



EBook Gratis

APRENDIZAJE

vbscript

Free unaffiliated eBook created from
Stack Overflow contributors.

#vbscript

Tabla de contenido

Acerca de.....	1
Capítulo 1: Empezando con vbscript.....	2
Observaciones.....	2
Versiones.....	2
Examples.....	2
Mensaje de Hello World usando cscript y wscript.....	2
Capítulo 2: Arreglos y Bucles.....	4
Examples.....	4
1. Arreglos - Estático.....	4
2. Arrays - Dynamic.....	4
5. Creando una matriz a partir de un archivo de texto.....	4
7. Para cada bucle.....	4
6. Para Loop.....	5
8. hacer mientras bucle.....	5
9. Haz Hasta Loop.....	5
3. Arreglos - multidimensionales.....	5
4. Arrays - Multidimensional - Dinámico.....	6
Capítulo 3: Consultas WMI.....	7
Introducción.....	7
Examples.....	7
Extraer el nombre del PC local.....	7
Obtención de número de instancias de cualquier proceso.....	7
Obtención de la resolución de pantalla del monitor activo.....	7
Capítulo 4: Creando tu primer guión.....	9
Introducción.....	9
Parámetros.....	9
Examples.....	9
Hola Mundo.....	9
Explicación.....	10
Capítulo 5: Incluir archivos.....	11

Introducción.....	11
Observaciones.....	11
Examples.....	11
Creando un método de "incluir archivo".....	11
Incluyendo archivos.....	12
Inicialización global.....	12
Capítulo 6: InputBox.....	13
Sintaxis.....	13
Parámetros.....	13
Observaciones.....	13
Examples.....	13
Utilice InputBox para asignar la entrada del usuario a una cadena.....	13
Capítulo 7: Instrumentos de cuerda.....	15
Observaciones.....	15
Examples.....	15
1. Cuerda estándar.....	15
2. Fundamentos de manipulación de cuerdas.....	15
3. Buscando una cadena.....	16
5. Rellenar la matriz con texto específico de la cadena a través de los caracteres de inic.....	17
4. Encadenar métodos de manipulación de cuerdas juntos.....	17
Capítulo 8: Objetos del diccionario.....	19
Examples.....	19
Crear diccionario y agregar artículos al diccionario.....	19
Comprobar si existe la clave en el diccionario.....	19
Eliminar artículo del diccionario.....	19
Iterar todos los elementos en el diccionario.....	19
Iterar todas las claves en el diccionario.....	19
Eliminar clave / claves del diccionario.....	20
Capítulo 9: Objetos del sistema de archivos.....	21
Examples.....	21
Comprobando la existencia de un archivo / carpeta / unidad.....	21
Eliminar una carpeta existente y crear una nueva carpeta.....	21

Copiando un archivo / carpeta	22
Mover un archivo / carpeta	22
Objeto de referencia a una carpeta	23
Objeto de referencia a un archivo	24
Capítulo 10: Usando clases	25
Examples	25
Creando una clase	25
Usando una instancia de clase	25
Función de fábrica global para emular un constructor parametrizado	26
Método inicial para emular un constructor parametrizado	26
Cargando archivos de clase externos en script	26
Creditos	27

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [vbscript](#)

It is an unofficial and free vbscript ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official vbscript.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con vbscript

Observaciones

VBScript (VBS) es un lenguaje de secuencias de comandos de Visual Basic para Internet Explorer y Windows. En principio, se puede usar en la web, como JavaScript, pero no tiene mucha compatibilidad, por lo que generalmente se limita a scripts independientes o de servidor en entornos empresariales que usan Windows exclusivamente.

Versiones

Versión	Fecha de lanzamiento
1.0	1996-08-13
2.0	1996-12-14
3.0	1997-10-01
4.0	1998-06-01
5.0	1999-03-01
5.1	1999-12-01
5.5	2000-07-01
5.6	2001-08-27
5.7	2006-10-18
5.8	2009-03-19

Examples

Mensaje de Hello World usando cscript y wscript.

```
WScript.Echo "Hello world!"
```

Esto muestra un mensaje en la consola si se ejecuta con `cscript.exe` (el host de la consola) o en un cuadro de mensaje si se ejecuta con `wscript.exe` (el host de la GUI).

Si está utilizando VBScript como el lenguaje de script del lado del servidor para una página web (para ASP clásico, por ejemplo),

```
Response.Write "Hello world!"
```

coloca el mensaje en el HTML enviado al cliente (navegador).

