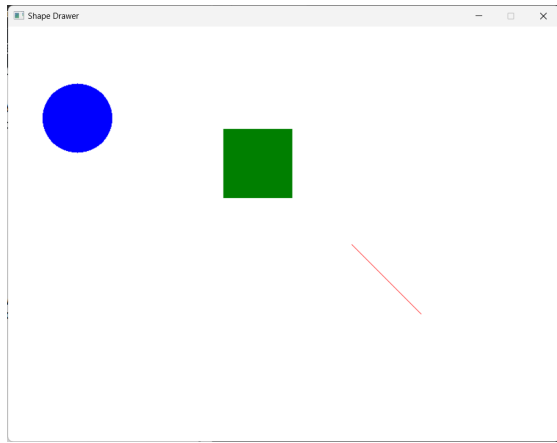


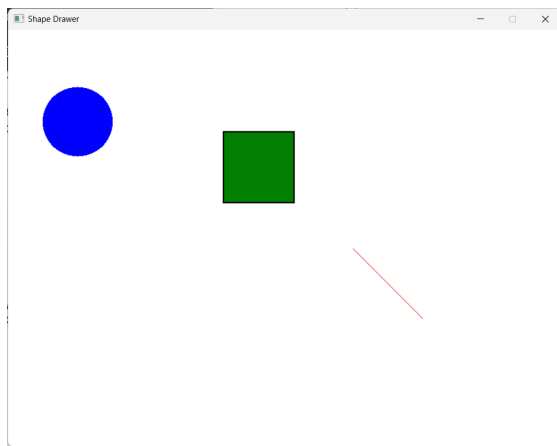
4.1: Drawing Program — Multiple Shape Kinds

Jayden Kong, 104547242

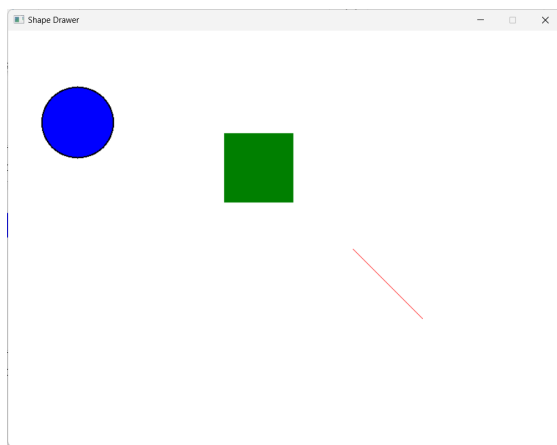
Drawing all shapes:



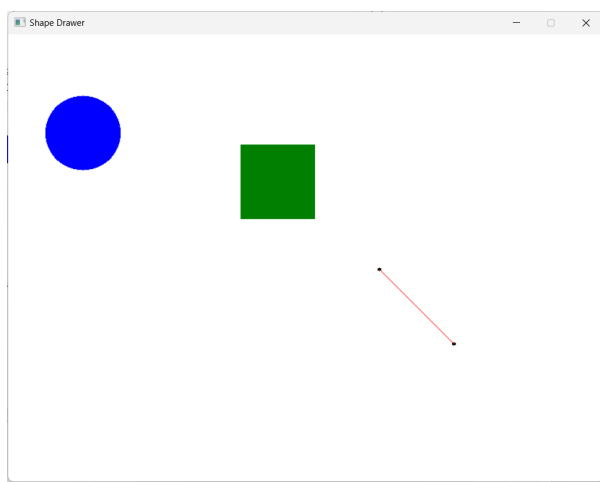
Selecting rectangle:



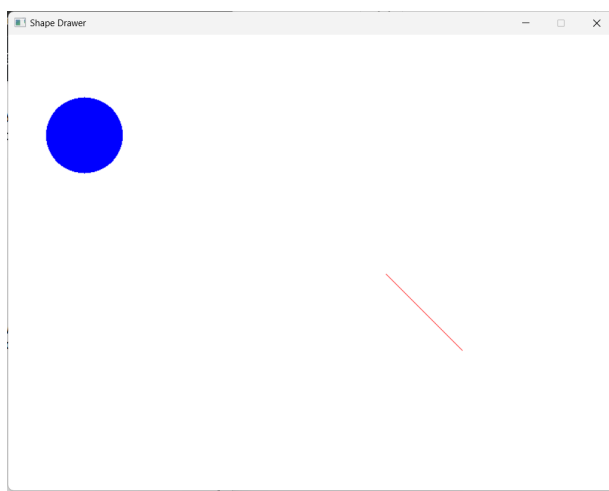
Selecting circle:



