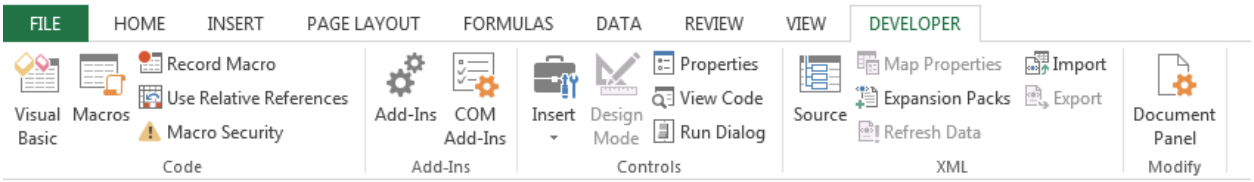
Introduction to VBA for Excel-Tutorial 3

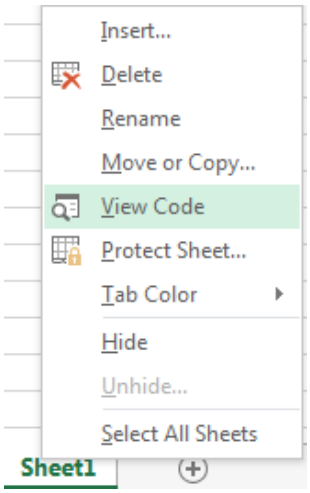
In this tutorial, we will put all the pieces together. You are about to write your first procedure to compute the sum of three numbers. You will also learn how to execute the program, and how to debug the code.

As we discussed before, the two ways of coding VBA are either by writing a sub or a function. Of course, there are other objects available in VBA, but we won't be using these in this class. During this course, I hope to inspire you to learn more VBA functionality and become a more capable programmer. In this tutorial will make use of sub first.

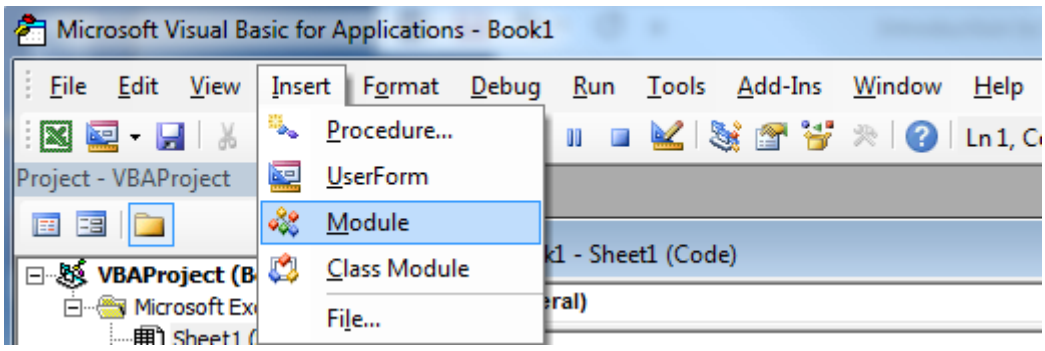
First, let us open the VB Editor. The easiest way is Alt+F11, or you can click on the Developer tab and then click Visual Basic.



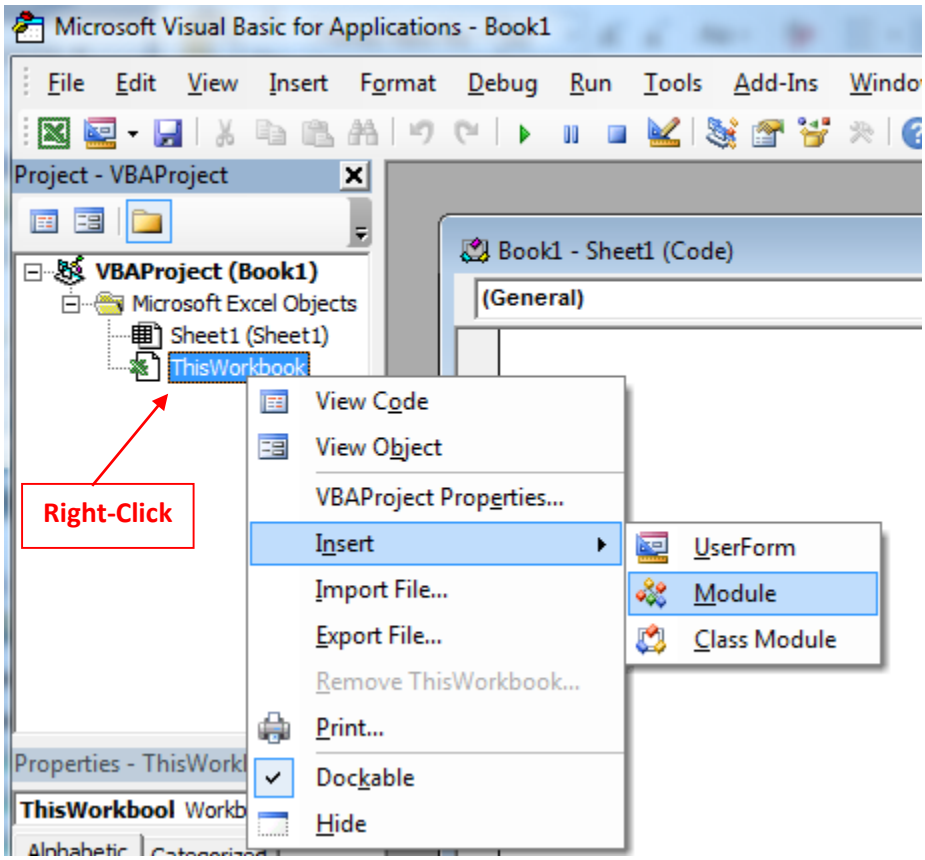
OR right click any of the sheet tabs and select View Code.



