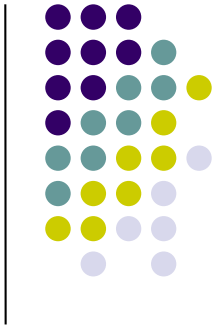


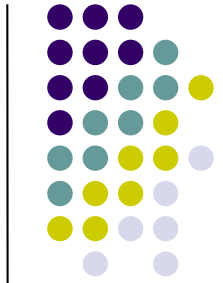
Drawing Operations

NameSpaces :

System.Drawing

System.Drawing.Drawing2D

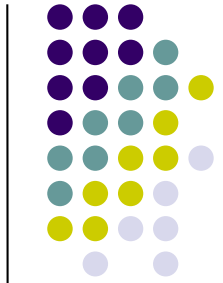
Color



- Used for representing color

- There are two ways to create color object
  - RGB values
    - Color c = Color.FromArgb(100, 100, 255);
  - Predefined Colors
    - Color c1 = Color.LavenderBlush;

Graphics Class



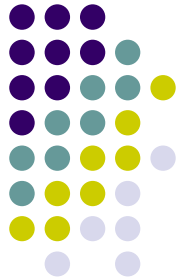
- Using for drawing a surface.
- There are two ways to obtain graphics object
  - In Paint event handler

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;
}
```

  - In other functions

```
private void Form1_Click(object sender, EventArgs e)
{
    Graphics g = CreateGraphics();
}
```

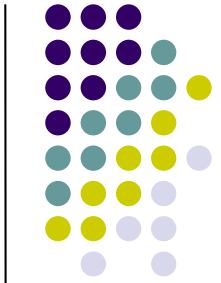
Graphics Class Functions



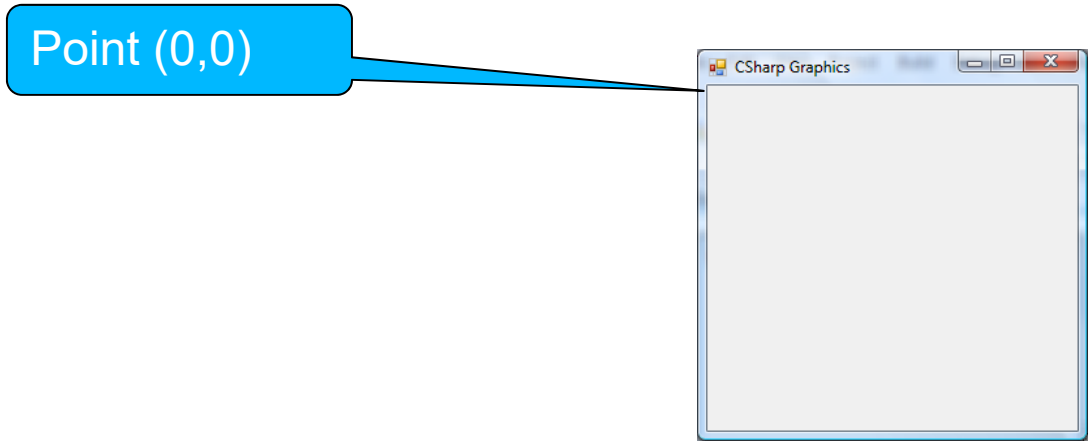
- DrawCurve
- DrawEllipse
- DrawImage
- DrawPath
- DrawPolygon

- DrawRectangle
- DrawString
- FillEllipse
- FillPath
- FillRectangle

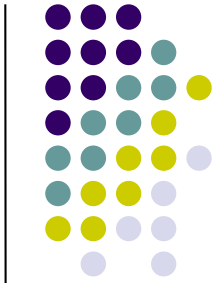
Point Structure



- Represents a point
- Used in some functions or structures to represent a location
- Definition example Point p = new Point(2, 1);
- Point (0,0) is upper-left corner of the form

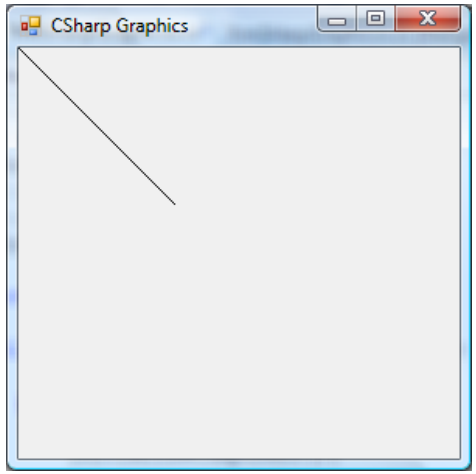


Pen Class

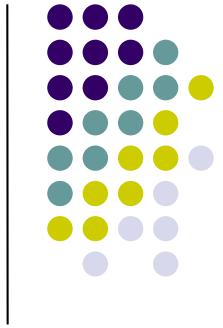


- Used for drawing lines, arcs and shapes.
- Simple usage;

