Swinburne University of Technology

COS20007 Object Oriented Programming

Drawing Program - A Drawing Class

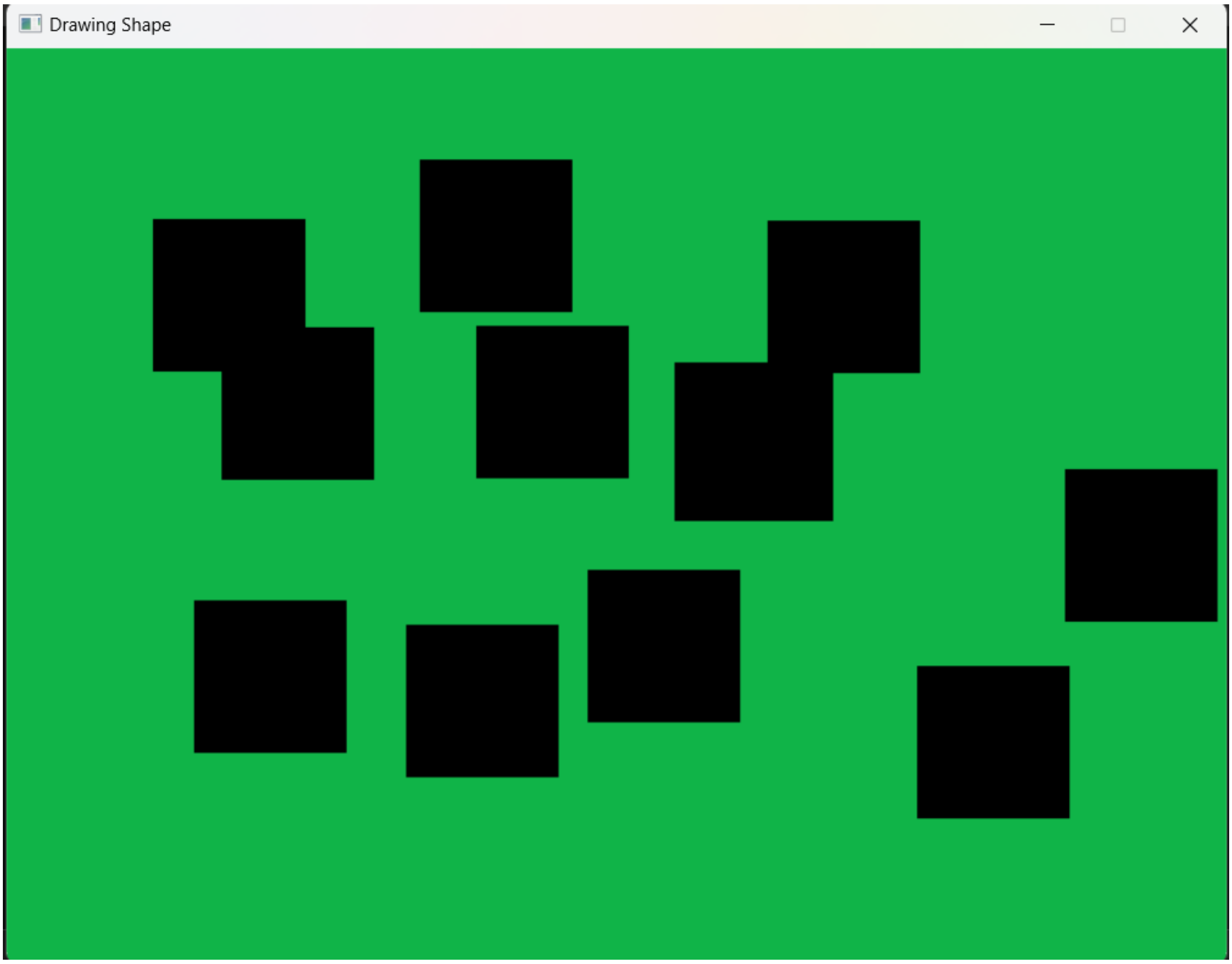
PDF generated at 13:48 on Tuesday 29th August, 2023

Produced by OOP Submission Formatter

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Program
    {
        public static void Main()
        {
            Drawing myDrawing = new Drawing();

            new Window("Drawing Shape", 800, 600);
            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();

                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    myDrawing.AddShape(new Shape());
                }

                if (SplashKit.MouseClicked(MouseButton.RightButton))
                {
                    myDrawing.SelectedShapeAt(SplashKit.MousePosition());
                }

                if (SplashKit.KeyTyped(KeyCode.BackspaceKey) || SplashKit.KeyTyped(KeyCode.DeleteKey))
                {
                    myDrawing.RemoveShape();
                }

                if (SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    myDrawing.Background = SplashKit.RandomRGBColor(255);
                }

                myDrawing.Draw();

                SplashKit.RefreshScreen();

            } while (!SplashKit.WindowCloseRequested("Drawing Shape"));
        }
    }
}
```

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

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

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

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

        public List<Shape> SelectedShapes()
        {
            List<Shape> _selectedShapes = new List<Shape>();
            foreach (Shape s in _selectedShapes)
            {
                if (s.Selected)
                {
                    _selectedShapes.Add(s);
                }
            }
            return _selectedShapes;
        }

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

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

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

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

        public void SelectedShapeAt(Point2D pt)
        {
            foreach (Shape s in _shapes)
            {
                if (s.IsAt(pt))
                {
                    s.Selected = true;
                }
                else
                {
                    s.Selected = false;
                }
            }
        }

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

        public void RemoveShape()
        {
            foreach (Shape s in _shapes.ToList())
            {
                if (s.Selected)
                {
                    _shapes.Remove(s);
                }
            }
        }
    }
}
```

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Shape
    {
        private Color _color;
        private float _x, _y;
        private int _width, _height;
        private bool _selected;

        public Shape()
        {
            _color = Color.Black;
            _x = SplashKit.MouseX();
            _y = SplashKit.MouseY();
            _width = _height = 100;
        }

        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 void Draw()
        {
            if (Selected)
            {
                DrawOutline();
            }
            SplashKit.FillRectangle(Color, X, Y, _width, _height);
        }

        public bool IsAt(Point2D p)
        {
            return SplashKit.PointInRectangle(p, SplashKit.RectangleFrom(X, Y,
 ↪  _width, _height));
        }

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

        public void DrawOutline()
        {
            SplashKit.FillRectangle(Color, X - 2, Y - 2, _width + 4, _height + 4);
        }
    }
}
```


Drawing Shape