

**Discovering Knowledge**

**COURSE: CSL 113**

**COMPUTER PROGRAMMING**

**PROJECT REPORT**

**CLASS: BSE – 1B (FALL - 2024)**

**Project Title**

**Paint Application**

|  |  |
| --- | --- |
| **Student Name** | **Enrollment#** |
| M Waleed Ahmed | 02-131242-119 |

**Submitted to:**

Course Instructor: Engr. Muhammad Faisal

Lab Instructor: Engr. Noor us Sabah

**Department of Software Engineering**

**BAHRIA UNIVERSITY KARACHI CAMPUS**

* **PROJECT ABSTRACT:**
* Modern Paint Application is a simple desktop drawing tool created in C# and Windows Forms. Providing features such as drawing tools, shape tools, color selection, and the ability to save files. It aims to target users in search of a simple yet modern replacement for basic paint programs.
* **INTRODUCTION & BACKGROUND**
* Drawing and designing tools becomes the second nature in digital creativeness. Microsoft paint is a popular application but it is time that we are in need for an old-age lightweight drawing alternative due to basic drawing features. Using the familiar user experience with an ugly interface and simple functionality, but we are building a modern paint application using c# and windows forms.
* The creative side of this project combines programming skills with creativity to finally end up providing a tool that is intuitive for beginner users and still functional for advanced users.
* **PROBLEM STATEMENT**
* Professional painting tools are usually paid and students cannot afford it . Unpaid painting applications take much more space or storage . Moreover, these applications gets crashed while opening , take a lot time to open and also shows some runtime errors and bugs. This project seeks to address this gap by creating a modern paint application with a user-friendly design and practical functionality.
* **PROPOSED SOLUTION**
* **3.1. FEATURES OF THE PROJECT**
* The modern paint application will offer:
* Drawing tools – pencil, brush, eraser, and advanced brush styles.
* Shape tools – options for predefined shapes like rectangle, line ,circles, and square.
* Color palette – customizable color picker with rgb .
* Canvas features – resize shape functionality.
* File management – save and open files in popular formats .
* **3.2. METHODOLOGY**
* Planning – identify features and define a timeline.
* Designing – use windows forms to create an intuitive user interface.
* Coding – develop drawing logic using c#.
* Testing – test each feature rigorously to ensure performance and accuracy.
* Deployment – package the application for end-user distribution.
* **3.3. TECHNOLOGIES TO BE USED**
* Programming language: c#
* Ide: visual studio
* Libraries: windows forms
* File handling: system.io for saving and opening files
* **PROJECT SCOPE**
* **1.The project will focus on:**
* Implementing core drawing tools and canvas management features.
* Providing an intuitive interface for end-users.
* Supporting essential file operations like saving and opening files.
* **2.Exclusions:**
* Compatibility with operating systems other than windows.
* **REFERENCES**
* Visual studio
* Pinterest
* Windows forms
* **PROJECT CODE:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Drawing.Imaging;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace CP\_PAINT\_APPLICATION

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

bm = new Bitmap(pic.Width, pic.Height);

g = Graphics.FromImage(bm);

g.Clear(Color.White);

pic.Image = bm;

this.Resize += Form1\_Resize;

this.MinimumSize = new Size(950, 461);

trackBar1.Scroll += trackBar1\_Scroll;

label1.Text = $"Thickness: {thickness}";

}

Bitmap bm;

Graphics g;

bool paint = false;

Point px, py;

Pen p = new Pen(Color.Black, 2);

int index;

Pen erase = new Pen(Color.White, 10);

int x, y, sX, sY, cX, cY;

ColorDialog cd = new ColorDialog();

Color new\_color;

private Stack<Bitmap> undoStack = new Stack<Bitmap>();

private Stack<Bitmap> redoStack = new Stack<Bitmap>();

int thickness = 2;

private void btn\_rect\_Click(object sender, EventArgs e)

{

index = 4;

}

private void btn\_line\_Click(object sender, EventArgs e)

{

index = 5;

