Swinburne University of Technology

COS20007 Object Oriented Programming

# Drawing Program - Saving and Loading

PDF generated at 15:01 on Saturday 7th October, 2023

```
using System;
using System.Drawing;
using System.Runtime.CompilerServices;
using SplashKitSDK;
using System.IO;

namespace ShapeDrawer
{
    public class Program
    {
        private enum ShapeKind
        {
            Rectangle,
            Circle,
            Line,
        }

        public static void Main()
        {
            Window window = new Window("Shape Drawer", 800, 600);
            Drawing drawing = new Drawing();
            ShapeKind kindToAdd = ShapeKind.Circle;

            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();

                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    Shape newShape = null;
                    if (kindToAdd == ShapeKind.Circle)
                    {
                        MyCircle newCircle = new MyCircle();
                        newCircle.X = SplashKit.MouseX();
                        newCircle.Y = SplashKit.MouseY();
                        newShape = newCircle;
                    }
                    else if (kindToAdd == ShapeKind.Rectangle)
                    {
                        MyRectangle newRect = new MyRectangle();
                        newRect.X = SplashKit.MouseX();
                        newRect.Y = SplashKit.MouseY();
                        newShape = newRect;
                    }
                    else if (kindToAdd == ShapeKind.Line)
                    {
                        MyLine newLine = new MyLine();
                        newLine.startpoint = new Point2D()
                        {
                            X = SplashKit.MouseX(),
                            Y = SplashKit.MouseY()
                        };
```

```
54                      newShape = newLine;
55                  }
56
57                  if (newShape != null)
58                  {
59                      drawing.AddShape(newShape);
60                  }
61              }
62
63              if (SplashKit.KeyTyped(KeyCode.SpaceKey))
64              {
65                  drawing.Background = SplashKit.RandomRGBColor(255);
66              }
67
68              if (SplashKit.KeyTyped(KeyCode.RKey))
69              {
70                  kindToAdd = ShapeKind.Rectangle;
71              }
72
73              if (SplashKit.KeyTyped(KeyCode.CKey))
74              {
75                  kindToAdd = ShapeKind.Circle;
76              }
77
78              if (SplashKit.KeyTyped(KeyCode.LKey))
79              {
80                  kindToAdd = ShapeKind.Line;
81              }
82
83              if (SplashKit.MouseClicked(MouseButton.RightButton))
84              {
85                  float x = SplashKit.MouseX();
86                  float y = SplashKit.MouseY();
87                  Point2D mouseposition = new Point2D()
88                  {
89                      X = x,
90                      Y = y
91                  };
92                  drawing.SelectShapeAt(mouseposition);
93              }
94
95              if (SplashKit.KeyTyped(KeyCode.SKey))
96              {
97                  drawing.Save("/Users/vufanity/Desktop/TestDrawing.txt");
98                  Console.WriteLine("File saved at
    ↪ /Users/vufanity/Desktop/TestDrawing.txt");
99              }
100             drawing.Draw();
101             SplashKit.RefreshScreen();
102
103             if (SplashKit.KeyTyped(KeyCode.OKey))
104             {
105                 try
```

```
106                     {
107                         drawing.Load("/Users/vufanity/Desktop/TestDrawing.txt");
108                         Console.WriteLine("File loaded from
    ↪ /Users/vufanity/Desktop/TestDrawing.txt");
109                     }
110                     catch (Exception e)
111                     {
112                         Console.Error.WriteLine("Error loading file: {0}",
    ↪ e.Message);
113                     }
114                 }
115 
116                 drawing.Draw();
117                 SplashKit.RefreshScreen();
118 
119             } while (!window.CloseRequested);
120         }
121     }
122 }
```

```
using System;
using System.IO;
using SplashKitSDK;

namespace ShapeDrawer
{
    public static class ExtensionMethods
    {
        public static int ReadInteger(this StreamReader reader)
        {
            return Convert.ToInt32(reader.ReadLine());
        }
        public static float ReadSingle(this StreamReader reader)
        {
            return Convert.ToSingle(reader.ReadLine());
        }
        public static Color ReadColor(this StreamReader reader)
        {
            return Color.RGBColor(reader.ReadSingle(), reader.ReadSingle(),
            reader.ReadSingle());
        }
        public static void WriteColor(this StreamWriter writer, Color clr)
        {
            writer.WriteLine("{0}\n{1}\n{2}", clr.R, clr.G, clr.B);
        }
    }
}
```

```
using System.Collections.Generic;
using System.Drawing;
using System.IO;
using SplashKitSDK;
using Color = SplashKitSDK.Color;

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

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

        public List<Shape> SelectedShapes
        {
            get
            {
                List<Shape> result = new List<Shape>();
                foreach (Shape shape in _shapes)
                {
                    if (shape.Selected)
                    {
                        result.Add(shape);
                    }
                }
                return result;
            }
        }

        public void AddShape(Shape shape)
        {
            _shapes.Add(shape);
```

```
54          }
55
56          public void Draw()
57          {
58              SplashKit.ClearScreen(_background);
59              foreach (Shape shape in _shapes)
60              {
61                  shape.Draw();
62              }
63              SplashKit.RefreshScreen();
64          }
65
66          public void SelectShapeAt(Point2D pt)
67          {
68              foreach (Shape shape in _shapes)
69              {
70                  if (shape.IsAt(pt))
71                  {
72                      shape.Selected = true;
73                  }
74                  else
75                  {
76                      shape.Selected = false;
77                  }
78              }
79          }
80
81          public void Save(string filename)
82          {
83              using (StreamWriter writer = new StreamWriter(filename))
84              {
85                  writer.WriteColor(Background);
86                  writer.WriteLine(ShapeCount);
87                  foreach (Shape s in _shapes)
88                  {
89                      s.SaveTo(writer);
90                  }
91              }
92          }
93
94          public void Load(string filename)
95          {
96              using (StreamReader reader = new StreamReader(filename))
97              {
98                  try
99                  {
100                     Background = reader.ReadColor();
101                     int count = reader.ReadInteger();
102                     _shapes.Clear();
103
104                     for (int i = 0; i < count; i++)
105                     {
106                         string kind = reader.ReadLine();
```