Si desea mostrar un mensaje en el cuadro de mensaje, puede utilizar:

```
Msgbox "Hello World!"
```

Lea Empezando con vbscript en línea: <https://riptutorial.com/es/vbscript/topic/463/empezando-con-vbscript>

Capítulo 2: Arreglos y Bucles

Examples

1. Arreglos - Estático

```
Dim cars(2)
cars(0) = "Ford"
cars(1) = "Audi"
cars(2) = "Prius"
```

2. Arrays - Dynamic

```
Dim cars()
Redim cars(0) 'Give it 1 entry
Dim tmp
tmp = "Ford"

'ubound(arrayvariable) is the count of array size.
'in this case, it would be 1, since there is 1 entry.
cars(ubound(cars)) = tmp 'cars(0)
Redim preserve cars(ubound(cars)+1)
```

5. Creando una matriz a partir de un archivo de texto.

```
Dim cars
Dim fullname : fullname = "C:\testenv\test.txt"
'If you can, create an instantaneous read for text file data for better memory handling.
'Unless it's a large file and that's impossible.
cars = Split(CreateObject("Scripting.FileSystemObject").OpenTextFile(fullname, 1).ReadAll, vbCRLF)
```

7. Para cada bucle.

No puede alterar el contenido de la matriz a través de la variable de bucle porque es un elemento temporal al que se asigna cada elemento.

```
Dim cars(2) 'collection of different cars
Dim trace 'track iteration details
cars(0) = "Ford"
cars(1) = "Audi"
cars(2) = "Prius"
For Each car in cars
    trace = trace & car & " temporarily changed to "
    car = "Jeep" 'affects car but not the cars array
    trace = trace & car & vbNewLine
Next

MsgBox trace 'show what happened during the loop
```



```

Dim jeeps : jeeps = 0
For Each car in cars
    If car = "Jeep" Then jeeps = jeeps +1
Next

MsgBox jeeps & " of the cars are Jeeps."

```

6. Para Loop

```

Dim i, cars(2)
cars(0) = "Ford"
cars(1) = "Audi"
cars(2) = "Prius"
For i=0 to ubound(cars)
    If cars(i) = "Audi" Then Exit For
Next

```

8. hacer mientras bucle

```

Dim x, cars
x = 0
cars = Split(CreateObject("Scripting.FileSystemObject").OpenTextFile("C:\testenv\example.txt",
1).ReadAll, vbcrLf)
Do While x < ubound(cars)
    If cars(x) = "Audi" Then Exit Loop
    x = x + 1
Loop

```

9. Haz Hasta Loop

```

Dim copycars(), cars(2), x
Redim copycars(0)
x = 0
cars(0) = "Ford"
cars(1) = "Audi"
cars(2) = "Prius"
Do Until x = ubound(cars)
    copycars(ubound(copycars)) = cars(x)
    redim preserve copycars(ubound(copycars)+1)
    x = x + 1
Loop
redim preserve copycars(ubound(copycars)-1) 'trim off the empty last entry

```

3. Arreglos - multidimensionales

```

Dim mdArray(2,3)
mdArray(0, 0) = "test1"
mdArray(0, 1) = "test2"
mdArray(0, 2) = "test3"
mdArray(0, 3) = "test4"
mdArray(1, 0) = "test5"
mdArray(1, 1) = "test6"
mdArray(1, 2) = "test7"
mdArray(1, 3) = "test8"

```

```
mdArray(2, 0) = "test9"  
mdArray(2, 1) = "test10"  
mdArray(2, 2) = "test11"  
mdArray(2, 3) = "test12"
```

4. Arrays - Multidimensional - Dinámico

```
Dim mddArray()  
ReDim mddArray(0)  
Dim ti, testinc: testinc = "test": ti = 1  
For i = 0 To 4  
    Dim tmpArray(): ReDim tmpArray(0)  
    For j = 0 To 3  
        tmpArray(UBound(tmpArray)) = testinc & ti  
        ti = ti + 1  
        ReDim Preserve tmpArray(UBound(tmpArray) + 1)  
    Next  
    ReDim Preserve tmpArray(UBound(tmpArray) - 1)  
    mddArray(i) = tmpArray  
    ReDim Preserve mddArray(UBound(mddArray) + 1)  
Next  
ReDim Preserve mddArray(UBound(mddArray) - 1)
```

Lea Arreglos y Bucles en línea: <https://riptutorial.com/es/vbscript/topic/7520/arreglos-y-bucles>

Capítulo 3: Consultas WMI

Introducción

VBScript puede consultar el Instrumental de administración de Windows (WMI) para obtener información vital relacionada con la PC local y remota. Podemos utilizar las consultas de WMI para realizar diversas tareas, como extraer el nombre de la PC, obtener la resolución de la pantalla, obtener información sobre el usuario y el nombre de usuario, extraer información vital sobre cualquier proceso, modificar la configuración del sistema central, etc.

A continuación se muestran algunos ejemplos que utilizan consultas de WMI para llevar a cabo tareas específicas.

Examples

Extraer el nombre del PC local

```
strComputer = "."
Set objWMIService = GetObject("winmgmts:" _& "{impersonationLevel=impersonate}!\" &
strComputer & "\root\cimv2")

Set colSettings = objWMIService.ExecQuery _("Select * from Win32_ComputerSystem")

For Each objComputer in colSettings
wscript.echo objComputer.Name
Next
```

Este código hará eco del nombre de la PC en el que se ejecuta.

