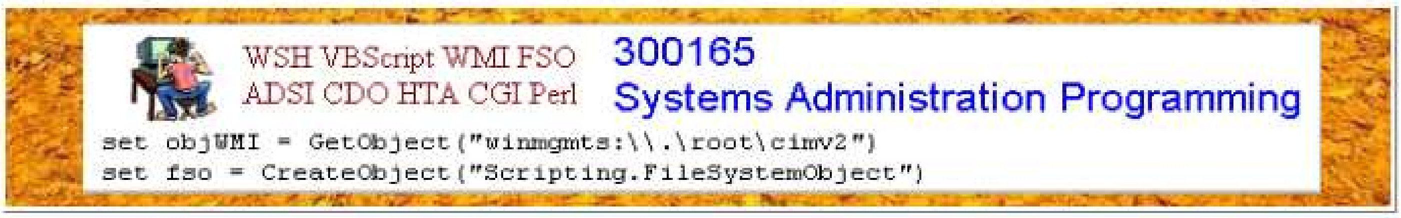


```
set objWMI = GetObject("winmgmts:\\.\root\cimv2")
set fso = CreateObject("Scripting.FileSystemObject")
```

Lecture 2
VBScript Basics

Print page

Did you enjoy the first lecture? Trust you did - at least you enjoyed your coffee.

VBScript is powerful and mostly used by computing professionals. While you can write some fairly simple scripts, you can also write sophisticated scripting code. In this lecture, we will learn the basics of VBScript - working with variables, calling functions, controlling structures etc.

Key words

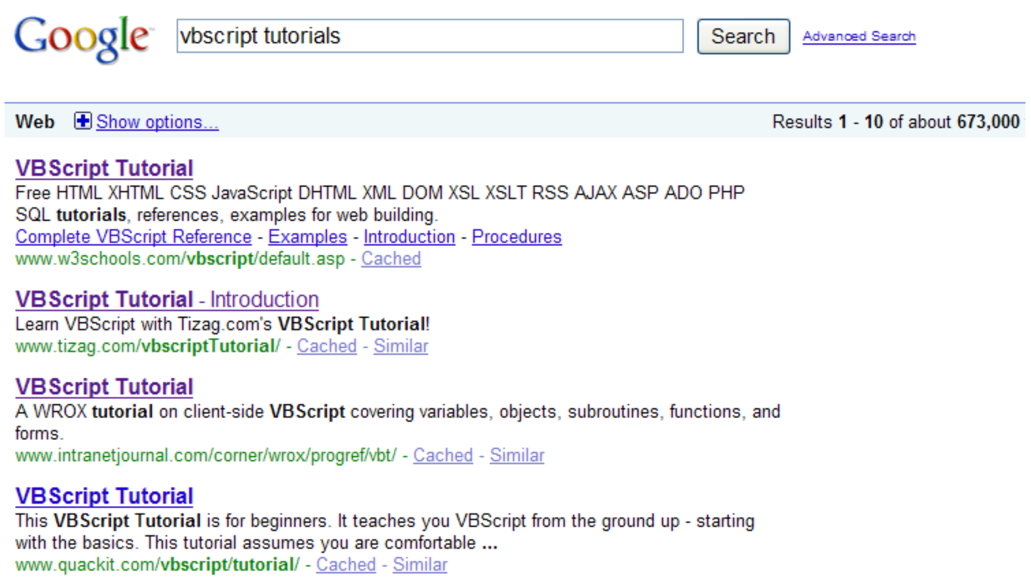
Variable, Function, Procedure/Subroutine, Control Structure

Reference to textbook chapters

Just in case the lengthy code in Chapter 3 scares you off, we skip Chapter 3 & 4 temporarily and will return to these chapters in next week. This lecture will cover Chapter 5-10. You are strongly encouraged to read through these chapters to acquire the knowledge that may not be mentioned in the lecture notes. (Don Jones, VBScript, WMI and ADSI unleashed : using VBSscript, WMI, and ADSI to automate Windows administration eBook: Part II. VBScript Tutorial: Chapter 5-10).

Part II: VBScript Tutorial
- **Chapter 5. Functions, Objects, Variables, and More**
  - **Variables Defined**
    - Sample Script
    - Declaring Variables
    - Understanding Data Types
    - Assigning Data to Variables
    - Data Coercion
  - **Functions Defined**
    - Input Parameters
    - Output Values
    - Intrinsic Versus Custom Functions
  - **Statements and Subroutines Defined**
    - Functions, Without the Output
  - **Objects Defined**

Another common practice is referencing to online VBScript tutorials if necessary. This is very handy, and there are heaps of tutorial lessons over the Web - why don't we share the experience?


vbscript tutorials

Search Advanced Search

Web Show options... Results **1** - **10** of about **673,000**

VBScript Tutorial
Free HTML XHTML CSS JavaScript DHTML XML DOM XSL XSLT RSS AJAX ASP ADO PHP SQL **tutorials**, references, examples for web building.
Complete VBScript Reference - Examples - Introduction - Procedures
www.w3schools.com/**vbscript**/default.asp - Cached

VBScript Tutorial - Introduction
Learn VBScript with Tizag.com's **VBScript Tutorial**!
www.tizag.com/**vbscriptTutorial**/ - Cached - Similar

VBScript Tutorial
A WROX **tutorial** on client-side **VBScript** covering variables, objects, subroutines, functions, and forms.
www.intranetjournal.com/corner/wrox/progref/vbt/ - Cached - Similar

VBScript Tutorial
This **VBScript Tutorial** is for beginners. It teaches you VBScript from the ground up - starting with the basics. This tutorial assumes you are comfortable **...**
www.quackit.com/**vbscript**/**tutorial**/ - Cached - Similar

In fact, the lecture notes cannot cover everything in details. I just pick the most important issues and explain them with examples. Since you have done at least Programming Fundamentals, you know C++. What you need to know are the differences of the language basics between C++ and VBScript.

