

---

## 5.3C - Drawing Program - Saving and Loading

---

PDF generated at 12:33 on Tuesday 4<sup>th</sup> April, 2023

```
1  using System;
2  using SplashKitSDK;
3
4  namespace ShapeDrawer
5  {
6      public class Program
7      {
8          private enum ShapeKind { Rectangle, Circle, Line };
9          public static void Main()
10         {
11             Window window = new Window("Shape Drawer", 800, 600);
12             Drawing myDrawing = new Drawing(Color.White);
13             ShapeKind kindToAdd = ShapeKind.Circle;
14
15             do
16             {
17                 SplashKit.ProcessEvents();
18                 SplashKit.ClearScreen();
19
20                 if (SplashKit.MouseClicked(MouseButton.LeftButton))
21                 {
22                     Shape newShape;
23
24                     if (kindToAdd == ShapeKind.Circle)
25                     {
26                         newShape = new MyCircle();
27                     }
28                     else if (kindToAdd == ShapeKind.Rectangle)
29                     {
30                         newShape = new MyRectangle();
31                     }
32                     else
33                     {
34                         newShape = new MyLine();
35                     }
36
37                     newShape.X = SplashKit.MouseX();
38                     newShape.Y = SplashKit.MouseY();
39
40                     myDrawing.AddShape(newShape);
41                 }
42
43                 if (SplashKit.KeyDown(KeyCode.RKey))
44                 {
45                     kindToAdd = ShapeKind.Rectangle;
46                 }
47                 else if (SplashKit.KeyDown(KeyCode.CKey))
48                 {
49                     kindToAdd = ShapeKind.Circle;
50                 }
51                 else if (SplashKit.KeyDown(KeyCode.LKey))
52                 {
53                     kindToAdd = ShapeKind.Line;
```

```
54         }
55
56         if (SplashKit.KeyDown(KeyCode.SpaceKey))
57             myDrawing.Background = SplashKit.RandomRGBColor(255);
58
59         if (SplashKit.MouseClicked(MouseButton.RightButton))
60             myDrawing.SelectedShapesAt(SplashKit.MousePosition());
61
62         if ((SplashKit.KeyTyped(KeyCode.BackspaceKey)) ||
↪ (SplashKit.KeyTyped(KeyCode.DeleteKey)))
63         {
64             foreach (Shape shape in myDrawing.SelectedShapes)
65                 myDrawing.RemoveShape(shape);
66         }
67
68         if (SplashKit.KeyDown(KeyCode.SKey))
69         {
70
↪ myDrawing.Save("E:/COS20017_Object_Oriented_Programming/Portfolio/5.3C - Drawing
↪ Program - Saving and Loading/TestDrawing.txt");
71         }
72         if (SplashKit.KeyDown(KeyCode.OKey))
73         {
74             try
75             {
76
↪ myDrawing.Load("E:/COS20017_Object_Oriented_Programming/Portfolio/5.3C - Drawing
↪ Program - Saving and Loading/TestDrawing.txt");
77             }
78             catch (Exception e)
79             {
80                 Console.Error.WriteLine("Error loading file: {0}",
↪ e.Message);
81             }
82         }
83
84         myDrawing.Draw();
85
86         SplashKit.RefreshScreen();
87
88         } while (!window.CloseRequested);
89     }
90 }
91 }
```

```
1  using System;
2  using System.IO;
3  using SplashKitSDK;
4
5  namespace ShapeDrawer
6  {
7      public static class ExtensionMethods
8      {
9          public static int ReadInteger(this StreamReader reader)
10         {
11             return Convert.ToInt32(reader.ReadLine());
12         }
13
14         public static float ReadSingle(this StreamReader reader)
15         {
16             return Convert.ToSingle(reader.ReadLine());
17         }
18
19         public static Color ReadColor(this StreamReader reader)
20         {
21             return Color.RGBColor(reader.ReadSingle(), reader.ReadSingle(),
↪ reader.ReadSingle());
22         }
23
24         public static void WriteColor(this StreamWriter writer, Color clr)
25         {
26             writer.WriteLine("{0}\n{1}\n{2}", clr.R, clr.G, clr.B);
27         }
28     }
29 }
```

```
1  using SplashKitSDK;
2  using System;
3  using System.Collections.Generic;
4  using System.IO;
5
6  namespace ShapeDrawer
7  {
8      public class Drawing
9      {
10         private readonly List<Shape> _shapes;
11         private Color _background;
12
13         public Drawing(Color background)
14         {
15             _shapes = new List<Shape>();
16             _background = background;
17         }
18         public Drawing() : this(Color.White)
19         {
20         }
21
22         public List<Shape> SelectedShapes
23         {
24             get
25             {
26                 List<Shape> selectedShapes = new List<Shape>();
27                 foreach (Shape shape in _shapes)
28                 {
29                     if (shape.Selected)
30                         selectedShapes.Add(shape);
31                 }
32                 return selectedShapes;
33             }
34         }
35         public int ShapeCount
36         {
37             get
38             {
39                 return _shapes.Count;
40             }
41         }
