第 3 章　VBScript 脚本语言

本章学习目标

本章主要介绍 VBScript 脚本语言的基础知识。通过本章的学习，读者应该掌握以下内容:

- VBScript 服务器端脚本和客户端脚本语言的概念
- VBScript 的基本特点
- VBScript 的基本数据类型、运算符和表达式
- VBScript 条件控制语句、函数和子过程
- VBScript 对象和事件处理

3.1　VBScript 脚本语言概述

3.1.1　服务器端脚本和客户端脚本

制作网页时，可以使用 HTML 标记来组织 Web 页面上的静态信息，例如，显示文本、制作表格、加入多媒体与超级链接、制作表单及生成框架结构等。但是，HTML 是一种简单的语言，它很难满足用户和 Web 页面之间实现交互功能的需要。

VBScript 是 Microsoft Visual Basic Scripting Edition 的简称，是一种 Script 脚本语言。把它嵌入到 HTML 中，可以实现制作动态交互页面的要求。脚本语言是介于 HTML 和 Java、Visual Basic 等编程语言之间的语言，其最大优点是语言编写简单，可以使用任何文本编辑器编写，只要保存为纯 ASCII 文本文件即可。

目前比较流行的脚本语言有两种：VBScript 和 JavaScript。其中 VBScript 基于 Microsoft 公司的 Visual Basic 语言；而 JavaScript 基于 SUN 公司的 Java 语言。

使用 VBScript 和 JavaScript，既可以编写服务器端脚本，也可以编写客户端脚本。服务器端脚本和客户端脚本的主要区别是：

- 服务器端脚本在 Web 服务器上执行，由服务器根据脚本的执行结果生成相应的 HTML 页面并发送到客户端浏览器中显示。只有服务器端脚本才能真正地实现“动态网页”。服务器端脚本的执行不受浏览器的限制，脚本在网页通过网络传送给浏览器之前被执行，Web 浏览器收到的只是标准的 HTML 文件。
- 客户端脚本由浏览器解释执行。由于客户端脚本随着 HTML 页面下载到客户端浏览器，在用户本地执行，因此其执行速度明显快于服务器端脚本。客户端脚本常用于做简单的客户端验证（例如用户名非空验证）或实现网页特效等。

3.1.2　VBScript 脚本语言编程实例

1. 服务器端脚本

在 ASP 中编写服务器端脚本的方法有两种：一是使用分隔符<%和%>将脚本括起来，如例 3-1-1 中是使用分隔符<%和%>编写的服务器端脚本程序；二是使用<SCRIPT></SCRIPT>标记，并在其中用 RUNAT=Server 表示脚本在服务器端执行，如例 3-1-2 中是使用<SCRIPT></SCRIPT>标记完成的操作。

例 3-1-1：

```
<%@ LANGUAGE = "VBScript" %>
<HTML>
        <BODY>
            <FONT SIZE=7>
            <% Response.Write "欢迎使用 VBScript 脚本语言！"%>
            </FONT>
        </BODY>
</HTML>
```

程序的第一行通过 LANGUAGE 来指明本程序使用的脚本语言类型，通过 Response.Write 在客户端的浏览器上显示输出指定信息，运行结果如图 3-1-1 所示。

图 3-1-1　使用<%和%>进行服务器端编程

例 3-1-2：

```
<HTML>
<HEAD>
   <SCRIPT LANGUAGE=VBScript RUNAT=Server>
      Sub Welcome
         For i = 1 To 2
           Response.Write"<font size=7>欢迎使用 VBScript 脚本语言!</font size><BR>"
         Next
      End Sub
   </SCRIPT>
</HEAD>
<BODY>
    <% Call Welcome %>
</BODY>
</HTML>
```

在这个例子中定义了一个过程 Welcome，循环显示多个欢迎字符串，然后在程序主体中使用 Call Welcome 语句调用这个过程，在浏览器中显示结果如图 3-1-2 所示。

图 3-1-2　使用<SCRIPT>和</SCRIPT>进行服务器端编程

在进行服务器端编程时，可以使用<%和%>把 VBScript 脚本语言嵌入到 HTML 语言中，也可以使用<SCRIPT>和</SCRIPT>把 VBScript 脚本编写成可以完成某种特殊功能的过程或函数，在程序需要的位置进行调用，使得程序的结构更加清晰。

注意：VBScript 的用户界面元素（如：MsgBox 和 InputBox）不能在服务器端脚本中使用。MsgBox 用于显示一个信息框，而 InputBox 用于产生一个输入框，它们只能在客户端执行。另外，在服务器端的脚本中，也不能使用 VBScript 函数 CreateObject 和 GetObject，而要使用 Server.CreateObject，用 CreateObject 或 GetObject 创建的对象不能访问 ASP 内建对象，也不能参与事务处理。

2. 客户端脚本

在动态网页设计中，必须把客户端脚本的代码写在<SCRIPT>和</SCRIPT>标记之间，并将其嵌入到 HTML 页面中去。

一般来说，脚本代码可以放在 HTML 文档的任何地方，常见的位置是将脚本代码放在 HTML 文档的<HEAD></HEAD>标记中。

脚本代码以<SCRIPT>开头，以</SCRIPT>结束，其一般形式如下：

```
<SCRIPT  LANGUAGE="language" [EVENT="event"] [FOR="object"]>
<!--
    脚本代码
-->
</SCRIPT>
```

<SCRIPT>标记主要有 3 个属性，它们的意义如下：

- LANGUAGE：指定使用哪一种脚本语言，不同的浏览器支持的脚本语言是不一样的，例如 Microsoft Internet Explorer 可以解释并执行 VBScript 语言和 JavaScript 语言，而 Netscape 只支持 JavaScript 语言。
- EVENT：指定与此段脚本相关联的事件。
- FOR：指定与事件相关联的对象。

另外还有可选的<!--和-->（注释标记）。将客户端脚本放在注释标记中，当遇到不支持 VBScript 脚本语言的浏览器时，这段代码可以不做任何处理，否则会将脚本代码显示在浏览器中，影响网页的显示效果。建议在进行客户端编程时，为脚本语言增加注释语句。

例 3-1-3：

```
<HTML>
<HEAD><TITLE>客户端脚本编写举例程序</TITLE>
```

```
<SCRIPT  LANGUAGE=VBScript EVENT="OnClick" FOR="Button1">
<!--
Dim frmTmp '声明一个变量
Set frmTmp=Document.Form1
If IsNumeric(frmTmp.Text1.Value) Then
  If  frmTmp.Text1.Value<10 or frmTmp.Text1.Value>30  Then
     MsgBox "请输入 10 到 30 之间的数字!"
  Else
     FrmTmp.Submit '输入正确，传递到服务器
  End If
Else
  MsgBox"请输入数字!"
End if
-->
</SCRIPT>
</HEAD>
<BODY bgColor="White">
   <H2>请输入 10 到 30 之间的数字：</H2><HR>
   <FORM NAME="Form1" >
      <INPUT NAME="Text1" TYPE="TEXT">
      <INPUT NAME="Button1" TYPE="BUTTON"VALUE="提交">
   </FORM>
</BODY></HTML>
```

将上述程序保存后，在浏览器中打开运行时，显示结果如图 3-1-3（a）所示，在文本框中输入数字后，单击“提交”按钮，浏览器会查找并执行 Button1 按钮的 OnClick 事件。如果输入正确，将调用表单的 Submit 方法将数据提交到服务器。若输入不正确，例如输入的数据包含字符，则会调用 MsgBox 语句产生信息框来提示应该输入数字（图 3-1-3（b））；如果输入的数字小于 10 或大于 30，则会提示输入数字的范围（如图 3-1-3（c））。

（a）　（b）　（c）

图 3-1-3　使用 VBScript 进行客户端编程

为了使程序更加结构化，可以将完成特定功能的代码编写成一个过程或函数，例如在例 3-1-3 中可以把事件处理语句编写成一个过程，采用“对象名_事件名”的格式指定某个对象的事件代码，现将例 3-1-3 中的<SCRIPT>和</SCRIPT>中的脚本部分改写为如下形式：

```
<SCRIPT  LANGUAGE=VBScript>
<!--
```

```
Sub Button1_OnClick
Dim frmTmp '声明一个变量
Set frmTmp=Document.Form1
If IsNumeric(frmTmp.Text1.Value) Then
  If  frmTmp.Text1.Value<10 or frmTmp.Text1.Value>30  Then
     MsgBox "请输入 10 到 30 之间的数字!"
  Else
     FrmTmp.Submit '输入正确，传递到服务器
  End If
Else
  MsgBox"请输入数字!"
End if
End Sub
-->
</SCRIPT>
```

用这段代码替换程序中<SCRIPT>和</SCRIPT>部分的代码，其他的代码不变。程序修改完成后，在浏览器中再次运行时，显示的结果和修改前相同。

3.1.3 VBScript 和 JavaScript

VBScript 和 JavaScript 都是脚本语言，虽然它们不属于同一个公司，但是同为 Scripting 脚本语言，有很多性能都比较相近，性能比较如表 3-1-1 所示。另外，可以在同一个 HTML 文档中同时使用这两种语言。

表 3-1-1 VBScript 和 JavaScript 性能比较

相关性能	VBScript	JavaScript
程序格式	嵌入到 HTML 中	嵌入到 HTML 中
数据类型	采用复合的单一变量类型，使用 DIM 定义后使用	采用松散数据类型，变量不用定义可直接使用
对象概念	无法自定义对象类型，使用系统对象，无类和继承的概念，可定义函数过程和子程序过程	无法自定义对象类型，使用系统对象，无类和继承的概念，只能定义函数
执行方式	有浏览器内部虚拟机处理	有浏览器内部虚拟机处理
安全性	安全性高，严禁写入磁盘	安全性高，严禁写入磁盘

在 ASP 中的缺省语言是 VBScript。脚本语言的设置方法主要有：

- 在 IIS 中设定。
- 在网页中指定脚本语言，可以利用如下格式设置脚本语言为 VBScript：

```
<% @LANGUAGE=VBScript%>
```

