

SWINBURNE UNIVERSITY OF TECHNOLOGY

COS20007 OBJECT ORIENTED PROGRAMMING

Drawing Program - Saving and Loading

PDF generated at 00:03 on Monday 4th September, 2023

```
1  using System;
2  using SplashKitSDK;
3  using DrawingProgram;
4
5  namespace DrawingMultipleShape
6  {
7      public class Program
8      {
9          private enum ShapeKind
10         {
11             Rectangle,
12             Circle,
13             Line
14         }
15
16         private const string fPath = "/Users/cobeo/Desktop/Codes/SchoolWork/COS20007
↪ - OOP/5.3C - Drawing Program - Saving and Loading/TestDrawing.txt";
17
18         public static void Main(string[] args)
19         {
20             Drawing drawing = new Drawing();
21             ShapeKind kindToAdd = ShapeKind.Rectangle;
22             Window window = new Window("Drawing Shape 5.3C", 800, 600);
23             while (!window.CloseRequested)
24             {
25                 SplashKit.ProcessEvents();
26                 SplashKit.ClearScreen();
27                 if (SplashKit.KeyDown(KeyCode.RKey))
28                     kindToAdd = ShapeKind.Rectangle;
29                 if (SplashKit.KeyDown(KeyCode.CKey))
30                     kindToAdd = ShapeKind.Circle;
31                 if (SplashKit.KeyDown(KeyCode.LKey))
32                     kindToAdd = ShapeKind.Line;
33                 if (SplashKit.MouseClicked(MouseButton.LeftButton))
34                 {
35                     Shape chosenShape;
36                     switch (kindToAdd)
37                     {
38                         case (ShapeKind.Rectangle):
39                             chosenShape = new MyRectangle();
40                             break;
41                         case (ShapeKind.Circle):
42                             chosenShape = new MyCircle();
43                             break;
44                         case (ShapeKind.Line):
45                             chosenShape = new MyLine();
46                             break;
47                         default:
48                             chosenShape = null!;
49                             break;
50                     }
51
52                     if (chosenShape != null)
```

```
53         {
54             chosenShape.X = SplashKit.MouseX();
55             chosenShape.Y = SplashKit.MouseY();
56             drawing.AddShape(chosenShape);
57         }
58     }
59
60     if (SplashKit.MouseClicked(MouseButton.RightButton))
61     {
62         drawing.SelectShapesAt(SplashKit.MousePosition());
63     }
64
65     if (SplashKit.KeyDown(KeyCode.SpaceKey))
66     {
67         drawing.Background = SplashKit.RandomRGBColor(255);
68     }
69
70     if (SplashKit.KeyDown(KeyCode.EscapeKey))
71     {
72         foreach (Shape s in drawing.SelectedShapes)
73         {
74             s.Color = SplashKit.RandomRGBColor(255);
75         }
76     }
77
78     if (SplashKit.KeyDown(KeyCode.DeleteKey) ||
↪ SplashKit.KeyDown(KeyCode.BackspaceKey))
79     {
80         foreach (Shape s in drawing.SelectedShapes)
81         {
82             drawing.RemoveShape(s);
83         }
84     }
85
86     if (SplashKit.KeyDown(KeyCode.SKey))
87         drawing.Save(fPath);
88     if (SplashKit.KeyDown(KeyCode.OKey))
89     {
90         try
91         {
92             drawing.Load(fPath);
93         } catch (Exception e)
94         {
95             Console.WriteLine("Error: {0}", e.Message);
96         }
97     }
98
99     drawing.Draw();
100    SplashKit.RefreshScreen();
101 }
102
103 }
104 }
```

