VB Script Advanced

Index

VB Script Advanced - Arrays

What is an Array?
We know very well that a variable is a container to store a value. Sometimes, developers are in a position to hold more than one value in a single variable at a time. When a series of values are stored in a single variable, then it is known as array variable.

Array Declaration
Arrays are declared the same way a variable has been declared except that the declaration of an array variable uses parenthesis. In the below example, the size of the array is mentioned in the brackets.

```
' 1 : Using Dim
Dim myarr1()                'Without Size

' 2 : Mentioning the Size
Dim myarr2(7)  'Declared with size of 8

' 3 : using 'Array' Parameter
Dim myarr3
myarr3 = Array("one","Two","Three")
```

1. Although, the Array size is indicated as 7, it can hold 8 values as array index starts from ZERO.

2. Array Index Cannot be Negative.

3. VBScript Arrays can store any type of variable in an array. Hence, an array can store an integer, string or characters in a single array variable.

VB Script Fundamentals - Arrays

```
Dim arr(5)
arr(0) = "1"                'Number as String
arr(1) = "VBScript"    'String
arr(2) = 100                'Number
arr(3) = 2.45               'Decimal Number
arr(4) = #10/07/2013#  'Date
arr(5) = #12.45 PM#    'Time

msgbox "Value stored in Array index 0 : " & arr(0)
msgbox "Value stored in Array index 1 : " & arr(1)
msgbox "Value stored in Array index 2 : " & arr(2)
msgbox "Value stored in Array index 3 : " & arr(3)
msgbox "Value stored in Array index 4 : " & arr(4)
msgbox "Value stored in Array index 5 : " & arr(5)
```

VB Script Advanced - Arrays

Multi Dimension Arrays

Arrays are not just limited to single dimension and can have a maximum of 60 dimensions. Two-dimension arrays are the most commonly used ones.

```
Dim arr(2,2)            ' Which has 3 rows and 3 columns
arr(0,0) = "One"
arr(0,1) = "Two"
arr(0,2) = "Three"
arr(0,3) = "Four"
arr(1,0) = "Five"
arr(1,1) = "Six"
arr(1,2) = "Seven"
arr(1,3) = "Eight"
arr(2,0) = "Nine"
arr(2,1) = "Ten"
arr(2,2) = "Eleven"


msgbox "Value in Array index 0,1 : " &  arr(0,1)
msgbox "Value in Array index 2,2 : " &  arr(2,2)
```

VB Script Fundamentals - Arrays

Redim Statement

ReDim Statement is used to Declare dynamic-array variables and allocate or reallocate storage space.

ReDim [Preserve] varname(subscripts) [, varname(subscripts)]

•Preserve - An Optional parameter used to preserve the data in an existing array when you change the size of the last dimension.
•varname - A Required parameter, which denotes Name of the variable, which should follow the standard variable naming conventions.
•subscripts - A Required parameter, which indicates the size of the array.

```
Dim a()
 i=0
 redim  a(5)
 a(0)="XYZ"
 a(1)=41.25
 a(2)=22
REDIM PRESERVE a(7)
 For i=3 to 7
 a(i)= i
 Next
 'to Fetch the output
 For i=0 to ubound(a)
   Msgbox a(i)
 Next
```

VB Script Advanced - Arrays

Array Methods
There are various inbuilt functions within VBScript which help the developers to handle arrays effectively.

Function	Description
LBound	A Function, which returns an integer that corresponds to the smallest subscript of the given arrays.
UBound	A Function, which returns an integer that corresponds to the Largest subscript of the given arrays.
Split	A Function, which returns an array that contains a specified number of values. Splitted based on a Delimiter.
Join	A Function, which returns a String that contains a specified number of substrings in an array. This is an exact opposite function of Split Method.
Filter	A Function, which returns a zero based array that contains a subset of a string array based on a specific filter criteria.
IsArray	A Function, which returns a boolean value that indicates whether or not the input variable is an array.
Erase	A Function, which recovers the allocated memory for the array variables.

