Swinburne University of Technology

COS20007 Object Oriented Programming

5.3C - Drawing Program - Saving and Loading

PDF generated at 21:46 on Monday 17th April, 2023

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

namespace DrawingProgram
{
    public class Program
    {

        private enum ShapeKind
        {
            Rectangle,
            Circle,
            Line
        }

        public static void Main()
        {
            ShapeKind kindToAdd = ShapeKind.Circle;

            Drawing drawing = new Drawing();

            Window window = new Window("Shape Drawer", 800, 600);

            //event loop
            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();

                //shape depending on key
                if (SplashKit.KeyTyped(KeyCode.RKey))
                {
                    kindToAdd = ShapeKind.Rectangle;
                }
                if (SplashKit.KeyTyped(KeyCode.CKey))
                {
                    kindToAdd = ShapeKind.Circle;
                }
                if (SplashKit.KeyTyped(KeyCode.LKey))
                {
                    kindToAdd = ShapeKind.Line;
                }

                //new shape
                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    Shape ShapeDrawn;
                    if (kindToAdd == ShapeKind.Rectangle)
                    {
                        MyRectangle myRectangle = new();
                        ShapeDrawn = myRectangle;
                    }
```

```
54                      else if (kindToAdd == ShapeKind.Circle)
55                      {
56                          MyCircle myCircle = new();
57                          ShapeDrawn = myCircle;
58                      }
59                      else
60                      {
61                          MyLine myLine = new();
62                          ShapeDrawn = myLine;
63                      }
64                      ShapeDrawn.X = SplashKit.MouseX();
65                      ShapeDrawn.Y = SplashKit.MouseY();
66                      drawing.AddShape(ShapeDrawn);
67                  }
68                  //delete
69                  if (SplashKit.KeyTyped(KeyCode.BackspaceKey) ||
    ↪ SplashKit.KeyTyped(KeyCode.DeleteKey))
70                  {
71                      foreach (Shape s in drawing.SelectedShapes())
72                      {
73                          drawing.RemoveShape(s);
74                      }
75                  }
76
77                  //select
78                  if (SplashKit.MouseClicked(MouseButton.RightButton))
79                  {
80                      drawing.SelectShapesAt(SplashKit.MousePosition());
81                  }
82
83                  //background color
84                  if (SplashKit.KeyTyped(KeyCode.SpaceKey))
85                  {
86                      drawing.Background = SplashKit.RandomRGBColor(255);
87                  }
88                  drawing.Draw();
89                  SplashKit.RefreshScreen();
90
91                  //save
92                  if (SplashKit.KeyTyped(KeyCode.SKey))
93                  {
94                      drawing.Save("TestDrawing.txt");
95                  }
96
97                  //load
98                  if (SplashKit.KeyTyped(KeyCode.OKey))
99                  {
100                     try
101                     {
102                         drawing.Load("TestDrawing.txt");
103                     }
104                     catch (Exception e)
105                     {
```

```
                        Console.Error.WriteLine($"Error loading file: {e.Message}");
                    }
                }
            } while (!window.CloseRequested);
        }
    }
}
```

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

namespace DrawingProgram
{
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;
using SplashKitSDK;
using System.Linq;
using System.Collections.Generic;

namespace DrawingProgram
{
    public class Drawing
    {
        private readonly List<Shape> _shapes;
        private Color _background;


        public Drawing(Color background)
        {
            _shapes = new List<Shape>();
            _background = background;
        }

        //default constructor
        public Drawing() : this(Color.White)
        {

        }

        //list of currently selected shapes
        public List<Shape> SelectedShapes()
        {
            List<Shape> result = new List<Shape>();
                foreach (Shape s in _shapes)
                {
                    if (s.Selected == true)
                    {
                        result.Add(s);
                    }
                }
                return result;
        }

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

        //background color
        public Color Background
        {
            get
            {
                return _background;
```

