

 eBook Gratuit

# APPRENEZ

---

# vbscript

eBook gratuit non affilié créé à partir des  
**contributeurs de Stack Overflow.**

#vbscript

# Table des matières

À propos.....	1
<b>Chapitre 1: Démarrer avec vbscript.....</b>	<b>2</b>
Remarques.....	2
Versions.....	2
Exemples.....	2
Message Hello World en utilisant cscript et wscript.....	2
<b>Chapitre 2: Cordes.....</b>	<b>4</b>
Remarques.....	4
Exemples.....	4
1. Chaîne standard.....	4
2. Les bases de la manipulation des cordes.....	4
3. Rechercher une chaîne.....	5
5. Remplir le tableau avec un texte spécifique à partir d'une chaîne via des caractères de.....	6
4. Enchaînant ensemble les méthodes de manipulation des chaînes.....	6
<b>Chapitre 3: Créer votre premier script.....</b>	<b>7</b>
Introduction.....	7
Paramètres.....	7
Exemples.....	7
Bonjour le monde.....	7
Explication.....	8
<b>Chapitre 4: Fichiers à inclure.....</b>	<b>9</b>
Introduction.....	9
Remarques.....	9
Exemples.....	9
Créer une méthode "include file".....	9
Y compris les fichiers.....	10
Initialisation globale.....	10
<b>Chapitre 5: Objets Dictionnaire.....</b>	<b>11</b>
Exemples.....	11
Créer un dictionnaire et ajouter des éléments au dictionnaire.....	11

Vérifier si la clé existe dans le dictionnaire.....	11
Supprimer un élément du dictionnaire.....	11
Itérer tous les éléments du dictionnaire.....	11
Itérer toutes les clés du dictionnaire.....	11
Supprimer la clé / les clés du dictionnaire.....	12
<b>Chapitre 6: Objets FileSystem.....</b>	<b>13</b>
Exemples.....	13
Vérification de l'existence d'un fichier / dossier / lecteur.....	13
Suppression d'un dossier existant et création d'un nouveau dossier.....	13
Copier un fichier / dossier.....	14
Déplacement d'un fichier / dossier.....	14
Référence d'objet à un dossier.....	15
Référence d'objet à un fichier.....	16
<b>Chapitre 7: Requêtes WMI.....</b>	<b>17</b>
Introduction.....	17
Exemples.....	17
Extraire le nom du PC local.....	17
Obtenir le nombre d'instances de n'importe quel processus.....	17
Obtenir la résolution d'écran du moniteur actif.....	17
<b>Chapitre 8: Tableaux et boucles.....</b>	<b>19</b>
Exemples.....	19
1. Tableaux - Statique.....	19
2. tableaux - dynamique.....	19
5. Créer un tableau à partir d'un fichier texte.....	19
7. Pour chaque boucle.....	19
6. Pour la boucle.....	20
8. Faites en boucle.....	20
9. Do Until Loop.....	20
3. Tableaux - multidimensionnels.....	20
4. Tableaux - multidimensionnels - dynamiques.....	21
<b>Chapitre 9: Utiliser des classes.....</b>	<b>22</b>
Exemples.....	22

Créer une classe.....	22
Utilisation d'un instance de classe.....	22
Fonction Factory Global pour émuler un constructeur paramétré.....	23
Méthode d'initialisation pour émuler un constructeur paramétré.....	23
Chargement de fichiers de classe externes dans un script.....	23
<b>Chapitre 10: Zone de saisie.....</b>	<b>24</b>
Syntaxe.....	24
Paramètres.....	24
Remarques.....	24
Exemples.....	24
Utilisez InputBox pour attribuer une entrée utilisateur à une chaîne.....	24
<b>Crédits.....</b>	<b>26</b>

---

# À propos

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](mailto:info@zzzprojects.com)

---

# Chapitre 1: Démarrer avec vbscript

## Remarques

VBScript (VBS) est un langage de script à saveur Visual Basic pour Internet Explorer et Windows. Il peut en principe être utilisé sur le Web, comme JavaScript, mais n'a pas beaucoup de support. Il est donc généralement limité aux scripts autonomes ou côté serveur dans les environnements professionnels utilisant exclusivement Windows.