Obtención de número de instancias de cualquier proceso.

```
strComputer = "."
instances = 0
processName = "chrome.exe"

Set objWMIService = GetObject("winmgmts:\\\" & strComputer & "\root\cimv2")
Set colProcess = objWMIService.ExecQuery("Select * from Win32_Process")

For Each objProcess in colProcess
If objProcess.Name = processName Then instances = instances + 1
Next

wscript.echo "Process - "&processName&" has "&instances&" instances running."
```

Obtención de la resolución de pantalla del monitor activo

```
strComputer = "."
Set objWMIService = GetObject("winmgmts:\\\" & strComputer & "\root\cimv2")
Set colItems = objWMIService.ExecQuery("Select * from Win32_DesktopMonitor",,48)
```

```
For Each objItem in colItems
WScript.Echo "ScreenHeight: " & objItem.ScreenHeight
WScript.Echo "ScreenWidth: " & objItem.ScreenWidth
Next
```

Lea Consultas WMI en línea: <https://riptutorial.com/es/vbscript/topic/9572/consultas-wmi>

Capítulo 4: Creando tu primer gui3n

Introducci3n

Para comenzar, en Windows, cree un documento de texto en su escritorio (haga clic con el bot3n derecho > Nuevo > Documento de texto). Cambie la extensi3n de ".txt" a ".vbs". En este punto, es ejecutable haciendo doble clic en 3l (no pasar3a nada si lo intentas, todav3a no hay nada). Para editar, haz clic derecho en el documento y haz clic en editar. Agregue el c3digo de ejemplo para su primer programa.

Par3metros

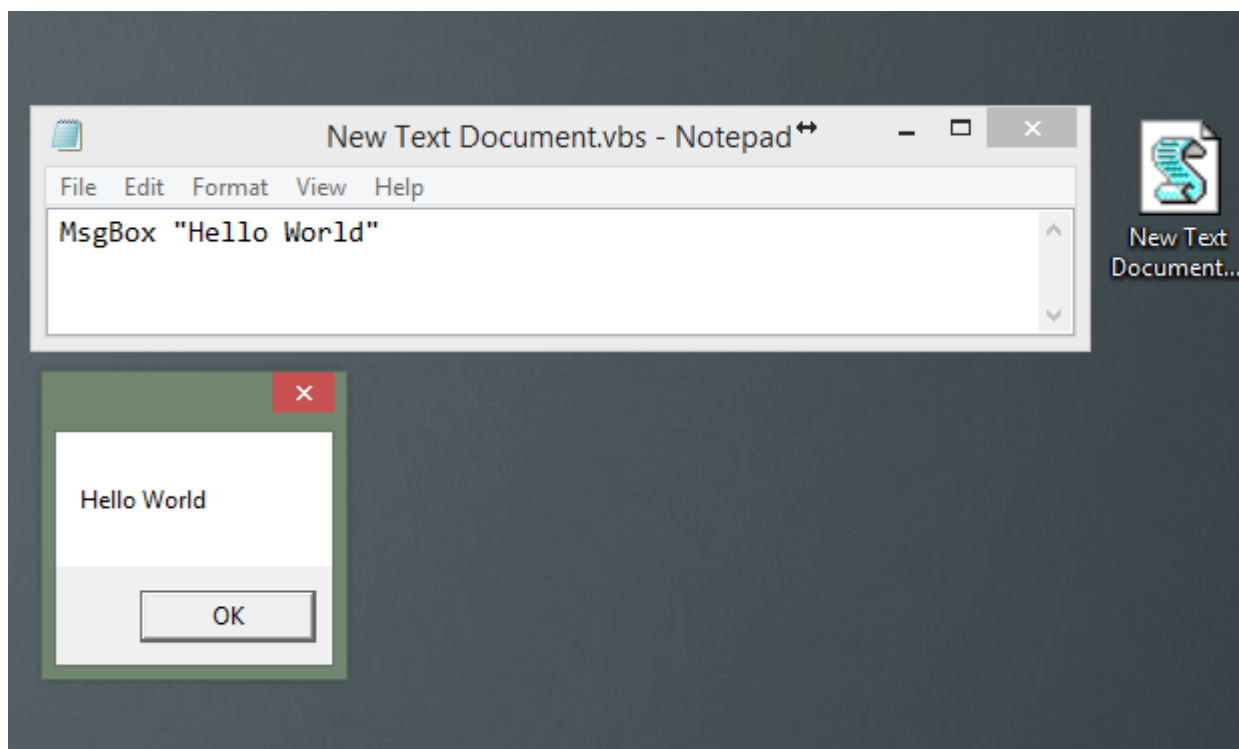
Columna	Columna
C3lula	C3lula

Examples

Hola Mundo

Solo un simple hola mundo para empezar. Copie y pegue lo siguiente en el documento, guarde y luego doble clic.

```
MsgBox "Hello World"
```



Explicación

"MsgBox" muestra un mensaje en un cuadro de diálogo y espera a que el usuario responda. Este es un buen método para informar a los usuarios de las acciones a realizar o simplemente al final del script.