VB Script Fundamentals - Arrays

Function	Description
LBound	A Function, which returns an integer that corresponds to the smallest subscript of the given arrays.
UBound	A Function, which returns an integer that corresponds to the Largest subscript of the given arrays.
	days= Array("Sun","Mon","Tue","Wed","Thu","Fri","Sat") msgbox LBound(days) ‘ Returns 0 Msgbox UBound(days) ‘ Returns 6
Split	**A Function, which returns an array that contains a specified number of values. Splitted based on a Delimiter.** **Split**(expression[,delimiter[,count[,compare]]]) Splitting based on delimiter comma ‘@' a=Split("One @ Two @ Three",”@") b=ubound(a) For i=0 to b msgbox "The value of array in " & i & " is :" & a(i) Next

VB Script Advanced - Arrays

Function	Description
Join	A Function, which returns a String that contains a specified number of substrings in an array. This is an exact opposite function of Split Method. **Join**(List[,delimiter]) ' Join using spaces a = array("Red","Blue","Yellow") b = join(a) msgbox "The value of b " & " is :" & b ' Join using $ b = join(a,"$") msgbox "The Join result after using delimiter is : " & b

Filter	A Function, which returns a zero based array that contains a subset of a string array based on a specific filter criteria.
	
	Filter(inputstrings,value[,include[,compare]])
	
	a= array("Red","Blue","Yellow")
	b = Filter(a,"B")
	c = Filter(a,"e")
	d = Filter(a,"Y")
	For each x in b
	  msgbox "The Filter result 1: " & x
	Next
	For each y in c
	  msgbox "The Filter result 2: " & y
	Next
	For each z in d
	  msgbox "The Filter result 3: " & z

VB Script Fundamentals - Arrays

Function	Description
IsArray	A Function, which returns a boolean value that indicates whether or not the input variable is an array. a = array("Red","Blue","Yellow") b = "67919" msgbox "The IsArray result 1 : " & IsArray(a) msgbox "The IsArray result 2 : " & IsArray(b)
Erase	A Function, which recovers the allocated memory for the array variables. Dim NumArray(3) NumArray(0) = "VBScript" NumArray(1) = 4.05 NumArray(2) = 15 NumArray(3) = #21/06/2015# Dim DynamicArray() ReDim DynamicArray(9) ' Allocate memory space. Erase NumArray ' Each element is reinitialized. Erase DynamicArray ' Free memory used by array. ' All values would be erased. msgbox "The value at Zeroth index of NumArray is " & NumArray(0) msgbox "The value at First index of NumArray is " & NumArray(1) msgbox "The value at Second index of NumArray is " & NumArray(2) msgbox "The value at Third index of NumArray is " & NumArray(3)

VB Script Advanced - File System Object [FSO]

Working with Files using FSO

FSO:

File system object is an object model which is used to handle the drives, folders, and files of a system or server.

♦ If an user needs to work on Driver, Folder, Files properties,methods or events then the first step he need to setup is filesystemobject

♦ File system object is an interface between QTP and the local system. using FSO we can create/delete folder, create/delete/read from/write to text files

♦ The FileSystemObject (FSO) object model allows you to use the familiar object method syntax with a rich set of properties, methods, and events to process folders and files

VB Script Advanced - File System Object [FSO]

Object/Collection Description:

FileSystemObject:

File system object is a Main object. Contains methods and properties that allow you to create, delete, gain information about, and generally manipulate drives, folders, and files. Many of the methods associated with this object duplicate those in other FSO objects; they are provided for convenience.

Drive:

Drive is a Object. Contains methods and properties that allow you to gather information about a drive attached to the system, such as its share name and how much room is available. Note that a "drive" isn't necessarily a hard disk, but can be a CD-ROM drive, a RAM disk, and so forth. A drive doesn't need to be physically attached to the system; it can be also be logically connected through a network.

Drives:

Drives are Collection. Provides a list of the drives attached to the system, either physically or logically. The Drives collection includes all drives, regardless of type. Removable-media drives need not have media inserted for them to appear in this collection.

VB Script Advanced - File System Object [FSO]

File:

File is a Object. Contains methods and properties that allow you to create, delete, or move a file. Also allows you to query the system for a file name, path, and various other properties.

Files:

Files are Collection. Provides a list of all files contained within a folder.