Second, let us add a new module (also called code module*), by clicking on Insert/Module.



OR right click on Microsoft Excel Object on the Project Tree in the Project Explorer window



* A module containing public code that can be shared among all modules in a project. A code module is referred to as a standard module in later versions of Visual Basic.

The next figure is a screenshot of the code with comments, illustrating different input/output (I/O) methods. I encourage you to practice using all of them. Please, input the following code to the module just created.

Notes:

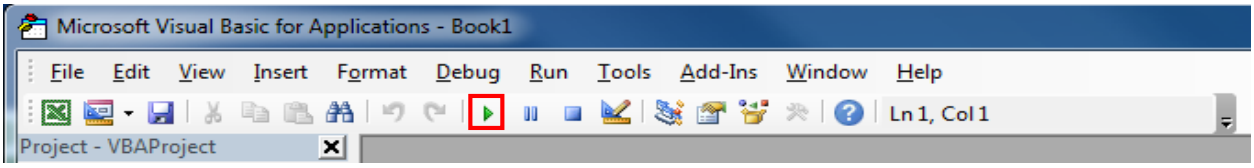
1- Option Explicit (top line of the code), means you are required to define all used variables. Do you remember how to activate/deactivate this feature? ***Tools/Options/Editor tab/ check or uncheck Required Variable Declaration***.

2- Multiple code lines can be written on the same line, as long as they are separated by a colon (:). This is illustrated on the Dim line.

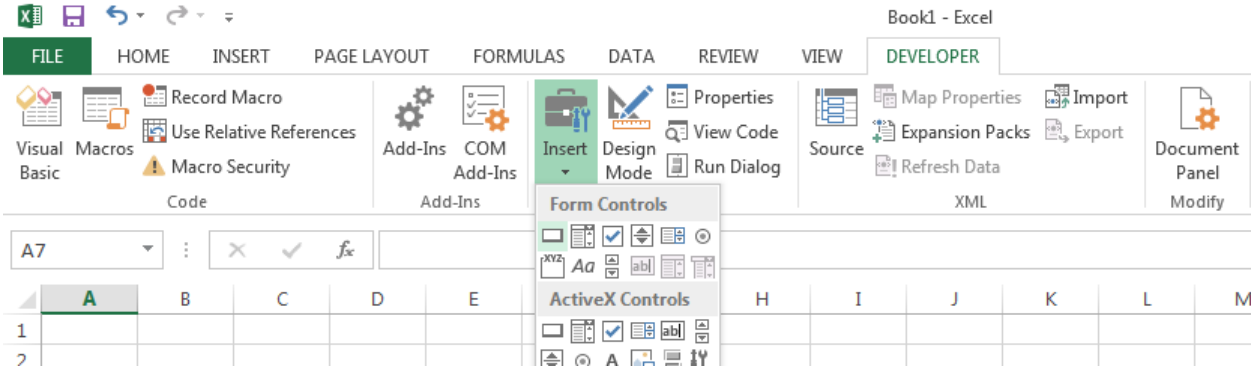
3- Green font indicates comment lines, a comment line must start with a single quote or apostrophe (‘). Comments are not complied when the code is executed. Consider comments as help for you when you are debugging the code. Also they are a good reminder when you revisit these codes in a couple of years. Take my word on this, write as many legible and meaningful comments as you can, it will become very helpful.