注意：在%、@之间有空格，并且该语句要在任何一个命令之前使用。

- 利用<Script>设定脚本语言，例如：

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
```

本章主要介绍 VBScript 脚本语言的使用方法。

3.2　基本数据类型及输入输出

3.2.1　将单行语句分成多行

在编写程序代码时，有的语句可能会很长，为了在阅读和对程序查错时直观、方便，可使用续行符“-”（由一个空格和一个下划线组成），将长的语句分成多行书写。

例 3-2-1：

```
<HTML>
<HEAD>
<TITLE>将单行语句分成多行程序举例</TITLE>
<SCRIPT LANGUAGE="VBScript">
Sub ShowMessage
  strExmp="欢迎您光临本网站，"& _
          "您可以浏览最新图书信息，"& _
          "如果您有什么意见和建议，请来信！"
  MsgBox strExmp
End Sub
Call ShowMessage
</SCRIPT></HEAD>
<BODY>
……
</BODY>
</HTML>
```

上面程序代码中的&号用于将两个字符串连接成一个字符串。例 3-2-1 中首先定义了一个 ShowMessage 过程，该过程完成在客户端浏览器显示一个信息框的功能，然后调用了该过程。运行结果如图 3-2-1 所示。

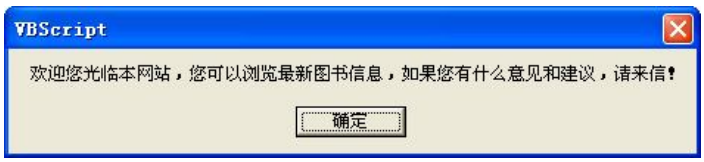


图 3-2-1　将单行语句分成多行书写

3.2.2　在代码中加注释

为了方便以后进行程序维护，能够很容易地读懂程序，需要在程序中增加适当的注释。注释语句是非执行语句，它不被系统解释和编译，也不在程序执行结果中显示出来，可通过在编辑器中打开程序的源代码来查看。

注释语句可以有两种表现形式，一种是通过使用 Rem 语句，Rem 后的任何文本都会被认为是对程序的注释，不会被处理；另一种是采用西文单引号“'”，即以撇号作为注释的开始，注释可以和语句在同一行并写在语句的后面，也可以单独占一行。

注释的一般格式为：

格式一：

```
Rem 注释内容
```

格式二：

```
' 注释内容
```

或

```
语句' 注释内容
```

3.2.3 使用不同进制的数字

在 VBScript 中，除了可以使用默认的十进制来表示数字外，还允许使用十六进制或八进制来表示数字。对于不同进制的数，VBScript 在表达方式上有明确规定，即十六进制数要加前缀＆H（如＆H9），八进制数要加前缀＆0（数字零）或&O（字母 O）（如＆011 或&O11），十进制数不用加任何前缀。

3.2.4 数据类型及其子类型

严格的说，VBScript 只有一种特殊的数据类型，即变体（Variant）数据类型，它可以随着变量被使用方式的不同而包含不同的数据信息，会根据不同的应用环境而将不同地方的数据变量区别对待，例如，在数字上下文中会把它作数字信息处理，而在字符串上下文中，会当作字符串处理，也可以在数字中加上引号强迫它成为字符串数据。由变体类型引申出来的类型称为子类型，表 3-2-1 中介绍了几种常用的子类型。因为 VBScript 只有一种数据类型，所以所有 VBScript 的过程和函数的返回值都是变体数据类型。

表 3-2-1 VBScript 数据类型的子类型

数据类型	备注
Byte（字节）	以一个字节的无符号二进制数存储，其取值范围是 0～255
String（字符串）	由 ASCII 字符组成的变长度字符串，长度范围从 0～2^{31} 个字符，由双引号作为定界符
Integer（整型）	一般整数变量，用 2 个字节（16 位）的二进制码表示，范围从-32768～32767 之间
Long（长整型）	用 4 个字节（32 位）的二进制码表示，范围从-2147483648～2147483647 之间
Single（单精度浮点数）	用 4 个字节（32 位）的二进制码表示，可精确到 7 位十进制数。负数范围从 -3.402823E38～-1.401298E-45，正数范围从 1.401298E-45～3.402823E38
Double（双精度浮点数）	用 8 个字节（64 位）的二进制码表示，可精确到 15 或 16 位十进制数。负数范围从 -1.79769313486232E308 ～ -4.94065645841247E-324 ， 正数范围从 4.94065645841247E-324～1.79769313486232E308
Currency（货币）	支持小数点右面 4 位和左面 15 位，是一个精确的定点类型，适用于货币计算。取值范围从-922337203685477.5808～922337203685477.5807
Boolean（布尔型）	用 2 个字节存储，包含逻辑值，取值只能为 Ture 或 False
Date/Time（日期/时间）	表示日期和时间值，日期值的有效范围从公元 100 年 1 月 1 日～公元 9999 年 12 月 31 日，时间值从 00:00:00～23:59:59。AM 表示上午，PM 表示下午。在代码中要用日期和时间值，必须用一对#将其括起来，如#10:28:36AM#

续表

数据类型	备注
Empty（空类型）	没有初始化的变量，数值变量值为 0，字符串变量值为零长度字符串（“”），可以用 IsEmpty()函数来测试变量是否已被初始化
Null（空值）	不包含任何有效数据的变量，可以使用 IsNull()函数来测试表达式是否不含任何有效的数据
Object（对象）	用 4 个字节存储，用来表示引用程序所能识别的任何对象
Error（错误）	保存程序产生的错误代码

注意：任何变体数据类型变量经过声明后如果未指定值，则其内含值为未定义（Empty）。这个未定义值与空值（Null）是不同的，Null 代表无效的数据，Empty 的变量在使用时是值为 0 或者为空的字符串，而 Null 的变量必须为其赋上初值才能使用。

3.2.5　变量

VBScript 中的变量实际上是在计算机内存中预留的用于存储数据的内存区。使用变量时，用户并不需要关心变量在计算机的内存中是如何存储的，只需要引用变量名来查看或改变变量的值就可以了。VBScript 中的变量不区分大小写，常用来存储在程序运行过程中需要用到的数据。

1. 变量命名规则

变量用变量名来区分。在 VBScript 中，变量命名必须满足以下条件：

- 变量的名字必须以字母开头。
- 名字中不能含有句号。
- 名字不能超过 255 个字符。
- 名字不能和 VBScript 中的关键字同名。
- 变量名在被声明的作用域内必须惟一。

为了提高程序的可读性，在给变量命名时，应尽量使变量名称含义清楚，便于记忆，另外，最好能在变量的名称中表示出变量的子类型及变量中所存放数据的信息。

2. 声明变量

VBScript 声明变量时有两种不同的方式，一种是不用声明变量，直接使用，称为隐式声明；另一种方式是像其他语言一样先声明变量后使用，称为显式声明。

（1）隐式声明方式。由于在 VBScript 中只有一种数据类型，即变体类型，因此严格的说，在 VBScript 中使用一个变量前并不需要声明，而可以直接在脚本代码中使用。在程序运行过程中检查到这种变量时，系统会自动地在内存中开辟存储区域登记变量名。例如，隐式声明一个变量 studentAge，变量值为 22。

```
studentAge=22
```

这种变量声明方式的优点是简单方便，可以在需要时随意声明变量。但是如果程序出现错误，例如在使用 studentAge 变量的时候，拼错了变量的名字，拼成 studentA，运行时会认为 studentA 是一个新的变量，不能得到预期的效果，而且在调试过程中，追踪和发现变量错误也会变得很困难，因为并没有标记变量在什么位置进行定义的。为了避免隐式声明变量时写错变

量名引起的问题，VBScript 提供了 Option Explicit 语句来强制显式声明变量。如果在程序中使用该语句，则所有变量必须先声明，然后才能使用，否则会出错。强制声明会增加代码量，但可以提高程序的可读性，减少出错的机会。Option Explicit 语句必须位于 ASP 处理命令之后、任何 HTML 文本或脚本命令之前，如例 3-2-2 所示。

例 3-2-2：

```
<% @ LANGUAGE=VBScript %>
<% Option  Explicit
  Dim  studentAge
  Dim  studentA
%>
```

注意：Option Explicit 语句只影响用 VBScript 编写的脚本代码，对其他脚本代码不起作用。

（2）显式声明方式。使用变量声明语句来声明变量的方式。变量声明语句有 Dim 语句、Public 语句和 Private 语句。显式声明可以在定义变量的时候为变量在内存中预留空间，登记变量名。声明多个变量时，可以在同一条声明语句中，用逗号将多个变量分开。例如：

```
Dim  studentAge
Dim  teacherAge，workerAge
studentAge=22
teacherAge=35
workerAge=40
```

以上代码声明了三个变量：studentAge，teacherAge，workerAge，并为三个变量都赋了初值，在声明这三个变量时没有指明具体的类型，但在设计程序时，所声明的变量一般都用来存放某种子类型的数据，为了区分不同类型的变量，可以通过变量名的前缀来指明该变量的子类型，例如：

```
Dim intSum
```

变量 intSum 是一个 Integer 子类型的变量，除此之外还有其他前缀用来声明不同子类型的变量，具体子类型定义前缀如表 3-2-2 所示。

表 3-2-2　子类型前缀

子类型	前缀	示例	子类型	前缀	示例
Integer	Int	IntYear	Long	Lng	LngNumber
Currency	Cur	CurMoney	Single	Sng	SngSalary
Double	Dbl	DblPopulation	Byte	Byt	BytCharacter
Boolean	Bln	BlnRetired	String	Str	StrName
Date(Time)	Dtm	DtmSystem	Object	Obj	ObjTemp

表 3-2-2 中的前缀只是一种定义变量的约定，不是语言本身的规定，在设计过程中使用可以提高程序的可读性，减少错误，当然也可以不使用。

Public 和 Private 语句声明变量的格式和 Dim 语句是一样的。例如：

