Javascript

2017年8月2日

17:11

演示百度地图等

* + **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)
  + **java和javascript一样吗?**

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 ，NaN为false

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**

String

Number

Boolean

Object

Undefined

Null

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

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。**

**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

话术：

* + **定时操作通常有两种使用目的，一种是周期性地执行脚本，例如在页面上显示时钟，需要每隔一秒钟更新一次时间的显示，另一种则是将某个操作延时一段时间执行，例如迫使用户等待一段时间才能进行操作，可以使用window.setTimeout函数使其延时执行，而后面的脚本可以继续运行不受影响。**
  + **需要注意的是，定时操作函数还是利用JavaScript制作网页动画效果的基础，例如网页上的漂浮广告，就是每隔几毫秒更新一下漂浮广告的显示位置。其他的如文字打字效果，图片轮转显示等，可以说一切用Javascipt实现的动画效果都离不开setTimeout函数。**
  1. **对象**

**计算机编程中抽取现实生活中的物理物件（实体）**

特点

属性：变量表示属性

事件：如何与对象进行交互

方法：通过函数表示方法

创建对象

A、字面量方式

var porsche = {

color : 'red',

name : '911',

currentSpeed : 200,

changeSpeed : function(){

this.currentSpeed--;

}

};

属性用变量表示，方法用函数表示，采用键值对方式进行定义，多个成员用，分割，最后一个成员不能有，

访问对象属性

1.采用.操作符进行访问

2.采用['属性名']进行访问

B、对象构造函数方式

var hotel = new Object();

给对象添加属性及方法

hotel.name = '希尔顿大酒店';

hotel.booking = function(){

alert(this.name);

}

删除属性等可以采用delete关键字删除

C、采用构造函数语法进行创建

function Car(name,color){

this.name = name;

this.color = color;

this.run = function(){

alert(this.name + "=" + this.color);

}

}

**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

**浏览器对象模型BOM(了解，拓展内容，可关注红色部分)**

2017年8月3日

10:44

* + **BOM（Browser Object Model，浏览器对象模型）**

**JavaScript是运行在浏览器中的，因此提供了一系列对象用于与浏览器窗口进行交互。**

**这些对象主要有：**

**window、document、location、navigator和screen等。**

**window对象**

对应着Web浏览器的窗口，使用它可以直接对浏览器窗口进行操作。

window对象提供的主要功能可以分为以下5类：

1. **调整窗口的大小和位置（了解）**

**window.moveBy(dx, dy)**

该方法将浏览器窗口相对于当前的位置移动指定的距离（相对定位），当dx和dy为负数时则向反方向移动。

**window.moveTo(x, y)**

该方法将浏览器窗口移动到屏幕指定的位置（x、y处）（绝对定位）。同样可使用负数，只不过这样会把窗口移出屏幕。

**window.resizeBy(dw, dh)**

相对于浏览器窗口的当前大小，把宽度增加dw个像素，高度增加dh个像素。两个参数也可以使用负数来缩写窗口。

**window.resizeTo(w, h)**

把窗口大小调整为w像素宽，h像素高，不能使用负数。

1. **打开新窗口（\*）**

**window.open([url] [, target] [, options])**

options参数可能的选项包括：

height: 窗口的高度，单位为像素

width：窗口的宽度，单位为像素

left：窗口的左边缘位置

top：窗口的上边缘位置

**父子窗口交互**

window.opener; 可以得到打开本页面的父窗口的window对象

1. **系统提示框**

**window.alert([message]);**

**window.confirm([message])**

该方法将显示一个确认提示框，其中包括“确定”和“取消”按钮。

用户单击“确定”按钮时，window.confirm返回true

单击“取消”按钮时，window.confirm返回false。

<a href="目标地址" onclick='return confirm("确认删除?")'></a>

**window.prompt([message] [, default])**

该方法将显示一个消息提示框，其中包含一个文本输入框。输入框能够接受用户输入参数，从而实现进一步的交互。该方法接受两个参数，第一个参数是显示给用户的文本，第二个参数为文本框中的默认值（可以为空）。整个方法返回字符串，值即为用户的输入

1. **状态栏控制**

**status属性**

状态栏控制

window.status="看看状态栏中的文字变化了吗？";

