


## Controls

The icons on the toolbox represent controls (or objects) that can be placed on the form.

1. Start a new Windows Application and name it Colors.
2. If the toolbox is not visible click  or from the menu select View, ToolBox.
3. Place four buttons on the form and name them BtnRed, BtnBlue, BtnYellow, and BtnPink. and change the properties to match the table and illustration below:

Control	Property	Value
BtnRed	Text	&Red
	BackColor	Red
BtnBlue	Text	&Blue
	BackColor	Blue
BtnYellow	Text	&Yellow
	BackColor	Yellow
BtnPink	Text	&Pink
	BackColor	Pink



Note: An ampersand, **&**, in text makes the next letter a short-cut key. After you press Enter, the text will appear as Blue. Typing ALT+B is the same as clicking the button that says Blue.

4. Double-click on BtnRed, the code opens with the `btnRed_Click` event ready to add code.
5. Change the name of the procedure to `Color_Click` and write the code as shown below, using the `handles` clause to make the code execute for each of the buttons.

```
Private Sub Color_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles BtnRed.Click, BtnBlue.Click, BtnPink.Click,
    BtnYellow.Click
    Me.BackColor = sender.BackColor
    Me.Text = sender.Text
End Sub 'Color_Click
```

## Sender

Notice the word **sender** in the argument list: `sender` refers to the button that was clicked. `Sender` can be used instead of the name of the button:

The main difference between using `sender` and the name of the button is that if you type a dot after `btnYellow`, a list of properties pops up. When you type a dot after `sender` the list of properties of a button does not appear and you have to type the property name yourself.


6. Run the program and click each button. The background will turn the same color as the `BackColor` of the button that was clicked.

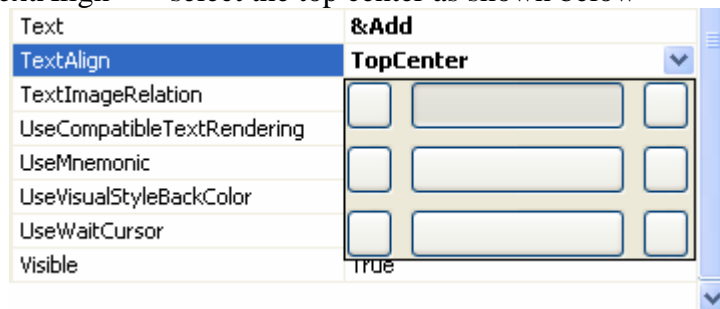
Writing one procedure and using `Handles` to handle several events makes the code more general and less specific.

## Pictures

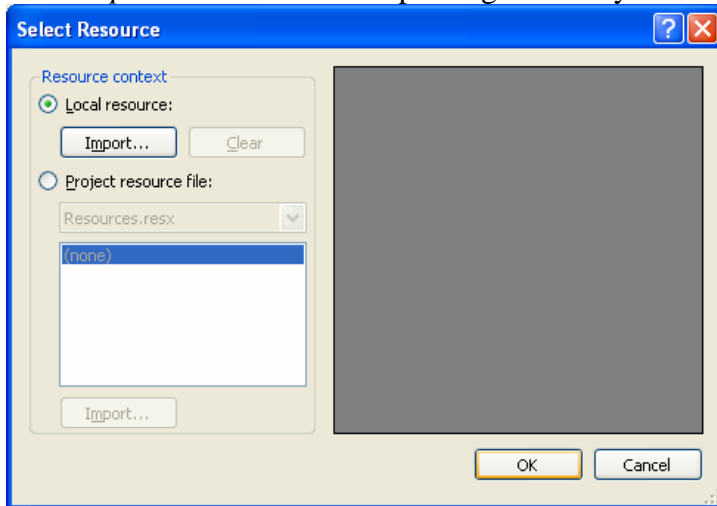
The next application uses two pictures: plus.bmp  and minus.bmp . These are available at [www.hello-world.com/vb2005](http://www.hello-world.com/vb2005) as part of this lesson.

## Adding

1. Save the color program, then close it and start a new windows application. Name the application Calculate.
2. Put a button  on the form and name it BtnAdd.
3. Select **BtnAdd** and change the following properties:
  - BackColor     Select Black
  - ForeColor     Select Red
  - Text           &Add
  - TextAlign     select the top center as shown below

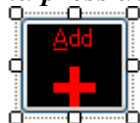


**Image:** When you select the Image property a dialog window opens: Select *Local Resource*, then lick *Import* to browse for the plus sign. When you see the picture in the window, click OK.



