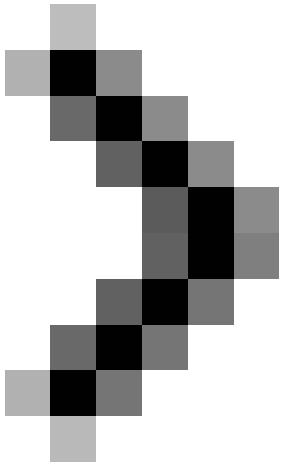
Manage Windows networks using scripts - Part 2: Complete scripts

In the previous section we have learned some basic concepts of scripting techniques such as objects, methods, properties and writing a simple scritp that changes the IP address assigned to adapters. In this second part, we will implement variable definition, error control, use input data and confirm the output data we need to add to get a scr.



Part 1: Basic concepts

Mitch Tulloch

In the previous section we have learned some basic concepts of scripting techniques such as objects, methods, properties and writing a simple scritp that changes the IP address assigned to adapters. Then we used the first four scripts, called ChangeIPAddress.vbs:

```
strComputer = "."
arrIPAddress = Array ("172.16.11.99")
arrSubnetMask = Array ("255.255.255.0")
Set objWMIService = GetObject ("winmgmts:" & strComputer & "rootcimv2")
Set colNetAdapters = objWMIService.ExecQuery ("Select * from Win32_NetworkAdapterConfiguration")
For Each objNetAdapter in colNetAdapters
errEnableStatic = objNetAdapter.EnableStatic (arrIPAddress, arrSubnetMask)
next
```

When running this script on a Windows server, it successfully changed the machine's IP address from .45 to .99. (Check with ipconfig command before and after running the script). Completely good results.

But the script we have built has been quite simple. There are many other important factors missing such as variable definitions, error control, input data validation and output data validation to be added to get a relatively complete script. We will do that in this second part.

Variable definitions

The first thing we need to do to neatly arrange the script is to define the variables to use. VBScript allows implicitly defining variables simply by using it in a statement, but it is better to explicitly declare them at the beginning of the script. Declaring a variable tells VBScript about its existence to allocate storage memory. Why is explicit variable declaration better? For example, in a long script, you often commit one or more wrong typing mistakes. And when typing the wrong name of a variable, your script will not run. If you declare your wall variable at the beginning of the script, any variables declared implicitly in the script (which may be the cause of a typo) will generate a runtime error. The error message may help you locate the mistake and debug your script.

Let VBScript know you explicitly declare all variables in the script, add the following command to the beginning of the script:

```
Option Explicit
```

If you add this command to the top of the ChangeIPAddress.vbs script and run it from the Command Prompt, the result is:

```
C: Documents and SettingsAdministrator.DC-1Desktop>ChangeIPAddress.vbs
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001.All rights reserved.

C: Documents and SettingsAdministrator.DC-1DesktopChangeIPAddress.vbs (2, 1) Microsoft VBScript runtime error: Variable is undefined: 'strComputer'
```

What VBScript (or the Windows Script Host's registered script engine to run VBScript scripts) says here is an error in line 2 of the script:

```
strComputer = "."
```

Why does this error appear? That's because we are assigning values ??to a string variable (strComputer) that has not been declared. So now we need to add declarations for variables used in the script:

```
Option Explicit
Dim objWMIService
Dim objNetAdapter
Dim strComputer
Dim arrIPAddress
Dim arrSubnetMask
Dim colNetAdapters
Dim errEnableStatic

strComputer = "."
arrIPAddress = Array ("172.16.11.93")
arrSubnetMask = Array ("255.255.255.0")
Set objWMIService = GetObject ("winmgmts:" & strComputer & "rootcimv2")
Set colNetAdapters = objWMIService.ExecQuery ("Select * from Win32_NetworkAdapterConfiguration where IPEnabled = TRUE")
For Each objNetAdapter in colNetAdapters
errEnableStatic = objNetAdapter.EnableStatic (arrIPAddress, arrSubnetMask)
next
```

Note that when using Option Explicit (explicit declaration option), you must declare all variables in your script, including objects, strings, arrays, collections, variables for error codes . It sounds very complicated but in fact only need to run some long pages, even if you run the vibrationning time debugging program, you will know how useful it is. Also note that there is no need to declare variables in sequence, just remember to declare for each variable before using it. Usually people put all variable declarations into a separate section on the top of the script as we did above.

