# CO453 Application Programming

Week 11 – Graphics using GDI+

.NET part 6

# The Graphics Class

# Graphics Class

This is in the **System.Drawing** namespace, so include the line:

**using System.Drawing;**

The Graphics Class has methods such as:

- DrawString()
- DrawLine()
- DrawEllipse
- DrawRectangle()
- DrawPolygon()
- FillRectangle()
- etc.

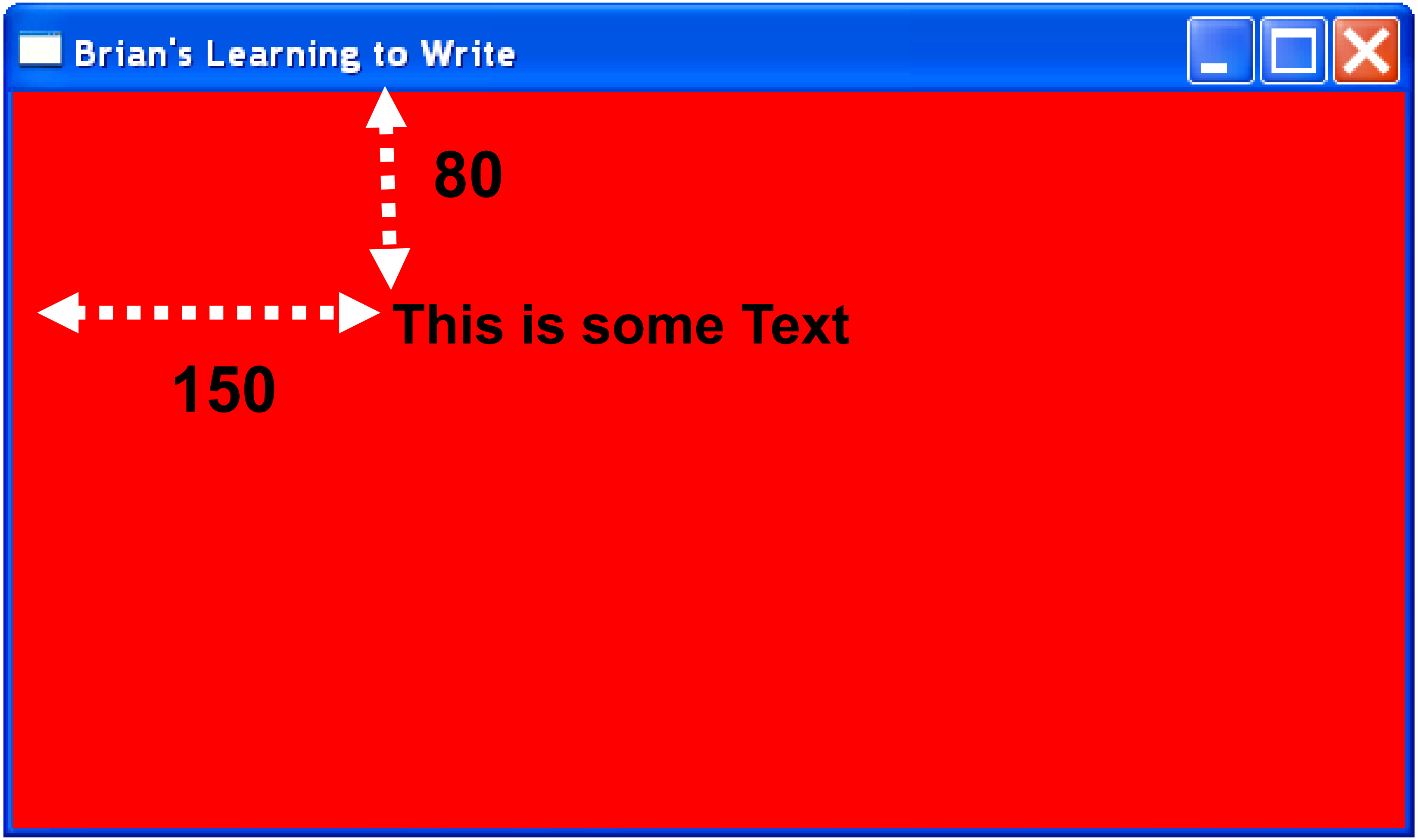
To use these methods we must have a Graphics **object**

```
private void Form1_Paint (object sender, PaintEventArgs  e){

    Graphics g = e.Graphics;

}
```

# DrawString()

# Using DrawString()

```
Font  myFont = new Font("Arial", 17);
g.DrawString("This is some Text", myFont, Brushes.Black, 150, 80);
```