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

```
using System;
using SplashKitSDK;
using System.IO;

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

        public Shape(Color color)
        {
            _color = color;
        }

        public Shape() : this(Color.Yellow) { }

        public abstract bool IsAt(Point2D pt);

        public void ChangeColor()
        {
            _color = SplashKit.RandomRGBColor(255);
        }

        public float X
        {
            get
            {
                return _x;
            }
            set
            {
                _x = value;
            }
        }

        public float Y
        {
            get
            {
                return _y;
            }
            set
            {
                _y = value;
            }
        }

        public bool Selected
        {
```

```
54              get { return _selected; }
55              set { _selected = value; }
56          }
57
58          public abstract void Draw();
59          public abstract void Drawoutline();
60          public virtual void SaveTo(StreamWriter writer)
61          {
62              writer.WriteColor(_color);
63              writer.WriteLine(X);
64              writer.WriteLine(Y);
65          }
66          public virtual void LoadFrom(StreamReader reader)
67          {
68              _color = reader.ReadColor();
69              X = reader.ReadInteger();
70              Y = reader.ReadInteger();
71          }
72      }
73  }
```

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

namespace ShapeDrawer
{
    public class MyRectangle : Shape
    {
        private float _width;
        private float _height;

        public float Width
        {
            get { return _width; }
            set { _width = value; }
        }

        public float Height
        {
            get { return _height; }
            set { _height = value; }
        }

        public float X { get; set; }
        public float Y { get; set; }

        public MyRectangle(Color _color, float x, float y, int width, int height) :
    ↪  base(_color)
        {
            X = x;
            Y = y;
            Width = width;
            Height = height;
        }

        public MyRectangle() : this(Color.Green, 0, 0, 100, 100)
        {
        }

        public override void Draw()
        {
            if (_selected)
            {
                Drawoutline();
            }
            SplashKit.FillRectangle(_color, X, Y, Width, Height);
        }

