How to Set Object Properties During Runtime

Open the Code Editor window

The Code Editor provides the code template to help you follow the rules of the Visual Basic language. To open the Code Editor window there are several ways:

1- Double-click on any tool from the Design window that you want to write the code inside it.
2- Right-click on the form and then click View Code from the menu that will appear.
3- Right-click on the *form file* in the Solution Explorer and then click View Code from the menu that will appear.

The Code Editor window opens in the IDE, as shown in Figure 1-7.

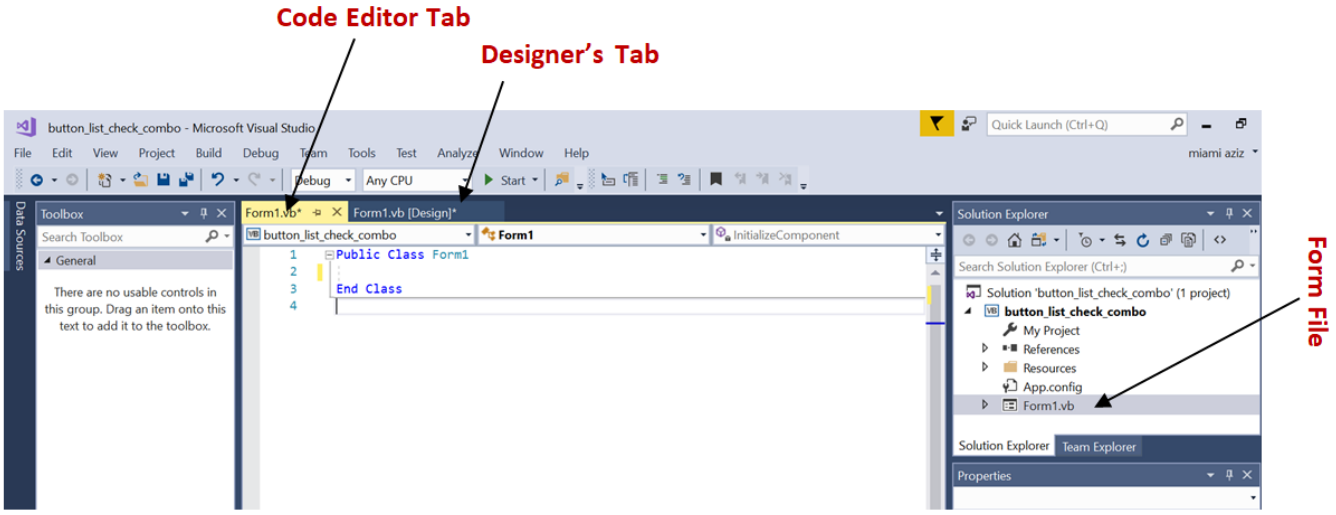

Figure 1-7 Code Editor window opened in the IDE

Suppose you add a button tool to the form. Press double click on this button to go to the Code Editor window; specifically to button's code template. The first line in the button's code template is called the **header**, and the last line is called the **footer**. The header begins with the keywords *Private Sub*. A **keyword** is a word that has a special meaning in a programming language, and it appears in a different color from the rest of the code. The *Sub* keyword is an abbreviation of the term **sub procedure,** which is a block of code that

performs a specific task. Following the *Sub* keyword is the name of the **object (tool), an underscore, the name of the event**. This clause indicates that the procedure handles (or is associated with) the button control's Click event. It tells the computer to process the procedure only when the button control is clicked. The code template ends with the procedure footer, which contains the keywords *End Sub*. You enter your Visual Basic instructions at the location of the insertion point, which appears between the Private Sub and End Sub clauses in Figure 1-8.

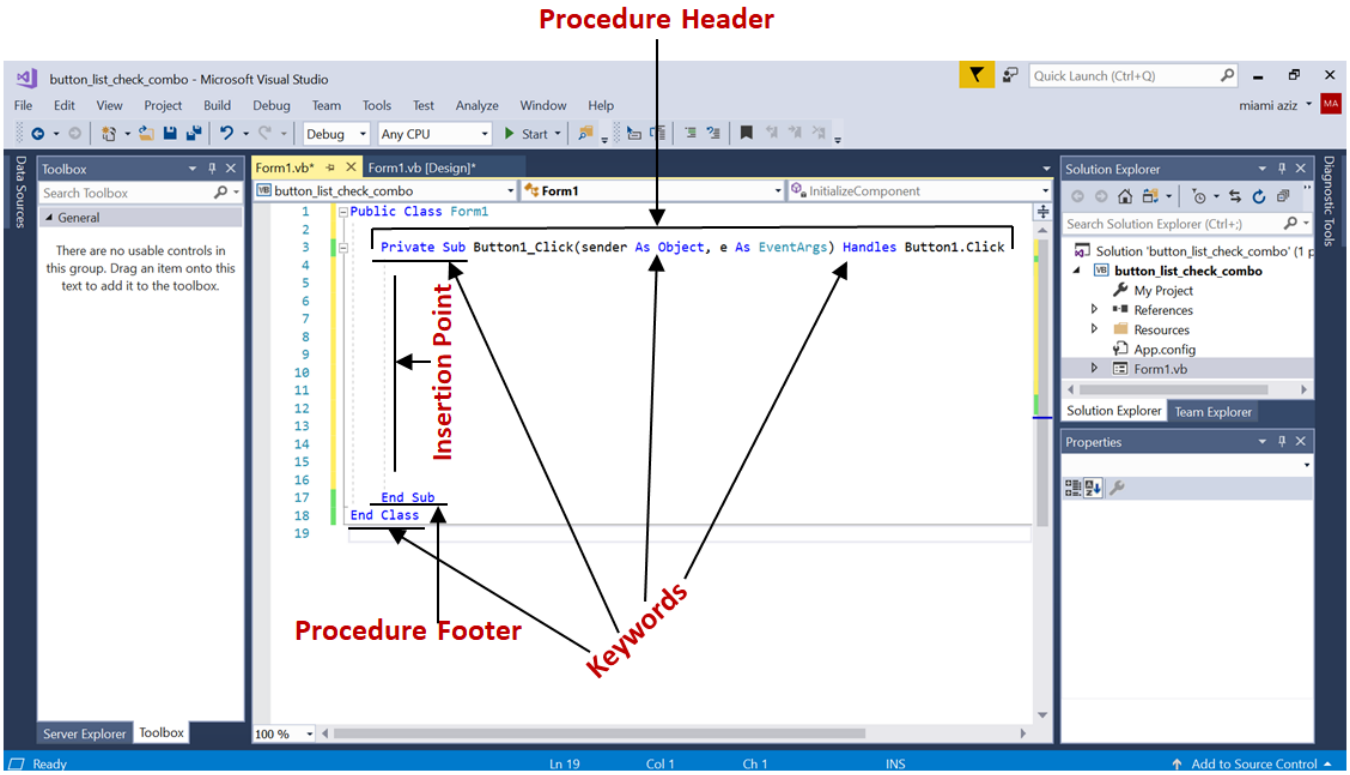

Figure 1-8 The procedure associated with the button control's click event

It is a good idea to test a procedure after you have coded it so you will know where to look if an error occurs. You can test this button's procedure by starting the application and then clicking the button. When the button is clicked, the computer will process all the instruction statements contained in the button.

To test the button procedure:

- ✓ Save the solution and then click the Start button on the Standard toolbar. The user interface appears on the screen.
- ✓ Click the button to check the execution of the instructions.

Assignment Statements

You learned how to use the Properties window to set an object's properties during **design time**, which is when you are building the interface. You can also set an object's properties during **run time**, which occurs while the application is running; you do this by using an assignment statement. An **assignment statement** is one of many different types of Visual Basic instructions. Its purpose is to assign a value to something, such as to the property of a tool. The syntax of an assignment statement is shown in Figure 1-9 along with examples of using the syntax. In the syntax, *tool* and *property* are the programmatic name of the tool and its property to which you want the value of the *expression* assigned. The expression can be a **keyword**, a **number**, or a **string literal** (*string literal* is defined as zero or more characters enclosed in quotation marks). The expression can also be a calculation; you will learn how to assign calculations in the next lectures. You use the ***dot*** member access operator to separate the tool name from the property name. The dot operator indicates that the *property* is a member of the *tool*. You use an equal sign between the *tool.property* and the *expression*. The equal sign in an assignment statement is called the **assignment operator**. The **assignment operator** assigns the value of the expression that appears on the right side of the assignment operator to the object's property that appear on the left side of the assignment operator

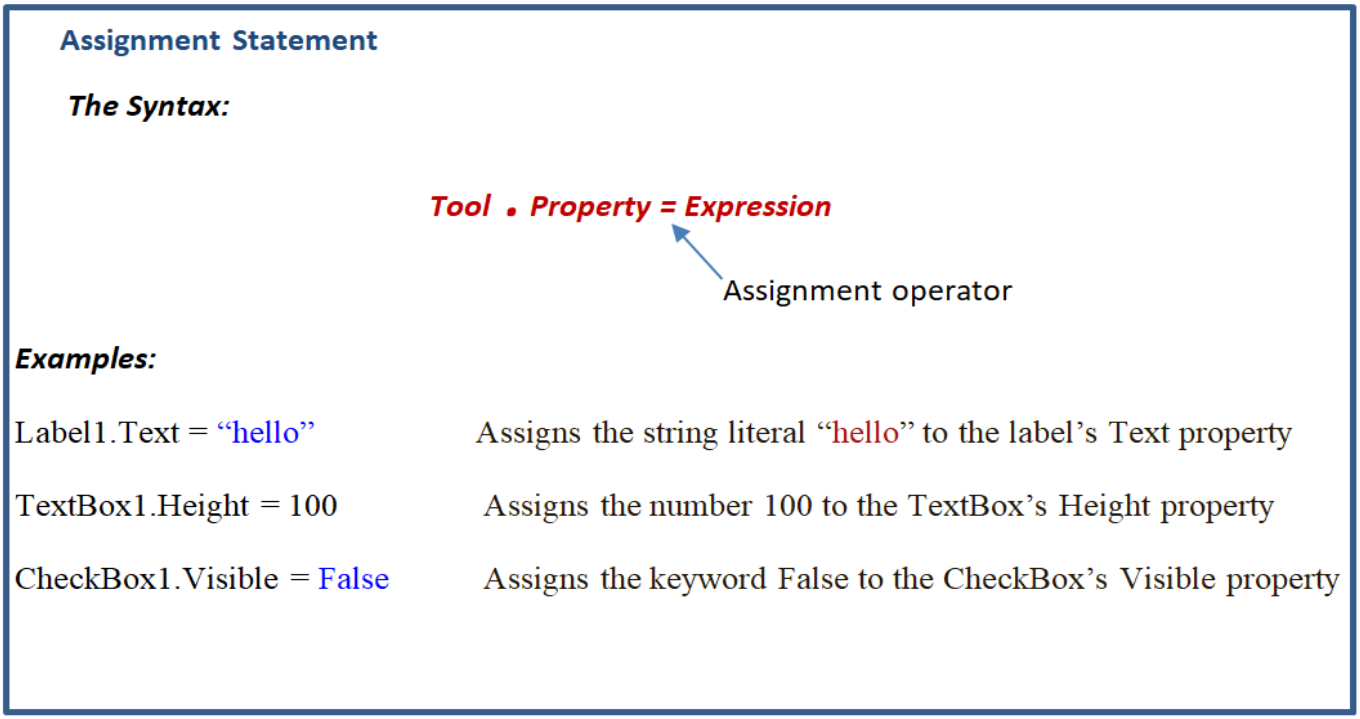

Figure 1-7 Syntax and examples of assigning a value to a property during run time

Now, Label and Button have been added to the form. The label represents the tool which we want to change its properties. The command button is the place we will use to type most of the code to change the properties of the label. Table 1-7 shows some examples of changing properties of a Label:

The Assignment Statement	Its Meaning
Label1.Text = "مرحبا"	assigns the string literal "مرحبا" to the label's Text property
Label1.BackColor = Color.Pink	assigns the pink color to the label's BackColor property
Label1.Height = 100	assigns the number 100 to the label's Height property
Label1.Width = 100	assigns the number 100 to the label's Width property
Label1.AutoSize = False	assigns the keyword False to the label's AutoSize property
Label1.ForeColor = Color.Blue	assigns the blue color to the text that appears on the face of the label
Label1.Left = 0	specify zero as the distance of the left side of the tool relative to the left side of its container (for example: the form)
Label1.Top = 0	specify zero as the distance of the upper side of the tool relative to the upper side of its container (for example: the form)
Label1.RightToLeft = 1	specify 1 (which means yes) to the label's RightToLeft property
Label1.TextAlign = ContentAlignment.BottomRight	assigns BottomRight to the label's TextAlign property
Label1.Dock = DockStyle.Left	assigns Left to the textbox's Dock property
Label1.Visible = False	assigns the keyword False to the label's Visible property
Label1.Cursor = Cursors.Hand	assigns the Hand shape to the label's Cursor property

Table 1-7 Some examples of changing properties of a Label

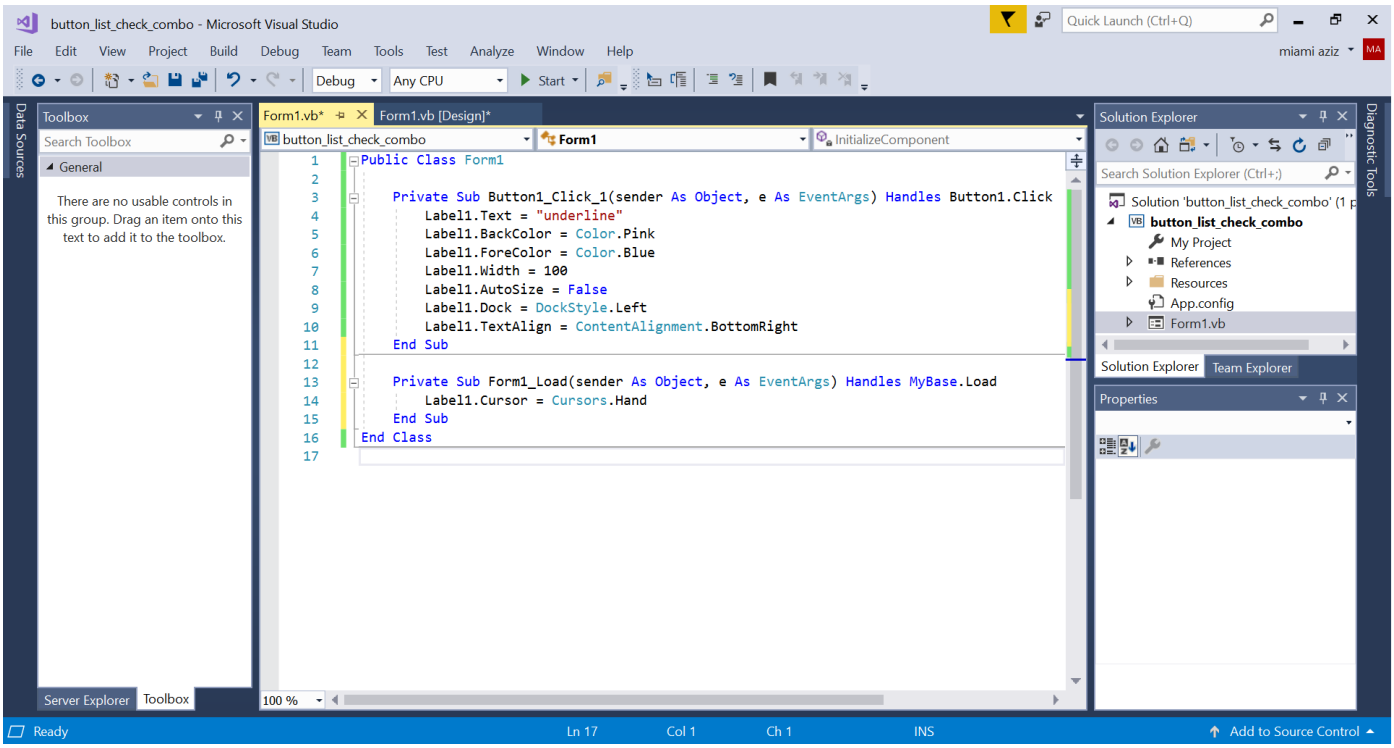

Figure 1-7 Shows examples of how to change Label properties in the Code Editor window

TextBox and Button have been added to the form. The TextBox represents the tool which we want to change its properties. The command Button is the place we will use to type most of the code to change the properties of the TextBox. Table 1-8 shows some examples of changing properties of a TextBox:

