完美 Excel 技术系列(11)

Excel VBA 文件操作技术大全

完美 Excel 出品
微信公众号：*excelperfect*

内容提要

在电脑中，我们将信息保存为一个个文件，并通过文件夹来组织文件，将不同的文件整理归类起来，方便文件的查找和使用。在 VBA 程序开发中，经常会遇到文件操作。VBA 提供了一些专门用于处理文件和文件夹的函数，以及对象模型。

本电子书《**Excel VBA 文件操作技术大全**》主要讲述使用 VBA 操作文件和文件夹的相关知识，尽量不涉及对文件中具体内容的操作。

目录

温馨提示：

在完美 Excel 微信公众号中发送消息：**文件操作**，即可获得本电子书文档下载链接和密码。

如果您对本书有什么建议或者还有什么好的示例，欢迎前往**完美 Excel** 微信公众号沟通交流。

欢迎分享本电子书，让更多的人方便地得到所需要的知识。

本书内容日首发于
[完美 Excel]微信公众号 *excelperfect*
原标题为
VBA 进阶 | 文件操作

VBA 提供了一些专门用于处理文件和文件夹的函数，以及对象模型，下面我们来详细介绍这些内容。

CurDir 函数

本节介绍 CurDir 函数。

例如，代码：

```
Sub test()
    Debug.Print CurDir
End Sub
```

将在立即窗口中输出当前工作簿所在文件夹路径，如下图 1 所示。

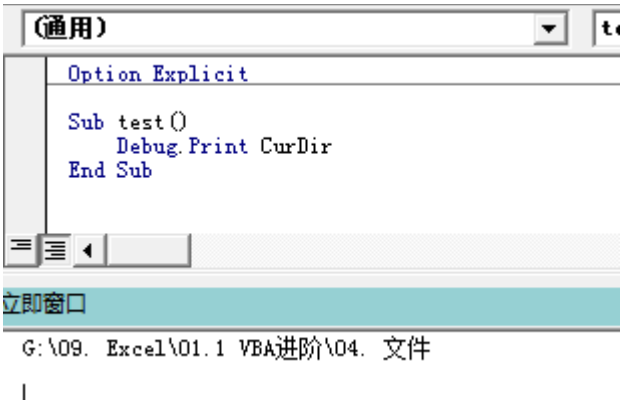

图 1

CurDir 函数用于获取当前文件夹的路径名称，其语法为：

```
CurDir([drive])
```

其中：

- 参数 drive 可选，代表所指定的驱动器名称的字符串。若省略该参数或者是零长字符串（""），则表示当前驱动器。
- 参数 drive 可以是一个表示驱动器名的带有冒号或者不带有冒号的字母，例如，“C”和“C:”对于参数 drive 来说都是有效值。
- 如果参数 drive 无效，则产生运行时错误：设备不可用。
- CurDir 函数的返回值代表文件的路径，Variant 型。若要返回字符串型的文件路径，则使用 CurDir$函数。例如，代码：Left(CurDir$, 1)返回当前工作簿所在驱动器的盘符，对于图 1 所示的工作簿来说，返回 G。
- 因为CurDir只能接受一个字符的字符串，所以不可以使用网络驱动器名称、共享名称等。

Dir 函数

Dir 函数返回文件或者文件夹的名称，常用来检查磁盘中是否存在指定的文件或文件夹，如果不存在，那么返回空字符串（""）。

Dir 函数的语法如下：

```
Dir[(pathname[,attributes])]
```

其中，参数 pathname 为可选参数，是代表文件或文件夹的名称的字符串，包括驱动器名称。参数 attributes 可以使用下图 1 列出的任一常量或者数值，来指定文件的属性。

常量	值	属性含义
vbNormal	0	普通文件
vbHidden	2	隐藏文件
vbSystem	4	系统文件
vbVolume	8	卷标。若指定了该属性，则会忽略其他所有属性
vbDirectory	16	目录或文件夹

图 2

attributes 常量能够组合在一起产生相应的组合属性进行匹配，例如 vbHidden+vbDirectory 组合表示隐藏目录。对象浏览器和 VBA 列表中还给出了其他几个常量（如 vbReadOnly、vbArchive），它们能作为 attributes 参数的值，但却不能在 Windows 平台上使用，也不能影响函数的操作。

尽管参数 pathname 为可选参数，但在第一次调用 Dir 函数时必须包含参数 pathname。而且如果指定了参数 attributes，则同时必须指定参数 pathname。此外，一旦 Dir 函数返回一个零长字符串，以后再次调用 Dir 函数之前必须指定参数 pathname，否则将产生错误“无效的过程调用或参数”。

Dir 函数返回字符串子类型的变体，而 Dir$函数返回字符串数据类型。如果没有发现与传递给函数的模式或文件属性相匹配的文件，则返回零长字符串（""）。

例如，下面的指令：

```
Dir("C:\",vbNormal)
```

返回指定文件夹 C：中第一个文件的名称。常量参数 vbNormal 指除隐藏、卷标、目录、文件夹或系统文件之外的任何文件。

Dir 函数允许在指定的文件路径名称中使用通配符返回多个文件，其中星号（*）代表多个字符，问号（？）代表单个字符。

示例 1：列出指定目录中的文件名称

下面的代码列出指定文件夹中所有文件的名称。

```
Sub ListFiles()

    Dim strFile As String

    Dim strPath As String


    strPath = InputBox("请输入路径名称,例如 C:\MyDirectory")

    If Right(strPath, 1) <> "\" Then strPath = strPath & "\"


    strFile = Dir(strPath & "*.*")

    If strFile <> "" Then Debug.Print "在文件夹" & strPath & "中的文件有:"

    Debug.Print LCase$(strFile)

    If strFile = "" Then

        MsgBox "没有找到文件."

        Exit Sub

    End If

    Do While strFile <> ""

       strFile = Dir

      Debug.Print LCase$(strFile)

  Loop
```

```
End Sub
```

ListFiles 过程首先询问用户输入文件路径名称，如果输入的路径末尾没有反斜杠，则在其后添加反斜杠。接着，在指定的文件路径中搜索所有文件（*）。如果该文件夹中没有文件，则显示提示信息；如果存在文件，则将所有文件名显示在立即窗口中。

示例 2：获取并在工作表中输入文件名称

下面的过程获取 C 盘根目录下的所有文件名称并将其写入工作表 Sheet1 中：

```
Sub GetFiles()

    Dim strFile As String

    Dim iRow As Integer

    iRow = 1

    With Worksheets("Sheet1").Range("A1")

        strFile = Dir("C:\", vbNormal)

        .Value = strFile

        Do While strFile <> ""

            strFile = Dir

            .Offset(iRow, 0).Value = strFile

            iRow = iRow + 1

        Loop

    End With

End Sub
```

示例 3：在列表框中放置指定类型的文件

下面的代码在列表框中列出指定目录 C:\Windows 中指定类型为 txt 的所有文件：

```
Private Sub CommandButton1_Click()
```


```
    Dim sFilename As String
    Dim sPath As String

    sPath = "C:\Windows\*.txt"
    sFilename = Dir$(sPath)

    Do While sFilename <> ""
        ListBox1.AddItem sFilename
        sFilename = Dir$
    Loop
End Sub
```

运行代码后的效果如图 3 所示。

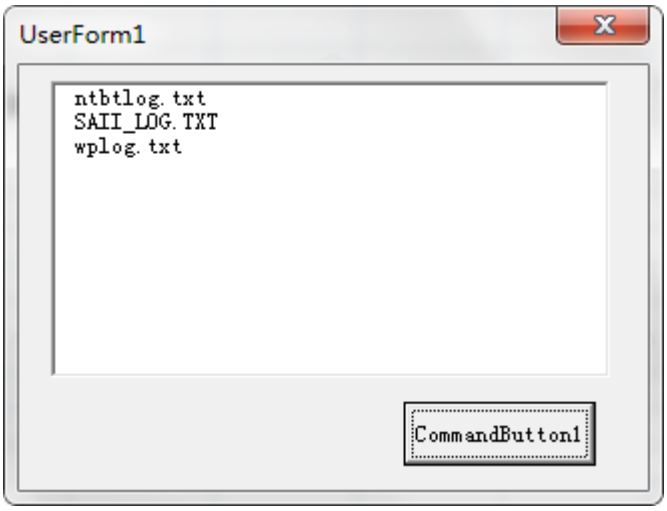

图 3

上面的示例在用户窗体中使用一个命令按钮和一个列表框控件，说明如何在 VBA 应用程序中使用 Dir 函数。

示例 4：判断文件是否存在

Dir 函数是惟一不需要创建错误处理程序而能够判断文件是否存在的方法。使用 Dir 函数判断文件是否存在很简单，只需要将测试文件名传递给该函数即可。如果 Dir 函数返回一个零长字符串，则表明文件不存在，如下面的代码：

```
Public Function FileExists(sPath As String) As Boolean

   If Dir(sPath) <> "" Then

      FileExists = True

   Else

      FileExists = False

   End If

End Function
```

关于 Dir 函数的进一步说明

Dir 函数一次只能返回一个文件名。如果要获得多个与参数 pathname 相匹配的文件，那么第一次调用 Dir 函数时必须使用必要的参数，而在以后调用 Dir 函数时则不再使用参数。如果没有一个文件名与初始的设置相匹配，那么函数将返回一个零长字符串。一旦 Dir 函数返回一个零长字符串，以后再次调用 Dir 函数时就必须指定参数 pathname，否则将产生错误。

Dir 函数返回的文件名在文件地址分配表中按顺序显示。如果需要以特定的顺序显示文件名，则必须在排序前先把文件名存储在一个数组中，然后再按需要的顺序排列。

Dir 函数保存它在调用之前的状态，这意味着不能递归调用 Dir 函数。例如，如果 Dir 函数返回一个目录的名称，那么就不能调用 Dir 函数迭代该目录下的文件并返回到原来的目录。

如果将参数 attributes 只设置为 vbDirectory，则该函数的作用将与原来有所不同。如果参数 pathname 使用了通配符，则该函数仅仅返回与搜索条件相匹配的第一个目录。如果不提供新的参数 pathname 而连续调用 Dir 函数，则按文件的顺序返回文件名。

如果调用 Dir 函数返回的文件名不止一个，则必须提供一个显式的文件规范。即如果希望获

得 Windows 目录下所有文件的名称，不能使用下面的语句：

```
strFile=Dir("C:\Windows",vbNormal)
```

而应该使用如下列语句指定的参数 pathname 来调用 Dir 函数：

```
strFile=Dir("C:\Windows\*.*",vbNormal)
```

Dir 函数和 Dir$函数仅仅返回文件名称，而不提供文件大小、日期和数据、时间戳或者文件属性等信息。此时，利用 File System 对象模型中的 File 对象的 Attributes 属性很容易获得这些信息。

使用 Dir 函数的一个关键在于要清楚各种属性常量如何影响文件或者 Dir 函数如何返回文件。在默认情况下，Dir 函数返回一个普通文件（即属性没有设置为隐藏或者系统的文件）。vbHidden 属性返回隐藏文件或者常规文件，而不是返回系统文件或者隐藏的系统文件。vbSystem 属性返回系统文件或常规文件，而不是返回隐藏文件或隐藏的系统文件。vbSystem+vbHidden 属性则返回任何文件，无论是常规文件、隐藏文件还是系统文件。

Name 函数

Name 函数用于更改文件或者文件夹的名称，其语法为：

```
Name old_pathname As new_pathname
```

其中：

- 参数 old_pathname 代表希望重命名的文件或文件夹的路径和名称
- 参数 new_pathname 用于指定文件或文件夹的位置和新名称。
- 使用 Name 函数，可以将文件从一个文件夹移至另一个文件夹，但是该函数不能移动文件夹。

在使用 Name 函数时，应该注意以下几点：

- 参数 old_pathname 所指定的文件必须存在，否则会发生错误：文件未找到。
- 参数 new_pathname 所指定的文件名称不能与文件夹中已经存在的文件名称相同。例如，在 C 盘中已经存在文件名 testNewFile.txt，下面的语句：

```
Name "C:\testOldFile.txt" As "C:\testNewFile.txt"
```

将出现错误：文件已存在。

- 如果参数 new_pathname 所指定的文件名称与参数 old_pathname 所指定的文件名称不同，则在移动文件的同时将该文件名称更改为参数 new_pathname 所指定的新名称。例如：

```
Name "C:\testOldFile.txt" As "D:\testNewFile.txt"
```

将文件 testOldFile.txt 移至 D 盘并将其名称更改为 testNewFile.txt。

- 如果参数 new_pathname 和参数 old_pathname 所指定的文件路径不同但文件名称相同，则 Name 函数将指定的文件移动到参数 new_pathname 指定的文件路径下并保持文件名称不变。
- 不能够更改已打开的文件的名称。
- 在更改文件名称时，文件名称中不能包含通配符“*”或者“？”。
- 在更改文件名称时，new_pathname 和 old_pathname 不能在不同的驱动器上。
- Name 语句可以将文件从一个文件夹移到另一个文件夹中，同时也可以改变文件的名