        public override void Drawoutline()
        {
            float outlineX = X - 2;
            float outlineY = Y - 2;
```

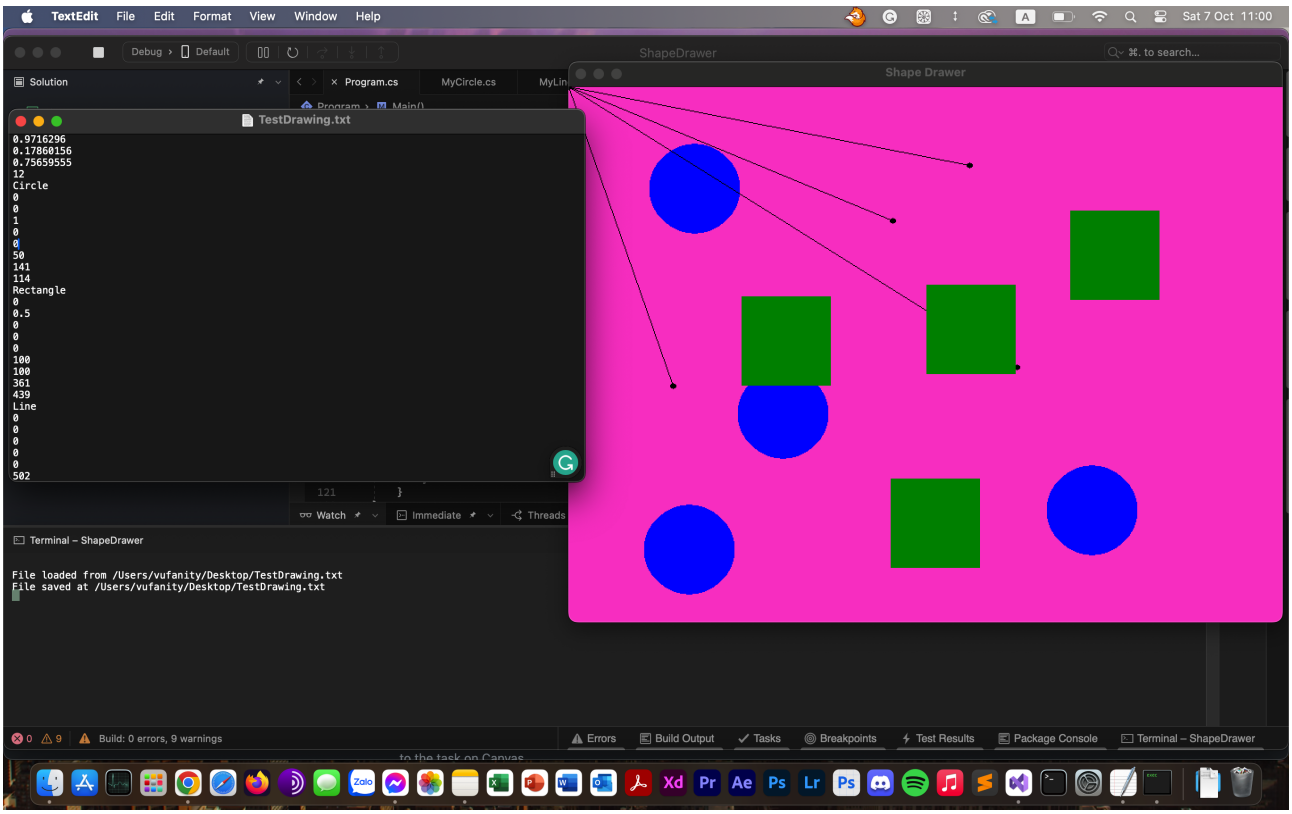
```
53              float outlineWidth = Width + 4;
54              float outlineHeight = Height + 4;
55              SplashKit.DrawRectangle(Color.Black, outlineX, outlineY, outlineWidth,
    ↪  outlineHeight);
56          }
57
58          public override bool IsAt(Point2D pt)
59          {
60     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