}

private void pic\_Paint(object sender, PaintEventArgs e)

{

Graphics g = e.Graphics;

if (paint)

{

if (index == 3)

{

g.DrawEllipse(p, cX, cY, sX, sY);

}

if (index == 4)

{

g.DrawRectangle(p, cX, cY, sX, sY);

}

if (index == 5)

{

g.DrawLine(p, cX, cY, x, y);

}

}

}

private void btn\_clear\_Click(object sender, EventArgs e)

{

SaveToUndoStack();

g.Clear(Color.White);

pic.Image = bm;

index = 0;

}

private void btn\_color\_Click(object sender, EventArgs e)

{

cd.ShowDialog();

new\_color = cd.Color;

pic\_color.BackColor = cd.Color;

p.Color = cd.Color;

}

private void btn\_ellipse\_Click(object sender, EventArgs e)

{

index = 3;

}

private void color\_picker\_MouseClick(object sender, MouseEventArgs e)

{

Point point = set\_point(color\_picker, e.Location);

pic\_color.BackColor = ((Bitmap)color\_picker.Image).GetPixel(point.X, point.Y);

new\_color = pic\_color.BackColor;

p.Color = pic\_color.BackColor;

}

private void pic\_MouseMove(object sender, MouseEventArgs e)

{

if (paint)

{

if (index == 1)

{

px = e.Location;

g.DrawLine(p, px, py);

py = px;

}

if (index == 2)

{

px = e.Location;

g.DrawLine(erase, px, py);

py = px;

}

}

pic.Refresh();

x = e.X; y = e.Y;

sX = e.X - cX; sY = e.Y - cY;

}

private void pic\_MouseUp(object sender, MouseEventArgs e)

{

paint = false;

sX = x - cX; sY = y - cY;

if (index == 3)

{

g.DrawEllipse(p, cX, cY, sX, sY);

}

if (index == 4)

{

g.DrawRectangle(p, cX, cY, sX, sY);

}

if (index == 5)

{

g.DrawLine(p, cX, cY, x, y);

}

}

private void pic\_MouseClick(object sender, MouseEventArgs e)

{

if (index == 7)

{

Point point = set\_point(pic, e.Location);

Fill(bm, point.X, point.Y, new\_color);

}

}

private void btn\_fill\_Click(object sender, EventArgs e)

{

index = 7;

}

private void btn\_save\_Click(object sender, EventArgs e)

{

using (var sfd = new SaveFileDialog())

{

sfd.Filter = "JPEG Image (\*.jpg)|\*.jpg|PNG Image (\*.png)|\*.png|Bitmap Image (\*.bmp)|\*.bmp|All files (\*.\*)|\*.\*";

sfd.Title = "Save an Image File";

sfd.DefaultExt = "jpg";

if (sfd.ShowDialog() == DialogResult.OK)

{

try

{

// Ensure dimensions are valid

if (pic.Width <= 0 || pic.Height <= 0 || bm.Width <= 0 || bm.Height <= 0)

{

MessageBox.Show("Invalid image dimensions. Unable to save.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

Bitmap tempBitmap = new Bitmap(pic.Width, pic.Height);

using (Graphics tempGraphics = Graphics.FromImage(tempBitmap))

{

tempGraphics.DrawImage(bm, new Rectangle(0, 0, pic.Width, pic.Height));

}

ImageFormat format = ImageFormat.Jpeg;

string extension = System.IO.Path.GetExtension(sfd.FileName).ToLower();

switch (extension)

{

case ".png":

format = ImageFormat.Png;

break;

case ".bmp":

format = ImageFormat.Bmp;

break;

case ".jpg":

case ".jpeg":

format = ImageFormat.Jpeg;

break;

default:

MessageBox.Show("Unsupported file format. Saving as JPEG.", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

format = ImageFormat.Jpeg;

break;

}

tempBitmap.Save(sfd.FileName, format);

MessageBox.Show("Image saved successfully!", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information);

tempBitmap.Dispose();

}

catch (Exception ex)