Error control

Now that we have eliminated the wrong typing errors when we run the revised script, the script works. But what if it doesn't work? For example, what happens if we change the script a bit to run on a remote machine instead of the local machine in which the remote machine is not on the network? Again the runtime error (ie an error occurs when the script is being executed, whereas a syntax error that VBScript can recognize when compiling the script before running it) will appear and the script will be stopped, Displays an error message similar to the message we saw above. What happens if we write a script to do some operations? In this case, of course, we don't want to have a runtime error causing the script to stop halfway, but at least the script must do all the other operations that have been built. A good example is a script that monitors settings on some computers without changing those settings. In this case you will need to build a script that runs continuously even if one or more of the damaged machines are not running.

The simplest way to control runtime errors is to ignore them when they appear. You can tell VBScript to do this by adding the following command near the beginning of the script, such as immediately after Option Explicit:

```
On Error Resume Next
```

Of course, you also want to do more things in error control. For example, check for the existence of a runtime error condition at some point in the script (as soon as connecting to the WMI service on a remote machine) to determine if an operation is If the script is specified to perform successfully or not. Then, based on the result of the error condition test, you can decide the script of the following actions of the script. For example, if an error

occurs, you may receive a message saying: 'Computer X not found' (the computer cannot be found X) and then the script continues to run. We will learn more about error control in some other articles, but now you just need to add the above command to ignore any runtime errors that appear.

User input data

What to do if we want to describe the new IP address for the machine when we run the script instead of coding it into the script as 172.16.11.99? In this case we need to edit the script to allow the user to input data when running it. Execute by adding parameters when running the script from the command line. For example, type **ChangeIPAddress.vbs 172.16.11.188** will change the IP address of the network adapter to 172.16.11.188 . We can do the following:

```
Option Explicit
On Error Resume Next

Dim objWMIService
Dim objNetAdapter
Dim strComputer
Dim strAddress
Dim arrIPAddress
Dim arrSubnetMask
Dim colNetAdapters
Dim errEnableStatic

If WScript.Arguments.Count = 0 Then
Wscript.Echo "Usage: ChangeIPAddress.vbs new_IP_address"
WScript.Quit
End If

strComputer = "."
strAddress = Wscript.Arguments.Item (0)
arrIPAddress = Array (strAddress)
arrSubnetMask = Array ("255.255.255.0")
Set objWMIService = GetObject ("winmgmts:" & strComputer & "rootcimv2")
Set colNetAdapters = objWMIService.ExecQuery ("Select * from Win32_NetworkAdapterConfiguration where IPEnabled = TRUE")
For Each objNetAdapter in colNetAdapters
errEnableStatic = objNetAdapter.EnableStatic (arrIPAddress, arrSubnetMask)
next
```

Let's analyze each part one by one. First, declare a new variable:

```
Dim strAddress
```

This is a string variable that will contain the parameter (IP address) we describe when we run the script. Next is to add the following lines after the declaration section:

```
If WScript.Arguments.Count = 0 Then
Wscript.Echo "Usage: ChangeIPAddress.vbs new_IP_address"
```

```
WScript.Quit
End If
```

What do these lines do? The Arguments property of the WScript object returns the set of parameters described when running the script. The Count method returns the number of parameters we enter and the purpose of this code section is to check whether we forgot to enter any parameters (number of parameters equal to zero). If so, it will signal (or display) a message telling you how to use the script properly and the script's program is completely stopped.