ImageAlign    Select BottomCenter

*If you accidentally get the picture in the wrong place, for instance on the form, just click on the name of the image in the property window and press delete.*



4. Drag the size so that the button looks like this:

5. Select btnAdd and copy it. (Ctrl+C), then paste (Ctrl+V).

You will get a new button named Button1 with all of the same properties as BtnAdd.

6. Name the button BtnMinus. Change the following properties:

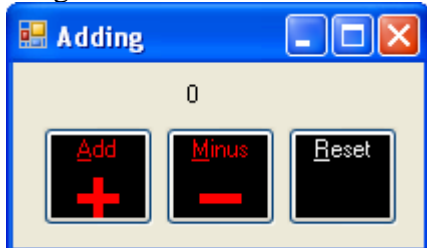
Text           &Minus  
Image          Select the minus picture.

7. Make another copy of the button. Name it btnReset. Change the following properties:

Text           &Reset  
Image          Select the image property and type the delete key so that the property says (none)

8. Add a label **A** and name it LblNumber. Set text to 0. Set AutoSize to true.

Drag the buttons and labels so that the form will look like this:




9. Go to the code window and write the code as shown below. Make sure that you select the events and only write the lines in the middle of the procedure!

10. Declare a global variable Number. Write the code for the click event for each button to Add or subtract from Number and display the result in LblNumber. Try to write the code on your own before looking at the code.

```
Public Class Form1
    Dim Number As Integer = 0
    Private Sub BtnAdd_Click(ByVal sender As System.Object, ByVal e As
        System.EventArgs) Handles BtnAdd.Click
        Number = Number + 1
        Me.LblNumber.Text = Number
    End Sub 'BtnAdd_Click


    Private Sub BtnMinus_Click(ByVal sender As System.Object, ByVal e As
        System.EventArgs) Handles BtnMinus.Click
        Number = Number - 1
        Me.LblNumber.Text = Number
    End Sub 'BtnMinus_Click

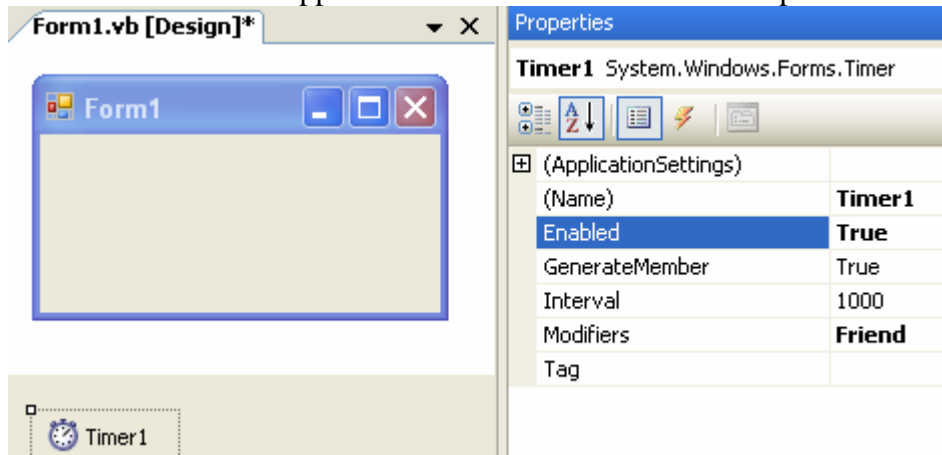
    Private Sub BtnReset_Click(ByVal sender As System.Object, ByVal e As
        System.EventArgs) Handles BtnReset.Click
        Number = 0
        Me.LblNumber.Text = Number
    End Sub 'BtnReset_Click
End Class
```

Test the program to make sure each button does what it should. Save the program by clicking save all .

## Timer



The controls that we have used so far had procedures that were executed in response to mouse events: clicking a button, or scrolling the scrollbar. The timer control has just one procedure, *Tick* that is executed at regular intervals. The *Interval* is a property that can be changed: smaller values for interval mean that the timer “goes off” faster. Large values for interval mean that the action is slower. The interval can not be 0.

1. Start a new windows application, and name it TimerFun.
2. In the toolbox, double click on the timer control: . The timer control is not visible at run time and it does not appear on the form. Instead it is in the pane at the bottom of the screen.



