Program.cs

using ShapeDrawing;

using SplashKitSDK;

using System;

namespace ShapeDrawing

{

public class Program

{

public static void Main()

{

Window window = new("Shape Drawer", 800, 600);

Drawing myDrawing = new();

do

{

SplashKit.ProcessEvents();

if (SplashKit.KeyDown(KeyCode.SpaceKey))

{

myDrawing.Background = Color.Random();

}

if (SplashKit.MouseClicked(MouseButton.LeftButton))

{

myDrawing.AddShape(new Shape((int)SplashKit.MousePosition().X, (int)SplashKit.MousePosition().Y));

}

if (SplashKit.KeyDown(KeyCode.DeleteKey) || SplashKit.KeyDown(KeyCode.BackspaceKey))

{

myDrawing.RemoveShapes();

}

if (SplashKit.MouseClicked(MouseButton.RightButton))

{

myDrawing.SelectShapeAt(SplashKit.MousePosition());

}

myDrawing.Draw();

SplashKit.RefreshScreen();

}

while (!window.CloseRequested);

}

}

}

Shape.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Reflection;

using SplashKitSDK;

using System;

namespace ShapeDrawing

{

public class Shape(int x, int y)

{

private Color \_color = Color.Green;

private float \_x = x, \_y = y;

private float \_width = 100, \_height = 100;

private bool \_selected;

public Color Color

{

get

{

return \_color;

}

set

{

\_color = value;

}

}

public float X

{

get

{

return \_x;

}

set

{

\_x = value;

}

}

public float Y

{

get

{

return \_y;

}

set

{

\_y = value;

}

}

public float Width

{

get

{

return \_width;

}

set

{

\_width = value;

}

}

public float Height

{

get

{

return \_height;

}

set

{

\_height = value;

}

}

public Shape() : this(0, 0)

{

}

public bool Selected

{

get

{

return \_selected;

}

set

{

\_selected = value;

}

}

public void Draw()

{

if (Selected)

{

DrawOutLine();

}

SplashKit.FillRectangle(\_color, \_x, \_y, \_width, \_height);

}

public bool IsAt(Point2D pt)

{

return pt.X >= \_x && pt.X < \_x + \_width && pt.Y >= \_y && pt.Y <= \_y + \_height;

}

void DrawOutLine()

{

SplashKit.DrawRectangle(Color.Black, \_x - 2, \_y - 2, \_width + 4, \_height + 4);

}

}

}

Drawing.cs

using System.Collections.Generic;

using System.Linq;

using ShapeDrawing;

using SplashKitSDK;

namespace ShapeDrawing

{

public class Drawing(Color background)

{

public Drawing() : this(Color.White)

{

}

private readonly List<Shape> \_shapes = [];

public List<Shape> SelectedShape()

{

List<Shape> \_selectedShapes = [];

foreach (Shape s in \_shapes)

{

if (s.Selected)

{

\_selectedShapes.Add(s);

}

}

return \_selectedShapes;

}

public int ShapeCount

{

get { return \_shapes.Count; }

}

private Color \_background = background;

public Color Background

{

get

{

return \_background;

}

set

{

\_background = value;

}

}

public void Draw()

{

SplashKit.ClearScreen(\_background);

foreach (Shape shape in \_shapes)

{

shape.Draw();

}

}

public void SelectShapeAt(Point2D pt)

{

foreach (Shape s in \_shapes)

{

if (s.IsAt(pt))

s.Selected = true;

else

s.Selected = false;

}

}

public void AddShape(Shape shape)

{

\_shapes.Add(shape);

}

public void RemoveShapes()

{

foreach (Shape s in \_shapes.ToList())

{

if (s.Selected)

{

\_shapes.Remove(s);

}

}

}

}

}

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Mô tả được tạo tự động

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Mô tả được tạo tự động

Ảnh có chứa ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện, văn bản

Mô tả được tạo tự động