Finally, the old line:

```
arrIPAddress = Array ("172.16.11.93")
```

where we coded the new array-assigned IP address is now replaced by the following two lines:

```
strAddress = Wscript.Arguments.Item (0)
arrIPAddress = Array (strAddress)
```

The first line retrieves the first element (element 0) of the WScript.Arguments collection and assigns it to the string variable strAddress. The second line then takes this strAddress string variable and assigns it to be the first element of the array arrIPAddress.

Let's see what happens when we run this new script, first without describing the parameter, then running with a parameter:

```
C: Documents and SettingsAdministrator.DC-1Desktop> ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix.:
IP Address. . . . . . . . . . . . : 172.16.11.31
Subnet Mask. . . . . . . . . . . . : 255.255.255.0
Default Gateway.....: 172.16.11.1

C: Documents and SettingsAdministrator.DC-1Desktop>ChangeIPAddress.vbs
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001.All rights reserved.

Usage: ChangeIPAddress.vbs new_IP_address

C: Documents and SettingsAdministrator.DC-1Desktop>ChangeIPAddress.vbs 172.16.11.188
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001.All rights reserved.

C: Documents and SettingsAdministrator.DC-1Desktop> ipconfig

Windows IP Configuration
```

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix.:
IP Address. . . . . : 172.16.11.188
Subnet Mask. . . . . : 255.255.255.0
Default Gateway.....: 172.16.11.1

The program runs pretty well!

Verify the output

If you get bored of typing ipconfig after running the script to check the results, there is another way to help you: use the following lines:

For Each objNetAdapter in colNetAdapters
errEnableStatic = objNetAdapter.EnableStatic (arrIPAddress, arrSubnetMask)
next

The purpose of this code is to change the IP address assigned to the network adapter, using the objNetAdapter.EnableStatic method. But you should note that an err- variable is needed (here errEnableStatic) used to store the error code returned when running the method. The error code list can be returned from the EnableStatic method of the Win32_NetworkAdapterConfiguration class, which can be found on MSDN. And from this list we can see that the result returned by 0 means that the script operation has been successful (for example, the adapter IP address has been successfully changed). The easiest way to check is to add the line below the end of the script:

Wscript.Echo errEnableStatic

Run the script again:

C: Documents and SettingsAdministrator.DC-1Desktop>**ChangeIPAddress.vbs 172.16.11.237**
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001.All rights reserved.

0

C: Documents and SettingsAdministrator.DC-1Desktop> **ipconfig**

Windows IP Configuration

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix.:
IP Address. . . . . : 172.16.11.237
Subnet Mask. . . . . : 255.255.255.0
Default Gateway.....: 172.16.11.1

Surely that is enough. The result returned by 0 indicates that the IP address has been successfully changed. A better method will display a message by changing the signaling command again with the following command:

```
If errEnableStatic = 0 Then
Wscript.Echo "Adapter's IP address has been successfully changed to" & strAddress
Else
Wscript.Echo "Changing the adapter's address was not successful. Error code" & errEnableStatic
End If
```

Add the following commands to the end of the script and run twice, one with the correct IP address and one with the arbitrary IP address:

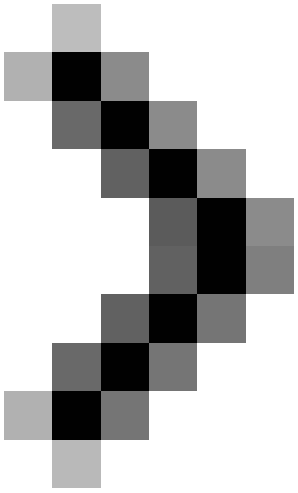
```
C: Documents and SettingsAdministrator.DC-1Desktop>ChangeIPAddress.vbs 172.16.11.173
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001.All rights reserved.

??a ch? IP Adapter ?ã ???c thay ??i ?? 172.16.11.173

C: Documents and SettingsAdministrator.DC-1Desktop>ChangeIPAddress.vbs 172.16.11.1492567
Microsoft (R) Windows Script Host Version 5.6
Copyright (C) Microsoft Corporation 1996-2001.All rights reserved.

?ang thay ??i adapter c?a ??a ch? không thành công.Error code 70

C: Documents and SettingsAdministrator.DC-1Desktop>
```

Conclude

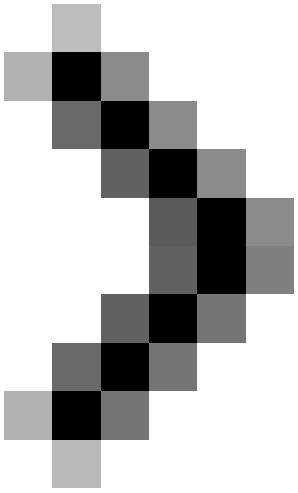
The last thing we need to do is add some comments to the script to explain the script. This is always a good idea because maybe a year later to read it again and want to change something in the script, you will easily find what you need. This is our final complete script to change the network adapter IP address:

```
==========================
'NAME: ChangeIPAddress.vbs
'
'AUTHOR: Mitch Tulloch
'DATE: October 2006
'
ARGUMENTS:
'first. new_IP_address
'========================== -

Option Explicit
On Error Resume Next

Dim objWMIService
Dim objNetAdapter
Dim strComputer 'Can specify IP address or hostname or FQDN
Dim strAddress' Contains the new IP address
Dim arrIPAddress
Dim arrSubnetMask
Dim colNetAdapters
```

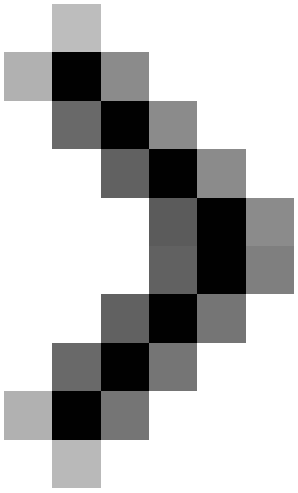
```
Dim errEnableStatic

'L?u cho thi?u ??i s?

If WScript.Arguments.Count = 0 Then
Wscript.Echo "Usage: ChangeIPAddress.vbs new_IP_address"
WScript.Quit
End If

strComputer = "."
strAddress = Wscript.Arguments.Item (0)
arrIPAddress = Array (strAddress)
arrSubnetMask = Array ("255.255.255.0")
Set objWMIService = GetObject ("winmgmts:" & strComputer & "rootcimv2")
Set colNetAdapters = objWMIService.ExecQuery ("Select * from Win32_NetworkAdapterConfiguration where
IPEnabled = TRUE")
For Each objNetAdapter in colNetAdapters
errEnableStatic = objNetAdapter.EnableStatic (arrIPAddress, arrSubnetMask)
next

'Display k?t qu? ho?c mã l?i

If errEnableStatic = 0 Then
Wscript.Echo "Adapter's IP address has been successfully changed to" & strAddress
Else
Wscript.Echo "Changing the adapter's address was not successful. Error code" & errEnableStatic
End If
```



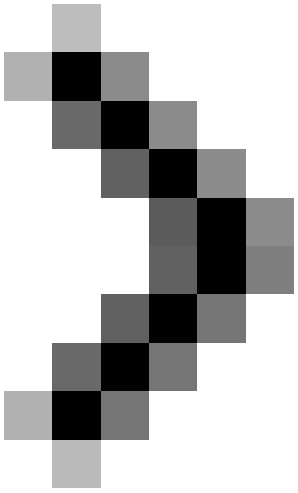
Part 3: Understanding WMI



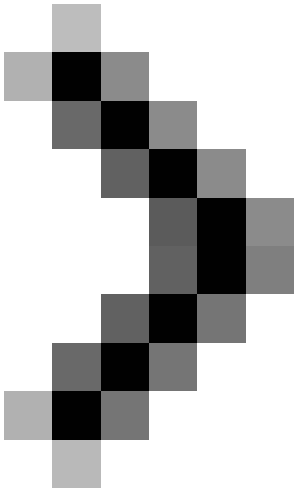
Part 4: Use Win32_NetworkAdapterConfiguration



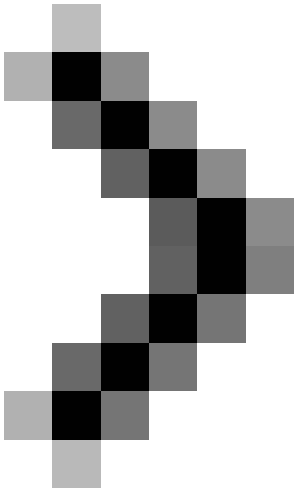
Part 5: Overcoming challenges



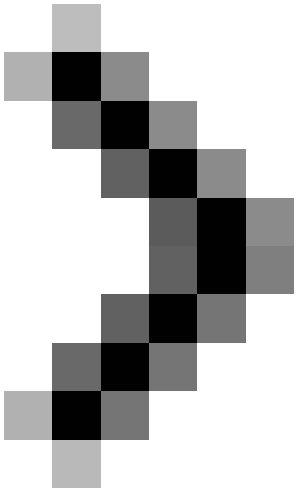
Part 6: The first steps for remote scripting



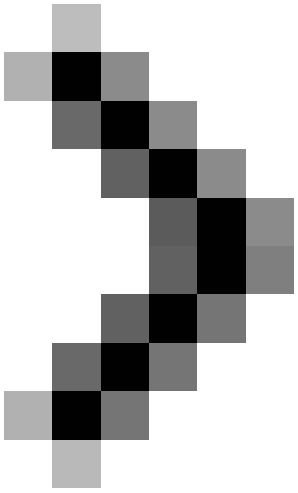
Part 7: Troubleshooting errors



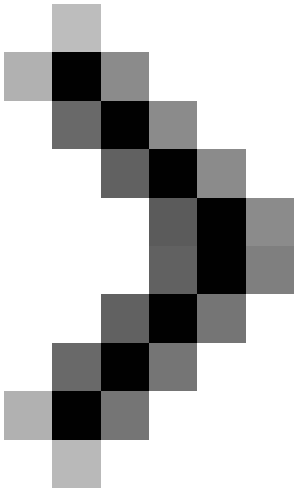
Part 8: Remote script error handling with Network Monitor 3.0



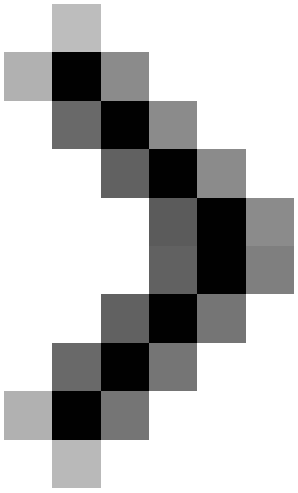
Part 9: Understanding remote control scenarios



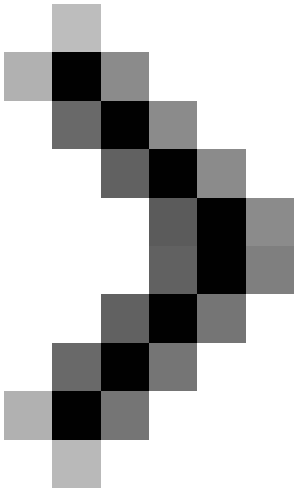
Part 10: Tricks of remote control scenarios



Part 11: Other script tricks



Part 12: Properties of the WMI class



Part 13: The script returns all values

You finished reading the article "**Manage Windows networks using scripts - Part 2: Complete scripts**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.