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

```
107             }
108             writer.Close();
109         }
110 
111         public void Load(string filename)
112         {
113             StreamReader reader = new StreamReader(filename);
114             try
115             {
116                 Shape s;
117                 string kind;
118 
119                 Background = reader.ReadColor();
120                 int count = reader.ReadInteger();
121 
122                 _shapes.Clear();
123 
124                 for (int i = 0; i < count; i++)
125                 {
126                     kind = reader.ReadLine();
127 
128                     switch (kind)
129                     {
130                         case "Rectangle":
131                             s = new MyRectangle();
132                             break;
133                         case "Circle":
134                             s = new MyCircle();
135                             break;
136                         case "Line":
137                             s = new MyLine();
138                             break;
139                         default:
140                             throw new InvalidDataException("Unknown shape kind: " +
    ↪  kind);
141                     }
142 
143                     s.LoadFrom(reader);
144                     AddShape(s);
145                 }
146             }
147             finally
148             {
149                 reader.Close();
150             }
151         }
152     }
153 }
```

```
using System;
using SplashKitSDK;


namespace DrawingProgram
{
    public abstract class Shape
    {
        private Color _color;
        private float _x, _y;
        private bool _selected;

        //constructor
        public Shape(Color clr)
        {
            _color = clr;
        }

        //default constructor
        public Shape() : this(Color.Yellow)
        {
        }

        //property
        public Color COLOR
        {
            get
            {
                return _color;
            }
            set
            {
                _color = value;
            }
        }

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

        public float Y
        {
            get
            {
                return _y;
```

```
            }
            set
            {
                _y = value;
            }
        }

        public bool Selected
        {
            get
            {
                return _selected;
            }
            set
            {
                _selected = value;
            }
        }

        //methods
        public abstract void Draw();
        public abstract bool IsAt(Point2D pt);
        public abstract void DrawOutLine();

        public virtual void SaveTo(StreamWriter writer)
        {
            writer.WriteColor(COLOR);
            writer.WriteLine(X);
            writer.WriteLine(Y);
        }
        public virtual void LoadFrom(StreamReader reader)
        {
            COLOR = reader.ReadColor();
            X = reader.ReadInteger();
            Y = reader.ReadInteger();
        }
    }
}
```

```
using SplashKitSDK;

namespace DrawingProgram
{
    public class MyRectangle : Shape
    {
        private int _width;
        private int _height;

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

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


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


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


        //methods
        public override void Draw()
        {
            if (Selected)
            {
```