The Assignment Statement	Its Meaning
TextBox1.Text = "hello"	assigns the string literal "hello" to the textbox's Text property
TextBox1.BackColor = Color.Pink	assigns the pink color to the textbox's BackColor property
TextBox1.ForeColor = Color.Blue	assigns the blue color to the text that appears on the face of the textbox
TextBox1.Left = 0	specify zero as the distance of the left side of the tool relative to the left side of its container (for example: the form)
TextBox1.Top = 0	specify zero as the distance of the upper side of the tool relative to the upper side of its container (for example: the form)
TextBox1. RightToLeft = 1	specify 1 (which means yes) to the textbox's RightToLeft property
TextBox1.TextAlign = ContentAlignment.BottomRight	assigns BottomRight to the textbox's TextAlign property
TextBox1.Visible = False	assigns the keyword False to the textbox's Visible property
TextBox1.Height = 100	assigns the number 100 to the textbox's Height property
TextBox1.Width = 100	assigns the number 100 to the textbox's Width property
TextBox1.Multiline = True	assigns the keyword True to the textbox's Multiline property
TextBox1.MaxLength = 5	assigns number 5 to the textbox's MaxLength property
TextBox1.PasswordChar = "*"	assigns * symbol to the textbox's PasswordChar property
TextBox1.Dock = DockStyle.Top	assigns Top to the textbox's Dock property

Table 1-8 Some examples of changing properties of a TextBox

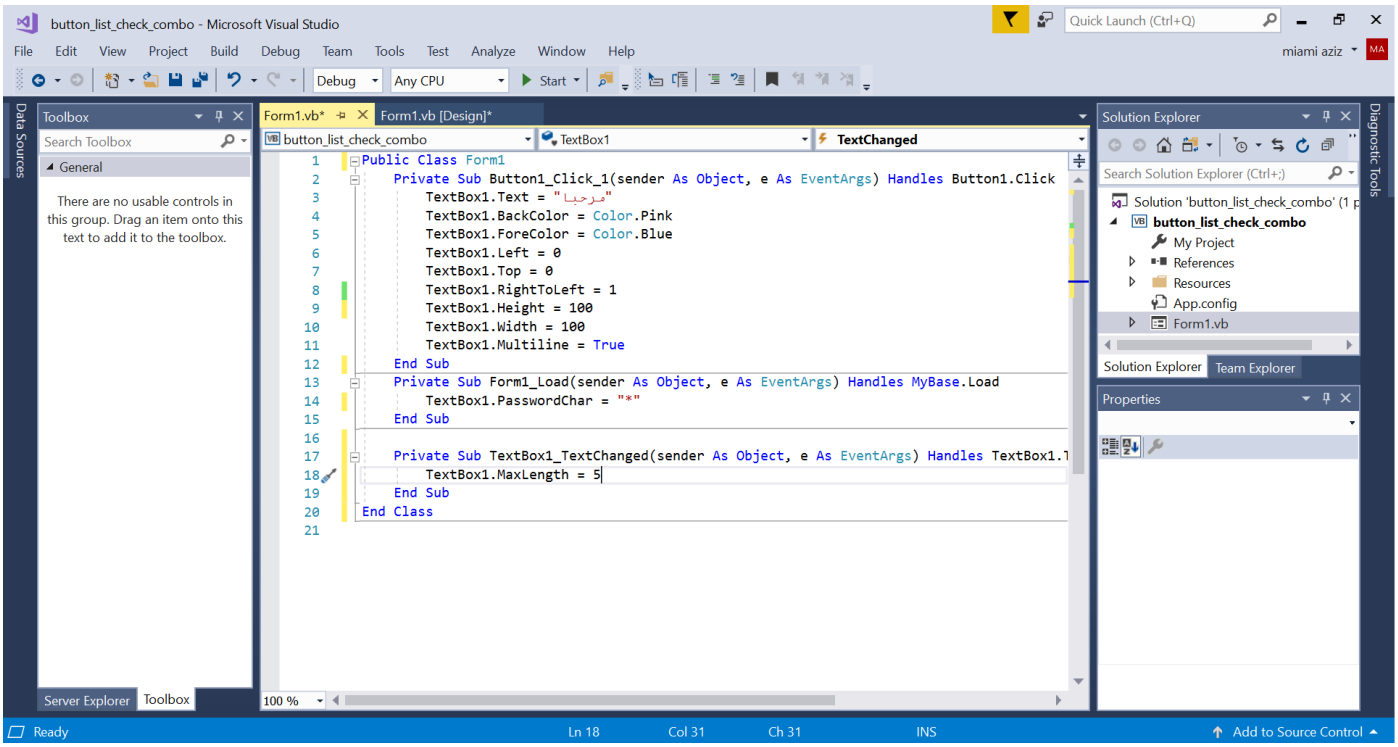

Figure 1-7 Shows examples of how to change TextBox properties in the Code Editor window