Course: EGDF20
Module: EGE202 Application Programming

Practical 7a: Paint Application: Handling Graphics

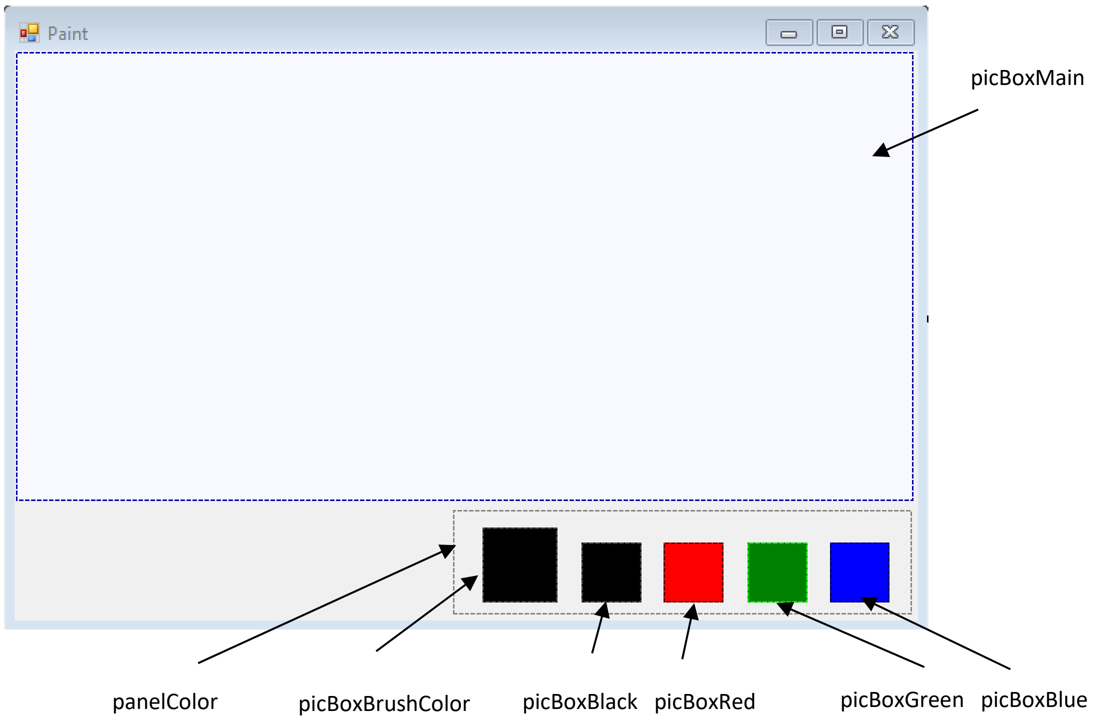
Objectives: At the end of this lab, the student should be able to describe some of the core elements and operations involved in graphics handling and advanced GUI interactions.

Exercise 1 – Develop the GUI and Event Handling for Paint Program

Part 1: Tracking and Drawing with Mouse Move Event

1. Under the ***File*** menu, click ***New Project*** or use the ***New Project*** button to create a new project. Alternatively, use the ***Create New Project*** link in the ***Get Started*** *popup* dialog.
2. From the pop-up dialog, select **"C#"** for the ***Language filter***, **"Windows"** for ***the Platform filter*** and **"Desktop"** for the ***Project type filter***.
3. Then choose ***Windows Forms App (.Net Framework)*** and click the ***Next*** button.
4. Type the name of your new project as ***Paint*** and keep the Solution name the same as Project name.
5. Set the Location to put the project in your own created folder and finally click on the ***OK*** button.
6. **Right click on "Form1.cs"** ***Solution Explorer*** window and select the Rename option to chang from ***Form1*** to ***MainForm***.
7. When prompted to rename all references to code element "Form1", **choose 'Yes'**.
8. Double click on "*MainForm.cs*" ***Solution Explorer*** window to launch the ***Form Designer*** tab.
9. Change the ***Text*** property of the ***Form*** from '*Form1*' to '*Paint*'.
10. From the ***Toolbar***, drag in 1 *Panel* and 6 *PictureBox* control into the *MainForm* window. Modify the properties based on the table below.

