# 一、VBScript介绍

Vbscript是由微软提供的一种脚本语言,它是Visual Basic编程语言的子集。

Vbscript通常都是和HTML结合在一起使用,也就是说,vbscript是融合在HTML或者ASP文件中。在一个HTML文件中,vbscript的声明方式和javascript的声明方式比较类似,具体如下:

*<script language="VBScript">*

*<! ---*

*//输入VBScript代码*

*---!>*

*</script>*

VBScript代码写在成对的<script>标记之间。代码的开始和结束部分都有<script>标记。其中language属性用于指定所使用的script语言。其实浏览器是可以使用多种script语言的,其声明方式都差不多,只需要在language属性中指定不同的语言即可。需要注意的是,<script>标记中的VBScript代码被嵌入在注释标记(<!--和--> )中,这样能够让那些不能够识别<script>标记的浏览器将代码当成注释显示在页面中。

VBScript代码同样也是放在放在HTML文档中头标记<head>和</head>或体标记<body>和</body>之间。

**例**：

<html>

<head>

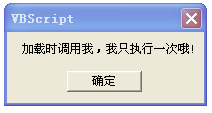
<title>html文件加载时调用VBScript</title>

</head>

<SCRIPT LANGUAGE="VBScript">

<!--

Msgbox "加载时调用我，我只执行一次哦!" //Msgbox是VBScript的一个内置函数,作用是弹出一个对话框

-->

</SCRIPT>

<body>

学习VBScipt

</body>

</html>

该例子在HTML加载时会自动弹出一个对话框,且仅在加载的时候调用一次,在整个页面活动期间不会再弹出该对话框。

事件同样也会激发在函数里的VBScript代码。

**例**：

<html>

<head>

<title>html文件加载时调用VBScript</title>

</head>

<SCRIPT LANGUAGE="VBScript">

Sub Window\_Onload()

Msgbox "这是在函数内部自动执行的代码哦！"

End Sub

</SCRIPT>

<body>

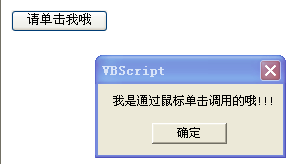
学习VBScipt

</body>

</html>

如果出现了Sub关键字,那就表示这是一个过程，Sub过程是包含在Sub和End Sub语句之间的一组VBScript语句，执行操作但不返回值。由于Window\_Onload是在页面加载的时候自动激活的事件调用,所以在该过程中的代码引入会被执行。VBScript的代码或者函数,是通过外部事件来激活调用的。

**例**：

<html>

<head>

<title>html文件加载时调用VBScript</title>

<SCRIPT FOR="Button1" EVENT="onClick" LANGUAGE="VBScript">

MsgB ox "通过鼠标单击调用的哦!!!"

</SCRIPT>

</head>

<body>

<FORM NAME="Form1">

<INPUT TYPE="Button" NAME="Button1" VALUE="请单击我哦">

</FORM>

</body>

</html>

FOR="Button1"表示该代码响应名称为"Button1"的按钮,而代码EVENT="onClick"表示该段代码响应"Button1"按钮的单击事件。

# 二、数据类型Variant

唯一一个：Variant，它根据具体数据作出处理。

Variant包含的数据信息称为子类型。通常情况下，将所需的数据放入Variant后，Variant会自动使用最合适其包含的数据方式操作。

|  |  |
| --- | --- |
| 子类型 | 描述 |
| Empty | 未初始化的Variant，对于数字变量其值为0；对于字符变量其值为零长度。 |
| Null | 不包含任何有效数据的Variant |
| Boolean | 其值为Ture或者False |
| Byte | 取值范围：0~255内的整数 |
| Integer | 取值范围：-32,768~32,767 |
| Currency | 取值范围：-922,337,203,685,477.5808~922,337,203,685,477.5807 |
| Long | 取值范围：-2,147,483,648~2,147,483,647内的整数 |
| Single | 包含单精度浮点数,负数范围为-3.402823E38~-1.401298E-45  正数的范围为1.401298E-45~3.402823E38 |
| Double | 包含双精度浮点数，负数的范围为-1.79769313486232E308~-4.94065645841247E-324  正数的范围为4.94065645841247E-324~1.79769313486232E308 |
| Date(Time) | 包含表示日期的数字,日期范围从公元100年1月1日到公元9999年12月31日 |
| String | 包含变长字符串,最大长度可为20亿个字符 |
| Object | 包含对象 |
| Error | 包含错误代码 |