Variables

Variables are used to store data. Note that data can be anything from characters, strings, integers to encapsulated objects. Different from the general purpose languages, such as C, C++, and Java, VBScript is not a strongly typed language, which means that you don't have to tell the interpreter what type of data you are going to store in the variable you define. You can simply use `Dim` to declare any variables:

```
Dim myInteger
Dim myDouble
Dim myObject
```

The rules for variable names are similar to other languages but they are not case sensitive (see Textbook page 80).

Strange enough, you even don't have to declare a variable.

Download the script, variables1.vbs, and run it.

Also

Download the script, variables2.vbs, and run it.

However, good scripts are planned. It's better to declare the variables that you are going to use. VBScript provides a simple statement 'Option Explicit' to tighten up your scripts, which says that: if a variable is not declared, then you cannot use it. (see Textbook page 81).

A variable can also store an array of elements. See the following example. Please pay attention to the declaration of an array variable and assignment of elements to the array.

```
Dim aList(3)

aList(0) = "Monday"
aList(1) = "Tuesday"
aList(2) = "Wednesday"
aList(3) = "Thursday"
```

In this code,

```
Dim aList(3)
```

defines an array with Four members. If you do not believe that you can store four members in it,

Download the script, array1.vbs, and execute it.

Alternatively, we do the same thing as follows:

```
Dim aList

aList = Array("Monday","Tuesday","Wednesday","Thursday")
```

Practice it with array2.vbs.

You can even resize an array after you have defined it.

Download the script, array3.vbs, and execute it.

Functions

As for all other programming languages, VBScript allows you to create functions to modularize your code. Functions are isolated bits of code that are called to perform specific or repetitive operations. The following piece of code shows you how to create a function:

```
function sum(a,b)
    sum = a + b
end function
```

Note that we don't have a return statement. We use the function name to return the value of the function. So convenient, isn't it? Use the following code for practice.

Download the script, functions.vbs, and execute it.

Procedures/Subroutines

Procedures/Subroutines are functions without return values. Therefore it is like a void function. However, the syntax used to define a subroutine is different from that used to define a function (sub vs. function), and the way calling a subroutine is also different from calling functions.

```
Dim sum

a = 10
b = 20

calculate_sum a, b

MsgBox sum

sub calculate_sum( a, b)
    sum = a + b
end sub
```

Note that all variables used here are global, except that defined inside functions or subroutines.

Display messages

There are two built-in functions available for us to output messages: WScript.echo and MsgBox. The former can work in both a window-based scripting host and a command-line scripting host. The latter can only work in the Windows environment but is more versatile.

Download the script, output1.vbs, and execute it.

There is only one line in the code:

```
MsgBox "Are you sure?",4,"Check"
```

The function MsgBox takes three arguments. The first one is the message you want displayed. The third one gives the title of the message box. The most tricky one, the second one, specifies the type of the message box. For instance, "4" indicates that the box contains 'Yes' and 'No' buttons. Change it to 1 to see what will happen. See details from the following webpage to learn about the MsgBox function:

http://msdn.microsoft.com/en-us/library/sfw6660x.aspx

Download the script, output2.vbs, and execute it.

This code shows you how to get a return value from MsgBox function.

Asking for input

To get input from users, you can simply use the built-in function InputBox. The following code illustrates its usage:

```
Dim vInput

vInput = InputBox("Enter a server name", "Server")
MsgBox "You entered " & vInput
```

The function returns an empty string if the cancel button is pressed. It returns -1 if esc key is pressed.

Controlling the flow of execution

As a powerful programming language, VBSript provides enough controlling structures for you. The following summarize the syntax of these structures. Most of them have been used in my example code.

Conditional execution:

```
If  condition_expression Then
    statement1
ElseIf
```

```
    statement2
else
    statement3
End if
```

Selection execution:

```
Select Case caseNumber
    Case 1
        statement1
    Case 2
        statement1

    ...

    Case Else
        statementn
End Select
```

Loops:

```
Do While condition
    statement
Loop
```

or

```
Do
    statement
Loop While condition
```

or

```
Do Until condition
    statement
Loop
```

or

```
Do
    statement
Loop Until
```

or

```
For counter = start To end
    statement
Next
```

or

```
For counter = start To end Step number
    statement
Next
```

or

```
For Each element In collection
    statement
Next
```

If you are a good C++ or Java programmer, it should be easy for you to get used to these structures. Otherwise, you need extra practice.

That's all for today. You may have spent more than two hours on this lecture. Thanks for your attention.