**history属性**

浏览器后退和前进属性。它可以访问历史页面，但不能获取到历史页面的URL

window.history.go(-1);//后退一页

**window.history.back();**

window.history.go(1);//前进一页

**window.history.forward();**

window.history.go(0);//刷新显示当前页

**location.reload();**

1. **定时操作**

function popwin() {

|  |  |
| --- | --- |
|  | var oWin = window.open("pop.html", "new", "width=400,height=300"); |
|  | setTimeout(init, 2000); |
|  | oWin.resizeTo(600,400); |
|  | oWin.moveTo(200,200); |
|  | } |

function search(){

|  |  |
| --- | --- |
|  | var parent = window.opener; |
|  | if (!parent) return; |
|  | // 从父窗口中获取id为title的文本框中输入的内容，把它填入子窗口相关位置 |
|  | $("title").value = parent.document.getElementById("title").value; |
|  | } |
|  | function $(id) { |
|  | return document.getElementById(id); |
|  | } |
|  | </script> |

|  |  |
| --- | --- |
|  | var win; |
|  | // 打开检索条件输入窗口 |
|  | function search() { |
|  | // 设置窗口的位置，使它位于屏幕的中央 |
|  | win = window.open("7-20子窗口.html", "win", "height=200px,width=300px,left=" + |
|  | (screen.availWidth/2 - 150) + ",top=" + (screen.availHeight/2 - 100)); |
|  | } |

**地址栏控制（location对象）**

**location对象的主要作用是分析和设置页面的URL地址**

**它是window对象和document对象的属性**

**location对象表示窗口地址栏中的URL**

**一般做分页，后面我们在项目中使用**

**<select name="select" onchange="window.location.href=this.value">**

**<option>请选择需要的网址</option>**

**<option value="http://www.sohu.com">搜狐</option>**

**<option value="http://www.sina.com">新浪</option>**

**<option value="http://www.MSN.com.cn">MSN中国</option>**

**</select>**

**replace()方法：**

**禁用“后退”按钮返回原来的页面**

<p onclick="location.replace('http://www.sohu.com');">访问搜狐</p>

**navigator对象**

**用来检测客户端浏览器信息，如：**

**浏览器的类型、版本信息以及操作系统的类型**

**常用的属性：userAgent**

通常浏览器及操作系统的判断都是通过该属性来实现的

**screen对象**

**用来获取用户电脑的屏幕信息，包括屏幕的分辨率，屏幕的颜色位数，窗口可显示的最大尺寸。**

1.根据屏幕分辨率打开适合的网页

2. 使浏览器窗口自动满屏显示

window.moveTo(0,0);

window.resizeTo(screen.availWidth,screen.availHeight);

**文档对象模型DOM**

2017年8月3日

11:21

**文档对象模型DOM（Document Object Module）**

**定义了用户操纵文档对象的接口，它使得用户对HTML文档有了空前的访问能力。**

**DOM编程在HTML页面中的应用 ：**

**访问指定节点**

* getElementById(id)
* querySelector(css选择器);
* getElementsByName()
* getElementsByTagName()
* getElementsByClassName()
* querySelectorAll()

**访问相关节点**

根据已知的节点，寻找和它存在联系的节点，如父节点、子节点、兄弟节点等。

访问html节点和body节点

访问子节点

* **c**hildNodes：子节点的列表；
* firstChild：第一个子节点；
* lastChild：最后一个子节点。
* parentNode:父节点
* previousSibling/nextSibling:前一个/后一个节点

**访问节点属性(一般用于改变样式)**

* **hasAttribute(name)**
* getAttribute(name)
* setAttribute(name, value)
* removeAttribute(name)

|  |  |
| --- | --- |
|  | <title>修改元素属性</title>  <script language="JavaScript"> |
|  | function changePic(){ |
|  | var myImg = document.getElementsByTagName("img")[0]; //获取图片 |
|  | myImg.setAttribute("src","images/02.jpg"); //设置图片src和title属性 |
|  | myImg.setAttribute("title","兴盛公寓"); |
|  | myImg.setAttribute("width","300"); |
|  | } |
|  | </script> |

**检查节点类型（nodeType）**

DOM中的节点主要有三种类型，分别是元素节点、属性节点和文本节点

* 元素节点的nodeType值为1；
* 属性节点的nodeType值为2；
* 文本节点的nodeType值为3。

**操作节点**

**createElement() /** cloneNode(true)**：创建节点**

**createTextNode():设置内容**

**appendChild()：附加节点**

**removeChild()：删除节点**

**操作节点内容**

**innerHTML：html内容**

**textContent：元素文本内容**

|  |  |
| --- | --- |
| 遍历子元素 |  |
|  | function init() { |
|  | var oUl = document.body; //获取body节点 |
|  | var DOMString = ""; |
|  | if(oUl.hasChildNodes()){ //判断是否有子节点 |
|  | var oCh = oUl.childNodes; |
|  | for(var i=0;i<oCh.length;i++) //逐一查找 |
|  | DOMString += oCh[i].nodeName + "\n"; |
|  | } |
|  | alert(DOMString); |
|  | } |
|  | </script> |
|  | </head> |
|  | <body onload="init()"> |
|  | <h1> HTML Sample Page</h1> |
|  | <p> Hello, This is a sample page.</p> |
|  | </body> |

<title>访问元素属性</title>

|  |  |
| --- | --- |
|  | <script type="text/JavaScript"> |
|  | function init() { |
|  | var img = document.getElementById("img1"); |
|  | alert(img.className); |
|  | alert(img.getAttribute("className")); //IE支持 |
|  | alert(img.getAttribute("class")); //Firefox支持 |
|  | alert(img.src); |
|  | alert(img.getAttribute("src")); |
|  | } |
|  | </script> |
|  | </head> |
|  | <body onload="init()"> |
|  | <img id="img1" src="[images/01.jpg](file:///G:\old\UESTC\html\51CTO%E4%B8%8B%E8%BD%BD-%E3%80%8A%E5%9F%BA%E4%BA%8EWeb%E6%A0%87%E5%87%86%E7%9A%84%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1%E4%B8%8E%E5%88%B6%E4%BD%9C%E3%80%8B%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E3%80%8A%E5%9F%BA%E4%BA%8EWeb%E6%A0%87%E5%87%86%E7%9A%84%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1%E4%B8%8E%E5%88%B6%E4%BD%9C%E3%80%8B%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E3%80%8A%E5%9F%BA%E4%BA%8EWeb%E6%A0%87%E5%87%86%E7%9A%84%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1%E4%B8%8E%E5%88%B6%E4%BD%9C%E3%80%8B%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E7%AC%AC%E4%B8%83%E7%AB%A0\images\01.jpg)" alt="img1" class="class1"/> |
|  | </body> |

<script language="JavaScript"> <title>修改元素属性2</title>

|  |  |
| --- | --- |
|  | function changePic(){ |
|  | var myImg = document.getElementsByTagName("img")[0]; //获取图片 |
|  | myImg.setAttribute("src","images/02.jpg"); //设置图片src和title属性 |
|  | myImg.setAttribute("title","兴盛公寓"); |
|  | myImg.setAttribute("className","bk"); |
|  | myImg.setAttribute("class","bk"); |
|  | } |
|  | function remov(){ |
|  | var myImg = document.getElementsByTagName("img")[0]; //获取图片 |
|  | myImg.setAttribute("src","images/01.jpg"); //设置图片src和title属性 |
|  | myImg.setAttribute("title","酃湖"); |
|  | myImg.removeAttribute("className"); |
|  | myImg.removeAttribute("class"); |
|  | } |
|  | </script> |
|  | <style type="text/css"> |
|  | <!-- |
|  | .bk { |
|  | border: 5px dashed #666666; |
|  | } |
|  | --> |
|  | </style> |
|  | </head> |
|  | <body bgcolor="#F7F7F7"> |
|  | <img src="[images/01.jpg](file:///G:\old\UESTC\html\51CTO%E4%B8%8B%E8%BD%BD-%E3%80%8A%E5%9F%BA%E4%BA%8EWeb%E6%A0%87%E5%87%86%E7%9A%84%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1%E4%B8%8E%E5%88%B6%E4%BD%9C%E3%80%8B%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E3%80%8A%E5%9F%BA%E4%BA%8EWeb%E6%A0%87%E5%87%86%E7%9A%84%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1%E4%B8%8E%E5%88%B6%E4%BD%9C%E3%80%8B%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E3%80%8A%E5%9F%BA%E4%BA%8EWeb%E6%A0%87%E5%87%86%E7%9A%84%E7%BD%91%E9%A1%B5%E8%AE%BE%E8%AE%A1%E4%B8%8E%E5%88%B6%E4%BD%9C%E3%80%8B%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E6%BA%90%E4%BB%A3%E7%A0%81%E5%92%8C%E7%B4%A0%E6%9D%90\%E7%AC%AC%E4%B8%83%E7%AB%A0\images\01.jpg)" width="119" height="105" title="酃湖" onmouseover="changePic()" onmouseout="remov()"/> |

事件驱动

2017年8月3日

11:39

**事件**

指可以被浏览器识别的、发生在页面上的用户动作或状态变化。其中：

* 用户动作:用户对页面的鼠标或键盘操作。例如，click、keypress 等
* 状态变化：页面的状态发生变化。例如，load、resize、change 等

**事件驱动**

程序的一种执行方式，即响应事件的发生而执行相关的程序代码片段

在事件驱动执行方式下，程序执行顺序不是完全按照程序代码从头至尾顺序执行，

而是依据事件的发生顺序而执行

**事件处理程序**

* 事件处理：是对发生事件进行处理的行为、操作
* 事件处理程序：是对发生事件进行处理的程序代码片段

**事件驱动编程**

是指为需要处理的事件编写相应的事件处理程序

一般步骤：

第1步，确定响应事件的元素

第2步，为指定元素确定需要响应的事件类型

第3步，为指定元素的指定事件编写相应的事件处理程序

第4步，将事件处理程序绑定到指定元素的指定事件

**事件绑定**

将一个函数与某个 HTML 元素的事件属性关联起来，使得当相应事件发生时就会触发该函数的执行

事件绑定的两种方法：

**静态绑定**

指将处理事件的程序代码直接指定为 HTML 元素的事件属性值。如：

<button name="A1" onclick="alert('先生,您好!')">问侯先生</button>

函数调用形式：

<button name="A2" onclick="hello\_girl()">问侯小姐</button>

**动态绑定**

是指通过语句建立事件绑定

形式：

对象.事件属性 = 函数引用;

注：对象的事件属性是对函数对象的引用，不能赋值为字符串

**绑定多个事件处理函数**

**attachEvent( )**

**addEventListener("click",fnClick1,false)**

**detachEvent( )**

**removeEventListener()**

**事件属性**

使用事件属性可以为元素的指定事件绑定事件处理程序

事件属性名：在事件名的基础上，加上前缀“on”，如 onclick

**事件分类**

**1、页面事件：**是指因页面状态发生变化而产生的事件，包括 onload、onunload、onresize、onabort、 onerror等

**2、鼠标事件：**是指用户操作鼠标（点击或移动）而触发的事件，包括 onclick、ondblclick、onmousedown等

**3、键盘事件：**是指用户在键盘上敲击、输入时触发的事件，包括 onkeypress、onkeydown、onkeyup等

**4、表单事件：**是指与表单或表单控件相关的事件，包括 onsubmit、onreset、onchange、onselect、onblur和 onfocus 等

**事件对象Event || e**

* 通过 Event 对象，可以访问事件的发生状态，

如事件名、键盘按键状态、鼠标位置等信息

* 在 IE 中，通过 window 对象的 event 属性可以访问 Event 对象
* 只能在事件处理程序中访问 Event 对象

**事件冒泡**

* 指事件的冒泡传递过程

源元素→父元素→根对象（即 Document 对象）

* 由于存在事件流，因此当一个事件发生时，不仅可以由产生事件的元素响应，也可以由其他元素响应。
  + 有些事件（如 load、unload、blur、focus 等事件）不会传递
  + 访问源元素: event.srcElement
  + 阻止事件传递:将event.cancelBubble 赋值为 true

**阻止事件的默认行为**

1.阻止事件冒泡,使成为捕获型事件触发机制.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | function stopBubble(e) {  //如果提供了事件对象，则这是一个非IE浏览器  if ( e && e.stopPropagation )      //因此它支持W3C的stopPropagation()方法      e.stopPropagation();  else      //否则，我们需要使用IE的方式来取消事件冒泡      window.event.cancelBubble = true;  } |

2.当按键后,不希望按键继续传递给如HTML文本框对象时,可以取消返回值.即停止默认事件默认行为.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | //阻止浏览器的默认行为  function stopDefault( e ) {      //阻止默认浏览器动作(W3C)      if ( e && e.preventDefault )          e.preventDefault();      //IE中阻止函数器默认动作的方式      else          window.event.returnValue = false;      return false;  } |

**try...catch语句**

作用：捕获可能出现的异常

格式：

try {

tryStatements

}catch(exception){

catchStatements

}finally {

finallyStatements

}

**Error 对象**

catch 子句可以有参数 exception，引用一个描述错误信息的 Error 对象

属性：number、name、description

例

try {

alert(window.eval(Text1.value));

}catch(e){

alert("错误类型:"+e.name +"\n错误信息:"+e.description);

}

**throw 语句**

语法：

throw exception;

效果：相当于一条出现异常的语句

例如，以下语句执行时将生成一个异常：

throw "An error occurred";