

```
1 using System;
2 using _4._1._1;
3 using SplashKitSDK;
4
5 namespace _4_1_1
6 {
7     public class Program
8     {
9         private enum ShapeKind
10        {
11            Rectangle,
12            Circle,
13            Line
14        }
15        public static void Main()
16        {
17            Window window = new Window("Shape Drawer Task 4.1", 800, 600);
18            Drawing myDraw = new Drawing();
19            ShapeKind kindToAdd = ShapeKind.Rectangle;
20
21            do
22            {
23                SplashKit.ProcessEvents();
24
25                if (SplashKit.KeyTyped(KeyCode.SpaceKey))
26                {
27                    myDraw.background = SplashKit.RandomColor();
28                }
29
30                if (SplashKit.KeyTyped(KeyCode.RKey))
31                    kindToAdd = ShapeKind.Rectangle;
32                if (SplashKit.KeyTyped(KeyCode.CKey))
33                    kindToAdd = ShapeKind.Circle;
34                if (SplashKit.KeyTyped(KeyCode.LKey))
35                    kindToAdd = ShapeKind.Line;
36
37
38                if (SplashKit.MouseClicked(MouseButton.LeftButton))
39                {
40                    Shape newShape = new MyRectangle();
41
42                    switch (kindToAdd)
43                    {
44                        case ShapeKind.Rectangle:
45                            newShape = new MyRectangle();
46                            break;
47
48                        case ShapeKind.Circle:
```

```
49         newShape = new MyCircle();
50         break;
51
52
53         case ShapeKind.Line:
54             newShape = new MyLine();
55             break;
56     }
57
58     newShape.X = SplashKit.MouseX();
59     newShape.Y = SplashKit.MouseY();
60
61
62     myDraw.AddShape(newShape);
63 }
64
65 if (SplashKit.MouseClicked(MouseButton.RightButton))
66 {
67     Point2D mousePos;
68     mousePos.X = SplashKit.MouseX();
69     mousePos.Y = SplashKit.MouseY();
70
71     myDraw.SelectShapesAt(mousePos);
72 }
73
74 if (SplashKit.KeyTyped(KeyCode.BackspaceKey) ||
75     SplashKit.KeyTyped(KeyCode.DeleteKey))
76 {
77     var selectedShapes = myDraw.SelectedShapes;
78
79     foreach (var shape in selectedShapes)
80     {
81         myDraw.RemoveShape(shape);
82     }
83 }
84
85 myDraw.Draw();
86
87
88 SplashKit.RefreshScreen();
89 } while (!window.CloseRequested);
90 }
91 }
92 }
93
```

```
1 using SplashKitSDK;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace _4._1._1
9 {
10     public abstract class Shape
11     {
12         private Color _color;
13         private float _x;
14         private float _y;
15
16         private bool _selected;
17         public Shape(Color color)
18         {
19             _color = Color.Green;
20             _x = 0.0f;
21             _y = 0.0f;
22         }
23         public Shape(): this(Color.Yellow) { }
24
25         public Color Color
26         {
27             get { return _color; }
28             set { _color = value; }
29         }
30
31         public float X
32         {
33             get { return _x; }
34             set { _x = value; }
35         }
36         public float Y
37         {
38             get { return _y; }
39             set { _y = value; }
40         }
41         public bool Selected
42         {
43             get
44             {
45                 return _selected;
46             }
47             set { _selected = value; }
48         }
49         public abstract void Draw();
```

```
50         public abstract void DrawOutline();
51         public abstract bool IsAt(Point2D pt);
52
53     }
54 }
```

```
1 using SplashKitSDK;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace _4._1._1
9 {
10     public class Drawing
11     {
12         private readonly List<Shape> _shapes;
13         private SplashKitSDK.Color _background;
14         private SplashKitSDK.Color Color;
15
16         public Drawing(SplashKitSDK.Color background)
17         {
18             _background = background;
19             _shapes = new List<Shape>();
20         }
21         public SplashKitSDK.Color background
22         {
23             get
24             {
25                 return _background;
26             }
27             set
28             {
29                 _background = value;
30             }
31         }
32         public Drawing() : this(SplashKitSDK.Color.White)
33         {
34
35         }
36         public List<Shape> SelectedShapes
37         {
38             get
39             {
40                 var result = new List<Shape>(); ;
41                 foreach (var shape in _shapes)
42                 {
43                     if (shape.Selected)
44                     {
45                         result.Add(shape);
46                     }
47                 }
48                 return result;
49             }
49         }
50     }
51 }
```

```
50     }
51     public int ShapeCount
52     {
53         get { return _shapes.Count(); }
54     }
55     public void AddShape(Shape shape)
56     {
57         _shapes.Add(shape);
58     }
59     public void RemoveShape(Shape shape)
60     {
61         _shapes.Remove(shape);
62     }
63     public void Draw()
64     {
65         SplashKit.ClearScreen(_background);
66         foreach (Shape shape in _shapes)
67         {
68             shape.Draw();
69         }
70     }
71
72     public void SelectShapesAt(Point2D pt)
73     {
74         foreach (var shape in _shapes)
75         {
76             if (shape.IsAt(pt))
77             {
78                 shape.Selected = true;
79             }
80             else { shape.Selected = false; }
81         }
82     }
83
84 }
85 }
86
87
```

```
1 using SplashKitSDK;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace _4._1._1
9 {
10     public class MyRectangle : Shape
11     {
12         private int _width;
13         private int _height;
14
15         public MyRectangle(Color color, float x, float y, int width, int height) : base(color)
16         {
17             X = x;
18             Y = y;
19             Width = width;
20             Height = height;
21         }
22         public MyRectangle() : this(Color.Green, 0.0f, 0.0f, 100, 100)
23         {
24
25         }
26
27         public int Width
28         {
29             get { return _width; }
30             set { _width = value; }
31         }
32
33         public int Height
34         {
35             get { return _height; }
36             set { _height = value; }
37         }
38         public override void Draw()
39         {
40             SplashKit.FillRectangle(Color, X, Y, Width, Height);
41             if (Selected)
42             {
43                 DrawOutline();
44             }
45         }
46
47         public override void DrawOutline()
48         {
```

```
49         SplashKit.DrawRectangle(Color.Black, X - 2, Y - 2, Width + 4, ↗
        Height + 4);
50     }
51
52     public override bool IsAt(Point2D pt)
53     {
54         return SplashKit.PointInRectangle(pt, SplashKit.RectangleFrom ↗
        (X, Y, Width, Height));
55     }
56 }
57 }
58
59
```



```
1 using SplashKitSDK;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace _4._1._1
9 {
10     public class MyCircle : Shape
11     {
12         private int _radius;
13
14         public MyCircle(Color color, int radius) : base(color)
15         {
16             Radius = radius;
17         }
18
19         public MyCircle() : this(Color.Blue, 50)
20         {
21         }
22
23         public int Radius
24         {
25             get { return _radius; }
26             set { _radius = value; }
27         }
28
29         public override void Draw()
30         {
31             if (Selected)
32             {
33                 DrawOutline();
34             }
35
36             SplashKit.FillCircle(Color, X, Y, _radius);
37         }
38
39         public override void DrawOutline()
40         {
41             SplashKit.DrawCircle(Color.Black, X, Y, Radius + 2);
42         }
43
44         public override bool IsAt(Point2D pt)
45         {
46             double distanceX = Math.Abs(pt.X - X);
47             double distanceY = Math.Abs(pt.Y - Y);
48
49             return (distanceX <= Radius) && (distanceY <= Radius);
```

```
50         }  
51  
52     }  
53 }
```

```
1 using SplashKitSDK;
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace _4._1._1
9 {
10     public class MyLine : Shape
11     {
12         private float _endX;
13         private float _endY;
14
15         public MyLine(Color color, float startX, float startY, float endX, float endY) : base(color)
16         {
17             X = startX;
18             Y = startY;
19             EndX = endX;
20             EndY = endY;
21         }
22
23         public MyLine() : this(Color.Blue, 0.0f, 0.0f, 50.0f, 20.0f)
24         {
25         }
26
27         public float EndX
28         {
29             get { return _endX; }
30             set { _endX = value; }
31         }
32
33         public float EndY
34         {
35             get { return _endY; }
36             set { _endY = value; }
37         }
38
39         public override void Draw()
40         {
41             if (Selected)
42                 DrawOutline();
43
44             SplashKit.DrawLine(Color, X, Y, X + EndX, Y + EndY);
45         }
46
47         public override void DrawOutline()
48         {
```

```
49         SplashKit.FillCircle(Color.Black, X, Y, 3);
50         SplashKit.FillCircle(Color.Black, X + EndX, Y + EndY, 3);
51     }
52
53     public override bool IsAt(Point2D pt)
54     {
55         return (pt.X >= X) && (pt.X <= (X + EndX)) &&
56             (pt.Y >= Y) && (pt.Y <= (Y + EndY));
57     }
58 }
59 }
60
61
```