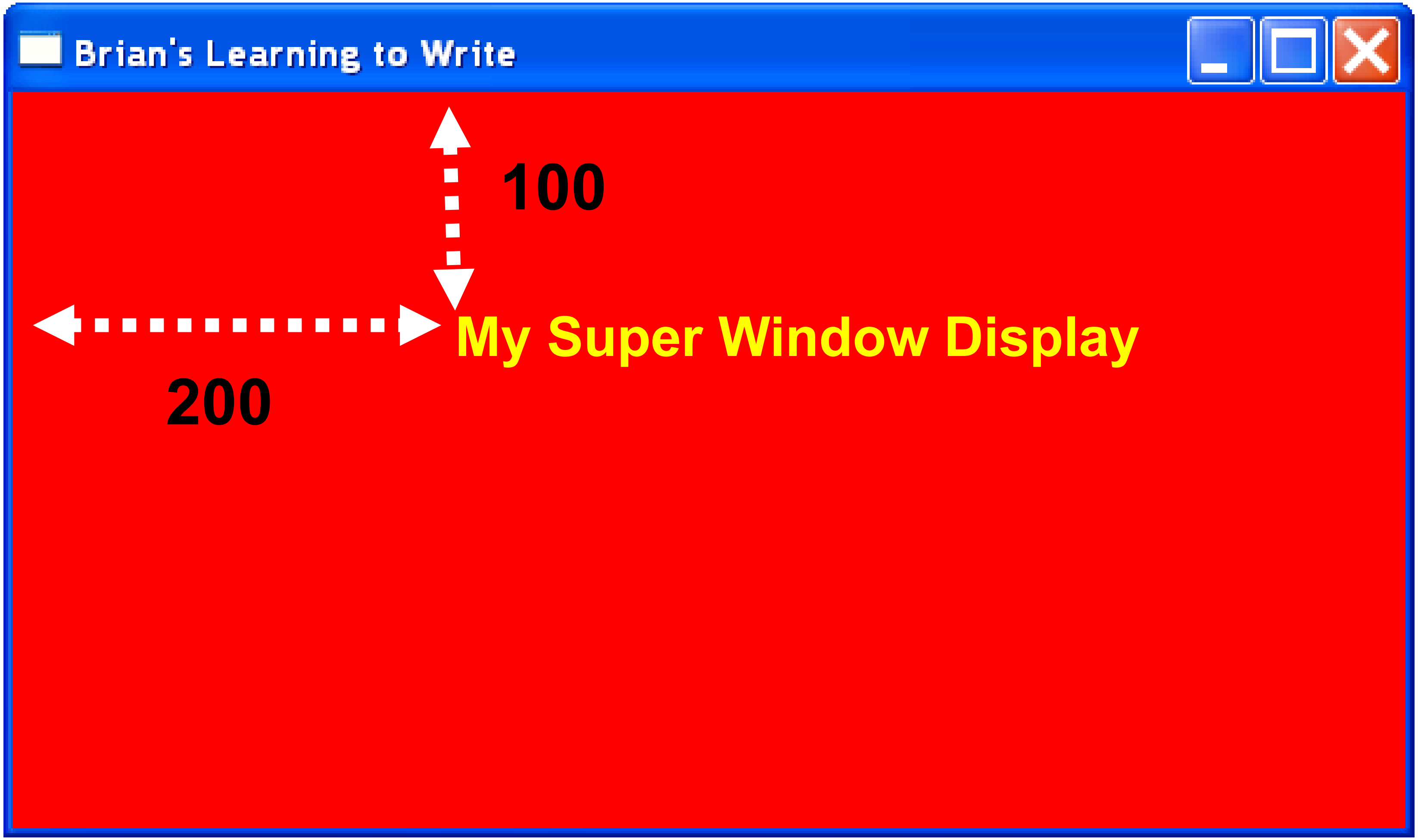


# Using DrawString()-2

```
private void Form1_Paint (object sender, PaintEventArgs  e)
 {
          // this Paint method is called whenever the form is repainted
      int x = 200, y = 100 ;
      string s = "My Super Window Display";    // string to print

      Graphics g = e.Graphics;                 // get a graphics object g

      Font myFont = new Font("Arial", 20);     // create new font object

      g.DrawString(s, myFont, Brushes.Yellow, x, y);

 }
```