# 三、变量声明

定义一个变量的方法是直接使用关键字"dim"，同时还可以使用public或者private关键字来确定这个变量的作用范围。如果需要同时定义多个变量，可以使用如下定义语句：

*Dim myName,yourname,hisname*

同时为了系统自动帮助减少错误，可以在程序代码开始加上一句限制语句即"Option Explicit"，如果在编程的过程中没有对变量进行声明，那系统就会报错。

一个变量的作用域是由它声明的位置所决定的，如果在过程中声明变量，则只有在该过程中的代码可以访问或更改变量值，此时变量具有局部作用域并称为过程级变量。如果在过程之外声明变量，则该变量可以被Script中所有的过程识别，称为脚本级或者模块级变量，具有Script级作用域。

**例**：

<script Language=VBScript>

<!--

Sub hello()

Dim aaa ' 声明一个aaa的过程级变量

aaa=aaa+1

End Sub '过程级变量的生命期结束

Sub welcome()

Dim bbb '声明本地变量

bbb=aaa+10 ' 会产生错误，因为aaa超出了作用范围

End Sub

-->

</script>

使用Const语句可以创建名称具有一定含义的字符串型或数值型常数，并给他们赋值。例如：

*Const MyString="这是一个字符串"*

*Const MyAge=20*

字符串文字包含在两个引号("")之间，这是区分字符串型常数和数值型常数最明显的方法；日期文字和时间文字包含在两个井号(#)之间，例如：

Const cutoffDate=#2007-1-11#

# 四、控制语句

使用条件语句和循环语句可以控制Script的流程，使用条件语句可以编写进行判断和重复操作的VBScript代码。

**循环语句：**

● Do………Loop：当（或直到）条件为Ture时循环

多次运行语句块。当条件为Ture的时候，或者条件变为Ture之前，重复执行语句块。

**(1)** **当条件为Ture时重复执行语句这种方式的使用格式如下所示：**

Do while (条件语句)

'执行语句块'

Loop

Do While……Loop对条件求值，如果条件为真，则对条件后面的语句求值。完成了上述操作后再次对条件求值，如果条件还是为真，则再次对语句进行求值。它不断的重复该过程，直到条件为false的时候为止。操作后再次对条件求值，如果条件还是为真，则再次对语句进行求值。它不断的重复该过程，直到条件为false的时候为止。

**例**：

<html>

<head>

<title>第一次出现数字的位置</title>

</head>

<SCRIPT LANGUAGE="VBScript">

<!--

Dim inString,c

inString = "曾云好是于1987年10月出生的"

Dim i,strLen,formula

i=1

strLen=Len(inString) / /函数Len是返回字符串内字符的数目，这里返回的16

formula=-1

Do While (i<=strLen) And (formula=-1)

c=Mid(inS tring,i,1) //函数Mid从字符串中返回指定数目的字符

If IsNumeric(c) Then formula=i //函数IsNumeric是返回 Boo lean 值指明表达式的值是否为数字

i=i+ 1

Loop

msgbox "数字第一次出现的位置是"&formula

-->

</SCRIPT>

<body>

第一次出现数字的位置

</body>

</html>

搜索输入字符串中数字第一次出现的位置。如果发现了数字，则返回该数字的位置，否则返回-1。

**(2) 重复执行语句直到条件变为Ture：**

格式如下：

*Do until (条件语句)*

*'执行语句'*

*Loop*

Do until……. Loop与Do While…..Loop很相似，不同的是它是一直对语句求值直到条件为真，而不是当条件为真时对语句求值，即只要条件表达式为False或零，而不是为True或非零时，循环才被执行。与上面相对应，此种方式还有另外一种使用方式，如下所示：