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

namespace ShapeDrawer
{
    public class MyCircle : Shape
    {
        public float X { get; set; }
        public float Y { get; set; }
        public float _radius { get; set; }

        public MyCircle(Color _color, float x, float y, float radius) : base(_color)
        {
            X = x;
            Y = y;
            _radius = radius;
        }

        public MyCircle() : this(Color.Blue, 0, 0, 50)
        {
        }

        public override void Draw()
        {
            if (_selected)
            {
                Drawoutline();
            }
            SplashKit.FillCircle(_color, X, Y, _radius);
        }

        public override void Drawoutline()
        {
            float outlineX = X - _radius;
            float outlineY = Y - _radius;
            float outlineDiameter = _radius * 2;
            SplashKit.DrawCircle(Color.Black, X, Y, outlineDiameter);
        }

        public override bool IsAt(Point2D pt)
        {
            double distance = System.Math.Sqrt(System.Math.Pow(pt.X - X, 2) +
↪   System.Math.Pow(pt.Y - Y, 2));
            return distance <= _radius;
        }

        public override void SaveTo(StreamWriter writer)
        {
            writer.WriteLine("Circle");
            base.SaveTo(writer);
            writer.WriteLine(_radius);
            writer.WriteLine(X);
```

```
53              writer.WriteLine(Y);
54          }
55
56          public override void LoadFrom(StreamReader reader)
57          {
58              base.LoadFrom(reader);
59              _radius = reader.ReadInteger();
60              X = reader.ReadInteger();
61              Y = reader.ReadInteger();
62          }
63      }
64  }
```

```
using SplashKitSDK;
using System;
using System.Net;
using System.IO;

namespace ShapeDrawer
{
    public class MyLine : Shape
    {
        public Point2D startpoint { get; set; }
        public Point2D endpoint { get; set; }

        public MyLine(Point2D startPoint, Point2D endPoint, Color _color) :
    ↪  base(_color)
        {
            startpoint = startPoint;
            endpoint = endPoint;
        }

        public MyLine() : this(new Point2D(), new Point2D(), Color.Black)
        {
        }

        public override void Draw()
        {
            if (_selected)
            {
                Drawoutline();
            }
            SplashKit.DrawLine(_color, startpoint.X, startpoint.Y, endpoint.X,
    ↪  endpoint.Y);
            Drawoutline();
        }

        public override void Drawoutline()
        {
            const int outlineRadius = 3;
            SplashKit.FillCircle(Color.Black, startpoint.X, startpoint.Y,
    ↪  outlineRadius);
            SplashKit.FillCircle(Color.Black, endpoint.X, endpoint.Y, outlineRadius);
        }

        public override bool IsAt(Point2D pt)
        {
            const int tolerance = 2;
            Line line = SplashKit.LineFrom(startpoint, endpoint);
            return SplashKit.PointOnLine(pt, line, tolerance);
        }

        public override void SaveTo(StreamWriter writer)
        {
            writer.WriteLine("Line");
            base.SaveTo(writer);
```

```
51              writer.WriteLine(startpoint.X);
52              writer.WriteLine(startpoint.Y);
53              writer.WriteLine(endpoint.X);
54              writer.WriteLine(endpoint.Y);
55          }
56
57          public override void LoadFrom(StreamReader reader)
58          {
59              base.LoadFrom(reader);
60              startpoint = new Point2D
61              {
62                  X = reader.ReadInteger(),
63                  Y = reader.ReadInteger()
64              };
65              endpoint = new Point2D
66              {
67                  X = reader.ReadInteger(),
68                  Y = reader.ReadInteger()
69              };
70          }
71      }
72  }
```


TextEdit File Edit Format View Window Help
Sat 7 Oct 11:00
ShapeDrawer
Shape Drawer
TestDrawing.txt
0.9716296
0.17860156
0.75659555
12
Circle
0
0
1
0
0
50
141
114
Rectangle
0
0.5
0
0
0
100
100
361
439
Line
0
0
0
0
0
502
Terminal – ShapeDrawer
File loaded from /Users/vufanity/Desktop/TestDrawing.txt
File saved at /Users/vufanity/Desktop/TestDrawing.txt
Build: 0 errors, 9 warnings
Errors
Build Output
Tasks
Breakpoints
Test Results
Package Console
Terminal – ShapeDrawer