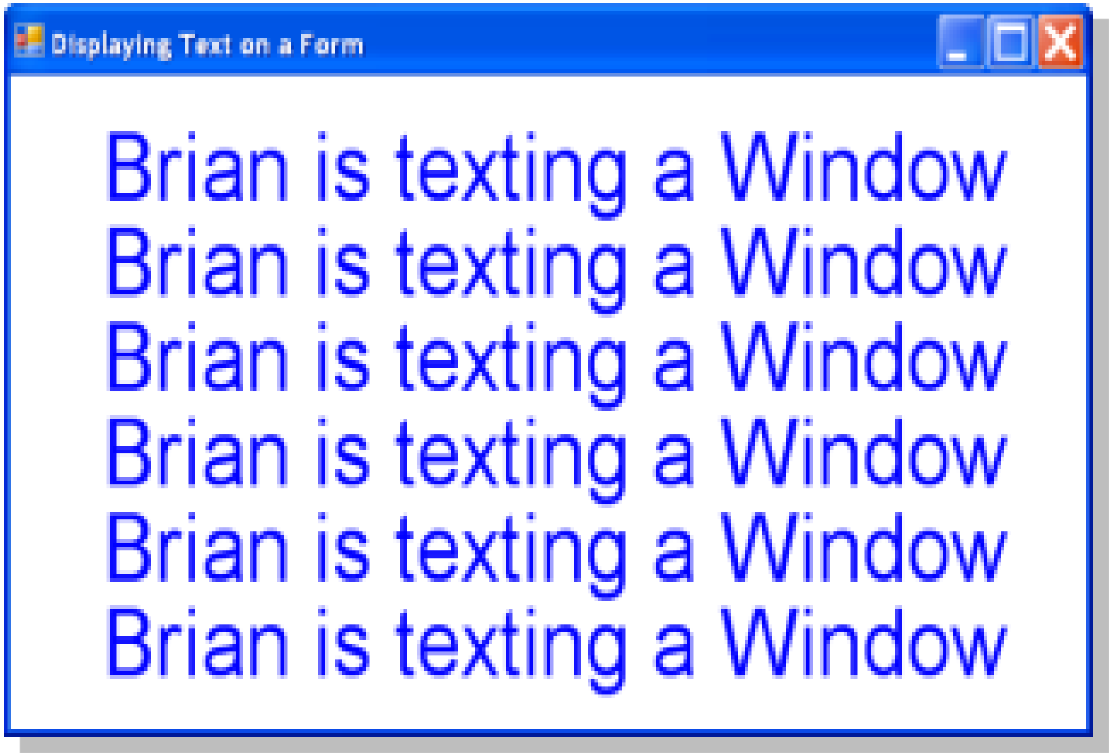
# Result

**g.DrawString**(s, myFont, Brushes.Yellow, x, y);



# Activity

- Output text by using the DrawString method in Task 6.1



Brian is texting a Window
Brian is texting a Window
Brian is texting a Window
Brian is texting a Window
Brian is texting a Window
Brian is texting a Window

# Drawing Shapes

# and

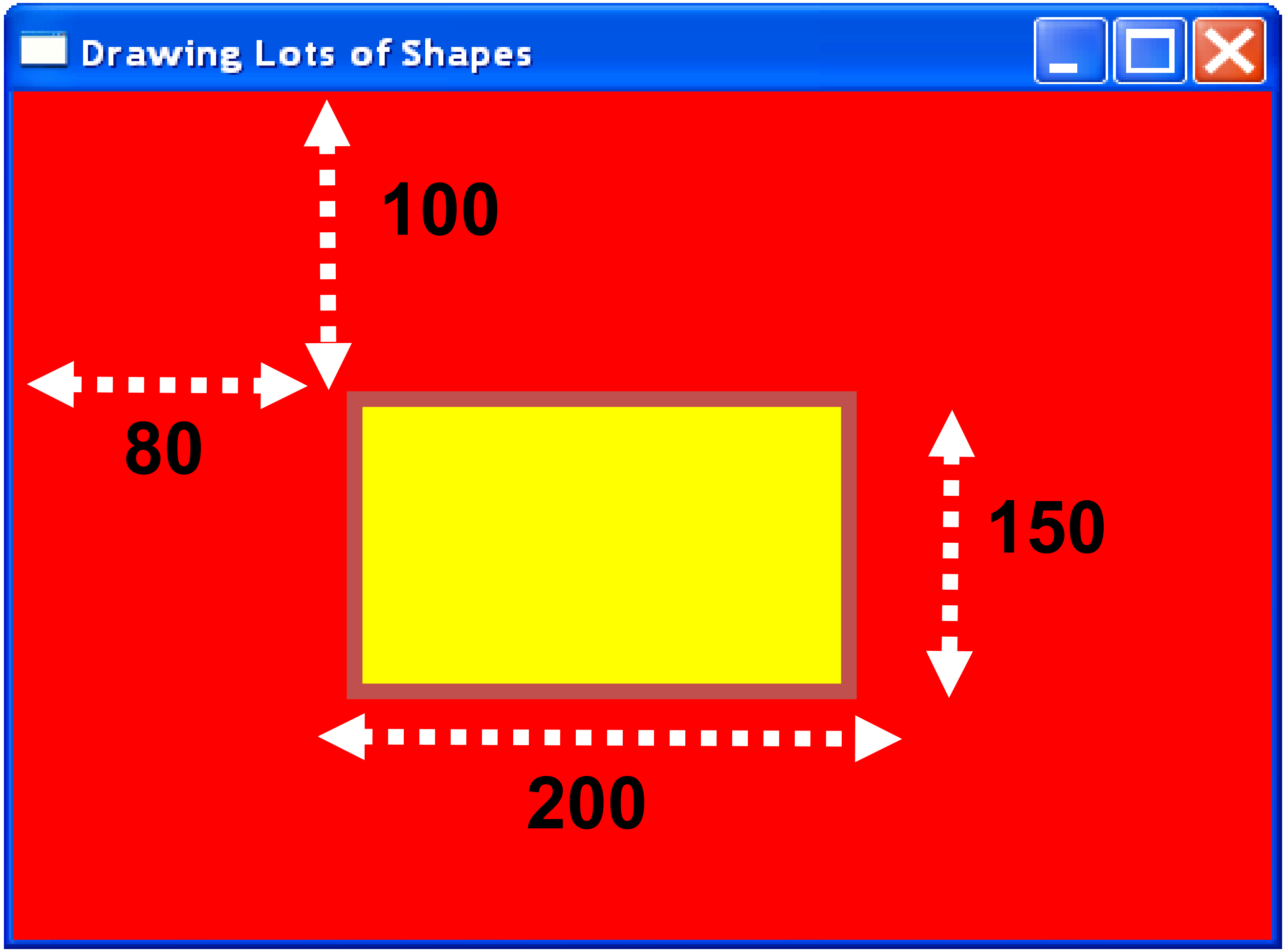
# Lines

# Drawing Rectangles and Squares

```
private void Form1_Paint (object sender, PaintEventArgs  e)
{
        // this Paint method is called whenever the form is repainted

    Graphics g = e.Graphics;                    // get a graphics object g

    Pen myPen = new Pen(Color.Blue, 5);   // create new Pen object

    g.DrawRectangle(myPen, 80, 100, 200, 150 );
    g.FillRectangle(Brushes.Yellow, 80, 100, 200, 150 );

}
```