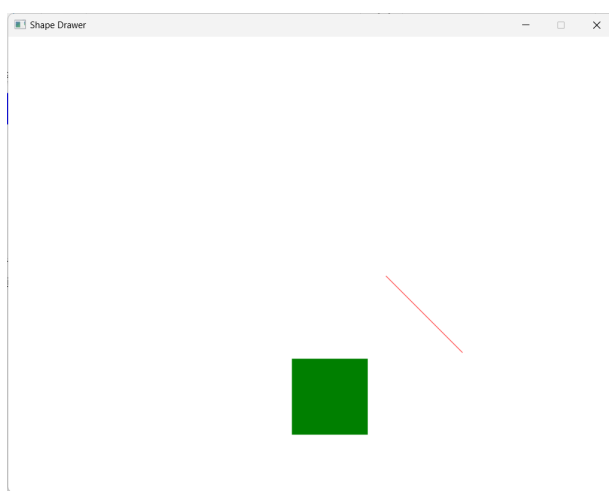
Selecting line:



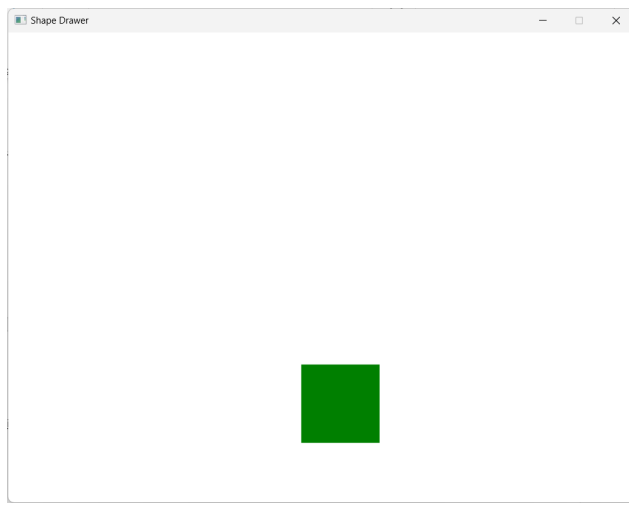
Removing rectangle:



Removing circle (added rectangle to show this is the same window):



Removing line:



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

```
50         newShape = new MyLine();
51         break;
52     default:
53         newShape = new MyRectangle();
54         break;
55     }
56
57     newShape.X = SplashKit.MouseX();
58     newShape.Y = SplashKit.MouseY();
59     myDrawing.AddShape(newShape);
60 }
61
62 if (SplashKit.KeyTyped(KeyCode.SpaceKey))
63 {
64     myDrawing.Background = SplashKit.RandomColor();
65 }
66
67 if (SplashKit.MouseClicked(MouseButton.RightButton))
68 {
69     myDrawing.SelectShapesAt(SplashKit.MousePosition());
70 }
71
72 if (SplashKit.KeyTyped(KeyCode.DeleteKey) ||
73     SplashKit.KeyTyped(KeyCode.BackspaceKey))
74 {
75     foreach (Shape s in myDrawing.SelectedShapes)
76     {
77         myDrawing.RemoveShape(s);
78     }
79 }
80
81 myDrawing.Draw();
82
83 SplashKit.RefreshScreen();
84 } while (!window.CloseRequested);
85 }
86 }
87
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using SplashKitSDK;
7
8 namespace ShapeDrawer
9 {
10     public class Drawing
11     {
12         private readonly List<Shape> _shapes;
13         private Color _background;
14
15         public Color Background
16         {
17             get
18             {
19                 return _background;
20             }
21             set
22             {
23                 _background = value;
24             }
25         }
26
27         public int ShapeCount
28         {
29             get
30             {
31                 return _shapes.Count;
32             }
33         }
34
35         public List<Shape> SelectedShapes
36         {
37             get
38             {
39                 List<Shape> result = new List<Shape>();
40                 foreach (Shape s in _shapes)
41                 {
42                     if (s.Selected)
43                     {
44                         result.Add(s);
45                     }
46                 }
47                 return result;
48             }
49         }
50     }
51 }
```

```
50
51
52     public Drawing(Color background)
53     {
54         List<Shape> shapes = new List<Shape>();
55         _shapes = shapes;
56         _background = background;
57     }
58
59     public Drawing() : this (Color.White) { }
60
61     public void AddShape(Shape s)
62     {
63         _shapes.Add(s);
64     }
65
66     public void RemoveShape(Shape s)
67     {
68         _shapes.Remove(s);
69     }
70
71     public void Draw()
72     {
73         SplashKit.ClearScreen(_background);
74         foreach (Shape s in _shapes)
75         {
76             s.Draw();
77         }
78     }
79
80     public void SelectShapesAt(Point2D pt)
81     {
82         foreach (Shape s in _shapes)
83         {
84             s.Selected = s.IsAt(pt);
85         }
86     }
87
88 }
89 }
90
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using SplashKitSDK;
7
8 namespace ShapeDrawer
9 {
10     public abstract class Shape
11     {
12         private Color _color;
13         private float _x;
14         private float _y;
15         private bool _selected;
16
17         public float X
18         {
19             get
20             {
21                 return _x;
22             }
23             set
24             {
25                 _x = value;
26             }
27         }
28
29         public float Y
30         {
31             get
32             {
33                 return _y;
34             }
35             set
36             {
37                 _y = value;
38             }
39         }
40
41         public Color Color
42         {
43             get
44             {
45                 return _color;
46             }
47             set
48             {
49                 _color = value;
```



```
50     }
51 }
52
53 public bool Selected
54 {
55     get
56     {
57         return _selected;
58     }
59     set
60     {
61         _selected = value;
62     }
63 }
64
65 public Shape() : this (Color.Yellow) { }
66
67 public Shape(Color color)
68 {
69     _color = color;
70     _x = 0.0f;
71     _y = 0.0f;
72 }
73
74 public abstract void Draw();
75
76 public abstract void DrawOutline();
77
78 public abstract bool IsAt(Point2D pt);
79 }
80 }
81
```