Como:

Lea **Creando tu primer guión en línea**: <https://riptutorial.com/es/vbscript/topic/8036/creando-tu-primer-guion>

Capítulo 5: Incluir archivos

Introducción

Al ejecutar VbScript en el shell de Windows, no hay una función incorporada para incluir un archivo, por lo tanto, para organizar su código en diferentes archivos, deberá crear un método para hacerlo.

Observaciones

Algunas cosas a tener en cuenta al usar el `IncludeFile(p_Path)` :

- No hay limitación de tipo de archivo que se pueda incluir, pero el contenido de los archivos incluidos debe ser VbScript.
- Si hay un error de sintaxis en el archivo incluido, no obtendrá la línea / columna del error.
- Debe definir e inicializar `std_internal_LibFiles` antes de la primera llamada a `IncludeFile(p_Path)`
- Puede usar `IncludeFile(p_Path)` en cualquier parte de su código, incluidos otros métodos.

Examples

Creando un método de "incluir archivo"

Entonces el objetivo principal de esta función es:

- Sea independiente porque necesita estar escrito en el archivo principal de VbScript y no puede estar en un archivo incluido (porque define la función de inclusión)
- Proporcione suficiente información si algo sale mal (es decir, el archivo que se estaba incluyendo, el error que ocurrió, ...)
- Incluya un archivo una vez y solo una vez para evitar incluir bucles.

```
'
*****
'! Includes a VbScript file
'! @param p_Path    The path of the file to include
'
*****

Sub IncludeFile(p_Path)
    ' only loads the file once
    If std_internal_LibFiles.Exists(p_Path) Then
        Exit Sub
    End If

    ' registers the file as loaded to avoid to load it multiple times
    std_internal_LibFiles.Add p_Path, p_Path

    Dim objFso, objFile, strFileContent, strErrorMessage
```

```

Set objFso = CreateObject("Scripting.FileSystemObject")

' opens the file for reading
On Error Resume Next
Set objFile = objFso.OpenTextFile(p_Path)
If Err.Number <> 0 Then
    ' saves the error before resetting it
    strErrorMessage = Err.Description & " (" & Err.Source & " " & Err.Number & ")"
    On Error Goto 0
    Err.Raise -1, "ERR_OpenFile", "Cannot read '" & p_Path & "' : " & strErrorMessage
End If

' reads all the content of the file
strFileContent = objFile.ReadAll
If Err.Number <> 0 Then
    ' saves the error before resetting it
    strErrorMessage = Err.Description & " (" & Err.Source & " " & Err.Number & ")"
    On Error Goto 0
    Err.Raise -1, "ERR_ReadFile", "Cannot read '" & p_Path & "' : " & strErrorMessage
End If

' this allows to run vbscript contained in a string
ExecuteGlobal strFileContent
If Err.Number <> 0 Then
    ' saves the error before resetting it
    strErrorMessage = Err.Description & " (" & Err.Source & " " & Err.Number & ")"
    On Error Goto 0
    Err.Raise -1, "ERR_Include", "An error occurred while including '" & p_Path & "' : " &
vbCrLf & strErrorMessage
End If
End Sub

```

Incluyendo archivos

Para incluir un archivo en otro archivo, solo use el forro:

```
IncludeFile "myOtherFile.vbs"
```

Inicialización global

Antes de usar el método IncludeFile, necesitamos:

- Declarar `std_internal_LibFiles` globalmente
- Inicialízalo con un nuevo diccionario.

```
Dim std_internal_LibFiles
Set std_internal_LibFiles = CreateObject("Scripting.Dictionary")
```

Lea Incluir archivos en línea: <https://riptutorial.com/es/vbscript/topic/8345/incluir-archivos>

Capítulo 6: InputBox

Sintaxis

- Cuadro de entrada (indicador [, título] [, predeterminado] [, xpos] [, ypos] [, archivo de ayuda, contexto])

Parámetros

Argumento	Detalle
<code>prompt</code>	Texto que se muestra sobre el campo de entrada (generalmente una instrucción sobre lo que se requiere del usuario).
<code>title</code>	Subtítulo mostrado en la barra de título del cuadro de entrada.
<code>default</code>	Un marcador de posición para el campo de texto, que se utiliza como valor de retorno si el usuario no sobrescribe.
<code>xpos</code>	Distancia horizontal en twips para mostrar el cuadro de entrada desde el borde izquierdo de la pantalla.
<code>ypos</code>	Distancia vertical en twips para mostrar el cuadro de entrada desde el borde superior de la pantalla.
<code>helpfile</code>	Una cadena para determinar el archivo de ayuda que se utilizará para proporcionar ayuda contextual con el cuadro de entrada.
<code>context</code>	Si se proporciona el argumento del archivo de ayuda, también se debe proporcionar un número de contexto para identificar el tema de ayuda apropiado en el archivo de ayuda.

Observaciones

¹ Todos los argumentos, excepto el `prompt` son opcionales.

² `helpfile` de `helpfile` y `context` están acoplados: si se proporciona uno, el otro también debe proporcionarse.