字。如果 new_pathname 中指定的文件夹存在而且和 old_pathname 中给出的不同，文件就会被移到 new_pathname 指定的文件夹中。如果 new_pathname 和 old_pathname 中的文件名也不同，这个文件就会被改名。但是，以这种方式移动对象只对文件有效，不能使用 Name 语句来移动文件夹。

ChDir 语句和 ChDrive 语句

使用 ChDir 语句来更改默认的文件夹，其语法为：

```
ChDir Path
```

其中，参数 Path 是指定新的默认文件夹名称的字符串，可以包含驱动器名。如果在参数 Path 中没有包括驱动器名，那么默认文件将会更改为当前驱动器。注意，此时默认文件夹的改变并不会改变当前的默认驱动器。例如，如果当前驱动器为 C:\，使用下列语句：

```
ChDir "D:\MyFolder"
```

将驱动器 D:中的当前文件夹改为“D：\MyFolder”，但当前驱动器仍然是 C：\。

如果没有找到参数 Path 指定的路径，则会导致错误“找不到路径”；如果参数 Path 指向网络上的另一台计算机，则会导致错误“路径/文件访问错误”。

要更改当前默认的驱动器，则使用 ChDrive 语句，其语法为：

```
ChDrive Drive
```

其中，参数 Drive 是指定新的默认驱动器的字符串，用来设置默认驱动器的名称（A～Z）。如果指向一个并不存在的驱动器，则会出现信息框“设备不可用”；如果 Drive 指定为零长的字符串，则不会改变驱动器；如果 Drive 包含不止一个字符，则只有第一个字符代表用来设置的驱动器。

示例：确定指定的驱动器是否可用

下面的自定义函数 IsAvailableDrive 用来确定指定的驱动器是否可用。

```
Private Function IsAvailableDrive(sDrive As String) As Boolean
    On Error Resume Next
    Dim sCurDrv As String
    '获取代表当前驱动器的字母
    sCurDrv = Left$(CurDir$, 1)
    '试着改变驱动器
```


```
    ChDrive sDrive
    '根据是否有错误发生判断驱动器是否可用
    If Err.Number = 0 Then
        '没有发生错误-该驱动器可用
        IsAvailableDrive = True
    Else
        '有错误发生-该驱动器不可用
        IsAvailableDrive = False
    End If
    '恢复驱动器默认设置
    ChDrive sCurDrv
End Function
```

使用下面的测试程序来测试该函数：

```
Sub test()
    Dim sDrv As String
    sDrv = "T" '改为您要判断的驱动器字母或者改为由用户输入
    If IsAvailableDrive(sDrv) Then
        ChDrive sDrv
    Else
        MsgBox "不能够使用驱动器" & sDrv & ":\"
    End If
End Sub
```

该函数是一个通用函数，在需要使用 ChDrive 语句的情形时，能够减少代码的数量。

MkDir 语句和 RmDir 语句

使用 MkDir 语句可以创建一个新文件夹，其语法为：

```
MkDir Path
```

其中，Path 指定希望创建的新文件夹名称，若省略驱动器名称，则在当前驱动器上创建新文件夹。

在调用 MkDir 语句之后，VB 并不自动把新建文件夹视作当前文件夹，如果要这样，必须使用 ChDir 语句。

使用 RmDir 语句删除不需要的文件夹，其语法为：

```
RmDir Path
```

其中，Path 指定希望删除的文件夹的名称，若省略驱动器名称，则删除当前驱动器中存在的相同名称的文件夹。若指定的文件夹不存在，则会出现错误信息“路径未找到”。

注意，RmDir 语句只能删除空文件夹，也就是说，如果文件夹中包含有文件，那么使用 RmDir 语句删除文件夹时会导致错误。此时，应该先使用 Kill 语句删除文件夹中的文件，再使用 RmDir 删除该文件夹。

使用 Kill 语句和 ReDim 语句的删除是不可逆的，因为这些语句不会把删除掉的文件放到回收站中。

为了删除文件夹，可以递归调用 Dir 函数引导到文件夹中的子文件夹。要注意的是，该函数可以每次在调用之前保存状态信息。

示例：删除文件夹以及其中的文件

下面的过程删除一个文件夹中的所有文件并移除它的子文件夹，如果那些文件夹中还包含文件或文件夹，则可以通过递归调用该过程来删除它们，直到把所有的子文件夹以及它们的文

件都删除为止。

```
Private Sub RemoveFolder(ByVal strFolder As String)
   Static blnLowerLevel As Boolean '递归调用-不需要提示用户
   Dim blnRepeated As Boolean '在重复调用中使用 Dir 状态信息
   Dim strFile As String '在 strFolder 中包含文件/目录
   '删除所有文件
   Do
      strFile = Dir(strFolder & "\*.*", vbNormal Or vbHidden Or vbSystem)
      If strFile <> "" Then
         If Not blnLowerLevel Then
            If MsgBox("删除目录中的文件" & _
                  strFolder & "?", _
                  vbQuestion Or vbOKCancel, _
                  "确定删除文件") _
                  = vbCancel Then Exit Sub
         End If
         strFile = strFolder & "\" & strFile
         Kill strFile
      End If
   Loop While strFile <> ""
   '删除所有目录
   Do
      If Not blnRepeated Then
         strFile = Dir(strFolder & "\*.*", vbDirectory)
         blnRepeated = True
      Else
         strFile = Dir(, vbDirectory)
      End If
      If strFile <> "" And strFile <> "." And strFile <> ".." Then
         If Not blnLowerLevel Then
            blnLowerLevel = True
            If MsgBox("删除 " & _
```

```
                        strFolder & " 的子目录吗?", _
                        vbQuestion Or vbOKCancel, _
                        "确认删除目录") _
                        = vbCancel Then Exit Sub
            End If
            RemoveFolder strFolder & "\" & strFile
            blnRepeated = False
        End If
    Loop While strFile <> ""
    RmDir strFolder
End Sub
```

在 File System 对象模型包含 Folders 集合对象和 Folder 对象，它们比内置的文件夹管理函数更加灵活、方便，后面会详细讲解。

Kill 语句

Kill 语句可以删除文件夹中的文件，其语法为：

```
Kill Pathname
```

其中，Pathname 指定希望删除的一个或多个文件的名称，可以包括驱动器和文件夹名称，可以使用通配符（*或？）来快速删除文件，但不能删除已经打开的文件或者被设置成只读属性的文件。

下面的程序代码删除 C:\MyDir 文件夹中的文件，最后删除该文件夹自身：

```
Sub RemoveFolder()
    Dim strFolder As String
    Dim strFile As String

    '将文件夹名称赋值给变量,注意加上末尾处的反斜杠
    strFolder = "C:\MyDir\"
    strFile = Dir(strFolder, vbNormal)
    Do While strFile <> ""
        Kill strFolder & strFile
        strFile = Dir
    Loop
    RmDir strFolder
End Sub
```

在有些操作系统环境下，删除的文件并不放在回收站里面，但是下面的代码段说明如何利用 Shell32.DLL 中的 FileOperation API 函数将文件移动到回收站中。

```
'声明文件操作结构
Type SHFILEOPSTRUCT
     hWnd As Long
     wFunction As Long
```


```
    pFrom As String
    pTo As String
    fFlags As Integer
    fAborted As Boolean
    hNameMaps As Long
    sProgress As String
End Type

'声明删除操作需要的两个常数
Private Const FO_DELETE = &H3
Private Const FO_FLAG_ALLOWUNDO = &H40
'声明 API 调用函数
Declare Function SHFileOperation Lib "shell32.dll" _
        Alias "SHFileOperationA" _
        (lpFileOp As SHFILEOPSTRUCT) As Long

Public Function WinDelete(sFileName As String) As Long
    '创建文件操作结构的副本
    Dim SHFileOp As SHFILEOPSTRUCT
    '需要以 Null 结尾的字符串
    sFileName = sFileName & vbNullChar
    '将相关值赋给结构
    With SHFileOp
        .wFunction = FO_DELETE
       .pFrom = sFileName
        .fFlags = FO_FLAG_ALLOWUNDO
    End With
    '将结构传递给 API 函数
    WinDelete = SHFileOperation(SHFileOp)
End Function
```

如果要删除文件夹，那么应使用 RmDir 语句。

FileCopy 语句

FileCopy 语句用于在文件夹之间复制文件，其语法为：

```
FileCopy source,destination
```

两个参数都是必需的参数。其中，第一个参数 source 指定要复制的文件名称，可以包含驱动器名称，String 类型。第二个参数 destination 指定复制到的地点，同样可以包含驱动器和文件夹的名称，String 类型。不能复制当前已经打开的文件。

如果没有在参数 source 和参数 destination 中指定驱动器或文件夹，则认为文件在当前的驱动器或文件夹中。在使用 FileCopy 语句时，只输入一个 destination 路径是不够的，必须提供一个文件名，即使它与 source 的内容相同也可以，否则会产生运行时错误“路径/文件访问错误”。

FileCopy 是一个语句而不是函数，没有返回值。因此，可以假定如果调用 FileCopy 语句时没有出现错误，就认为成功复制了文件，所以应该为 FileCopy 语句编写良好的错误处理程序。注意，如果 destination 指定的文件已经存在，就会直接覆盖该文件而不会预先提示用户。

下面的代码将用户指定的文件复制到名为“C:\MyDir”的文件夹中：

```
Sub CopyToMyDir()
     Dim strFolder As String
     Dim strSource As String
     Dim strDestination As String
     Dim strMsg1 As String
     Dim strMsg2 As String
     Dim iP As Integer
     Dim iSS As Integer
     Dim i As Long

     On Error GoTo ErrorHandler
```

```
    strFolder = "C:\MyDir"
    strMsg1 = "该文件已存在于文件夹中."
    strMsg2 = "复制到"
    iP = 1
    i = 1

    '获取文件名
    strSource = Application.GetOpenFilename
    '如果单击取消则不做任何操作
    If strSource = "False" Then Exit Sub
    '获取变量 strSource 中的反斜杠总数
    Do Until iP = 0
        iP = InStr(i, strSource, "\", 1)
        If iP = 0 Then Exit Do
        iSS = iP
        i = iP + 1
    Loop
    '创建目标文件名
    strDestination = strFolder & Mid(strSource, iSS, Len(strSource))
    '使用该名称创建新文件夹
    MkDir strFolder
    '检查指定的文件是否已存在于目标文件夹中
    If Dir(strDestination) <> "" Then
        MsgBox strMsg1
    Else
        '复制所选文件到 C:\MyDir 文件夹中
        FileCopy strSource, strDestination
        MsgBox strSource & " " & strMsg2 & " " & strDestination
    End If
    Exit Sub
ErrorHandler:
    If Err = "75" Then
```


```
        Resume Next
    End If
    If Err = "70" Then
        MsgBox "不可以复制已经打开的文件."
    End If
End Sub
```

CopyToMyDir 过程使用 GetOpenFilename 方法从用户处获取文件名称，此时将出现内置的“打开”对话框，从中选择文件，如果用户单击“取消”按钮，则返回值“False”，从而结束程序；如果用户单击“打开”按钮，则所选文件的完整路径和名称就被赋值给变量 strSource。因为仅需要文件名称，所以使用 Do Until 循环找到最后一个反斜杠在变量 strSource 中的位置。接着，为 FileCopy 语句的第二个参数准备字符串，并将其值赋给变量 strDestination，该变量存储的字符串为目标文件夹加反斜杠加用户指定的文件名。如果目标文件夹不存在，则使用 MkDir 函数创建新文件夹。然后，程序检查所选择的文件是否已经存在于目标文件夹中，如果文件已存在，则为用户给出提示信息；如果文件不存在且该文件没有被打开则将该文件复制到指定的文件夹并显示相应的信息。如果文件被打开，则会出现错误从而显示相应的提示。

认识 Windows Scripting Host

Windows Scripting Host，简称为 WSH，是一种脚本语言，可以自动运行命令，能够控制 Windows 操作系统和应用程序，也能够从操作系统中获取信息。WSH 是名为 wshom.ocx 的 ActiveX 控件文件，通常该文件会自动安装在 Windows System32 文件夹中。WSH 有自己的对象层次模型。

要使用 WSH 对象模型，必须先添加对“Microsoft Scripting Runtime”库的引用。在 VBE 中，单击菜单“工具”——“引用”，在“引用”对话框中找到“Microsoft Scripting Runtime”项并选中其前面的复选框，如下图 4 **错误!未找到引用源。**所示。