*Do*

*'执行语句'*

*Loop until(条件语句)*

他们之间的区别和上面的是一样的，只是一个判断条件变成相反而已。

**(3) 退出循环**

*Do until或While (条件语句)*

*'如果条件符合*

*'执行代码*

*'如果条件不符合*

*Exit Do '退出循环*

*Loop*

Exit Do语句用于退出Do….Loop循环，因为通常只是在某些特殊的情况下要退出循环。

● While………Wend：当条件为Ture时循环

该循环的使用格式如下：

While ( 条件语句)

'执行语句'

Wend

While循环与Do While…..Loop相似，不同的是它不支持Exit语句。

● For…………Next：制定循环次数，使用计数器重复运行语句

在编程的时候如果不知道循环多少次语句时，Do循环可以发挥很好的作用。但是如果知道循环的次数，那么For…………Next是更好的一个选择。与Do循环不同，For循环可以使用counter的变量，该变量的值将在每次重复循环的过程中递增或者递减。语法如下：

For counter [As datatype] = start To end [Step step]

[执行语句]

[Exit For]

[执行语句]

Next

符号[]表示的里面的参数是可选的。变量counter必须是支持大于或等于(>=)、小于或等于(<=)和加法(+)运算符的数字数据类型，通常为Integer。迭代值start、end和step是必须计算数据类型的表达式，扩展为counter类型，通常他们都是Integer，可选项step可以为正也可以为负。如果省略的话，则采用1。如果没有在循环外面声明counter，那么可以在For语句中使用As子句声明该变量。

开始执行For……..Next循环时，VBScript计算start、end和step。然后将start赋予counter。执行语句之前，先将counter与end进行比较，如果counter已经超过了结束值，则For循环终止，控制传递给Next语句后面的语句，否则执行该语句块。

每次VBScript遇到Next语句时，都按step递增计数器，然后返回到For语句。它再次将计数器与end进行比较，并根据结果执行块或者终止循环。这个过程不断的继续，直到计数器超过end或执行了Exit For语句为止。

到counter超过结束值后，循环就会终止。如果counter等于end，则循环继续。如果step为正数，确定是否执行循环代码块的比较运算为counter<=end；如果step为负数，则为counter>=end。

若要增加计数器的变量，可以使用正数的step。这样做之后，应指定大于起始值的结束值。在下面的示例中，计数器变量J每次循环重复递增2。循环结束后，Total为2、4、6、8和10的总和。

Dim J, Total

For J = 2 To 10 step 2 //J的值从2到10，而且每次增加的为2，所以J的值有2、4、6、8、10

Total = Total + 2 //循环体的动作是将每次循环产生的J的值相加，直到J的值为10之后，循环结束

Next

MsgBox("The Total is "&Total)

若要减少计数器变量，可使用负数的Step值，这样做之后，应指定小于起始值的结束值。在下面的示例中，计数器变量J每次循环重复递减2。循环结束后，Total为16、14、12、10、8、6、4的总和。

Dim N, Total

For N = 16 To 4 Step -2 //J的值从16到4，而且每次递减的为2，所以J的值有16、14、12、10、8、6、4

Total = Total + N //循环体的动作是将每次循环产生的J的值相加，直到J的值为4之后，循环结束

Next

MsgBox("The Total is "&Total)

使用Exit For语句可在计数器超过其结束值之前退出For…….Next循环。例如，如果检测到使继续跌代不必要或不可能的条件，则要退出循环。

下面示例利用For循环，每次递增为1，求出了1到100的总和，代码如下：

<html>

<head>

<title>1到100的总和</title>

</HEAD>

<body>

<Script Language=VBScript>

<!--

Dim j, total

For j = 1 To 100

total = total + j

Next

MsgBox "1到100的总和为 " & total

-->

</Script >

</body>

</html>

运行之后结果为5050。

● For Each………….Next：对于集合中每项或数组中的每个元素，重复执行一组语句

**条件语句：**

● If……….Then……Else语句

If……….Then……Else语句用于计算条件是否为True或False，并且根据计算结果制定要运行的语句。通常，条件是使用比较运算符对值或变量进行比较的表达式。If……….Then……Else语句的使用格式如下：