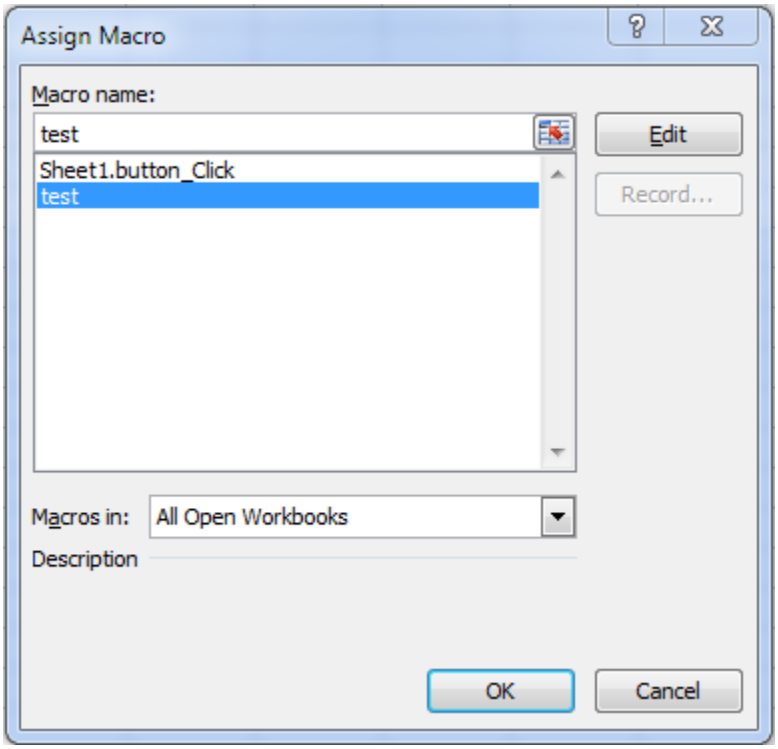
```
Option Explicit

Dim a As Single: Dim b As Single: Dim c As Single: Dim d As Single

Sub test()

'use different input methods to input multiple numbers
'find the sum of all entered numbers and then output the results
'to the active worksheet

'input
'method 1 for input by accessing the cell on worksheet, two steps
Sheets("Sheet1").Select
Range("B2").Select
a = ActiveCell.Value
'method 2 for input by accessing the cell on worksheet, one step
b = Range("B3").Value
'method 3 for input using Msgbox
c = Val(InputBox("Please enter number0-10", "Input", "0-10"))
'since there will be no track of this input number
'we will outpu this to the worksheet to keep record
Cells(3, 2) = c 'this method explained in tutorial #2

'performing calculation
d = a + b + c

'output the sum
Range("B5").Value = d 'note different way of output

'yet another way to display the results
MsgBox "sum = " & d, , "Results"

End Sub
```

Now that the code is inserted in the module, it is time to run the macro (or the code). Simply press the run button  on the standard tool bar in VBE or press **F5**. The code will be executed.



These methods described to run the code are suitable for the development phase. However, the user should not need to access VBE every time they want to run the code. We should give the user accessibility to the code from the Excel UI (user interface). To do so, add a Control Button by clicking on Developer tab/Controls /Insert/Form Controls, then select Button. Place the button anywhere you like on the worksheet, Excel will prompt you with the Assign Macro window, select the Macro name you wish to run when the user clicks the button.





Once you select the macro and press ok, Excel will take you back to the button where you can change the name. Name this button "Run" and press enter to get out of the editing mode. The final product should look like:

Every time you click Run, the code will execute.

So, what is the difference between a function and sub procedure?

The procedure is a set of commands that the user or another procedure can execute. The user, in this case, needs a means of execution such as a Control Button or keyboard shortcut. In contrast, the function returns a single value (or array) based on input arguments passed in by the user. For example, if you would like to use the built-in square-root function in Excel to calculate $\sqrt{4}$, then you go to a cell and type:

=sqrt(4)

and press Enter,. Note that the results show in the same cell you selected and entered the formula in. In this example, ***sqrt*** is the function name and ***4*** is the argument of the function. The argument of the function is the user's input. So, can the function have only one argument? No, I'm sure you can recall other built-in functions that require two or more arguments. For example, the Round function

=Round(number,num_digits)

again, ***round*** is the function name and ***number*** and ***num_digits*** are the required two arguments. The question, why bother to create an additional function beside the some 340 functions already exist in Excel? This question, to me, is simply answered philosophically and practically: to make our lives worth living!

Before we dig in to the first example, it is worth noting that all programming skills and syntaxes you have learned thus far are valid and will be used in the same manner in creating a function as it was used in creating a procedure.

Example:

Let us the write a function to calculate the sum of three numbers.

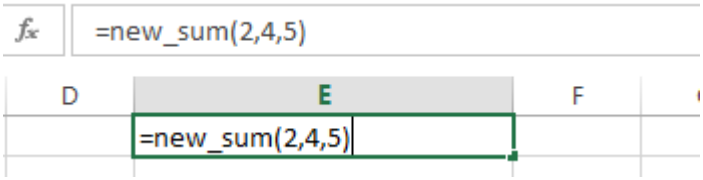
```
Function new_sum(a As Double, b As Double, c As Double)
'note the input is declared in the argument of the function
'working variables declaration
Dim sum As Double
'logic
sum = a + b + c
'output
new_sum = sum
End Function
```

Notes:

1- The inputs or the arguments of the function are included in the bracket and each variable is declared as its appropriate datatype. The declaration is optional.

2- The most important difference between Function and Sub is the output. Note that the last line is equating the function name (i.e. new_sum) to the calculated results. Without this line, the answer would be always equal to zero,

Once the function is written and debugged, you can use this function through the Excel interface such as:



where, a= 2, b = 4 and c = 5. Once you type in this formula and press Enter, Excel will output 11 as the result.