

Drawing.cs

...C\Desktop\COS20007\Week_4\4.1P\ShapeDrawer\Drawing.cs

1

```
1  using SplashKitSDK;
2  using System.ComponentModel;
3  using System.Security.Cryptography.X509Certificates;
4
5  namespace ShapeDrawer
6  {
7      public class Drawing
8      {
9          private readonly List<Shape> _shapes;
10         private Color _background;
11
12         public Color Background
13         {
14             get => _background;
15             set => _background = value;
16         }
17
18         public Drawing(Color background)
19         {
20             _shapes = new List<Shape>();
21             _background = background;
22         }
23
24         public Drawing() : this (Color.White)
25         {
26             _shapes = new List<Shape>();
27             _background = Color.White;
28         }
29
30         public int ShapeCount
31         {
32             get => _shapes.Count;
33         }
34
35         public void AddShape(Shape shape)
36         {
37             _shapes.Add(shape);
38         }
39
40         public void RemoveShape(Shape shape)
41         {
42             _= _shapes?.Remove(shape);
43         }
44
45         public void Draw()
46         {
47             SplashKit.ClearScreen(_background);
48             for (int i = 0; i < _shapes.Count; i++)
49             {
```

```
50             if (_shapes[i].Selected)
51             {
52                 _shapes[i].DrawOutline();
53             }
54             _shapes[i].Draw();
55         }
56     }
57
58     public void SelectShapesAt(Point2D point)
59     {
60         foreach (Shape s in _shapes)
61         {
62             s.Selected = s.IsAt(point);
63         }
64     }
65
66     public List<Shape> SelectedShapes
67     {
68         get
69         {
70             List<Shape> result = new List<Shape>();
71             foreach (Shape s in _shapes)
72             {
73                 if (s.Selected)
74                 {
75                     result.Add(s);
76                 }
77             }
78             return result;
79         }
80     }
81 }
82 }
83 }
```

MyCircle.cs

..\Desktop\COS20007\Week_4\4.1P\ShapeDrawer\MyCircle.cs

1

```
1  using SplashKitSDK;
2
3  namespace ShapeDrawer
4  {
5      public class MyCircle : Shape
6      {
7          private int _radius;
8          public int Radius
9          {
10              get => _radius;
11              set => _radius = value;
12          }
13
14          public MyCircle(Color color, int radius) : base(color)
15          {
16              _radius = radius;
17          }
18
19          public MyCircle() : this (Color.Blue, 50)
20          {
21              _radius = 50;
22              Color = Color.Blue;
23          }
24
25          public override void DrawOutline()
26          {
27              SplashKit.FillCircle(Color.Black, X, Y, _radius + 2);
28          }
29
30          public override void Draw()
31          {
32              if (Selected)
33              {
34                  DrawOutline();
35              }
36
37              SplashKit.FillCircle(Color, X, Y, _radius);
38          }
39
40          public override bool IsAt(Point2D point)
41          {
42              if (point.X > X - _radius && point.X < X + _radius)
43              {
44                  if (point.Y > Y - _radius && point.Y < Y + _radius)
45                  {
46                      return true;
47                  }
48              }
49              return false;
50          }
51      }
52  }
```

...\\Desktop\\COS20007\\Week_4\\4.1P\\ShapeDrawer\\MyCircle.cs

```
50     }
51 }
52 }
53
```