Examples

Utilice `InputBox` para asignar la entrada del usuario a una cadena

```
' Omitting the 4th and 5th argument ("xpos" and "ypos") will result in the prompt  
' being display center of the parent screen
```

```
exampleString = InputBox("What is your name?", "Name Check", "Jon Skeet", 2500, 2000)
WScript.Echo "Your name is " & exampleString
```

Lea InputBox en línea: <https://riptutorial.com/es/vbscript/topic/634/inputbox>

Capítulo 7: Instrumentos de cuerda

Observaciones

Fecha / hora de MSDN, cadena y funciones numéricas

[https://msdn.microsoft.com/en-us/library/3ca8tfek\(v=vs.84\).aspx](https://msdn.microsoft.com/en-us/library/3ca8tfek(v=vs.84).aspx)

Examples

1. Cuerda estándar

En vbscript, un objeto no necesita necesariamente un tipo designado. Similar a la variable var de C #.

```
Dim ExampleString1 As String
Dim ExampleString2
```

2. Fundamentos de manipulación de cuerdas

```
'Base string
Dim exStr : exStr = " <Head>data</Head> "

'Left
Dim res: res = Left(exStr,6) 'res now equals " <Head"
'Right
Dim res: res = Right(exStr,6) 'res now equals "Head> "
'Mid
Dim res: res = Mid(exStr,8,4) 'res now equals "data"
'Replace
Dim res: res = Replace("variable", "var", "") 'res now equals "riable"
'LCase
Dim res: res = LCase(exStr) 'res now equals " <head>data</head> "
'UCase
Dim res: res = UCase(exStr) 'res now equals " <HEAD>DATA</HEAD> "
'LTrim
Dim res: res = LTrim(exStr) 'res now equals "<Head>data</Head> " notice no space on left side
'RTrim
Dim res: res = RTrim(exStr) 'res now equals "<Head>data</Head> " notice no space on right side
'Trim
Dim res: res = Trim(exStr) 'res now equals "<Head>data</Head>"
'StrReverse
Dim res: res = StrReverse(exStr) 'res now equals " >daeH/<atad>daeH< "
'String
Dim res: res = String(4,"c") 'res now equals "cccc"

'StrComp - String Compare, by default, compares the binary of 2 strings.
'The third parameter allows text comparison, but does not compare case(capitalization).
'Binary
'-1 = if Binary structure of "cars" < "CARS"
' 0 = if Binary structure of "cars" = "cars"
```

```

' 1 = if Binary structure of "CARS" > "cars"
Dim res: res = StrComp("cars", "CARS") 'res now equals -1
Dim res: res = StrComp("cars", "cars") 'res now equals 0
Dim res: res = StrComp("CARS", "cars") 'res now equals 1

'Text
'-1 = if Text structure of "cars" < "CARSSS"
' 0 = if Text structure of "cars" = "cars"
' 1 = if Text structure of "CARSSS" > "cars"
Dim res: res = StrComp("cars", "CARSSS", 1) 'res now equals -1
Dim res: res = StrComp("cars", "cars", 1) 'res now equals 0
Dim res: res = StrComp("CARSSS", "cars", 1) 'res now equals 1

'Space
Dim res: res = "I" & Space(1) & "Enjoy" & Space(1) & "Waffles" 'res now equals "I Enjoy
Waffles"

'Instr - Returns position of character or string in the variable.
Dim res: res = Instr(exStr, ">") ' res now equals 6
'InstrRev - Returns position of character or string in the variable from right to left.
Dim res: res = Instr(exStr, ">") ' res now equals 2

'Split and Join
'These are methods that can be used with strings to convert a string to an array
'or combine an array into a string
Dim res1 : res1 = Split(exStr, ">")
'res1(0) = " <Head"
'res1(1) = "data</Head"
'res1(2) = " "
Dim res2 : res2 = Join(res1, ">")
'res2 now equals " <Head>data</Head> "

```

3. Buscando una cadena

Dado un archivo de texto test.txt :

```

Ford
Jeep
Honda

```

El siguiente script está procesando este archivo de texto:

```

'Read in File Data to an array, separate by newline vb equivalent (vbCrLf)
Dim car, cars
Dim filefullname : filefullname = "C:\testenv\test.txt"
cars = Split(CreateObject("Scripting.FileSystemObject").OpenTextFile(filefullname, 1).ReadAll,
vbCrLf)

'Exact Match search.
Dim searchstring : searchstring = "Jeep"
For Each car In cars
    If Car = searchstring Then Exit For
Next

'Partial Match search
'Instr returns >0 if any result is found, if none, Instr returns -1
'The If statement will use -1 = false, >0 = true
Dim searchstring : searchstring = "Jee"

```

```

Dim position
For car = 0 To ubound(cars)
    If InStr(cars(car), searchstring) Then
        position = car
        Exit For
    End If
Next

```

5. Rellenar la matriz con texto específico de la cadena a través de los caracteres de inicio y final.

'Note: I use this method to extract non-html data in extracted GET telnet results
 'This example effectively grabs every other vehicle that have start and finish
 'characters, which in this case is "/".