42         public Color Background
43         {
44             get
45             {
46                 return _background;
47             }
48             set
49             {
50                 _background = value;
51             }
52         }
53     }
```

```
54     public void Draw()
55     {
56         SplashKit.ClearScreen(_background);
57
58         foreach (Shape shape in _shapes)
59             shape.Draw();
60     }
61     public void SelectedShapesAt(Point2D point)
62     {
63         foreach (Shape shape in _shapes)
64         {
65             if (shape.IsAt(point))
66                 shape.Selected = true;
67             else
68                 shape.Selected = false;
69         }
70     }
71
72     public void AddShape(Shape shape)
73     {
74         _shapes.Add(shape);
75     }
76     public void RemoveShape(Shape shape)
77     {
78         _shapes.Remove(shape);
79     }
80
81     public void Save(string filename)
82     {
83         StreamWriter writer = new StreamWriter(filename);
84         try
85         {
86             writer.WriteColor(Background);
87             writer.WriteLine(ShapeCount);
88
89             foreach (Shape shape in _shapes)
90                 shape.SaveTo(writer);
91         }
92         finally
93         {
94             writer.Close();
95         }
96     }
97
98     public void Load(string filename)
99     {
100         StreamReader reader = new StreamReader(filename);
101         try
102         {
103             Background = reader.ReadColor();
104             int count = reader.ReadInteger();
105
106             _shapes.Clear();
```

```
107
108     for (int i = 0; i < count; i++)
109     {
110         string kind = reader.ReadLine();
111         Shape s;
112         switch (kind)
113         {
114             case "Rectangle":
115                 s = new MyRectangle();
116                 break;
117             case "Circle":
118                 s = new MyCircle();
119                 break;
120             case "Line":
121                 s = new MyLine();
122                 break;
123             default:
124                 throw new InvalidDataException("Unknown shape kind: " +
↪ kind);
125         }
126         s.LoadFrom(reader);
127         AddShape(s);
128     }
129 }
130 finally
131 {
132     reader.Close();
133 }
134 }
135 }
136 }
```

```
1  using SplashKitSDK;
2  using System;
3
4  namespace ShapeDrawer
5  {
6      public abstract class Shape
7      {
8          private Color _color;
9
10         private float _x;
11         private float _y;
12
13         private bool _selected;
14         public Shape(Color color, float x, float y)
15         {
16             _color = color;
17
18             _x = x;
19             _y = y;
20         }
21         public Shape() : this(SplashKit.RandomRGBColor(255), 0, 0)
22         {
23         }
24
25         public Color Color
26         {
27             get
28             {
29                 return _color;
30             }
31             set
32             {
33                 _color = value;
34             }
35         }
36         public float X
37         {
38             get
39             {
40                 return _x;
41             }
42             set
43             {
44                 _x = value;
45             }
46         }
47         public float Y
48         {
49             get
50             {
51                 return _y;
52             }
53             set
```



```
54         {
55             _y = value;
56         }
57     }
58
59     public abstract void Draw();
60
61     public abstract bool IsAt(Point2D point2D);
62
63     public bool Selected
64     {
65         get
66         {
67             return _selected;
68         }
69         set
70         {
71             _selected = value;
72         }
73     }
74
75     public abstract void DrawOutline();
76
77     public virtual void SaveTo(StreamWriter writer)
78     {
79         writer.WriteColor(Color);
80         writer.WriteLine(X);
81         writer.WriteLine(Y);
82     }
83
84     public virtual void LoadFrom(StreamReader reader)
85     {
86         Color = reader.ReadColor();
87         X = reader.ReadSingle();
88         Y = reader.ReadSingle();
89     }
90 }
91 }
```

```
1  using SplashKitSDK;
2  using System;
3
4  namespace ShapeDrawer
5  {
6      public class MyRectangle : Shape
7      {
8          private int _width;
9          private int _height;
10
11         public MyRectangle(Color color, float x, float y, int width, int height)
12         {
13             Color = color;
14             X = x;
15             Y = y;
16             Width = width;
17             Height = height;
18         }
19         public MyRectangle() : this(Color.Green, 0, 0, 100, 100)
20         {
21         }
22
23         public int Width
24         {
25             get
26             {
27                 return _width;
28             }
29             set
30             {
31                 _width = value;
32             }
33         }
34         public int Height
35         {
36             get
37             {
38                 return _height;
39             }