```
Public varName1，varName2
Private studentAge，studentName
```

- Public 语句用来声明全局变量，这些变量可以在网页页面中的所有脚本和所有的过程中使用。

- Private 语句用来声明私有变量，这些变量只能在声明它的脚本中使用，即在声明它们的<Script>和</Script>标记中间使用。Public 和 Private 语句声明变量时都必须在过程之前的脚本级使用，控制变量的作用范围。

3. 变量的作用域

变量的作用域指的是变量的有效范围，因为变量被声明后不是在任何地方都可以被使用的，在作用域内可以使用变量，在作用域外变量则不可见。

在 VBScript 中，变量的作用域分为过程内有效和整个程序中都有效。在变量过程内部声明的变量称为过程级变量或局部变量，这样的变量只有在声明它们的过程中才能使用，即无法在过程外部访问；过程外部声明的变量称为脚本级变量或全局变量，即在同一个.asp 文件中的任何脚本命令均可访问和修改该变量的值。

例 3-2-3：

```
<%Option Explicit
 Dim intX '声明脚本级变量
 intX=1 '给脚本级变量赋值
 SetLocaIVariable '调用过程修改过程级变量的值
 Response.Write intX '将脚本级变量的值发送到浏览器，值仍为 1
 Sub SetLocalVariable
    Dim  intX '声明过程级变量
    IntX=2 '给过程级变量赋值
 End  Sub  %>
```

如果在过程中没有声明变量，直接使用变量，有可能在无意中修改脚本级变量的值。例如，在下面的例子中，由于没有在过程中声明变量，因此当过程调用设置变量 intX 为 2 时，脚本引擎认为过程要修改的是脚本级变量。

例 3-2-4：

```
<%   Option  Explicit
 Dim  intX '声明脚本级变量
 IntX=1 '给脚本级变量赋值
 SetLocalVariable '调用过程修改变量的值
 Response.Write intX '将脚本级变量的值发送到浏览器，值为 2
 Sub SetLocalVariable
    IntX=2 '给脚本级变量赋值
 End Sub  %>
```

为了避免这样的问题，应该养成显式声明所有变量的习惯。这一点当使用#include 命令在.asp 文件中包含其他文件时尤其重要，因为被包含的脚本虽然在单独的文件中，但却当作是包含文件的一部分。除非声明变量，否则很容易忘记，必须在主脚本和被包含脚本中使用不同的变量名。

注意：脚本级变量只能在单个.asp 文件内访问。如果要从文件的外部访问变量，则必须提供变量的 Session 或 Application 作用域。关于 Session 和 Application 的内容请参阅后面的章节。

3.2.6　常量

常量是具有一定含义的名称，用于代替数值或字符串。在程序执行期间，常量的值不会发生改变。可以在代码的任何位置使用常量代替实际值。VBScript 中的常量分为两种，即文

字常量和符号常量。

1. 文字常量

（1）字符串常量：用双引号作为定界符，由 ASCII 码字符组成（除双引号和回车符外），长度不能超过 20 亿个字符。例如："中华人民共和国"、"1233.45"等。

（2）数值常量：包括整型数、长整型数和浮点数。例如：0、300、−4125、&H85（表示十六进制数 85）、&O226（表示八进制数 226）、1.23E8、3.54E−5 等。

（3）日期时间型常量：用#号括起来。例如：#2001-3-15#、#2003-4-20 8:38:25 AM#等。

2. 符号常量

在 VBScript 中，可以通过关键字 Const 定义符号常量。例如：

```
Const PI=3.1415926
Const My_Address="北华航天工业学院计算机系计算机软件教研室"
```

另外，VBScript 本身也定义了许多固有常量，如表 3-2-3 所示。

表 3-2-3 数据类型的子类型

常量名称	常量含义
True	表示布尔真值
False	表示布尔假值
Null	表示空值
Empty	表示没有初始化之前的值
vbCr	表示回车
vbCrLf	表示回车/换行
vbTab	表示制表符

3.2.7 数组

在编程过程中，如果要成批处理数据，可以使用数组。数组是具有相同名字的一组变量，数组中包含多个元素，由不同的下标值来区分数组的各个元素。

VBScript 中的数组有以下几个特点：

- 使用数组之前要先进行定义，然后才能使用。通常用 Dim 语句来定义数组。
- 数组下标的下界一律从 0 开始。
- 一个数组中可以含有各种子类型的数据元素。

在 VBScript 中，数组分为两种类型，即静态数组和动态数组。

1. 静态数组

静态数组可分为一维数组、二维数组或多维数组。数组的维数和大小由数组名之后紧跟的括号中的数字的个数和数值的大小来决定。静态数组在编译时开辟内存区，因此它的大小在运行时是不可以改变的。例如，定义一个一维数组 arrStudent(3)：

```
Dim arrStudent(3)
```

其中 arrStudent 是数组名，数组的下界为 0，上界为 3，数组元素从 arrStudent(0)到 arrStudent(3)，共有 4 个元素。定义一个二维数组 arrTwoDim(2,3)：

```
Dim arrTwoDim(2,3)
```

上面定义的二维数组包含有 12 个元素，分别为：

arrTwoDim(0,0)，arrTwoDim(0,1)，arrTwoDim(0,2)，arrTwoDim(0,3)，arrTwoDim(1,0)，arrTwoDim(1,1)，arrTwoDim(1,2)，arrTwoDim(1,3)，arrTwoDim(2,0)，arrTwoDim(2,1)，arrTwoDim(2,2)，arrTwoDim(2,3)。

定义一个三维数组 arrThreeDim(2,3,2)：

```
Dim arrThreeDim(2,3,2)
```

数组元素从 arrThreeDim(0,0,0)到 arrThreeDim(2,3,2)，共有 36 个元素。

给数组赋值的方法是给数组中的各个元素分别赋值，如同每一个数组元素是一个独立的变量。例如，为前面定义的一维数组赋初值：

```
ArrStudent(0)="9952101"
ArrStudent(1)="王大海"
ArrStudent(2)=19
ArrStudent(3)=#3-15-86#
```

由以上的赋值操作可以看出，在 VBScript 中，一个数组中的不同元素可以赋给不同数据类型的数据。

2. *动态数组*

动态数组是运行时大小可变的数组。当程序没有运行时，动态数组不占内存，在程序运行时才为其开辟内存区。

动态数组的定义一般分两步：首先用 Dim 语句声明一个括号内不包含下标的数组，然后在使用数组之前用 ReDim 语句根据实际需要重新定义下标值。也可以用 ReDim 语句直接定义数组。

ReDim 语句的格式为：

```
ReDim [Preserve] 变量(下标)
```

例如，定义动态数组 arrStudent：

```
Dim arrStudent()
```

在使用 arrStudent 之前，必须用 ReDim 语句分配实际的元素个数。例如：

```
ReDim arrStudent(10)
```

可以用 ReDim 语句不断地改变元素数目。例如：

```
Dim arrStudent()
ReDim  arrStudent(4) '声明含 5 个元素的数组，下标从 0 到 4
 ……
ReDim  arrStudent（6）'声明含 7 个元素的数组，下标从 0 到 6
```

每次执行 ReDim 语句时，存储在数组中的当前值都会全部丢失。如果希望改变数组大小而又不丢失数组中的数据，则要使用关键字 Preserve。例如：

```
ReDim  Preserve  arrStudent(Ubound(arrStudent)+1)
```

以上代码将数组扩大一个元素，现有元素值不变，Ubound()函数返回数组的上界。

注意：*用 ReDim 语句重新定义数组时，只能改变数组元素的个数，不能改变数组的维数。*

3.2.8　基本输入输出

在 VBScript 中提供了两种非常方便的输入输出方法，即用来输出消息的消息对话框

（Message Box）和用来接收用户输入数据的数据输入对话框（Input Box）。

1. 消息对话框

用户对网页进行操作时，经常会显示一些提示信息，在这种情况下，可以使用消息对话框来通知用户。消息对话框的使用是很方便的，它的一般格式如下：

```
MsgBox(prompt[,buttons] [,title] [,helpfile,context])
```

说明：

- prompt 是一个字符串表达式，作为显示在对话框中的消息，它是必需的。如果需要显示多行信息，可以在每一行之间用回车符(Chr(13))、换行符(Chr(10))或是回车与换行符的组合(Chr(13) & Chr(10))将各行分开。
- buttons 是一个数值表达式，用来指定显示按钮的数目及形式、使用的图标样式、缺省按钮是什么等。buttons 是可选的，如果省略，则缺省值为 0。buttons 参数常用的设置值如表 3-2-4 所示。

表 3-2-4 buttons 参数常用的设置值

常数	值	描述
vbOKOnly	0	只显示 OK 按钮
VbOKCancel	1	显示 OK 及 Cancel 按钮
VbAbortRetryIgnore	2	显示 Abort，Retry 及 Ignore 按钮
VbYesNoCancel	3	显示 Yes，No 及 Cancel 钮
VbYesNo	4	显示 Yes 及 No 按钮
VbRetryCancel	5	显示 Retry 及 Cancel 按钮
VbCritical	16	显示 Critical Message 图标
VbQuestion	32	显示 Warning Query 图标
VbExclamation	48	显示 Warning Message 图标
VbInformation	64	显示 Information Message 图标
vbDefaultButton1	0	第一个按钮是缺省值
vbDefaultButton2	256	第二个按钮是缺省值
vbDefaultButton3	512	第三个按钮是缺省值
vbDefaultButton4	768	第四个按钮是缺省值

其中第一组值（0～5）描述了对话框中显示的按钮的类型与数目；第二组值（16、32、48、64）描述了图标的样式；第三组值（0、256、512、768）说明哪一个按钮是缺省值。在每组值中取一个数字，将这些数字相加生成 Buttons 参数值，不同的组合会得到不同的结果。

注意：由于每一组数值都有相应的字符串常量，它们的作用和使用数值是一样的，在进行程序编写中尽量使用字符串常量，这样可以增加程序的可读性。

- title 是一个字符串表达式，它是显示在对话框标题栏中的提示信息。title 是可选的，如果省略，则将应用程序名放在标题栏中。

- helpfile 是一个字符串表达式，识别用来向对话框提供上下文相关帮助的帮助文件。helpfile 是可选的，如果提供了 helpfile，也必须提供 context。
- context 是一个数值表达式，由帮助文件的作者指定给适当的帮助主题的帮助上下文编号。context 是可选的，如果提供了 context，则也必须提供 helpfile。

出现对话框后，根据用户在对话框上选择的不同命令按钮，会返回不同的值，作为函数的返回值。表 3-2-5 给出了不同的命令按钮对应的函数的返回值。

表 3-2-5　函数返回值

常数	值	描述	常数	值	描述
vbOK	1	OK	vbIgnore	5	Ignore
vbCancel	2	Cancel	vbYes	6	Yes
vbAbort	3	Abort	vbNo	7	No
vbRetry	4	Retry			

例 3-2-5：