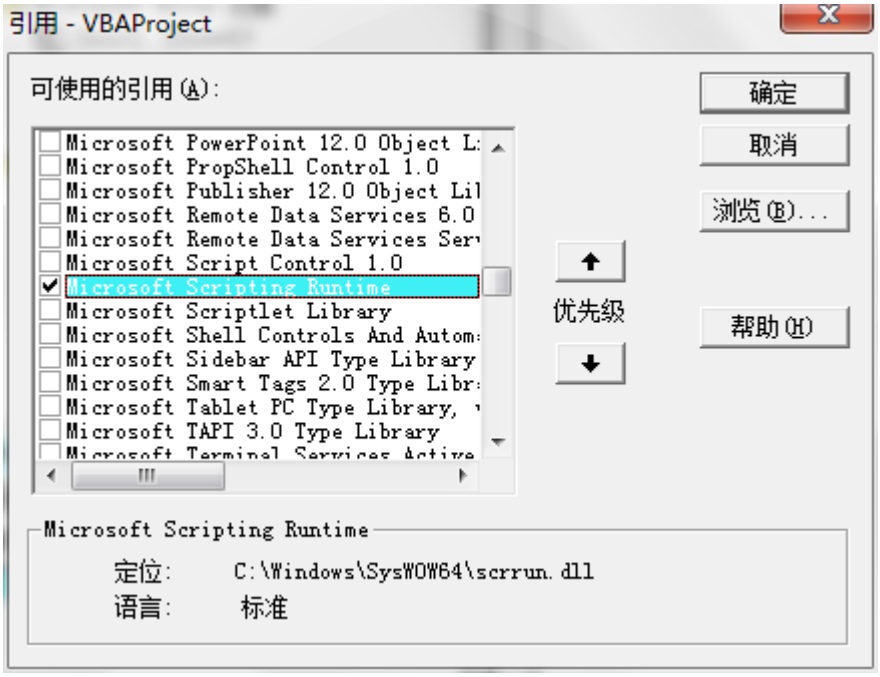

图 4

此时，可以在对象浏览器的“所有库”组合框中选择“Scripting”后，搜索并查看相关帮助信息，如下图 5 所示。

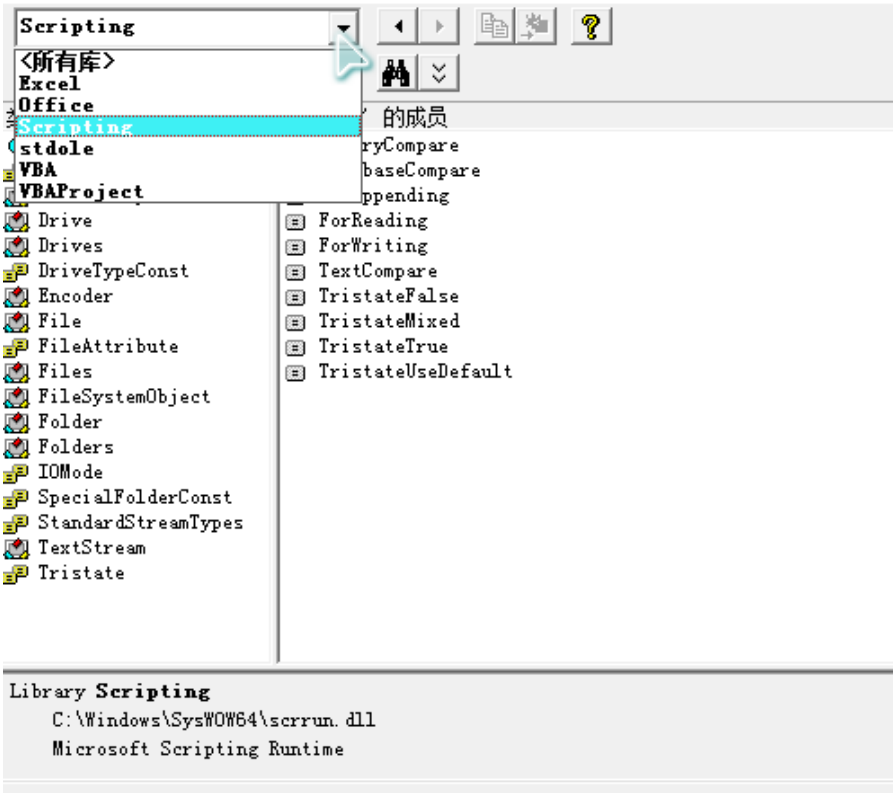

图 5

File System 对象模型是 Scripting Runtime 库的一部分，如下图 6 所示，可以用于查找、创建、删除或者用其他方法操作文件夹和文本文件。

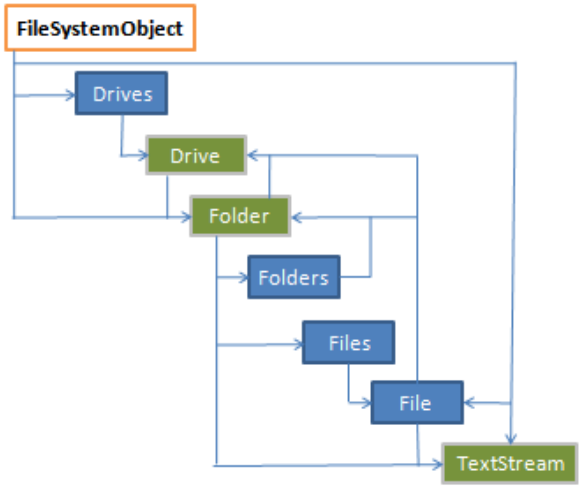

图 6

本书后续内容将逐步分别介绍 FileSystemObject 对象、File 对象、Files 集合、Folder 对象、Folders 集合、Drive 对象、Drives 集合，以及使用 WSH 进行其他操作。

FileSystemObject 对象及其方法和属性

FileSystemObject 对象位于 File System 对象模型的最高层，并且是该层次中惟一可以在外部创建的对象，也就是说它是惟一能使用 New 关键字的对象。

FileSystemObject 对象有许多用来操作文件系统的方法和属性。下面先看一个例子，如下面的代码：

```
Sub FileInfo()
    Dim fs As Object
    Dim objFile As Object
    Dim strMsg As String

    Set fs = CreateObject("Scripting.FileSystemObject")
    Set objFile = fs.GetFile("C:\Windows\System.ini")
    strMsg = "文件名:" & objFile.Name & vbCrLf
    strMsg = strMsg & "硬盘:" & objFile.Drive & vbCrLf
    strMsg = strMsg & "创建日期:" & objFile.DateCreated & vbCrLf
    strMsg = strMsg & "修改日期:" & objFile.DateLastModified & vbCrLf
    MsgBox strMsg, , "文件信息"
End Sub
```

FileInfo 过程首先使用 CreateObject 函数创建一个 FileSystemObject 对象，用来访问计算机的文件系统。然后，使用 GetFile 方法创建一个 File 对象并返回对 System.ini 文件的引用。接着，利用 File 对象的 Name 属性、Drive 属性、DateCreated 属性、DateLastModified 属性返回文件的相应信息。

下面的系列文章将详细介绍 FileSystemObject 对象的方法和属性。

BuildPath 方法

BuildPath 方法的语法为：

```
oFileSysObj.BuildPath (Path,Name)
```


其中：

- oFileSysObj 为任何能够返回 FileSystemObject 对象的对象变量。
- 参数 Path 必需，指定驱动器或文件夹路径，String 类型，可以是绝对路径也可以是相对路径，不一定要包含驱动器名。
- 参数 Name 必需，指定附加在 Path 后的文件夹或文件路径，String 类型。
- 参数 Path 或 Name 都不一定要求是当前已经存在的路径或文件夹。

BuildPath 方法通过合并参数 Path 和文件夹或文件名生成一个字符串，并且在必要的地方加上正确的主机系统路径分隔符。该方法不能检验新的文件夹或文件名的有效性。

与人工合并两个字符串相比，使用 BuildPath 函数的惟一好处就是它能够选择正确的路径分隔符。

FileExists 方法

FileExists 方法用于判断指定的文件是否存在，若存在则返回 True。其语法为：

```
oFileSysObj.FileExists(FileSpec)
```

其中：

- oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象。
- 参数 FileSpec 必需，代表文件的完整路径，String 类型，不能包含有通配符。

如果用户有充分的权限，FileSpec 可以是网络路径或共享名，例如：

```
If ofs.FileExists("\\TestPath\Test.txt") Then
```

示例

```
Sub IfFileExists()
    Dim fs As Object
    Dim strFile As String
    Set fs = CreateObject("Scripting.FileSystemObject")
    strFile = InputBox("请输入文件的完整名称:")
    If fs.FileExists(strFile) Then
        MsgBox strFile & "已经找到."
```

```
    Else
        MsgBox "该文件不存在."
    End If
End Sub
```

GetFile 方法

GetFile 方法用来返回一个 File 对象。其语法为：

```
oFileSysObj.GetFile (FilePath)
```

其中：

- oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。
- 参数 FilePath 必需，指定路径和文件名，String 类型。可以是绝对路径或相对路径。如果 FilePath 是一个共享名或网络路径，GetFile 确认该驱动器或共享名是 File 对象创建进程的一部分。如果参数 FilePath 指定的路径的任何部分不能连接或不存在，就会产生错误。
- GetFile 方法返回的是 File 对象，而不是 TextStream 对象。File 对象不是打开的文件，主要是用来完成如复制或移动文件和询问文件的属性之类的方法。尽管不能对 File 对象进行写或读操作，但可以使用 File 对象的 OpenAsTextStream 方法获得 TextStream 对象。
- 要获得所需的 FilePath 字符串，首先应该使用 GetAbsolutePathName 方法。如果 FilePath 包含网络驱动器或共享，可以在调用 GetFile 方法之前用 DriveExists 方法来检验所需的驱动器是否可用。
- 因为在 FilePath 指定的文件不存在时会产生错误，所以应该在调用 GetFile 之前调用 FileExists 方法确定文件是否存在。
- 必须用 Set 语句将 File 对象赋给一个局部对象变量。

GetFileName 方法

GetFileName 方法返回给定路径的文件名称部分。其语法为：

```
oFileSysObj.GetFileName (Path)
```


其中：

- oFileSysObj 表示任何能够返回 FileSystemObject 对象的对象变量。
- 参数 Path 必需，指定路径说明，String 类型。如果不能从给定的 Path 确定文件名，则返回一个零长字符串（""）。Path 可以为绝对路径或相对路径。
- GetFileName 方法不能检验 Path 中是否存在指定的文件。Path 可以为网络驱动器或共享。GetFileName 本身不具有智能，它认为字符串中不属于驱动器说明的最后部分就是一个文件名，更像是一个字符串处理函数而不是对象处理方法。

GetFileVersion 方法

GetFileVersion 方法返回文件的版本。

CopyFile 方法详解

CopyFile 方法用来复制文件，将文件从一个文件夹复制到另一个文件夹。其语法为：

```
oFileSysObj.CopyFile Source,Destination [,OverwriteFiles]
```

或：

```
oFileSysObj.CopyFile 被复制的源文件,复制到的目标文件 [,是否覆盖现有文件]
```

说明：

1. oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2. 参数 Source，必需，指定要复制的文件的路径和名称，String 类型。

3. 参数 Destination，必需，代表要将文件复制到哪里，其目标路径和文件名（可选），String 类型。

4. 参数 OverwriteFiles 可选，表示是否覆盖现有文件，True 表示覆盖，False 表示不覆盖，Boolean 类型，默认值为 True。

5. 参数 source 中源路径可以是绝对路径或相对路径，源文件名可包含通配符但源路径不能。在参数 Destination 中不能包含通配符。

6. 如果目标路径或文件设置为只读，则无论参数 OverwriteFiles 的值如何，都将无法完成 CopyFile 方法。如果参数 OverwriteFiles 设置为 False 且 Destination 指定的文件已经存在，则会产生一个运行时错误“文件已经存在”。如果在复制多个文件时出现错误，CopyFile 方法将立即停止复制操作，该方法不具有撤销发生错误前文件复制操作的返回功能。如果用户

有充分的权限，那么 source 或 destination 可以是网络路径或共享名，例如：

```
CopyFile “\\NTSERV1\RootTest\test.txt”,”C:\RootOne”
```

7. CopyFile 方法可以复制一个保存在特定文件夹中的文件。如果文件夹本身有包含文件的子文件夹，则使用 CopyFile 方法不能复制这些文件，应该使用 CopyFolder 方法。

示例

下面的示例代码将 C 盘下的文件 logExcel.csv 复制到 C 盘下的 Program Files 文件夹中：

```
Sub CopyFile()
    '声明变量
    Dim fs As Object
    Dim strFile As String
    Dim strNewFile As String

    '指定要复制的文件
    strFile = "C:\logExcel.csv"
    '指定要将文件复制到的文件路径
    '其中文件名可选
    strNewFile = "C:\Program Files\logExcel.csv"

    '创建 FileSystemObject 对象
    Set fs = CreateObject("Scripting.FileSystemObject")

    '复制操作
    fs.CopyFile strFile, strNewFile

    '给出提示消息
    MsgBox "已经创建了指定文件的副本."

    '释放对象变量
```


```
    Set fs = Nothing
End Sub
```