{Name} From	**{Name} To**	**{Text}**	**{Width}**	**{Height}**	**{BackColor}**
Form1	MainForm	Paint	620	420	🟦
pictureBox1	picBoxMain	🟦	600	300	GhostWhite
panel1	panelColor	🟦	310	70	🟦
pictureBox2	picBoxBrushColor	🟦	50	50	Black
pictureBox3	picBoxBlack	🟦	40	40	Black
pictureBox4	picBoxRed	🟦	40	40	Red
pictureBox5	picBoxGreen	🟦	40	40	Green
pictureBox6	picBoxBlue	🟦	40	40	Blue



11. Add the following code at the starting of the **MainForm** class.

```
public partial class MainForm : Form
{
    Bitmap bm;

    public MainForm()
    {
        InitializeComponent();
    }
```

12. At the ***Form Designer***, double click on Form to create a *MainForm_Load(…)* event handler. Add the following codes.

```
    private void MainForm_Load(object sender, EventArgs e)
    {
        bm = new Bitmap(picBoxMain.Width, picBoxMain.Height);
        picBoxMain.Image = bm;
    }
```

13. Next at the ***Form Designer***, **select the picBoxMain control** and then at the **properties panel** click and select the ***event*** ⚡ button. Next double click the ***MouseDown*** event to create *picBoxMain_MouseDown (…)* event handler.

14. Repeat step 13 for ***MouseMove*** event and ***MouseUp*** event

15. Add the following code at the starting of the **MainForm** class.

```
public partial class MainForm : Form
{
    Bitmap bm;

    Graphics g;
    Pen pen = new Pen(Color.Black, 5);
    SolidBrush brush = new SolidBrush(Color.Black);
    Point startP = new Point(0, 0);
    Point endP = new Point(0, 0);
    bool flagDraw = false;

    public MainForm()
    {
        InitializeComponent();
    }
```

16. Add the following code to the 3 event handlers that was just created

```
    private void picBoxMain_MouseDown(object sender, MouseEventArgs e)
    {
        startP = e.Location;
        if (e.Button == MouseButtons.Left)
            flagDraw = true;
    }

    private void picBoxMain_MouseMove(object sender, MouseEventArgs e)
    {
        if (flagDraw == true)
        {
            endP = e.Location;
            g = Graphics.FromImage(bm);
            g.FillEllipse(brush, endP.X, endP.Y, 10, 10);
            g.Dispose();
            picBoxMain.Invalidate();
        }
        startP = endP;
    }

    private void picBoxMain_MouseUp(object sender, MouseEventArgs e)
    {
        flagDraw = false;

    }
```

17. Build and test your application.

No	Actions	Observation
1	Left click on anywhere in the picBoxMain and move the mouse to draw.	What is the default color? How do I change the default color to red color?

2	Describe what happens when you comment off the codes for ***MouseUp*** event handling:
	`private void picBoxMain_MouseUp(object sender, MouseEventArgs e)`
	`{`
	`    //flagDraw = false;`
	`}`
	
	"Un-comment" the codes before proceeding to next step

3	Describe what happens when you comment off the codes for ***MouseDown*** event handling:
	`private void picBoxMain_MouseDown(object sender, MouseEventArgs e)`
	`{`
	`    //startP = e.Location;`
	`    //if (e.Button == MouseButtons.Left)`
	`    //    flagDraw = true;`
	`}`
	
	"Un-comment" the codes before proceeding to next step

4	Briefly describe how ***flagDraw*** Boolean variable is used to control the drawing process.

18. Analyze and explain the codes in the MouseMove event.

No	Actions	Explanation

In the *MouseMove* event
`if (flagDraw == true)`
`{`
`    endP = e.Location;`
`    g = Graphics.FromImage(bm);`
`    g.FillEllipse(b, endP.X, endP.Y, 10, 10);`
`    g.Dispose();`
`    picBoxMain.Invalidate();`
`}`
`startP = endP;`		

1	`endP = e.Location;`	What is the information store in ***e.Location***?

	`Graphics g;`
	
	`g = Graphics.FromImage(bm);`
		Graphics datatype represents a drawing canvas or drawing surface where application can perform drawing (lines, rectangle …)
		
		We retrieve the drawing canvass from ***Bitmap (bm)*** object

	What is the function performed by the codes below?
	`g.FillEllipse(brush, endP.X, endP.Y, 10, 10);`
	
	Replace the above codes with the following. Any changes observed?
	`g.FillRectangle(brush, endP.X, endP.Y, 10, 10);`
	
	Again Replace the above codes with the following. Any changes observed?
	`g.DrawLine(pen, startP, endP);`	

	`g.Dispose();`	Graphics object takes up memory space due to complexity in drawing objects. Hence we need to dispose the graphics object to free up memory (RAM) space
	`picBoxMain.Invalidate();`	What is the use of Invalidate()? *Hint: Try commenting it

19. On the ***Form Designer*** **double click** the **picBoxRed control** to create the *picBoxRed_Click()* event handler. Add the following codes:

```
private void picBoxRed_Click(object sender, EventArgs e)
{
    pen.Color = picBoxRed.BackColor;
    picBoxBrushColor.BackColor = pen.Color;
    picBoxBrushColor.Image = null;
}
```

(Assignment) Write the codes for Black, Green and Blue *pictureBox* control.