```
<HTML>
<HEAD><TITLE>MsgBox 使用方法</TITLE>
<SCRIPT LANGUAGE="VBScript">
<!--
Dim intResult
  intResult=MsgBox("VBScript 是很有用的，"&chr(13)&chr(10)&"你很_
想学好 VBScript 吗？",4+32,"请你选择：")
  if intResult=6 then
     MsgBox("你是个好学生，我们会尽全力的！"&chr(13)&chr(10)&"一起努力吧！")
  else
     MsgBox("你好残忍，你就这样放弃了我！")
  end if
-->
</Script>
</HEAD>
<BODY>
</BODY></HTML>
```

上面这个例子在浏览器中运行时，首先会弹出如下的消息对话框要求用户做出选择（如图 3-2-2（a）所示）：如果用户选择按钮“是”，则会出现图 3-2-2（b）所示的对话框；如果选择“否”，则会出现图 3-2-2（c）所示的对话框。

（a）选择对话框　　（b）肯定回答　　（c）否定回答

图 3-2-2　MsgBox 消息对话框的使用

2. 输入对话框

利用消息对话框可以实现和用户的交互，但是，这种交互仅仅靠几个按钮的返回值来实现，有较大的局限性，不能够很好的领悟用户的思想，而且很多时候还需要用户输入有关的数据和信息。这种情况下使用消息对话框就远远不够了，这时可以使用数据输入对话框来接受用户输入的信息和数据。

输入对话框的一般格式如下：

```
InputBox(prompt[,title] [,default] [,xpos] [,ypos] [,helpfile,context])
```

说明：

- prompt 是要显示的消息。
- title 是显示在标题栏的字符串。
- default 是一个字符串表达式，显示在文本框中，在没有其他输入时作为缺省值。default 是可选的，如果省略，则文本框为空。
- xpos 和 ypos 是数值表达式，成对出现，指定对话框在屏幕中出现的位置。

其余几个选项的意义同 MsgBox 函数。

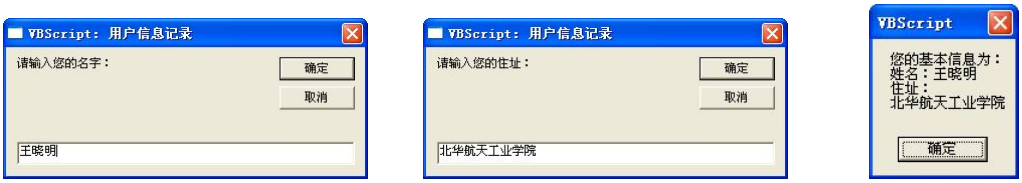
系统默认在使用 InputBox 时，使用“确认”和“取消”两个按钮，用户在输入完毕后，单击“确定”按钮，InputBox 会将输入文本框中的数据返回给程序；若单击“取消”按钮，InputBox 将会返回一个空字符串。下面的例子演示了如何使用 InputBox 函数接收用户输入的两个字符串，并将两个字符串的内容显示输出。

例 3-2-6：

```
<HTML>
   <HEAD><TITLE>InputBox 函数的使用</TITLE></HEAD>
   <BODY>
   <SCRIPT LANGUAGE="VBScript">
   <!--
      Dim strUserName,strUserAddress
      strUserName=InputBox("请输入您的名字：","用户信息记录")
      strUserAddress=InputBox("请输入您的住址：","用户信息记录")
      MsgBox("您的基本信息为："&chr(13)&chr(10)&"姓名："&strUserName_
      &chr(13)&chr(10)&"住址："&chr(13)&chr(10)&strUserAddress)
   -->
   </SCRIPT>
   </BODY>
</HTML>
```

上面的代码在浏览器中执行的结果如图 3-2-3 所示，首先出现图 3-2-3（a）中的界面要求用户输入姓名，单击“确定”后，出现图 3-2-3（b）的界面，要求用户输入住址，全部信息输入完毕单击“确定”，显示图 3-2-3（c）将用户输入的基本信息进行综合显示。

注意：每执行一次 InputBox 只能输入一个值，如果需要输入多个值，则必须多次调用，输入数据单击“确定”按钮或按回车键后，才能将输入的数据作为参数返回给一个变量，否则输入的数据不能保留。在实际编程中，InputBox 经常与循环语句、数组结合使用，这样可连续输入多个值，并将数据存储到数组当中。



（a）输入姓名　　（b）输入住址　　（c）显示输入结果

图 3-2-3　InputBox 输入对话框的使用

3.3　表达式和运算符

3.3.1　表达式简介

VBScript 中有三种不同类型的表达式：数学表达式、条件表达式和字符串表达式。数学表达式常用于常规的数值运算，运算结果仍然为数值；条件表达式常用于根据一系列事件的最后结果做出判断，并采取相应的动作，运算结果为布尔值 True 或者 False；字符串表达式用来将多个字符串连接成一个较长的字符串，运算结果仍为字符串。

VBScript 中的每一种表达式都要使用一些特殊的运算符来帮助完成其功能，运算符一般分为两种：单目运算符和双目运算符。

- 单目运算符：只有一个前置的运算符对操作数进行操作。一般格式如下：

  ```
  Operator Operand
  ```

  例如：单目减：-。
- 双目运算符：在运算符的两端各有一个操作数，即必须要有两个操作数。一般形式如下：

  ```
  Operand1 Operator Operand2
  ```

  例如：+，*，/等。

3.3.2　数学表达式及其运算符

1. 数学表达式

数学表达式用于对数据进行加工，必须要有数值和运算符。数值又称操作数，包括数字和字符串，运算符就是+，-，*，/等运算符。下面的语句都可以称为数学表达式。

```
varA=varB+varC
intResult=A-B*C+D/E
```

2. 算术运算符

VBScript 中除了常用的加、减、乘、除等算术运算符外，还提供了很多其他的算术运算符，主要有以下几种：

- 指数运算符^：完成幂运算。例如：2^3=8。
- 取负运算符-：对一个数据取它的负数。例如：-3。
- 乘法运算符*：求两个数字的乘积。例如：3*5=15。
- 除法运算符/：求两个数字的除法。例如：5/2=2.5。
- 整数除法运算符\：求两个数字的整数除法，即第一个数除以第二个数所得的整数值，不进行舍入处理。例如：5\2=2。

- 取模运算符 Mod：求两个数字的模运算，即第一个数除以第二个数所得的余数。例如：5 Mod 2=1。
- 加法运算符+：求两个数字的和。
- 减法运算符-：求两个数字的差。

以上算术运算符的优先级为从上向下逐渐降低，其中乘法和除法的优先级相同，加法和减法的优先级相同。如果想使“+”比“*”先计算，可以使用括号“()”来改变优先级的顺序。

3.3.3 条件表达式及其运算符

1. 条件表达式

条件表达式的计算结果只有两种值：True 和 False。通过计算条件表达式的值，使得条件控制语句能够进一步执行。例如：

```
If (a<10) Then
  b=b+1
End If
```

条件表达式可以通过两种布尔运算符来进行运算：关系运算符和逻辑运算符。关系运算符可以比较任意两个值之间的大小，并产生布尔值的运算结果。逻辑运算符可以将多个关系运算符进行组合，最后返回一个布尔值的运算结果。

2. 关系运算符

关系运算符用来对两个表达式的值进行大小的比较。关系运算的结果是布尔值 True 或 False。关系运算符可用于数值间的比较，也可用于字符串间的比较。当用于字符串间的比较时，将按 ASCII 码值的大小由左向右依次逐个字符进行比较，如果第 1 个字符相同，则比较第 2 个，直到比较出结果为止。

用于完成关系运算的运算符有以下几种：

- 等于运算符=：比较两个表达式是否相等。例如：3=4 的结果为 False。
- 不等于运算符<>或><：比较两个表达式是否不相等。例如：3<>4 的结果为 True。
- 小于运算符<：比较一个表达式是否小于另一个表达式。例如："abc"<"acd"的结果为 True。
- 大于运算符>：比较一个表达式是否大于另一个表达式。例如："abc">"acd"的结果为 False。
- 小于或等于运算符<=：比较一个表达式是否小于或等于另一个表达式。例如："LangFang"<="LangFang"的结果为 True。
- 大于或等于运算符>=：比较一个表达式是否大于或等于另一个表达式。例如："LangFang">="LangFang"的结果为 True。

所有关系运算符的优先顺序都相同，即按出现顺序从左到右进行运算。

注意：当在 VBScript 中进行比较的表达式是一种混合型表达式，即在表达式中，既有数学表达式、字符串表达式，又有逻辑表达式时，系统总是认为字符串的值永远比数值和布尔值大。在数值和布尔值进行比较时，和一般的数值比较相同，布尔值在 VBScript 中是以整数形式进行存储，True 为-1，而 False 为 0。数学表达式和比较表达式混用时，一般先进行数学运算然后再比较。

3. 逻辑运算符

逻辑运算符通常也称为布尔运算符，专门用于逻辑值之间的运算。

用于完成逻辑运算的运算符有以下几种：

- 取反运算符 Not：对逻辑真取反结果为逻辑假，反之为逻辑真。例如： Not (3<>4)的结果为 False。
- 逻辑与运算符 And：如果两个表达式的值都为真，结果才为真，否则结果为假。例如：(3<>4) And (4>5)的结果为 False。
- 逻辑或运算符 Or：两个表达式中只要有一个为真，结果就为真，只有两个都为假，结果才为假。例如： (3<>4) Or (4>5)的结果为 True。
- 异或运算符 Xor：如果两个表达式同时为真或同时为假，则结果为假，否则结果为真。例如：(3<>4) Xor (4>5)的结果为 True。
- 等价运算符 Eqv：是异或运算取反的结果。如果两个表达式同时为真或同时为假，则结果为真，否则结果为假。例如：(3<>4) Eqv (4>5)的结果为 False。
- 蕴含运算符 Imp：当第一个表达式为真，第二个表达式为假时，结果为假，否则结果为真。例如：(3<>4) Imp (4>5)的结果为 False。

以上逻辑运算符的优先级按从上到下的顺序逐渐降低。

3.3.4　字符串表达式

1. 字符串表达式

在进行字符串处理时，经常要把两个或者更多个字符串进行连接，形成一句完整的语句。VBScript 提供了字符串表达式。

2. 连接运算符

连接运算符是将两个字符串表达式连接起来，生成一个新的字符串。连接运算符有两个：“+”和“＆”。

使用＆运算符时，参与连接的两个表达式可以不全是字符串，即＆运算符能强制性地将两个表达式做字符串连接。

使用“+”运算符时，如果连接的两个操作数中一个是非数字字符串，另一个是数字，则会出错。例如，以下代码就会产生错误：

