COS20007 - Object Oriented Programming

5.3C - Drawing Program — Saving and Loading with Customized Payload

Student name: Nguyen Duc Manh
ID: 105547489

```
 1  using System;
 2  using SplashKitSDK;
 3
 4  namespace ShapeDrawer
 5  {
 6      public class Program
 7      {
 8          enum ShapeKind
 9          {
10              Rectangle,
11              Circle,
12              Line
13          }
14
15          public static void Main()
16          {
17              ////the first Shape is the type of object, which is the class
                  i've made earlier
18              //Shape myShape = new Shape(); //Shape() to call the
                  constructor
19              Window window = new Window("Shape Drawer", 800, 600);
20              Drawing myDrawing = new Drawing();
21              ShapeKind kindToAdd = ShapeKind.Circle;
22              do
23              {
24                  //8.4
25                  if (SplashKit.KeyTyped(KeyCode.RKey))
26                  {
27                      kindToAdd = ShapeKind.Rectangle;
28                  }
29                  else if (SplashKit.KeyTyped(KeyCode.CKey))
30                  {
31                      kindToAdd = ShapeKind.Circle;
32                  }
33                  else if (SplashKit.KeyTyped(KeyCode.LKey))
34                  {
35                      kindToAdd = ShapeKind.Line;
36                  }
37
38                  //earlier code
39                  SplashKit.ProcessEvents();
40                  SplashKit.ClearScreen();
41                  if (SplashKit.MouseClicked(MouseButton.LeftButton))
42                  {
43                      Shape newShape = null;
44                      var lines = myDrawing.AllShapes.OfType<MyLine>
                          ().ToList();
45                      int linesCount = lines.Count;
46                      switch (kindToAdd)
```

```
47                      {
48                          case ShapeKind.Circle:
49                              newShape = new MyCircle();
50                              break;
51                          case ShapeKind.Line:
52                              if (linesCount < 9)
53                              {
54                                  float X = SplashKit.MouseX();
55                                  float Y = SplashKit.MouseY();
56                                  newShape = new MyLine(Color.Red, X, Y,
                                      X + 100, Y);
57                              }
58                              break;
59                          default:
60                              newShape = new MyRectangle();
61                              break;
62
63                      }
64                      if (newShape != null)
65                      {
66                          newShape.X = SplashKit.MouseX();
67                          newShape.Y = SplashKit.MouseY();
68                          myDrawing.AddShape(newShape);
69                      }
70                  }
71                  if (SplashKit.KeyDown(KeyCode.SpaceKey))
72                  {
73                      myDrawing.Background = Color.RandomRGB(255);
74                  }
75                  if (SplashKit.MouseClicked(MouseButton.RightButton))
76                  {
77                      myDrawing.SelectShapeAt(SplashKit.MousePosition());
78                  }
79                  if (SplashKit.KeyDown(KeyCode.DeleteKey) ||
                        SplashKit.KeyDown(KeyCode.BackspaceKey))
80                  {
81                      foreach (Shape newShape in myDrawing.SelectedShapes)
82                      {
83                          myDrawing.RemoveShape(newShape);
84                      }
85                  }
86                  if (SplashKit.KeyTyped(KeyCode.SKey))
87                  {
88                      myDrawing.Save(@"C:\Users\Bill\Desktop\COS20007\Week 5
                          \ShapeDrawer(Week5)\TextDrawing.txt");
89                  }
90                  if (SplashKit.KeyTyped(KeyCode.OKey))
91                  {
92                      try
```

```
93                      {
94                          myDrawing.Load(@"C:\Users\Bill\Desktop\COS20007
                              \Week 5\ShapeDrawer(Week5)\TextDrawing.txt");
95                      }
96                      catch (Exception e)
97                      {
