

Chapter 4 Types of Statements

A **statement** is a complete instruction in Visual Basic programs. It may contain keywords, operators, variables, literal values, constants and expressions.

Statements could be categorized as:

- **Declaration statements** – these are the statements where you name a variable, constant, or procedure, and can also specify a data type.
- **Executable statements** – these are the statements, which initiate actions. These statements can call a method or function, loop or branch through blocks of code or assign values or expression to a variable or constant.

❖ Declaration Statements

When you declare a programming element, you can also define its data type, access level, and scope. The programming elements you may declare include variables, constants, enumerations, classes, structures, modules, interfaces, procedures, procedure parameters, functions return etc.

Following are the declaration statements in VB.Net –

Sr.No	Statements and Description	Example
1	Dim Statement Declares and allocates storage space for one or more variables.	<pre>Dim x As Integer = 22</pre>
2	Const Statement Declares and defines one or more constants.	<pre>Const no1 As Long = 10</pre>
3	Enum Statement Declares an enumeration and defines the values of its members.	<pre>Enum CoffeeMugSize Large Medium End Enum</pre>
4	Class Statement Declares the name of a class and introduces the definition of the variables, properties, events, and procedures that the class comprises.	<pre>Class Box Public lng As Double Public hght As Double End Class</pre>
5	Sub Statement Declares the name, parameters, and code that define a Sub procedure.	<pre>Sub muSub(By Val s As String) Return End Sub</pre>
6	Structure Statement Declares the name of a structure and introduces the definition of the	<pre>Structure Box Public length As Double</pre>

	variables, properties, events, and procedures that the structure comprises.	Public breadth As Double End Structure
7	<p>Module Statement</p> <p>Declares the name of a module and introduces the definition of the variables, properties, events, and procedures that the module comprises.</p>	<pre>Public Module myModule Sub Main() Dim user As String = InputBox("What is your name?") MsgBox("User name is" & user) End Sub End Module</pre>
8	<p>Function Statement</p> <p>Declares the name, parameters, and code that define a Function procedure.</p>	<pre>Function myFunction (ByVal n As Integer) As Double Return 5.87 * n End Function</pre>

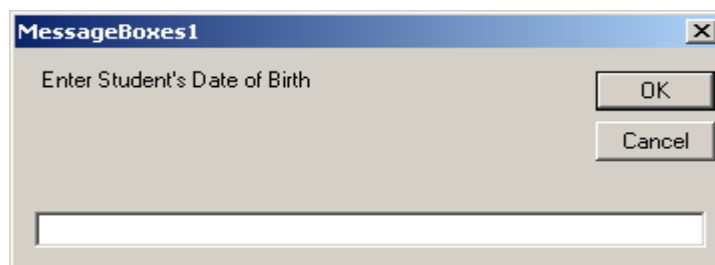
❖ Executable Statements

An executable statement performs an action. Statements calling a procedure, decision statements, looping through several statements, or evaluating an expression are executable statements. An assignment statement is a special case of an executable statement.

➤ Input Operations

In VB2010, there are no explicit instructions or statement for input operation. Vb2010 used some controls to execute the input operation.

1- InputBox: An input box is a specially designed dialog box that allows the programmer to request a value (some time huge data values) from the user and use that value as necessary. When calling the **InputBox()** function, if you pass only the first two arguments, the input box would display the title for the InputBox in the title bar, a prompt message to indicate the requested value, a text box for the user, and two buttons: OK and Cancel.

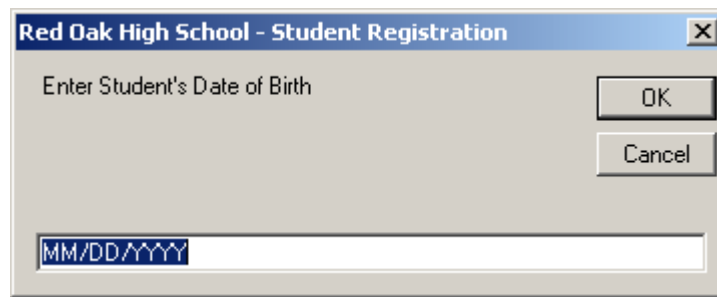


After using the input box, the user can change his or her mind and press **Esc** or **click Cancel**. If the user provided a value and want to acknowledge it, he or she can click **OK** or press **Enter**. This would transfer the contents of the text box to the application that displayed the input box.

To assist the user of type a value that you are expecting, you can give an example by using the third argument as a string. When passing it, you can provide a sample value that the user would follow.

Example:

```
InputBox("Enter Student's Date of Birth", "Red Oak High School - Student Registration", _
"MM/DD/YYYY")
```



The last two arguments, *XPos* and *YPos*, allow you to specify the default position of the input box when it comes up the first time.

Example:

```
InputBox("Enter Student's Date of Birth", "Red Oak High School - Student Registration", _
"MM/DD/YYYY", 100, 100)
```

2- TextBox: Is one of the common controls that use to accept data form user. It's good for small amount of input data. The Text Property can be used for both setting and retrieving text. *See CH3.P21 for more information.*

Example:

```
Dim mysalary As Integer
mysalary = Cint (TextBox1.Text)
```

Where,

mysalary: Integer variable

Cint : A function used to convert text value to numeric value to be compatible with the variable “mysalary” which is declared as integer type.