```
intNum=22
strTemp="学号是" + intNum
```

如果两个操作数中的一个是数字字符串，另一个是数字，则结果为两个数字相加。

例 3-3-1：

```
<HTML>
<HEAD><TITLE>连接运算程序举例</TITLE>
<SCRIPT language="VBScript">
Sub ShowMessage
  strCountry="中国"
  strCity="上海"
  intNum = 22
  strResult1 = "11" & intNum
  strResult2 = "11" + intNum
```

```
    strAddress= strCity& "是" + strCountry& "的一个城市。"
    MsgBox strResult1&" (&连接)"&chr(13)&chr(10)&_
          strResult2&" (+连接)"&chr(13)&chr(10)&_
          strAddress&" (&连接)"
End Sub
call ShowMessage
</SCRIPT></HEAD>
<BODY>
……
</BODY></HTML>
```

程序的执行结果如图 3-3-1 所示。

图 3-3-1　连接运算符的使用

3.3.5　表达式中的优先级

在进行算术运算时，经常是多种运算符或表达式混合使用，这种情况下，VBScript 将按照如下原则执行：首先执行算术运算符，其次是字符串运算符，再次执行关系运算符，最后执行逻辑运算符。但是如果表达式中含有括号，则最先执行括号里边的表达式，如果有多层括号，先计算最内层括号内的表达式的值。具体优先级关系如表 3-3-1 所示。

表 3-3-1　VBScript 中运算符的综合优先级

运算符及名称	优先级	运算符及名称	优先级	运算符及名称	优先级
()　括号	1	=　等于	9	Or　逻辑或	17
^　乘方	2	<>　不等于	10	Not　逻辑非	18
-　单目减	3	>　大于	11	Xor　逻辑异或	19
*和/　乘和除	4	<　小于	12	Eqv　逻辑等于	20
\　整除	5	>=　大于等于	13	Imp　逻辑包含	21
Mod　取余	6	<=　小于等于	14		
+和-　加和减	7	Is　对象相等	15		
&　字符串连接	8	And　逻辑与	16		

3.4　VBScript 中的控制语句

3.4.1　控制语句

在 VBScript 中编写脚本时，脚本的执行顺序一般是从上向下执行的，程序的结构属于顺序结构。但在某些情况下，也可以根据需要使用其他的控制结构，VBScript 提供了两种类型的控制语句：流程控制语句和循环控制语句。通常情况下，流程控制语句用来控制程序流程的条件转向和选择问题等，包括选择语句（If…Then…Else）和多分支选择语句（Select…Case）。循环控制语句用来编写程序中所需要的特定条件下执行过程相似的循环流程，包括 For 循环控制语句（For…Next）、Do 循环控制语句（Do…Loop）和 While 循环控制语句（While…Wend）。

3.4.2　条件控制语句

条件控制语句分简单条件控制语句和嵌套条件控制语句。

1. 简单条件控制语句

其单行结构的语法如下：

```
If 条件表达式 Then 语句体 1 [Else 语句体 2]
```

其中，条件表达式的返回值如果为 True，则执行 Then 后面的“语句体 1”，否则执行 Else 后面的“语句体 2”；如果省略 Else 部分，则执行 If 后面的语句。

例 3-4-1：

```
<HTML>
<HEAD><TITLE>选择结构程序举例 1</TITLE>
<SCRIPT LANGUAGE="VBScript">
Sub ShowMessage
   dim x,y,z
   x=2
   if x>0 then
          y=x
          z=-x
       else
          y=-x
          z=x
   end if
'以对话框的形式输出 x，y 和 z 的值
   MsgBox "x 的值为" & x &",  y 的值为" & y & ",  z 的值为" & z
End Sub
    call ShowMessage '调用 ShowMessage 过程
</SCRIPT></HEAD><BODY>
……
</BODY></HTML>
```

图 3-4-1　选择结构程序举例 1

程序的运行结果如图 3-4-1 所示。

2. 嵌套条件控制语句

嵌套条件控制语句也称块结构，它的一般语法格式如下：

```
If 条件表达式 1 Then
   [语句体 1]
[Else If 条件表达式 2 Then
   [语句体 2]]
    …
[Else
   [语句体 n]]
End  If
```

VBScript 先测试条件表达式 1，如果为 True，则执行 Then 后面的“语句体 1”，然后退出块结构，执行下面的语句。如果条件表达式 1 的结果为 False，则再测试条件表达式 2，依次类推。如果找到一个为 True 的条件时，就执行相应的语句体，然后执行 End If 后面的语句。如果条件都不为 True，且有 Else，则执行 Else 后的“语句体 n”，如果没有 Else，则直接执行 End If 后面的语句。

例 3-4-2：

```
<HTML>
   <HEAD><TITLE>选择结构程序举例 2</TITLE>
   <SCRIPT LANGUAGE="VBScript">
   Sub ShowMessage
       dim x,y
       x=InputBox("请输入 x 的值：")
     if not isnumeric(x) then
        MsgBox "输入错误，请输入数字！"
     elseif x>0 then
             y=x
         elseif x<0 then
                 y=-x
             else
                 y=0
     end if
     if isnumeric(x) then
        MsgBox "x 的值为" & x &",  y 的值为" & y '以对话框的形式输出 x 和 y 的值
     end if
End Sub
call ShowMessage '调用 ShowMessage 过程
</SCRIPT></HEAD>
<BODY>
……
</BODY></HTML>
```

程序执行的结果如图 3-4-2 所示。当 x 的值是一个正数时，y 的值等于 x 的值；当输入一个负数时，y 的值等于−x，如图 3-4-2（a）和图 3-4-2（b）所示；当输入 0 时，y 的值为 0；当输入字符串时，提示错误信息，如图 3-4-2（c）所示。

注意：在编写程序时，要使用分层缩进的书写格式，使得程序的结构一目了然，这样的好处是能够很容易的识别每一个 End If 是结束哪一层条件控制语句的。

（a）　　（b）　　（c）

图 3-4-2　选择结构程序举例 2

3. 多分支结构

可以用多分支结构来替代块结构的条件语句，以便在多个语句块中有选择地执行其中的一个。多分支结构比块结构的条件语句容易阅读。多分支结构的语法如下：

```
Select Case 测试表达式
  [Case 表达式 1
    [语句体 1]]
  [Case 表达式 2
```

```
[语句体 2]]
    …
  [Case Else
      [语句体 n]]
End Select
```

VBScript 先计算测试表达式，然后将表达式的值与每个 Case 后面的表达式的值进行比较。若相等，就执行该 Case 语句下的语句体。每个 Case 的值是一个或几个值的列表。如果在一个列表中有多个值，要用逗号把值隔开。如果不止一个 Case 与测试表达式匹配，那么只对第一个匹配的 Case 执行与之相关联的语句体。如果在表达式列表中没有一个值与测试表达式相匹配，若有 Case Else 子句，则执行 Case Else 子句中的“语句体 n”，否则，一条语句也不执行。

例 3-4-3：

```
<HTML>
<HEAD><TITLE>多分支结构程序举例</TITLE>
<SCRIPT LANGUAGE="VBScript">
Sub ShowMessage
  dim x
  x=InputBox("请输入 x 的值（1-7)：")
  if not isnumeric(x) then x=""
  select case x
  case ""
     MsgBox "输入错误，请输入数字！"
  case 1
     MsgBox "星期一"
  case 2
     MsgBox "星期二"
  case 3
     MsgBox "星期三"
  case 4
     MsgBox "星期四"
  case 5
     MsgBox "星期五"
  case 6
     MsgBox "星期六"
  case 7
     MsgBox "星期日"
  case else
     MsgBox "请输入 1-7 中的数字！"
  end select
End Sub
call ShowMessage '调用 ShowMessage 过程
</SCRIPT>
</HEAD><BODY>
……
</BODY></HTML>
```

程序的运行结果如图 3-4-3 所示。当输入 1～7 之间的数字时，会转换成对应的星期值输出。当输入字符串或 1～7 以外的数字时，则输出错误提示信息。

（a）　　（b）

图 3-4-3　多分支结构程序举例

3.4.3　循环控制语句

在编写脚本时，如果某段程序需要反复执行多次，可以通过循环结构来实现。在 VBScript 中，提供了三种不同风格的循环结构，即 For 循环、Do 循环和 While 循环。

1．For 循环

（1）For…Next 语句。For 循环用来完成已知循环次数的循环，也称计数循环。For 循环含有一个计数变量，每重复一次循环，计数变量的值就会增加或减少。For 循环的语法格式如下：

```
For 循环变量=初值 To 终值 [Step 步长]
    循环体
  [Exit For]
Next [循环变量]
```

For 循环按指定的次数执行循环体。执行 For 循环时，先将循环变量设为初值。测试循环变量是否小于（当步长值为正）或大于（当步长值为负）终值，若是，则执行循环体，否则退出循环，执行 For 循环后面的语句。Step 子句的值可以为正数，也可以为负数，负数代表负增长，如果省略，步长默认值是 1。

一般情况下，当循环变量到达终值时，正常结束 For 循环。但在有些情况下，在循环变量没有到达终值时，就要结束循环的执行，可以使用 Exit For 语句立即结束 For 循环。

下面使用 For 循环求 3 到 100 之间的所有素数。

例 3-4-4：