CreateTextFile 方法

CreateTextFile 方法创建一个新的文件并返回其 TextStream 对象，其语法为：

```
oFileSysObj.CreateTextFile Filename [,Overwrite[,Unicode]]
```

其中：

- oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。
- 参数 Filename 必需，代表任何有效文件名，String 类型。在 Filename 中不允许使用通配符。Filename 可以是相对路径也可以是绝对路径，如果没有指定路径，则使用应用程序的当前驱动器或文件夹作为路径。如果指定的路径不存在，则该方法将失败。
- 参数 Overwrite 可选，作为一个标志，指定是否覆盖一个具有相同文件名的现有文件，Boolean 类型。默认值为 False。
- 参数 Unicode 可选，作为一个标志，指明用 Unicode 格式还是 ASCⅡ格式写文件，Boolean 类型。如果设置为 True，则以 Unicode 格式创建文件，否则创建一个 ASCⅡ文本文件。默认值为 False。

说明：

- 只有写操作才能使新创建的文本文件自动打开，如果以后希望读取该文件，则必须选关闭它再以读模式重新打开该文件。
- 如果参数 Filename 中指定的路径设置为只读，则不论参数 Overwrite 的值如何，CreateTextFile 方法都将失败。
- 如果用户有充分的权限，那么参数 Filename 可以是网络路径或共享名，例如：CreateTextFile “\\NTSERV1\RootTest\myFile.doc”
- 必须使用 Set 语句将 TextStream 对象赋值给局部对象变量。

MoveFile 方法

MoveFile 方法用来移动文件，将文件从一个文件夹移动到另一个文件夹。其语法为：

```
oFileSysObj.MoveFile source,destination
```

其中：

- oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。
- 参数 source 必需，指定要移动的文件的路径，String 类型。
- 参数 destination 必需，指定要将文件移动到哪里，即文件移动操作中的目标位置的路径，String 类型。
- 如果 Source 包含通配符或者 destination 以路径分隔符结尾，则认为 destination 是一个路径，否则认为 destination 的最后一部分是文件名。
- 如果目标文件已经存在，则将出现一个错误。
- source 可以包含通配符，但只能出现在它的最后一部分中。
- destination 参数不能包含通配符。
- source 或 destination 可以是相对路径或绝对路径，可以是网络路径或共享名。
- MoveFile 方法在开始操作前先解析 source 和 destination 这两个参数。

DeleteFile 方法

DeleteFile 方法删除指定的一个或多个文件。其语法为：

```
oFileSysObj.DeleteFile FileSpec[,Force]
```

其中：

- oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。
- 参数 FileSpec 必需，代表要删除的单个文件或多个文件的名称和路径，String 类型，可以在路径的最后部分包含通配符，可以为相对路径或绝对路径。如果在 FileSpec 中只有文件名，则认为该文件在应用程序的当前驱动器和文件夹中。
- 参数 Force 可选，如果将其设置为 True，则忽略文件的只读标志并删除该文件，Boolean 类型，默认值为 False。

说明：

- 如果指定要删除的文件已经打开，该方法将失败并出现一个“Permission Denied”错误。如果找不到指定的文件，则该方法失败。

- 如果在删除多个文件的过程中出现错误，DeleteFile 方法将立即停止删除操作，即不能删除余下的文件部分。该方法不具有撤销产生错误前文件删除操作的返回功能。
- 如果用户有充分的权限，源路径或目标路径可以是网络路径或共享名。例如：DeleteFile “\\NTSERV1\RootTest\MyFile.doc”
- DeleteFile 方法永久性地删除文件，并不把这些文件移到回收站中。

示例

下面的代码删除 C 盘中的 test.doc 文件。

```
Sub DeleteFile()

    Dim fs As FileSystemObject

    Set fs = New FileSystemObject

    fs.DeleteFile "C:\test.doc"

    MsgBox "删除了该文件."

End Sub
```

DriveExists 方法

DriveExists 方法用来判断在本地计算机或者网络上是否存在指定的磁盘，若存在则返回 True。其语法为：

```
oFileSysObj.DriveExists (DriveSpec)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 DriveSpec 必需，代表路径或驱动器名，String 类型。如果 DriveSpec 是一个 Windows 驱动器名，则其后面不需要跟冒号，例如“C”和“C:”是一样的。

说明：

1.DriveExists 方法不能返回可移动驱动器的当前状态，要实现这一目的，必须使用指定驱动器的 IsReady 属性。

2.如果用户有充分的权限，DriveSpec 可以是网络路径或共享名，例如：

```
If ofs.DriveExists(“\\NTESERV1\d$”) Then
```

3.在调用位于某驱动器上一个远程 ActiveX 服务器中的函数前，最好先使用 DriveExists 方法检测网络上是否存在该驱动器。

示例

下面的代码判断指定的盘是否存在并返回相应的信息。

```
Function DriveExists(disk)
    Dim fs As Object
    Dim strMsg As String
    Set fs = CreateObject("Scripting.FileSystemObject")
    If fs.DriveExists(disk) Then
        strMsg = "驱动器" & UCase(disk) & "盘已存在."
    Else
        strMsg = UCase(disk) & "盘未找到."
    End If
    DriveExists = strMsg
End Function
```

GetDrive 方法

GetDrive 方法返回 Drive 对象，即获得对指定驱动器的 Drive 对象的引用。其语法为：

```
oFileSysObj.GetDrive (drivespecifier)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 drivespecifier 必需，代表驱动器名、共享名或网络路径，String 类型。如果 drivespecifier 是一个共享名或网络路径，GetDrive 确认它是 Drive 对象创建进程的一部分，否则会产生运行时错误“找不到路径”。如果指定的驱动器没有连接上或者不存在，则会出现运行时错误“设备不可用”。

说明：

1.如果要从路径中导出 drivespecifier 字符串，应该首先用 GetAbsolutePathName 来确保驱动器是路径的一部分，然后在调用 GetDriveName 从全限定路径中提取出驱动器之前，用 FolderExists 方法检验路径是否有效，例如：

```
Dim oFileSys As New FileSystemObject

Dim oDrive As Drive
```


```
sPath=oFileSys.GetAbsolutePathName(sPath)
If oFileSys.FolderExists(sPath) Then
   set oDrive=oFileSys.GetDrive(oFileSys.GetDriveName(sPath))
End If
```

2.如果 driverspecifier 是一个网络驱动器或共享，在调用 GetDrive 方法之前，应该用 DriveExists 方法检验所需的驱动器是否可用。

3.必须用 Set 语句将 Drive 对象赋给局部对象变量。

示例

下面的代码获取指定的驱动器的相关信息：

```
Sub DriveInfo()
   Dim fs, disk, infoStr, strDiskName
   strDiskName = InputBox("输入驱动器盘符:", "驱动器名称", "C:\")
   Set fs = CreateObject("Scripting.FileSystemObject")
   Set disk = fs.GetDrive(fs.GetDriveName(strDiskName))
   infoStr = "驱动器:" & UCase(strDiskName) & vbCrLf
   infoStr = infoStr & "驱动器盘符:" & UCase(disk.DriveLetter) & vbCrLf
   infoStr = infoStr & "驱动器类型:" & disk.DriveType & vbCrLf
   infoStr = infoStr & "驱动文件系统:" & disk.FileSystem & vbCrLf
   infoStr = infoStr & "驱动器系列号:" & disk.SerialNumber & vbCrLf
   infoStr = infoStr & "字节的总大小:" & FormatNumber(disk.TotalSize / 1024, 0)
& "kb" & vbCrLf
   infoStr = infoStr & "驱动器中的自由空间:" & FormatNumber(disk.FreeSpace / 1024,
0) & "kb" & vbCrLf
   MsgBox infoStr, vbInformation, "驱动器信息"
End Sub
```


GetDriveName 方法

GetDriveName 方法返回一个包含硬盘名称或者网络共享名称的字符串。即返回给定路径的驱动器名，如果从给定的路径中不能确定驱动器名，则返回一个零长字符串（""）。其语法为：

```
oFileSysObj.GetDriveName (Path)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Path 必需，指定路径，String 类型。

说明：

1.GetDriveName 不能检验 Path 中是否存在指定的驱动器。

2.Path 可以是网络驱动器或共享。

示例

下面的代码返回参数中的驱动器名。

```
Function DriveName(disk)
    Dim fs As Object
    Dim strDiskName As String
    Set fs = CreateObject("Scripting.FileSystemObject")
    strDiskName = fs.GetDriveName(disk)
    DriveName = strDiskName
End Function
```

GetExtensionName 方法

返回给定路径中文件的扩展名。其语法为：

```
oFileSysObj.GetExtensionName (Path)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Path 必需，表示路径说明，String 类型。如果不能确定 Path 中的扩展名，则返回一

个零长字符串。

说明：

1.GetExtensionName 方法不能检验 Path 是否有效，Path 可以为网络路径或共享。

2.GetExtensionName 没有智能功能，它简单地解析一个字符串，并返回 Path 最后部分中最后一个点后的文本。

FolderExists 方法

FolderExists 方法可以判断指定的文件夹是否存在，若存在则返回 True。其语法为：

```
oFileSysObj.FolderExists(FolderSpec)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 FolderSpec 指定文件夹的完整路径，String 类型，不能包含通配符。

3.如果用户有充分的权限，FolderSpec 可以是网络路径或共享名，例如：

```
If ofs.FileExists ("\\NTSERV1\d$\TestPath\") Then
```

示例

下面的代码判断是否存在给定的文件夹：

```
Sub IfFolderExists()
    Dim fs As Object
    Set fs = CreateObject("Scripting.FileSystemObject")
    MsgBox fs.FolderExists("C:\Program Files")
End Sub
```

GetAbsolutePathName 方法

将相对路径转变为一个全限定路径（包括驱动器名），返回一个字符串，包含一个给定的路径说明的绝对路径。其语法为：

```
oFileSysObj.GetAbsolutePathName (Path)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Path 必需，代表路径说明，String 类型。

“.”返回当前文件夹的驱动器名和完整路径。“..”返回当前文件夹的父文件夹的驱动器名和路径。“filename”返回当前文件夹中的文件的驱动器名、路径及文件名。

所有相对路径名均以当前文件夹为基准。

如果没有明确地提供驱动器作为 Path 的一部分，就以当前驱动器作为 Path 参数中的驱动器。在 Path 中可以包含任意个通配符。

说明：

1.对于映射网络驱动器和共享而言，这种方法不能返回完整的网络地址，而是返回全限定的本地路径和本地驱动器名。

2.GetAbsolutePathName 不能检验指定路径中是否存在某个给定的文件或文件夹。

GetBaseName 方法

返回路径的最后部分的名称，不包含扩展名。其语法为：

```
oFileSysObj.GetBaseName (Path)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Path 必需，代表路径说明，String 类型。Path 中最后部分的文件扩展名不包含在返回的字符串中。

说明：

1.GetBaseName 方法不能检验 Path 中是否存在给定的文件或文件夹。

2.GetBaseName 方法没有舍去文件扩展名并返回 Path 的基本名称的智能功能。也就是说，它不能识别路径的最后部分是路径还是文件名。如果最后部分包括一个或多个点“.”，它仅仅删除最后一个占以及该点后的文本。所以如果 Path 为“.”，GetBaseName 方法返回一个空字符串；如果 Path 为“..”，GetBaseName 方法返回“.”。换句话说，它只不过是一个字符串处理函数，而不是一个文件函数。

GetFolder 方法

GetFolder 方法返回 Folder 对象。其语法为：

```
oFileSysObj.GetFolder (FolderPath)
```

其中：

1.oFileSysObj 代表任何能返回 FileSystemObject 对象的对象变量。

2.参数 FolderPath 必需，指定所需文件夹的路径，String 类型，可以为相对路径或绝对路径。如果 FolderPath 是共享名或网络路径，GetFolder 确认该驱动器或共享是 File 对象创建进程的一部分。如果 FolderPath 的任何部分不能连接或不存在，就会产生一个错误。

说明：

1.要获得所需的 Path 字符串，首先应该使用 GetAbsolutePathName 方法。如果 FolderPath 包含一个网络驱动器或共享，可以在调用 GetFolder 方法之前使用 DriveExists 方法确认指定的驱动器是否可用。由于 GetFolder 方法要求 FolderPath 是一个有效文件夹的路径，所以应调用 FolderExists 方法来检验 FolderPath 是否存在。

2.必须使用 Set 语句将 Folder 对象赋给一个局部对象变量。

示例

下面的程序列出 C 盘中的所有文件：

```
Sub FilesInFolder()

    Dim fs As Object

    Dim objFolder As Object

    Dim objFile As Object

    Set fs = CreateObject("Scripting.FileSystemObject")