```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Black);

    g.DrawLine(myPen, 0, 0, 100, 100);

    myPen.Dispose();

}
```



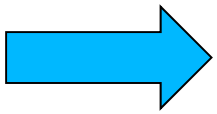
Pen Class - Properties

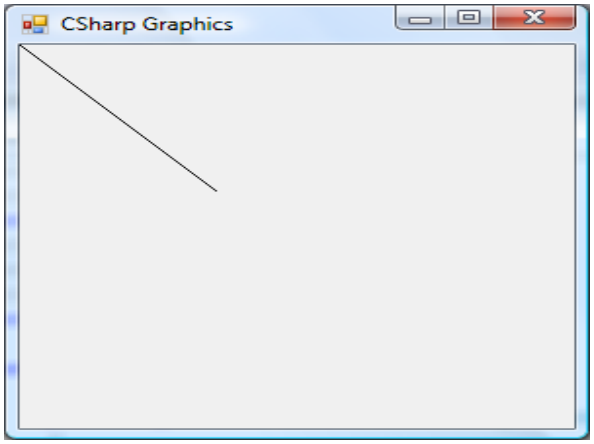


- Color : Color for pen.
- Width : Width of the pen.
- DashStyle: Style of dashed lines.
- StartCap – EndCap : Cap style used in start and end of the line.
- LineJoin : ıntersection style of consecutive lines (smooth, sharp)

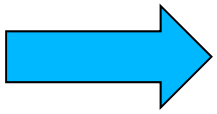
Pen Class – Color & Width Properties

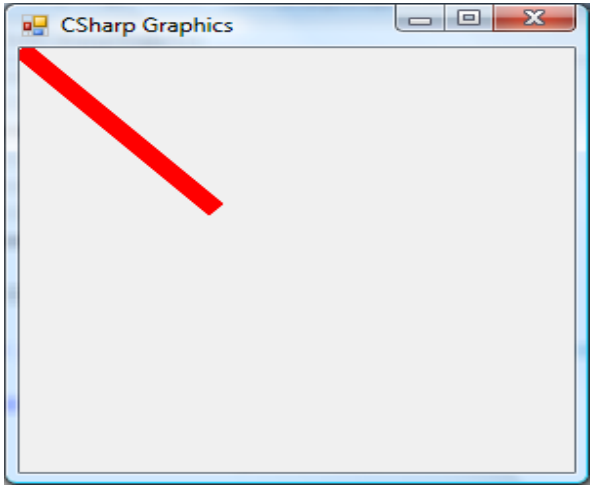
```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Black);
    myPen.Width = 1;

    g.DrawLine(myPen, 0, 0, 100, 100);

    myPen.Dispose();
}
```






```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Red);
    myPen.Width = 10;

    g.DrawLine(myPen, 0, 0, 100, 100);

    myPen.Dispose();
}
```

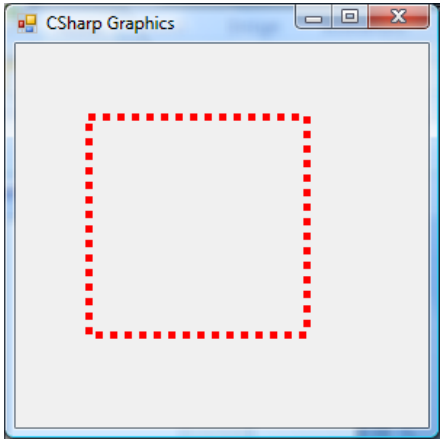




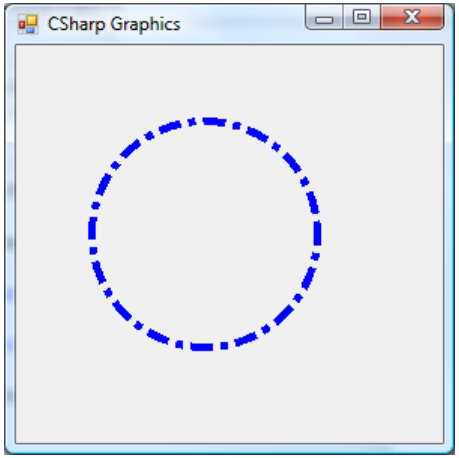
Pen Class – DashStyle

- Line drawing style