```
<HTML>
<HEAD><TITLE>For 循环结构程序举例</TITLE>
<SCRIPT LANGUAGE=VBScript>
<!--
Sub ShowMessage
  dim i,intSum,flag,result
  intSum=0
  result=""
  For i=3 To 100 Step 2
     flag=1
     For j=2 To i-1 Step 1
      If i mod j = 0 Then
        flag=0
        Exit For
      End If
     Next
     If flag=1 then
```

```
      result=result&i&";"
    End If
  Next
  MsgBox "3 到 100 之间的素数为：" & result
End Sub
call ShowMessage '调用 ShowMessage 过程
-->
</SCRIPT></HEAD><BODY>
......
</BODY></HTML>
```

上述程序在浏览器中的执行结果如图 3-4-4 所示。

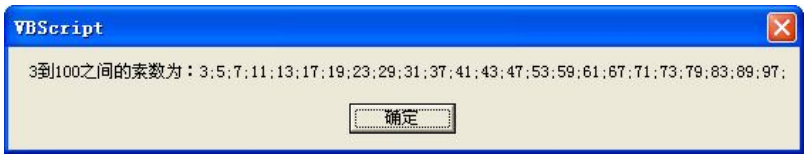


图 3-4-4　For 循环的使用

（2）For Each…Next 语句

在 VBScript 中，还可以使用 For Each 循环。For Each 循环与 For 循环类似，但 For Each 循环只对数组或对象集合中的每个元素重复一组语句，而不是重复一定的次数。如果不知道一个集合有多少个元素，则用 For Each 循环非常方便。For Each 循环的语法如下：

```
For Each 元素 In 集合
    语句体
Next [元素]
```

例如，列举使用 HTML 表单提交的所有值，可以使用下面的程序段：

```
<% '列举使用 HTML 表单提交的所有值
 For Each item In Request.Form
    Response.Write Request.Form(item)
 Next%>
```

2. Do 循环

在设计脚本时，对需要重复的程序段如果不知道循环的准确次数，可以使用 Do 循环。

Do 循环的语法格式有两种：

第一种：

```
Do [While|Until 循环条件]
    循环体
   [Exit Do]
Loop
```

第二种：

```
Do
    循环体
   [Exit Do]
Loop [While|Until 循环条件]
```

上面两种格式可以完成相同的功能，但是在执行流程上有区别：对于第一种格式的 Do 循环，是先判断循环条件，然后再根据循环条件的值来决定是否执行循环体；而第二种格式是先

执行一次循环体之后，再判断循环条件，根据循环条件的值决定是退出循环还是继续执行下一次循环，所以第二种格式不管条件是否成立都至少要执行一次循环体。

对于包含 While 关键字的 Do 循环，是在循环条件为真或不为 0 时一直重复执行循环体，直到循环条件不满足时退出 Do 循环。而对于包含 Until 关键字的 Do 循环，是在循环条件不为真或为 0 时一直重复执行循环体，直到循环条件为真或不为 0 时退出 Do 循环。

与 For 循环类似，Do 循环也可以使用 Exit 语句立即中止循环的执行。上面的程序也可以使用 Exit Do 语句来实现。下面的例子是对例 3-4-4 进行修改，使之实现使用 Exit Do 语句完成求 3 到 100 之间的所有素数。程序的执行结果与例 3-4-4 的结果相同。

例 3-4-5：

```
<HTML><HEAD><TITLE>Do 循环结构程序举例</TITLE>
<SCRIPT LANGUAGE=VBScript>
<!--
Sub ShowMessage
  dim i,intSum,flag,result
  intSum=0
  result=""
  i=3
  Do While i<=100
     flag=1
     j=2
     Do While j<=i-1
      If i mod j = 0 Then
        flag=0
        Exit Do
      End If
      j=j+1
     Loop
     If flag=1 then
       result=result&i&";"
     End If
     i=i+1
  Loop
  MsgBox "3 到 100 之间的素数为：" & result
End Sub
call ShowMessage '调用 ShowMessage 过程
-->
</SCRIPT></HEAD>
<BODY>
......
</BODY></HTML>
```

上面的程序在 i 的值小于等于 100 时，执行循环体，当 i 能整除 j 时，满足 if 语句的执行条件，通过 Exit Do 语句结束内层 Do 循环。

3．While 循环

在设计脚本时，对于循环次数不定的循环，也可以使用 While 循环。

While 循环的语法格式如下：

```
While 循环条件
    循环体
Wend
```

While 循环是先判断循环条件，根据循环条件的值来决定是否执行循环体。如果循环条件为真或不为 0 时执行循环体，直到循环条件不满足时退出 While 循环。下面使用 While 循环求 1 加到 100 的整数和。

例 3-4-6：

```
<HTML><HEAD><TITLE>While 循环结构程序举例</TITLE>
<SCRIPT LANGUAGE=VBScript>
<!--
Sub ShowMessage
  dim i,intSum
  i=1
  intSum=0
  While i<=100
      intSum=intSum +i
      i=i+1
  Wend
  MsgBox "从 1 加到 100 的整数和是：" & intSum
End Sub
call ShowMessage '调用 ShowMessage 过程
-->
</SCRIPT></HEAD>
<BODY>
……
</BODY></HTML>
```

程序的执行结果如图 3-4-5 所示。

图 3-4-5　1 加到 100 的整数和

3.4.4　其他常用简单语句

在 VBScript 中有许多非常简单但又很有用处，使用频率很高的语句、例如前面介绍的 Dim 语句和 ReDim 语句。

1. Rem 语句

注释语句，与单引号实现注释的作用相同。其用法如下：

```
Rem 注释语句
```

例如：Dim strName　　'定义一个变量用于存储姓名
　　Dim intAge　　Rem 定义变量存储年龄

2. Erase 语句

用来将非动态数组中的所有元素的值重新设置为空类型，其用法如下：

```
Erase  数组名
```

例如：Dim intArray(5)

```
For i=0 to 5
    intArray(i)=i+1
Next
Erase intArray
```

3. Set 语句和 Let 语句

Set 用来把对象的引用赋给变量或属性，其用法如下：

```
Set objectVar=数值
```

例如：Set rs=Server.CreateObject("ADODB.RECORDCOUNT")

Let 用来实现将一个表达式的值赋值给一个变量。其用法如下：

```
Let 变量名＝数值
```

例如：Let intVar=intVar1+intVar2

4. Rnd 函数

用于产生随机数，数值在 0 和 1 之间。要想得到某一范围内的值可以通过扩大倍数来实现。其用法如下：

```
变量＝Rnd*倍数
```

例如：intVar=int(Rnd*100)

在上例中，使用了数值强制类型转换，将括号里面的内容强制转换成了整数。

3.5 VBScript 函数及子过程

3.5.1 过程

在 VBScript 中，可以通过定义过程来完成特定的功能。VBScript 根据过程是否有返回值将过程划分为 Sub（子）过程和 Function（函数）过程。VBScript 的过程有如下几个特点：

- 过程只能有单一入口，但可以有多个出口。
- 在浏览器的任何一个网页中都可以定义过程，习惯上将过程定义在<HEAD>和</HEAD>中。
- 可以用浏览器所特有的事件来调用。
- 通过过程可以将重复使用的代码单独定义，提高代码利用率。
- 过程使得查错和改错工作变得简单。
- 可以向过程中传递任何有效的参数。

3.5.2 子过程

Sub 过程是没有返回值的过程，一般格式如下：

```
[Private][Public]Sub 过程名[(参数列表)]
    [语句块]
    [Exit Sub]
    [语句块]
End Sub
```

上面格式中的 Private 关键字表示此过程是私有过程，只能被进行过声明的脚本中的其他过程调用；而 Public 表示此过程是公有过程，可以被脚本中的其他任何过程调用。如果省略此关键字，则默认为 Public。

参数列表是可选项，表示子过程的参数，参数用于在调用过程和被调用过程之间传递信息，多个参数之间用逗号分开。

Sub 过程以 Sub 开头，以 End Sub 结束。每次调用子过程都会执行 Sub 和 End Sub 之间的语句。也可以在过程体内使用一个或多个 Exit Sub 语句终止过程的执行。

Sub 过程不能嵌套，即在一个 Sub 过程内，不能再定义另一个 Sub 过程或 Function 过程。只能通过调用执行另一个 Sub 过程，即可以嵌套调用。下面的例子完成的功能是从键盘输入两个数，求两数之和。

例 3-5-1：

```
<HTML>
<HEAD><TITLE>Sub 过程程序举例</TITLE>
<SCRIPT LANGUAGE=VBScript>
   <!--
   dim x,y
   x=CDbl(InputBox("请输入 x 的值"))
   y=CDbl(InputBox("请输入 y 的值"))
   call OutputAdd(x,y) '调用 OutputAdd 过程，并传递 x 和 y 的值
   Sub OutputAdd(a,b)
     dim z
     z=a+b
     MsgBox "两数之和是：" & z
   End Sub
-->
</SCRIPT></HEAD>
<BODY>
……
</BODY></HTML>
```

由上面的例子可以看出，定义子过程后，就可以在程序代码中调用。调用的方式有两种，一种是用 Call 语句，另一种是直接用子过程名。调用子过程时，调用语句必须是一个独立的语句。

用 Call 语句调用子过程的语法如下：

```
Call  子过程名([参数列表])
```

直接使用子过程名的语法如下：

```
子过程名 [参数列表]
```

这种调用子过程的特点是子过程名和后面的参数列表之间用空格隔开，不需要加括号。

上面例子中的过程调用语句 call OutputAdd(x,y)也可以改为如下形式：

```
OutputAdd  x,y
```

在子过程中可以使用 Exit Sub 语句强制从子过程中退出并返回。

3.5.3　函数

函数与子过程一样，也是用来完成特定功能的独立的程序代码。两者的区别是：子过程没有返回值，而函数在调用时将返回一个值。函数的语法如下：