40             set
41             {
42                 _height = value;
43             }
44         }
45
46         public override void Draw()
47         {
48             if (Selected)
49                 DrawOutline();
50
51             SplashKit.FillRectangle(Color, X, Y, Width, Height);
52         }
53
```

```
54     public override bool IsAt(Point2D point2D)
55     {
56         if (point2D.X >= X && point2D.X < X + Width && point2D.Y >= Y &&
↪ point2D.Y < Y + Height)
57             return true;
58         else
59             return false;
60     }
61
62     public override void DrawOutline()
63     {
64         SplashKit.FillRectangle(Color.Black, X - 2, Y - 2, Width + 4, Height +
↪ 4);
65     }
66
67     public override void SaveTo(StreamWriter writer)
68     {
69         writer.WriteLine("Rectangle");
70         base.SaveTo(writer);
71         writer.WriteLine(Width);
72         writer.WriteLine(Height);
73     }
74
75     public override void LoadFrom(StreamReader reader)
76     {
77         base.LoadFrom(reader);
78         Width = reader.ReadInteger();
79         Height = reader.ReadInteger();
80     }
81 }
82 }
```

```
1  using SplashKitSDK;
2  using System;
3
4  namespace ShapeDrawer
5  {
6      public class MyCircle : Shape
7      {
8          private int _radius;
9
10         public MyCircle(Color color, float x, float y, int radius)
11         {
12             Color = color;
13             X = x;
14             Y = y;
15             Radius = radius;
16         }
17         public MyCircle() : this(Color.Blue, 0, 0, 50)
18         {
19         }
20
21         public int Radius
22         {
23             get
24             {
25                 return _radius;
26             }
27             set
28             {
29                 _radius = value;
30             }
31         }
32         public override void Draw()
33         {
34             if (Selected)
35                 DrawOutline();
36             SplashKit.FillCircle(Color, X, Y, _radius);
37         }
38
39         public override void DrawOutline()
40         {
41             SplashKit.FillCircle(Color.Black, X, Y, _radius + 2);
42         }
43
44         public override bool IsAt(Point2D point2D)
45         {
46             float a = (float)(point2D.X - X);
47             float b = (float)(point2D.Y - Y);
48
49             if (Math.Sqrt(a * a + b * b) < _radius)
50                 return true;
51
52             return false;
53         }
54     }
```

```
54
55     public override void SaveTo(StreamWriter writer)
56     {
57         writer.WriteLine("Circle");
58         base.SaveTo(writer);
59         writer.WriteLine(Radius);
60     }
61
62     public override void LoadFrom(StreamReader reader)
63     {
64         base.LoadFrom(reader);
65         Radius = reader.ReadInteger();
66     }
67 }
68 }
```

```
1  using SplashKitSDK;
2  using System;
3
4  namespace ShapeDrawer
5  {
6      public class MyLine : Shape
7      {
8          // Endpoint(_x2, _y2)
9          private float _x2;
10         private float _y2;
11
12         public MyLine(Color color, float startX, float startY, float endX, float
↵ endY)
13         {
14             Color = color;
15             X = startX;
16             Y = startY;
17             X2 = endX;
18             Y2 = endY;
19         }
20
21         public MyLine() : this(Color.Red, 0, 0, 100, 100)
22         {
23         }
24
25         public float X2
26         {
27             get
28             {
29                 return _x2;
30             }
31             set
32             {
33                 _x2 = value;
34             }
35         }
36
37         public float Y2
38         {
39             get
40             {
41                 return _y2;
42             }
43             set
44             {
45                 _y2 = value;
46             }
47         }
48
49         public override void Draw()
50         {
51             if (Selected)
52                 DrawOutline();
```

```
53
54      // By default, the start point (X, Y) will be the coordinates of the
↪  mouse when clicking, and the end point (X2, Y2) will be (0, 0)
55      SplashKit.DrawLine(Color, X, Y, X2, Y2);
56  }
57
58  public override bool IsAt(Point2D point2D)
59  {
60      return SplashKit.PointOnLine(point2D, SplashKit.LineFrom(X, Y, X2, Y2));
61  }
62
63  public override void DrawOutline()
64  {
65      Point2D[] points = new Point2D[4];
66      points[0] = SplashKit.PointAt(X - 2, Y - 2);
67      points[1] = SplashKit.PointAt(X + 2, Y - 2);
68      points[2] = SplashKit.PointAt(X2 - 2, Y2 + 2);
69      points[3] = SplashKit.PointAt(X2 + 2, Y2 + 2);
70
71      Quad outlineQuad = new Quad();
72      outlineQuad.Points = points;
73
74      SplashKit.FillQuad(Color.Black, outlineQuad);
75  }
76
77  public override void SaveTo(StreamWriter writer)
78  {
79      writer.WriteLine("Line");
80      base.SaveTo(writer);
81      writer.WriteLine(X2);
82      writer.WriteLine(Y2);
83  }
84
85  public override void LoadFrom(StreamReader reader)
86  {
87      base.LoadFrom(reader);
88      X2 = reader.ReadSingle();
89      Y2 = reader.ReadSingle();
90  }
91  }
92 }
```