Folder:

Folder is a Object. Contains methods and properties that allow you to create, delete, or move folders. Also allows you to query the system for folder names, paths, and various other properties.

Folders:

Folders are Collection. Provides a list of all the folders within a Folder.

TextStream:

TextStream is a Object. Allows you to read and write text files.

VB Script Advanced - File System Object [FSO]

Creating a FileSystemObject Object

First, create a FileSystemObject object by using the CreateObject method.

The following code displays how to create an instance of the FileSystemObject:

```
Dim fso
Set fso = CreateObject("Scripting.FileSystemObject")
```

VB Script Advanced - File System Object [FSO]

Method: CreateTextFile

Description: Creates a specified file name and returns a TextStream object that can be used to read from or write to the file

Syntax: Set objfile = fso.CreateTextFile(filename[, overwrite[, unicode]])

Example:

```
'Create a filesystemObject
Set fso=createobject("Scripting.FileSystemObject")
'Create a non existing file "qtptest.txt " with overwrite option as True
Set qfile1=fso.CreateTextFile("C:\qtptest.txt",True,False)
'Output --> New File "qtptest.txt " is created

'Close the files
qfile1.Close
'Release the allocated objects
Set qfile1=nothing
```

Example:

```
'Create a filesystemObject
Set fso=createobject("Scripting.FileSystemObject")
'Create a   file "qtptest.txt "  in C Drive .
'Then  run the below statement with overwrite option as False
 'Output --> Error message "Fie already exists" is displayed
Set qfile2=fso.CreateTextFile("C:\qtpexist.txt",False,False)
Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: CopyFile
Description: Copies one or more files from one location to a new location
Syntax: fso.CopyFile (source, destination[, overwrite])

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'File to be copied Sourcefile="C:\copy.txt"'Dest folder where the file has to be copied
Destination="D:\final1\"

'If the destination does not exist then create the destination folder

If fso.FolderExists(Destination) = false Then
   fso.CreateFolder (Destination)
End If

'Copy the file

fso.CopyFile Sourcefile,Destination,True
Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: DeleteFile

Description: Deletes a specified file

Syntax: fso.DeleteFile (filename[, force])

Example:

```
Set fso=createobject("Scripting.FileSystemObject")
'File to be  deleted.
Sourcefile="C:\copy.txt"   'Delete the file
fso.DeleteFile Sourcefile

Set fso=nothing
```

Method: CreateFolder

Description: Creates a new folder in the specified location

Syntax: fso.CreateFolder(foldername)

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'Folder to be created
Foldername="D:\Folder_create"
'If the folder doenot exst then create the folder
If fso.FolderExists(Foldername) = false Then

 fso.CreateFolder (Foldername)

End If

Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: CopyFolder

Description: Copies a folder to a new location

Syntax: fso.CopyFolder (source, destination[, overwrite])

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'Folder to be created

SourcePath="D:\Folder_create"

DestinationPath="D:\Destination\"

'If the folder does not exist then create the folder

If fso.FolderExists(DestinationPath)  = false Then
  fso.CreateFolder (DestinationPath)
End If

fso.CopyFolder  SourcePath,DestinationPath,True
Set fso=nothing
```

Method: MoveFolder

Description: Moves one or more folders from one location to another.

Syntax: fso.MoveFolder (source, destination)

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'Folder to be created

SourcePath="D:\Folder_move"
DestinationPath="D:\Destination\"

'If the folder doesnot exst then create the folder

If fso.FolderExists(DestinationPath)  = false Then

 fso.CreateFolder (DestinationPath)
End If

fso.MoveFolder  SourcePath,DestinationPath

Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: DeleteFolder

Description: Deletes the specified folder and its contents

Syntax: fso.DeleteFolder (folderspec[, force])

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'Folder to be  deleted.