    Set objFolder = fs.GetFolder("C:\")

    Workbooks.Add

    For Each objFile In objFolder.Files

        ActiveCell.Select

        Selection.Formula = objFile.Name

        ActiveCell.Offset(0, 1).Range("A1").Select

        Selection.Formula = objFile.Type
```

```
        ActiveCell.Offset(1, -1).Range("A1").Select
    Next
    Columns("A:B").Select
    Selection.Columns.AutoFit
End Sub
```

在我的电脑上运行上述代码后的结果如下图 7 所示。

	A	B
1	1.dat	DAT 文件
2	bootmgr	系统文件
3	DIY.jpg	JPEG 图像
4	DTDHI	系统文件
5	excelperfect.xlsm	Microsoft Office Excel 启用了宏的工作表
6	hiberfil.sys	系统文件
7	IFRToolLog.txt	文本文档
8	InstallConfig.ini	配置设置
9	LibAntiPrtSc_ERROR.log	文本文档
10	LibAntiPrtSc_INFORMATION.log	文本文档
11	logExcel.csv	Microsoft Office Excel 逗号分隔值文件
12	pagefile.sys	系统文件
13	result.daw	DAW 文件
14	testNewFile111.txt	文本文档
15	yx1.jpg	JPEG 图像
16	完美Excel.xlsx	Microsoft Office Excel 工作表
17		
18		

图 7

GetParentFolderName 方法

返回给定路径中最后部分前的文件夹名，其语法为：

```
oFileSysObj.GetParentFolderName (Path)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Path 必需，指定路径说明，String 类型。

说明：

1.如果从 Path 中不能确定父文件夹名，就返回一个零长字符串（""）。Path 可以为相对路径或绝对路径。可以是网络驱动器或共享。

2.GetParentFolderName 方法不能检验 Path 的某个部分是否存在。

3.GetParentFolderName 方法认为 Path 中不属于驱动器说明的那部分字符串除了最后一部分外余下的字符串就是父文件夹。除此之外它不做任何其他检测，更像是一个字符串解析和处理例程而不是与对象处理有关的例程。

GetSpecialFolder 方法

GetSpecialFolder 方法返回操作系统文件夹路径，其中 0 代表 Windows 文件夹，1 代表 System（系统）文件夹，2 代表 Temp（临时）文件夹。其语法为：

```
oFileSysObj.GetSpecialFolder (SpecialFolder)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 SpecialFolder 必需，为特殊的文件夹常数，表示三种特殊系统文件夹中其中一个的值。

说明：

1.可以使用 Set 语句将 Folder 对象赋给一个局部对象变量，但是如果只对检索特殊的文件夹感兴趣，就可以使用下列语句来实现：

```
sPath=oFileSys.GetSpecialFolder (iFolderConst)
```

或者：

```
sPath=oFileSys.GetSpecialFolder (iFolderConst).Path
```

2.由于 Path 属性是 Folder 对象的缺省属性，所以第一个语句有效。因为不是给一个对象变量赋值，所以赋给 sPath 的值是缺省的 Path 属性值，而不是对象引用。

示例

获取并显示特定文件夹路径：

```
Sub SpecialFolders()
```

```
    Dim fs As Object
    Dim strWindowsFolder As String
    Dim strSystemFolder As String
    Dim strTempFolder As String
    Set fs = CreateObject("Scripting.FileSystemObject")
    strWindowsFolder = fs.GetSpecialFolder(0)
    strSystemFolder = fs.GetSpecialFolder(1)
    strTempFolder = fs.GetSpecialFolder(2)
    MsgBox strWindowsFolder & vbCrLf & strSystemFolder & vbCrLf _
        & strTempFolder, vbInformation + vbOKOnly, "特定文件夹"
End Sub
```

在我的电脑上运行上述代码后的结果如下图 8 所示。

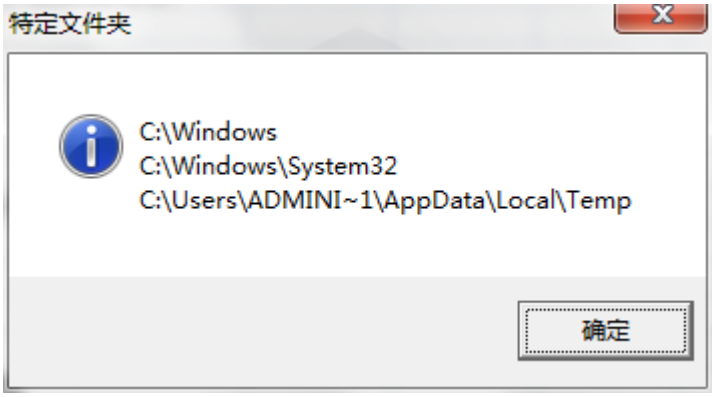

图 8

GetTempName 方法

返回系统创建的一个临时文件或文件夹名。其语法为：

```
oFileSysObj.GetTempName
```

其中：

oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

说明：

1.GetTempName 方法不能创建临时文件或文件夹，它仅仅提供一个可用于 CreateTextFile 方法的文件或文件夹名。

2.一般来说，不必创建自己的临时文件名。Windows 在 Windows API 中提供了一种算法来创建特殊的临时文件或文件夹名，这样 Windows 才能识别它们。GetTempName 很好地包装了 GetTempFilename API 函数。

CreateFolder 方法

CreateFolder 方法用于在指定的路径下创建一个新文件夹，并返回其 Folder 对象。其语法为：

```
oFileSysObj.CreateFolder (Path)
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Path 必需，为一个返回要创建的新文件夹名的表达式，String 类型。Path 指定的路径可以是相对路径也可以是绝对路径，如果没有指定路径则使用当前驱动器和目录作为路径。在新的文件夹名中不能使用通配符。

3.如果参数 Path 指定的路径为只读，则 CreateFolder 方法将失败；如果参数 Path 指定的文件夹已经存在，就会产生运行时错误“文件已经存在”。如果用户有充分的权限，则参数 Path 可以指定为网络路径或共享名，例如：

CreateFolder “\\NTSERV1\RootTest\newFolder”

4.在实际使用时，必须使用 Set 语句将 Folder 对象赋给对象变量，例如：

```
Dim oFileSys As New FileSystemObject

Dim oFolder As Folder

Set oFolder=oFileSys.CreateFolder(“MyFolder”)
```

示例

下面的代码在 C 盘创建一个名为 TestFolder 的文件夹并显示创建信息：

```
Sub MakeNewFolder()

   Dim fs, objFolder

   Set fs = CreateObject("Scripting.FileSystemObject")
```

```
    Set objFolder = fs.CreateFolder("C:\TestFolder")

    MsgBox "创建了一个名称为" & objFolder.Name & "的文件夹."
End Sub
```

运行代码后的结果如下图 9 所示。

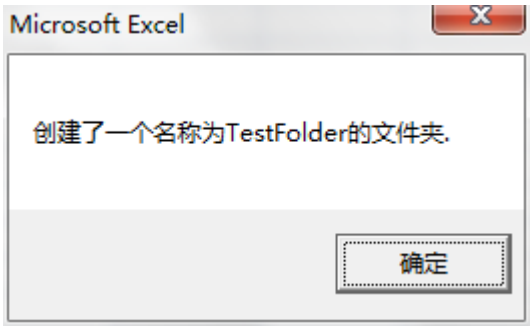

图 9

CopyFolder 方法

CopyFolder 方法用于复制文件夹，即将一个文件夹的内容（包括其子文件夹）复制到其他位置。其语法为：

```
oFileSysObj.CopyFolder Source,Destination[,OverwriteFiles]
```

其中：

1.参数 oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Source 必需，指定要复制的文件夹的路径和文件夹名，String 类型，必须使用通配符或者非路径分隔符来结束。

3.参数 Destination 必需，指定文件夹复制操作的目标文件夹的路径，String 类型。

4.参数 OverwriteFiles 可选，表示是否被覆盖一个现有文件的标志，True 表示覆盖，False 表示不覆盖，Boolean 类型。

说明：

1.通配符只能在参数 Source 中使用，但是只能放在最后的组件中。在参数 Destination 中不能使用通配符。

2.除非不允许使用通配符，否则就可以把源文件夹中的所有子文件夹和文件都复制到 Destination 指定的文件夹中，也就是说 CopyFolder 方法是递归的。

3.如果参数 Destination 以一个路径分隔符结束或者参数 Source 以一个通配符结束，CopyFolder 方法就认为参数 Source 中的指定的文件夹存在于参数 Destination 中，否则就

创建这样一个文件夹。例如，假设有如下图 10 的文件夹结构：

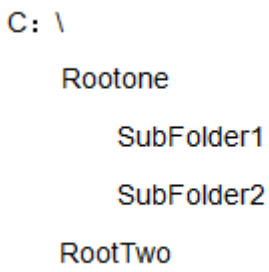

图 10

`CopyFolder "C:\Rootone\*","C:\RootTwo"`产生如下的文件夹结构：

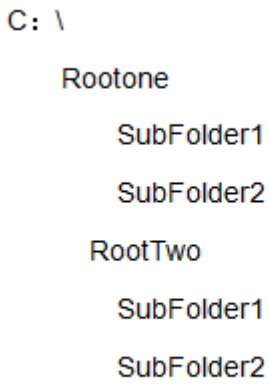

图 11

`CopyFolder "C:\Rootone","C:\RootTwo\"`产生如下的文件夹结构：

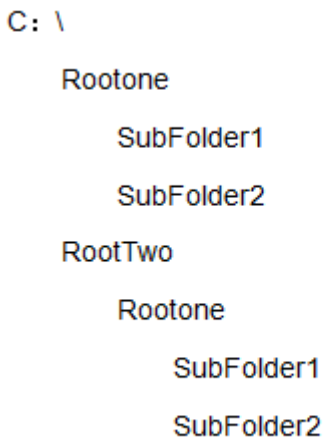

图 12

4.如果参数 Destination 指定的目标路径或任意文件被设置成只读属性，则不论 OverwriteFiles 的值如何，CopyFolder 方法者将失效。

5.如果 OverwriterFiles 设置为 False，而参数 Source 指定的源文件夹或任何文件存在于参数 Destination 中，将产生运行时错误“文件已经存在”。

6.如果在复制多个文件夹时出现错误，CopyFolder 方法立即停止复制操作，不再复制余下要复制的文件。该方法不具有撤销产生错误前文件复制操作的返回功能。

7.如果用户有充分的权限，source 或 destination 都可以是网络路径或共享名，例如：

CopyFolder “C:\Rootone”,\\NTSERV1\d$\RootTwo\

示例

下面的代码将文件夹 TestFolder 复制到文件夹 FinalFolder 中：

```
Sub MakeFolderCopy()
    Dim fs As FileSystemObject
    Set fs = New FileSystemObject
    If fs.FolderExists("C:\TestFolder") Then
        fs.CopyFolder "C:\TestFolder", "C:\FinalFolder"
        MsgBox "已复制该文件夹."
    End If
End Sub
```

MoveFolder 方法

MoveFolder 方法用来移动文件夹，将文件夹及其文件和子文件夹一起从某个位置移动到另一个位置。其语法为：

```
oFileSysObj.MoveFolder source,destination
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 Source 指定要移动的文件夹的路径，String 类型。

3.参数 destination 指定文件夹移动操作中目标位置的路径，String 类型。

说明：

1.Source 必须以通配符或非路径分隔符结束，可以使用通配符，但必须出现在最后一部分中。

2.destination 不能使用通配符。除非不允许使用通配符，否则源文件夹中所有的子文件夹和文件都被复制到 destination 指定的位置，也就是说 MoveFolder 方法是递归的。

3.如果 destination 用路径分隔符结束或者 source 用通配符结束，MoveFolder 就认为 source 中指定的文件夹存在于 destination 中。例如，假设有如下文件夹结构：

C:\
Rootone
SubFolder1
SubFolder2
RootTwo

图 13

`MoveFolder "C:\Rootone\*","C:\RootTow\"`产生如下文件夹结构：

C:\
Rootone
SubFolder1
SubFolder2
RootTwo
SubFolder1
SubFolder2

图 14

`MoveFolder "C:\Rootone","C:\RootTwo\"`产生如下文件夹结构：

C:\
Rootone
SubFolder1
SubFolder2
RootTwo
Rootone
SubFolder1
SubFolder2

图 15

4.Source 和 destination 可以为绝对路径或相对路径，可以为网络路径或共享名。

5.MoveFile 方法在开始操作前先解析 source 和 destination 这两个参数。

DeleteFolder 方法

DeleteFolder 方法用于删除指定的文件夹及其所有的文件和子文件夹。其语法为：

```
oFileSysObj.DeleteFolder FileSpec[,Force]
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 FileSpec 必需，指定要删除的文件夹的名称和路径，String 类型。在参数 FileSpec 中，可以在路径的最后部分包含通配符，但不能用路径分隔符结束，可以为相对路径或绝对路径。

