页中（嵌入式）**

**在html文档的head标签或者body标签中**

<script type="text/javascript">

alert(666);

</script>

* + **直接将脚本嵌入到HTML标记的事件中（行内式）**

<a href="javascript:alert(1 + 1);">计算</a>

* + **通过script标记的src属性链接外部脚本文件（链接式）**

<script src=“外部js文件路径” type="text/javascript"></script>

Javascript语言基础

2017年8月2日

17:22

**标识符：变量名，方法名称，常量，参数名称等等**

**命名规则：任意数字，字母，下划线及$组成，**

**但是数字不能开头，不能使用关键字及保留字**

**，所谓弱类型变量是指它的变量无特定类型，定义任何变量都是用“var”关键字，并可以将其初始化为任何值，而且可以随意改变变量中所存储的数据类型，当然为了程序规范应该避免这样操作**

变量

**JavaScript的变量是一种弱类型变量**

**var name ="Six Hang";**

**var age＝28;**

**var school="CSU";**

**var male=true;**

**变量命名规范：第一个单词所有字母都小写，以后每个单词第一个字母大写，例如：sMyString**

**变量名对大小写敏感**

声明变量需使用var关键字

声明变量而不指定值，默认值为undefined

变量可以指向其他类型

**运算符**

* + **算术运算符**

**Operator Description Example Result**

**+ Addition x=2 y=2 x+y 4**

**- Subtraction x=5 y=2 x-y 3**

**\* Multiplication x=5 y=4 x\*y 20**

**/ Division 15/5 5/2 3 2.5**

**% Modulus 5%2 10%8 10%2 1 2 0**

**++ Increment x=5 x++ x=6**

**-- Decrement x=5 x-- x=4**

在+操作符中如果遇到字符串，会转换为字符串，进行字符串拼接

可以使用parseInt()及parseFloat()转换为对应的数值进行运算

* + **赋值运算符**

**= x=y x=y**

**+= x+=y x=x+y**

**-= x-=y x=x-y**

**\*= x\*=y x=x\*y**

**/= x/=y x=x/y**

**%= x%=y x=x%y**

* + **比较运算符**

==、===

* + **逻辑运算符**
    - 逻辑与 &&
    - 逻辑或 ||

逻辑与 与 逻辑非表达式返回的是第一个短路运算后的值

在js中，条件可以接收任意类型的数据

String：非空字符串为true，空字符串(“”)为false；

Number：0为false，非0数字为true

Boolean：true/false

NaN：false

Null：false

Undefined：false

逻辑非

逻辑或和逻辑与不会改变变量本身的值，而逻辑非会改变变量本身的值（boolean）

技巧：两个取反操作可以将任意类型的数据转换为boolean值

* + **条件运算符**

**varablename=(condition)?value1:value2**

字符串String

* + **字符串由零个或多个字符构成，字符可以是字母、数字、标点符号或空格。字符串必须放在单引号或双引号中。例如：**

**var course="data structure"**

**var case='the birthday"19801106"'**

* + **还可以使用转义字符（escaping）“\”实现特殊字符按原样输出：**

**var score = " run time 3\' 15\""**

* + **常见属性及方法**

**var myString = "This is a sample";**

**（1）length属性：它返回字符串中字符的个数，例如：**

**var name="Six tang";**

**alert (name.length); //返回11**

**（2）charAt方法：它返回字符串对象在指定位置处的字符，第一个字符位置是0。例如：**

**myString.charAt(2)；//返回i**

**（3）charCodeAt：返回字符串对象在指定位置处字符的10进制的ASCII码**

**myString.charCodeAt(2)；//返回105**

**（4）indexOf：要查找的子串在字符串对象中的位置**

**myString.indexOf(“is”)；//返回2**

**（5）substr方法：截取字串**

**myString.substr(10,3)；//返回sam，其中10表示位置，3表示长度**

**（6）substring方法：截取字串**

**myString.substring(5,9)；//返回is a,其中5表示开始位置，9表示结束位置**

**（7）split方法：分隔字串到一个数组中**

**var a = myString.split(" ");**

**//a[0] = “This” a[1]=“is” a[2]=“a” a[3]=“sample”**

**（8）replace方法**

**myString.replace(“sample”,”apple”);**

**//结果”This is a apple”**

**（9）toLowerCase方法：变成小写字母**

**myString.toLowerCase()；// this is a sample**

**(10) toUpperCase()方法：变成大写写字母**

**“this is a sample”.toUpperCase()// THIS IS A SAMPLE**

**数值型（number）**

**在Javascript中，数值型数据不区分整型和浮点型，数值型数据和字符型数据的区别是数值型数据不要用引号括起来。例如下面都是正确的数值表示法，**

**var num1=23.45**

**var num2=76**

**var num3=-9e5 //科学计数法，即-900000**

**alert(num1+ " "+ num2+" "+ num3);**

NaN：Not a Number

