Al-Mustaqbal University College
Department of Medical Instrumentation Techniques Engineering
Class: Second Class
Subject: Computer Applications
Lecturer: Assistant Lecturer Miami Abdul Aziz
Lecture: Ninth Lecture

Determine a Memory Location's Scope and Lifetime

Besides a ***name***, a ***data type***, and an ***initial value***, every variable also has a ***scope*** and a ***lifetime***. The scope indicates where the declared memory location can be used in an application's code, and the lifetime indicates how long the variable remains in the computer's main memory.

The scope and lifetime are determined by where you declare the memory location in your code. Memory locations declared in a procedure have procedure scope and are called ***procedure-level variables***. These variables can be used only within the procedure that contains their declaration statement, and only after their declaration statement. They are removed from the computer's main memory when the procedure ends. In other words, a procedure level variable has the same lifetime as the procedure that declares it .

Memory locations declared in a form class's declarations section, but outside of any procedures, have class scope and are referred to as ***class-level variables***. The form class's declarations section is the area between the form's Public Class and End Class clauses in the Code Editor window. However, class-level declaration statements typically appear immediately after the Public Class <formname> clause .

Class-level variables can be used by all of the procedures in the class that contains their declaration statement. In addition, they have the same lifetime as the application, which means they retain their values and remain in the computer's main memory until the application ends.

Al-Mustaqbal University College
Department of Medical Instrumentation Techniques Engineering
Class: Second Class
Subject: Computer Applications
Lecturer: Assistant Lecturer Miami Abdul Aziz
Lecture: Ninth Lecture

Use Procedure-Level Variables

Procedure-level variables are typically declared at the beginning of a procedure, and they can be used only after their declaration statement within the procedure. They remain in the computer's main memory only while the procedure is running, and they are removed from memory when the procedure ends. As mentioned earlier, most of the variables in your applications will be procedure-level variables. This is because fewer unintentional errors occur in applications when the variables are declared using the minimum scope needed, which usually is procedure scope.

The Commission Calculator application illustrates the use of procedure-level variables. As the interface shown in Figure 2-18 indicates, the application displays the amount of a salesperson's commission. The commission is calculated by multiplying the salesperson's sales by the appropriate commission rate: either 8% or 10%.

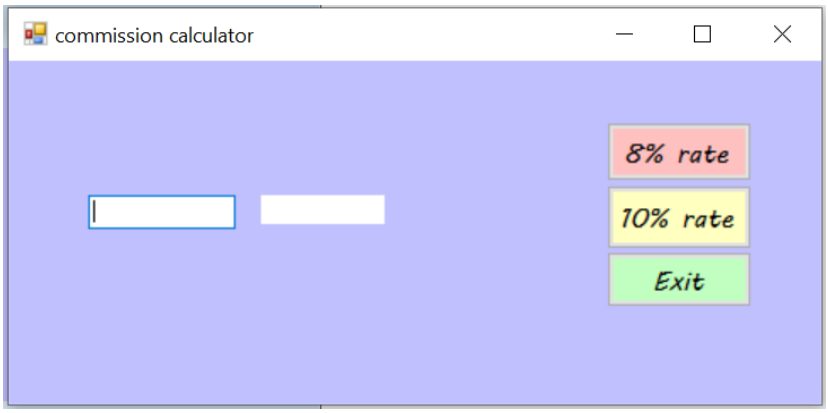

Figure 2-18 User interface for the Commission Calculator application

Figure 2-19 shows the Click event procedures for the 8% rate and 10% rate buttons. When each procedure ends, its procedure-level variables are removed from the computer's memory. The variables will be created again the next time the user clicks the button.

```
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Dim sales As Double
        Dim comm As Double
        sales = TextBox1.Text
        comm = sales * 0.08
        Label1.Text = comm
        Label1.BackColor = Button1.BackColor
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
        Dim sales As Double
        Dim comm As Double
        sales = TextBox1.Text
        comm = sales * 0.1
        Label1.Text = comm
        Label1.BackColor = Button2.BackColor
    End Sub

    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
        Close()
    End Sub
End Class
```

Figure 2-19 Click event procedures using procedure-level variables

Notice that both procedures in Figure 2-19 declare a variable named *sales*. When you use the same name to declare a variable in more than one procedure, each procedure creates its own variable when the procedure is invoked. Each procedure also destroys its own variable when the procedure ends. So, although both procedures declare a variable named *sales*, each sales variable will refer to a different section in the computer's main memory, and each will be both created and destroyed independently from the other.