```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Red,5);
    myPen.DashStyle = DashStyle.Dot;

    g.DrawRectangle(myPen, 50, 50, 150, 150);

    myPen.Dispose();
}
```

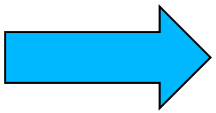


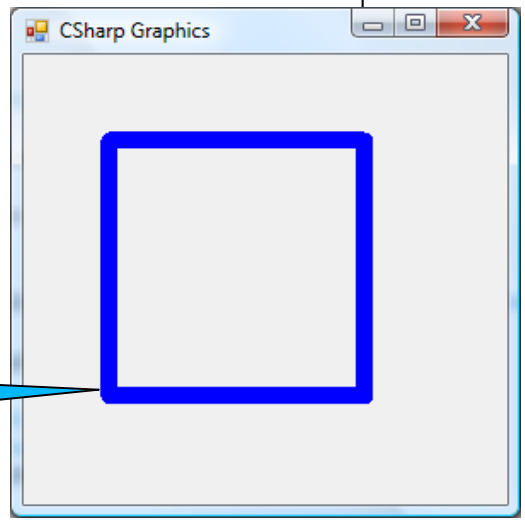

```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Blue,5);
    myPen.DashStyle = DashStyle.DashDot;

    g.DrawEllipse(myPen, 50, 50, 150, 150);

    myPen.Dispose();
}
```



Pen Class – LineJoin & LineCaps

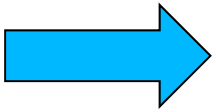
```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Blue,10);
    myPen.LineJoin = LineJoin.Round;
    g.DrawRectangle(myPen, 50, 50, 150, 150);

    myPen.Dispose();
}
```

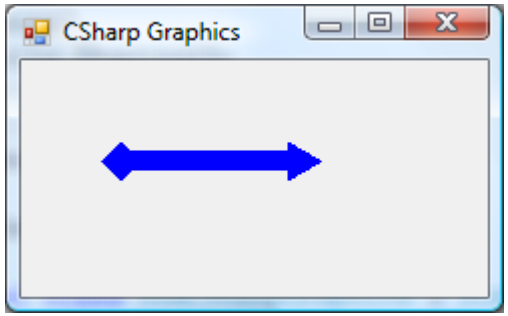




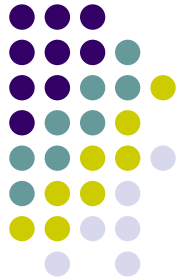
Edges are smooth

```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen myPen = new Pen(Color.Blue,10);
    myPen.StartCap = LineCap.DiamondAnchor;
    myPen.EndCap = LineCap.ArrowAnchor;
    g.DrawLine(myPen, 50, 50, 150, 50);

    myPen.Dispose();
}
```



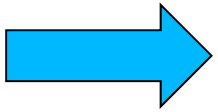


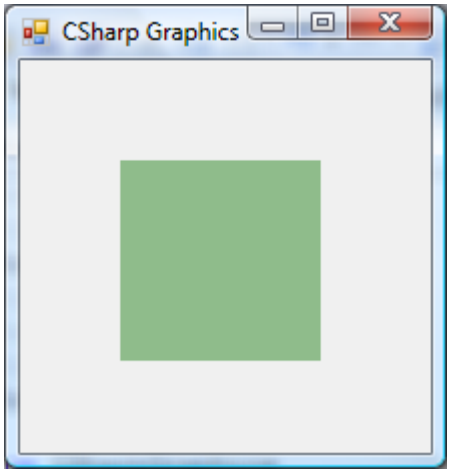
Brush Class



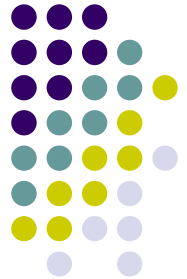
- Used for painting graphical objects.

```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;
    Rectangle r = new Rectangle(50,50,100,100);

    SolidBrush myBrush = new SolidBrush(Color.DarkSeaGreen);

    g.FillRectangle(myBrush, r);

}
```



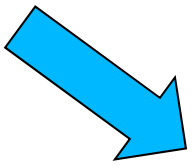


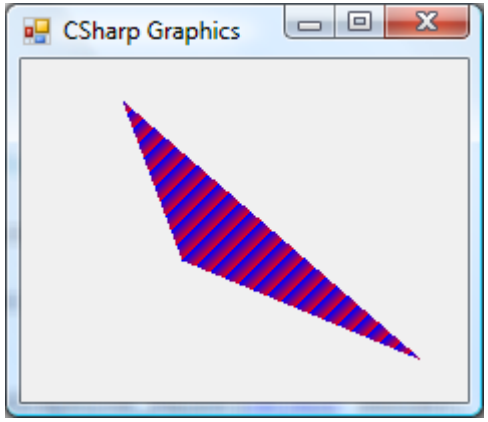
LinearGradientBrush



- Used for drawing object in desired pattern.

```
private void frmCSharpGraphics_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Point[] myPolygon = {
                            new Point(50, 20),
                            new Point(80, 100),
                            new Point(200, 150),

                        };


    LinearGradientBrush myBrush = new LinearGradientBrush(new Point(0, 0), new Point(5, 5), Color.Blue, Color.Red);

    g.FillPolygon(myBrush, myPolygon);

}
```





Exercise

- Write a drawing program that is used with mouse movements.
- Let there be an option of picking color.
- Let there be an option of picking line width.