105 }

```
1  using System;
2  using System.IO;
3  using SplashKitSDK;
4
5  namespace DrawingProgram
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 color)
25         {
26             writer.WriteLine("{0}\n{1}\n{2}", color.R, color.G, color.B);
27         }
28     }
29 }
30
```

```
1  using System;
2  using SplashKitSDK;
3  using System.Collections.Generic;
4
5  namespace DrawingProgram
6  {
7      public class Drawing
8      {
9          private readonly List<Shape> _shapes;
10         private Color _background;
11
12         public Drawing(Color background)
13         {
14             _shapes = new List<Shape>();
15             _background = background;
16         }
17
18         public Drawing() : this(Color.White) { }
19
20         public List<Shape> SelectedShapes
21         {
22             get
23             {
24                 List<Shape> _selectedShapes = new List<Shape>();
25                 foreach (Shape s in _shapes)
26                 {
27                     if (s.Selected)
28                         _selectedShapes.Add(s);
29                 }
30
31                 return _selectedShapes;
32             }
33         }
34
35         public int ShapeCount
36         {
37             get
38             {
39                 return _shapes.Count;
40             }
41         }
42
43         public Color Background
44         {
45             get
46             {
47                 return _background;
48             }
49
50             set
51             {
52                 _background = value;
53             }
54         }
55     }
```

```
54     }
55
56     public void Draw()
57     {
58         SplashKit.ClearScreen(_background);
59         foreach (Shape s in _shapes)
60         {
61             s.Draw();
62         }
63
64     }
65     public void SelectShapesAt(Point2D point)
66     {
67         foreach (Shape s in _shapes)
68         {
69             s.Selected = s.IsAt(point);
70         }
71     }
72     public void AddShape(Shape s)
73     {
74         _shapes.Add(s);
75     }
76     public void RemoveShape(Shape s)
77     {
78         _shapes.Remove(s);
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 s in _shapes)
90             {
91                 s.SaveTo(writer);
92             }
93         }
94         finally
95         {
96             writer.Close();
97         }
98     }
99
100
101     public void Load(string fileName)
102     {
103         StreamReader reader = new StreamReader(fileName);
104         try
105         {
106             int count;
```

```
107         Shape s;
108         string kind;
109
110         Background = reader.ReadColor();
111         count = reader.ReadInteger();
112         _shapes.Clear();
113         for (int i = 0; i < count; i++)
114         {
115             kind = reader.ReadLine();
116
117             switch (kind)
118             {
119                 case "Rectangle":
120                     s = new MyRectangle();
121                     break;
122                 case "Circle":
123                     s = new MyCircle();
124                     break;
125                 case "Line":
126                     s = new MyLine();
127                     break;
128                 default:
129                     throw new InvalidDataException("Unknown Shape Kind: " +
↵ kind);
130             }
131             s.LoadFrom(reader);
132             AddShape(s);
133         }
134     }
135     finally
136     {
137         reader.Close();
138     }
139 }
140 }
141 }
142
```



```
1  using System;
2  using SplashKitSDK;
3
4  namespace DrawingProgram
5  {
6      public abstract class Shape
7      {
8          private Color _color;
9          private float _x, _y;
10         private bool _selected;
11         private int _width, _height;
12
13         public Shape(Color color)
14         {
15             _color = color;
16             _x = 0;
17             _y = 0;
18             _width = 100;
19             _height = 100;
20
21         }
22
23         public Shape() : this(Color.Yellow) { }
24
25         public Color Color
26         {
27             get
28             {
29                 return _color;
30             }
31             set
32             {
33                 _color = value;
34             }
35         }
36
37         public float X
38         {
39             get
40             {
41                 return _x;
42             }
43             set
44             {
45                 _x = value;
46             }
47         }
48
49         public float Y
50         {
51             get
52             {
53                 return _y;
```

```
54         }
55         set
56         {
57             _y = value;
58         }
59     }
60
61     public int Width
62     {
63         get
64         {
65             return _width;
66         }
67         set
68         {
69             _width = value;
70         }
71     }
72
73     public int Height
74     {
75         get
76         {
77             return _height;
78         }
79         set
80         {
81             _height = value;
82         }
83     }
84
85     public bool Selected
86     {
87         get
88         {
89             return _selected;
90         }
91         set
92         {
93             _selected = value;
94         }
95     }
96
97     public abstract void Draw();
98     public abstract bool IsAt(Point2D pt);
99     public abstract void DrawOutline();
100
101     public virtual void SaveTo(StreamWriter writer)
102     {
103         writer.Write(_color);
104         writer.WriteLine(X);
105         writer.WriteLine(Y);
106     }
```

```
107
108     public virtual void LoadFrom(StreamReader reader)
109     {
110         Color = reader.ReadColor();
111         X = reader.ReadInteger();
112         Y = reader.ReadInteger();
113     }
114 }
115 }
116
```

```
1  using System;
2  using SplashKitSDK;
3
4  namespace DrawingProgram
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) :
↪     base(color)
12         {
13             X = x;
14             Y = y;
15             _width = width;
16             _height = height;
17         }
18
19         public MyRectangle() : this(Color.Green, 0, 0, 100, 100) { }
20
21         public new int Width
22         {
23             get
24             {
25                 return _width;
26             }
27             set
28             {
29                 _width = value;
30             }
31         }
32
33         public new int Height
34         {
35             get
36             {
37                 return _height;
38             }
39             set
40             {
41                 _height = value;
42             }
43         }
44
45         public override void Draw()
46         {
47             if (Selected)
48                 DrawOutline();
49             SplashKit.FillRectangle(Color, X, Y, _width, _height);
50         }
51
52         public override bool IsAt(Point2D pt)
```

```
53     {
54         return SplashKit.PointInRectangle(pt, SplashKit.RectangleFrom(X, Y,
↪ _width, _height));
55     }
56
57     public override void DrawOutline()
58     {
59         SplashKit.FillRectangle(Color.Black, X - 2, Y - 2, _width + 4, _height +
↪ 4);
60     }
61
62     public override void SaveTo(StreamWriter writer)
63     {
64         writer.WriteLine("Rectangle");
65         base.SaveTo(writer);
66         writer.WriteLine(Width);
67         writer.WriteLine(Height);
68     }
69
70     public override void LoadFrom(StreamReader reader)
71     {
72         base.LoadFrom(reader);
73         Width = reader.ReadInteger();
74         Height = reader.ReadInteger();
75     }
76 }
77 }
78
```

```
1  using System;
2  using SplashKitSDK;
3
4  namespace DrawingProgram
5  {
6      public class MyCircle : Shape
7      {
8          private int _radius;
9
10         public MyCircle(Color color, float x, float y, int radius) : base(color)
11         {
12             X = x;
13             Y = y;
14             _radius = radius;
15         }
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
33         public override void Draw()
34         {
35             if (Selected)
36             {
37                 DrawOutline();
38             }
39             SplashKit.FillCircle(Color, X, Y, _radius);
40         }
41
42         public override bool IsAt(Point2D pt)
43         {
44             //c = sqrt(a^2 + b^2)
45             return Math.Sqrt((pt.X - X) * (pt.X - X) + (pt.Y - Y) * (pt.Y - Y)) <
↪ _radius;
46         }
47
48         public override void DrawOutline()
49         {
50             SplashKit.FillCircle(Color.Black, X, Y, _radius + 4);
51         }
52
```