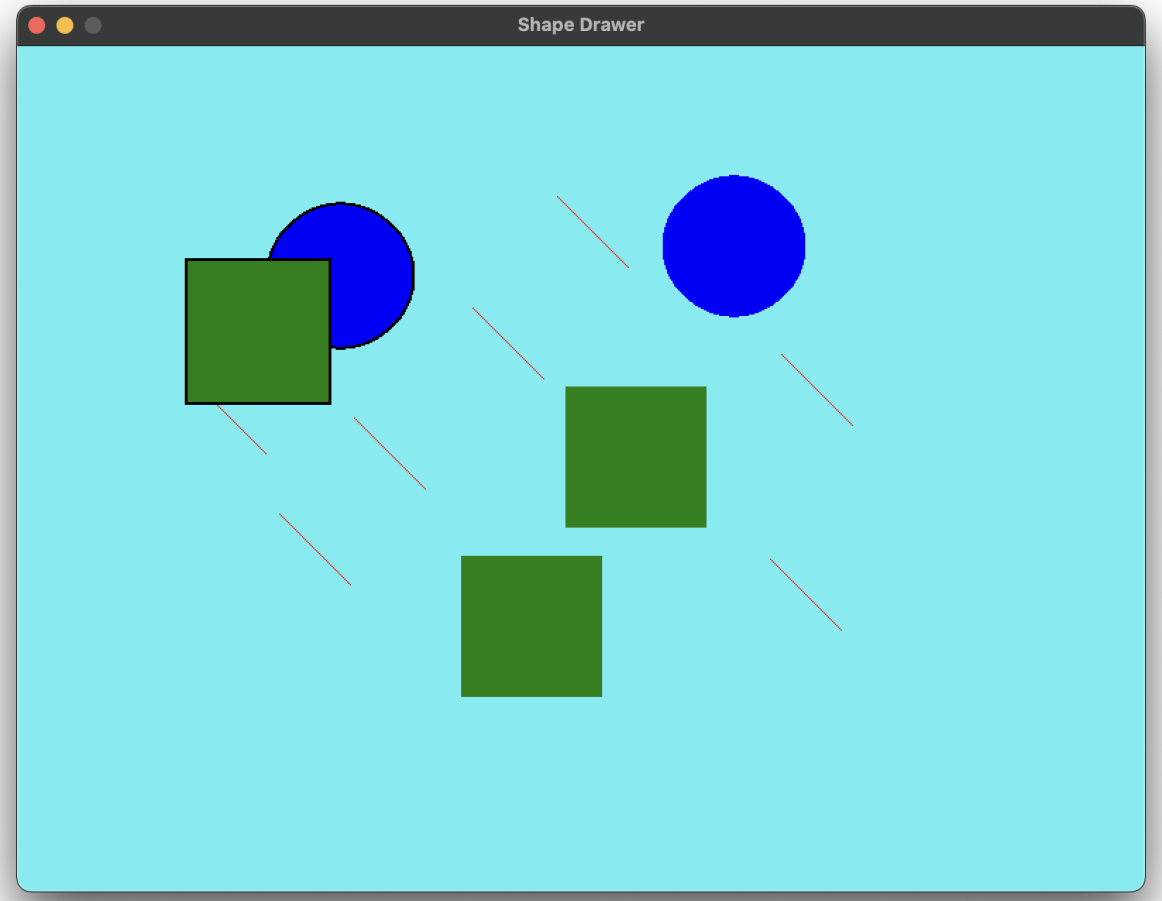
```
53                  DrawOutLine();
54              }
55              SplashKit.FillRectangle(COLOR, X, Y, Width, Height);
56          }
57
58          public override void DrawOutLine()
59          {
60              SplashKit.FillRectangle(Color.Black, X - 2, Y - 2, _width + 4, _height +
    ↪  4);
61          }
62
63          public override bool IsAt(Point2D pt)
64          {
65              return SplashKit.PointInRectangle(pt, SplashKit.RectangleFrom(X, Y,
    ↪  Width, Height));
66          }
67
68          public override void SaveTo(StreamWriter writer)
69          {
70              writer.WriteLine("Rectangle");
71              base.SaveTo(writer);
72              writer.WriteLine(Width);
73              writer.WriteLine(Height);
74          }
75
76          public override void LoadFrom(StreamReader reader)
77          {
78              base.LoadFrom(reader);
79              Width = reader.ReadInteger();
80              Height = reader.ReadInteger();
81          }
82      }
83  }
84
```

```
using System;
using SplashKitSDK;

namespace DrawingProgram
{
    public class MyCircle : Shape
    {
        //local var
        private int _radius;

        public int Radius
        {
            get
            {
                return _radius;
            }
            set
            {
                _radius = value;
            }
        }

        //constructor
        public MyCircle(Color clr, int radius) : base(clr)
        {
            _radius = radius;
        }

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

        //methods
        public override void Draw()
        {
            if (Selected)
            {
                DrawOutLine();
            }
            SplashKit.FillCircle(COLOR, X, Y, _radius);
        }

        public override void DrawOutLine()
        {
            SplashKit.FillCircle(Color.Black, X, Y, _radius + 2);
        }

        public override bool IsAt(Point2D pt)
        {
            Circle circle = new Circle()
            {
                Center = new Point2D()
                {
                    X = X,
                    Y = Y,
```

```
54                  },
55                  Radius = _radius
56              };
57              return SplashKit.PointInCircle(pt, circle);
58          }
59
60          public override void SaveTo(StreamWriter writer)
61          {
62              writer.WriteLine("Circle");
63              base.SaveTo(writer);
64              writer.WriteLine(Radius);
65          }
66
67          public override void LoadFrom(StreamReader reader)
68          {
69              base.LoadFrom(reader);
70              Radius = reader.ReadInteger();
71          }
72      }
73  }
74
```

```
using System;
using SplashKitSDK;

namespace DrawingProgram
{
    public class MyLine : Shape
    {
        private float _endX;
        private float _endY;

        public float EndX
        {
            get
            {
                return _endX;
            }
            set
            {
                _endX = value;
            }
        }

        public float EndY
        {
            get
            {
                return _endY;
            }
            set
            {
                _endY = value;
            }
        }

        public MyLine(Color clr, float endX, float endY) : base(clr)
        {
            _endX = SplashKit.MouseX() + 50;
            _endY = SplashKit.MouseY() + 50;
        }

        public MyLine() : this(Color.Red, 0, 0) { }


        public override void Draw()
        {
            if (Selected)
            {
                DrawOutLine();
            }
            SplashKit.DrawLine(COLOR, X, Y, EndX, EndY);
        }

        public override void DrawOutLine()
```

```
54          {
55              int radius = 2;
56              SplashKit.FillCircle(Color.Black, X, Y, radius);
57              SplashKit.FillCircle(Color.Black, EndX, EndY, radius);
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
```


Shape Drawer