# Result

```
g.DrawRectangle(myPen, 80, 100, 200, 150 );
g.FillRectangle(Brushes.Yellow, 80, 100, 200, 150 );
```

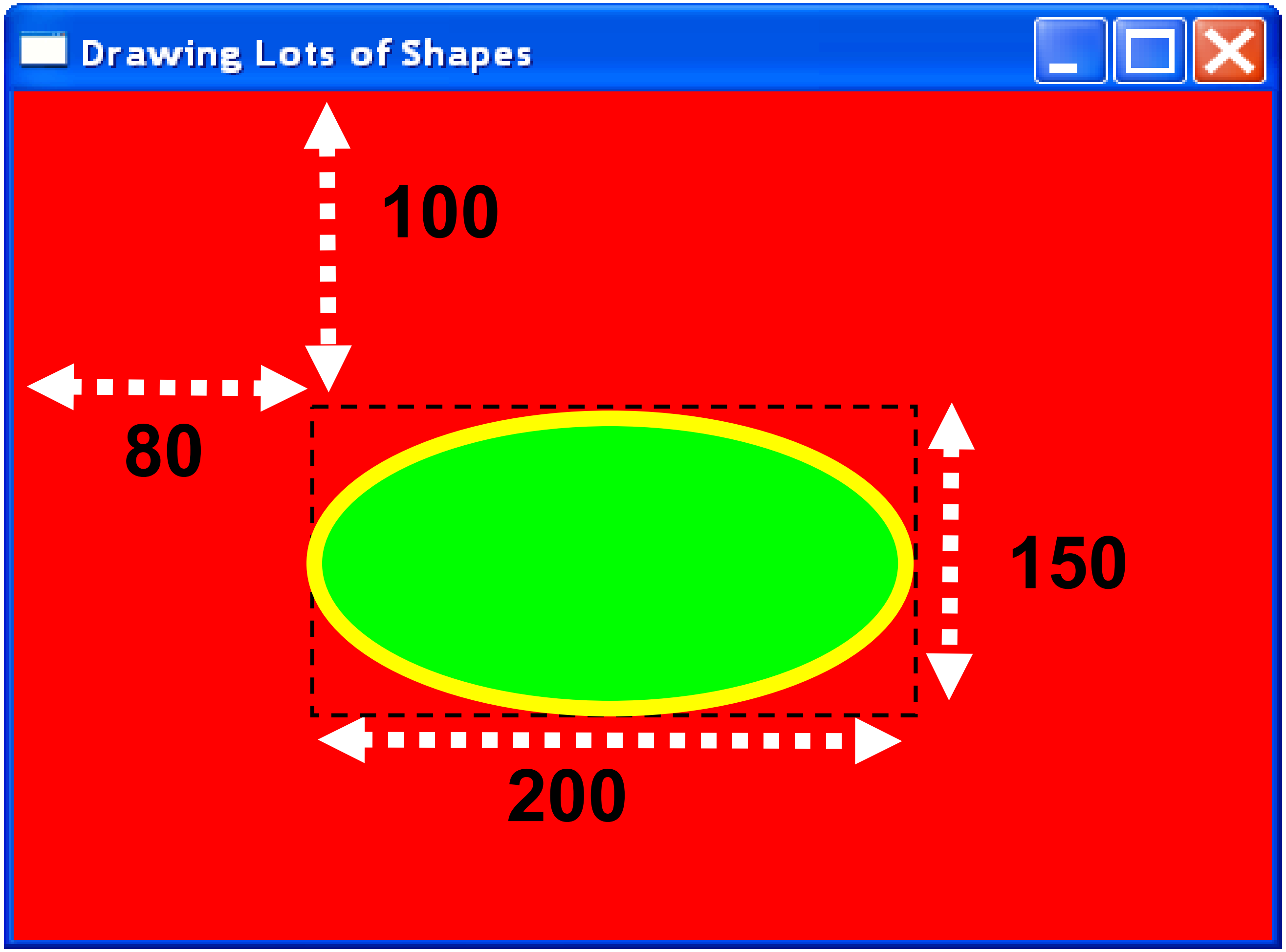


# Drawing Ellipses and Circles

```
private void Form1_Paint (object sender, PaintEventArgs  e)
 {
         // this Paint method is called whenever the form is repainted

    Graphics g = e.Graphics;                    // get a graphics object g

    Pen myPen = new Pen(Color.Yellow, 6);   // create new Pen object

     g.DrawEllipse(myPen, 80, 100, 200, 150 );
     g.FillEllipse(Brushes.Green, 80, 100, 200, 150 );

 }
```

# Result

```
g.DrawEllipse(myPen, 80, 100, 200, 150 );
g.FillEllipse(Brushes.Green, 80, 100, 200, 150 );
```



# Changing the drawing Brush

# Brushes

You can set up a **brush** and use this to colour a graphic

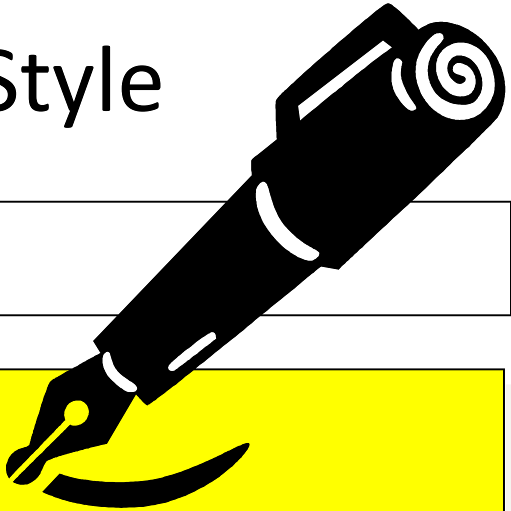
Example

```
BRUSH myBrush;                    // define a brush variable
myBrush = new SolidBrush (Color.Red);
g.FillEllipse (myBrush, x, y, w, h);  // use brush to paint
```

```
// or you can use the full range of RGB colour values:
myBrush = new SolidBrush(Color.FromArgb(0, 255, 0));
```