FolderDel="D:\final1"

'Delete the folder
fso.DeleteFolder(FolderDel)

Set fso=nothing
```

Method: DriveExists

Description: Determines whether or not a specified drive exists

Syntax: fso.DriveExists (drivespec)

Example:

```
  Set fso=createobject("Scripting.FileSystemObject")

    'The drive to check the existence
  drivepath="D:\"

 If fso.DriveExists(drivepath)  then
    msgbox  "Drive Exists"
Else
   Msgbox "Drive doesnot Exist"
End If

Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: FileExists

Description: Determines whether or not a specified file exists

Syntax: fso.FileExists (filespec)

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'The file to check the existence

filepath="D:\qtptest.txt"

If fso.FileExists(filepath) then
  msgbox "File Exists"
Else
  Msgbox "File doesnot Exist"
End If

Set fso=nothing
```

Method: FolderExists

Description: Determines whether or not a specified folder exists

Syntax: fso.FolderExists (folderspec)

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

'The Folder to check the existence

folderpath="D:\qtp"

If fso.FolderExists(folderpath)  then
  msgbox "Folder Exists"
Else
  Msgbox "Folder doesnot Exist"
End If

Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Text Stream Object Methods:

Method: Close
Description: Closes an open TextStream file
Syntax: objTso.Close

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

Set qfile=fso.OpenTextFile("C:\qtptest.txt",2,True)

      qfile.Write  "Welcome to the World of QTP"

      qfile.Write "the file name is qtptest.txt"

Set qfile=fso.OpenTextFile("C:\qtptest.txt",1,True)

   Do while qfile.AtEndOfStream  <> true
          Msgbox  qfile.ReadLine
    Loop

       qfile.Close

Set qfile=nothing
Set fso=nothing
```

Method: Read
Description: Reads a specified number of characters from a TextStream file and returns the resulting string.
Syntax: strChars = objTso.Read(numCharacters)

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

Set qfile=fso.OpenTextFile("C:\qtptest.txt",2,True)

      qfile.Writeline  "Welcome to the World of QTP"

      qfile.Writeline  "the file name is qtptest.txt"

Set qfile=fso.OpenTextFile("C:\qtptest.txt",1,True)

     Msgbox  qfile.Read(10)

      qfile.Close

Set qfile=nothing
Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: ReadAll
Description: Reads the entire TextStream file and returns the resulting string.
Syntax: strChars = objTso.ReadAll

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

Set qfile=fso.OpenTextFile("C:\qtptest.txt",2,True)

qfile.Writeline  "Welcome to the World of QTP"
qfile.Writeline  "the file name is qtptest.txt"

Set qfile=fso.OpenTextFile("C:\qtptest.txt",1,True)


Msgbox  qfile.ReadAll

qfile.Close

Set qfile=nothing
Set fso=nothing
```

Method: ReadLine
Description: Reads an entire line (up to, but not including, the newline character) from a TextStream file and returns the resulting string.
Syntax: strChars = objTso.ReadLine

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

Set qfile=fso.OpenTextFile("C:\qtptest.txt",2,True)

    qfile.Writeline  "Welcome to the World of QTP"
    qfile.Writeline  "the file name is qtptest.txt"

Set qfile=fso.OpenTextFile("C:\qtptest.txt",1,True)

Do while qfile.AtEndOfStream  <> true
    Msgbox  qfile.ReadLine
Loop

qfile.Close

Set qfile=nothing
Set fso=nothing
```

VB Script Advanced - File System Object [FSO]

Method: Write:
Description: Writes a specified string to a TextStream file.
Syntax: objTso.Write(string)

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

Set qfile=fso.OpenTextFile("C:\qtptest.txt",2,True)
      qfile.Write  "Welcome to the World of QTP"
      qfile.Write "the file name is qtptest.txt"
Set qfile=fso.OpenTextFile("C:\qtptest.txt",1,True)

Do while qfile.AtEndOfStream  <> true
     Msgbox  qfile.ReadLine
Loop

    qfile.Close

Set qfile=nothing
Set fso=nothing
```

Method: WriteLine
Description: Writes a specified string and newline character to TextStream file.
Syntax: objTso.WriteLine([string])

Example:

```
Set fso=createobject("Scripting.FileSystemObject")

Set qfile=fso.OpenTextFile("C:\qtptest.txt",2,True)
      qfile.Writeline  "Welcome to the World of QTP"
      qfile.Writeline  "the file name is qtptest.txt"
Set qfile=fso.OpenTextFile("C:\qtptest.txt",1,True)

