

Lab 2: Write your first .NET code

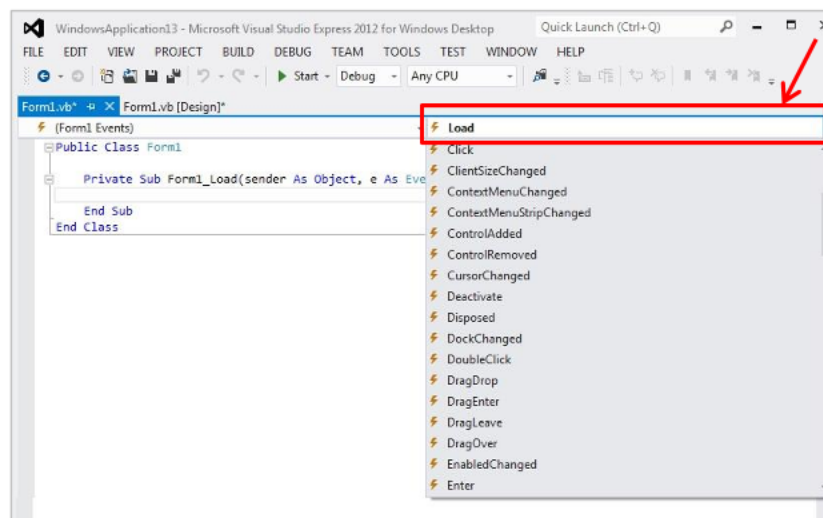
In the previous lesson, you have learned to write some simple programs without much understanding some underlying foundations and theories. In this lesson, you will learn some basic theories about Visual Basic.NET programming but we will focus more on learning by doing, i.e. learning by writing programs.

1. The event Procedure

Visual Basic.NET is an object oriented and event driven programming language. In fact, all windows applications are event driven. Event driven means the user will decide what to do with the program, whether he or she wants to click the command button, enter text in a text box, or close the application and etc. An event is related to an object, it is an incident that happens to the object due to the action of the user, such as a click or pressing a key on the keyboard. A class has events as it creates an instant of a class or an object. When we start a windows application in Visual Basic in previous lab session, we will see a default form with the name Form1 appears in the IDE, it is actually the Form1 Class that inherits from the Form class System.Windows.Forms.Form.

To start writing code in Visual Basic.NET, double click on any part of the form to go into the code window as shown in Figure. This is the structure of an event procedure. In this case, the event procedure is to load Form1 and it starts with Private Sub and end with End Sub. This procedure includes the Form1 class and the event Load, and they are bind together with an underscore, i.e. Form_Load. It does nothing other than loading an empty form. You don't have to worry the rest of the stuff at the moment, because they will be explained in later lessons.

There are other events associated with the Form1 class, such as click, cursorChanged, DoubleClick, DragDrop, Enter and more, as shown in the Figure above (It appears when you click on the upper right pane of the code window).

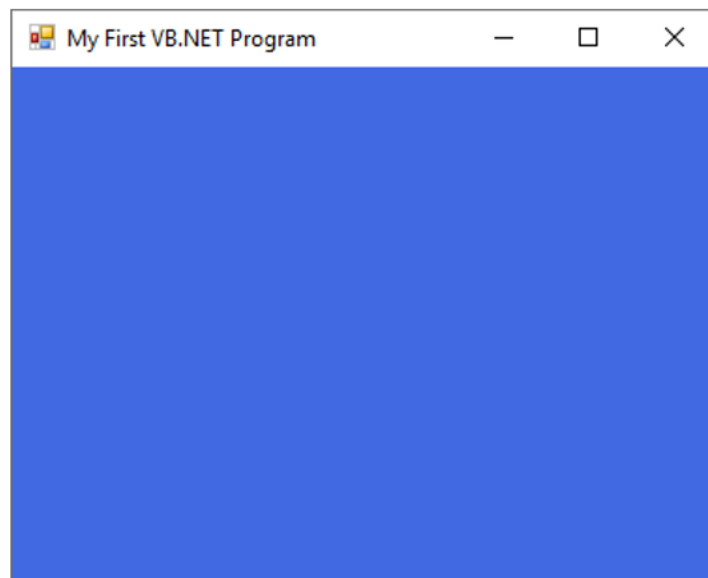


Let's enter the following code between Private Sub.....End Sub:

```
Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
    Me.Text = "My First VB.NET Program"
    Me.ForeColor = Color.LightGoldenrodYellow
    Me.BackColor = Color.RoyalBlue
End Sub
```

The first line of the code will change the title of the form to "My First Visual Basic.NET Program", the second line will change the foreground object to LightGoldenrodYellow (in this case, it is a label that you insert into the form and change its text to Foreground) and the last line changes the background to RoyalBlue color.

