第3章 JavaScript 与 VBScript

JavaScript 和 VBScript 是网页设计中常用的两种脚本描述语言。本章将介绍这两种脚本语言的语法规则和在网页设计中的应用。

- 了解常用的脚本描述语言及其特点，理解客户端脚本与服务器端脚本的区别。
- 掌握网页中嵌入使用 JavaScript 脚本的方法，掌握 JavaScript 的语法与常用的函数。
- 掌握常用的浏览器对象的功能与用法，掌握 JavaScript 的事件响应过程的编写方法。
- 掌握 VBScript 脚本语言的基本语法与常用函数。

3.1 客户端脚本与服务器端脚本

根据脚本代码是在客户端执行还是在服务器端执行，脚本程序分为客户端脚本和服务器端脚本两种。客户端脚本是由浏览器来解释执行的，服务器端脚本是由 Web 服务器负责解释执行的。

3.1.1 客户端脚本简介

客户端脚本由浏览器负责解释执行，由于客户端浏览器的版本种类较多，不同的版本对脚本的支持程度不同，为了使制作的网页兼容性更好，客户端脚本通常使用 JavaScript 来编写。

脚本是一种能够完成某些特殊功能的小程序段。JavaScript 是 Netscape 公司推出的一种嵌入 HTML 文档的、基于对象的脚本描述语言。利用 JavaScript 可进一步增强网页的交互性，并可通过访问和操作浏览器对象，实现控制浏览器外观、状态和运行方式

等目的。

JavaScript是一种解释性的描述语言，不同于一般的程序设计语言，它不能用来开发独立的应用程序，只能嵌入到网页中使用。

Microsoft在Netscape发布的JavaScript基础上，独立地开发了一套符合JavaScript语法的JScript语言，在JScript脚本语言中，可以使用原JavaScript中的一切语法和语句，同时还新增了许多实用的功能，JScript可以视为是JavaScript的Microsoft版。二者的用法相同，JScript包含了JavaScript的全部内容，并进行了功能扩充。因此，Microsoft的IE浏览器支持JavaScript或JScript和VBScript。

3.1.2 服务器端脚本简介

服务器端脚本运行于服务器端，由Web服务器负责解释执行。ASP网页中的ASP代码就属于服务器端脚本。

ASP网页的服务器端脚本通常采用VBScript来编写，也可采用JavaScript来编写。IIS Web服务器默认的脚本语言是VBScript，可在IIS Web服务器中利用网站的属性对话框来进行更改。

3.2 JavaScript脚本语言

3.2.1 网页中嵌入使用JavaScript

在网页中嵌入和使用JavaScript时，必须将脚本代码放在＜Script＞与＜/Script＞标记符之间，以便将脚本代码与HTML标记符区分开来。Script块可放在＜head＞与＜/head＞之间，也可放在＜body＞与＜/body＞之间。其嵌入方法为：

```
<Script Language="JavaScript">
    '此处放置 JavaScript 代码
</Script>
```

＜Script＞的Language属性用于指定脚本的语言类型。若嵌入的是VBScript，则相应的嵌入方法为：

```
<Script Language="VBScript">
    '此处放置 VBScript 代码
</Script>
```

下面的示例将演示在网页中如何利用JavaScript弹出一个消息框，从中可看出在网页中使用JavaScript的方法。

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

```
    <title> 网页中使用 JavaScript</title>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
  </head>
  <body>
    <Script Language="JavaScript">
      window.alert("欢迎进入 JavaScript 世界!");
    </Script>
  </body>
</html>
```

此处利用了浏览器提供的 window 对象的 alert 方法，来实现以消息框方式显示数据。window 对象可默认，因此可简单表达为：

```
alert("欢迎进入 JavaScript 世界!");
```