Do while qfile.AtEndOfStream  <> true
      Msgbox  qfile.ReadLine
Loop

      qfile.Close

Set qfile=nothing
Set fso=nothing
```

VB Script Advanced -Database

UFT Scripts for connecting to MS Access:

```
Option Explicit
Dim con,rs

Set con=createobject("adodb.connection")
Set rs=createobject("adodb.recordset")

con.open "Driver={Microsoft Access Driver (*.mdb)};Dbq=C:\mydatabase.mdb;Uid=Admin;Pwd=;"

rs.open "select * from emp",con

Do while not rs.eof
VbWindow("Form1").VbEdit("val1").Set rs.fields("v1")
VbWindow("Form1").VbEdit("val2").Set rs.fields("v2")
VbWindow("Form1").VbButton("ADD").Click
rs.movenext
Loop

'Release objects'Release objects
Set rs= nothing
Set con= nothing
```

VB Script Advanced -Database

UFT Script for connecting to sqlserver:

```
Option Explicit
Dim con,rs

Set con=createobject("adodb.connection")
Set rs=createobject("adodb.recordset")

con.open"Driver={SQL Server};server=MySqlServer;uid=MyUserName;pwd=MyPassword;database=pubs"
rs.open "select * from emp",con

Do while not rs.eof
VbWindow("Form1").VbEdit("val1").Set rs.fields("v1")
VbWindow("Form1").VbEdit("val2").Set rs.fields("v2")
VbWindow("Form1").VbButton("ADD").Click
rs.movenext
Loop

'Release objects'Release objects
Set rs= nothing
Set con= nothing
```

VB Script Advanced -Database

UFT Script for connecting to oracle:

```
Option Explicit
Dim con,rs

Set con=createobject("adodb.connection")
Set rs=createobject("adodb.recordset")

con.open "Driver={Microsoft ODBC for Oracle};Server=QTPWorld;
Uid=your_username;Pwd=your_password;"
rs.open "select * from emp",con

Do while not rs.eof
VbWindow("Form1").VbEdit("val1").Set rs.fields("v1")
VbWindow("Form1").VbEdit("val2").Set rs.fields("v2")
VbWindow("Form1").VbButton("ADD").Click
rs.movenext
Loop

'Release objects
Set rs= nothing
Set con= nothing
```

VB Script Advanced -Database

UFT Script for connecting to MySQL:

```
Option Explicit
Dim con,rs

Set con=createobject("adodb.connection")
Set rs=createobject("adodb.recordset")

con.open"Driver={MySQL ODBC 3.51
Driver};Server=localhost;Database=myDB;User=Uname;Password=Pwd;Option=3;"
rs.open "select * from emp",con

Do while not rs.eof
VbWindow("Form1").VbEdit("val1").Set rs.fields("v1")
VbWindow("Form1").VbEdit("val2").Set rs.fields("v2")
VbWindow("Form1").VbButton("ADD").Click
rs.movenext
Loop

'Release objects
Set rs= nothing
Set con= nothing
```

VB Script Advanced -Database

UFTScript for connecting to Excel:

```
Option Explicit
Dim con,rs

Set con=createobject("adodb.connection")
Set rs=createobject("adodb.recordset")

con.open "DRIVER={Microsoft Excel Driver (*.xls)};DBQ=C:\TestStatus.xls;Readonly=True"
rs.open "SELECT count(*) FROM [Status$] where Status = 'Failed' ",con

Msgbox rs(0)

'Release objects
Set rs= nothing
Set con= nothing
```

VB Script Advanced -Database

UFT Script for connecting to Sybase:

```
Option Explicit
Dim con,rs

Set con=createobject("adodb.connection")
Set rs=createobject("adodb.recordset")

' Open a session to the database
con.open"Driver={SYBASE SYSTEM 11};Srvr=myServerAddress;Uid=Uname;Pwd=Pwd;Database=myDataBase;"
rs.open "select * from emp",con

Do while not rs.eof
VbWindow("Form1").VbEdit("val1").Set rs.fields("v1")
VbWindow("Form1").VbEdit("val2").Set rs.fields("v2")
VbWindow("Form1").VbButton("ADD").Click
rs.movenext
Loop

'Release objects
Set rs= nothing
Set con= nothing
```

Thank You ☺