*If condition [ then ]*

*[ statements ]*

*[ ElseIf elseifconfition [ then ]*

*[ elseifstatements ] ]*

*[ Else*

*[elsestatements ] ]*

*End If*

**condition**：必选项，表达式。为condition提供的表达式必须计算为Ture或Flase，或者计算为能够隐式转换为Boolean的数据类型。

**Statements**：在多行格式中是可选项，在没有else子句的单行格式中是必选项。跟在if……then后面的一条或多条语句，当condition为True时将执行这些语句。

**Elseifconfition**：如果存在elseif，则必选项。功能与condition相同。

**Elseifstatements**：可选项，跟在elseif……..Then后面的一条或多条语句，当相关的elseifcondition为True时将执行这些语句。

**Elsestatements**：在多行格式中是可选项；在有Else子句的单行格式中是必选项。当前面的condition或Elseifconfition表达式都不为True时执行的一条或多条语句。

**End If**：终止if….else块。

(1) 如果条件为Ture，则执行语句。

如果要在条件为True时运行单行语句，则可使用If……….Then……Else的单行语法。下面示范了单行语法，需注意的是这里省略了关键字else。

Dim myDate

myDate = #2007/1/25#

If myDate < Now Then myDate = Now

如果要运行多行代码，则必须使用多行(或块)语法。多行语法包含End If语句，如下所示：

Dim age

Age = 15

If age > 18 Then

MsgBox "你已经长大成人了！"

End If

(2) 条件为Ture和False时分别运行某些语句

可以使用If……….Then……Else语句定义两个可执行语句块：条件为Ture时运行某一条语句块，条件为False时运行另一语句块，如下示例：

Dim age

Age =15

If age>18 Then

MsgBox "你已经长大成人了！"

Else

MsgBox "你还是小孩子咯！"

End If

(3) 对多个条件进行判断

If……….Then……Else语句的变形允许从多个条件中选择，即增加ElseIf子句以扩充

If……….Then……Else语句的功能，使得可以控制基于多种可能的程序流程。如下示例：

Dim age

Age =15

If age<18 Then

MsgBox "你还未成年哦"

Elseif age>18 and age<40 then

MsgBox "人到中年了哦"

Else

MsgBox "人到老年了哦"

End If

下面的例子中，如果学生的成绩在60分以内为不及格，60-80分为及格，80-90分为良好，90分以上为优秀。

<html>

<head>

<title>if...then...else示例</title>

</HEAD>

<body>

<Script Language=VBScript>

<!--

dim myValue

myValue=95

if myValue<60 then

document.write "你不及格！"

elseif myValue>=60 and myValue<80 then

document.write "你刚刚及格！"

elseif myValue>=80 and myValue<90 then

document.write "你的成绩不错，获得良好！"

else

document.write "恭喜你，你获得优秀！"

end if

-->

</Script >

</body>

</html>

● Select…..Case语句

提供了If……….Then……Else结构的一个变通形式，可以从多个语句块中选择执行其中的一个。Select…..Case结果在其开始处使用一个只计算一次的简单测试表达式。表达式的结果将与结构中的每个case的值比较，如果匹配，则执行与该case关联的语句块。Select…..Case语句的使用格式如下：

*Select [case] testexpression*

*[Case expressionlist*

*[statements]]*

*[Case Else*

*[elsestatements]]*

*End Select*

**Testexpression**：必选项。必须是某个基本数据类型，如数值、字符等。

**Expressionlist**：在Case语句中是必选项。表达式子句列表表示Testexpression的匹配值。多个表达式子句以逗号隔开。每个子句都可以采用下面的某一种形式：

expression1 To expression2

[Is] comparisonoperator expression

Expression

使用关键字To制定Testexpression的匹配值范围的边界。expression1的值必须小于或等于expression2的值。

**Statements**：可选项，Case后面的一条或多条语句，当Testexpression匹配Expressionlist中的任何子句时执行。

**Elsestatements**：可选项，Case Else后面的一条或多条语句，当Testexpression不匹配任何Case语句的Expressionlist中的任何子句时执行。

**End Select**：终止Select…..Case语句。

**例**：

Select Case Number

Case 1 To 5

MsgBox "between 1 to 5"

Case 6,7,8

MsgBox "between 6 to 8"

Case 9 To 10

MsgBox "between 9 to 10"

Case Else

MsgBox "Not between 1 to 10"