The equal operator = in the code is used to assign something to the object, like assigning yellow color to the foreground of the Form1 object (or an instance of Form1). Me is the name given to the Form1 class. The output is shown Figure below:



2. Mathematical Operations

Computer can perform mathematical calculations much faster than human beings. However, computer itself will not be able to perform any mathematical calculations without receiving instructions from the user. In Visual Basic.NET, we can write code to instruct the computer to perform mathematical calculations such as addition, subtraction, multiplication, division and other kinds of arithmetic operations. The Visual Basic.NET arithmetic operators are very similar to the normal arithmetic operators, only with slight variations. The plus and minus operators are the same while the multiplication operator use the * symbol and the division operator use the / symbol. The list of Visual Basic.NET arithmetic operators are shown in table below:

Operator	Operation	Example
+	Addition	$1 + 2 = 3$
-	Subtraction	$4 - 1 = 3$
*	Multiplication	$4 * 3 = 12$
/	Real Division	$10 / 4 = 2.5$
\	Integer Division (discards the decimal places)	$19 \setminus 4 = 4$
Mod	Modulus (return the remainder from an integer division)	$17 \text{ Mod } 4 = 1$
^	Exponential	$2 ^ 4 = 16$

Exercise 1

In this program, you need to insert two Textboxes, four labels and one button. Click the button and enter the code as shown below. When you run the program, it will perform the four basic arithmetic operations and displays the results on the four labels.

```
Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click
```

```
    Dim num1, num2, difference, product, quotient As Single
    num1 = TextBox1.Text
    num2 = TextBox2.Text

    sum=num1+num2
    difference=num1-num2
    product = num1 * num2
    quotient=num1/num2
    Label1.Text=sum
    Label2.Text=difference
    Label3.Text = product
    Label4.Text = quotient
```

```
End Sub
```

Exercise 2

The program can use Pythagoras Theorem to calculate the length of hypotenuse c given the length of the adjacent side a and the opposite side b . In case you have forgotten the formula for the Pythagoras Theorem, it is written as

$$c^2 = a^2 + b^2$$

```
Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles  
    Button1.Click  
  
        Dim a, b, c As Single  
        a = TextBox1.Text  
        b = TextBox2.Text  
        c = (a^2+b^2)^(1/2)  
        Label3.Text=c  
  
End Sub
```

Exercise 3: BMI Calculator

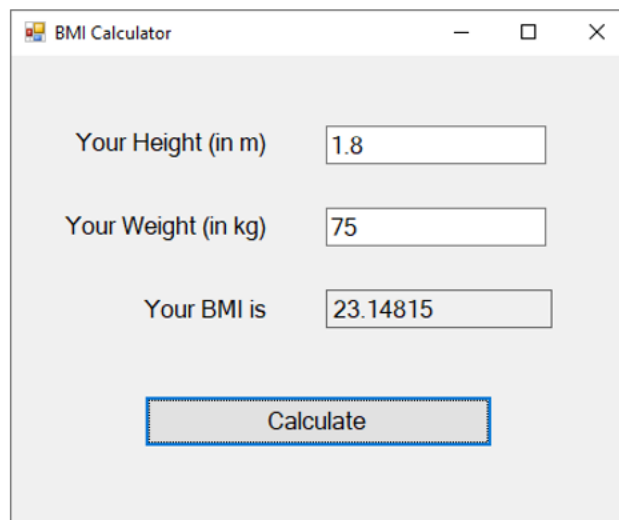
A lot of people are obese now and it could affect their health seriously. Obesity has proven by the medical experts to be a one of the main factors that brings many adverse medical problems, including the cardiovascular disease. If your BMI is more than 30, you are considered obese. You can refer to the following range of BMI values for your weight status.

Underweight = <18.5
Normal weight = 18.5-24.9
Overweight = 25-29.9
Obesity = BMI of 30 or greater

BMI can be calculated using the formula $\text{weight}/(\text{height})^2$, where weight is measured in kg and height in meter. If you only know your weight and height in lb and feet, then you need to convert them to the metric system (you could indeed write a VB program for the conversion).

```
Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles  
    Button1.Click  
  
        Dim height, weight, bmi As Single  
        height = TextBox1.Text  
        weight = TextBox2.Text  
        bmi = (weight) / (height ^ 2)  
        Label1.Text = bmi  
  
End Sub
```

The output is shown in the figure below. In this example, your height is 1.80m (about 5 foot 11), your weight is 75 kg(about 168lb), and your BMI is about 23.14815. The reading suggests that you are healthy. (Note; 1 foot=0.3048, 1 lb=.45359237 kilogram)



Input	Value
Your Height (in m)	1.8
Your Weight (in kg)	75
Your BMI is	23.14815

Calculate

3. String Manipulation

In Visual Basic.NET, string manipulation is important because it helps to process data that come in the form of non-numeric types such as names, addresses, gender, cities, book titles and more.

In Visual Basic.NET, strings can be manipulated using the & sign and the + sign, both perform the string concatenation which means combining two or more smaller strings into larger strings. For example, we can join "Visual", "Basic" and ".NET" into "Visual Basic.NET" using "Visual"&"Basic" or "Visual" + "Basic", as shown in the Examples below

Example 1

```
Private Sub Button1_Click(sender As System.Object, e As System.EventArgs) Handles  
Button1.Click  
  
    Dim text1, text2, text3, text4 As String  
    text1 = "Visual"  
    text2 = "Basic"  
    text3 = ".NET"  
    text4 = text1 + text2 + text3  
    MsgBox(text4)  
  
End Sub
```

The line `text4=text1+ text2 + text3` can be replaced by `text4=text1 & text2 & text3` and produces the same output. However, if one of the variables is declared as numeric data type, you cannot use the `+` sign, you can only use the `&` sign.

Example 2

```
Dim text1, text3 as string  
Dim Text2 As Integer  
text1 = "Visual"  
text2=22  
text3=text1+text2  
Label1.Text = text3
```

This code will produce an error because of data mismatch. However, using `&` instead of `+` will be all right.

```
Dim text1, text3 as string  
Dim Text2 As Integer  
text1 = "Visual"  
text2=22  
text3=text1& text2  
Label1.Text = text3
```