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

```
1  using System;
2  using SplashKitSDK;
3
4  namespace DrawingProgram
5  {
6      public class MyLine : Shape
7      {
8          private float _endX, _endY;
9
10         public MyLine(Color color, float startX, float startY, float endX, float
↵ endY) : base(color)
11         {
12             X = startX;
13             Y = startY;
14             _endX = endX;
15             _endY = endY;
16         }
17
18         public MyLine() : this(Color.RandomRGB(255), 0, 0, 10, 10)
19         {
20         }
21
22         public float EndX
23         {
24             get
25             {
26                 return _endX;
27             }
28             set
29             {
30                 _endX = value;
31             }
32         }
33
34         public float EndY
35         {
36             get
37             {
38                 return _endY;
39             }
40             set
41             {
42                 _endY = value;
43             }
44         }
45
46
47
48         public override void Draw()
49         {
50             if (Selected)
51                 DrawOutline();
52             SplashKit.DrawLine(Color, X, Y, _endX, _endY);
```



```
53     }
54
55     public override void DrawOutline()
56     {
57         SplashKit.DrawLine(Color, X, Y, _endX + 5, _endY + 5);
58     }
59
60     public override bool IsAt(Point2D pt)
61     {
62         return SplashKit.PointOnLine(pt, SplashKit.LineFrom(X, Y, _endX, _endY));
63     }
64
65     public override void SaveTo(StreamWriter writer)
66     {
67         writer.WriteLine("Line");
68         base.SaveTo(writer);
69         writer.WriteLine(EndX);
70         writer.WriteLine(EndY);
71     }
72
73     public override void LoadFrom(StreamReader reader)
74     {
75         base.LoadFrom(reader);
76         EndX = reader.ReadInteger();
77         EndY = reader.ReadInteger();
78     }
79 }
80 }
81
```