TextBox1: Is a control that used to accept data.

Text : Is the property that used for this purpose.

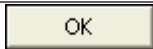

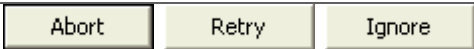
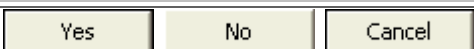


➤ Output Operations

- 1- Label:** It is generally used to display some informative text on the GUI which is not changed during runtime by assigning the value to the *Text* property.
- 2- Message Boxes:** A message box is a special dialog box used to display a piece of information to the user. To support message boxes, the Visual Basic language provides a *MsgBox* function, also the .NET Framework, provide the Show() method of the *MessageBox* class.
 - **MsgBox:** It takes one or more arguments and the results of the function can be assigned to a variable. The syntax for MsgBox..

MsgBox (Prompt, Buttons, Title)

Where, Prompt: Is the text displayed in the message box.
 Button: Is a number (0, 1, 2, 3, 4, 5) that specifies the buttons.
 Title : Is the text displayed in the message box title bar.

If you create a simple message box by providing only the message, it would appear with only one button labeled OK. If you want the user to be able to make a decision and communicate it to you, provide a second argument. The second argument must be based on the **MsgBoxStyle** enumeration. When it comes to buttons, some members of this enumeration are:

To Display	MsgBoxStyle	Integral Value
	OKOnly	0
	OKCancel	1
	AbortRetryIgnore	2
	YesNoCancel	3
	YesNo	4
	RetryCancel	5





To use any of these buttons, call the **MessageBoxStyle** enumeration and access the desired combination. Here is an example:

MsgBox("Now move to the next step", MsgBoxStyle.OkCancel, "Lesson Objective")

OR

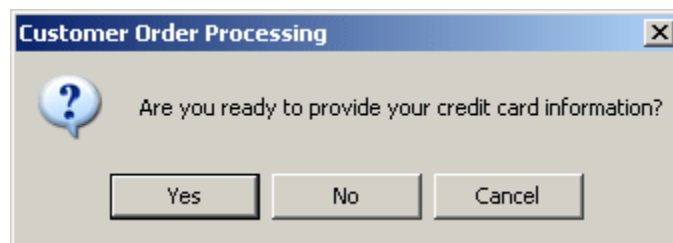
MsgBox("Now move to the next step", 1, "Lesson Objective")

To enhance the appearance of a message box, you can display an icon on it. To support icons, the **MsgBoxStyle** enumeration provides the following additional members . To applying one of these buttons, combine its style with that of the button, using the **OR** operator.

To Display	MsgBoxStyle	Integral Value
	Critical	16
	Question	32
	Exclamation	48
	Information	64

Example:

MsgBox("Are you ready to provide your credit card information?", MsgBoxStyle.YesNoCancel Or **MsgBoxStyle.Question**, "Customer Order Processing")

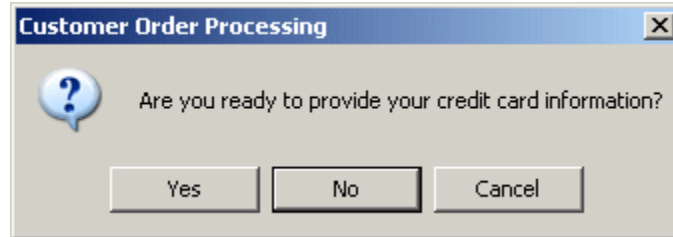


When a message box is configured to display more than one button, the operating system is set to decide which button is the default. The default button has a thick border that sets it apart from the other button(s). If the user presses Enter, the message box would behave as if the user had clicked the default button. If the message box has more than one button, you can decide what button would be the default. To support the default button, the **MsgBoxStyle** enumeration provides the following additional options:

MsgBoxStyle	Integral Value	If the message box contains more than one button, the default button would be
DefaultButton1	0	the first
DefaultButton2	256	the second
DefaultButton3	512	the third

Example:

MsgBox("Are you ready to provide your credit card information?", MsgBoxStyle.YesNoCancel Or MsgBoxStyle.Question Or **MsgBoxStyle.DefaultButton2**, "Customer Order Processing")



The value returned by a message box corresponds to a button the user would have clicked (on the message box). The return value of the MsgBox() function is based on the **MsgBoxResult** enumeration. The buttons and the returned values are as follows:

If the User Clicks	Button Caption	Integral Value
	OK	1
	Cancel	2
	Abort	3
	Retry	4
	Ignore	5
	Yes	6
	No	7

- **MessageBox.Show():** It takes one or more arguments. It has the following format:

MessageBox.Show(Text, Caption, Buttons, Icon)

Example:

```
MessageBox.Show("Welcome to Microsoft VB", "Program 1",0,48)
```

Note: we can use ‘&’ operator to join a message and a variable value. The ‘&’ operator can also use to join strings together.

H.W.: Design and write a program to read your name by using InputBox and to display your name as the following example.

Your Name Is Baida’a

H.W.: Design and write a program to display the result of adding three numbers. Use 3 (TextBox to read three numbers), 2 (Buttons one for Exit and the other for Add), 1 (Label to display the result). Change back color and for color for all controls and use any picture as a back color for form.