'Resulting in an array like this:

'extractedData(0) = "/Jeep"

'extractedData(1) = "/Ford"

'extractedData(2) = "/Honda"

```

Dim combined : combined = Join(cars, "/" ) & "/" & Join(cars, "/" )

```

'combined now equals Ford/Jeep/Honda/Ford/Jeep/Honda

```

Dim record, trigger : record = false : trigger = false

```

```

Dim extractedData() : ReDim extractedData(0)

```

```

For I = 1 to len(combined) 'searching the string one character at a time

```

```

    If trigger Then 'if I've already started recording values

```

```

        If Mid(combined, I, 1) = "/" Then 'End Character is found, stop recording

```

```

            record = false

```

```

            trigger = false

```

```

            ReDim Preserve extractedData(ubound(extractedData)+1) 'Prep next Array Entry

```

```

        End If

```

```

    Else

```

```

        If Mid(combined, I, 1) = "/" Then record = true 'Start recording on start character

```

```

    End If

```

```

    If record Then

```

```

        'Increment text on array entry until end variable is found.

```

```

        extractedData(ubound(extractedData)) = extractedData(ubound(extractedData)) &

```

```

        Mid(combined, I, 1)

```

```

        trigger = true

```

```

    End If

```

```

Next

```

4. Encadenar métodos de manipulación de cuerdas juntos.

```

Dim exStr : exStr = " <Head>data</Head> "

```

```

Dim res

```

```

res = Ucase(Replace(Mid(exStr, instr(exStr, ">")+1,4), "ata", "ark"))

```

'res now equals DARK

'instr(exStr, ">") returns 7

'Mid(" <Head>data</Head> ", 7+1, 4) returns "data"

'Replace("data", "ata", "ark") returns "dark"

'Ucase("dark") returns "DARK"

Lea Instrumentos de cuerda en línea: <https://riptutorial.com/es/vbscript/topic/7540/instrumentos->

de-cuerda

Capítulo 8: Objetos del diccionario

Examples

Crear diccionario y agregar artículos al diccionario

```
Dim oDic
Set oDic = CreateObject("Scripting.Dictionary")
oDic.Add "US", "United States of America"
oDic.Add "UK", "United Kingdom"
```

Comprobar si existe la clave en el diccionario

```
If oDic.Exists("US") Then
    MsgBox "The Key US Exist. The value is " + oDic("US")
Else
    MsgBox "Key Does not exist."
End If
```

Eliminar artículo del diccionario

```
If oDic.Exists("UK") Then
    oDic.remove("UK")
End If
```

Iterar todos los elementos en el diccionario

```
set oDic = CreateObject("Scripting.Dictionary")
oDic.add "USA", "United States of America"
oDic.add "UK", "United Kingdom"
oDic.add "CAN", "Canada"

For Each obj in oDic.Items
    MsgBox obj
Next
Set oDic = Nothing
```

*** Salida:**

Estados Unidos de America

Reino Unido

Canadá

Iterar todas las claves en el diccionario

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

```
oDic.add "USA", "United States of America"  
oDic.add "UK", "United Kingdom"  
oDic.add "CAN", "Canada"  
  
For Each obj in oDic.keys  
    MsgBox "Key: " & obj & " Value: " & oDic(obj)  
Next  
Set oDic = Nothing
```

Eliminar clave / claves del diccionario

```
set oDic = CreateObject("Scripting.Dictionary")  
oDic.add "USA", "United States of America"  
oDic.add "UK", "United Kingdom"  
oDic.add "CAN", "Canada"  
  
' Delete only if Key exists  
If oDic.Exists("UK") Then  
    oDic.Remove "UK"  
End If  
  
' Delete all keys from Dictionary  
oDic.removeAll  
  
Set oDic = Nothing
```

Lea Objetos del diccionario en línea: <https://riptutorial.com/es/vbscript/topic/8232/objetos-del-diccionario>

Capítulo 9: Objetos del sistema de archivos

Examples

Comprobando la existencia de un archivo / carpeta / unidad

Métodos utilizados:

```
.DriveExists(strDrive) returns (True/False)
.FileExists(strFile) returns (True/False)
.FolderExists(strFolder) returns (True/False)
```

El siguiente código comprueba la existencia de un archivo utilizando el método " **FileExists** " de un objeto del sistema de archivos. Para verificar la existencia de una carpeta o una unidad, se puede utilizar el método " **FolderExists** " o " **DriveExists** " respectivamente.

Código:

```
Dim strPath, objFso
strPath = "C:\Users\GS\Desktop\tasks.txt"           'Enter the absolute path of the
File/Folder/Drive
Set objFso = CreateObject("Scripting.FileSystemObject")

'Checking for the File's existence
If objFso.FileExists(strPath) then                 'returns True if the file exists, else False
    MsgBox "File Exists!"
Else
    MsgBox "File does not Exist!"
End If
Set objFso = Nothing
```

Eliminar una carpeta existente y crear una nueva carpeta

Métodos utilizados:

```
.DeleteFolder(FileSpec, Force (True/False))
.CreateFolder(Path)
.DeleteFile(FileSpec, Force (True/False))
```

El siguiente ejemplo ilustra la eliminación y creación de una carpeta utilizando los métodos " **DeleteFolder** " y " **CreateFolder** ".