## Versions

Version	Date de sortie
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

## Exemples

### Message Hello World en utilisant cscript et wscript

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

Cela affiche un message sur la console si elle est exécutée avec `cscript.exe` (l'hôte de la console) ou dans une boîte de message si elle est exécutée avec `wscript.exe` (l'hôte de l'interface graphique).

Si vous utilisez VBScript comme langage de script côté serveur pour une page Web (pour ASP classique, par exemple),

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

met le message dans l'envoi HTML au client (navigateur).

Si vous souhaitez afficher un message dans la boîte de message, vous pouvez utiliser:

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

Lire Démarrer avec vbscript en ligne: <https://riptutorial.com/fr/vbscript/topic/463/demarrer-avec-vbscript>

# Chapitre 2: Cordes

## Remarques

Fonctions Date / Heure, Chaîne et Numérique MSDN

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

## Exemples

### 1. Chaîne standard

En vbscript, un objet n'a pas nécessairement besoin d'un type désigné. Semblable à la variable var de C #.

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

### 2. Les bases de la manipulation des cordes

```
'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 = InstrRev(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. Rechercher une chaîne

Étant donné un fichier texte test.txt :

```

Ford
Jeep
Honda

```

Le script suivant traite ce fichier texte:

```

'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. Remplir le tableau avec un texte spécifique à partir d'une chaîne via des caractères de début et de fin.

```

'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. Enchaînant ensemble les méthodes de manipulation des chaînes.

```

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"

```

Lire Cordes en ligne: <https://riptutorial.com/fr/vbscript/topic/7540/cordes>

# Chapitre 3: Créer votre premier script

## Introduction

Pour commencer, dans Windows, créez un document texte sur votre bureau (cliquez avec le bouton droit de la souris> Nouveau> Document texte). Modifiez l'extension de ".txt" à ".vbs". À ce stade, il est exécutable en double-cliquant dessus (rien ne se passera si vous essayez, il n'y a rien encore.) Pour modifier, cliquez avec le bouton droit de la souris sur le document et cliquez sur Modifier. Ajoutez l'exemple de code pour votre premier programme.

## Paramètres

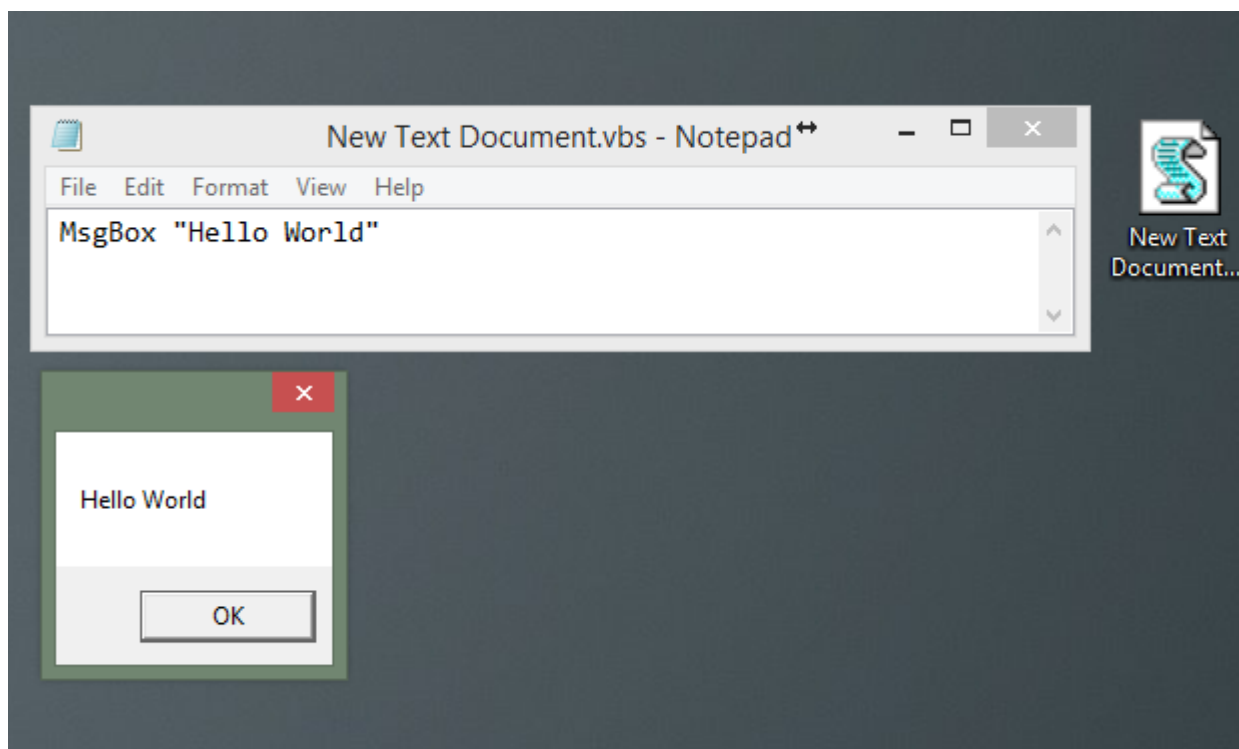
Colonne	Colonne
Cellule	Cellule

## Exemples

### Bonjour le monde

Juste un simple monde bonjour pour commencer. Copiez collez le ci-dessous dans le document, sauvegardez puis double-cliquez.