MyLine.cs

...PC\Desktop\COS20007\Week_4\4.1P\ShapeDrawer\MyLine.cs

1

```
1  using SplashKitSDK;
2
3  namespace ShapeDrawer
4  {
5      public class MyLine : Shape
6      {
7          private float _endX; // The distance from X
8          private float _endY; // The distance from Y
9
10         public MyLine(Color color, float x, float y, float endX, float endY) : base(color)
11         {
12             X = x;
13             Y = y;
14             _endX = endX;
15             _endY = endY;
16         }
17
18         public MyLine() : this (Color.Red, 0.0f, 0.0f, 100.0f, 0.0f)
19         {
20             Color = Color.Red;
21             X = 0.0f;
22             Y = 0.0f;
23             _endX = 100.0f;
24             _endY = 0.0f;
25         }
26
27         public float EndX
28         {
29             get => _endX;
30             set => _endX = value;
31         }
32
33         public float EndY
34         {
35             get => _endY;
36             set => _endY = value;
37         }
38
39         public override bool IsAt(Point2D point)
40         {
41             if (point.X >= X && point.X <= X + EndX)
42             {
43                 if (point.Y >= Y - 2 && point.Y <= Y + 2)
44                 {
45                     return true;
46                 }
47             }
48             return false;
```

```
49     }
50
51     public override void Draw()
52     {
53         if (Selected)
54         {
55             DrawOutline();
56         }
57         SplashKit.DrawLine(Color, X, Y, X + EndX, Y + EndY);
58     }
59
60     public override void DrawOutline()
61     {
62         SplashKit.FillCircle(Color.Black, X, Y, 4);
63         SplashKit.FillCircle(Color.Black, X + EndX, Y + EndY, 4);
64     }
65 }
66 }
67 }
```

MyRectangle.cs

```
..\sktop\COS20007\Week_4\4.1P\ShapeDrawer\MyRectangle.cs 1
1  using SplashKitSDK;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace ShapeDrawer
9  {
10     public class MyRectangle : Shape
11     {
12         private int _width;
13         private int _height;
14
15         public MyRectangle(Color color, float x, float y, int width, int height) : base(color)
16         {
17             X = x;
18             Y = y;
19             _width = width;
20             _height = height;
21         }
22
23         public MyRectangle() : this (Color.Green, 0.0f, 0.0f, 100, 100)
24         {
25             Color = Color.Green;
26             X = 0.0f;
27             Y = 0.0f;
28             _width = 100;
29             _height = 100;
30         }
31         public int Width
32         {
33             get => _width;
34             set => _width = value;
35         }
36
37         public int Height
38         {
39             get => _height;
40             set => _height = value;
41         }
42
43         public override void DrawOutline()
44         {
45             SplashKit.FillRectangle(Color.Black, X - 2, Y - 2, _width + 4, _height + 4);
46         }
47 }
```

```
48     public override void Draw()
49     {
50         if (Selected)
51         {
52             DrawOutline();
53         }
54         SplashKit.FillRectangle(Color, X, Y, _width, _height);
55     }
56
57     public override bool IsAt(Point2D point)
58     {
59         if (point.X >= X && point.X <= X + _width)
60         {
61             if (point.Y >= Y && point.Y <= Y + _height)
62             {
63                 return true;
64             }
65         }
66         return false;
67     }
68
69 }
70 }
71 }
```

Program.cs

...C\Desktop\COS20007\Week_4\4.1P\ShapeDrawer\Program.cs

1

```
1  using System;
2  using Microsoft.VisualBasic;
3  using SplashKitSDK;
4
5  namespace ShapeDrawer
6  {
7      public class Program
8      {
9          private enum ShapeKind
10         {
11             Rectangle,
12             Circle,
13             Line
14         }
15         public static void Main()
16         {
17             ShapeKind kindToAdd = ShapeKind.Circle;
18             Window window = new Window("Shape Drawer", 800, 600);
19             Drawing myDrawing;
20
21             myDrawing = new Drawing();
22
23             do
24             {
25                 SplashKit.ProcessEvents();
26                 SplashKit.ClearScreen();
27
28                 if (SplashKit.KeyTyped(KeyCode.RKey))
29                 {
30                     kindToAdd = ShapeKind.Rectangle;
31                 }
32
33                 if (SplashKit.KeyTyped(KeyCode.CKey))
34                 {
35                     kindToAdd = ShapeKind.Circle;
36                 }
37
38                 if (SplashKit.KeyTyped(KeyCode.LKey))
39                 {
40                     kindToAdd = ShapeKind.Line;
41                 }
42
43                 if (SplashKit.MouseClicked(MouseButton.LeftButton))
44                 {
45                     Shape myShape;
46                     switch(kindToAdd)
47                     {
48                         case ShapeKind.Circle:
49                             myShape = new MyCircle();
```

```
50                     break;
51
52             case ShapeKind.Line:
53                 myShape = new MyLine();
54                 break;
55
56             default:
57                 myShape = new MyRectangle();
58                 break;
59         }
60
61         myShape.X = SplashKit.MouseX();
62         myShape.Y = SplashKit.MouseY();
63
64         myDrawing.AddShape(myShape);
65     }
66
67     Point2D myPoint = new Point2D()
68     {
69         X = SplashKit.MouseX()
70         , Y = SplashKit.MouseY()
71     };
72
73     if (SplashKit.KeyTyped(KeyCode.SpaceKey))
74     {
75         myDrawing.Background = SplashKit.RandomColor();
76     }
77
78     if (SplashKit.MouseClicked(MouseButton.RightButton))
79     {
80         myDrawing.SelectShapesAt(myPoint);
81     }
82
83     if ((SplashKit.KeyTyped(KeyCode.DeleteKey)) ||
84         (SplashKit.KeyTyped(KeyCode.BackspaceKey)))
85     {
86         foreach (Shape shape in myDrawing.SelectedShapes)
87         {
88             myDrawing.RemoveShape(shape);
89         }
90
91         myDrawing.Draw();
92
93         SplashKit.RefreshScreen();
94     }
95     while (!window.CloseRequested);
96 }
97 }
```