```
[Private][Public] Function 过程名[(参数列表)]
   [语句块]
   函数名=表达式
   [Exit Function]
```

```
    [语句块]
End Function
```

其中“函数名=表达式”语句用于为函数设置返回值，该值将返回给调用的语句，函数中至少要含有一条这样的语句。类似于子过程，函数中可以用 Exit Function 语句直接退出函数。

在 VBScript 中编写的函数可以像 VBScript 提供的内部函数一样在表达式中使用。下面的程序使用函数求两个数之和。

例 3-5-2：

```
<HTML>
<HEAD><TITLE>Function 程序举例</TITLE>
<SCRIPT LANGUAGE=VBScript>
<!--
dim x,y,FuncResult
x=cdbl(InputBox ("请输入 x 的值"))
y=cdbl(InputBox ("请输入 y 的值"))
FuncResult=OutputAdd(x,y)  '调用 OutputAdd 函数，并传递 x 和 y 的值
MsgBox "两数之和是：" & FuncResult
Function OutputAdd(a,b)
  dim z
  z=a+b
  OutputAdd=z
End Function
-->
</SCRIPT></HEAD>
<BODY>
……
</BODY></HTML>
```

函数可以在表达式中进行调用。调用函数时，参数两边的括号不能省略。

同样，也可以用 Call 语句来调用函数。用 Call 语句调用时，VBScript 将放弃返回值。例如：

```
Call OutputAdd(x,y)
```

无参数函数的调用与变量的使用一样，只要使用函数名即可。

3.5.4 VBScript 内部函数摘要

1. 数学函数

数学函数包括求平方根函数、求绝对值函数、指数函数和对数函数等。用来完成各种数学运算。在 VBScript 中提供的常用数学函数有以下几种（假设 x 为一个数值表达式）：

- 求平方根函数（Sqr）：返回自变量 x 的平方根，x 必须大于或等于 0。例如：
  `Sqr(2)=1.4142135623731`
- 求绝对值函数（Abs）：返回自变量 x 的绝对值。例如：
  `Abs(-2)=2`
- 指数函数（Exp）：返回以 e 为底、以 x 为指数的值，即求 e 的 x 次幂。例如：
  `Exp(2)=7.38905609893065`
- 对数函数（Log）：返回自变量 x 的自然对数。例如：
  `Log(2)=0.693147180559945`

- 符号函数（Sgn）：返回自变量 x 的符号。当 x 为正数时，函数返回 1；当 x 为负数时，函数返回-1；当 x 为 0 时，函数返回 0。例如：

```
Sgn(-5)=-1
```

- 三角函数

Sin(x)函数：返回自变量 x 的正弦值。

Cos(x)函数：返回自变量 x 的余弦值。

Tan(x)函数：返回自变量 x 的正切值。

Atn(x)函数：返回自变量 x 的反正切值。

2. 字符串函数

字符串函数用于对字符串进行相应的处理。在 VBScript 中，常用的有以下几种：

- 空格函数 Space(n)：返回 n 个空格。
- 删除空白字符函数 Trim(字符串)：去掉字符串两端的空白字符。空白字符包括空格、Tab 键等。例如：

```
Trim("    欢迎您！    ")="欢迎您！"
```

- 字符串长度测试函数 Len(字符串|变量名)：如果 Len 函数的自变量为字符串，则返回字符串的长度；如果自变量为变量名，则返回变量的存储空间。例如：

```
Len("欢迎您！")=4
```

- 字符串截取函数：截取字符串，可以从字符串的左部、中部或右部截取。包括：

Left(字符串，n)：左部截取，返回字符串的前 n 个字符。例如：

```
Left(" 华北航天工业学院", 4)= "华北航天"
```

Mid(字符串，p，n)：中部截取，返回从第 p 个字符开始，向后的 n 个字符。例如：

```
Mid("华北航天工业学院", 5, 2)= "工业"
```

Right(字符串，n)：右部截取，返回字符串的后 n 个字符。例如：

```
Right("华北航天工业学院", 2)= "学院"
```

- 字母大小写转换函数：用来对字母的大小写进行转换。包括：

Ucase(字符串)：将字符串中的小写字母转换为大写字母。例如：

```
Ucase("Chinese")="CHINESE"
```

Lcase(字符串)：将字符串中的大写字母转换为小写字母。例如：

```
Lcase("CITY")="city"
```

- 字符串匹配函数：用来在一个字符串中查找另一个字符串。格式为：

```
InStr([首字符位置，]字符串 1，字符串 2[，n])
```

该函数在字符串 1 中查找字符串 2，如果找到了，则返回字符串 2 的第一个字符在字符串 1 中的位置；如果没找到，则返回 0。例如：

```
InStr("华北航天工业学院"，" 航天")=3
```

3. 日期和时间函数

- 日期函数，包括：

Year(Now)：返回当前系统的年份。

Month(Now)：返回当前系统的月份。

Day(Now)：返回当前系统的日期。

WeekDay(Now[，n])：返回当前系统的星期。

其中，参数 n 是可选的，它的取值范围从 0～7。用来设定每周的第 1 天从星期几开始，如果不设定此参数，则默认从星期日开始。如果设置此参数，则 0 表示采用系统默认设置，1～7 分别代表星期日到星期六。如当前系统日期为 2003 年 5 月 23 日，则：

```
WeekDay(Now)=6
```

- 时间函数，包括：

  Hour(Now)：返回当前系统的小时（0～23）。

  Minute(Now)：返回当前系统的分钟（0～59）。

  Second(Now)：返回当前系统的秒（0～59）。

4. 数据类型转换函数

数据类型转换函数用于将一种类型的数据转换成其他类型的数据。常用的有以下几种：

- CStr 函数：将数据转换成一个字符串。例如：

  ```
  CStr(123.45)= "123.45 "
  ```

- CInt 函数：将数据转换成一个整数。如果有小数部分则先进行四舍五入。例如：

  ```
  CInt(123.5)=124
  ```

- CDate 函数：将数据转换成一个日期。例如：

  ```
  CDate(123.5)=#1900-5-2 12:00:00#
  ```

- CBool 函数：将数据转换成一个布尔值。例如：

  ```
  CBool(123)=True
  ```

- CLng 函数：将数据转换成一个长整型数。如果有小数部分，则先进行四舍五入。例如：

  ```
  CLng(123456.51)=123457
  ```

- CSng 函数：将数据转换成一个单精度数。例如：

  ```
  CSng(12.4556752)=12.45568
  ```

- CDbl 函数：将数据转换成一个双精度数。例如：

  ```
  CDbl(12345.4556752)=12345.4556752
  ```

5. 数据类型判别函数

数据类型判别函数用于测试数据的子类型。常用的有以下几种：

- IsNull 函数：测试自变量是否是 Null，如果是则返回真，否则返回假。例如，假设执行语句 x=Null，则

  ```
  IsNull(x)=True
  ```

- IsEmpty 函数：测试自变量是否是 Empty，如果是则返回真，否则返回假。例如，如果用 Dim varTemp 先声明一个变量，然后测试该变量的值，此时变量的值为 Empty，则：

  ```
  IsEmpty(varTemp)=True
  ```

- IsNumeric 函数：测试自变量是否是一个数值，如果是则返回真，否则返回假。例如：

  ```
  IsNumeric(123.45)=True
  ```

- IsArray 函数：测试自变量是否是一个数组，如果是则返回真，否则返回假。例如，dim arrStudent(3)，则：

  ```
  IsArray(arrStudent)=True
  ```

- IsDate 函数：测试自变量是否是一个日期型数据，如果是则返回真，否则返回假。例如：

