Swinburne University of Technology

COS20007 Object Oriented Programming

Drawing Program - A Drawing Class

PDF generated at Friday 1st September, 2023

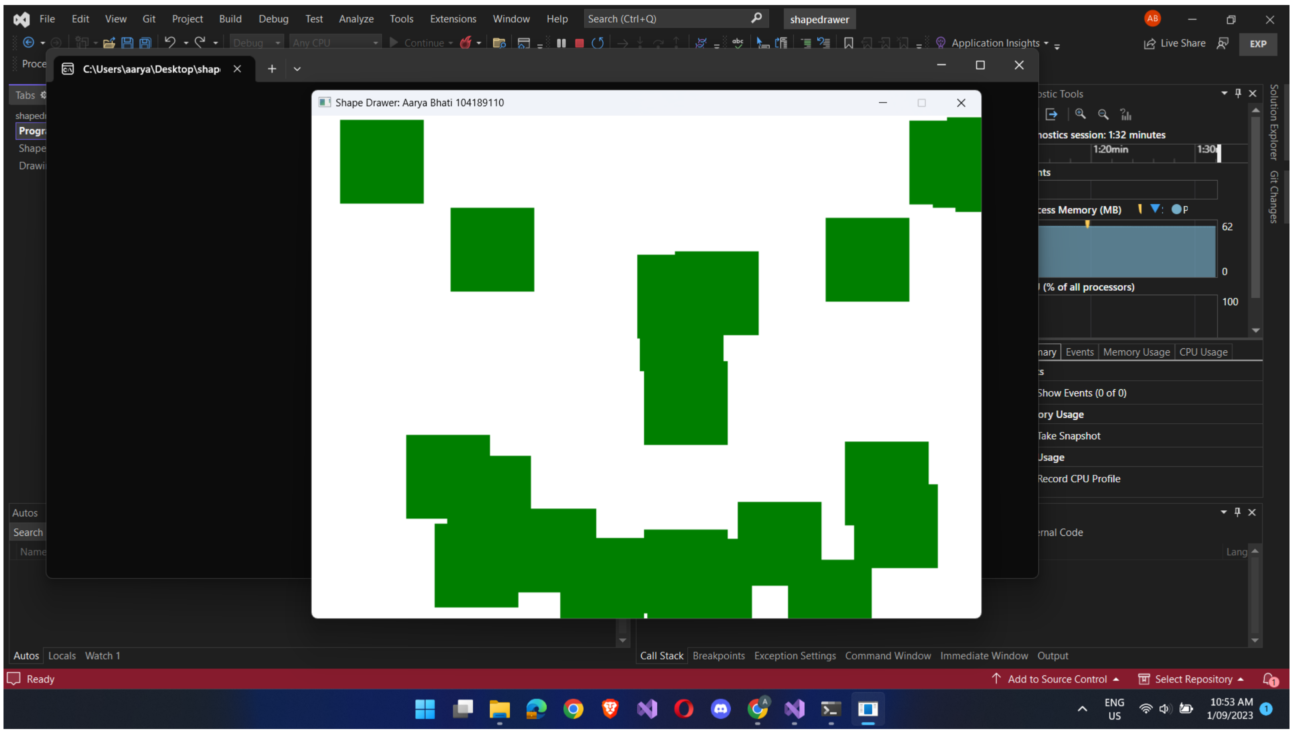
```
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Program
    {
        public static void Main()
        {

// create a window

            Window window = new Window("Shape Drawer:                          300, 600);

// create a drawing to manage the shape

            Drawing drawing = new Drawing();

// main program loop

            do
            {

// clear the screen
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();

                // check for when left key clicked

                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                 {
                    drawing.AddShape(new Shape(SplashKit.MouseX(), SplashKit.MouseY()));

                    //Shape initialShape = new Shape();(Color.Green, 0, 0, 100, 100, false);



                   // create a shape with initial values



                   // change the shape's x and y to the mouse's x and y

                    //initialShape.X = SplashKit.MouseX();
                    //initialShape.Y = SplashKit.MouseY();

                   // add the shape to the drawing

                   //drawing.AddShape(initialShape);
                   }





                // check for when space key clicked and chage the background colour

                if (SplashKit.KeyDown(KeyCode.SpaceKey))
                {
                    drawing.Background = SplashKit.RandomRGBColor(255);
                }


// check for when escape key clicked and clear the drawing

                if (SplashKit.MouseClicked(MouseButton.RightButton))
                {
                    drawing.SelectShapesAt(SplashKit.MousePosition());
                }

// check for when delete or backspace key clicked and remove the selected shapes

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

// iterate through the selected shapes and remove them from the drawing

                    foreach (Shape shape in drawing.SelectedShapes)
                    {
                        drawing.RemoveShape(shape);
                    }
                }

// draw the shapes on the screen

                drawing.Draw();

                SplashKit.RefreshScreen();
            } while (!window.CloseRequested);   // continue until the window is
 ↪  closed
        }
    }
}
```

```
using System;
using SplashKitSDK;
using System.Collections.Generic;

namespace ShapeDrawer
{

    class Drawing
    {
        private readonly List<Shape> _shapes;
        private Color _background;

// constructor for the initial drawing with background colour

        public Drawing(Color background)
        {
            _background = background;
            _shapes = new List<Shape>();
        }

// default constructor for the initial drawing with white background

        public Drawing() : this(Color.White)
        {

        }

// background colour property

        public Color Background
        {
            get { return _background; }
            set { _background = value; }
        }

// get the number of shapes in the drawing

        public int ShapeCount
        {
            get { return _shapes.Count; }
        }

// get the total area of all the shapes in the drawing

        public List<Shape> SelectedShapes
        {
            get
            {
                List<Shape> selectedShapes = new List<Shape>();

                foreach (Shape shape in _shapes)
                {
                    if (shape.Selected == true)
                    {
                        selectedShapes.Add(shape);
                    }
                }

                return selectedShapes;
            }
        }

// add a shape to the drawing

        public void AddShape(Shape shape)
        {
            _shapes.Add(shape);
        }

// draw the shapes in the drawing

        public void Draw()
        {
            SplashKit.ClearScreen(Background);

            foreach (Shape shape in _shapes)
            {
                shape.Draw();
            }
        }

//
        public void SelectShapesAt(Point2D pt)
        {
            foreach (Shape shape in _shapes)
            {
                if (shape.Selected == false)
                {
                    shape.Selected = shape.IsAt(pt);
                }
                else
                {
                    shape.Selected = !shape.IsAt(pt);
                }
            }
        }
// remove a shape from the drawing

        public void RemoveShape(Shape shape)
        {
            _shapes.Remove(shape);
        }
    }
}
```

```
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Shape
    {

        private Color _color;
        private float _x;
        private float _y;
        private int _width;
        private int _height;

        private bool _selected;


// constructor
        public Shape(float x, float y) //Color color, float x, float y, int width,
 ↪  int height, bool selected)
        {
            _color = Color.Green;
            _x = x;
            _y = y;
            _width = 100;
            _height = 100;
            _selected = false;
        }

     // properties
        // colour of the shape

    public Color Color
        {
            get { return _color; }
            set { _color = value; }
        }

// x coordinate of the top left corner of the shape
        public float X
        {
            get { return _x; }
            set { _x = value; }
        }

// y coordinate of the top left corner of the shape
        public float Y
        {
            get { return _y; }
            set { _y = value; }
        }

// width of the shape