# Changing the drawing Pen

# Pen Colour, Thickness and Style



Set up a **pen** and then use it to draw lines

Example

```
Pen myPen;     // define a pen variable
myPen = new Pen (Color.FromArgb(50, 200, 30), 10 );
myPen.DashStyle = DashStyle.Solid;  // the default
```

Other DashStyles

.Dash

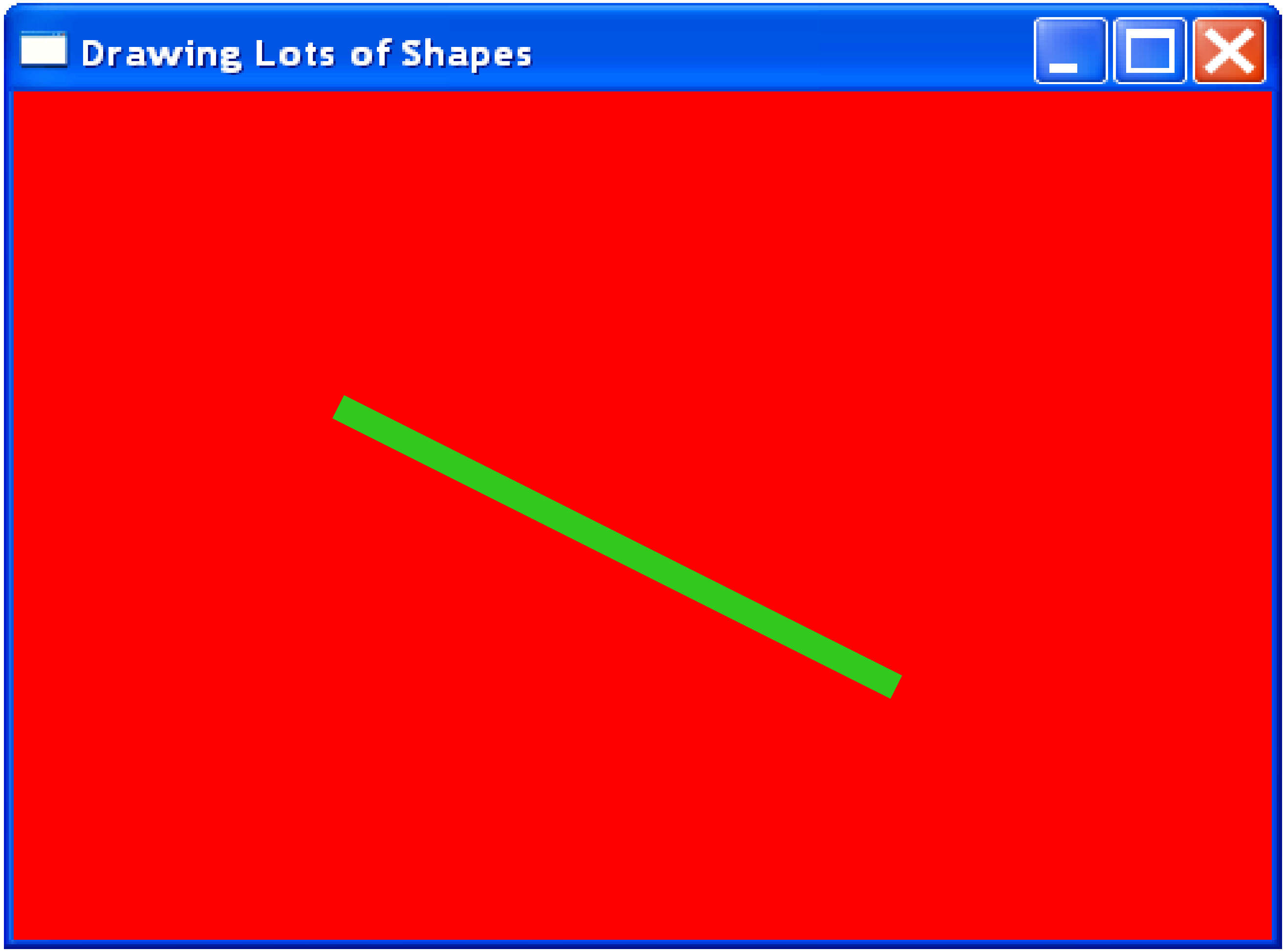
.DOT

.DASHDOT

.DASHDOTDOT

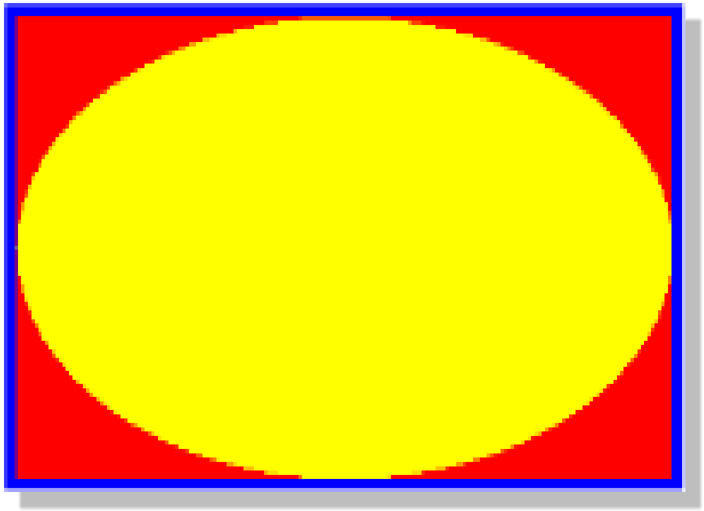
This creates a solid pen, 10 pixels wide, greenish colour