3. Change the Interval to 1000 and change enabled to True.
4. Double click the timer and write the code as shown below to give the screen a new background color at each tick of the timer. (*This is the code from the lesson on variables.*)

```
Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Timer1.Tick
    Dim Red, Green, Blue As Integer
    Red = Rnd() * 255
    Green = Rnd() * 255
    Blue = Rnd() * 255
    Me.BackColor = Color.FromArgb(255, Red, Green, Blue)
    Me.Text = Red & ", " & Green & ", " & Blue
End Sub 'Timer1_Tick
```


5. Add a scroll bar to the form ( or ). Change the name to ScrSpeed. The scroll bar will be used to adjust the interval of the timer.
6. The Timer1.Interval can not be 0, so set ScrSpeed.Min to 1.
7. Set ScrSpeed.Max to 10000.
8. Double click on the scroll bar and write the code shown below:

```
Private Sub ScrSpeed_Scroll(ByVal sender As System.Object, ByVal e As
    System.Windows.Forms.ScrollEventArgs) Handles ScrSpeed.Scroll
    Me.Timer1.Interval = Me.ScrSpeed.Value
End Sub 'ScrSpeed_Scroll
```

**Experiment:** Add two buttons to the form, "Stop" and "Go". Set Timer1.Enabled to True or False when you click the buttons.

## ToolTip

The tool tip is the little words in a yellow box that pops up when you move the mouse over an area.

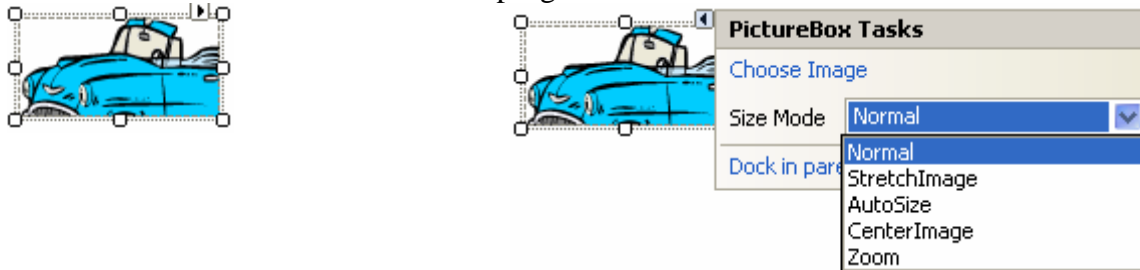
1. In the toolbox, double click on the ToolTip icon . The tool tip does not appear on the form. Instead it is in the panel at the bottom of the screen.
2. Select ScrSpeed and scroll down to the Tooltip property and you will see that there is now a property called Tooltip on Tooltip1. Set this property to “Change speed.”


Run the program and move the mouse over the Scrollbar. You should see the word “Change speed” pop up. You may have to be patient: the default properties for the tooltip make it rather slow.

3. Select Tooltip1 and look at the properties. Change ShowAlways to True. Adjust any of the variables that refer to Delay and run the program again. Adjust the values so that you know what they do and find settings you like.

## PictureBox

Start a new project called Car. From the toolbox add a PictureBox to the form. Name it PicCar. Select the image property and select the picture of the car. The picture probably does not fit in the box. Click the little arrow in the top right corner of the PicCar and select AutoSize.



Add a timer , Timer1 to the project. Set the timer property enabled to true. Make the interval 20.

Write the following code:

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles MyBase.Load
    Me.PicCar.Left = Me.Width 'just off the form on the right
End Sub 'Form1_Load

Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Timer1.Tick
    Me.PicCar.Left = Me.PicCar.Left - 1
    If Me.PicCar.Left < -1 * Me.PicCar.Width Then
        Me.PicCar.Left = Me.Width 'just off the form on the right
    End If
End Sub 'Timer1_Tick
```

Run the program. Notice that the car exits on the left and reappears on the right.

Add a button and name it BtnStopGo. Add the following code for the button:

```
Private Sub BtnStopGo_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles BtnStopGo.Click
    Me.Timer1.Enabled = Not Me.Timer1.Enabled
    If Me.Timer1.Enabled Then
        Me.BtnStopGo.Text = "&Stop"
        Me.BtnStopGo.BackColor = Color.Red
    Else
        Me.BtnStopGo.Text = "&Go"
        Me.BtnStopGo.BackColor = Color.Green
    End If
End Sub 'BtnStopGo_Click
```

The first line of code says to make the enabled property the opposite of whatever it is now. If it is true it becomes false; if it is false, it becomes true. The rest of the code changes the color and text of the button.

This button is said to “toggle” the enabled value of the timer.

**Experiment:** Add a scroll bar to change the speed.