Código:

```
Dim strFolderPath, objFso
strFolderPath = "C:\Users\GS\Desktop\testFolder"
Set objFso = CreateObject("Scripting.FileSystemObject")

'Checking for the folder's existence and deleting it, if found
If objFso.FolderExists(strFolderPath) then
```

```

        objFso.DeleteFolder strFolderPath, True                'True indicates forceful
deletion
End If

'Creating a new Folder
objFso.CreateFolder strFolderPath

Set objFso = Nothing

```

De manera similar, uno puede eliminar un archivo usando el método " **DeleteFile** ":

```

Dim strFilePath:strFilePath = "C:\Users\GS\Desktop\tasks.txt"
If objFso.FileExists(strFilePath) then
    objFso.DeleteFile strFilePath, True                    'true indicates forceful deletion
End If

```

Copiando un archivo / carpeta

Métodos utilizados:

```

.CopyFile(Source, Dest [,Overwrite (True/False)]
.CopyFolder(Source, Dest [,Overwrite (True/False)]

```

El siguiente código ilustra el uso del método **CopyFile** para copiar un archivo a una nueva ubicación. Lo mismo se puede lograr para las carpetas utilizando el método **CopyFolder** .

Código:

```

Dim objFso, strSourcePath, strDestPath
strSourcePath = "C:\Users\GS\Desktop\Source.txt"
strDestPath = "C:\Users\GS\Desktop\Dest.txt"
Set objFso = CreateObject("Scripting.FileSystemObject")
If objFso.FileExists(strSourcePath) then
    objFso.CopyFile strSourcePath, strDestPath, True        'True indicates the
overwriting of the file at the destination path i.e, if the file already exists, it will be
overwritten
End If
Set objFso = Nothing

```

Mover un archivo / carpeta

Métodos utilizados:

```

.MoveFile(Source, Dest)
.MoveFolder(Source, Dest)

```

El siguiente código ilustra el uso del método **MoveFile** para mover un archivo a una nueva ubicación. Lo mismo se puede lograr para las carpetas utilizando el método **MoveFolder** .

Código:

```
Dim objFso, strSourcePath, strDestPath
strSourcePath = "C:\Users\GS\Desktop\Source.txt"
strDestPath = "C:\Users\GS\Desktop\Folder\Dest.txt"
Set objFso = CreateObject("Scripting.FileSystemObject")
If objFso.FileExists(strSourcePath) then
    objFso.MoveFile strSourcePath, strDestPath
End If
Set objFso = Nothing
```

NOTA: No tenemos ningún método de un objeto de sistema de archivos que nos permita cambiar el nombre de un archivo. Sin embargo, esto se puede lograr con el método **MoveFile** moviendo el archivo a la misma ubicación con un nombre diferente como se muestra a continuación:

```
Dim objFso, strSourcePath, strDestPath
strSourcePath = "C:\Users\GS\Desktop\OldName.txt"
strDestPath = "C:\Users\GS\Desktop\NewName.txt"           'Location is same but the name is
different
Set objFso = CreateObject("Scripting.FileSystemObject")
If objFso.FileExists(strSourcePath) then
    objFso.MoveFile strSourcePath, strDestPath
End If
Set objFso = Nothing
```

Objeto de referencia a una carpeta.

Métodos utilizados:

```
.GetFolder(strPath) - Returns an object referring to the path
```

Podemos establecer una referencia de objeto a una carpeta utilizando el método **getFolder** y realizar diferentes operaciones en ellos.

Código:

```
Dim strFolderPath, objFso, objFolder
strFolderPath = "C:\Users\GS\Desktop\LogsFolder"
Set objFso = CreateObject("Scripting.FileSystemObject")
Set objFolder = objFso.getFolder(strFolderPath)

'Accessing the Folder's Properties
Msgbox objFolder.Name           'Returns the Folder's Name
Msgbox objFolder.Size          'Returns the Folder's size in Bytes
Msgbox objFolder.DateCreated   'Returns the Folder's creation date
Msgbox objFolder.DateLastModified 'Returns the Folder's last modified date
Msgbox objFolder.Path          'Returns the Folder's Absolute Path

Dim objChildFolders
Set objChildFolders = objFolder.SubFolders 'Returns the collection of all subfolder

Dim objChildFiles
Set objChildFiles = objFolder.Files      'Returns the collection of all files
contained in the folder

'Using the Folder's methods
objFolder.Copy strDestPath, True        'Copies the folder to path contained in
```

```

strDestPath and overwrite Flag=True
objFolder.Delete True 'Deletes the Folder; True indicates forceful
Deletion
objFolder.Move strDestPath 'Moves the Folder to the path contained in
strDestPath variable
objFolder.CreateTextFile strFileName, True 'Created a new text file inside the folder
and overwrites the existing file(if it exists)
Set objChildFiles = Nothing
Set objChildFolders = Nothing
Set objFolder = Nothing
Set objFso = Nothing

```

Objeto de referencia a un archivo

Métodos utilizados:

```
.GetFile(strPath) - Returns an object referring to a file.
```

Podemos establecer una referencia de objeto a un archivo utilizando el método **getFile** y realizar diferentes operaciones en ellos.