98                          Console.WriteLine("Error loading file: {0}" +
                              e.Message);
99                      }
100                 }
101                 myDrawing.Draw();
102                 SplashKit.RefreshScreen();
103             } while (!window.CloseRequested);
104         }
105     }
106 }
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Security.Cryptography.X509Certificates;
5  using System.Text;
6  using System.Threading.Tasks;
7  using SplashKitSDK;
8  using System.IO;
9
10 namespace ShapeDrawer
11 {
12     public abstract class Shape
13     {
14         //private fields
15         Color _color;
16         float _x, _y;
17         bool _selected; //bool field is "false" by default
18
19         public Shape() : this(Color.Yellow) //Constructor
20         {
21             //other steps
22         }
23
24         public Shape(Color color) //Overloaded constructor
25         {
26             _color = color;
27             _x = 10;
28             _y = 10;
29         }
30
31         //Properties
32         public Color FillColor
33         {
34             get
35             {
36                 return _color;
37             }
38             set
39             {
40                 _color = value;
41             }
42         }
43
44         public float X
45         {
46             get
47             {
48                 return _x;
49             }
```

```
50              set
51              {
52                  _x = value;
53              }
54          }
55
56          public float Y
57          {
58              get
59              {
60                  return _y;
61              }
62              set
63              {
64                  _y = value;
65              }
66          }
67
68          public bool Selected
69          {
70              get { return _selected; }
71              set { _selected = value; }
72          }
73
74          //methods
75          public abstract void Draw();
76
77          public abstract bool IsAt(Point2D pt);
78
79          public abstract void DrawOutLine();
80
81          public virtual void SaveTo(StreamWriter writer)
82          {
83              writer.WriteColor(_color);
84              writer.WriteLine(X);
85              writer.WriteLine(Y);
86              //writer.WriteLine(Selected);
87          }
88
89          public virtual void LoadFrom(StreamReader reader)
90          {
91              FillColor = reader.ReadColor();
92              X = reader.ReadInteger();
93              Y = reader.ReadInteger();
94          }
95      }
96  }
```

```
 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  using System.IO;
 8
 9  namespace ShapeDrawer
10  {
11      public class Drawing
12      {
13          readonly List<Shape> _shapes;
14          Color _background;
15
16          public Drawing(Color background)
17          {
18              _shapes = new List<Shape>();
19              _background = background;
20          }
21
22          public Drawing() : this(Color.White) //default constructor -
              initializes objs with predefined values
23          {
24              //other steps
25          }
26
27          //methods
28          public void AddShape(Shape shape)
29          {
30              _shapes.Add(shape);
31          }
32
33          public void RemoveShape(Shape shape)
34          {
35              _ = _shapes.Remove(shape); //to discard the value it returns
36          }
37
38          public void Draw()
39          {
40              SplashKit.ClearScreen(_background);
41              foreach (Shape shape in _shapes)
42              {
43                  shape.Draw();
44              }
45          }
46
47          public void SelectShapeAt(Point2D pt)
48          {
```

```
49              foreach (Shape shape in _shapes)
50              {
51                  shape.Selected = shape.IsAt(pt);
52              }
53          }
54
55          public void Save(string filename)
56          {
57              StreamWriter writer = new StreamWriter(filename);
58
59              try
60              {
61                  writer.WriteColor(_background);
62                  writer.WriteLine(_shapes.Count);
63
64                  foreach (Shape s in _shapes)
65                  {
66                      s.SaveTo(writer);
67                  }
68              }
69              finally
