Program Example 13  Moving Buttons 1

Objectives

Program Purpose

- To annoy the user greatly, as well as delight the programmer, by:
  1. moving a ‘Register Later’ button as the user attempts to click it
  2. setting a 25 second time limit on the program
  3. shrinking the form after the 25 seconds has finished

Learning Goals

- To use timers, constants and loops in an annoying registration form
- To demonstrate the use of ‘Message Box’ options
- To alter the size of the form at runtime
- To introduce separate procedures that can be reused
- To use PUBLIC declarations
- To use ‘Reserved Visual Basic Constants’ (vbYes, vbNo)
- To use the ‘MouseMove’ event

Design Notes

‘vbYes’ and ‘vbNo’ are built-in Visual Basic constants, used with message boxes. For example in the line:

vbResponse = MsgBox(msgRegister3, vbYesNo + vbDefaultButton2 + vbCritical, msgTitle3)

an integer will be assigned to the variable ‘vbResponse’. The value of this integer depends on whether the user clicks a ‘Yes’ or ‘No’ button. The following line of code will then be:

If vbResponse = vbYes Then

The value of vbResponse is checked against the built-in value of VbYes (which corresponds to the user clicking ‘Yes’) and the decision whether or not to open the registration form is made accordingly.

The built-in constants vbDefaultButton2, vbCritical, vbYesNo, vbExclamation, vbQuestion and vbInformation are used to determine what sort of message box appears.

Interface

Create the interface as shown below. Use 3 command buttons and 1 timer.

Names of Objects

<table>
<thead>
<tr>
<th>Type of Object</th>
<th>Number</th>
<th>Names of Objects</th>
<th>Simple Initial Properties of Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form</td>
<td>1</td>
<td>frmRegistrationNow</td>
<td>Caption – “Moving Buttons 1”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Startup Position - 2 – Center Screen</td>
</tr>
<tr>
<td>Timer</td>
<td>1</td>
<td>tmrCountDown</td>
<td>Font – Bold, 10</td>
</tr>
<tr>
<td>Command Buttons</td>
<td>3</td>
<td>CmdRegisterLater, cmdRegisterNow, cmdHelp, cmdExit</td>
<td>Font – Bold, 12 Captions – as per Interface</td>
</tr>
</tbody>
</table>
Further Initial Properties of Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Property</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>tmrCountDown</td>
<td>Enabled</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Interval</td>
<td>100</td>
</tr>
</tbody>
</table>

Events – Code

** PUT THE FOLLOWING CODE IN THE GENERAL SECTION OF THE CODE WINDOW **

Const ButtonWidth = 1000
Const ButtonHeight = 500
Const MouseGap = 200
Const RegistrationTimeLimit = 25

Const msgHelp1 = "Welcome to the XYZ Program. Please register now to unlock the 30 day expiry code on this program. If you don't activate registration then the program will end automatically."
Const msgHelp2 = "seconds to register!"

Const msgRegister0 = "Congratulations! You outwitted the slow old computer!"
Const msgRegister1 = "Are you sure you want to register later?"
Const msgRegister2 = "Are you really sure you want to register later?"
Const msgRegister3 = "Are you really, really sure you want to register later?"
Const msgRegister4 = "Wrong Choice, Dude!"

Const msgTitle1 = "Think Carefully!"
Const msgTitle2 = "Persistent Aren’t You?"
Const msgTitle3 = "Warning!"
Const msgTitle4 = "No More Chances!"