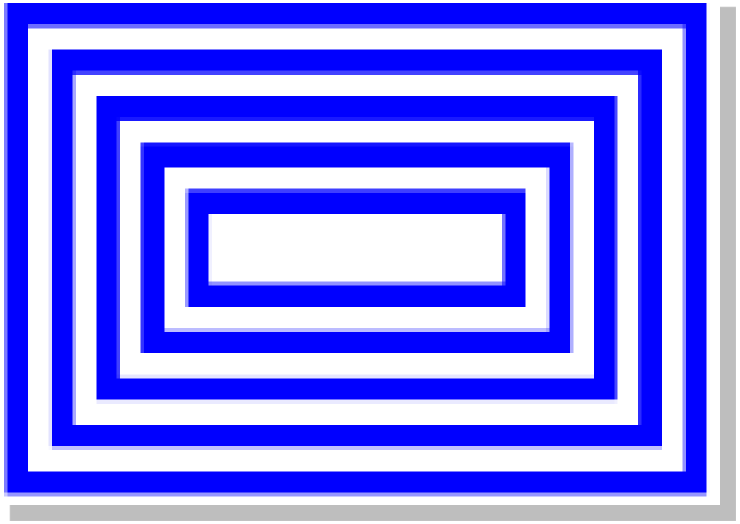
# New Result



# Activity

- Attempt to draw the shapes described in Task 6.2 with different coloured pens





# Drawing

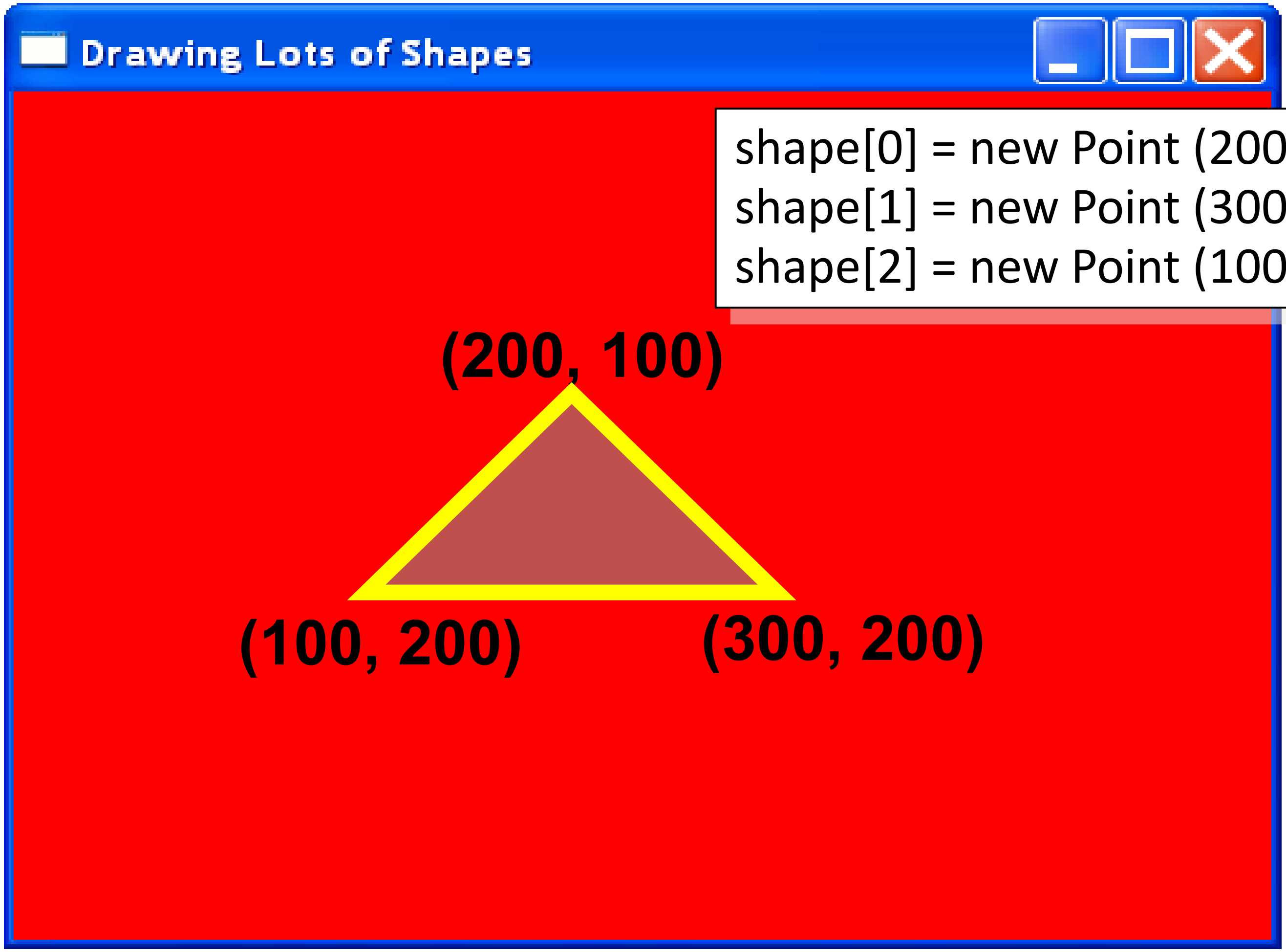
# Polygons

# Drawing Polygons

```
private void Form1_Paint (object sender, PaintEventArgs  e)
 {
            // this Paint method is called whenever the form is repainted

    Graphics g = e.Graphics;                         // get a graphics object g
    Pen myPen = new Pen(Color.Yellow, 6);   // create new Pen object

