Assignment 17 – Graphics and Timers

The timer control triggers an event after a specified amount of time has elapsed. It is not visible during runtime. Double-click on the **Timer Control** in the toolbox and it will appear at the bottom of the form design.
The length of time is measured in milliseconds and is set with the **Interval property**. A timer fires off its Tick event again and again, using the **Interval** property to determine how many milliseconds to wait between ticks.

The **Enabled property** must be set to **TRUE** in order to begin the timer. The timer is turned off by setting the property to FALSE.

A **Timer** object uses a **Start()** method to starts the timer, and a **Stop()** method that stop it.

When you set the timer's **Enabled** property to **True** in the **Properties** window, it starts ticking as soon as the program begins. But when you leave it set to **False**, it doesn't start ticking until its **Start()** method is called or the **Enabled property** is set to **True.**

```
timer1.Stop()

timer1.Start()
```

- Create a form with a button and a timer. Set the text property of the button to ***Start***.
- Double click on the button then copy and past the code below:

```
        Me.Timer1.Start()
```

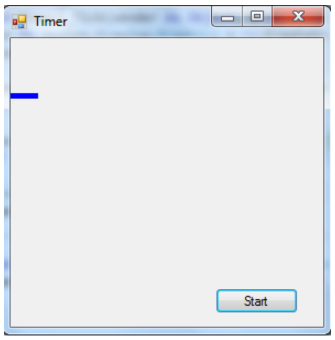
- Globally declare a variable named **bluex** as an integer.
- Double click on the timer then copy and paste the code below:

```
    Dim box As System.Drawing.Graphics = Me.CreateGraphics()
    Dim blueBrush As New SolidBrush(Color.Blue)

    If bluex = 250 Then  ' if box reaches finish line, disable timer
        Me.Timer1.Enabled = False
    Else
        box.FillRectangle(blueBrush, bluex, 50, 25, 5)
        ' draws a rectangle using (brush, x-coordinate, y-coordinate, width, height)
    End If

    bluex = bluex + 50   ' increments the x-coordinate
```

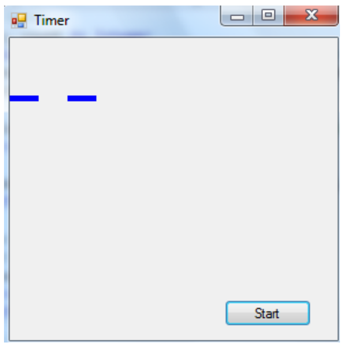
- Run the application.

- Adjust the speed of the timer by changing the values in the Interval property of the timer.

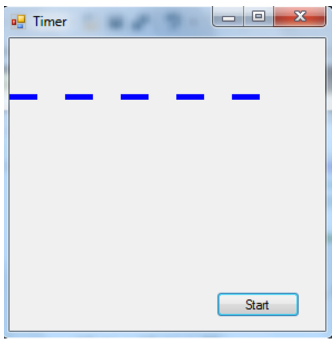
- By adjusting the value for **bluex** (the horizontal start position or *x*-coordinate) the distance between successive rectangles will change**.** Adjust the value to make the successive rectangles appear as a solid line.



After 1 tick of timer



After 2 ticks of timer



After 5 ticks of timer

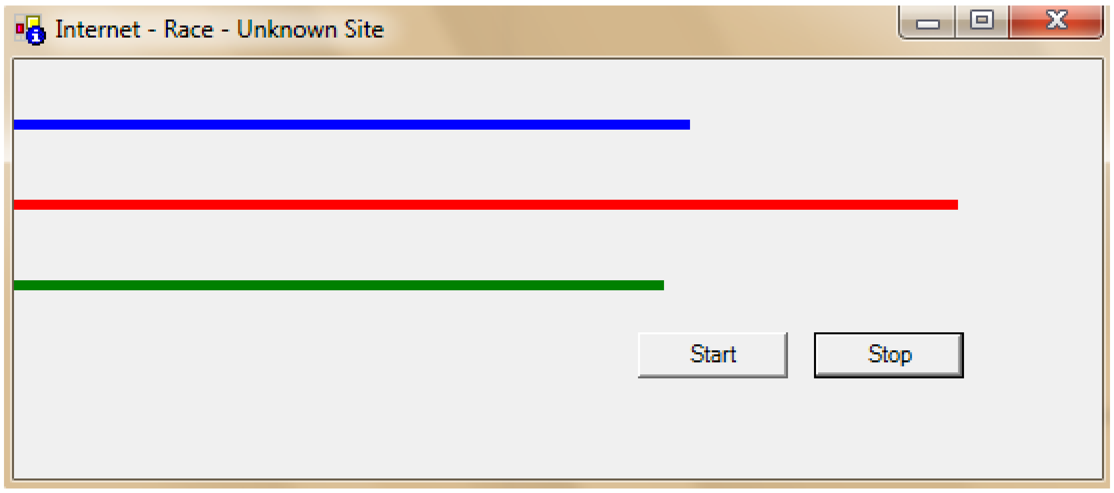
Timer Assignment

Graphic Races

Part 1:
Create an application to replicate geometric shapes racing horizontally or vertically to a finish line.
The same shape should be used for all contestants, each being a different color.
The movement of each contestant should be determined using a randomly generated number.
When one of the contestants reaches the finish line, the time should be disabled and a winner declared.

Part 2:
Adjust your randomly generated numbers in way that makes the path of the racers appears as a solid line if it doesn't already appear as such.

Part 3:
Add an addition button to continue the race (re-enable the timer)
Continue the race until all contestants have crossed the finish line except for the last place racer, displaying 1st, 2nd, 3rd,…..place finishers.
Allow the user to choose a color before the beginning of the race.


Internet - Race - Unknown Site
Start
Stop