End Select

下面的例子是根据数字来判断某人是哪里人。

<html>

<head>

<title>select case示例</title>

</HEAD>

<body>

<Script Language=VBScript>

<!--

dim whereNum

WhereNum=3

select case WhereNum

case 1

msgbox "你是江西人"

case 2

msgbox "你是湖南人"

case 3

msgbox "你是北京人"

case else

msgbox "你是上海人"

end select

-->

</Script >

</body>

</html>

# 五、过程和函数

过程和函数是提高编码效率，有效的减少代码复杂以便于维护的重要手段。对于一些常用的功能可以利用过程和函数来模块化。过程和函数的区别在于过程是没有返回值的，而函数有返回值。

**VBScript的过程**

Sub过程是包含在Sub和End Sub语句之间的一组VBScript语句,执行操作但不返回值。Sub过程可以使用参数(由调用过程传递的常数、变量或表达式)。如果Sub过程没有任何参数，则Sub语句必须包含空括号()。

过程的定义格式如下：

*Sub 过程名(参数)*

*'代码'*

*End Sub*

调用过程的语法是：call 过程名(参数)

解释过程最好的方法就是通过示例来说名，下面示例中的判断语句用一个过程来实现。代码如下：

<html>

<head>

<title>过程演示</title>

</HEAD>

<body>

<Script Language=VBScript>

<!--

Call FindPeople(2)

sub FindPeople(whereNum)

select case WhereNum

case 1

msgbox "你是江西人"

case 2

msgbox "你是湖南人"

case 3

msgbox "你是北京人"

case else

msgbox "你是上海人"

end select

end sub

-->

</Script >

</body>

</html>

上述代码中，我们使用了标记对即Sub和End Sub将判断语句都包含在里面，而该过程有一个名称即为FindPeople，FindPeople括号中的变量为参数，whereNum参数可以被过程中的代码使用。因为过程不会自动运行，因此必须有外面的语句调用，在过程的前面使用了调用语句Call FindPeople(2)，其中2就是传递过来的参数，然后过程FindPeople根据参数2计算得到当前来自湖南的，并弹出对话框。

**VBScript的函数**

函数和过程基本类似，唯一的区别就是函数是有返回值的。同时在函数的定义上，使用关键字是"Function"，因此函数包含在Function和end Function之间的一段代码。

函数的定义格式如下：

*Function 函数名(参数) [ As 类型]*

*'代码'*

*End Function*

由于函数过程可以返回值，因此函数过程也就有了数据类型，其通过语法中的As子句确定。

在函数过程内，可以通过以下语句格式将结果返回：函数名= 表达式

**例**：

Function MyFunc(X As Integer)

MyFunc=X\*X+10

End Function

还可以调用函数过程，其基本语法为：变量=函数名(参数)。例如：

*Dim MyReturn*

*MyReturn= FunName(参数)*

上面例子是计算1到100之和，下面将通过一个函数来实现这个过程。给定一个正整数，通过函数SumTotal可以计算出从1到该整数之间所有数值的和。代码如下：

<html>

<head>

<title>函数演示</title>

</HEAD>

<body>

<Script Language=VBScript>

<!--

dim myNum

myNum=100

function SumTotal(number) / /定义一个函数

Dim j

For j = 1 To number

Su mTotal= SumTotal+ j

Next

end function

MsgBox "从1到" & myNum & "的总和为" & SumTotal(myNum) //调用函数

-->

</Script >

</body>

</html>

# 六、常用的函数

**(1)、Trim(string)**

除去字符串string两端的空格，这个函数经常在过滤空格上运用的最多，是取得参数后所必须进行的一个操作。与之类似的函数还有**Ltrim(string)**，作用为除去字符串左边的空格；还有**Rtrim(string)**，作用是字符串右边的空格。

**(2)、Ucase(string)**

把字符串string中的字符换成大写。与之相反的函数是**Lcase(string)**，作用当然是把字符串string中的字符换成小写。

**(3)、Abs(num)**

返回num的绝对值。

**(4)、Now()**

返回系统的日期和时间。与之相类似的函数有**Date()**，返回系统当前日期；**Time()**，返回系统当前的时间。

**(5)、IsNumber(var)**

如果var是个数字，则返回ture，否则返回false。与之相对应的函数有**IsArry(var)**、**IsDate(var)**、**IsEmpty(var)**等等。