Infinity：除数为0

Tip:执行算术会自动类型转换

**布尔型数据Boolean**

**取值只有两个：true和false。**

**布尔型数据不能用引号引起来，否则就变成字符串了**

**var married = true;**

typeof

数组Array

* + **数组使用关键字Array来声明，同时还可以指定这个数组元素的个数，也就是数组的长度（length），例如：**

**var rank =new Array(12); //论坛的用户共分12级**

**var Map =["China", "USA", "Britain"];**

**var myColor= new Array();**

**myColor[0]= "blue";**

**myColor[1]= "yellow";**

**myColor[2]= "purple";**

**myColor[3]= "red";**

**Tip：js中数组长度是可变的**

* + **数组的常用属性和方法：**

**（1）length属性：用来获取数组的长度，而数组的位置同样也是从0开始的。例如：**

**var Map =new Array("China", "USA", "Britain");**

**alert(Map.length+" "+ Map[2]); //返回3 Britain**

**（2）toString方法：将数组转化为字符串**

**var Map =new Array("China", "USA", "Britain");**

**alert(Map.toString()+" "+ typeof(Map.toString()));**

|  |  |
| --- | --- |
| [**concat()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_concat_array.asp.htm) | **连接两个或更多的数组，并返回结果** |

|  |  |  |  |
| --- | --- | --- | --- |
| [**join()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_join.asp.htm) | **把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔。** |  |  |
| [**pop()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_pop.asp.htm) | **删除并返回数组的最后一个元素** |  |  |
| [**push()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_push.asp.htm) | **向数组的末尾添加一个或更多元素，并返回新的长度。** |  |  |
| [**reverse()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_reverse.asp.htm) | **颠倒数组中元素的顺序。** |  |  |
| [**shift()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_shift.asp.htm) | **删除并返回数组的第一个元素** |  |  |
| [**slice()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_slice_array.asp.htm) | **从某个已有的数组返回选定的元素** |  |  |
| [**sort()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_sort.asp.htm) | **对数组的元素进行排序** |  |  |
| [**splice()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_splice.asp.htm) | **删除元素，并向数组添加新元素。** |  |  |
| [**toSource()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_tosource_array.asp.htm) | **返回该对象的源代码。** |  |  |
| [**toString()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_toString_array.asp.htm) | **把数组转换为字符串，并返回结果。** |  |  |
| [**toLocaleString()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_toLocaleString_array.asp.htm) | **把数组转换为本地数组，并返回结果。** |  |  |
| [**unshift()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_unshift.asp.htm) | **向数组的开头添加一个或更多元素，并返回新的长度。** |  |  |
| [**valueOf()**](mk:@MSITStore:C:\Users\Administrator\Desktop\W3CSchool.chm::/www.w3school.com.cn/js/jsref_valueof_array.asp.htm) | 返回数组对象的原始值 |  |  |

