最终目的：实现一个动态刷新数据的网页，并且带有声音提示。

JS 测验

**结果：**7/20

**您的回答：**

**1.我们可以在下列哪个 HTML 元素中放置 Javascript 代码？**

您的回答：<script>

**2.写 "Hello World" 的正确 Javascript 语法是？**

您的回答：response.write("Hello World")

正确答案：**document.write("Hello World")**

**3.插入 Javacript 的正确位置是？**

您的回答：<body> 部分

正确答案：<body> 部分和 <head> 部分均可

**4.引用名为 "xxx.js" 的外部脚本的正确语法是？**

您的回答：<script src="xxx.js">

**5.外部脚本必须包含 <script> 标签吗？**

您的回答：是

正确答案：否

**6.如何在警告框中写入 "Hello World"？**

您的回答：alert("Hello World")

**7.如何创建函数？**

您的回答：function=myFunction()

正确答案：function myFunction()

**8.如何调用名为 "myFunction" 的函数？**

您的回答：myFunction()

**9.如何编写当 i 等于 5 时执行一些语句的条件语句？**

您的回答：if i=5

正确答案：if (i==5)

**10.如何编写当 i 不等于 5 时执行一些语句的条件语句？**

您的回答：if <>5

正确答案：if (i != 5)

**11.在 JavaScript 中，有多少种不同类型的循环？**

您的回答：四种。for 循环、while 循环、do...while 循环以及 loop...until 循环。

正确答案：两种。for 循环和 while 循环。

**12.for 循环如何开始？**

您的回答：for (i = 0; i <= 5; i++)

**13.如何在 JavaScript 中添加注释？**

您的回答：<!--This is a comment-->

正确答案：//This is a comment

**14.可插入多行注释的 JavaScript 语法是？**

您的回答：<!--This comment has more than one line-->

正确答案：/\*This comment has more than one line\*/

**15.定义 JavaScript 数组的正确方法是？**

您的回答：var txt = new Array("George","John","Thomas")

**16.如何把 7.25 四舍五入为最接近的整数？**

您的回答：round(7.25)

正确答案：Math.round(7.25)

**17.如何求得 2 和 4 中最大的数？**

您的回答：ceil(2,4)

正确答案：Math.max(2,4)

**18.打开名为 "window2" 的新窗口的 JavaScript 语法是？**

您的回答：window.open("http://www.w3school.com.cn","window2")

**19.如何在浏览器的状态栏放入一条消息？**

您的回答：window.status("put your message here")

正确答案：window.status = "put your message here"

**20.如何获得客户端浏览器的名称？**

您的回答：browser.name

正确答案：navigator.appName

把脚本置于 **<body>** 元素的底部，可改善显示速度，因为脚本编译会拖慢显示。

## 外部脚本

脚本可放置与外部文件中：

### 外部文件：myScript.js

function myFunction() {

document.getElementById("demo").innerHTML = "段落被更改。";

}

外部脚本很实用，如果相同的脚本被用于许多不同的网页。

JavaScript 文件的文件扩展名是 **.js**。

如需使用外部脚本，请在 <script> 标签的 src (source) 属性中设置脚本的名称：

## 使用 document.write()

出于测试目的，使用 document.write() 比较方便：

## 使用 window.alert()

您能够使用警告框来显示数据：

## 使用 console.log()

在浏览器中，您可使用 console.log() 方法来显示数据。

请通过 F12 来激活浏览器控制台，并在菜单中选择“控制台”。

## JavaScript 语句

JavaScript 语句由以下构成：

值、运算符、表达式、关键词和注释。

这条语句告诉浏览器在 id="demo" 的 HTML 元素中输出 "Hello Kitty."：

### 实例

document.getElementById("demo").innerHTML = "Hello Kitty.";

## 定义和用法

innerHTML 属性设置或返回表格行的开始和结束标签之间的 HTML。

## 语法

HTMLElementObject.innerHTML=*text*

## JavaScript 空白字符

JavaScript 会忽略多个空格。您可以向脚本添加空格，以增强可读性。

下面这两行是相等的：

var person = "Bill";

var person="Bill";

在运算符旁边（ = + - \* / ）添加空格是个好习惯：

var x = y + z;

## JavaScript 行长度和折行

为了达到最佳的可读性，程序员们常常喜欢把代码行控制在 80 个字符以内。

如果 JavaScript 语句太长，对其进行折行的最佳位置是某个运算符：

### 实例

document.getElementById("demo").innerHTML =

"Hello Kitty.";

## JavaScript 代码块

JavaScript 语句可以用花括号（{...}）组合在代码块中。

代码块的作用是定义一同执行的语句。

您会在 JavaScript 中看到成块组合在一起的语句：

### 实例

function myFunction() {

document.getElementById("demo").innerHTML = "Hello Kitty.";

document.getElementById("myDIV").innerHTML = "How are you?";

}

## JavaScript 关键词

JavaScript 语句常常通过某个关键词来标识需要执行的 JavaScript 动作。

下面的表格列出了一部分将在教程中学到的关键词：

|  |  |
| --- | --- |
| **关键词** | **描述** |
| break | 终止 switch 或循环。 |
| continue | 跳出循环并在顶端开始。 |
| debugger | 停止执行 JavaScript，并调用调试函数（如果可用）。 |
| do ... while | 执行语句块，并在条件为真时重复代码块。 |
| for | 标记需被执行的语句块，只要条件为真。 |
| function | 声明函数。 |
| if ... else | 标记需被执行的语句块，根据某个条件。 |
| return | 退出函数。 |
| switch | 标记需被执行的语句块，根据不同的情况。 |
| try ... catch | 对语句块实现错误处理。 |
| var | 声明变量。 |

**注释：**JavaScript 关键词指的是保留的单词。保留词无法用作变量名。

## JavaScript 关键词

JavaScript **关键词**用于标识被执行的动作。

var 关键词告知浏览器创建新的变量：

var x = 7 + 8;

var y = x \* 10;

## JavaScript 注释

并非所有 JavaScript 语句都被“执行”。

双斜杠 // 或 /\* 与 **\*/** 之间的代码被视为**注释**。

注释会被忽略，不会被执行：

var x = 7; // 会执行

// var x = 8; 不会执行

## JavaScript 对大小写敏感

所有 JavaScript 标识符**对大小写敏感**。

变量 lastName 和 lastname，是两个不同的变量。

lastName = "Gates";

lastname = "Jobs";

## JavaScript 与驼峰式大小写

历史上，程序员曾使用三种把多个单词连接为一个变量名的方法：

### 连字符：

first-name, last-name, master-card, inter-city.

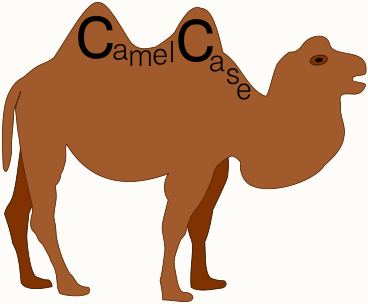
**注释：**JavaScript 中不能使用连字符。它是为减法预留的。

### 下划线：

first\_name, last\_name, master\_card, inter\_city.

### 驼峰式大小写（Camel Case）：

FirstName, LastName, MasterCard, InterCity.



JavaScript 程序员倾向于使用以小写字母开头的驼峰大小写：

firstName, lastName, masterCard, interCity

## JavaScript 字符集

JavaScript 使用 **Unicode** 字符集。

Unicode 覆盖世界上几乎所有的字符、标点和符号。

如需了解详情，请访问我们完整的 [Unicode 参考手册](http://www.w3school.com.cn/tags/html_ref_utf8.asp)。

## 单行注释

单行注释以 // 开头。

任何位于 // 与行末之间的文本都会被 JavaScript 忽略（不会执行）。

本例在每条代码行之前使用单行注释：

### 实例

// 改变标题：

document.getElementById("myH").innerHTML = "我的第一张页面";

// 改变段落：

document.getElementById("myP").innerHTML = "我的第一个段落。";

本例在每行结尾处使用了单行注释来解释代码：

### 实例

var x = 5; // 声明 x，为其赋值 5

var y = x + 2; // 声明 y，为其赋值 x + 2

本例使用注释块来阻止多行代码的执行：

**实例**

/\*

document.getElementById("myH").innerHTML = "我的第一张页面";

document.getElementById("myP").innerHTML = "我的第一个段落。";

\*/

## JavaScript 标识符

所有 JavaScript **变量**必须以**唯一的名称**的**标识**。

这些唯一的名称称为**标识符**。

标识符可以是短名称（比如 x 和 y），或者更具描述性的名称（age、sum、totalVolume）。

构造变量名称（唯一标识符）的通用规则是：

* 名称可包含字母、数字、下划线和美元符号
* 名称必须以字母开头
* 名称也可以 $ 和 \_ 开头（但是在本教程中我们不会这么做）
* 名称对大小写敏感（y 和 Y 是不同的变量）
* 保留字（比如 JavaScript 的关键词）无法用作变量名称

**提示：**JavaScript 标识符对大小写敏感。

## JavaScript 数据类型

JavaScript 变量可存放数值，比如 100，以及文本值，比如 "Bill Gates"。

在编程中，文本值被称为字符串。

JavaScript 可处理多种数据类型，但是现在，我们只关注数值和字符串值。

字符串被包围在双引号或单引号中。数值不用引号。

如果把数值放在引号中，会被视作文本字符串。

### 实例

var pi = 3.14;

var person = "Bill Gates";

var answer = 'How are you!';

## JavaScript 算数运算符

算数运算符用于对数字执行算数运算：

|  |  |
| --- | --- |
| **运算符** | **描述** |
| + | 加法 |
| - | 减法 |
| \* | 乘法 |
| / | 除法 |
| % | 系数 |
| ++ | 递加 |
| -- | 递减 |

**注释：**[JS 算数](http://www.w3school.com.cn/js/js_arithmetic.asp)这一章对算数运算符进行了完整描述。

## JavaScript 赋值运算符

赋值运算符向 JavaScript 变量赋值。

|  |  |  |
| --- | --- | --- |
| **运算符** | **例子** | **等同于** |
| = | 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 |

加法赋值运算符（+=）向变量添加一个值。

### 赋值

var x = 7;

x += 8;

## JavaScript 比较运算符

|  |  |
| --- | --- |
| **运算符** | **描述** |
| == | 等于 |
| === | 等值等型 |
| != | 不相等 |
| !== | 不等值或不等型 |
| > | 大于 |
| < | 小于 |
| >= | 大于或等于 |
| <= | 小于或等于 |
| ? | 三元运算符 |

**注释：**[JS 比较](http://www.w3school.com.cn/js/js_comparisons.asp)这一章中完整描述了比较运算符。

## JavaScript 逻辑运算符

|  |  |
| --- | --- |
| **运算符** | **描述** |
| && | 逻辑与 |
| || | 逻辑或 |
| ! | 逻辑非 |

**注释：**[JS 比较](http://www.w3school.com.cn/js/js_comparisons.asp)这一章中完整描述了逻辑运算符。

## JavaScript 类型运算符

|  |  |
| --- | --- |
| **运算符** | **描述** |
| typeof | 返回变量的类型。 |
| instanceof | 返回 true，如果对象是对象类型的实例。 |

**注释：**[JS 类型转换](http://www.w3school.com.cn/js/js_type_conversion.asp)这一章完整描述了类型运算符。

## JavaScript 算数运算符

算术运算符对数值（文字或变量）执行算术运算。

|  |  |
| --- | --- |
| **运算符** | **描述** |
| + | 加法 |
| - | 减法 |
| \* | 乘法 |
| \*\* | 幂（[ES2016](http://www.w3school.com.cn/js/js_es6.asp)） |
| / | 除法 |
| % | 系数 |
| ++ | 递增 |
| -- | 递减 |

## 算数运算符

典型的算术运算会操作两个数值。

这两个数可以是字面量：

### 实例

var x = 7 + 8;

或变量：

### 实例

var x = a + b;

## JavaScript 运算符优先级值

|  |  |  |  |
| --- | --- | --- | --- |
| **值** | **运算符** | **描述** | **实例** |
| 20 | ( ) | 表达式分组 | (3 + 4) |
|  |  |  |  |
| 19 | . | 成员 | person.name |
| 19 | [] | 成员 | person["name"] |
| 19 | () | 函数调用 | myFunction() |
| 19 | new | 创建 | new Date() |
|  |  |  |  |
| 17 | ++ | 后缀递增 | i++ |
| 17 | -- | 后缀递减 | i-- |
|  |  |  |  |
| 16 | ++ | 前缀递增 | ++i |
| 16 | -- | 前缀递减 | --i |
| 16 | ! | 逻辑否 | !(x==y) |
| 16 | typeof | 类型 | typeof x |
|  |  |  |  |
| 15 | \*\* | 求幂 (ES7) | 10 \*\* 2 |
|  |  |  |  |
| 14 | \* | 乘 | 10 \* 5 |
| 14 | / | 除 | 10 / 5 |
| 14 | % | 模数除法 | 10 % 5 |
|  |  |  |  |
| 13 | + | 加 | 10 + 5 |
| 13 | - | 减 | 10 - 5 |
|  |  |  |  |
| 12 | << | 左位移 | x << 2 |
| 12 | >> | 右位移 | x >> 2 |
| 12 | >>> | 右位移（无符号） | x >>> 2 |
|  |  |  |  |
| 11 | < | 小于 | x < y |
| 11 | <= | 小于或等于 | x <= y |
| 11 | > | 大于 | x > y |
| 11 | >= | 大于或等于 | x >= y |
| 11 | in | 对象中的属性 | "PI" in Math |
| 11 | instanceof | 对象的实例 | instanceof Array |
|  |  |  |  |
| 10 | == | 相等 | x == y |
| 10 | === | 严格相等 | x === y |
| 10 | != | 不相等 | x != y |
| 10 | !== | 严格不相等 | x !== y |
|  |  |  |  |
| 9 | & | 按位与 | x & y |
| 8 | ^ | 按位 XOR | x ^ y |
| 7 | | | 按位或 | x | y |
| 6 | && | 逻辑与 | x && y |
| 5 | || | 逻辑否 | x || y |
| 4 | ? : | 条件 | ? "Yes" : "No" |
|  |  |  |  |
| 3 | = | 赋值 | x = y |
| 3 | += | 赋值 | x += y |
| 3 | -= | 赋值 | x -= y |
| 3 | \*= | 赋值 | x \*= y |
| 3 | %= | 赋值 | x %= y |
| 3 | <<= | 赋值 | x <<= y |
| 3 | >>= | 赋值 | x >>= y |
| 3 | >>>= | 赋值 | x >>>= y |
| 3 | &= | 赋值 | x &= y |
| 3 | ^= | 赋值 | x ^= y |
| 3 | |= | 赋值 | x |= y |
|  |  |  |  |
| 2 | yield | 暂停函数 | yield x |
| 1 | , | 逗号 | 7 , 8 |

**注意：**淡红色指示实验性或建议性的技术（ECMASScript 2016 或 ES7）

当数值和字符串相加时，JavaScript 将把数值视作字符串

JavaScript 从左向右计算表达式。不同的次序会产生不同的结果：

## JavaScript 拥有动态类型

JavaScript 拥有动态类型。这意味着相同变量可用作不同类型：

### 实例

var x; // 现在 x 是 undefined

var x = 7; // 现在 x 是数值

var x = "Bill"; // 现在 x 是字符串值

## JavaScript 数组

JavaScript 数组用方括号书写。

数组的项目由逗号分隔。

下面的代码声明（创建）了名为 cars 的数组，包含三个项目（汽车品牌）：

### 实例

var cars = ["Porsche", "Volvo", "BMW"];

数组索引基于零，这意味着第一个项目是 [0]，第二个项目是 [1]，以此类推。

您将在本教程中学到更多有关数组的知识。

## Undefined

在 JavaScript 中，没有值的变量，其值是 undefined。typeof 也返回 undefined。

### 实例

var person; // 值是 undefined，类型是 undefined

任何变量均可通过设置值为 undefined 进行清空。其类型也将是 undefined。

## 空值

空值与 undefined 不是一回事。

空的字符串变量既有值也有类型。

### 实例

var car = ""; // 值是 ""，类型是 "string"

## Null

在 JavaScript 中，null 是 "nothing"。它被看做不存在的事物。

不幸的是，在 JavaScript 中，null 的数据类型是对象。

您可以把 null 在 JavaScript 中是对象理解为一个 bug。它本应是 null。

您可以通过设置值为 null 清空对象：

### 实例

var person = null;

您也可以通过设置值为 undefined 清空对象：

**实例**

var person = undefined; // 值是 undefined，类型是 undefined

## Undefined 与 Null 的区别

Undefined 与 null 的值相等，但类型不相等：

typeof undefined // undefined

typeof null // object

null === undefined // false

null == undefined // true