**(6)、InputBox函数**

是一个输入框函数，在对话框中显示提示，等待用户输入文本或单击按钮，并返回文本框内容。语法为

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

其参数如下:

**Prompt**：作为消息显示在对话框中。prompt的最大长度大约是1024个字符，这取决于所使用的字符的宽度。如果 prompt中包含多个行，则可在各行之间用回车符(Chr(13))、换行符(Chr(10))或回车换行符的组合(Chr(13) & Chr(10))以分隔各行。

**Title**:显示在对话框标题栏中的字符串表达式。如果省略 ti tle，则应用程序的名称将显示在标题栏中。

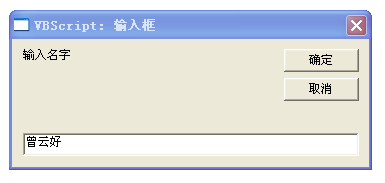
**Default**:显示在文本框中的字符串表达式，在没有其它输入时作为默认的响应值。如果省略default，则文本框为空。

**Xpos**:数值表达式，用于指定对话框的左边缘与屏幕左边缘的水平距离（单位为缇）。如果省略xpos，则对话框会在水平方向居中。

**Ypos**:数值表达式，用于指定对话框的上边缘与屏幕上边缘的垂直距离（单位为缇）。如果省略ypos，则对话框显示在屏幕垂直方向距下边缘大约三分之一处。

**Helpfile**:字符串表达式，用于标识为对话框提供上下文相关帮助的帮助文件。如果已提供helpfile，则必须提供context。

**Context**:数值表达式，用于标识由帮助文件的作者指定给某个帮助主题的上下文编号。如果已提供context，则必须提供helpfile。

如果同时提供了helpfile和context，就会在对话框中自动添加“帮助”按钮。如果用户单击确定或按下ENTER，则InputBox 函数返回文本框中的内容。如果用户单击取消，则函数返回一个零长度字符串("") 。下面例子利用InputBox 函数显示一输入框并且把字符串赋值给输入变量：

<html>

<head>

<title>函数演示</title>

</HEAD>

<body>

<Script Language=VBScript>

Dim Input

Input = InputBox("输入名字","输入框","曾云好")

MsgBox ("输入：" & Input)

</Script >

</body>

</html>

**(7)、MsgBox函数**

这是一个消息框函数，在对话框中显示消息，等待用户单击按钮，并返回一个值指示用户单击的按钮。语法为 *MsgBox(prompt[, buttons][, title][, helpfile,context])*

其参数说明如下:

**Prompt**：作为消息显示在对话框中的字符串表达式。prompt的最大长度大约是1024个字符，这取决于所使用的字符的宽度。如果prompt中包含多个行，则可在各行之间用回车符(Chr(13))、换行符(Chr(10))或回车换行符的组合(Chr(13)&Chr(10))分隔各行。

**Buttons**：是表示指定显示按钮的数目和类型、使用的图标样式，默认按钮的标识以及消息框样式的数值的总和。有关数值，请参阅“设置”部分。如果省略，则buttons的默认值为0。

buttons 参数可以有以下值：

|  |  |  |
| --- | --- | --- |
| 常数 | 值 | 描述 |
| vbOKOnly | 0 | 只显示确定按钮 |
| vbOKCancel | 1 | 显示确定和取消按钮。 |
| vbAbortRetryIgnore | 2 | 显示放弃、重试和忽略按钮。 |
| vbYesNoCancel | 3 | 显示是、否和取消按钮。 |
| vbYesNo | 4 | 显示是和否按钮。 |
| vbRetryCancel | 5 | 显示重试和取消按钮。 |
| vbCritical | 16 | 显示临界信息图标。 |
| vbQuestion | 32 | 显示警告查询图标。 |
| vbExclamation | 48 | 显示警告消息图标。 |
| vbInformation | 64 | 显示信息消息图标。 |
| vbDefaultButton1 | 0 | 第一个按钮为默认按钮。 |
| vbDefaultButton2 | 256 | 第一个按钮为默认按钮。 |
| vbDefaultButton3 | 512 | 第一个按钮为默认按钮。 | |
| vbDefaultButton4 | 768 | 第一个按钮为默认按钮。 |
| vbApplicationModal | 0 | 应用程序模式：用户必须响应消息框才能继续在当前应用程序中工作。 |
| vbSystemModal | 4096 | 系统模式：在用户响应消息框前，所有应用程序都被挂起。 |