  ```
  IsDate(#5/28/2003#)=True
  ```

- IsObject 函数：测试自变量是否是一个对象，如果是则返回真，否则返回假。

6. 数组处理函数

- Lbound 函数：返回数组下界函数，VBScript 中数组的下界都是为 0，一般不常用。
- Ubound 函数：返回数组上界函数。

3.6　VBScript 的对象和事件

3.6.1　对象和事件的概念

VBScript 是基于对象的脚本语言，因此对象和事件是与网页设计密切相关的两个概念，对象是在浏览器中或者脚本编写中用于综合地描述一组功能和事件的组合体。每一个 HTML 文档都是以浏览器为执行环境，把自身作为一个 Document 文档对象，在浏览器对象 Windows 中执行所有代码。所有的网页对象都有一定的属性和方法，在 VBScript 中使用对象和属性名称时是区分大小写的，这一点要特别注意。

当使用 VBScript 的网页在浏览器中产生事件时，浏览器会把消息传递给 VBScript 的虚拟机，再将程序转到某对象的事件处理过程去处理。在网页中使用 VBScript 脚本时，增加了浏览器和网页的处理功能，可以使 VBScript 处理网页中的每一个对象。常用的事件有 Click 单击事件、Focus 聚焦事件、Load 加载事件和 Submit 提交事件等。

3.6.2　网页及浏览器对象

网页和浏览器对象包括：Windows 窗口对象、Frame 框架对象、History 历史对象、Navigator 漫游对象、Location 位置对象、Script 脚本对象和 Document 文档对象等。它们包括很多的属性、方法和事件。

1. Windows 窗口对象

在 VBScript 中控制 Windows 窗口对象，也就相当于控制浏览器，Windows 对象实际上代表了 Internet Explorer 对象本身。它有很多的属性、方法和事件，具体介绍如下：

（1）Windows 窗口对象的属性。

- DefaultStatus：字符串类型。
  功能：用来设置状态栏中的缺省文字。例如：DefaultStatus="OK"。
- Document：对象类型。
  功能：返回当前窗口的文档对象的引用。例如：Set Object(1)=Document。
- Frames：对象数组。
  功能：返回当前窗口中的框架。例如：Set Object(2)=Frame(1)。
- History：对象类型。
  功能：返回当前窗口的历史对象。例如：Set Object(3)=History。
- Location：对象类型。
  功能：返回位置类型对象。
- Name：字符串类型。
  功能：返回当前窗口的名字。

- Parent：对象类型。
  功能：返回当前窗口的父窗口的名字。
- Self：对象类型。
  功能：对当前窗口对象的另一个引用。
- Status：字符串类型。
  功能：返回或设置显示在状态栏中的文字。例如：Status="Loading…"。
- Top：对象类型。
  功能：返回代表最高级窗口的一个对象。

（2）Windows 窗口对象的方法。

- Alert：显示一个带“OK”按钮的警告消息框，没有返回值。例如：Alert（"谢谢使用！"）。
- ClearTimeout：删除一个指定的计数器，无返回值。例如：ClearTimeout（计数器名）。
- Close：关闭窗口，无返回值。
- Conform：显示一个带有 OK/Cancel 的消息框，返回布尔类型值。例如：blnVar=Conform("准备好了吗？")。
- Open：打开一个新窗口或创建一个新窗口并在其中显示一个文档，返回一个 Windows 对象。例如：Open(URL,Title,Features,Width,Height)。
- Prompt：显示一个带有 OK/Cancel 类型的输入型对话框，返回字符串类型值。
- SetTimeout：经过指定时间后执行特定的代码，返回长整数值。

（3）Windows 窗口对象的事件。

- OnLoad：加载页面时调用相应的事件。例如：<BODY OnLoad="处理事件过程">。
- OnUnload：卸载页面时调用相应的事件。例如：<BODY OnUnLoad="处理事件过程">。

2. Document 文档对象

（1）Document 文档对象的属性。

- LinkColor：返回或设置文档中链接的颜色。这个属性和“<BODY>”标签中的“LINK”属性相同，颜色值为十六进制数或颜色名。
- AlinkColor：返回或设置文档中的活动链接的颜色。所谓活动链接就是当鼠标光标指向一个链接并按下鼠标按键而未释放。
- VlinkColor：返回或设置未曾被访问过的链接，与“<BODY>”标签中的“VLINK”属性相同。
- BGColor：返回或设置文档的背景色。例如，Document.BGColor="green"。
- FGColor：返回或设置文档的前景色。
- Forms：此对象表示在一个 HTML 文档中的一个窗体，可以通过 Forms 数组得到文档中所有的 Form 对象。
- LastModified：返回当前文档最近一次被修改的时间。
- Title：返回当前文档的标题，这个属性是只读的，不允许实时改变。
- Cookie：可以设置客户方的 Cookie。

（2）Document 文档对象的方法。

- Write：将字符串变量写入当前文档中。例如：Document.Write strVar。
- WriteLn：写入到当前文档时，将字符串变量作为一个新行附加到结尾。

- Open：为输出数据打开一个新的文档。
- Close：关闭文档流。
- Clear：关闭已经开启的文档输出流，并且清除屏幕上所有的内容。

3. Location 位置对象

Location 位置对象的属性：

- Href：返回或设置载入浏览器窗口的完整的 URL。
- Protocol：返回或设置 URL 使用的协议，例如 HTTP 协议、FTP 协议等。
- Host：返回或设置 URL 的宿主和端口，宿主和端口之间用冒号隔开。
- HostName：读取或设置 URL 的宿主，可以是一个 IP 地址或是一个名字。
- Port：返回或设置 URL 的端口。
- PathName：返回或设置 URL 的路径名。
- Search：返回或设置 URL 的搜索部分，搜索部分是当浏览器提交数据到服务器时，在 URL 中问号后面的字符串。例如，http://www.server.com/search.asp?id="123"。
- Hash：返回或设置 URL 的无用部分。

4. History 历史对象

History 历史对象可以控制浏览器已经访问过的网页，它只有一种属性就是其长度 Length。它有三种方法分别为：

- History.back(n)：就像单击“Back”按钮一样，可以回到最近访问过的 URL。
- History.forward(n)：可以在历史清单中前移 n 步进行搜索，相当于单击“Forward”按钮。
- History.go(n)：在历史清单中跳到第 n 项。

5. Form 表单对象

Form 表单对象的属性：

- Action：返回或设置表单的动作属性。
- Elements：返回或设置表单的元素属性。
- Method：返回或设置表单的方法属性。
- Target：返回或设置表单的目标属性。
- Encoding：返回或设置表单的代码属性。

3.6.3　浏览器内嵌 HTML 控件

IE 浏览器有一些嵌入其内部的 HTML 控件，关于它们的使用在第二章中已经有了较详细地介绍，这些控件都可以触发相应的事件如表 3-6-1 所示。

当事件发生后，会有相应的处理事件的方法开始执行。处理事件的一般过程有以下几种方式：

- 当表单对象被鼠标单击时，产生 Click 事件，OnClick()事件处理过程开始执行。
- 表单内的选择对象或者文本对象不再被聚焦时，产生 Blur 事件，OnBlur()事件处理过程开始执行。
- 相应的对象被改变时，产生 Change 事件，OnChange()事件处理过程开始执行。
- 对象被聚焦时，例如当鼠标移动到对象上时即产生了 Focus 事件，OnFocus()事件处理过程开始执行。

- 当用户在文本框区域内选择了一段文字时，就会产生 Select 事件，OnSelect()事件处理过程开始执行。

表 3-6-1 浏览器内嵌 HTML 控件及其触发的事件

控件	事件	方法	使用说明
Button	OnClick OnFocus	Click Focus	<Input Type="Button" [Name=] [Value=] [OnClick=] [OnFocus=] >
CheckBox	OnClick OnFocus	Click Focus	<Input Type=" CheckBox" [Name=] [Value=] [Checked] [OnClick=] [OnFocus=] >
PassWord	OnFocus	Focus	<Input Type="Password" [Name=] [Value=] [OnClick=] [OnFocus=] >
Radio	OnClick OnFocus	Click Focus	<Input Type=" Radio" [Name=] [Value=] [Checked] [OnClick=] [OnFocus=] >
Reset	OnClick OnFocus	Click Focus	<Input Type="Reset" [Name=] [Value=] [OnClick=] [OnFocus=] >
Select	OnFocus OnBlur OnChange	Focus Blur	<Select Name=[Size=][Multiple] [OnFocus=] [OnBlur=] [OnChange=] > <Option [Selected][value=]>Items</Option> </Select>
Submit	OnClick OnFocus	Click Focus	<Input Type="Submit" [Name=] [Value=] [OnClick=] [OnFocus=] >
Text	OnFocus OnBlur OnChange OnSelect	Focus Blur Select	<Input Type="Text" [Name=] [Value=] [Size=] [MaxLength=] [OnBlur=] [OnFocus=] [OnChange=] [OnSelect=]>
TextArea	OnChang OnSelect	Select	<TextArea [Name=] [Rows=] [OnBlur=] [OnFocus=] [OnChange=] [OnSelect=]></TextArea>

3.6.4 对象和事件实例

本实例实现的功能是，在网页上按照顺序输入用户信息，用户输入完“姓名”以后，按回车键光标自动移到“年龄”文本框中等待输入，“电话”和“住址”的输入也类似。在程序中使用了 Window 对象及其事件。

例 3-6-1：

```
<HTML><HEAD><TITLE>按顺序输入信息</TITLE>
<Script Language="VBScript">
<!--
sub Windows_onload
Form1.xm.Focus
end sub
sub xm_onkeypress
  if Window.Event.KeyCode=13 then
    Form1.nl.Focus
```

```
  end if
end sub
sub nl_onkeypress
  if Window.Event.KeyCode=13 then
    Form1.dh.Focus
  end if
end sub
sub dh_onkeypress
  if Window.Event.KeyCode=13 then
    Form1.zz.Focus
  end if
end sub
sub zz_onkeypress
  if Window.Event.KeyCode=13 then
    Form1.Button1.Focus
  end if
end sub
sub button1_OnClick
   Dim txt1,txt2,txt3,txt4,alltxt
   txt1=Form1.xm.value
   txt2=Form1.nl.value
   txt3=Form1.dh.value
   txt4=Form1.zz.value
   alltxt="姓名为:"+txt1+vbcrlf
   alltxt=alltxt+"年龄为:"+txt2+vbcrlf
   alltxt=alltxt+"电话为:"+txt3+vbcrlf
   alltxt=alltxt+"住址为:"+txt4
   msgbox alltxt
 end sub
 sub button1_onkeypress
   if Windows.Event.KeyCode=13 then
     button1_onclick
   end if
 end sub
-->
</Script></HEAD>
<BODY bgcolor=White>
<form id=Form1>
<center>
<p>姓名：<input id=xm name=xm></p><br>
<p>年龄：<input id=nl name=nl></p><br>
<p>电话：<input id=dh name=dh></p><br>
<p>住址：<input id=zz name=zz></p><br>
<input id=button1 name=button1 type=button value="提交">
</form></BODY></HTML>
```

在浏览器中运行上段代码，显示效果如图 3-6-1 所示。

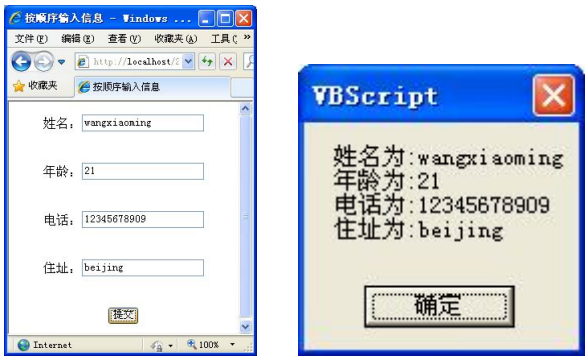


图 3-6-1　用户信息输入实例

思考题

1．服务器端脚本和客户端脚本的主要区别是什么？

2．VBScript 脚本语言的子数据类型和常用的控制语句有哪些？

3．什么时候使用静态数组？什么时候使用动态数组？

4．如何编写子过程和函数？

上机实验

1．设计一个可以在网页上使用的简易的计算器。

2．使用上一章介绍的 XML 文件访问的方法，实现在第一个下拉列表中选择省份，第二个下拉列表显示市的功能。

3．使用 VBScript 输出显示九九乘法口诀表，使用 CSS 样式改变表格的样式。

4．使用数组定义 20 个人的信息，然后开始报数，报到 3 的倍数的人出局，输出最后剩下的人的信息以及他在数组中的位置。要求：使用函数或过程。