70              {
71                  writer.Close();
72              }
73          }
74
75          public void Load(string filename)
76          {
77              StreamReader reader = new StreamReader(filename);
78              int count;
79              string kind;
80              Shape s;
81
82              try
83              {
84                  Background = reader.ReadColor();
85                  count = reader.ReadInteger();
86
87                  _shapes.Clear(); //clear the list before loading new
                      shapes
88
89                  for (int i = 0; i < count; i++)
90                  {
91                      kind = reader.ReadLine();
92                      if (kind == "Rectangle")
93                      {
94                          s = new MyRectangle();
95                      }
96                      else if (kind == "Circle")
```

```
97                      {
98                          s = new MyCircle();
99                      }
100                     else if (kind == "Line")
101                     {
102                         s = new MyLine();
103                     }
104                     else
105                     {
106                         throw new Exception("Unknown shape type: " +
                              kind);
107                     }
108                     s.LoadFrom(reader);
109                     _shapes.Add(s);
110                 }
111             }
112             finally
113             {
114                 reader.Close();
115             }
116         }
117
118         //properties
119         public Color Background
120         {
121             get
122             {
123                 return _background;
124             }
125             set
126             {
127                 _background = value;
128             }
129         }
130
131         public int ShapeCount => _shapes.Count;
132
133         public List<Shape> SelectedShapes
134         {
135             get
136             {
137                 List<Shape> result = new List<Shape>();
138                 foreach (Shape shape in _shapes)
139                 {
140                     if (shape.Selected)
141                     {
142                         result.Add(shape);
143                     }
144                 }
```

```
145                     return result;
146                 }
147             }
148 
149             public List<Shape> AllShapes
150             {
151                 get
152                 {
153                     return _shapes;
154                 }
155             }
156         }
157 }
158 
```

```
 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 //Shape is the base class
11      {
12          int _radius = 50;
13
14          //constructor
15          public MyCircle(): this(Color.Blue, 50 + 1) //SWH02701
16          {
17              //other steps
18          }
19
20          public MyCircle(Color color, int radius) : base(color)
21          {
22              _radius = radius;
23          }
24
25          //method
26          public override void Draw()
27          {
28              if (Selected)
29              {
30                  DrawOutLine();
31              }
32              SplashKit.FillCircle(FillColor, X, Y, _radius);
33          }
34
35          public override void DrawOutLine()
36          {
37              SplashKit.DrawCircle(Color.Black, X, Y, _radius + 5);
38          }
39
40          public override bool IsAt(Point2D pt)
41          {
42              double distance = SplashKit.PointPointDistance(pt, new Point2D
                  () { X = this.X, Y = this.Y });
43              if (distance <= _radius)
44              {
45                  return true;
46              }
47              return false;
48          }
```

```
49
50          public override void SaveTo(StreamWriter writer)
51          {
52              writer.WriteLine("Circle");
53              base.SaveTo(writer);
54              writer.WriteLine(X);
55              writer.WriteLine(Y);
56              writer.WriteLine(_radius);
57          }
58
59          public override void LoadFrom(StreamReader reader)
60          {
61              base.LoadFrom(reader);
62              X = reader.ReadInteger();
63              Y = reader.ReadInteger();
64              _radius = reader.ReadInteger();
65          }