Código:

```

Dim strFilePath, objFso, objFile
strFilePath = "C:\Users\GS\Desktop\LogsFolder\file.txt"
Set objFso = CreateObject("Scripting.FileSystemObject")
Set objFile = objFso.getFile(strFilePath)

'Accessing the File's Properties
Msgbox objFile.Name 'Returns the File's Name
Msgbox objFile.Size 'Returns the File's size in Bytes
Msgbox objFile.DateCreated 'Returns the File's creation date
Msgbox objFile.DateLastModified 'Returns the File's last modified date
Msgbox objFile.Path 'Returns the File's absolute path

'Using the File's Methods
objFile.Delete True 'Forcefully deletes the File
objFile.Copy strDestPath, True 'Copies the file to path contained in variable
strDestPath
objFile.Move strDestPath 'Moves the file to the path contained in the
variable strDestPath
objFile.OpenAsTextStream mode 'Opens the file as a text stream in either Read
mode(mode=1), write mode(mode=2) or Append mode(mode=8)
Set objFile = Nothing
Set objFso = Nothing

```

Lea Objetos del sistema de archivos en línea:

<https://riptutorial.com/es/vbscript/topic/10062/objetos-del-sistema-de-archivos>

Capítulo 10: Usando clases

Examples

Creando una clase

```
Class Car

    Private wheels_
    Private distances_

    ' Property getter
    Public Property Get Wheels()
        Wheels = wheels_
    End Property

    ' Property setter
    Public Property Let Wheels(v)
        wheels_ = v
    End Property

    ' Parameterless Constructor
    Public Sub Class_Initialize()
        distances_ = Array(0)
    End Sub

    ' Method
    Public Function GetTotalDistance()
        dim d
        'GetTotalDistance = 0
        For Each d in distances_
            GetTotalDistance = GetTotalDistance + d
        Next
    End Function

    ' Void Method
    Public Sub Drive(distance)
        distances_(ubound(distances_)) = distance
        Redim Preserve distances_(ubound(distances_)+1)
    End Sub

End Class
```

Usando una instancia de clase

```
' Initialize the object
Dim myCar
Set myCar = new Car

' Setting a property
myCar.Wheels = 4

' Getting a property value
wscript.echo myCar.Wheels
```

```
' Using a subroutine in a class
myCar.Drive 10
myCar.Drive 12

' Using a function in a class
wscript.echo myCar.GetTotalDistance() ' returns 22
```

Función de fábrica global para emular un constructor parametrizado

```
' Making a factory with parameter to the class
Public Function new_Car(wheels)
    Set new_Car = New Car
    new_Car.Wheels = wheels
End Function

' Creating a car through a factory
Dim semiTrailer
Set semiTrailer = new_Car(18)
```

Método inicial para emular un constructor parametrizado

```
Class Car
    ...
    ' Parameterless Constructor
    Public Sub Class_Initialize()
        distances_ = Array(0)
    End Sub

    ' Default initialization method that can be invoked without
    ' explicitly using the method name.
    Public Default Function Init(wheels)
        wheels_ = wheels
        Set Init = Me
    End Function
    ...
End Class

Set car1 = (New Car) (18) ' implicit invocation
Set car2 = (New Car).Init(8) ' explicit invocation
```

Cargando archivos de clase externos en script.

```
Dim classFile : classFile = "carClass.vbs"
Dim fsObj : Set fsObj = CreateObject("Scripting.FileSystemObject")
Dim vbsFile : Set vbsFile = fsObj.OpenTextFile(classFile, 1, False)
Dim myFunctionsStr : myFunctionsStr = vbsFile.ReadAll
vbsFile.Close
Set vbsFile = Nothing
Set fsObj = Nothing
ExecuteGlobal myFunctionsStr

Dim car1 : Set car1 = (New Car) (18)
```

Lea Usando clases en línea: <https://riptutorial.com/es/vbscript/topic/3911/usando-clases>

Creditos

S. No	Capítulos	Contributors
1	Empezando con vbscript	Community , Derpcode , Ekkehard.Horner , Lankymart , Martha , Nathan Tuggy , Purendra Agrawal
2	Arreglos y Bucles	Rich , Wolf
3	Consultas WMI	Abhishek
4	Creando tu primer guión	jondanson
5	Incluir archivos	Foxtrot Romeo
6	InputBox	Ansgar Wiechers , Macro Man , Shawn V. Wilson
7	Instrumentos de cuerda	Foxtrot Romeo , Rich , Wolf
8	Objetos del diccionario	Barney
9	Objetos del sistema de archivos	Gurman
10	Usando clases	Ansgar Wiechers , AutomatedChaos , Rich