     Point[ ] shape = new Point[3];      // an array of points
     shape[0] = new Point(200, 100);  // add 3 points
     shape[1] = new Point(300, 200);
     shape[2] = new Point(100, 200);

     g.DrawPolygon(myPen, shape );
     g.FillPolygon(Brushes.Blue, shape );

 }
```

# Result

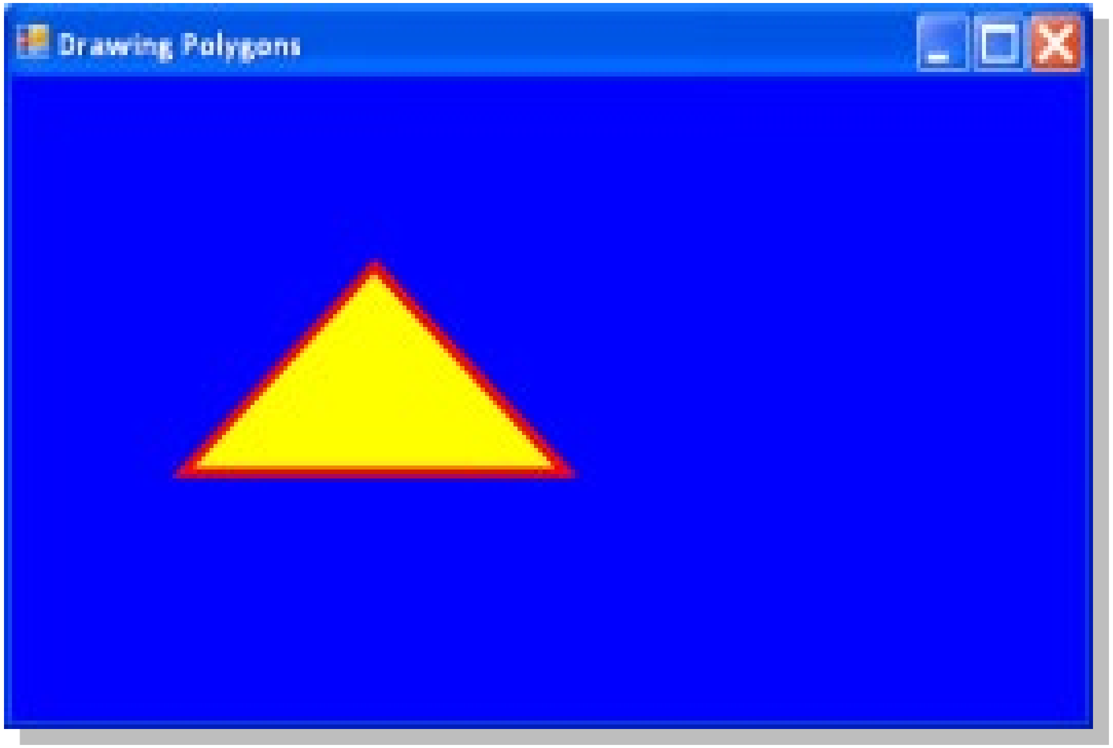
```
g.DrawPolygon(myPen, shape );
g.FillPolygon(Brushes.Blue, shape );
```



```
shape[0] = new Point (200, 100);
shape[1] = new Point (300, 200);
shape[2] = new Point (100, 200);
```

# Activity

- Draw the polygon described in Task 6.3

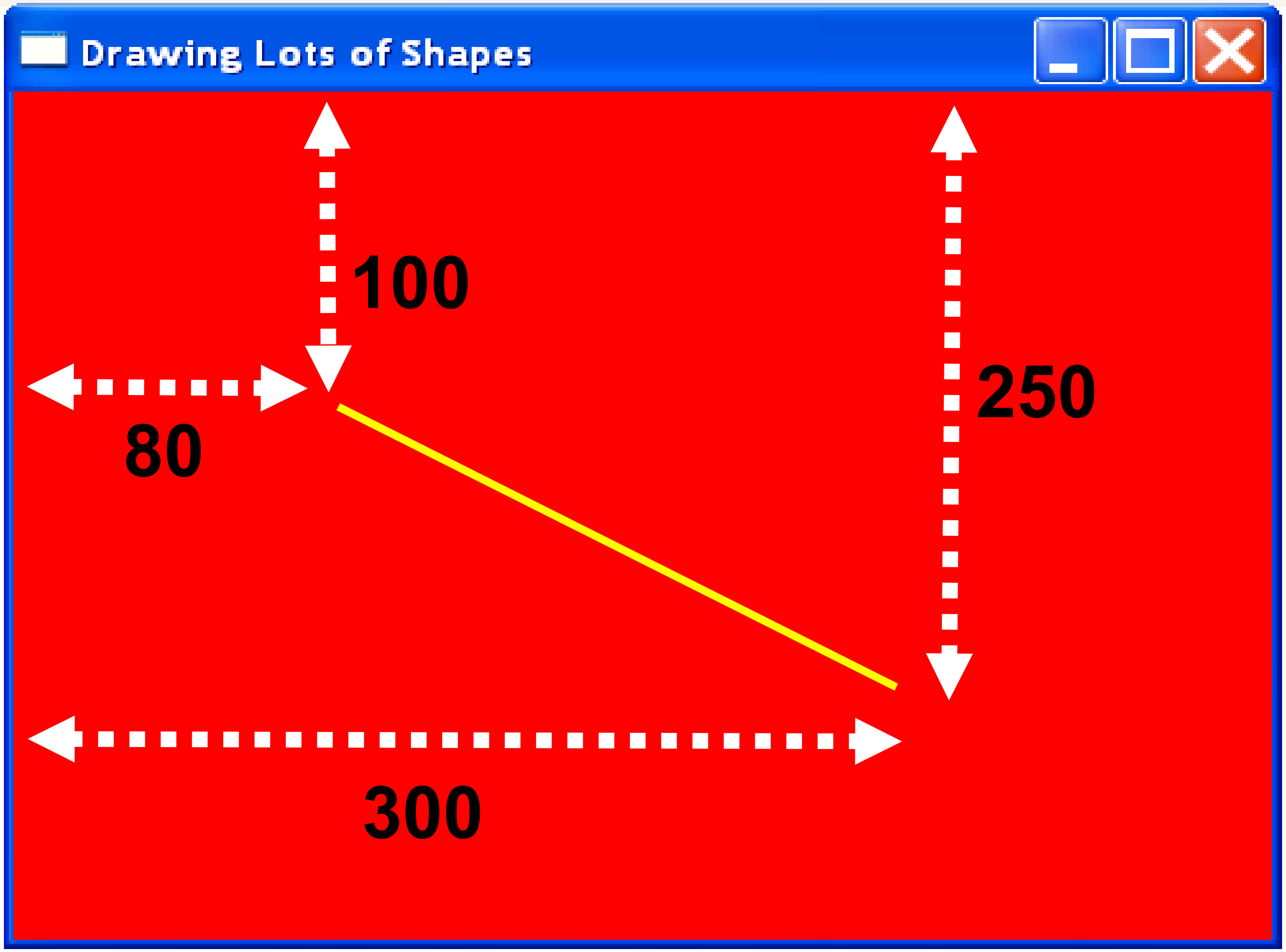


# Drawing Lines

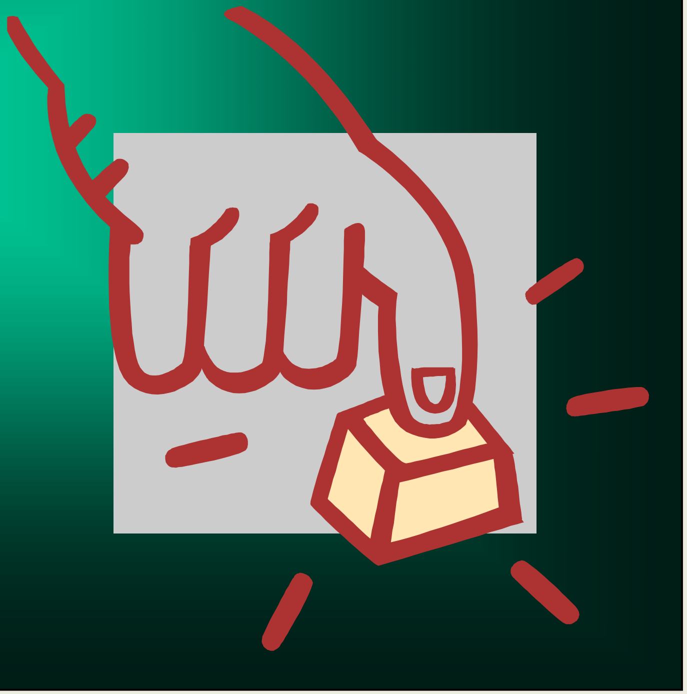
```
private void Form1_Paint (object sender, PaintEventArgs  e)
 {
          // this Paint method is called whenever the form is repainted

    Graphics g = e.Graphics;                 // get a graphics object g

    Pen myPen = new Pen(Color.Yellow, 10);   // create pen object

       // draw line from (80, 100) to (300, 250)
    g.DrawLine(myPen, 80, 100, 300, 250);

 }
```

# Result

```
g.DrawLine(myPen, 80, 100, 300, 250 );
```



# How to Detect A Key Press



# Overriding ProcessCmdKey() method

```
protected override bool ProcessCmdKey (ref Message msg,
                                        Keys keyData)
{
    string input;
    input = keyData.ToString();        // collect the key data
    if (input == "B")
    {
        MessageBox.Show("You pressed the B key");
        return  true;
    }

    return  false;          // return false if no key pressed
}
```

# Activity

- Design a program that will draw different shapes depending upon which button is pressed in Task 6.4

# The Last Slide