98 }

99

Shape.cs

..\PC\Desktop\COS20007\Week_4\4.1P\ShapeDrawer\Shape.cs

1

```
1 using SplashKitSDK;
2
3 namespace ShapeDrawer
4 {
5     public abstract class Shape
6     {
7         private Color _color;
8         private float _x;
9         private float _y;
10        private bool _selected;
11
12        public Shape(Color color)
13        {
14            _color = color;
15            _x = _y = 0.0f;
16        }
17
18        public Shape() : this (Color.Yellow)
19        {
20            _color = Color.Yellow;
21            _x = _y = 0.0f;
22        }
23
24        public bool Selected
25        {
26            get => _selected;
27            set => _selected = value;
28        }
29
30        public Color Color
31        {
32            get => _color;
33            set => _color = value;
34        }
35
36        public float X
37        {
38            get => _x;
39            set => _x = value;
40        }
41
42        public float Y
43        {
44            get => _y;
45            set => _y = value;
46        }
47
48        public abstract void Draw();
49    }
```

```
50     public abstract bool IsAt(Point2D point);
51
52     public abstract void DrawOutline();
53 }
54 }
55 }
```

Screenshot of Test Passing

The screenshot shows a Windows desktop environment with a code editor and a running application window.

Code Editor: The left side of the screen displays a code editor for a C# project named "ShapeDrawer". The current file is "Shape.cs". The code includes methods for drawing shapes like circles and rectangles. A break point is set at line 47.

```
25
26     get => _selected;
27     set => _selected = value;
28 }
29
30 public void Draw()
31 {
32     // Drawing logic here
33 }
34
35 public void Draw()
36 {
37     // Drawing logic here
38 }
39
40 public void Draw()
41 {
42     // Drawing logic here
43 }
44
45 public void Draw()
46 {
47     // Break point is set here
48 }
49
50 public void Draw()
51 {
52     // Drawing logic here
53 }
54 }
```

Output Window: The bottom-left corner shows the "Output" window with build logs. The build was successful, taking 02.782 seconds.

```
Build started at 3:50 PM...
1>----- Build started: Project: ShapeDrawer, Configuration: Debug
1>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
1>ShapeDrawer -> C:\Users\PC\Desktop\COS200\Week 4\4.1\ShapeDrawer\bin\Debug\net8.0\ShapeDrawer.dll
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ======
===== Build completed at 3:51 PM and took 02.782 seconds ======
```

Running Application: A window titled "Shape Drawer" is open in the center. It displays several drawn shapes: three blue circles arranged vertically on the left, four green squares arranged in two columns, and two red horizontal line segments on the right.

Solution Explorer: The right side of the screen shows the "Solution Explorer" pane, which lists the files in the "ShapeDrawer" project: Shape.cs, Program.cs, MyLine.cs, MyCircle.cs, and MyRectangle.cs.