```

```
        public int Width
        {
            get { return _width; }
            set { _width = value; }
        }

// height of the shape

        public int Height
        {
            get { return _height; }
            set { _height = value; }
        }


// whether the shape is selected or not

        public bool Selected
        {
            get { return _selected; }
            set { _selected = value; }
        }

// methods
// draw the shape on the screen

        public void Draw()
        {
            if (Selected == true)
            {
                DrawOutline();
            }
            SplashKit.FillRectangle(_color, _x, _y, _width, _height);
        }

// draw the outline of the shape on the screen

        public void DrawOutline()
        {
            SplashKit.FillRectangle(Color.Black, _x - 2, _y - 2, _width + 4, _height
↪   + 4);
        }

// determine if a point is inside the shape boundary
        public bool IsAt(Point2D pt)
        {

            if (_x < pt.X && pt.X < (_x + _width) && _y < pt.Y && pt.Y < (_y +
↪   _height))
            {
                return true;
            }
            else
```

```
                {
                    return false;
                }
            }
        }
}
```


Shape Drawer: Aarya Bhati 104189110
C:\Users\aarya\Desktop\shap
File
Edit
View
Git
Project
Build
Debug
Test
Analyze
Tools
Extensions
Window
Help
Search (Ctrl+Q)
shapedrawer
Live Share
Application Insights
Process Memory (MB)
Show Events (0 of 0)
Take Snapshot
Record CPU Profile
Autos
Locals
Watch 1
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Ready
Add to Source Control
Select Repository
10:53 AM
1/09/2023