```
MsgBox "Hello World"
```



## Explication

"MsgBox" affiche un message dans une boîte de dialogue et attend que l'utilisateur réponde. C'est une bonne méthode pour informer les utilisateurs des actions à effectuer ou simplement de la fin du script.

Tel que:

Lire [Créer votre premier script en ligne](https://riptutorial.com/fr/vbscript/topic/8036/creer-votre-premier-script): <https://riptutorial.com/fr/vbscript/topic/8036/creer-votre-premier-script>

# Chapitre 4: Fichiers à inclure

## Introduction

Lorsque vous exécutez VbScript dans un shell Windows, il n'y a pas de fonction intégrée pour inclure un fichier. Par conséquent, pour organiser votre code dans différents fichiers, vous devez créer une méthode pour ce faire.

## Remarques

Quelques points à garder à l'esprit lors de l'utilisation de la `IncludeFile(p_Path)` :

- Il n'y a pas de limitation du type de fichier pouvant être inclus, mais le contenu des fichiers inclus doit être VbScript.
- S'il y a une erreur de syntaxe dans le fichier inclus, vous n'obtiendrez pas la ligne / colonne de l'erreur.
- Vous devez définir et initialiser `std_internal_LibFiles` avant le premier appel à `IncludeFile(p_Path)`
- Vous pouvez utiliser `IncludeFile(p_Path)` n'importe où dans votre code, y compris d'autres méthodes.

## Exemples

### Créer une méthode "include file"

Le but principal de cette fonction est donc de:

- Soyez autonome car il doit être écrit dans le fichier VbScript principal et ne peut pas être dans un fichier inclus (car il définit la fonction include)
- Fournir suffisamment d'informations si quelque chose ne va pas (c.-à-d. Le fichier qui était inclus, l'erreur qui s'est produite, ...)
- Inclure un fichier une fois et une seule fois pour éviter d'inclure des boucles.

```
'
*****

'! 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

```

## Y compris les fichiers

Pour inclure un fichier dans un autre fichier, utilisez simplement le liner:

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

## Initialisation globale

Avant d'utiliser la méthode IncludeFile, nous devons:

- Déclarez `std_internal_LibFiles` globalement
- Initialisez-le avec un nouveau dictionnaire

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

Lire Fichiers à inclure en ligne: <https://riptutorial.com/fr/vbscript/topic/8345/fichiers-a-inclure>

# Chapitre 5: Objets Dictionnaire

## Exemples

### Créer un dictionnaire et ajouter des éléments au dictionnaire

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

### Vérifier si la clé existe dans le dictionnaire

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

### Supprimer un élément du dictionnaire

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

### Itérer tous les éléments du dictionnaire

```
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
```

#### \* **Sortie:**

les États-Unis d'Amérique

Royaume-Uni

Canada

### Itérer toutes les clés du dictionnaire

```
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
```

## Supprimer la clé / les clés du dictionnaire

```
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
```

Lire Objets Dictionnaire en ligne: <https://riptutorial.com/fr/vbscript/topic/8232/objets-dictionnaire>



# Chapitre 6: Objets FileSystem

## Exemples

### Vérification de l'existence d'un fichier / dossier / lecteur

#### Méthodes utilisées:

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

Le code suivant vérifie l'existence d'un fichier à l'aide de la méthode " **FileExists** " d'un objet du système de fichiers. Pour vérifier l'existence d'un dossier ou d'un lecteur, on peut utiliser respectivement la méthode " **FolderExists** " ou " **DriveExists** ".

#### Code:

```
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
```

### Suppression d'un dossier existant et création d'un nouveau dossier

#### Méthodes utilisées:

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

L'exemple suivant illustre la suppression et la création d'un dossier à l'aide des méthodes " **DeleteFolder** " et " **CreateFolder** ".

#### Code:

```
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 même, on peut supprimer un fichier en utilisant la méthode " **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

```

## Copier un fichier / dossier

### Méthodes utilisées:

```

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

```

Le code suivant illustre l'utilisation de la méthode **CopyFile** pour copier un fichier vers un nouvel emplacement. La même chose peut être obtenue pour les dossiers en utilisant la méthode **CopyFolder** .

### Code:

```

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

```

## Déplacement d'un fichier / dossier

### Méthodes utilisées:

```

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

```

Le code suivant illustre l'utilisation de la méthode **MoveFile** pour déplacer un fichier vers un nouvel emplacement. La même chose peut être obtenue pour les dossiers en utilisant la méthode **MoveFolder** .

### Code:

```
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
```

**NOTE:** Nous n'avons aucune méthode d'objet de système de fichiers qui nous permet de renommer un fichier. Cependant, cela peut être réalisé par la méthode **MoveFile** en déplaçant le fichier vers le même emplacement avec un nom différent, comme indiqué ci-dessous:

```
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
```

## Référence d'objet à un dossier

### Méthodes utilisées:

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

Nous pouvons définir une référence d'objet à un dossier à l'aide de la méthode **getFolder** et effectuer différentes opérations sur ceux-ci.

### Code:

```
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

```

## Référence d'objet à un fichier

### Méthodes utilisées:

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

Nous pouvons définir une référence d'objet à un fichier à l'aide de la méthode **getFile** et effectuer différentes opérations sur ces fichiers.

### Code:

```

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

```

Lire Objets FileSystem en ligne: <https://riptutorial.com/fr/vbscript/topic/10062/objets-filesystem>

---

# Chapitre 7: Requêtes WMI

## Introduction

VBScript peut interroger Windows Management Instrumentation (WMI) pour obtenir des informations vitales sur le PC local et distant. Nous pouvons utiliser les requêtes WMI pour effectuer diverses tâches telles que l'extraction du nom du PC, l'obtention de la résolution de l'écran, l'obtention d'informations sur l'utilisateur et le nom d'utilisateur, l'extraction d'informations essentielles sur un processus, la modification des paramètres système de base, etc.

Vous trouverez ci-dessous des exemples d'utilisation de requêtes WMI pour effectuer des tâches spécifiques.

## Exemples

### Extraire le nom du 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
```

Ce code fera écho au nom du PC dans lequel il est exécuté.

### Obtenir le nombre d'instances de n'importe quel processus

```
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."
```

### Obtenir la résolution d'écran du moniteur actif

```
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
```

Lire Requêtes WMI en ligne: <https://riptutorial.com/fr/vbscript/topic/9572/requetes-wmi>

# Chapitre 8: Tableaux et boucles

## Exemples

### 1. Tableaux - Statique

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

### 2. tableaux - dynamique

```
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. Créer un tableau à partir d'un fichier texte.

```
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. Pour chaque boucle.

Vous ne pouvez pas modifier le contenu du tableau via la variable de boucle car il s'agit d'un élément temporaire auquel chaque élément est affecté.

```
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 & vbCrLf
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. Pour la boucle

```

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. Faites en boucle

```

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. Do Until 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. Tableaux - multidimensionnels

```

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. Tableaux - multidimensionnels - dynamiques

```
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)
```

Lire Tableaux et boucles en ligne: <https://riptutorial.com/fr/vbscript/topic/7520/tableaux-et-boucles>

# Chapitre 9: Utiliser des classes

## Exemples

### Créer une classe

```
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
```

### Utilisation d'un instance de classe

```
' 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
```

## Fonction Factory Global pour émuler un constructeur paramétré

```
' 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)
```

## Méthode d'initialisation pour émuler un constructeur paramétré

```
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
```

## Chargement de fichiers de classe externes dans un 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)
```

Lire Utiliser des classes en ligne: <https://riptutorial.com/fr/vbscript/topic/3911/utiliser-des-classes>

# Chapitre 10: Zone de saisie

## Syntaxe

- `InputBox` (`invite` [, `titre`] [, `par défaut`] [, `xpos`] [, `ypos`] [, `fichier d'aide`, `contexte`])

## Paramètres

Argument	Détail
<code>prompt</code>	Texte à afficher au-dessus du champ de saisie (généralement une instruction indiquant ce qui est requis pour l'utilisateur).
<code>title</code>	Légende affichée dans la barre de titre de la zone de saisie.
<code>default</code>	Un espace réservé pour le champ de texte, utilisé comme valeur de retour si l'utilisateur n'écrase pas.
<code>xpos</code>	Distance horizontale en twips pour afficher la zone de saisie à partir du bord gauche de l'écran.
<code>ypos</code>	Distance verticale en twips pour afficher la zone de saisie à partir du bord supérieur de l'écran.
<code>helpfile</code>	Une chaîne pour déterminer le fichier d'aide à utiliser afin de fournir une aide contextuelle avec la zone de saisie.
<code>context</code>	Si l'argument <code>helpfile</code> est fourni, un numéro de contexte doit également être fourni pour identifier la rubrique d'aide appropriée dans le fichier d'aide.

## Remarques

<sup>1</sup> *Tous les arguments sauf `prompt` sont facultatifs.*

<sup>2</sup> *`helpfile` et le `context` sont couplés - si l'un est fourni, l'autre doit également être fourni.*

## Exemples

### Utilisez `InputBox` pour attribuer une entrée utilisateur à une chaîne

```
' 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
```

Lire Zone de saisie en ligne: <https://riptutorial.com/fr/vbscript/topic/634/zone-de-saisie>

# Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec vbscript	<a href="#">Community</a> , <a href="#">Derpcode</a> , <a href="#">Ekkehard.Horner</a> , <a href="#">Lankymart</a> , <a href="#">Martha</a> , <a href="#">Nathan Tuggy</a> , <a href="#">Purendra Agrawal</a>
2	Cordes	<a href="#">Foxtrot Romeo</a> , <a href="#">Rich</a> , <a href="#">Wolf</a>
3	Créer votre premier script	<a href="#">jondanson</a>
4	Fichiers à inclure	<a href="#">Foxtrot Romeo</a>
5	Objets Dictionnaire	<a href="#">Barney</a>
6	Objets FileSystem	<a href="#">Gurman</a>
7	Requêtes WMI	<a href="#">Abhishek</a>
8	Tableaux et boucles	<a href="#">Rich</a> , <a href="#">Wolf</a>
9	Utiliser des classes	<a href="#">Ansgar Wiechers</a> , <a href="#">AutomatedChaos</a> , <a href="#">Rich</a>
10	Zone de saisie	<a href="#">Ansgar Wiechers</a> , <a href="#">Macro Man</a> , <a href="#">Shawn V. Wilson</a>