{

MessageBox.Show($"Error saving image: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

private void SaveToUndoStack()

{

undoStack.Push((Bitmap)bm.Clone());

redoStack.Clear(); // Clear redoStack whenever a new action is performed

}

private void ScalePenWidth(float scaleFactor)

{

p.Width = p.Width \* scaleFactor;

erase.Width = erase.Width \* scaleFactor;

}

private void btn\_undo\_Click(object sender, EventArgs e)

{

if (undoStack.Count > 0)

{

redoStack.Push((Bitmap)bm.Clone()); // Save the current state to redoStack

bm = undoStack.Pop(); // Restore the last state from undoStack

g = Graphics.FromImage(bm); // Reassign Graphics object

pic.Image = bm;

pic.Refresh();

}

}

private void btn\_redo\_Click(object sender, EventArgs e)

{

if (redoStack.Count > 0)

{

undoStack.Push((Bitmap)bm.Clone()); // Save the current state to undoStack

bm = redoStack.Pop(); // Restore the last state from redoStack

g = Graphics.FromImage(bm); // Reassign Graphics object

pic.Image = bm;

pic.Refresh();

}

}

private void btn\_pencil\_Click(object sender, EventArgs e)

{

index = 1;

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

thickness = trackBar1.Value; // Update thickness based on TrackBar value

label1.Text = $"Thickness: {thickness}"; // Update label to show thickness

p.Width = thickness; // Update pen thickness

erase.Width = thickness;

}

private void btn\_eraser\_Click(object sender, EventArgs e)

{

index = 2;

}

private void pic\_MouseDown(object sender, MouseEventArgs e)

{

SaveToUndoStack(); // Save the current state before starting to draw

paint = true;

py = e.Location;

cX = e.X; cY = e.Y;

}

static Point set\_point(PictureBox pb, Point pt)

{

float pX = 1f \* pb.Image.Width / pb.Width;

float pY = 1f \* pb.Image.Height / pb.Height;

return new Point((int)(pt.X \* pX), (int)(pt.Y \* pY));

}

private void validate(Bitmap bm, Stack<Point> sp, int x, int y, Color old\_color, Color new\_color)

{

Color cx = bm.GetPixel(x, y);

if (cx == old\_color)

{

sp.Push(new Point(x, y));

bm.SetPixel(x, y, new\_color);

}

}

public void Fill(Bitmap bm, int x, int y, Color new\_c1r)

{

Color old\_color = bm.GetPixel(x, y);

Stack<Point> pixel = new Stack<Point>();

pixel.Push(new Point(x, y));

bm.SetPixel(x, y, new\_c1r);

if (old\_color == new\_c1r) return;

while (pixel.Count > 0)

{

Point pt = (Point)pixel.Pop();

if (pt.X > 0 && pt.Y > 0 && pt.X < bm.Width - 1 && pt.Y < bm.Height - 1)

{

validate(bm, pixel, pt.X - 1, pt.Y, old\_color, new\_c1r);

validate(bm, pixel, pt.X, pt.Y - 1, old\_color, new\_c1r);

validate(bm, pixel, pt.X + 1, pt.Y, old\_color, new\_c1r);

validate(bm, pixel, pt.X, pt.Y + 1, old\_color, new\_c1r);

}

}

}

private void Form1\_Resize(object sender, EventArgs e)

{

float aspectRatio = (float)bm.Width / bm.Height;

int newWidth = pic.Width;

int newHeight = (int)(newWidth / aspectRatio);

if (newHeight > pic.Height)

{

newHeight = pic.Height;

newWidth = (int)(newHeight \* aspectRatio);

}

Bitmap resizedBitmap = new Bitmap(newWidth, newHeight);

using (Graphics gResized = Graphics.FromImage(resizedBitmap))

{

gResized.Clear(Color.White);

gResized.DrawImage(bm, 0, 0, newWidth, newHeight); // Maintain aspect ratio

}

bm = resizedBitmap;

g = Graphics.FromImage(bm);

pic.Image = bm;

}}}

* **OUTPUT:**
* 