```
 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 DrawingShape
 9 {
10     internal class Drawing
11     {
12         private readonly List<Shape> _shapes;
13         private Color _background;
14
15         public Color Background
16         {
17             get { return _background; }
18             set { _background = value; }
19         }
20
21         public List<Shape> Shapes
22         {
23             get { return _shapes; }
24         }
25
26         public Drawing(Color background)
27         {
28             _shapes = new List<Shape>();
29             _background = background;
30         }
31
32         public Drawing() : this(Color.White)
33         {
34
35         }
36
37         public int ShapeCount
38         {
39             get { return _shapes.Count; }
40         }
41
42         public void AddShape(Shape s)
43         {
44             _shapes.Add(s);
45         }
46
47         public void Draw()
48         {
49             SplashKit.ClearScreen(_background);
```

```
50              foreach (Shape s in _shapes)
51              {
52                  s.Draw();
53              }
54              SplashKit.RefreshScreen();
55          }
56
57          public List<Shape> SelectedShapes
58          {
59              get
60              {
61                  List<Shape> result = new List<Shape>();
62                  foreach (Shape s in _shapes)
63                  {
64                      if (s.Selected)
65                      {
66                          result.Add(s);
67                      }
68                  }
69                  return result;
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          public void Save(string filename)
89          {
90              StreamWriter writer = new StreamWriter(filename);
91              try
92              {
93                  writer.WriteColor(_background);
94                  writer.WriteLine(ShapeCount);
95                  foreach (Shape s in _shapes)
96                  {
97                      s.SaveTo(writer);
98                  }
```

```
 99              }
100              finally
101              {
102                  writer.Close();
103              }
104          }
105
106          public void Load(string filename)
107          {
108              StreamReader reader = new StreamReader(filename);
109              try
110              {
111                  _background = reader.ReadColor();
112                  int count = reader.ReadInteger();
113                  for (int i = 0; i < count; i++)
114                  {
115                      string kind = reader.ReadLine();
116                      Shape s;
117                      switch (kind)
118                      {
119                          case "Line":
120                              s = new MyLine();
121                              break;
122                          case "Rectangle":
123                              s = new MyRectangle();
124                              break;
125                          case "Circle":
126                              s = new MyCircle();
127                              break;
128                          default:
129                              throw new InvalidDataException("Unknown shape
                                kind: " + kind);
130                      }
131                      s.LoadFrom(reader);
132                      _shapes.Add(s);
133                  }
134              }
135              finally
136              {
137                  reader.Close();
138              }
139          }
140      }
141  }
142
```