第一组值(0-5)用于描述对话框中显示的按钮类型与数目；

第二组值(16,32,48,64)用于描述图标的样式；

第三组值(0,256, 512)用于确定默认按钮；

第四组值(0,4096)决定消息框的样式。在将这些数字相加以生成buttons参数值时，只能从每组值中取用一个数字。

**Title**：显示在对话框标题栏中的字符串表达式。如果省略title，则将应用程序的名称显示标题栏中。

**Helpfile**：字符串表达式，用于标识为对话框提供上下文相关帮助的帮助文件。如果已提供helpfile，则必须提供 context。在16位系统平台上不可用。

**Context**：数值表达式，用于标识由帮助文件的作者指定给某个帮助主题的上下文编号。如果已提供context，则必须提供helpfile。在16位系统平台上不可用。

MsgBox 函数有以下返回值：

|  |  |  |
| --- | --- | --- |
| 常数 | 值 | 按钮 |
| vbOK | 1 | 确定 |
| vbCancel | 2 | 取消 |
| vbAbort | 3 | 放弃 |
| vbRetry | 4 | 重试 |
| vbIgnore | 5 | 忽略 |
| vbYes | 6 | 是 |
| vbNo | 7 | 否 |

MsgBox 函数的用法示例：

<html>

<head>

<title>函数演示</title>

</HEAD>

<body>

<Script Language=VBScript>

Dim MyVar

MyVar = MsgBox ("Hello World!", 65, "MsgBox Example")

</Script >

</body>

</html>

**(8)、Eval函数**

它的作用是计算一个表达式的值并返回结果。表达式为[result= ]eval( expression )。其中，result为可选项，是一个变量，用于接受返回的结果。如果未指定结果，应考虑使用Execute 语句代替；expression为必选项，可以是包含任何有效VBScript表达式的字符串。在VBScript中，x = y可以有两种解释。第一种方式是赋值语句，将y的值赋予x。第二种解释是测试x和y是否相等。如果相等，result为Tr ue；否则result为False 。Eval方法总是采用第二种解释，而Execute语句总是采用第一种。下面的例子说明了Eval函数的用法:

Sub GuessANumber

Dim Guess, RndNum

RndNum = Int((100) \* Rnd(1) + 1) //Int函数是返回数字的整数部分，Rnd函数返回一个随机数

Guess = CInt(InputBox("Enter your guess:",,0)) //Cint函数返回表达式，此表达式已被转换为Integer子类型

Do

If Eval("Guess = RndNum") Then

MsgBox "祝贺你！猜对了！"

Exit Sub

Else

Guess = CInt(InputBox("对不起，请再试一次",,0))

End If

Loop Until Guess = 0

End Sub

**(9)、Replace函数**

该函数返回字符串，其中指定数目的某子字符串被替换为另一个子字符串。语法为

*Replace(expression, find, replacewith[, compare[, count[, start]]])*

其中参数为：

**Expression**:必选项。字符串表达式包含要替代的子字符串。

**Find**:必选项。被搜索的子字符串。

**Replacewith**:必选项。用于替换的子字符串。

**Start**:可选项。expression中开始搜索子字符串的位置。如果省略，默认值为1。在和count关联时必须用

**Count**:可选项。执行子字符串替换的数目。如果省略，默认值为-1，表示进行所有可能的替换。在和start关联时必须用。

**Compare**:可选项。指示在计算子字符串时使用的比较类型的数值。有关数值，请参阅“设置”部分。如果省略,缺省值为0 ,这意味着必须进行二进制比较。

下面的示例利用Replace函数返回字符串：

Dim MyString

MyString = Replace("XXpXXPXXp", , "p", , "Y") '二进制比较从字符串左端开始。返回"XXYXXPXXY"。

MyString = Replace("XXpXXPXXp", , "p", , "Y", , '文本比较从第三个字符开始。返回"YXXYXXY"。3，, , -1, , 1)