66      }
67  }
68
```

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

```
49          public override void DrawOutLine()
50          {
51              SplashKit.FillCircle(Color.Black, X, Y, 5);
52              SplashKit.FillCircle(Color.Black, _endX, _endY, 5);
53          }
54
55          public override bool IsAt(Point2D pt)
56          {
57              double distance1 = SplashKit.PointPointDistance(pt, new Point2D
                  () { X = X, Y = Y });
58              double distance2 = SplashKit.PointPointDistance(pt, new Point2D
                  () { X = _endX, Y = _endY });
59              double result = distance1 + distance2;
60              if ((int)result == 100)
61              {
62                  return true;
63              } return false;
64          }
65
66          public override void SaveTo(StreamWriter writer)
67          {
68              writer.WriteLine("Line");
69              base.SaveTo(writer);
70              writer.WriteLine(X);
71              writer.WriteLine(Y);
72              writer.WriteLine(_endX);
73              writer.WriteLine(_endY);
74          }
75
76          public override void LoadFrom(StreamReader reader)
77          {
78              base.LoadFrom(reader);
79              X = reader.ReadInteger();
80              Y = reader.ReadInteger();
81              EndX = reader.ReadInteger();
82              EndY = reader.ReadInteger();
83          }
84      }
85  }
86
```

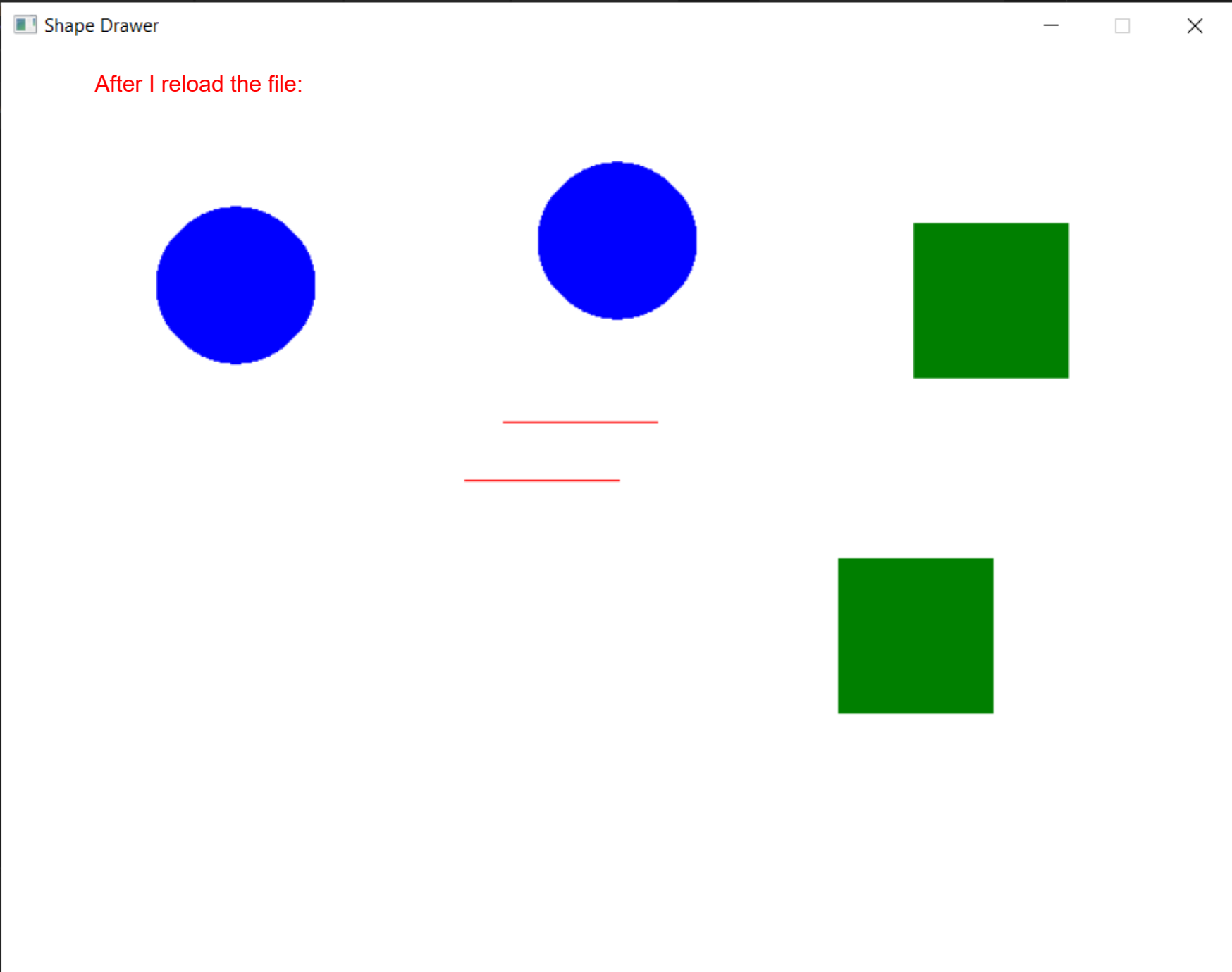
```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using SplashKitSDK;
7
8  namespace ShapeDrawer
9  {
10     public class MyRectangle : Shape //Shape is the base class
11     {
12         int _width, _height;
13
14         //constructor
15
16         public MyRectangle(): this(Color.Green, 0.0f, 0.0f, 101, 101) //
             SWH02701
17         {
18             //other steps
19         }
20
21         public MyRectangle(Color color, float x, float y, int width, int
             height) : base(color)
22         {
23             Width = width;
24             Height = height;
25             X = x; //X Y belongs to the Shape class
26             Y = y;
27         }
28
29         //method
30         public override void Draw()
31         {
32             if (Selected)
33             {
34                 DrawOutLine();
35             }
36             SplashKit.FillRectangle(FillColor, X, Y, Width, Height);
37         }
38
39         public override void DrawOutLine()
40         {
41             SplashKit.DrawRectangle(Color.Black, X - 7, Y - 7, _width +
                 14, _height + 14); //105547489
42         }
43
44         public override bool IsAt(Point2D pt)
45         {
46             return SplashKit.PointInRectangle(pt, SplashKit.RectangleFrom
```

```
                (X, Y, _width, _height));
        }

        public override void SaveTo(StreamWriter writer)
        {
            writer.WriteLine("Rectangle");
            base.SaveTo(writer); //tell the base class to not be overriden
            writer.WriteLine(_width);
            writer.WriteLine(_height);
        }

        public override void LoadFrom(StreamReader reader)
        {
            base.LoadFrom(reader);
            Width = reader.ReadInteger();
            Height = reader.ReadInteger();
        }

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

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

```

```
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         public static float ReadSingle(this StreamReader reader)
14         {
15             return Convert.ToSingle(reader.ReadLine());
16         }
17         public static Color ReadColor(this StreamReader reader)
18         {
19             return Color.RGBColor(reader.ReadSingle(), reader.ReadSingle(),
20             reader.ReadSingle());
21         }
22         public static void WriteColor(this StreamWriter writer, Color clr)
23         {
24             writer.WriteLine("{0}\n{1}\n{2}", clr.R, clr.G, clr.B);
25         }
26     }
27 }
28
```

```
1
1
1
6
Rectangle
0
0.49803922
0
593
113
101
101
Rectangle
0
0.49803922
0
544
331
101
101
Circle
0
0
1
152
153
152
153
51
Circle
0
0
1
400
124
400
124
51
Line
1
0
0
326
242
326
242
426
242
Line
```

```
50  1
51  0
52  0
53  301
54  280
55  301
56  280
57  401
58  280
59
```


Shape Drawer
After I reload the file: