第 5 章　JavaScript 基础

一、单元概述

通过本单元的学习，学生能够了解 JavaScript 的基本语法，学会使用 JavaScript 的内置对象、BOM 对象、DOM 对象和它们的常用方法，编写 JavaScript 事件处理函数，学习如何使用 JavaScript 控制客户端行为。通过 JavaScript 实际案例的编写过程，学习 JavaScript 语言的编程思路和编程经验；记忆 JavaScript 的基本语法结构。

二、单元重点与难点

重点：

(1)JavaScript 基本语法、变量定义、常见数据类型、运算符的使用。

(2)JavaScript 定义函数的语法、常见的事件和事件处理过程、事件处理函数的编写。

(3)JavaScript 内置对象的使用。

(4)JavaScript 的 BOM 对象和 DOM 对象的区别、常见属性和方法的使用。

难点：

(1)JavaScript 事件处理函数的编写。

(2)window 对象、document 对象的常见属性和方法的使用。

重点及难点学习指导建议：

(1)重点掌握 JavaScript 的函数和事件处理，先学会编写简单的事件处理响应函数。

(2)通过大量的编程练习记忆 JavaScript 中的常用 BOM 和 DOM 对象的方法的使用格式。

(3)在此基础上，独立完成每个章节中的知识运用部分，体会使用到的知识点的具体作用。

三、知识单元正文

5.1　JavaScript 语法介绍

JavaScript 是一种面向对象的客户端脚本语言，它的运行不需要 Web 服务器的支持，可以直接在客户端本地运行。因此，也就解决了由于网络速度的限制，而造成的响应速度缓慢

的问题，可以为用户提供更流畅快速的访问效果。有些功能比如用户输入数据验证，交由 JavaScript 这种客户端语言来实现更为合适。本章主要介绍 JavaScript 的基本语法、流程控制语句、函数和事件以及常用对象和方法的使用等内容。

5.1.1　JavaScript 简介

JavaScript 是由 Netscape 公司的 LiveScript 发展而来的。在 1996 年，Netscape 和 Microsof 公司的浏览器基本都已支持 JavaScript。JavaScript 的正式名称是“ECMAScript”。这个标准由 ECMA 组织发展和维护。ECMA-262 是正式的 JavaScript 标准。在 1998 年，该标准成为了国际 ISO 标准 (ISO/IEC 16262)。在 2005 年 12 月，ECMA 发布 ECMA-357 标准(ISO/IEC 22537)，主要增加了对扩展标记语言 XML 的有效支持。

JavaScript 是一种客户端语言，它可以同 HTML 标签结合到一起共同工作。利用 JavaScript，可以不依靠网络传输来直接响应客户端的需求事件。所以当用户输入信息时，不需要经历先将信息传给服务端处理，然后处理结果再通过网络传回的过程，而是直接可以被客户端的 JavaScript 程序所处理。

JavaScript 与 Java 之间的主要区别包括：

- 基于对象和面向对象。JavaScript 语言是基于对象和事件驱动的脚本语言；Java 语言是面向对象的语言，可以设计独立的程序。
- 解释执行和编译执行 JavaScript 是一种解释性的语言，是将文本格式的代码发送到客户端由浏览器负责解释执行。Java 代码在执行之前必须编译成字节码文件，由虚拟机负责执行。
- 弱类型和强类型。JavaScript 是弱类型变量，即变量在使用前不需要声明，由解释器负责确定变量的数据类型，而 Java 是一种强类型语言，所有的变量在事情之前必须声明，并且只能存放声明类型的数据。
- 动态联编和静态联编。JavaScript 采用动态联编，即对象的引用在运行时进行检查，如不运行就无法实现对对象引用的检查。而 Java 采用静态联编，即 Java 对象引用在编译时进行，以使得编译器能够实现强类型检查。
- 代码格式不一样，Java 是一种与 HTML 无关的语言，而 JavaScript 可以嵌在 HTML 页面中发挥作用。

通常，在页面中嵌入 JavaScript 代码有两种方式：

- 直接嵌入到 HTML 文件中

这是最常用的方法，大部分含有 Javascript 代码的页面都采用这种方法，如上面的例子。将 JavaScript 代码直接写在＜script＞＜/script＞标签中。

- 引用外部文件方式

为了提高程序代码的可重用性，可以将一些常用功能实现代码写在一个单独的 JavaScript 源文件中(扩展名为.js)，在页面中使用＜script＞标签将该文件引入进来即可。其基本格式如下：

```
＜script language="JavaScript" type="text/javascript" src="外部 js 文件 url 地址" ＞＜/script＞
```

其中，language 属性代表脚本语言的类型，可以为 javaScript 或 liveScript；type 属性代

表脚本的 MIME 类型，MIME 类型由两部分组成：媒介类型和子类型。对于 JavaScript，其 MIME 类型是 "text/javascript"；src 属性代表 JavaScript 源文件的 url 地址。

一般的，推荐在 HTML 页面中的<head>标签中嵌入 JavaScript 代码。

在 JavaScript 中的注释符号和 Java 中基本一致，也分为单行和多行注释。注释后的信息仅仅是为了说明程序代码的功能，在程序的解释和运行中是被忽略的。

注释符号分为以下两种：

- 单行注释：使用“//”符号对单行信息进行注释。
- 多行注释：使用“/ * * /”符号对多行信息进行注释。

本节介绍 JavaScript 的基本语法。主要从 JavaScript 中的数据类型、变量声明及类型转换、表达式和运算符、数组的使用等几个方面来介绍 JavaScript 的基本语法规则。

5.1.2 JavaScript 基本语法

1. 常用数据类型

JavaScript 中支持的数据类型有：

- 字符串：例如，“218”“hello”“客户端”。
- 数值：包括整数数字和浮点型数字。
- 布尔值：true 和 false。
- 对象：object。
- 空值：null。
- 未定义值：undefined。
- 特殊字符：又叫转义字符，主要有以下几种，如表 5.1 所示。

表 5.1　　特殊字符

特殊字符	含义
\b	表示退格
\n	表示换页
\t	表示 Tab 符号
\r	表示回车符
\"	表示双引号本身
\'	表示单引号本身
\\	表示反斜线
\b	表示退格

2. 变量声明

在 JavaScript 中，变量的类型采用弱类型的方式，即声明变量时不需要严格指明变量的数据类型，所有变量的声明均使用“var”关键字。当为变量赋值时，会根据赋给变量的值的类型确定变量的数据类型。

例如：`var varName="Hello JavaScript";`

在上面的例子中，变量 varName 的数据类型就是字符串类型。

变量命名时，需要遵守以下几个规则：

● 变量命名必须以一个英文字母或是下划线为开头，也就是说，变量名第一个字符必须是 A 到 Z 或是 a 到 z 之间的字母或是“_”。

● 变量名长度在 0～255 字符之间。

● 除了首字符，其他字符可以使用任何字符、数字及下划线，但是不可以使用空格。

● 不可以使用 JavaScript 的运算符号，例如：+、-、*、/等。

● 不可以使用 JavaScript 用到的保留字，例如：var，function 等。

● 在 JavaScript 中，变量名中的大小写字母是有所区别的，例如：变量 myVar 和 myvar 是不同的两个变量。

JavaScript 变量的作用域包括两种：

● 全局变量，定义在函数体之外，作用范围是所有函数。

● 局部变量，定义在函数体之内，作用范围是本函数。

3. 类型转换

JavaScript 中提供了显式的将值从一种数据类型转换为另一种数据类型的转换函数。基本数据类型转换的三种函数有：

● 转换为字符串类型：String()

例如：String(2012)的结果为字符串"2012"

● 转换为数值类型：Number()

例如：Number("2012")的结果为数值 2012

● 转换为布尔类型：Boolean()

例如：Boolean(false)的结果为布尔值 false

另外，在 ECMAScript v3 和 JavaScript 1.5 中，增加了三种将数字转换成字符串的方法：

● toFixed()：把数字转换成字符串，并显示小数点后指定的位数。

● toExponential()：用指数计数法把数字转换成字符串，该字符串中的小数点前有一位数字，小数点后有指定位数的数字。

● toPrecision()：用指定位数的有效数字显示数字，如果有效数字的位数不足以显示数字的整数部分，它将采用指数计数法显示数字。

JavaScript 也提供了更灵活的字符串到数值的转换函数：

● parseInt()：将字符串转换为整数，并忽略其后所有非数字后缀。对于以 0x 和 0X 开头的字符串，parseInt()将它解释为十六进制数。parseInt()还可以具有第二个参数，用来指定要被解析的数的基数。其合法值在 2 到 36 之间。

● parseFloat()：将字符串转换为浮点数，并忽略其后所有非数字后缀。

需要注意，如果 parseInt()和 parseFloat()不能将指定的字符串转换成数字，将返回 NaN。

4. 运算符

JavaScript 中支持的运算符主要有以下几种：

- 算术运算符：+、-、*、/、%、++、--
- 比较运算符：<、>、<=、>=、==、!=、===(严格等于)、!==(严格不等于)
- 逻辑运算符：&&、||、!
- 赋值运算符：=、+=、-=、*=、/=、%=、&=、|=、^=、<<=、>>=、>>>=
- 条件选择符：条件表达式? A:B

例如：(time>=12)? "上午":"下午";

5. 表达式

JavaScript 中的表达式是变量、常量、布尔以及运算符的集合，可以对变量进行赋值、改变、计算等一系列操作。

表达式可以分为：

- 算术表达式
- 字符串表达式
- 赋值表达式
- 布尔表达式

6. 流程控制语句

JavaScript 中提供的流程控制语句可以分为以下两种：

- 条件和分支语句：if…else 语句、switch 语句。

(1)if…else 语句，是最基本、最简单的条件语句。其语法格式如下：

```
if(条件 1)
    语句块 1;
else if(条件 2)
    语句块 2;
    …
else 语句块 n;
```

(2)switch 语句，用于对一个表达式进行多次判断，每一种的取值都采取不同的处理方法。其语法格式如下：

```
switch (表达式){
    case label 1:语句块 1;break;
    case label 2:语句块 2;break;
    …
    case label n:语句块 n;break;
    [default:语句块 n+1;]
}
```

- 循环语句：for 语句、while 语句、do…while 语句、break 语句和 continue 语句。

(1)for 语句，用于反复地执行一段程序，并且在每次循环后处理变量，直到循环条件表达式计算结果为假。其语法格式如下：

```
for(循环变量初始化语句；循环条件表达式；循环变量更新语句){
    循环语句块；
}
```

(2)while 语句,用于在循环条件成立时一直循环执行一些语句块,直到条件不成立为止,其循环次数不固定。while 语句在执行循环体语句块之前先检查循环条件,若条件不满足则一次也不执行,直接退出。其语法格式如下:

```
while(循环条件表达式){
    循环语句块;
}
```

(3)do…while 语句,与 while 语句类似,区别在于 do...while 语句在执行循环体语句块之前不会先检查循环条件,即使条件不满足,循环体语句块也会至少执行一次。其语法格式如下:

```
do{
    循环语句块;
}while(循环条件表达式)
```

(4)break 语句

break 语句用于无条件的跳出 switch 语句或者循环结构。

(5)continue 语句

continue 语句用于结束本轮循环,直接进行下一轮循环。

(6)return 语句

return 语句用于将函数的处理结果返回给调用函数。

5.1.3　项目:第一个 JavaScript 程序

1. 项目说明

编写第一个 JavaScript 程序,要求利用 JavaScript 语言,在页面中输出“Hello World”信息,如图 5.1 所示。

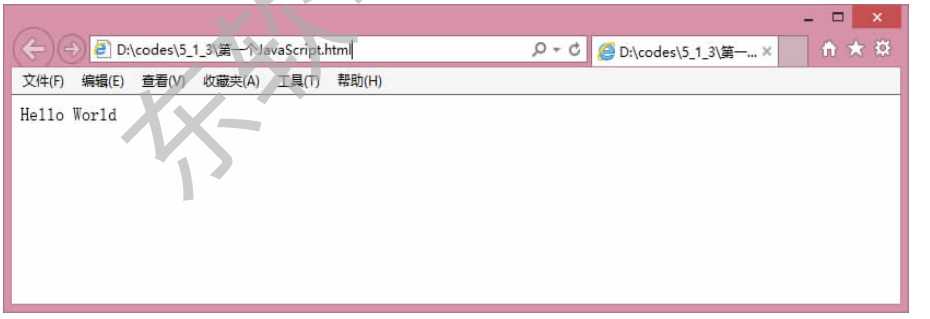

图 5.1　第一个 JavaScript 程序

2. 项目设计

本项目功能十分简单,可采用在 HTML 页面中直接嵌入 JavaScript 代码的方式来实现。JavaScript 是一种客户端脚本语言,目前所有主流浏览器都内嵌了 JavaScript 引擎来解释执行 JavaScript 代码,因此,在 HTML 页面中均可以直接嵌入 JavaScript 代码来为页面添加一些动态效果和交互功能。

使用 JavaScript 可以直接访问和操作 HTML 元素,例如使用 JavaScript 的 document 对象的方法可以直接访问和操作 HTML DOM 元素。本项目就可以利用 document 对象的方法来完成向页面输出信息的功能。

在本项目中，编写我们的第一个 JavaScript 程序。

(1)首先，在 HTML 页面中嵌入 JavaScript 代码，通常会写在＜head＞＜/head＞标签之间。JavaScript 代码必须用＜script＞＜/script＞标签括起。＜script＞标签中的 language 属性用来指明嵌入的脚本代码是使用哪种语言编写的，该属性可以省略，其默认值为“JavaScript”。嵌入的 JavaScript 代码不需要传给服务器处理，可以通过浏览器直接运行。

(2)接下来，在＜script＞＜/script＞编写一条 JavaScript 语句，这里使用了 JavaScript 中的 document 对象的 write()方法直接向页面输出信息，关于 document 对象的具体含义和常用方法的使用，会在 5.6 节中做详细介绍。

3. 项目实施

本项目代码如下：

```
<html>
<head>
    <script language="JavaScript">
        document.write("Hello World");
    </script>
</head>
<body>
</body>
</html>
```

将以上 HTML 文件保存为“第一个 JavaScript.html”，使用浏览器打开，会看到在页面上，输出了“Hello World”。

4. 知识运用

修改以上 JavaScript 代码，使其在页面上输出以下内容，效果如图 5.2 所示。

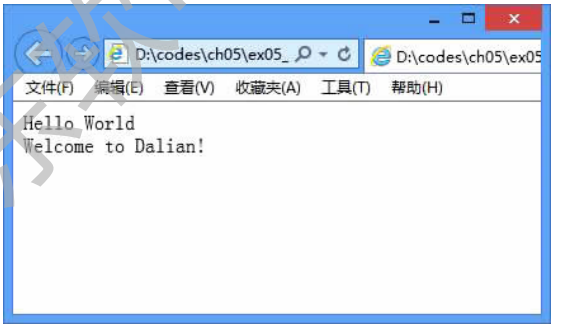

图 5.2 使用 JavaScript 输出信息

5.2 JavaScript 函数

5.2.1 JavaScript 函数定义

1. 函数定义

本节介绍如何定义 JavaScript 中的函数。

在 JavaScript 中，函数可以简单理解为一组语句，用来完成一系列工作。JavaScript 函数可以封装那些在程序中可能需要多次使用的模块，并可以作为事件驱动处理程序。

使用函数前一定要先进行定义，函数定义分为三个部分：函数名、参数列表和函数体。

定义函数的语法格式如下：

```
function 函数名(参数 1, 参数 2,..., 参数 N)
{
    函数体(语句集)
}
```

使用函数时需要注意以下几点：

(1)函数名是调用函数时引用的名称，它是大小写敏感的；

(2)参数表示传递给函数使用或操作的值，它可以是常量，也可以是变量，在函数内部可以通过 arguments 对象访问所有参数；

(3)函数体中的 return 语句用于返回函数的返回值，函数也可以不返回任何值；

(4)函数在定义时，其中的代码并不会被执行。只有当函数被调用时，其中的代码才会真正被执行。

2. 函数参数

调用函数时，变量、常量都可以作为参数传递。参数的传递是以传值的方式进行的。

例如：

```
hello("jason");
var user="jason"; hello(user);
```

也可以在定义函数时不指定使用的参数，JavaScript 会在每次调用该函数时自动生成 arguments 数组，并建立与参数列表有关的属性。

- Functionname.arguments，是一个数组，每一个参数对应其中的一个元素，可以使用该数组来访问调用时所传送的参数。
- Functionname.arguments.length，是一个整型变量，表示传递参数的个数。
- 可以使用这两个属性产生参数个数可变的函数。

3. 函数返回值

函数有时候需要有返回值，可以使用 return 语句，将需要返回的值放在 return 后面，可以是常量、变量以及有唯一确定值的有效表达式。

4. 函数中的变量

在 JavaScript 函数内部声明的变量称为局部变量，只能在函数内部访问它。可以在不同的函数中声明和使用名称相同的局部变量。JavaScript 变量的生命周期从它们被声明时开始，只要函数运行完毕，局部变量就会被删除。

在函数外声明的变量是全局变量，网页上的所有脚本和函数都能访问它。全局变量会在页面关闭后被删除。

5.2.2　项目：点击我

1. 项目说明

完成简单的 JavaScript 按钮点击程序，并弹出对话框。在页面上放置两个按钮，按钮上

的文字均为“点击我”，效果如图 5.3 所示。

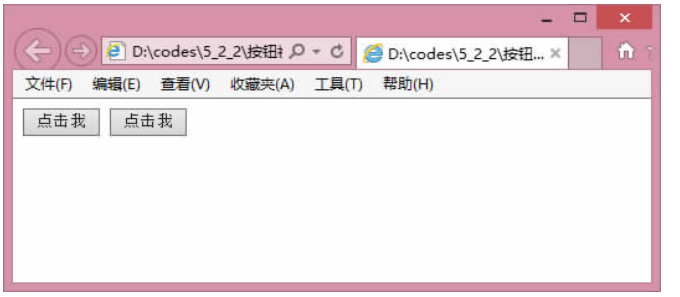

图 5.3 页面上放置按钮

当点击第一个按钮时，弹出对话框，提示信息“this 按钮被点击”；点击第二个按钮时，弹出对话框，提示信息“that 按钮被点击”。效果如图 5.4 所示。

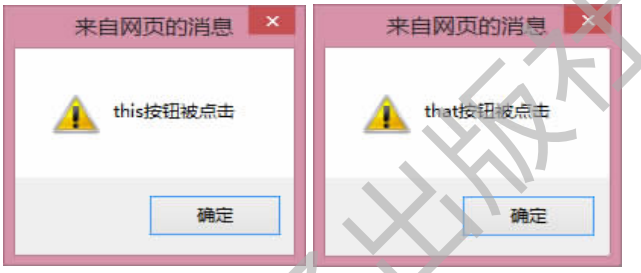

图 5.4 按钮被点击后弹出对话框

2. 项目设计

本项目是一个简单的 JavaScript 的函数定义和事件响应功能的实现。JavaScript 中的函数可以定义一组可重复使用的代码块，并且可以由相应的事件来驱动和调用。在某个事件发生时会自动调用所指定的事件处理函数(比如当用户点击按钮时)。

本项目中，就是要定义一个当发生了鼠标对页面上的按钮点击事件之后，进行规定的响应处理操作的函数。因此，在本项目中，要使用 JavaScript 编写对鼠标单击事件的事件处理函数，在其中完成具体事件响应处理操作。

(1)首先，在 HTML 页面主体部分，定义了两个按钮，他们的 onclick 事件的事件处理函数均声明为 clickme()，并在调用时，传递了不同的实参。其中，第一个按钮调用 clickme()函数时传递的实参值为“this”，第二个按钮调用 clickme()函数时传递的实参值为“that”。

(2)接下来，需要在 HTML 页面<head>的标签内，编写 JavaScript 函数。这里使用了 function 关键字定义了一个名为 clickme()的函数，并且该函数带一个参数，名为 str。在函数体内，使用 alert()方法弹出一个对话框(关于 alert()方法的含义和使用在 5.5 节中会做详细介绍)，对话框中显示的提示信息为：参数 str 的值＋“按钮被点击”。

因此，点击第一个按钮后，会弹出对话框，显示提示信息“this 按钮被点击”，点击第二个按钮后，对话框的提示信息变为“that 按钮被点击”。

3. 项目实施

本项目代码如下：

```
<html>
<head>
<script language="JavaScript">
    /*定义函数*/
    function clickme(str) {
        alert(str+"按钮被点击");
    }
</script>
</head>
<body>
<input type="button" onclick="clickme('this') " value="点击我">
<input type="button" onclick="clickme('that') " value="点击我">
</body>
</html>
```

将以上 HTML 文件保存为“按钮被点击.html”，使用浏览器打开，分别点击两个按钮，观察弹出对话框的显示的信息。

4. 知识运用

在以上 HTML 页面的主体部分，为两个按钮添加 name 属性，取值分别为“button1”和“button2”，并将按钮的 name 属性做为实参调用 clickme()函数，在 clickme()函数中弹出对话框，对话框中显示提示信息：按钮 name 属性值＋“被点击”。按钮被点击后，弹出的对话框效果如图 5.5 所示。

图 5.5　按钮被点击后弹出对话框

5.3　事件和事件处理

5.3.1　JavaScript 常用事件

在 JavaScript 中，事件就是指 Web 页面在浏览器处于活动状态时发生的各种事情，也就是用户与 Web 页面交互时产生的各种操作。如：浏览器加载、卸载一个页面，用户单击鼠标、移动鼠标，以及按下键盘的某个键。

浏览器为了响应某个事件而进行的处理过程，叫做事件处理，这个处理过程叫做事件处理函数。对事件的处理，一般都是通过调用相应的事件处理函数去完成的。事件调用函数的格式通常为：

on 事件名＝事件处理函数

常用的 JavaScript 事件如表 5.2 所示。

表 5.2　　常用 JavaScript 事件

事件调用函数	何时触发	支持的页面元素
onClick	鼠标单击时	所有元素
onDbClick	鼠标双击时	所有元素
onChange	域的内容改变时	＜input type＝"text"＞，＜input type＝"radio"＞，＜input type＝"checkbox"＞，＜select＞，＜textarea＞
onFocus	窗口或元素获得焦点时	所有元素
onBlur	窗口或元素失去焦点时	所有元素
onSelect	文本被选中时	＜input type＝"text"＞，＜textarea＞
onMouseDown	鼠标上的按钮被按下时	所有元素
onMouseOver	鼠标移动到某范围内时	所有元素
onMouseOut	鼠标离开某范围内时	所有元素
onMouseMove	鼠标移动时	所有元素
onMouseEnter	鼠标进入到某范围内时	所有元素
onMouseLeave	鼠标离开某范围内时	所有元素
onKeyDown	某个键盘按键被按下	所有元素
onKeyPress	某个键盘按键被按下并松开	所有元素
onKeyUp	某个键盘按键被松开	所有元素
onLoad	文档、图像或对象全部加载完毕时(全部加载完毕意味着不但 HTML 文件，而且包含的图片、插件、空间、小程序等全部加载完毕)	＜body＞，＜frame＞，＜frameset＞，＜iframe＞，＜img＞，＜link＞，＜script＞，＜object＞
onUnload	文档卸载时(或者关闭窗口、或者到另一个页面去时)	＜body＞，＜frameset＞
onSubmit	表单提交时	＜form＞
onReset	表单重置按钮被点击时	＜form＞
onResize	当窗口被调整大小时	所有元素
onError	发生错误时，其事件处理程序通常叫做“错误处理程序(Error Handler)”，用来处理错误	所有元素

需要注意，onMouseEnter 事件类似于 onMouseOver 事件，唯一的区别 onMouseEnter 事件不支持冒泡，也就是说，当鼠标指针进入某元素时会触发 onMouseOver 事件和

onMouseEnter 事件，并且在这个元素的所有子元素上也会触发 onMouseOver 事件，但不会触发子元素的 onMouseEnter 事件。对于 onMouseLeave 和 onMouseOut 事件的区别和以上类似，不再赘述。

5.3.2　项目：敏感的兔子

1. 项目说明

完成 JavaScript 的事件声明和事件处理程序。在页面上放置图片，并利用 Div＋CSS 在图片上划分出四个区域，要求：

(1)左上角区域响应鼠标进入事件，当鼠标进入该区域时，弹出对话框，显示“离我远点儿”；

(2)右上角区域响应鼠标悬浮事件，当鼠标在区域上方悬浮时，弹出对话框，显示“不要摸我！”；

(3)左下角区域响应鼠标单击事件，当鼠标在区域中单击时，弹出对话框，显示“谁打我？”；

(4)右下角区域响应鼠标双击事件，当鼠标在区域中双击时，弹出对话框，显示“谁打我两下？”。

另外，在该页面主体部分，响应键盘按键事件，当有键按下时，弹出对话框，显示该键的键盘码。

运行效果如图 5.6 所示。

图 5.6　鼠标进入左上角区域时弹出对话

2. 项目设计

本项目要完成一个简单的 JavaScript 的事件响应和处理功能。当用户与 Web 页面进行某种行为的交互时，就会发生事件。事件可能是鼠标在某些 HTML 元素上的点击、鼠标经过某个元素或按下键盘上的某些按键等等。事件还可能是浏览器中发生的事情，比如某个 Web 页面加载完成，或者是用户改变窗口大小等等。通过使用 JavaScript，可以监听某种

特定事件的发生，并规定在某些事件发生后对这些事件做出响应。

本项目中，就是要监听鼠标对页面上的指定元素的鼠标进入、悬浮、单击、双击和键盘事件，并在发生某个特定事件后，进行规定的响应处理操作。因此，在本项目中，要实现的是一个完整的事件响应处理流程。在事件源上，指定鼠标进入、悬浮、单击、双击和键盘事件的事件处理函数，并使用 JavaScript 编写这些事件处理函数，在其中完成具体的事件响应处理操作。

在本项目中利用 JavaScript 的常用事件如 onMouseEnter、onMouseOver、onclick、ondblclick 和 onKeyPress 等事件，编写其事件处理函数，来完成不同的用户行为响应。

(1)首先，为左上角区域的 标签添加 onMouseEnter 事件并指定事件处理代码是“alert('离我远点儿')”；

(2)为右上角区域的 标签添加 onMouseOver 事件并指定事件处理代码是“alert('不要摸我！')”；

(3)为左下角区域的 标签添加 onclick 事件并指定事件处理代码是“alert('谁打我？')”；

(4)为右下角区域的 标签添加 ondblclick 事件并指定事件处理代码是“alert('谁打我两下？')”；

(5)为页面主体<body>标签添加 onKeyPress 事件并指定事件处理函数是 fun()，传递了 event 对象做为参数。(event 对象是 DOM 对象中用于代表事件的状态的对象，会在 5.6 节做详细介绍)

3. 项目实施

本项目代码如下：

```
<html>
<head>
<style>
    #main{margin:0 auto; width:400px;height:400px;position:relative;}
    .box{width:180px;height:180px;opacity:0.2;background:#99FF00;position:absolute;}
    #box1{left:60px;top:30px;}
    #box2{left:250px;top:30px;}
    #box3{left:60px;top:220px;}
    #box4{left:250px;top:220px;}
</style>
<script>
    function fun(e){
        alert(e.keyCode);
    }
</script>
</head>
<body onkeypress="fun(event)">
<div id="main">
    <img src="../images/IMG_2755.JPG" border="0" alt="" >
    <div id="box1" class="box" onmouseenter="alert('离我远点儿')" >鼠标进入</div>
    <div id="box2" class="box" onmouseover="alert('不要摸我！')">鼠标悬浮</div>
```

```
    <div id="box3" class="box" onclick="alert('谁打我？')">鼠标单击</div>
    <div id="box4" class="box" ondblclick="alert('谁打我两下？')">鼠标双击</div>
</div>
</body>
</html>
```

将以上 HTML 文件保存为“敏感的兔子.html”，使用浏览器打开，分别使用鼠标在不同区域移动或点击，观察弹出的对话框显示的信息。按下键盘上的按键，观察弹出对话框显示的信息。

4. 知识运用

在以上 HTML 页面的主体部分，为图片标签<img>添加图像加载完毕事件和事件处理函数，当图片加载完毕时，弹出对话框，显示“我来了”。在图片下方，添加一个单行文本框，为文本框添加文本内容改变事件和事件处理函数，当文本框中的内容发生改变并失去焦点时，弹出对话框，显示“你写了什么?”。效果如图 5.7 所示。

图 5.7　图片加载完毕时弹出对话框以及文本框内容改变时弹出对话框

5.4　内置对象

5.4.1　认识 JavaScript 内置对象

JavaScript 的一个重要特点就是它是一种面向对象的语言。通过基于对象的程序设计，可以用更直观、模块化和可重复使用的方式进行程序开发。

一组包含数据的属性和对属性中包含的数据进行操作的方法，称为对象。比如要设定网页的背景颜色，所针对的对象就是 document，所用的属性名是 bgcolor。例如，document.bgcolor="blue"，就是设置背景的颜色为蓝色。

JavaScript 中支持的对象主要有以下几种：

1. 内置对象

JavaScript 将一些非常常用的功能预先定义成对象，用户可以直接使用，这种对象就是内置对象。

2. 浏览器对象

- 网页和浏览器本身的各种元素在 JavaScript 程序中的体现。
- 它使 JavaScript 可以定位、改变内容以及展示 HTML 页面的所有元素。

3. DOM 对象

HTML DOM 对象定义了用于 HTML 的一系列标准的对象，以及访问和处理 HTML 文档的标准方法。

4. 自定义对象

JavaScript 允许用户自定义对象进行使用。在本节主要介绍常用的内置对象。

在 JavaScript 中，内置对象都有自己的方法和属性，访问其属性的语法是："对象名.属性名"。访问其方法的语法是："对象名.方法名称(参数列表)"。

常用的内置对象如表 5.3 所示：

表 5.3　　常用内置对象

对象名	描述
Math	数学对象，提供了进行所有基本数学计算的功能和常量的属性和方法
Date	日期对象，提供了获取、设置日期和时间的属性和方法
String	字符串对象，提供了对字符串进行处理的属性和方法
Array	数组对象，用来描述数组并提供数组出理的属性和方法

5.4.2 Math 对象

内置的 Math 对象可以用来处理各种数学运算，其中定义了一些常用的数学常数和运算方法。Math 的属性和方法可以直接调用，其语法格式为：

- `Math.属性名`
- `Math.函数名(参数列表)`

Math 对象的常用属性和方法如表 5.4 所示：

表 5.4　　Math 对象的常用属性和方法

	属性名/方法名	描述
常用属性	LN2	2 的自然对数(约 0.693)
	LN10	10 的自然对数(约 2.302)
	LOG2E	以 2 为底的 e 的对数(约 1.442)
	LOG10E	以 10 为底的 e 的对数(约 0.434)
	PI	数学 PI 值 3.1415926
	SQRT1_2	0.5 的平方根(约 0.707)
	SQRT2	2 的平方根(约 1.414)

（续表）

	属性名/方法名	描述
常用方法	abs(x)	返回数字 x 的绝对值
	acos(x)	返回数字 x 的反余弦值
	asin(x)	返回数字 x 的反正弦值
	atan(x)	返回位于－PI/2 和 PI/2 的反正切值
	atan2(y,x)	返回(x,y)位于 －PI 到 PI 之间的角度
	ceil(x)	返回 x 四舍五入后的最大整数
	cos(x)	返回数字 x 的余弦值
	exp(x)	返回 E^x 值
	floor(x)	返回 x 的截尾取整结果
	log(x)	返回底数为 E 的自然对数
	max(x,y)	返回 x 和 y 之间较大的数
	min(x,y)	返回 x 和 y 之间较小的数
	pow(x,y)	返回 y^x 的值
	random()	返回位于 0 到 1 之间的随机函数
	round(x)	四舍五入后取整
	sin(x)	返回数字 x 的正弦值
	sqrt(x)	返回数字 x 的平方根
	tan(x)	返回数字 x 的正切值
	valueOf()	返回数学对象的原始值

5.4.3　Date 对象

Date 对象用来对日期和时间进行操作，它的大多数方法需要利用对象来调用，因此必须先声明和创建 Date 对象。其语法格式如下：

```
var Date 对象名＝new Date([日期参数]);
```

如果创建对象时没有提供日期参数，即 var today＝new Date()时，today 代表当前时间。

也可以在创建 Date 对象时提供日期参数，例如：

```
var someday＝new Date("February 26,2005 12:00:00"); //月 日,年 时:分:秒
var someday＝new Date(2009,4,7,0,0,0);          //年,月,日,时,分,秒
```

其中，年、月、日均为必填参数，而时、分、秒为可选参数。

Date 对象主要提供了以下 3 类方法：

1. 从系统中获得当前的时间和日期；

2. 设置当前的日期和时间；

3. 在时间、日期同字符串之间完成转换。

Date 对象没有提供可以直接访问的属性，只有获取和设置日期和时间的方法。Date 对象的常用方法如表 5.5 所示：

表 5.5　　Date 对象的常用方法

方法名	描述
getDay()	返回一周中的第几天(0～6)
getYear()	返回当前年份。2000 年以前为 2 位，2000(包含)以后为 4 位
getFullYear()	返回完整的 4 位年份数
getMonth()	返回当前月份(0～11)
getDate()	返回当前日(1～31)
getHours()	返回当前小时数(0～23)
getMinutes()	返回当前分钟数(0～59)
getSeconds()	返回当前秒数(0～59)
getMilliseconds()	返回当前毫秒数(0～999)
getUTCDay()	依据国际时间得到现在是星期几(0～6)
getUTCFullYear()	依据国际时间来得到完整的年份
getUTCMonth()	依据国际时间来得到当前月份(0～11)
getUTCDate()	依据国际时间来得到当前日(1～31)
getUTCHours()	依据国际时间来得到当前小时数(0～23)
getUTCMinutes()	依据国际时间来返回当前分钟数(0～59)
getUTCSeconds()	依据国际时间来返回秒数(0～59)
getUTCMilliseconds()	依据国际时间来返回毫秒数(0～999)
getTime()	返回从 1970 年 1 月 1 号 0:0:0 到现在一共花去的毫秒数
getTimezoneoffset()	返回时区偏差值，即格林威治平均时间(GMT)与运行脚本的计算机所处时区设置之间相差的分钟数)
parse(dateString)	返回在 Date 字符串中自从 1970 年 1 月 1 日 00:00:00 以来的毫秒数(parse 方法是 Date 对象的静态方法，可以通 Date 类直接调用，而不是使用一个 Date 类的对象来调用)。其中，短日期可以使用“/”或“-”作为日期分隔符，但是必须按“月/日/年”的顺序来表示。以“月名称 日 年”的形式来表示长日期，例如“July 08 2008”，年份值可以用 2 位也可以用 4 位数字来表示。如果使用 2 位数字来表示年份，那么该年份必须大于等于 70。小时、分钟、秒钟之间用冒号分隔，这三项不是必须都指明，例如，“12:”“12:10”“12:10:11”都是有效的
setYear(yearInt)	设置年份。2 位数或 4 位数
setFullYear(yearInt)	设置年份。4 位数

（续表）

方法名	描述
setMonth(monthInt)	设置月份(0～11)
setDate(dateInt)	设置日(1～31)
setHours(hourInt)	设置小时数(0～23)
setMinutes(minInt)	设置分钟数(0～59)
setSeconds(secInt)	设置秒数(0～59)
setMilliseconds(milliInt)	设置毫秒数(0～999)
setUTCFullYear(yearInt)	依据国际时间来设置年份
setUTCMonth(monthInt)	依据国际时间来设置月(0～11)
setUTCDate(dateInt)	依据国际时间来设置日(1～31)
setUTCHours(hourInt)	依据国际时间来设置小时数
setUTCMinutes(minInt)	依据国际时间来设置分钟数
setUTCSeconds(secInt)	依据国际时间来设置秒数
setUTCMilliseconds(milliInt)	依据国际时间来设置毫秒数
setTime(timeInt)	设置从 1970 年 1 月 1 日开始的时间.毫秒数
toGMTString()	根据格林威治时间将 Date 对象的日期(一个数值)转变成一个 GMT 时间字符串,如:Weds,15 June 1997 14:02:02 GMT
toUTCString()	根据通用时间将一个 Date 对象的日期转换为一个字符串
toLocaleString()	把 Date 对象的日期(一个数值)转变成一个字符串,使用所在计算机上配置使用的特定日期格式
toString()	将日期对象转换为字符串
UTC(yyyy, mm, dd, hh, mm, ss, msec)	返回从格林威治标准时间到指定时间的差距,单位为毫秒
valueOf()	返回日期对象的原始值

5.4.4　String 对象

String 对象提供了对字符串进行操作的各种方法以及一些属性。字符串变量的初始化通常有以下两种方式：

1.声明字符串变量时直接为其赋值。例如：

```
var str="Hello World";
```

2.使用 new 关键字创建字符串对象并在构造函数中提供初始化参数。例如：

```
var str=new String("Hello World");
```

String 对象的常用属性和方法如表 5.6 所示。

表 5.6 String 对象的常用属性和方法

	属性名/方法名	描述
常用属性	length	返回字符串的字符长度
常用方法	anchor()	创建书签链接，相当于<A name=..>
	big()	把字符串中的文本变成大字体(<BIG>)
	blink()	把字符串中的文本变成闪烁字体(<BLINK>)
	bold()	把字符串中的文本变成黑字体(<B>)
	fixed()	把字符串中的文本变成固定间距字体，即电报形式(<TT>)
	fontcolor(color)	设置字符串中文本的颜色(<FONT COLOR=>)
	fontsize(size)	把字符串中的文本变成指定大小(<FONTSIZE=>)
	italics()	把字符串中的文本变成斜字体(<I>)
	link(url)	创建超链接，相当于<a href=..>
	small()	把字符串中的文本变成小字体(<SMALL>)
	strike()	把字符串中的文本变成划掉字体(<STRIKE>)
	sub()	把字符串中的文本变成下标(subscript)字体(<SUB>)
	sup()	把字符串中的文本变成上标(superscript)字体(<SUP>)
	charAt(index)	返回指定索引处的字符
	charCodeAt(index)	返回一个整数，该整数表示 String 对象中指定位置处的字符的 Unicode 编码
	concat(newString1,...,newStringN)	连接两个或多个字符串
	indexOf(searchString)	返回字符串中第一个出现指定字符串 searchString 的位置
	lastIndexOf(searchString)	返回字符串中最后一个出现指定字符串 searchString 的位置
	replace(regex, newString)	将字符串中的某些字符 regex 替换成其他字符 newString
	slice(startIndex, endIndex)	将部分字符抽出并在新的字符串中返回剩余部分
	split(delimiter)	将字符串分配为数组
	substr (startIndex, length)	从 startIndex 位置取子串，取 length 个字符
	substring(startIndex, endIndex)	从 startIndex 和 endIndex 之间的字符，不包括 endIndex 位置的字符
	toLowerCase()	把字符串中的所有字符变成小写
	toUpperCase()	把字符串中的所有字符变成大写
	valueOf()	返回字符串对象的原始值

5.4.5　Array 对象

在 JavaScript 中使用 new 和 Array 关键字来创建数组对象。创建数组的语法有以下几种：

1. 创建一个数组。

```
var 数组对象名＝ new Array();
```

2. 创建一个数组并指定长度。

```
var 数组对象名＝ new Array(size);
```

3. 创建一个数组并赋初值。

```
var 数组对象名＝ new Array(element0, element1,..., elementN);
```

4. 创建一个数组并赋初值。

```
var 数组对象名＝[element0, element1,..., element];
```

需要注意的是，虽然第二种方法在创建数组时指定了长度，但实际上所有情况下数组都是可变长的，也就是说即使指定了长度为 3，仍然可以将元素存储在规定长度以外的位置，此时数组长度也会随之改变。

数组对象的常用属性和方法如表 5.7 所示：

表 5.7　　Array 对象的常用属性和方法

	属性名/方法名	描述
常用属性	length	返回数组的长度
常用方法	concat(array1,...,arrayN)	将两个或多个数组值连接起来，合并后返回结果
	join(delimiter)	将数组中元素合并为字符串，参数为分隔符，如果省略参数则直接合并，不再分隔
	pop()	移除数组中的最后一个元素并返回该元素
	push(value)	在数组的末尾加上一个或多个元素，并且返回新的数组长度值
	reverse()	颠倒数组中元素的顺序，反向排列
	shift()	移除数组中的第一个元素并返回该元素
	slice(startIndex,endIndex)	复制数组中的一个连续部分，返回复制的新数组。slice()方法不会改变原来的数组
	slice(startIndex,sliceCount)	从数组中分离一个子数组
	sort(compare Function)	在未指定排序号的情况下，按照元素的字母顺序排列，如果不是字符串类型则转换成字符串再排序，返回排序后的数组
	splice(startIndex,delCount)	从数组中移除元素，返回移除的数组元素。splice()方法会改变原来的数组
	toString()	将数组所有元素返回一个字符串，其间用逗号分隔
	unshift(value)	为数组的开始部分加上一个或多个元素，并且返回该数组的新长度
	valueOf()	返回数组对象的原始值

5.4.6 项目:数字电子时钟

1. 项目说明

在 HTML 页面上显示一个数字电子时钟,要求在页面上实现数字电子时钟动态显示时间的效果。数字电子时钟效果如图 5.8 所示。

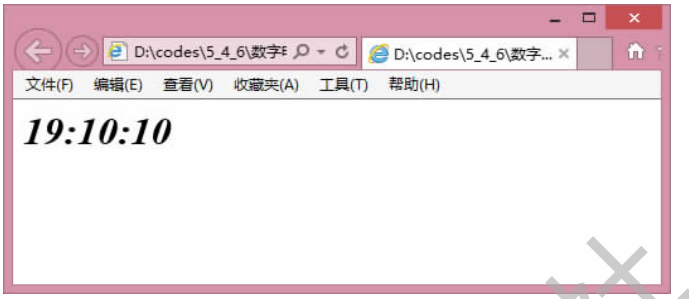

图 5.8 数字电子时钟

2. 项目设计

本项目需要使用 JavaScript 的内置对象 Date 来完成读取系统当前时间的功能。实际上,所有编程语言都具有内置的对象来创建语言的基本功能,内置对象是编写自定义代码的基础。JavaScript 提供了多种内置对象来完成对字符串、日期、数值、数组等数据类型的封装和操作。本项目就可以使用内置对象 Date 来完成对日期数据的访问和操作。

另外,要实现电子时钟效果,需要定时的周期性的调用用于获取并输出当前时间的函数。这里可以利用 window 对象的方法来完成。JavaScript 的 BOM(浏览器对象模型)对象提供了一整套与浏览器进行交互的方法和接口,window 对象就是 BOM 对象之一,它代表浏览器窗口的一个实例,提供了许多方法可以用于操作浏览器窗口,例如打开或关闭窗口、获取和调整窗口大小、获取窗口位置、移动窗口以及实现间歇定时调用、超时调用等功能。因此,本项目可以使用 JavaScript 的 window 对象来完成周期性调用函数的功能。

本项目中利用 JavaScript 的内置对象 Date 的相关函数完成时钟计时功能,首先获取系统当前时间,然后每隔一秒刷新一次时间显示区域。并且利用了 JavaScript 的 BOM 对象中的 window 对象的 setInterval()完成周期性调用函数的功能。

(1)首先,在 HTML 主体部分,在<script></script>标签中,调用了 JavaScript 中的 setInterval()方法。其效果是,每隔 1 秒,调用一次自定义的 JavaScript 方法 setTime()。在这里使用的 setInterval()方法是 JavaScript 的 BOM 对象中的 window 对象的方法,其作用是按照指定的周期(以毫秒计)来调用方法或计算表达式。(window 对象的常用属性和方法会在 5.5 节中做详细介绍)

(2)在 setTime()方法中,创建内置对象 Date 对象,使用它的 getHours()获取当前的小时数,使用 getMinutes()获取当前的分钟数,使用 getSeconds()获取当前的秒数。并使用 innerHTML 属性更新 id 为"timer"的标签的标签体内容。(innerHTML 属性是所有 DOM 元素对象所拥有的属性,其作用是设置或返回元素从起始标签到结束标签之间的 HTML,DOM 对象的常用属性和方法会在 5.6 节中做详细介绍)

3. 项目实施

本项目代码如下:

```
<html>
<head>
    <meta charset="utf-8">
    <script>
        function  setTime(){
            var  d=new Date();
            timer.innerHTML=" <i>" + d.getHours() + ":" + d.getMinutes() + ":" + d.
        getSeconds()+"</i>";
        }
    </script>
</head>
<body>
    <h1 id="timer">00:00:00</h1>
    <script>
        setInterval("setTime()",1000);
    </script>
</body>
</html>
```

将以上 HTML 文件保存为“数字电子时钟.html”，使用浏览器打开，观察页面上输出的当前时间信息。

4. 知识运用

(1)在 HTML 页面的主体部分，输出当前日期和星期信息，输出格式如下：

"今天是 XXXX 年 XX 月 XX 日，星期 X"

使用 JavaScript 的内置对象 Date 的方法获取当前日期和星期信息。

(2)创建一个长度为 2 的数组，其中分别存放两个随机数字，要求两个随机数字均为[1，100]区间内的整数。然后，以数组中的存放的两个元素为半径，分别计算对应的圆的面积，显示到 HTML 页面主体部分。

5.5　BOM 对象

5.5.1　认识 BOM 对象

BOM 是 browser object model 的缩写，简称浏览器对象模型。在 JavaScript 中，浏览器对象用于访问当前页面以及浏览器本身的信息。

BOM 提供了独立于内容而与浏览器窗口进行交互的对象，由于 BOM 主要用于管理窗口与窗口之间的通讯，因此其核心对象是 window。BOM 由一系列相关的对象构成，并且每个对象都提供了很多方法与属性。

常用的浏览器对象主要有以下几种：

1. 窗口对象(window)

window 对象处于对象层次的最顶端,它提供了处理浏览器窗口的方法和属性。

2. 位置对象(location)

location 对象提供了与当前打开的 URL 一起工作的方法和属性,它是一个静态的对象。

3. 历史对象(history)

history 对象提供了与历史清单有关的信息。

4. 文档对象(document)

document 对象包含了与文档元素(elements)一起工作的对象,它将这些元素封装起来供编程人员使用。可以使用 document 作为访问 HTML DOM 对象的入口。

5. 导航对象(navigator)

navigator 对象通常用于检测浏览器与操作系统的版本。

6. 屏幕对象(screen)

Screen 对象通常用于获取用户屏幕信息。

BOM 的层次结构图如图 5.9 所示。

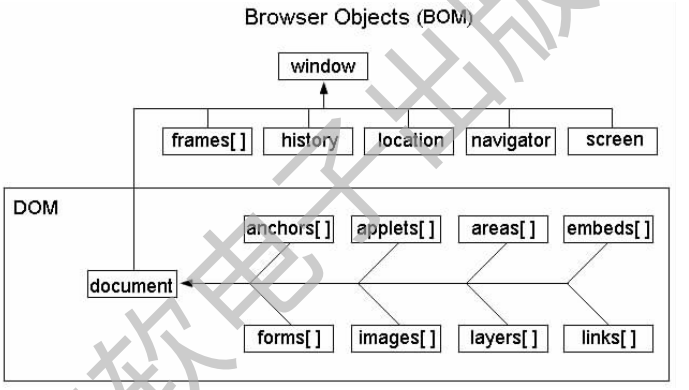

图 5.9 BOM 层次结构模型

浏览器对象的引用方式主要有以下几种:

1. 对应于文档对象模型中的层次关系,JavaScript 对浏览器对象的引用是逐层引用。

例如:在引用 forms 对象时,使用 window. document. forms。

2. 通过对象的 name 属性来引用。

例如:引用一个 name 属性是 form1 的表单对象,使用 window. document. form1。

3. 数组型浏览器对象的引用:在文档对象模型中,有些对象属于数组型对象,如 document 对象下一层的 images、links、forms 等对象,在引用这种数组对象时,可以使用对象在数组中的位置(下标)来引用。例如:window. document. forms[0],表示引用文档中的第一个表单。

另外,Window 对象作为文档对象模型中的最顶层对象,JavaScript 认为它是默认的,因此可以省略不写。如:window. document. forms 可以简写成 document. forms。

下面分别介绍几种常用浏览器对象的使用方法。

5.5.2　window 对象

window 对象也就是窗口对象，处于文档对象模型的最顶层，代表当前浏览器窗口。window 对象的常用属性和方法如表 5.8 所示：

表 5.8　　window 对象常用属性和方法

	属性名/方法名	描述
常用属性	name	窗口的名字
	closed	判断窗口是否已经被关闭，返回布尔值
	document，frames，history，location	四个下级对象
	length	窗口内的框架个数
	self	当前窗口
	top	当前框架的最顶层窗口
	status	状态栏的信息.
	scrollbars	浏览器的滚动条
	toolbar	浏览器的的工具栏
	menubar	浏览器的菜单栏
	locationbar	浏览器的地址栏
	innerHeight	浏览器窗口的内部高度
	innerWidth	浏览器窗口的内部宽度
常用方法	open(URL，[windowName[，windowFeature[，replace]]])	打开一个新窗口，返回值为该窗口对象的引用。其中，参数 URL 代表要打开窗口的 URL 地址；参数 windowName 是新打开窗口的名称；参数 windowFeature 是新窗口的实际特性（窗口的外观），包括 height（窗口高度）、width（窗口宽度）、top（窗口距离屏幕顶端的象素值）、left（窗口距离屏幕左端的象素值）、menubar（是否有菜单）、scrollbars（是否有滚动条）、resizable（窗口大小是否可改变）、toolbar（是否显示标准工具栏）、directories（是否显示目录按钮）、location（是否显示地址栏）、status（是否显示状态兰）、fullscreen（是否全屏显示）；参数 replace 是一个可选的布尔型参数，它规定了装载到窗口的 URL 是在窗口的浏览历史中创建一个新条目，还是替换浏览历史中的当前条目。true 代表替换浏览历史中的当前条目，false 代表在浏览历史中创建新的条目
	close()	关闭窗口
	moveBy(x，y)	x 代表水平位移，y 代表垂直位移。正值为窗口往右往下移动，负值相反

（续表）

	属性名/方法名	描述
常用方法	moveTo(x,y)	窗口左上角移到 x,y 坐标处
	resizeBy(x,y)	调整窗口大小,x 代表水平位移,y 代表垂直位移。正值代表往右往下调整
	resizeTo(w,h)	调整窗口大小为指定值,w 代表宽,h 代表高
	focus()	得到焦点
	blur()	失去焦点
	alert(text)	弹出警告对话框,在窗口中显示参数 text 指定的内容
	confirm(text)	弹出确认对话框,在窗口中显示参数 text 指定的内容
	promt(text,defaultText)	弹出带有文本输入框的提示对话框,在窗口中显示参数 text 指定的内容,并返回用户输入结果
	setInterval(expression,time)	按照指定的周期 time(以毫秒计)来调用函数或计算表达式 expression
	clearInterval(id)	取消由 setInterval()设置的周期性执行操作,参数 id 是由 setInterval()返回的 id 值
	setTimeout(expression,time)	按照指定的 time(以毫秒计)之后计算表达式 expression
	clearTimeout()	取消由 setTimeout()方法设置的延迟执行代码块,参数 id 是由 setTimeout ()返回的 id 值

5.5.3 history 对象

history 对象是 window 对象的一个属性,用来存储客户端最近访问过的网页清单。history 对象的常用属性和方法如表 5.9 所示:

表 5.9　　history 对象常用属性和方法

	属性名/方法名	描述
常用属性	length	存储在记录清单中的网页数目
	current	当前网页的地址
	next	下一个历史记录的网页地址
	previous	上一个历史记录的网页地址
常用方法	back()	后退到客户端查看过的上一页
	forward()	前进到客户端查看过的下一页
	go(整数或 URL 字符串)	当参数为整数时,表示前进或后退到已经访问过的某个页面。参数值为负数时代表后退到曾访问过的倒数第几个页面,等价于浏览器中的“后退”按钮;参数值为正数代表前进到曾访问过的第几个页面,等价于浏览器中的“前进”按钮。当参数为 URL 地址字符串时,表示前往该 URL 地址的网页

5.5.4　location 对象

location 对象提供了操作当前打开的窗口的 URL 的方法和属性。它的常用属性和方法如表 5.10 所示：

表 5.10　　location 对象常用属性和方法

	属性名/方法名	描述
常用属性	href	其值为当前页面的完整的 URL 地址字符串。也可以为其重新赋值，新值为导航到的新网页，其作用等价于＜a＞标签的功能
	protocol	当前 URL 中的通信协议
	host	当前 URL 中主机名
	hostname	当前 URL 中 host:port 部分
	port	当前 URL 中的端口号
	hash	当前 URL 中定位锚点名称
常用方法	reload()	刷新，重新加载当前的网页
	replace(URL)	用参数中的 URL 网址取代当前的网页
	assign(URL)	加载参数 URL 指定的新的文档

5.5.5　screen 对象

screen 对象包含有关用户屏幕的信息。它的常用属性如表 5.11 所示：

表 5.11　　screen 对象常用属性

	属性名/方法名	描述
常用属性	availWidth	可用的屏幕宽度
	availHeight	可用的屏幕高度

5.5.6　document 对象

document 对象的常用属性和方法在下一节 5.6 节中会单独介绍，在此不再赘述。

5.5.7　项目：打开新窗口

1. 项目说明

在 HTML 页面中放置两个按钮，分别为“创建窗口”和“关闭窗口”。利用 JavaScript 的 window 对象的方法，点击“创建窗口”按钮，打开一个新的浏览器窗口，里面显示上节完成的数字电子时钟页面。点击“关闭窗口”按钮，关闭当前的浏览器窗口。

2. 项目设计

本项目要实现的是操作浏览器窗口的功能，即打开新窗口和关闭当前窗口。可以使用 JavaScript BOM 中的 window 对象来完成。window 对象即是浏览器窗口的一个实例，提供了许多方法可以用于操作浏览器窗口，例如打开或关闭窗口、获取和调整窗口大小、获取窗

口位置、移动窗口等等。因此，本项目可以使用 JavaScript 的 window 对象来完成打开新窗口和关闭当前窗口的功能。

(1)首先，在 HTML 页面主体部分放置两个按钮，第一个按钮的 onclick 事件的事件处理函数为 createwindow()，第二个按钮的 onclick 事件的事件处理函数为 closewindow()。

(2)在 createwindow()函数中，使用浏览器对象中的顶层对象 window 的 open()方法，打开了一个新的浏览器窗口，其中显示“数字电子时钟. html”页面，新窗口名称设为“mywindow”，窗口设置成“无菜单栏，高 200，宽 300，不可改变大小”的状态。

(3)在 closewindow()函数中，使用 window 对象的 close()方法，关闭当前窗口。

3. 项目实施

本项目代码如下：

```
<html>
<head>
    <meta charset="utf-8">
    <script language="javascript">
        function creatwindow(){
            var w=window.open("数字式电子时钟.html","mywindow","menubar=no,height=
        200,width=300,resizable=no","true");
        }
        function closewindow(){
            w.close();
        }
    </script>
</head>
<body>
<form>
    <input type="button" value="创建窗口" onclick="creatwindow()">
    <input type="button" value="关闭窗口" onclick="closewindow()">
</form>
</body>
</html>
```

将以上 HTML 文件保存为“打开新窗口. html”，使用浏览器打开，分别点击“创建窗口”和“关闭窗口”按钮，观察弹出新窗口的信息和关闭当前窗口的效果。

4. 知识运用

(1)在 HTML 页面的主体部分，放置一个“删除”按钮和“输入用户名”按钮，点击“删除”按钮时，要弹出提示“确认要删除吗?”；点击“输入用户名按钮”，弹出输入用户名对话框(window. prompt())，用户输入用户名完毕点击“确认”按钮之后，弹出对话框，显示提示信息格式如下：“您的名字是：XXX”，如图 5. 10 所示。

图 5. 10　显示用户名对话框

(2)在 HTML 页面的主体部分,显示当前页面的完整的 URL 地址字符串、当前 URL 中的通信协议以及当前时间。另外,在下方放置一个“超链接按钮”和“刷新按钮”,点击“超链接按钮”时,跳转到 5.4 节的“数字电子时钟.html”页面,点击刷新按钮,刷新当前页面。效果如图 5.11 所示。

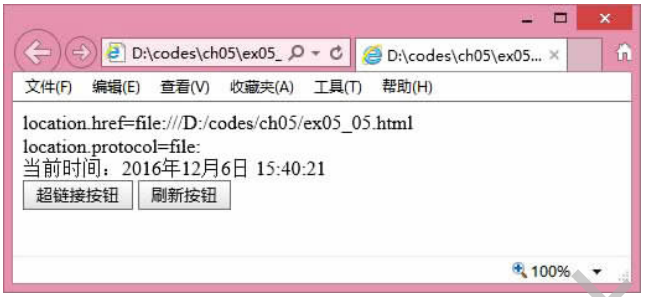

图 5.11　显示超链接和刷新按钮

5.6　DOM 对象

5.6.1　认识 DOM 对象

文档对象模型(Document Object Model,简称 DOM),是 W3C 组织推荐的处理可扩展标志语言的标准编程接口。HTML DOM 是 W3C 标准,定义了用于 HTML 的一系列标准的对象,以及访问和处理 HTML 文档的标准方法。HTML DOM 把 HTML 文档呈现为带有元素、属性和文本的树结构(节点树)。

通过 DOM,可以访问所有的 HTML 元素,连同它们所包含的文本和属性。可以对其中的内容进行修改和删除,同时也可以创建新的元素。HTML DOM 独立于平台和编程语言。它可被任何编程语言诸如 Java、JavaScript 和 VBScript 使用。

通过可编程的对象模型,JavaScript 获得了足够的能力来创建动态的 HTML,主要包括:

- JavaScript 能够改变页面中的所有 HTML 元素;
- JavaScript 能够改变页面中的所有 HTML 属性;
- JavaScript 能够改变页面中的所有 CSS 样式;
- JavaScript 能够对页面中的所有事件做出反应。

下面分别介绍一下 HTML DOM 中的 HTML DOM Document、HTML DOM Event、HTML DOM Element 和 HTML DOM Attribute 对象。

5.6.2　HTML DOM Document 对象

HTML DOM Document 对象表示每个载入浏览器的 HTML 文档,Document 对象使我们可以从脚本中对 HTML 页面中的所有元素进行访问。Document 对象是 Window 对象的一部分,可通过 window.document 属性对其进行访问。

document 对象是 JavaScript 实现网页各种功能中最常用的基本对象之一，它代表浏览器窗口中的文档，可以用来处理文档中包含的 html 元素，如表单，图像，超链接等。

document 对象的常用属性和方法如表 5.12 所示：

表 5.12　　document 对象常用属性和方法

	属性名/方法名	描述
常用属性	link	文挡中的一个<a href=>超链接标签，该属性本身是个对象
	links	文件中的所有链接对象，以数组索引值表示
	linkColor	文挡的链接颜色
	alinkColor	活动链接颜色
	vlinkColor	已点击的超链接颜色
	layer，link，image，form，area，applet，anchor，cookie	是 document 对象的几个子对象
	cookie	存储 cookie. txt 文件的一段信息
	forms	文件中的所有表单，以数组索引值表示
	images	文挡中所有 image，以数组索引值表示
	anchors	文挡中所有锚点，以数组索引值表示
	bgColor	文挡中的背景颜色
	fgColor	文挡中文本颜色
	URL	表示该文件的网址
	title	文挡的标题
	lastModified	文挡最后的修改日期
常用方法	write(text)	向当前页面输出信息 text
	getSelection()	获取当前选取的字符串，返回值是当前选取的字符串
	getElementsByName(name)	通过 HTML 标签的 name 属性来获得一些元素对象，返回的是具有相同 name 属性的 HTML 元素对象集合
	getElementById(id)	通过 HTML 标签的 id 属性来获得一个 HTML 元素对象，返回具有该 id 属性的 HTML 元素对象
	getElementsByTagName(tagname)	通过 HTML 标签名获得指定标签名 tagname 的 HTML 元素对象集合

通常，通过 JavaScript 需要操作 HTML 元素，会使用getElementById()、getElementsByName()或getElementsByTagName()三个方法之一，来获取指定的 HTML 元素对象或对象集合。

1. getElementById()

通过 HTML 标签的 id 属性来获得一个 HTML 元素对象，返回具有该 id 属性的 HTML 元素对象。

例如，若要获取表单中 id 属性值为“username”的文本框的输入内容，可以使用如下

语句：

```
var username=document.getElementById("username").value;
```

2. getElementsByName()

通过 HTML 标签的 name 属性来获得一些元素对象，返回的是具有相同 name 属性的 HTML 元素对象集合。

例如，若要获取表单中第一个 name 属性值为“username”的文本框的输入内容，可以使用如下语句：

```
var un=document.getElementsByName("username");
var username=un[0].value;
```

3. getElementsByTagName()

通过 HTML 标签名获得指定标签名的 HTML 元素对象集合。

例如，若要获取表单中第一个标签名为“input”的输入控件的输入内容，可以使用如下语句：

```
var un=document.getElementsByTagName("input");
var username=un[0].value;
```

在以上列出的 document 的常用属性中，forms 代表 form 对象集合。form 对象的常用属性和方法如表 5.13 所示：

表 5.13　　form 对象常用属性和方法

	属性名/方法名	描述
常用属性	name	表单名，相当于<form>标签中的 name 属性
	action	表单提交时执行的动作，相当于<form>标签的 action 属性
	method	表单的提交方式，相当于<form>标签的 method 属性
	elements	表单中的所有控件，以数组索引值表示
	length	表单中的控件个数
	textarea，text，file，passward，hidden，sumit，radio，checkbox，button，reset，select	表单对象的 11 个子对象
常用方法	submit()	提交表单，相当于单击表单中提交按钮的作用
	reset()	将所有表单中的控件值重置为默认值，相当于单击表单中重置按钮的作用

当 JavaScript 读到 HTML 标签中对应的 form 表单输入控件标签时，会自动建立一个该类型的 form 子对象，并将该对象存放到 form 对象的 elements 数组当中。

对表单的子对象的访问可以有以下几种格式：

1. 利用表单对象的 elements 属性访问：

```
document.forms[n].elements[n].子对象属性名
document.forms[n].elements[n].子对象方法名
```

2. 利用表单名和子对象名访问：

document.表单名.子对象名.子对象属性名

document.表单名.子对象名.子对象方法名

3.混合访问方式：

document.forms[n].子对象名.子对象属性名

document.forms[n].子对象名.子对象方法名

对于 form 对象的常用子对象，它们的属性和方法如表 5.14 所示：

表 5.14　　form 的常用子对象的属性和方法

子对象	属性名/方法名	描述
text password textarea	defaultValue	缺省值(相当于标签中 value 属性的值)
	name	文本对象的名字(相当于标签中 name 属性的值)
	value	文本对象的当前值(相当于标签中 value 属性的值)
	form	文本对象所在的表单
	blur()	文本对象失去焦点
	focus()	文本对象得到焦点
	select()	文本对象设置成选取状态
radio checkbox	checked	设置对象的选中状态，返回布尔值(true 代表选中)
	defaultchecked	默认选中状态
	name	对象的名字(相当于标签中 name 属性的值)
	value	该对象的值(相当于标签中 value 属性的值)
	form	对象所在的表单
	blur()	对象失去焦点
	focus()	对象得到焦点
	click()	在该对象上单击鼠标左键
button submit reset	name	按钮对象的名字(相当于标签中 name 属性的值)
	value	按钮对象上的文字(相当于标签中 value 属性的值)
	form	按钮对象所在的表单
	blur()	按钮对象失去焦点
	focus()	按钮对象得到焦点
	click()	在该按钮对象上单击鼠标左键
select	name	该列表对象的名字(相当于标签中 name 属性的值)
	length	该列表对象 option 的数目
	form	该列表对象所在表单
	options	存放＜select＞标签中的所有＜option＞标签的对象数组，＜option＞本身也对应为 option 子对象
	selectedIndex	选中项目的索引值(从 0 开始)
	blur()	列表对象失去焦点
	focus()	列表对象得到焦点

（续表）

子对象	属性名/方法名	描述
option （select 的子对象）	defaultSelected	指定该选项为默认选择状态
	index	所有选项构成的数组索引值
	length	select. options 数组的元素个数，与 select 对象的 length 相同
	selected	菜单项是否被选中，返回布尔值
	text	该菜单项所显示的文字
	value	该菜单项的值（相当于标签中 value 属性的值）

5.6.3　HTML DOM Event 对象

HTML DOM Event 对象表示事件的状态，比如事件在其中发生的元素、键盘按键的状态、鼠标的位置、鼠标按钮的状态等等。

event 对象用于访问鼠标、键盘状态的属性如表 5.15 所示：

表 5.15　　event 对象常用属性

属性名	描述
altKey	返回当事件被触发时，“ALT”是否被按下
button	返回当事件被触发时，哪个鼠标按钮被点击
clientX	返回当事件被触发时，鼠标指针的水平坐标
clientY	返回当事件被触发时，鼠标指针的垂直坐标
ctrlKey	返回当事件被触发时，“CTRL”键是否被按下
metaKey	返回当事件被触发时，“meta”键是否被按下
relatedTarget	返回与事件的目标节点相关的节点
screenX	返回当某个事件被触发时，鼠标指针的水平坐标
screenY	返回当某个事件被触发时，鼠标指针的垂直坐标
shiftKey	返回当事件被触发时，“SHIFT”键是否被按下

对于 IE 浏览器，还支持以下的 event 对象属性：

表 5.16　　event 对象属性（IE 浏览器）

属性名	描述
cancelBubble	如果事件句柄想阻止事件传播到包容对象，必须把该属性设为 true
keyCode	对于 keypress 事件，该属性声明了被敲击的键生成的 Unicode 字符码。对于 keydown 和 keyup 事件，它指定了被敲击的键的虚拟键盘码。虚拟键盘码可能和使用的键盘的布局相关
offsetX，offsetY	发生事件的地点在事件源元素的坐标系统中的 x 坐标和 y 坐标
returnValue	如果设置了该属性，它的值比事件句柄的返回值优先级高。把这个属性设置为 fasle，可以取消发生事件的源元素的默认动作

（续表）

属性名	描述
srcElement	对于生成事件的 Window 对象、Document 对象或 Element 对象的引用
toElement	对于 mouseover 和 mouseout 事件，该属性引用移入鼠标的元素
x，y	事件发生的位置的 x 坐标和 y 坐标，它们相对于用 CSS 动态定位的最内层包容元素

5.6.4 HTML DOM Element 对象

在 HTML DOM Element 对象表示任意的 HTML 元素。元素对象可以拥有类型为元素节点、文本节点、注释节点的子节点。NodeList 对象表示节点列表，比如 HTML 元素的子节点集合。元素也可以拥有属性，属性是属性节点。

element 对象的属性和方法可用于所有 HTML 元素上，element 对象的常用属性和方法如表 5.17 所示：

表 5.17　　element 对象常用属性和方法

	属性名/方法名	描述
常用属性	attributes	返回元素属性的 NamedNodeMap
	childNodes	返回元素子节点的 NodeList
	firstChild	返回元素的首个子元素
	lastChild	返回元素的最后一个子元素
	nodeName	返回元素的名称
	nodeType	返回元素的节点类型
	nodeValue	设置或返回元素值
	ownerDocument	返回元素的根元素（文档对象）
	parentNode	返回元素的父元素
	previousSibling	返回位于相同节点树层级的前一个元素
	nextSibling	返回位于相同节点树层级的下一个元素
	innerHTML	设置或返回元素的内容
	style	设置或返回元素的 style 属性
	textContent	设置或返回节点及其后代的文本内容
常用方法	insertBefore(newItem，existingItem)	在指定的已有的子节点 existingItem 之前插入新节点 newItem
	getAttribute(attrname)	返回元素节点的指定属性名 attrname 的属性值
	getElementsByTagName(tagname)	返回拥有指定标签名 tagname 的所有子元素的集合
	setAttribute(attrname，attrvalue)	把名为 attrname 的属性设置或更改为指定值 attrvalue

5.6.5　HTML DOM Attribute 对象

在 HTML DOM 中，Attribute 对象表示 HTML 属性，NamedNodeMap 对象表示元素属性节点的无序集合，其中的节点可通过名称或索引来访问。

attribute 对象的常用属性和方法如表 5.18 所示：

表 5.18　　attribute 对象常用属性和方法

	属性名/方法名	描述
常用属性	name	返回属性的名称
	value	设置或返回属性的值
	length	返回 NamedNodeMap 中的节点数
常用方法	getNamedItem(name)	从 NamedNodeMap 返回具有指定名称 name 的属性节点
	setNamedItem(name)	通过名称 name 设置或添加指定的属性节点
	removeNamedItem(name)	通过名称 name 删除指定的属性节点
	item(index)	返回 NamedNodeMap 中位于指定下标 index 的节点

5.6.6　项目：诗词鉴赏

1. 项目说明

在 HTML 页面中，放置单行文本框和“换一首诗”按钮，在文本框中输入诗的第一句，点击“换一首诗”按钮，将文本框输入的第一句显示在左侧区域。

继续放置单行文本框和“添加”按钮，在文本框中继续输入诗的下一句，点击“添加”按钮，将下一句显示在左侧区域之前诗句的下方。

继续放置单行文本框和“改颜色”按钮，在文本框中输入颜色名称，如“red”，“blue”，点击“改颜色”按钮，将左侧区域中的文本颜色改为相应的颜色。效果图如图 5.12 所示。

图 5.12　诗词鉴赏页面

2. 项目设计

本项目是一个对 JavaScript DOM(文档对象模型)对象的各个常用方法的使用实例。

在本项目中,需要利用 document 对象的方法来获取 HTML 页面上的各个指定元素,并获取和修改对应元素的内容。HTML DOM 定义了一整套访问和处理 HTML 文档元素的标准方法。通过 DOM,可以访问所有的 HTML 元素以及它们所包含的文本和属性,也可以对其中的内容进行修改和删除或者创建新的元素。document 对象就是 HTML DOM 的一个实例,可以使用它的方法来访问和操作 HTML DOM 元素。

在本项目中,利用 document 对象的 getElementsByName()、getElementsByTagName()方法获取指定元素。并使用了 event 对象和 element 对象的常用属性获取和设置相关内容。

(1)首先,在 HTML 页面的主体部分,放置第一个文本框(name="txt1"),并设置其 onfocus 事件的处理函数为 clr(),用于当文本框获得焦点时,清空文本框的内容。

(2)当在文本框上发生 onfocus 事件时,会调用 clr()函数并将当前事件对象 event 做参数传递到 clr()函数。在 clr()函数中,使用 event 对象的 srcElement 属性获取当前发生事件的 element 对象(即第一个文本框),并设置其 value 属性为空(即清空文本框中的内容)。

(3)对"换一首诗"按钮添加 onclick 事件并设置事件处理函数为 changecontent()。在 changecontent()函数中,使用 document 对象的 getElementsByName()方法获取名为"txt1"的元素集合,使用 document 对象的 getElementsByTagName()方法获取标签名为"div"的元素集合。需要注意的是,这两个方法均返回元素集合,需要使用下标访问其中的某一个元素。使用 element 对象的 innerHTML 属性将左侧显示区域 标签体内容设置为文本框中输入的内容(文本框输入内容使用 element 对象的 value 属性获取)。

(4)其余功能的实现,与上面使用的属性和方法类似,在此不再赘述。

3. 项目实施

本项目代码如下:

```
<html>
<head>
    <style>
        body{background:url(../images/juanzhou.jpg)no-repeat;
        background-position:50 % 0;}
        #box1{width:750px;height:350px;padding:10px;margin:140px auto;}
        #box2{font-family:隶书; font-size:1.5em;font-weight:bold;
        width:300px; height: 250px; float: left; margin: 40px 10px; padding: 10px; text-align:
    center;letter-spacing:2px;}
        #box3{width:350px;height:250px;padding:100px 10px;float:right}
        img{position:relative;right:-150px;border-radius:15px}
    </style>
    <script language="JavaScript">
        function changecolor(){
            var color= document.getElementsByName("colorpanel");
            document.getElementById("box2").style.color=color[0].value;
        }
```

```
        function changecontent(){
            var t1=document.getElementsByName("txt1");
            var div1=document.getElementsByTagName("div");
            div1[1].innerHTML="<p>"+t1[0].value+"<p>";
        }
        function hide(){
            document.getElementById("box2").style.display="none";
        }
        function show(){
            document.getElementById("box2").style.display="block";
        }
        function add(){
            var t2=document.getElementsByName("txt2");
            var div2=document.getElementsByTagName("div");
            div2[1].innerHTML+="<p>"+t2[0].value+"</p>";
        }
        function clr(event){
            event.srcElement.value="";
        }
    </script>
</head>
<body>
    <div id="box1">
        <div id="box2"><p>离离原上草</p></div>
        <div id="box3">
            <!--改内容-->
            <input type="text" name="txt1"  onfocus="clr(event)">
            <button onclick="changecontent()">换一首诗</button><br><br>
            <!--添加内容-->
            下一句诗
            <input type="text" name="txt2" onfocus="clr(event)">
            <button onclick="add()">添加</button><br><br>
            <!--背景色-->
            <input type="color" name="colorpanel">
            <button onclick="changecolor()">改颜色</button> <br><br>
            <!--隐藏、显示-->
            <button onclick="hide()">隐藏</button>
            <button onclick="show()">显示</button><br><br>
        </div>
    </div>
</body>
</html>
```

将以上 HTML 文件保存为“诗词鉴赏.html”，使用浏览器打开，输入诗的下一句，点击“添加”按钮，输入颜色名称，点击“改颜色”按钮，并点击“隐藏”和“显示按钮”，观察页面上的输出信息。

4. 知识运用

在以上 HTML 页面中，添加输入诗名与诗作者的文本框，并放置相应的“添加”按钮，点击添加按钮后，将诗名和诗作者信息显示在左侧区域。效果如图 5.13 所示。

图 5.13　添加诗名和作者

5.7　咖啡商城——购物车模块实现

本项目要实现的功能属于综合项目中的购物车页面中的功能模块。本项目要利用本章学习的 JavaScript 技术实现综合项目中的购物车页面的金额计算和显示。

1. 可以利用 JavaScript 的事件处理机制来完成对全选控件的事件监听和事件处理，并使用 JavaScript 的 document 对象的方法获取选中商品的总价，进行总金额计算，并把计算结果显示在指定的某个 HTML 元素中。

2. 同时，也可以利用 JavaScript 的事件监听机制，监听商品个数文本框的内容变化事件，一旦个数发生变化，重新计算商品总金额并显示。

5.7.1　项目说明

使用浏览器打开“购物车.html”页面，选中要购买的商品，输入商品数量，观察显示商品件数和总金额的区域，随之发生变化，如图 5.14 所示。

图 5.14　购物车计算商品金额功能

在本项目中，主要需要完成以下几个功能：

1. 实现购物车中所有商品的全选和取消全选的功能；
2. 实现计算所有选中的商品的个数和总金额的功能；
3. 实现选中商品的个数和总金额显示的功能；
4. 实现修改购买商品个数，则选中商品的个数和总金额也随之变化的功能。

5.7.2　项目设计

在本项目中需要完成的功能的设计思路如下：

1. 实现购物车中所有商品的全选和取消全选的功能：

对“全选商品信息”复选框的 onchange 事件进行监听，编写该事件的 JavaScript 事件处理函数。当发生 onchange 事件时，调用 checkAll()事件处理函数。在 checkAll()中，根据复选框的 checked 属性判断复选框是否处于选中状态，若是选中状态，则把购物车页面中所有复选框元素的状态均设置为 true，即设置为选中状态；否则，把购物车页面中所有复选框元素的状态均设置为 false，即设置为不选中状态。这样就实现了商品的全选和取消全选功能。

最后，需要在“全选商品信息”复选框的 onchange 事件处理函数中，调用计算总金额的函数 cal()，以重新计算当前选中商品的总金额。

2. 实现计算所有选中的商品的个数和总金额的功能：

计算所有选中的商品的个数和总金额的函数 cal()中，声明用于保存选中商品个数的变量 piece 和保存选中商品总金额的变量 money，初值均为 0。循环遍历每一个选中商品对应的显示商品总价价的＜div＞元素，商品个数累计到 piece 变量中，取出＜div＞元素中的内容即总价，累加到 money 变量中。

3. 实现选中商品的个数和总金额显示的功能：

在 cal()函数中，循环遍历完毕所有的商品之后，将最终累加得到的计算结果显示到 id

为“piece”和“money”的 div 元素所标记的区域中。

4. 实现修改购买商品个数，则选中商品的个数和总金额也随之变化的功能：

对每个商品输入购买数量的文本框的 onchange 事件进行监听，编写该事件的 JavaScript 事件处理函数 count()。count()函数中，获取商品的个数和单价信息，计算商品总价，并显示在用于显示商品总价的 元素中。最后，调用 cal()函数，重新计算选中商品的总金额和个数。

5.7.3 项目实施

1. 对“全选商品信息”复选框的 onchange 事件进行监听，指定事件处理函数是 checkAll()：

```
<input type="checkbox" name="selectAll"  onchange="checkAll(event)">全选商品信息</div>
```

然后编写以上复选框的 onchange 事件处理函数 checkAll()：

```
function checkAll(e){
    if(e.srcElement.checked){ //全选
        var o=document.getElementsByTagName("input");
        for(i=0;i<o.length;i++){
            if(o[i].type==="checkbox") o[i].checked=true;
        }
    }
    else{ //取消全选
        var o=document.getElementsByTagName("input");
        for(i=0;i<o.length;i++){
            if(o[i].type==="checkbox") o[i].checked=false;
        }
    }
    cal();
}
```

2. 编写计算总金额的函数 cal()：

```
//计算总价
function  cal(){
        var piece=0; var money=0;
        varo=document.getElementsByTagName("input");
        for(i=0;i<o.length;i++){
            if(o[i].type==="checkbox"){
                if(o[i].checked&&o[i].name==="goods"){
                    t=document.getElementById("u"+o[i].id).innerHTML;
                    piece+=1;money+=Number(t);
                }
            }
        }
        document.getElementById("piece").innerHTML=piece;
        document.getElementById("money").innerHTML=money.toFixed(2);
}
```

3. 编写购买数量的文本框的 onchange 事件处理函数 count()：

```
//计算金额
function count(x){
    var price=document.getElementById("p"+x).innerHTML;
    var num=document.getElementById("n"+x).value;
    document.getElementById("u"+x).innerHTML=price*num;
    cal();
}
```

习　题

一、不定项选择题

1. 点击页面的按钮，使之打开一个新窗口，加载一个网页，以下 JavaScript 代码中可行的是(　　)。

A.
```
<input type="button" value="new"
onclick="open('new.html', '_blank') "/>
```

B.
```
<input type="button" value="new"
onclick="window.location='new.html';"/>
```

C.
```
<input type="button" value="new"
onclick=" location.assign('new.html');"/>
```

D.
```
<form target="_blank" action="new.html">
    <input type="submit" value="new"/>
</form>
```

2. 使用 JavaScript 向网页中输出<h1>hello</h1>，以下代码中可行的是(　　)。

A.
```
<script type="text/javascript">
    document.write(<h1>hello</h1>);
</script>
```

B.
```
<script type="text/javascript">
    document.write("<h1>hello</h1>");
</script>
```

C.
```
<script type="text/javascript">
    <h1>hello</h1>
</script>
```

D.
```
<h1>
<script type="text/javascript">
    document.write("hello");
</script>
</h1>
```

3. 分析下面的代码：

```
<html>
<head>
```

```
    <script type="text/javascript">
        function writeIt (value) { document.myfm.first_text.value=value;}
    </script>
</head>
<body bgcolor="#ffffff">
    <form name="myfm">
        <input type="text" name="first_text">
         <input type="text" name="second_text" onchange="writeIt(value)">
    </form>
</body>
</html>
```

以下说法中正确的是(　　)。

A. 在页面的第二个文本框中输入内容后,当鼠标离开第二个文本框时,第一个文本框的内容不变

B. 在页面的第一个文本框中输入内容后,当鼠标离开第一个文本框时,将在第二个文本框中复制第一个文本框的内容

C. 在页面的第二个文本框中输入内容后,当鼠标离开第二个文本框时,将在第一个文本框中复制第二个文本框的内容

D. 在页面的第一个文本框中输入内容后,当鼠标离开第一个文本框时,第二个文本框的内容不变

4. 下面的 JavaScript 语句中,(　　)实现检索当前页面中的表单元素中的所有文本框,并将它们全部清空。

A.
```
for(var i=0;i< form1.elements.length;i++) {
    if(form1.elements[i].type=="text")
      form1.elements[i].value="";
}
```

B.
```
for(var i=0;i<document.forms.length;i++) {
    if(forms[0].elements[i].type=="text")
       forms[0].elements[i].value="";
}
```

C.
```
if(document.form.elements.type=="text")
    form.elements[i].value="";
```

D.
```
for(var i=0;i<document.forms.length; i++){
    for(var j=0;j<document.forms[i].elements.length; j++){
      if(document.forms[i].elements[j].type=="text")
        document.forms[i].elements[j].value="";
      }
}
```

5. 在 IE 中要想调整当前窗口的大小为指定的宽和高，可以使用 window 对象的(　　)方法。

A. windowX　　B. resizeTo　　C. moveTo　　D. windowLeft

6. 分析下面的 JavaScript 代码段，输出结果是(　　)。

```
a=new Array(2,3,4,5,6);
sum=0;
for(i=1;i<a.length;i++)
   sum+=a[i];
document.write(sum);
```

A. 20　　B. 18　　C. 14　　D. 12

7. 下面对于 JavaScript 中的单选按扭(Radio)的说法正确的是(　　)。

A. 单选按钮可以通过单击“选种”和“未选中”选项来进行切换

B. 单选按钮没有 checked 属性

C. 单选按钮支持 onClick 事件

D. 单选按钮的 length 属性返回一个选项组中单选项的个数

8. 下列(　　)属性为布尔属性(即只需要指定属性的存在，而不用指定其值的属性)。

A. noshade　　B. width　　C. bold　　D. size

9. 在某一页面下载时，要自动显示出另一页面，可通过在<body>中使用下边的哪一事件来完成(　　)。

A. onload　　B. onunload　　C. onclick　　D. onchange

10. 在 HTML 中，location 对象的(　　)属性用于设置或检索 URL 的端口号。

A. host　　B. port　　C. pathname　　D. href

11. 下面哪个选项中的对象与浏览历史记录列表有关(　　)。

A. location，history　　B. window，location

C. navigator，window　　D. historylist，location

12. 下列 JavaScript 语句中，(　　)能实现单击一个按钮时弹出一个消息框。

A. <button value ="鼠标响应" onClick=alert("确定")></button>

B. <input type="button" value ="鼠标响应" onClick=alert("确定")>

C. <input type="button" value ="鼠标响应" onChange=alert("确定")>

D. <button value="鼠标响应" onChange=alert("确定")></button>

13. 在 HTML 页面中，下面关于 Window 对象的说法不正确的是(　　)。

A. Window 对象表示浏览器的窗口，可用于检索有关窗口状态的信息

B. Window 对象是浏览器所有内容的主容器

C. 浏览器打开 HTML 文档时，通常会创建一个 Window 对象

D. 如果文档定义了多个框架，浏览器只为原始文档创建一个 Window 对象，无须为每个框架创建 Window 对象

14. 在 JavaScript 中，表单文本框(Text)不支持的事件包括(　　)。

A. onBlur　　B. onLostFocused　　C. onFocus　　D. onChange

15. 分析下面的 JavaScript 代码，m 的值为(　　)。

```
x=11;
y="number";
m= x+y ;
```

A. 11number　　B. number　　C. 11　　D. 程序报错

16. 在 HTML 页面中使用外部 JavaScript 文件的正确语法是(　　)。

A. <language="JavaScript" src="scriptfile.js">

B. <script language="JavaScript" src="scriptfile.js"></script>

C. <script language="JavaScript" =scriptfile.js></script>

D. < language src=" scriptfile.js">

17. 分析如下的 JavaScript 代码段，则运行后在页面上输出(　　)。

```
var c="10",d=10;
document.write(c+d)
```

A. 10　　B. 20　　C. 1010　　D. 页面报错

18. 网页编程中，运行下面的 JavaScript 代码，则提示框中显示(　　)。

```
<script language="javascript">
x=3;
y=2;
z=(x+2)/y;
alert(z);
</script>
```

A. 2　　B. 2.5　　C. 32/2　　D. 16

19. 在 JavaScript 中，命令按纽(Button)支持的事件包括(　　)。

A. onClick　　B. onChange　　C. onSelect　　D. onSubmit

20. 在当前页面的同一目录下有一名 show.js 的文件，下列(　　)代码可以正确访问该文件。

A. <script language="show.js"></script>

B. <script type="show.js"></script>

C. <script src="show.js"></script>

D. <script runat="show.js"></script>

21. 在 JavaScript 中，可以使用 Date 对象的(　　)方法返回该对象的日期 。

A. getDate　　B. getYear　　C. getMonth　　D. gerTime

22. 哪一个对象可以获得屏幕的大小(　　)。

A. window　　B. screen　　C. navigator　　D. screenX

23. 分析下面的 JavaScript 语句，执行后 str 的结果是(　　)。

```
Str="This apple costs "+5 0.5;
```

A. This apple costs 50.5　　B. This apple costs 5.5

C. "This apple costs" 50.5　　D. "This apple costs "5.5

24. setInterval("alert('welcome');",1000);这段代码的意思是(　　)。

A. 等待 1000 秒后,再弹出一个对话框　　B. 等待 1 秒钟后弹出一个对话框

C. 语句报错,语法有问题　　D. 每隔一秒钟弹出一个对话框

25. 要求用 JavaScript 实现下面的功能:在一个文本框中内容发生改变后,单击页面的其他部分将弹出一个消息框显示文本框中的内容,下面语句正确的是(　　)。

A. <input type="text" onChange="alert(this.value)">

B. <input type="text" onClick="alert(this.value)">

C. <input type="text" onChange="alert(text.value)">

D. <input type="text" onClick="alert(value)">

26. window 对象的 open 方法返回的是(　　)。

A. 没有返回值

B. boolean 类型,表示当前窗口是否打开成功

C. 返回打开新窗口的对象

D. 返回 int 类型的值,开启窗口的个数

27. 分析下面的 JavaScript 代码段,输出的结果是(　　)。

```
function employee(name,code){
  this.name="wangli";
  this.code="A001";
}
newemp=new employee("zhangming",'A002');
document.write("雇员姓名:"+newemp.name+"<br>");
document.write("雇员代号:"+newemp.code+"<br>");
```

A. 雇员姓名:wangli 雇员代码:A001

B. 雇员姓名:zhangming 雇员代码:A002

C. 雇员姓名:null, 雇员代码:null

D. 代码有错误,无输出结果

28. 分析下面的 JavaScript 代码段,输出结果是(　　)。

```
a=new Array("100","2111","41111");
for(var i=0;i < a.length;i++){
    document.write(a[i]+" ");
}
```

A. 100 2111 41111　　B. 1 2 3　　C. 0 1 2　　D. 1 2 4

29. 分析下面的 JavaScript 代码段,输出的结果是(　　)。

```
var a=15.49;
document.write(Math.round(a));
```

A. 15　　B. 16　　C. 15.5　　D. 15.4

30. 以下(　　)为 JavaScript 声明变量的语句。

A. dim x;　　B. int x;　　C. var x;　　D. x;

31. 分析如下的 JavaScript 代码片段，b 的值为(　　)。

```
var a=1.5,b;
b=parseInt(a);
```

A. 2　　B. 0.5　　C. 1　　D. 1.5

二、综合题

1. 补充按钮事件的函数，确认用户是否退出当前页面，确认之后关闭窗口。

```
<html>
<head>
<script  type="text/javascript" >
function closeWin(){
    //在此处添加代码
}
</script>
</head>
<body>
    <input type="button" value="关闭窗口" onclick="closeWin()"/>
</body>
</html>
```

2. 完成 foo()函数的内容，要求能够弹出对话框提示当前选中的是第几个单选框。

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
<body>
<script  type="text/javascript" >
function   foo()   {
    //在此处添加代码
}
</script>
<body>
<form   name="form1"   onsubmit="return foo();">
<input   type="radio"   name="radioGroup"/>
<input   type="radio"   name="radioGroup"/>
<input   type="radio"   name="radioGroup"/>
<input   type="radio"   name="radioGroup"/>
<input   type="submit"/>
</form>
</body>
</html>
```

3. 完成函数 showImg(),要求能够动态根据下拉列表的选项变化,更新图片的显示。

```
<body>
<script  type="text/javascript" >
functionshowImg (oSel)  {
    //在此处添加代码
}
</script>
  <img id="pic" src="img1.jpg" width="200" height="200" />
  <br />
  <select id="sel" onchange="showImg(this)">
    <option value="img1">城市生活</option>
    <option value="img2">都市早报</option>
    <option value="img3">青山绿水</option>
</select></body>
```

4. 设计一个含有一个表单的页面,并在表单中放入一个文本框。编写 JavaScript 程序,当鼠标在页面上移动时,将鼠标的坐标显示在这个文本框中。完成如图 5.15 所示的效果。

图 5.15　页面示例

5. 在页面上放置两个多选下拉列表,当用户在左侧列表中选择任意项,可以通过“>>”按钮添加到右侧列表中,也可以通过“<<”按钮将其从右侧列表移回到左侧列表中。编写 JavaScript 程序,实现如图 5.16 所示效果。

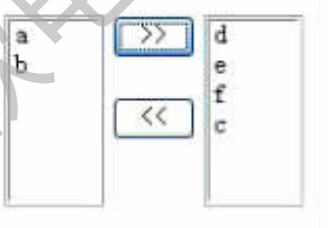

图 5.16　页面示例