3.参数 Force 可选，Boolean 类型，如果设置为 True，将忽略文件的只读标志并删除这个文件。默认为 False。如果参数 Force 设置为 False 并且文件夹中的任意一个文件为只读，则该方法将失败。如果找不到指定的文件夹，则该方法失败。

说明：

1.如果指定的文件夹中有文件已经打开，则不能完成删除操作，且产生一个“Permisson Denied”错误。DeleteFolder 方法删除指定文件夹中的所有内容，包括其他文件夹及其内容。

2.如果在删除多个文件或文件夹时出现错误，DeleteFolder 方法将立即停止删除操作，即不能删除余下的文件夹或文件。该方法不具有撤销产生错误前文件夹删除操作的返回功能。

3.DeleteFolder 方法永久性删除文件夹，并不把它们移到回收站中。

4.如果用户有充分的权限，源路径和目标路径可以是网络路径或共享名，例如：

DeleteFolder “\\RootTest”

示例

下面的代码删除 C 盘中指定的文件夹：

```
Sub RemoveFolder()

    Dim fs As FileSystemObject

    Set fs = New FileSystemObject

    If fs.FolderExists("C:\TestFolder") Then
```


```
        fs.DeleteFolder "C:\TestFolder"
        MsgBox "该文件夹已经被删除."
    End If
End Sub
```

OpenTextFile 方法

OpenTextFile 方法用于打开（或创建）文本文件以进行读取或写入操作，返回一个 TextStream 对象。其语法为：

```
oFileSysObj.OpenTextFile (FileName[,IOMode[,Create[,Format]]])
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.参数 FileName 必需，指定要打开的文件的路径和文件名，String 类型，FileName 的路径部分可为相对路径或绝对路径。

3.参数 IOMode 可选，指定文件打开模式的一个常数，默认设置为 ForReading（1）。

4.参数 Create 可选，一个 Boolean 型标志，说明如果在给定的路径中找不到文件，是否应该创建该文件。

5.参数 Format 可选，一个 Tristate 常数，指定打开文件的格式为 ASCⅡ或 Unicode 格式，默认设置为 ASCⅡ（False）。

说明：

1.如果另一个进程已经打开了指定文件，该方法将失败，并产生一个“Permission Denied”错误。

2.要保证 OpenTextFile 方法成功执行，可以在调用它之前使用 GetAbsolutePath 和 FileExists 方法。

3.IOMode 的值只能是一个常量值，例如，下面的方法调用：

```
lMode=ForReading or ForWriting
oFileSys.OpenTextStream(strFileName,lMode) '错误
```

将产生运行时错误“无效的过程调用或参数”。

4.如果用户有充分的权限，FileName 的路径部分可以是网络路径或共享名，例如：

OpenTextFile “\\NTSERV1\d$\RootTwo\myFile.txt”

示例

下面的代码读取文件 System.ini 中的内容并将其写入到工作表中：

```
Sub ReadTextFile()

    Dim fs As Object

    Dim objFile As Object

    Dim strContent As String

    Dim strFileName As String

    strFileName = "C:\Windows\System.ini"

    Set fs = CreateObject("Scripting.FileSystemObject")

    Set objFile = fs.OpenTextFile(strFileName)

    Do While Not objFile.AtEndOfStream

        strContent = strContent & objFile.ReadLine & vbCrLf

    Loop

    objFile.Close

    Set objFile = Nothing

    ActiveWorkbook.Sheets(1).Select

    Range("A1").Select

    Selection.Formula = strContent

End Sub
```

Drives 属性

Drives 属性是 FileSystemObject 对象唯一的属性，返回对硬盘驱动器集合（Drives）的引用，是一个只读属性。其语法为：

```
oFileSysObj.Drives
```

其中：

1.oFileSysObj 代表任何能够返回 FileSystemObject 对象的对象变量。

2.Drives 属性返回的集合中的每个成员都是 Drive 对象，表示系统中一个可用的驱动器。可以使用 For … Next 循环迭代系统中所有驱动器，或者使用 Drives 集合的 Item 方法读取某个 Drive 对象（代表系统中的某个驱动器）。

示例

下面的代码创建计算机驱动盘清单：

```
Sub DrivesList()

    Dim fs As Object

    Dim colDrives As Object

    Dim Drive As Object

    Dim strDrive As String

    Set fs = CreateObject("Scripting.FileSystemObject")

    Set colDrives = fs.Drives

    For Each Drive In colDrives

        strDrive = "驱动器" & Drive.DriveLetter & ":"

        Debug.Print strDrive

    Next

End Sub
```

File 对象与 Files 集合详解

本节将详解 FileSystem 对象模型的 File 对象与 Files 集合。

File 对象

File 对象表示某种类型的磁盘文件，允许访问指定文件的所有属性并移向文件系统的上一级访问文件驻留的系统。创建 File 对象的进程，例如不用打开文件就可以将 File 对象的 Item 属性的引用赋给局部对象变量。

File 对象的属性

- Attributes 属性：返回文件的属性，FileAttribute 常量。
- DateCreated 属性：返回文件的创建日期，Date 类型。
- DateLastAccessed 属性：返回文件最后访问的日期，Date 类型。
- DateLastModified 属性：返回文件最后修改的日期，Date 类型。
- Drive 属性：返回表示文件所在驱动器的 Drive 对象，后面带有冒号。
- Name 属性：返回文件名称，String 类型。
- ParentFolder 属性：返回文件的父文件夹，即文件所在的文件夹，Folder 对象。
- Path 属性：返回文件的完整路径，包括驱动器名或网络路径/共享名，String 类型。
- Size 属性：返回以字节表示的文件大小，Variant 类型。
- Type 属性：返回包含注册类型描述的字符串，Variant 类型。即显示在 Windows 资源管理器类型列中的文件类型字符串，例如配置设置、应用程序、快捷方式等。如果文件没有扩展名，则类型为“File”；如果文件类型没有注册，则类型显示为扩展名和“File”。

File 对象的方法

Copy 方法

将指定的文件从某位置复制到另一个位置，其语法为：

```
oFileObj.Copy Destination[,OverwriteFiles]
```

其中：

- oFileObj 代表任何能够返回 File 对象的对象变量。
- 参数 Destination 必需，指定被复制文件的路径和可选的文件名，String 类型。
- 参数 OverwriteFiles 可选，Boolean 类型。如果可以覆盖一个已经存在的文件，则 OverwriteFiles 的值为 True，否则为 False。

说明：

- 在 Destination 中不能使用通配符。
- 如果参数 Destination 将文件路径设置为只读，则无论参数 Overwrite 的设置如何，Copy 方法都将失败。
- 如果 OverwriteFiles 为 False，而在 Destination 中又存在该文件，则会产生运行时错误“文件已经存在”。
- 如果用户有足够的权限，则 Destination 可以是一个网络路径或共享名。例如：`Copy \\NTSERV1\RootTest`

Delete 方法

删除当前文件，其语法为：

```
oFileObj.Delete [Force]
```

其中：

- oFileObj 代表任何能够返回 File 对象的对象变量。
- 参数 Force 可选，Boolean 类型，如果设置为 True，则忽略文件的只读标志（如果已设置只读属性的话）并删除该文件。参数 Force 的默认设置为 False。如果参数 Force 设置为 False 而文件为只读，则 Delete 方法将失败。

说明：

- 如果要删除的文件已打开，则 Delete 方法将失败并产生一个“Permission Denied”错误。

- File 对象的 Delete 方法和 FileSystemObject 对象的 Delete 方法不同，后者允许路径参数中有通配符，因而能同时删除多个文件，而前者只能删除 oFileObj 参数指定的一个文件。
- 执行 Delete 方法后，包含 oFileObj 的 Files 集合对象会自动更新，被删除的文件从集合中移除且集合的计数减一。不能再次访问已删除的文件，而应该将 oFileObj 设置为 Nothing。

Move 方法

将文件从一个文件夹移动到另一个文件夹。其语法为：

```
oFileObj.Move destination
```

其中：

- oFileObj 代表任何能够返回 File 对象的对象变量。
- 参数 destination 必需，String 类型，指定要移动的文件所在位置的路径。

说明：

- 在参数 destination 中不能使用通配符，但可以是绝对路径也可以是相对路径。
- 要保证 Move 方法成功执行，可在调用前使用 FileSystemObject 对象的 FileExits 方法和 GetAbsolutePath 方法。
- File 对象的 Move 方法和 FileSystemObject 对象的 MoveFile 方法不同，后者允许路径参数中有通配符，因而能够同时移动多个文件，而前者只能移动参数 oFileObj 指定的一个文件。
- 执行 Move 方法后，包含 oFileObj 的 Files 集合对象会自动更新，从集合中删除被移动的文件且使集合的计数减一。不能在原来的 Folder 集合对象中再次访问已移动的文件。
- 如果用户有足够的权限，则 Destination 可以是一个网络路径或共享名。例如：
- oFile.Move \\NTSERV1\d$\RootTwo\myfile.doc

OpenAsTextStream 方法

打开被引用的文本文件以进行文件的读或写操作，返回一个 TextStream 对象。其语法为：

```
oFileObj.OpenAsTextStream ([IOMode[,Format]])
```

其中：

- oFileObj 代表任何能够返回 File 对象的对象变量。
- 参数 IOMode 可选，为下表中所列的 IOMode 常量，指定打开文件的模式，默认设置为 ForReading（1）。

- 参数 Format 可选，为下表所列的 Tristate 常量，指定打开文件的格式为 ASCⅡ格式或 Unicode 格式，默认设置为 ASCⅡ（False）。

常量	值	含义
ForAppending	8	以追加模式打开文件，即保护文件的当前内容并在文件的末尾写入新的数据。
ForReading	1	以只读模式打开文件，不能对以只读模式打开的文件进行写操作。
ForWriting	2	以写模式打开文件，文件原来的所有内容都被新的数据覆盖。

表 IOMode 常量

常量	值	含义
TristateUseDefault	-2	以系统缺省格式打开文件
TristateTrue	-1	以 Unicode 格式打开文件
TristateFalse	0	以 ASCⅡ格式打开文件

表 Tristate 常量

说明：

- 如果另外一个进程已经打开了该文件，则 OpenAsTextStream 方法失败并产生错误“权限被否定”。

Files 集合

Files 集合是任何 Folder 对象的 File 属性所返回的 File 对象的容器（即 Folder.Files）。Files 集合包含文件夹中的所有文件。使用 Files 集合的 Item 属性可以获得对某个 File 对象的引用，必须使用准确的文件名（包含文件的扩展名）作为参数。可使用 For Each … Next 语句迭代集合中的所有文件。

Count 属性

返回集合中的 File 对象的数目，Long 类型。

Item 属性

使用文件名（包括扩展名）作为参数，返回具有该文件名的文件对应的 File 对象。各个 File 对象不能通过它们在集合中的序号位置来访问。Item 属性是 Files 集合对象的默认属性。

示例

下面的代码列出 H 盘中的文件 drawcircle.xlsm 的信息。

```
Sub GetFileObject()
    Dim ofsFileSys As New FileSystemObject
    Dim ofsFiles As Files
    Dim ofsFile As File
    Set ofsFiles = ofsFileSys.Drives("H").RootFolder.Files
    Set ofsFile = ofsFiles.Item("drawcircle.xlsm")
    MsgBox "This file's Infomation:" & vbCrLf & _
          "File Name:" & ofsFile.Name & vbCrLf & _
          "Create Date:" & ofsFile.DateCreated & vbCrLf & _
          "Last Modify Date:" & ofsFile.DateLastModified & vbCrLf & _
          "Last Access Date:" & ofsFile.DateLastAccessed & vbCrLf & _
          "File Size:" & ofsFile.Size & vbCrLf & _
          "File Path:" & ofsFile.Path
    Set ofsFile = Nothing
    Set ofsFiles = Nothing
    Set ofsFileSys = Nothing
End Sub
```

运行代码后，显示如下图 16 所示的信息框。

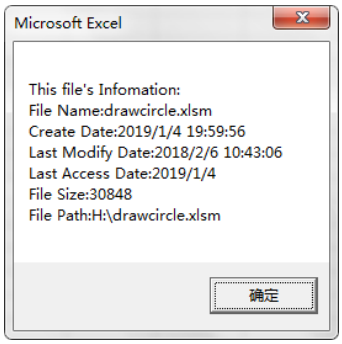

图 16

Folder 对象与 Folders 集合

本节详解 FileSystem 对象模型的 Folder 对象与 Folders 集合。

Folder 对象

使用 Folder 对象可以访问指定文件夹的所有属性，并提供了复制、移动和删除文件夹的方法，还可以在文件夹内新建一个文本文件。

Folder 对象的重要性在于可以通过它访问 Folders 集合对象。最常用的方法是提取集合中的一个成员以访问某个对象。然而，由于 Drive 对象在根文件夹仅给出了 Folder 对象，因此不得不从 Folder 对象（表示根文件夹中子文件夹的集合）中提取 Folders 集合对象。利用 Folders 集合，可以向下搜索整个文件系统以选择其他 Folder 对象和其他 Folders 集合。IsRootFolder 参数将告诉当前正在处理的 Folder 对象是否是 Drive 的根文件夹。

Attributes 属性

返回文件夹的属性。FileAttributes 常量是一组表示文件夹属性的标志。通过对 Attributes 返回值和要测试的常量值进行逻辑与（AND），可以确定设置了哪一个标志。例如：

```
If oFolder.Attributes And ReadOnly Then '文件夹是只读的
```

FileAttributes 常量如下：

常数	值
Alias	64
Archive	32
Compressed	2048
Directory	16
Hidden	2
Normal	0
ReadOnly	1
System	4
Volume	8

DateCreated 属性

返回文件夹的创建日期，Date 类型。

DateLastAccessed 属性

如果可以从操作系统获得时间信息，则属性表示最近一次访问文件夹的日期，Date 类型。

DateLastModified 属性

最近一次修改文件夹的日期，Date 类型。

Drive 属性

返回一个 Drive 对象，表示该文件夹所在的驱动器名称。

Files 属性

返回文件夹中的文件集合，表示当前文件夹中的所有文件。

IsRootFolder 属性

IsRootFolder 属性判断文件夹是否为根文件夹，如果是则返回 True，Boolean 类型。

Name 属性

返回文件夹的名称，String 类型。

ParentFolder 属性

返回指定文件夹的父文件夹。如果当前文件夹是它所在驱动器的根文件夹，则该属性无效。

Path 属性

返回文件夹的完整路径，String 类型。

Size 属性

返回文件夹的大小，以字节表示。包含当前文件夹中所有文件、子文件夹及其内容的全部大小，Variant 类型。

SubFolders 属性

返回文件夹中的子文件夹的集合，表示当前文件夹内的所有子文件夹。

Type 属性

返回文件夹的类型，例如文件夹或者回收站。该属性并未完全实现，总是返回“File Folder”，String 类型。

Copy 方法

将当前文件夹及其内容（包括其他文件夹）复制到另一个位置，其语法为：

```
oFolderObj.Copy Destination[,OverwriteFiles]
```

其中：

1.oFolderObj 代表任何能够返回 Folder 对象的对象变量。

2.参数 Destination 必需，表示文件夹复制中的目标路径（可包含文件名），String 类型，不能在其中使用通配符。

3.参数 OverwriteFiles 可选，表示是应该覆盖（True）还是不覆盖（False）已存在的文件和文件夹，Boolean 类型。

说明：

1.Copy 方法将源文件夹中包含的文件夹、所有子文件夹和文件复制到 Destination 位置，也就是说，Copy 方法是递归的。

2.Folder.Copy 方法与 FileSystemObject.CopyFolder 方法不同，对于使用路径分隔符还是非路径分隔符结束 Destination，在操作上都是相同的。

3.如果目标路径或者 Destination 结构中的任何一个文件设置为只读，则不论 OverwriteFiles 的值如何，Copy 方法都将失败。

4.如果 OverwriteFiles 设置为 False，且 Destination 结构中包含源文件夹或源文件夹中的任意一个文件，将产生错误“文件已经存在”。

5.如果在复制多个文件时出现错误，Copy 方法立即停止操作，不再复制剩下的未复制的文件。该方法不具有撤销产生错误之前复制操作的返回功能。

6.如果用户有充分的权限，Destination 可为网络路径或共享名。例如：oFolder.Copy \\NTSERV1\d$\RootTwo\

CreateTextFile 方法

在指定位置新建一个文件，并返回所创建文件的 TextStream 对象。其语法为：

```
oFolderObj.CreateTextFile FileName[,Overwrite[,Unicode]]
```

其中：

1.oFolderObj 代表任何能够返回 Folder 对象的对象变量。

2.参数 FileName 必需，表示任何有效的文件名（路径可选），String 类型，不允许使用通配符。

3.参数 Overwrite 可选，一个标志，表示是否覆盖具有相同文件名的已有文件，Boolean 类型，默认设置为 False。

4.参数 Unicode 可选，一个标志，表示是用 Unicode 格式还是 ASCⅡ格式写文件，默认设置为 False。如果 Unicode 设置为 True，将创建一个 Unicode 格式的文件，否则创建一个 ASCⅡ文本文件。

说明：

1.新建的文本文件只有在进行写操作时才自动打开。如果随后要读这个文件，必须先关闭这个文件，然后以读模式打开该文件。

2.必须使用 Set 语句把 TextStream 对象赋给一个局部对象变量。

3.Folder 对象中的 CreateTextFile 方法与 FileSystemObject 对象中的 CreateTextFile 方法在操作上是一样的。

Delete 方法

删除当前文件夹及其所有文件和子文件夹。其语法为：

```
oFolderObj.Delete [Force]
```

其中：

1.oFolderObj 代表任何能够返回 Folder 对象的对象变量。

2.参数 Force 可选，如果设置为 True，则忽略文件的只读标志并删除这个文件，Boolean 类型，默认设置为 False。

说明：

1.如果文件夹中某些文件已经打开，该方法失败并产生一个“Permission Denied”错误。

2.Delete 方法删除指定文件夹中的所有内容，包括其他文件夹和它们的内容。

3.如果参数 Force 设置为 False，且文件夹中的任一文件设置为只读，则该方法失败。

4.如果在删除多个文件或文件夹时出错，Delete 方法立即停止操作，不再删除余下没有删除的文件或文件夹，该方法不具有撤销出错前删除操作的返回功能。

5.Folder 对象的 Delete 方法与 FileSystemObject 的 DeleteFolder 方法不同，后者可以在路径参数中使用通配符，因而能同时删除多个文件夹，而前者只是删除 Folder 对象所表示的一个文件夹。

6.执行完 Delete 方法后，包含 Folder 对象的 Folders 集合对象会自动更新。从 Folders 集合中移除被删除的文件夹，且集合的计数减一。不应再访问已删除的文件夹，应该设置局部对象变量为 Nothing，例如下面的代码所示：

```
Set ofsSubFolder=ofsSubFolders.Item(“roottwo”)

MsgBox ofsSubFolders.Count

ofsSubFolder.Delete False

MsgBox ofsSubFolders.Count

Set ofsSubFolder=Nothing
```


Move 方法

将文件夹结构从某个位置移动到另一个位置。其语法为：

```
oFolderObj.Move destination
```

其中：

1.oFolderObj 代表任何能够返回 Folder 对象的对象变量。

2.参数 destination 必需，指定文件夹移动操作中目标位置的路径，String 类型，不能使用通配符，可以为绝对路径或相对路径。

说明：

1.在移动文件夹时，如果其中的某些文件已经打开，则将产生一个错误。

2.如果在移动操作的过程中出错，Move 方法将立即停止操作，且不再移动剩下的文件和文件夹。

3.要保证该方法能够完成，可以在调用它之前使用 FileSystemObject 的 FolderExists 和 GetAbsolutePath 方法。

4.Folder.Delete 方法和 FileSystemObject 的 MoveFolder 方法不同，后者可以在路径参数中使用通配符，所以能够同时移动多个文件夹，而前者只是删除 Folder 对象表示的一个文件夹及其内容。

5.执行完 Move 方法后，包含 Folder 对象的 Folders 集合对象会立即自动更新：从集合中移除刚被删除的文件夹，集合计数减一。不应该再访问同一 Folders 集合对象中已经删除的文件夹。

6.如果用户有足够的权限，目标路径可以是网络路径或共享名，例如：Move “\NTSERV1\d$\RootTwo\”

Folders 集合

Folders 集合对象是 Folder 对象的容器。通常我们希望从对象所在的集合访问某个对象。例如，我们可能希望从 Folders 集合对象访问一个 Folder 对象。然而，情况也可以反过来，我们可以从一个 Folder 对象的实例访问 Folders 集合对象。这是因为从 Drive 对象创建实例的第一个 Folder 对象是一个 Root Folder（根文件夹）对象，从该对象可以实例化一个子文件夹集合。然后又能够实例化其他 Folder 对象和子文件夹对象，这样就能操作整个驱动器的文件系统。

Item 属性

从 Folders 集合对象检索一个特定的 Folder 对象，可以通过提供的准确的文件夹名来访问某个文件夹，而不需要知道它的路径。但是，不能通过顺序号来访问该文件夹对象，例如下面的语句返回一个表示 roottwo 文件夹的 Folder 对象：

```
Set ofsSubFolder=ofsSubFolders.Item("roottwo")
```

Count 属性

Folders 集合包含的 Folder 对象的数目，Long 类型。

Add 方法

新建一个文件夹。其语法为：

```
oFoldersCollObj.Add newfoldername
```

其中：

1.oFolderCollObj 代表任何能够返回 Folders 集合对象的对象变量。

2.参数 newfoldername 必需，指定新文件夹的名称，String 类型。在 newfoldername 中不能使用路径标识符，只能使用新文件夹的名称。

新文件夹的位置由 Folders 集合对象所在的父文件夹决定。例如，如果从一个 Folders 集合对象调用 Add 方法，且这个 Folders 集合对象是根文件夹对象的一个子文件夹集合，则该方法将在根文件夹下创建新的文件夹（即向根 Folder 对象的子文件夹集合添加新的文件夹），例如：

```
Dim oFileSys As New FileSystemObject

Dim oRoot As Folder,oChild As Folder

Dim oRootFolders As Folders

Set oRoot=oFileSys.Drives("C").RootFolder

Set oRootFolders=oRoot.SubFolders

Set oChild=oRootFolders.Add("Downloads")
```


示例

下面的代码要求用户输入文件夹名称，然后统计该文件夹中的文件数量并显示结果。

```
Sub CountFilesInFolder()
    Dim fs, strFolder, objFolder, colFiles
    strFolder = InputBox("请输入文件夹名称(带有完整路径):")
    If Not IsFolderEmpty(strFolder) Then
        Set fs = CreateObject("Scripting.FileSystemObject")
        Set objFolder = fs.GetFolder(strFolder)
        Set colFiles = objFolder.Files
        MsgBox "文件夹中的文件数量为" & strFolder & "=" & colFiles.Count
    End If
End Sub


Function IsFolderEmpty(myFolder)
    Dim fs, objFolder
    Set fs = CreateObject("Scripting.FileSystemObject")
    Set objFolder = fs.GetFolder(myFolder)
    IsFolderEmpty = (objFolder.Size = 0)
End Function
```

Drive 对象与 Drives 集合详解

本节详解 FileSystem 对象模型的 Dirve 对象与 Drives 集合。

Drive 对象

Drive 对象可以用于访问计算机或服务器上指定的驱动器的属性。Drive 对象不支持任何方法。RootFolder 属性返回一个文件夹对象，表示驱动器的根文件夹或者根目录，此时能够获得一个包含根文件夹下子文件夹的 Folder 集合对象，这样就能访问该驱动器的所有部分。Drive 对象的所有属性都是只读的。

1.AvailableSpace 属性：返回磁盘可用的空间，以字节表示，Variant 类型。

2.FreeSpace 属性：与 AvailableSpace 属性相同，返回磁盘上的可用空间，Variant 类型。

3.DriveLetter 属性：返回驱动器盘符，不带冒号，String 类型。

4.DriveType 属性：返回驱动器类型，DriveType 常量，其中 0 代表未知，1 代表可移动，2 代表固定，3 代表网络，4 代表 CD-ROM，5 代表 RAM 磁盘。

下面的代码查找光驱所在的驱动器盘符并显示：

```
Sub CDROM_DriveLetter()
    Const CDROM = 4
    Dim fs, colDrives, Drive
    Set fs = CreateObject("Scripting.FileSystemObject")
    Set colDrives = fs.Drives
    For Each Drive In colDrives
```


```
        If Drive.DriveType = CDROM Then
            MsgBox "CD-ROM 驱动器:" & Drive.DriveLetter
        End If
    Next
End Sub
```

5.FileSystem 属性：返回所安装的文件系统，例如 FAT、NTFS 或 CDFS，String 类型。

6.IsReady 属性：Boolean 类型。对硬盘而言，应该总是返回 True。对于可移动驱动器而言，如果插入了合适的媒介（例如 CD-ROM 盘）并且可以访问，则返回 True。

下面的自定义函数在光驱可访问时返回 TRUE：