Public TimeGone As Single 'used to keep track of time left to register
Public SecondsLeft As Single 'as above
Public vbResponse As Integer 'used to store user's responses to Message Box dialog boxes

Sub delay(X As Single)
'useful for delaying program events
Dim starttime As Single
starttime = Timer
While starttime + X > Timer
   DoEvents
Wend
End Sub

Sub OpenRegistration()
'Create and open a registration form
MsgBox "This opens up a registration form which asks for payment along with all your personal details." , "Hand over your money!"
End Sub

Private Sub cmdRegisterNow_Click()
'Call the Module for Registration
OpenRegistration
End Sub
Sub ProgramEnd()

'SelfDestructSequence

'form contract settings
Const FormContractSpeed = 0.1
Const FormContractAmount = 200

'shrinkwidth
While frmRegistrationNow.Width > 1000 'dont make this value too small
    frmRegistrationNow.Width = frmRegistrationNow.Width - FormContractAmount
    delay FormContractSpeed
Wend

'shrinkheight
While frmRegistrationNow.Height > 500 'dont make this value too small
    frmRegistrationNow.Height = frmRegistrationNow.Height - FormContractAmount
    delay FormContractSpeed
Wend

'end program finally
End

End Sub

** COPY THE FOLLOWING CODE TO THE APPROPRIATE EVENTS **

Private Sub cmdRegisterLater_Click()

'Uses Nested IF-THEN-ELSE-END-IF structures to annoy the user

vbResponse = MsgBox(msgRegister0 & msgRegister1, vbYesNo + vbDefaultButton2 + vbQuestion, msgTitle1)

If vbResponse = vbYes Then
    vbResponse = MsgBox(msgRegister2, vbYesNo + vbDefaultButton2 + vbExclamation, msgTitle2)

    If vbResponse = vbYes Then
        vbResponse = MsgBox(msgRegister3, vbYesNo + vbDefaultButton2 + vbCritical, msgTitle3)

        If vbResponse = vbYes Then
            MsgBox msgRegister4, vbInformation, msgTitle4
            ProgramEnd
        Else
            OpenRegistration
        End If
    Else
        OpenRegistration
    End If
Else
    OpenRegistration
End If

End Sub

Private Sub cmdHelp_Click()

    tmrCountDown.Enabled = False
    MsgBox msgHelp1 & SecondsLeft & msgHelp2
    tmrCountDown.Enabled = True

End Sub
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If cmdRegisterLater.Left < ButtonWidth Then
        cmdRegisterLater.Left = frmRegistrationNow.Width - ButtonWidth
    Else
        cmdRegisterLater.Left = X - (ButtonWidth + MouseGap)
    End If

    If cmdRegisterLater.Top < ButtonHeight Then
        cmdRegisterLater.Top = frmRegistrationNow.Height - (2 * ButtonHeight)
    Else
        cmdRegisterLater.Top = Y - (ButtonHeight + MouseGap)
    End If
End Sub

Private Sub tmrCountDown_Timer()
    'keeps track of how much time left to register
    'the amount added to the variable TimeGone must equal the
    'Interval setting for the timer in the Properties Box
    'Remember that this is milliseconds
    TimeGone = TimeGone + 0.1
    SecondsLeft = RegistrationTimeLimit - TimeGone
    If SecondsLeft >= 0 Then
        frmRegistrationNow.Caption = "Registration Form - " & Format(SecondsLeft, ".##" & " Seconds Left!")
    Else
        ProgramEnd
    End If
End Sub

Notes

The ‘MouseMove’ event returns the position of the mouse in terms of X and Y coordinates. The X and Y coordinates are based on the form. The top left corner has the coordinates (0,0).

Understanding the role of the constants and their settings is the key to understanding this program. If the variable 'MouseGap' is set too low, then the user finds it too easy to 'catch' the button. 'ButtonHeight' and 'ButtonWidth' should be the measurement of the buttons height and width. These can be found in the command button properties in 'Design View'.

Suggestions for Consolidation and Extension

1. Create another form for registration. Open it when the user clicks the ‘Register Now’ button. Use the following code to hide and show the forms:
   
   frmRegistrationNow.Hide
   frmActuallyRegisterNow.Show

2. Modify each constant in turn and see what effect it has on your program. Don’t forget the two constants declared in the ‘ProgramEnd’ procedure itself:
   
   Const FormContractSpeed = 0.1
   Const FormContractAmount = 200

3. Experiment with the delay values in the ‘ProgramEnd’ procedure.