js中的语句

条件语句

If

If else

switch

循环语句

while（1到100）

do-while（输入评价）

for（九九乘法表）

for-in（从数组中查找）

Break

continue（1到100所有偶数的和）

**<table cellpadding="6" cellspacing="0" style="border-collapse:collapse; border:none;">**

**<script language="javascript">**

**for(var i=1;i<10;i++){ //乘法表一共九行**

**document.write("<tr>"); //每行是table的一行**

**for(j=1;j<10;j++) //每行都有9个单元格**

**if(j<=i) //有内容的单元格**

**document.write("<td style='border:2px solid #004B8A; background: white;'>"+i+"\*"+j+"="+(i\*j)+"</td>");**

**else //没有内容的单元格**

**document.write("<td style='border:none;'></td>");**

**document.write("</tr>");**

**}</script>**

**函数**

函数是一个可重用的代码块，可用来完成某个特定功能。每当需要反复执行一段代码时，可以利用函数来避免重复输入大量的相同内容。

**声明**

Function 函数名称(形式参数){

语句;

Returen ;

}

Tip:可以在声明之前调用

**函数声明(匿名函数）**

var volume = function(width,height,dept){

var area = width \* height;

var volume = area \* dept;

return [area,volume];

}

alert(volume(2,3,4));

Tip:该种方式只能在声明后调用

**立即调用函数表达式 IFFY**

var area1 = (function(width,height,dept){

//var width = 2,height=4,dept=6;

var area = width \* height;

var volume = area \* dept;

return [area,volume];

}(2,4,6));

alert(area1);

声明后立即调用，只调用一次，一般用作调用函数的参数值传递

**参数对象：arguments**

**通过参数对象arguments可得到所有的参数（参数是可变的）**

**全局变量与局部变量**

在方法里使用var声明的变量，不管出现在什么地方， 等价于在第一行进行声明。

var msg = “全局变量”;

function show(){

alert(msg);

var msg = “局部变量”;

alert(msg);

}

show();//undefined 局部变量

作用域链：

Js会从当前作用域一直向上追溯，到找到该变量（全局window范围内），js的当前执行环境永远放在作用域的最前端

在函数外部使用var声明以及没有使用var声明的变量（包含函数内部声明的）都是全局变量

* 1. **对象**

特点

创建

this

原型

* 1. **日期对象**

Date 对象用于处理日期和时间。

**语法：**

**var date=new Date()**

**注释：Date 对象会自动把当前日期和时间保存为其初始值。**

* 1. **Math对象**

Math对象也是JavaScript的一个内置对象，不需要由函数创建，用于执行数学任务

|  |  |  |
| --- | --- | --- |
| Random() | 返回 0 ~ 1 之间的随机数 | 返回 0 ~ 1 之间的随机数  ,不包括0和1 |
| Round(x) | 四舍五入取整 | 与 x 最接近的整数 |
| Ceil(x) | 对数进行上舍入 | 大于等于 x，并且与它最接近的整数 |
| Floor(x) | 对数进行下舍入 | 小于等于 x，且与 x 最接近的整数 |

**周期执行函数**

**setInterval()** 方法可按照指定的周期（以毫秒计）来调用函数或计算表达式。返回一个代表该周期函数（定时器对象）的id

setInterval() 方法会不停地调用函数，直到 clearInterval() 被调用或窗口被关闭。由 setInterval() 返回的 ID 值可用作 clearInterval() 方法的参数。

**语法**

setInterval(code,millisec[,"lang"])

|  |  |
| --- | --- |
| **参数** | **描述** |
| code | 必需。要调用的函数或要执行的代码串。 |
| millisec | 必须。周期性执行或调用 code 之间的时间间隔，以毫秒计。 |
|  |  |

**setTimeout()** 方法用于在指定的毫秒数后调用函数或计算表达式。

**语法**

setTimeout(code,millisec)

|  |  |
| --- | --- |
| **参数** | **描述** |
| code | 必需。要调用的函数后要执行的 JavaScript 代码串。 |
| millisec | 必需。在执行代码前需等待的毫秒数。 |

提示：setTimeout() 只执行 code 一次。如果要多次调用，请使用 setInterval() 或者让 code 自身再次调用 setTimeout()。

setInterval 不断地执行指定代码直到调用clearInterval清除定时器对象

setTimeout 执行一次指定代码，使用clearTimeout清除定时器对象