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

```
49     {
50         if (base.Selected)
51         {
52             DrawOutline();
53         }
54
55         SplashKit.FillRectangle(base.Color, X, Y, _width, _height);
56     }
57
58     public override void DrawOutline()
59     {
60         SplashKit.FillRectangle(Color.Black, X - 2, Y - 2, _width + 4, ↗
            _height + 4);
61     }
62
63     public override bool IsAt(Point2D pt)
64     {
65         return ((pt.X >= X) && (pt.X <= X + _width) && (pt.Y >= Y) && ↗
            (pt.Y <= Y + _height));
66     }
67
68 }
69 }
70 }
71
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using SplashKitSDK;
7
8 namespace ShapeDrawer
9 {
10     public class MyCircle : Shape
11     {
12         private int _radius;
13
14         public int Radius
15         {
16             get
17             {
18                 return _radius;
19             }
20             set
21             {
22                 _radius = value;
23             }
24         }
25         public MyCircle() : this(Color.Blue, 0.0f, 0.0f, 50) { }
26
27         public MyCircle(Color color, float x, float y, int radius) : base  ➤
            (color)
28         {
29             X = x;
30             Y = y;
31             _radius = radius;
32         }
33
34
35         public override void Draw()
36         {
37             if (Selected)
38             {
39                 DrawOutline();
40             }
41
42             SplashKit.FillCircle(base.Color, X, Y, _radius);
43         }
44
45         public override void DrawOutline()
46         {
47             SplashKit.FillCircle(Color.Black, X, Y, _radius + 2);
48         }
49     }
50 }
```

```
49
50     public override bool IsAt(Point2D pt)
51     {
52         return SplashKit.PointInCircle(pt, SplashKit.CircleAt(X, Y, ↗
            _radius));
53     }
54 }
55 }
56
```

```
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 MyLine : Shape
11     {
12         private float _endX;
13         private float _endY;
14         public float EndX
15         {
16             get
17             {
18                 return _endX;
19             }
20             set
21             {
22                 _endX = value;
23             }
24         }
25
26         public float EndY
27         {
28             get
29             {
30                 return _endY;
31             }
32             set
33             {
34                 _endY = value;
35             }
36         }
37
38         public MyLine() : this(Color.Red, 0.0f, 0.0f, 100.0f, 100.0f) { }
39
40         public MyLine(Color color, float startX, float startY, float endX,
41             float endY) : base(color)
42         {
43             X = startX;
44             Y = startY;
45             EndX = endX;
46             EndY = endY;
47         }
48
49         public override void Draw()
```

```
49     {
50         if (base.Selected)
51         {
52             DrawOutline();
53         }
54
55         SplashKit.DrawLine(base.Color, X, Y, X + EndX, Y + EndY);
56     }
57
58     public override void DrawOutline()
59     {
60         SplashKit.FillCircle(Color.Black, X, Y, 2);
61         SplashKit.FillCircle(Color.Black, X + EndX, Y + EndY, 2);
62     }
63
64     public override bool IsAt(Point2D pt)
65     {
66         return SplashKit.PointOnLine(pt, SplashKit.LineFrom(X, Y, X +
67             EndX, Y + EndY), 5);
68     }
69 }
70
```