```
Function IsCDROMReady(strDriveLetter)
    Dim fs, objDrive
    Set fs = CreateObject("Scripting.FileSystemObject")
    Set ojbdrive = fs.GetDrive(strDriveLetter)
    IsCDROMReady = (objDrive.DriveType = 4) And objDrive.IsReady = True
End Function
```

7.Path 属性：返回根文件夹的路径，驱动器名后紧跟一个冒号（例如 C:），String 类型。这是 Drive 对象的默认属性。

8.RootFolder 属性：返回表示根文件夹的 Folder 对象，可以以此访问文件系统的其余部分。

9.SerialNumber 属性：返回驱动器系列号，Long 类型。

10.ShareName 属性：用于网络共享，指计算机的网络共享名（例如\\NTSERV1\TestWork），String 类型。

11.TotalSize 属性：返回驱动器总容量，以字节表示，Variant 类型。

12.VolumeName 属性：指驱动器卷标名（如果指定了驱动器卷标的话），String 类型。

Drives 集合

Drives 集合包含所有连接到当前计算机的驱动器，甚至还包括那些当前不能读的驱动器（例如没有插入的可移动驱动器）。Drives 集合以零为基数，只读。

Count 属性

返回集合中 Drive 对象的数目，Long 类型。

Item 属性

语法为：

```
oDrives.Item(Key)
```

返回一个键为 Key（驱动器名）的 Drive 对象。

Drive 对象集合不是普通的集合，因此不能使用驱动器的索引值（在集合中的位置序号），否则将产生运行时错误“无效的过程调用或参数”。由于读取一个不存在的驱动器的 Drive 对象会产生运行时错误，因此最好先调用 FileSystemObject 对象的 DriveExists 方法。

```
Sub TestDrives()
      Dim ofsFileSys As FileSystemObject
      Dim ofsDrives As Drives
      Dim ofsDrive As Drive

      Set ofsFileSys = New FileSystemObject
      Set ofsDrives = ofsFileSys.Drives
      Set ofsDrive = ofsDrives.Item("C")
      MsgBox ofsDrive.DriveType
      Set ofsDrive = Nothing
      Set ofsDrives = Nothing
      Set ofsFileSys = Nothing
```


```
End Sub
```

在我的计算机上运行上述代码的结果如下图 17 所示：

图 17

返回的数字 2 代表 C 盘是固定盘。

利用 WSH 的强大功能

在前面的章节中，我们详细讲解了 Microsoft Scripting Runtime 库的 FileSystem 对象模型。我们说过 WSH 是一种脚本语言，可以自动运行命令，能够控制 Windows 操作系统和应用程序，也能够从操作系统中获取信息。下面，我们就来看看 WSH 的一些强大功能。

WSH 使得操作任何安装在计算机上的自动化对象成为可能，除了可以通过 FileSystemObject 对象访问文件系统外，也允许进行其他的一些操作，例如处理 WSH 和 ActiveX 对象、设定和去除打印机和远程驱动器、操控注册表、创建 Windows 和 Internet 的快捷方式、以及访问 Windows NT Active Directory 服务。

WSH 对象模型由下列三种主要对象组成：WScript、WshShell 和 WshNetwork。

运行其他应用程序

这里提供了从 Excel 中启动外部应用程序的一种方法。例如，下面的程序启动记事本：

```
Sub RunNotepad()
    Dim WshShell As Object
    Set WshShell = CreateObject("WScript.Shell")
    WshShell.Run "Notepad"
    Set WshShell = Nothing
End Sub
```

将相关语句修改为下列语句，分别可以用来启动计算器或浏览器：

```
WshShell.Run “Calc”
WshShell.Run “Explorer”
```


可以打开带有指定文档的应用程序而不是打开一个空的应用程序窗口，例如：

```
Sub OpenTxtFileInNotepad()
    Dim WshShell As Object
    Set WshShell = CreateObject("WScript.Shell")
    WshShell.Run "Notepad C:\Phones.txt"
    Set WshShell = Nothing
End Sub
```

下面的过程打开 MS-DOS 窗口并打印当前目录下的文件列表：

```
Sub RunDOSCommand()
    Dim WshShell As Object
    Set WshShell = CreateObject("WScript.Shell")
    WshShell.Run ("Command /c Dir >1pt1:")
End Sub
```

创建快捷方式

可以使用 Shell 对象创建应用程序或者网页的快捷方式。WshShell 对象有一个 CreateShortcut 方法，返回快捷方式对象：

```
Set myShortcut=WshShell.CreateShortcut(Pathname)
```

其中，参数 Pathname 是指定快捷文件完整路径的字符串。所有快捷方式文件都带有扩展名.Ink，并且该扩展名必须包括在文件路径名中。

ShortCut 对象的属性和方法介绍如下：

TargetPath 属性

代表可执行文件的路径。例如：WshShell.TargetPath=ActiveWorkbook.FullName

WindowStyle 属性

指定快捷方式使用的窗口类型。其中 1 代表普通窗口，3 代表最大化窗口，7 代表最小化窗口。例如，WshShell.WindowStyle=1。

HotKey 属性

指定键盘快捷方式（例如 Alt+f、Shift+g、Ctrl+Shift+z，等等），例如，WshShell.HotKey=”Ctrl+Alt+W”。

IconLocation 属性

指定快捷方式图标的位置。因为图标文件中通常不止一个图标，所以应该提供图标文件的路径，并且后面标明图标在文件里的索引号。如果不指定，则 Windows 使用缺省的图标，例如 WshShell.IconLocation=”notepad.exe,0”

Description 属性

包含一个描述快捷方式的字符串。例如，WshShell.Description=”Wordware Web Site”。

WorkingDirectory 属性

指定快捷方式的工作目录。例如：

```
strWorkDir=WshShell.SpecialFolders(“Desktop”)
WshShell.WorkingDirectory=strWorkDir
```

Save 方法

这是 ShortCut 对象的唯一方法。在使用 CreateShortcut 方法创建快捷方式对象并且设置该快捷方式的属性后，必须使用 Save 方法将快捷方式保存到磁盘上。

示例

下面的过程创建 WshShell 对象并使用 CreateShortcut 方法创建两个快捷方式：一个为当前工作簿的快捷方式，另一个为到微信公众号页面的快捷方式。这两个快捷方式均放置在用户桌面上。

```
Sub CreateShortcut()
    '在桌面创建两个快捷方式
    Dim WshShell As Object
    Dim objShortcut As Object
    Set WshShell = CreateObject("WScript.Shell")
    '创建一个 Internet 快捷方式
    Set objShortcut = _

WshShell.CreateShortcut(WshShell.SpecialFolders("Desktop")
_
& "\weixin.url")
    objShortcut.TargetPath = "https://mp.weixin.qq.com/"
    objShortcut.Save
    '创建一个文件快捷方式
    Set objShortcut = _

WshShell.CreateShortcut(WshShell.SpecialFolders("Desktop")
& _
          "\" & ActiveWorkbook.Name & ".lnk")


    With objShortcut
        .TargetPath = ActiveWorkbook.FullName
        .WindowStyle = 7
        .Save
    End With
    Set objShortcut = Nothing
    Set WshShell = Nothing
End Sub
```

CreateShortcut 过程使用 WshShell 对象的 SpecialFolders 属性返回 Windows 桌面的路径。

运行上述过程后的结果如下图 18 所示，在桌面放置了两个快捷图标。

图 18

说明：可以使用 SpecialFolders 属性查找计算机中特殊文件夹的位置。例如下列特殊文件夹：AllUsersDesktop（所有用户桌面）、AllUsersStartMenu（所有用户开始菜单）、AllUsersPrograms（所有用户程序）、AllUsersStartup（所有用户启动）、Desktop（桌面）、Favorites（收藏夹）、Fonts（字体）、MyDocuments（我的文档）、NetHood（网络连接）、PrintHood（打印机连接）、Programs（程序）、Recent（最近）、SendTo（发送到）、StartMenu（开始菜单）、Startup（启动）和 Templates（模板）。如果请求的特殊文件夹不可用，则 SpecialFolders 属性返回一个空字符串。

2 个示例代码

本节分享了文件操作相关的 2 个示例代码，以结束全书的内容。

示例 1：选择文件并打开

如果不知道要打开的文件的具体位置，可以通过用户选择后再打开该文件。

```
Sub GetOpenFileNameExample()
    Dim lCount As Long
    Dim vFilename As Variant
    Dim sPath As String
    Dim lFilecount As Long

    sPath = "c:\windows\temp\"

    ChDrive sPath
    ChDir sPath

    vFilename   =   Application.GetOpenFilename("Microsoft
Excel files (*.xls*),*.xls*", , "请选择要打开的文件", , True)

    If TypeName(vFilename) = "Boolean" Then Exit Sub
    For lCount = 1 To UBound(vFilename)
        Workbooks.Open vFilename(lCount)
    Next
End Sub
```

如果要选择打开图片文件，则可将文件类型修改为：

```
"图片文件 (*.jpg),*.jpg"
```

示例 2：基于文件名将文件移到相应的文件夹中

如果有许多文件，现在希望根据文件名将文件移至相应的文件夹。例如，将名称中包含“Bankhill”的文件移到“Bankhill”文件夹中，名称中包含“Lite”的文件移到“Lite”文件夹中，等等。

```
Sub LoopFolder()
    Const SourceFolder As String = "C:\fanjy\excelperfect\vba"
    Dim oFSO
    Dim oFolder As Object
    Dim oFile As Object
    Dim NewFolder As String

    Set oFSO = CreateObject("Scripting.FileSystemObject")

    Set oFolder = oFSO.GetFolder(SourceFolder)

    For Each oFile In oFolder.Files
        If oFile.Type Like "*逗号分隔*" Then
            Select Case True
            Case oFile Like "*Bankhill*"
                NewFolder = "Bankhill\"
            Case oFile Like "*Lite*"
                NewFolder = "Lite\"
                '添加类似的代码
            End Select
            Name oFile.Path As SourceFolder & NewFolder & oFile.Name
        End If
```


```
    Next oFile

    Set oFolder = Nothing
    Set oFSO = Nothing
End Sub
```


关于完美 Excel

完美 Excel 是一个坚持分享 Excel 与 VBA 技术知识的微信公众号，自 2017 年 5 月 15 日开始，每天推送一篇 Excel 与 VBA 技术和应用方面的文章。目前，共有 670 余篇实用文章可供广大 Excel 爱好者和使用者学习交流。这本电子书就是根据完美 Excel 上发表的关于文件操作知识的系列文章整理而成的。

每天清晨，**完美 Excel** 微信公众号：***excelperfect*** 都会推送一篇关于 Excel 与 VBA 的相关文章。如果你有兴趣学习 Excel 和 VBA 的相关知识和实战技巧，可以关注完美 Excel 微信公众号，绝对不会让你失望！

可以通过下列方法关注[完美 Excel]微信公众号：

方法 1—在通讯录中搜索“完美 Excel”或者“excelperfect”后点击关注。

方法 2—扫一扫下面的二维码

完美 Excel 微信公众号使用指南

下图 1 是完美 Excel 微信公众号的界面。公众号名称显示在屏幕正上方，屏幕底部显示有“菜单栏”，目前设置的菜单为“技术精粹”、“VBA 精选”、“联系 me”。在底部左侧的小键盘图标为消息框入口，单击可进入消息框界面给完美 Excel 公众号发送消息。

图 1

下图 2、图 3、图 4 分别为底部 3 个菜单的子菜单。目前，菜单“技术精粹”中设置有“2018 年文章合集”、“480 篇文章合集”、“又一波 20 个函数”、“快速学会 30 个函数”、“玩转数据验证”等 5 个子菜单；菜单“VBA 精选”中设置有“Excel VBA 编程基础”、“最最基础入门篇”、“VBA 学习经验”等 3 个子菜单；菜单“联系 me”中设置有“投稿须知”、“网站集粹”、“个人声明”、“坚持的美好”、“2019 年目标”等 5 个子菜单。

图 2

图 3

图 4

单击这些子菜单会进入详细的文章页面或者文章整理的入口页面，方便读者浏览或查阅本公众号的文章。同时，这些子菜单会随着完美 Excel 微信公众号内容的增加而适时调整。

可以单击底部左侧的小键盘图标，进入发送消息界面，如图 5 所示。在文本框中输入想要发送的文字，单击底部的“发送”按钮，就可以将消息发送给完美 Excel 微信公众号。

大家应留意完美 Excel 微信公众号推送的文章中的一些信息，例如，我会在百度网盘中存放一些文档资料或者示例工作簿文件，并在文章中给出进入百度网盘下载的文本信息，你只需在发送消息框中输入我给出的文本，单击发送后，就会收到一条关于下载链接和密码的信息。单击链接并按提示输入密码后，即可获得

相关的文档资料或示例工作簿文件了。

图 5

例如，在图 5 所示的界面中输入“Excel 动画图 2”后，会自动收到图 6 所示的信息，根据信息即可获取这个 Excel 动画图表文件。

图 6

希望大家在完美 Excel 微信公众号中能够学习到所需要的知识，获取到所需要的 Excel 应用技巧，提高自己的水平。但愿在今后的日子里，完美 Excel 微信公众号能够真正帮助大家发挥 Excel 的威力，为大家解决问题，提高工作效率。