

SWINBURNE UNIVERSITY OF TECHNOLOGY

COS20007 OBJECT ORIENTED PROGRAMMING

Drawing Program - Multiple Shape Kinds

PDF generated at 14:41 on Tuesday 15th August, 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         public static void Main(string[] args)
17         {
18             Drawing drawing = new Drawing();
19             ShapeKind kindToAdd = ShapeKind.Rectangle;
20             Window window = new Window("Drawing Shape 4.1P", 800, 600);
21             while(!window.CloseRequested)
22             {
23                 SplashKit.ProcessEvents();
24                 SplashKit.ClearScreen();
25                 if (SplashKit.KeyDown(KeyCode.RKey))
26                     kindToAdd = ShapeKind.Rectangle;
27                 if (SplashKit.KeyDown(KeyCode.CKey))
28                     kindToAdd = ShapeKind.Circle;
29                 if (SplashKit.KeyDown(KeyCode.LKey))
30                     kindToAdd = ShapeKind.Line;
31                 if(SplashKit.MouseDown(MouseButton.LeftButton))
32                 {
33                     switch(kindToAdd)
34                     {
35                         case(ShapeKind.Rectangle):
36                             MyRectangle rectangle = new MyRectangle();
37                             rectangle.X = SplashKit.MouseX();
38                             rectangle.Y = SplashKit.MouseY();
39                             drawing.AddShape(rectangle);
40                             break;
41                         case (ShapeKind.Circle):
42                             MyCircle circle = new MyCircle();
43                             circle.X = SplashKit.MouseX();
44                             circle.Y = SplashKit.MouseY();
45                             drawing.AddShape(circle);
46                             break;
47                         case (ShapeKind.Line):
48                             MyLine line = new MyLine();
49                             line.X = SplashKit.MouseX();
50                             line.Y = SplashKit.MouseY();
51                             drawing.AddShape(line);
52                             break;
53                     }
```

```
54         }
55     }
56
57     if(SplashKit.MouseDown(MouseButton.RightButton))
58     {
59         drawing.SelectShapesAt(SplashKit.MousePosition());
60     }
61
62     if(SplashKit.KeyDown(KeyCode.SpaceKey))
63     {
64         drawing.Background = SplashKit.RandomRGBColor(255);
65     }
66
67     if(SplashKit.KeyDown(KeyCode.EscapeKey))
68     {
69         foreach(Shape s in drawing.SelectedShapes)
70         {
71             s.color = SplashKit.RandomRGBColor(255);
72         }
73     }
74
75     if(SplashKit.KeyDown(KeyCode.DeleteKey) ||
↪ SplashKit.KeyDown(KeyCode.BackspaceKey))
76     {
77         foreach(Shape s in drawing.SelectedShapes)
78         {
79             drawing.RemoveShape(s);
80         }
81     }
82     drawing.Draw();
83     SplashKit.RefreshScreen();
84 }
85
86 }
87
88 }
```

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

```
54
55     public void Draw()
56     {
57         SplashKit.ClearScreen(_background);
58         foreach(Shape s in _shapes)
59         {
60             s.Draw();
61         }
62     }
63
64     public void SelectShapesAt(Point2D point)
65     {
66         foreach(Shape s in _shapes)
67         {
68             if (s.IsAt(point))
69                 s.Selected = true;
70             else
71                 s.Selected = false;
72         }
73     }
74     public void AddShape(Shape s)
75     {
76         _shapes.Add(s);
77     }
78     public void RemoveShape(Shape s)
79     {
80         _shapes.Remove(s);
81     }
82 }
83 }
84
```

```
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.RGBAColor(255,0, 255, 255)) { }
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 }
102
```

```
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.RandomRGB(255), 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 }
63
```

```
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.RandomRGB(255), 0, 0, 30)
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             double a = pt.X - X;
46             double b = pt.Y - Y;
47             double c = Math.Sqrt((a * a) + (b * b));
48             return c < _radius;
49         }
50
51         public override void DrawOutline()
52         {
53             SplashKit.FillCircle(Color.Black, X, Y, _radius + 4);
```

```
54         }  
55     }  
56 }  
57
```

```
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 }
66
```