如果一段 JavaScript 需要应用于多个网页，则可将该 JavaScript 代码单独保存到一个扩展名为.js 的文本文件中，当网页中需要使用该 JavaScript 代码时，只需利用＜Script＞标记的 src 属性将其包含到网页中即可。在网页中插入引用 JavaScript 文件的方法为：

```
<Script Language="JavaScript src="js-url"></Script>
```

src 属性用于指定所要插入的 JavaScript 文件名及路径。

例如，若要在网页中插入 inc 目录下面的 adnews.js 脚本文件，则插入方法为：

```
<Script Language="JavaScript Src="inc/adnews.js"></Script>
```

3.2.2 JavaScript 语法基础

1. JavaScript 的数据类型与语法特点

(1) JavaScript 的数据类型

JavaScript 提供四种基本的数据类型，分别为数值型、逻辑型、字符串型和 undefined 类型。

- 数值型：数值型由数字、小数点和正负符号组成，如：24361、3.14、－3 等。
- 逻辑型：逻辑型又称布尔型，用于代表事物的是与否、真与假这两种状态。逻辑真用 true 表示，逻辑假用 false 表示。
- 字符串型：字符串是由字符所构成的一个字符序列，在 JavaScript 中，字符串采用单引号或双引号括起来。
- undefined 型：undefined 是一种特殊的类型，对于一个已经声明但还未赋初值的变量，其类型就是 undefined 类型。

(2) JavaScript 的语法特点

JavaScript 的语法与 C 语言极为类似，是一种函数式的脚本语言。很多功能均是通过相应的函数或自定义函数来实现的。JavaScript 区分字母的大小写，语句以分号作为结束符，一条语句可表达在多行上，也可在同一行上书写表达多条语句。注释符采用 /*...*/或//。

JavaScript的赋值语句采用“=”,同时也支持+=、-=、*=、/=、%=等特殊用法。x+=2语句等价于x=x+2;x*=2语句等价于x=x*2,其余以此类推。

另外,赋值语句也支持将同一个值连续赋给多个变量。例如:

```
x=y=0;
```

2. 常量、变量与表达式

不同类型的数据既可以常量的形式出现,也可以变量的形式出现。常量又称字面量,是指在处理过程中其值固定不变的数据。变量实际上是一个存储单元在内存中的符号化地址,在该存储单元中,所存储的值是可改变的,故称之为变量。程序设计中,通常利用变量来保存临时数据,变量都有一个属于自己的名字,变量的名字称为变量名。

(1) 常量

根据数据类型的不同,常量可分为数值型常量、字符型常量和逻辑型常量。字符型常量必须用单引号或双引号括起来,逻辑型常量只有true和false两种。

另外,在JavaScript中还有一种特殊的常量,即转义字符,利用转义字符,可表达一些特殊的字符或控制符,JavaScript常用的转义字符如表3.1所示。

表3.1 转义字符

转义字符	代表字符	转义字符	代表字符
\b	退格符	\f	换行符
\n	换行符	\r	回车符
\t	Tab(横向跳格符)	\\	代表\字符
\'	代表单引号字符	\"	代表双引号字符

(2) 变量

JavaScript对变量的定义未做强制性规定,变量在使用之前,可以事先定义,也可不定义而直接使用。变量定义时不需要指定具体的数据类型,变量的数据类型完全由所赋的值的类型决定。

① 变量的定义

在JavaScript中,只要给变量赋一个值,就相当于定义了一个变量。另外也可用var语句来声明和定义一个变量,其定义语句用法为:

```
var 变量名1[=初值],变量名2[=初值]...
```

例如,若要定义变量tmpdate,则定义方法为:

```
var tmpdate;
```

在定义变量时,还可同时给变量赋一个初值,并且可以用var语句一次定义多个变量,各变量间用逗号分隔。例如:

```
var a=6, b=5;
```

② 变量的类型转换

JavaScript是一种对数据类型要求不太严格的脚本语言,在程序执行过程中,如果需

要，JavaScript 会自动进行一些必要的类型转换。在 IE 5.5 浏览器中，或安装了 JScript 5.6 版本后，当字符型与数值型进行“+”运算时，系统会将数值型数据转换成字符型，然后再进行字符串的连接运算。

在 JavaScript 中，也可显式地进行类型转换，将数码构成的字符串转换成数值型，可通过 Number()函数来实现，将数值型数据转换成字符串可用 String()函数来实现。

(3) 表达式与运算符

表达式就是由常量、变量、函数和相应的运算符所构成的式子。JavaScript 的表达式可分为条件表达式、数学运算表达式、关系运算表达式、字符表达式和逻辑表达式。

① 条件表达式

用法：

```
(条件)? A : B
```

功能：若条件成立，则表达式的值为 A；若条件不成立，则表达式的值为 B。A 和 B 可代表任何类型的值。

例如：

```
(age>=18)?"adult" : "minor"
```

若 age 变量的值大于或等于 18，则表达式运算后的最终值就为“adult”；若不大于 18，则表达式的值就为“minor”。

② 数学运算表达式

由数值型常量、变量或函数和数学运算符所构成的表达式，即为数学运算表达式。JavaScript 支持常见的+、-、*、/、取反和()运算符，同时也支持取模运算符(%)、增量运算符(++)和减量运算符(--)。

③ 关系运算表达式

关系运算表达式主要用于比较两个表达式之间的关系，其返回值为 true 或 false。若比较关系成立，则表达式返回值为 true，否则返回 false。常用的关系运算符主要有>(大于)、>=(大于或等于)、<(小于)、<=(小于或等于)、==(等于)、!=(不等于)。

④ 字符表达式

由字符型常量、变量、函数和相应的字符运算符所构成的表达式即为字符表达式。字符串运算主要是字符串的连接运算，其运算符为“+”。当在字符串连接运算中，若有数值型数据，系统会自动将数值型数据转换为字符型，然后再进行连接运算。

⑤ 逻辑表达式

由关系表达式、逻辑型值与逻辑运算符所构成的表达式即为逻辑表达式。运算后的最终结果仍为逻辑型值。JavaScript 支持的逻辑运算符有 &&(逻辑与)、||(逻辑或)、!(逻辑非)三种。

逻辑表达式常与条件分支语句、循环语句等配合使用，以提供循环或分支判断的条件。

3. 函数的定义与调用方法

(1) 函数的定义方法

函数是能实现某种运算或特定功能的程序段。用户可调用 JavaScript 语言提供的内

置函数或自定义的函数来实现所需的功能。

JavaScript用function语句定义函数，用return语句来返回函数的值。其定义格式为：

```
function 函数名(参数列表)
    {
        函数的执行主体
        return 表达式;
    }
```

在定义函数时，应注意以下方面：

- 函数名应保证在整个页面中不重名，命名上应与函数所实现的功能有直接关系。
- 函数的参数列表用于定义接收参数值的变量，各参数变量间用逗号作为分隔符。若函数不需要接收参数，可不定义参数列表。JavaScript的参数采用传值方式进行传递。
- return语句后的表达式为函数所要返回的值，若函数结束时没有使用return语句，则函数返回一个undefined类型的值。
- 整个函数体用一对花括号括起来。
- 通常在<head>与</head>部分定义要用到的JavaScript函数。

[**例3.1**] 试定义一个名为cuberoot的函数，以实现求一个数的立方根。

分析：求立方根的数可作为函数的参数传递给函数体计算。计算后的结果通过return语句返回。因此，该函数应定义一个用于接收参数的变量。

cuberoot函数的定义方法为：

```
<Script Language="JavaScript">
    function cuberoot(num) {
        return num* (1/3)
    }
</Script>
```

由于该函数的功能比较简单，要计算的数可直接放在return语句之后作为一个表达式来计算，然后再利用return语句将计算结果返回。

(2) 函数的调用方法

定义一个函数，仅是告知浏览器有这样一个函数，函数体中的语句并不会被执行，只有在调用该函数时，函数体中的语句才真正被执行。其调用方法为：

调用格式1：

```
varname=函数名(参数值)
```

调用格式2：

```
函数名(参数值)
```

说明：若函数调用后有返回值，而且需要保存该返回值，则采用格式1的调用方法；若不需要保存函数的返回值，或者需要直接使用函数的返回值，或者函数仅是实现某项

特殊的功能，没有明确的返回值时，通常采用格式 2 来调用。

定义了求立方根的函数后，当需要求立方根时，就可直接通过调用该函数来实现。比如，若要计算 27 的立方根，则调用方法为：


```
<Script Language="JavaScript">
    window.alert("27 的立方根为："+cuberoot(27));
</Script>
```

4. 变量的作用域

作用域是指变量的存活范围或变量的有效范围。在 JavaScript 中，变量的作用域分为内部(局部)变量的作用域和外部变量的作用域。

在函数中所定义的变量即为内部变量，其作用域仅限于在定义它的函数体内。在函数之外所定义的变量即为外部变量，作用域为定义它的页面，可供该页面的各个脚本块和各个函数过程所引用，相当于一个页面级的全局变量。

当一个外部变量的名字与内部变量的名字相同时，在函数体中，有效的是内部变量，同名的外部变量被自动屏蔽。在函数内部改变一个外部变量的值后，会使外部变量的值发生实际的改变。有关变量作用域的示例如下所示：

```
<Script Language="JavaScript">
    var x=13,y=29;                                    //外部变量
    function test(){
        var num,y=10;                                 //内部变量
        num=x+y;                        //函数体中可访问外部变量 x,此处的 y 为内部变量的 y
        x++;
        window.alert("num 的值为："+ num);            //输出 num 的值
    }
    test();                             //调用 test 函数,输出的 num 值为：23
    window.alert("x 的值为："+x+" y 的值为："+y);      //输出的 x 值为 14,y 为 29
    /*执行以下语句时,状态行将显示"完成,但网页上有错误"的提示信息,这是因为在此处
    num 变量已不存在了。*/
    window.alert("num 的值为："+num);
</Script>
```

5. 条件分支语句

程序代码的执行是按代码书写的先后顺序来执行的，在实际应用中，常需要根据条件的成立与否来选择执行不同的代码，以实现智能化的处理。这种能控制程序执行流向的语句，通常称为控制语句。流程控制语句包括条件分支语句和循环控制语句两大类。

(1) if...else 分支语句

语句用法：

```
if(条件表达式){
    语句体 1;
}
else{
    语句体 2;
}
```

语句功能：首先计算并判断条件表达式的值，若为 true，则执行语句体 1；若为 false，则执行语句体 2。

条件分支语句实现了从两组语句中根据条件的成立与否，选择其中一组来执行的智能化过程。其中 else 子句为可选项，若默认 else 子句，则分支语句可简化为如下形式：

```
if(条件表达式){
    语句体;
}
```

该语句所包含的语句体只有在条件成立时才会被执行。

[例 3.2] 在利用表单向服务器提交数据时，应先判断表单数据的有效性，以提高容错能力和减轻服务器的负担。现有一实现用户登录的表单，表单名为 Login，表单数据提交给 userlogin.asp。表单中用于输入用户名的文本框对象名为 username，用于输入密码的输入框对象名为 passwd。试利用 JavaScript 编程，实现对用户提交数据的有效性校验。

分析：可将表单的提交按钮改为普通按钮，并通过为该按钮指定 OnClick 事件处理函数来实现有效性验证。验证通过后，再调用表单对象的提交方法 submit()来实现表单数据的提交。

实现该要求的网页源代码为：

```
<html>
<head>
<title> 表单数据有效性验证</title>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<Script language="JavaScript">
    function checkform(){
        if(document.login.username.value==""){
            alert("请输入用户名!");
            document.login.username.focus();
            return false;
        }
        if(document.login.passwd.value==""){
            alert("请输入密码!");
            document.login.passwd.focus();
            return false;
        }
        document.login.submit();      //提交表单数据
    }
</Script>
</head>
<body>
<div align=center><h3> 用户登录</h3></div>
<form name="login" action="userlogin.asp" method="post">
<table align=center width="247">
  <tr><td width="100" height="26"> 用户名：</td>
  <td width="193" height="26"><input type="Text" name="username" size=10>
  </td></tr>
```

```
  <tr><td width="100" height="26"> 密 码：</td><td width="193" height="26">
  <Input type="password" Name="passwd" size=15 MaxLength=15></td></tr>
  <tr align=center><td colspan=2 height="26"><br>
  <Input type="button" Value=" 登 录 " OnClick="checkform();"></td></tr>
</table>
</form>
</body>
</html>
```

表单数据的有效性验证通过后，将提交给 userlogin.asp 页面作进一步处理。对表单数据在提交前进行有效性验证很有必要。比如用户注册页面，电子邮件地址的有效性验证，一些要求用户必须填写的项目等，都必须在表单提交前进行有效性验证。

(2) switch 语句

switch 语句可以根据给定表达式的取值不同选择执行不同的语句，常用于实现具有多种可能情况的判断处理。语句用法如下：

```
switch(表达式){
    case 值 1:
        语句组 1;
    case 值 2:
        语句组 2;
    ...
    case 值 n:
        语句组 n;
    default:
        语句组;
}
```

语句功能：首先计算表达式的值，然后与 case 后面给定的值进行比较，与哪一个相等，就执行该 case 后面的语句组，遇到 break 语句后，就结束 switch 语句的执行。若在执行过程中未遇到 break 语句，则将继续执行后面 case 下面的语句组，直到遇到 break 语句为止。若表达式的值与各个 case 后面给定的值均不相等，则无条件执行 default 后面的语句组；若无 default 则什么也不执行，default 为可选项。

[**例 3.3**] 试用 JavaScript 编程，在页面中输出当前的星期数，若为星期六或星期日，则用红色输出。提示：星期数可利用 JavaScript 内置的 Date 对象的 getDay()方法来获得。

分析：getDay 方法以 0～6 的数字来返回星期数，其中 0 代表星期天，1 代表星期一，6 代表星期六。为了能将数字转换为汉字式的星期数，可利用 switch 语句分别判断，并输出相应的星期数。

实现的 JavaScript 代码为：

```
<Script Language="JavaScript">
    var curday=new Date();          //创建 Date 对象的一个实例，并命名为 curday
    switch(curday.getDay()){        //利用 curday 对象的 getDay()方法获得星期数
        case 1:
```

```
            document.write("星期一");break;
        case 2:
            document.write("星期二");break;
        case 3:
            document.write("星期三");break;
        case 4:
            document.write("星期四");break;
        case 5:
            document.write("星期五");break;
        case 6:
            document.write("<font color='FF0000'> 星期六</font> ");break;
        case 0:
            document.write("<font color='FF0000'> 星期日</font> ");break;
    }
</Script>
```

将以上 JavaScript 代码放在网页的＜body＞与＜/body＞之间，然后运行网页，即可看到运行结果。

6. 循环控制语句

循环控制语句可根据给定条件，循环执行语句体多次，以实现一些特殊需求。JavaScript 常用的循环语句有 for、while 和 do... while 三种。

(1) for 循环

语句用法：

```
for(变量初始化部分;循环条件表达式;变量更新部分){
    循环执行的语句体;
}
```

语句说明

① 变量初始化部分：通常用于给循环控制变量赋初值，为可选项。

② 循环条件表达式：用于指定循环的条件，为可选项。若表达式值为 true，则将继续执行语句体；若为 false，则结束循环的执行。

③ 变量更新部分：用于更改循环控制变量的值，以使循环趋于结束，为可选项。

[**例 3.4**] 试用 JavaScript 编程，计算并输出 1～100 间偶数的累加和。

实现的 JavaScript 代码为：

```
<Script Language="JavaScript">
    var s= 0;
    for(var n=2;n<=100;n+=2){
        s+=n;                //等价于：s=s+n;
    }
    document.write("1~100 间偶数的累加和为："+s);
</Script>
```

[**例 3.5**] 试用 JavaScript 编程，分别用＜H1＞至＜H6＞的字体输出字符串