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

```
50          {
51              get
52              {
53                  return _background;
54              }
55              set
56              {
57                  _background = value;
58              }
59          }
60
61          public void Draw()
62          {
63              SplashKit.ClearScreen(_background);
64              for (int i = 0; i < ShapeCount; i++)
65              {
66                  _shapes[i].Draw();
67              }
68          }
69
70          public void SelectShapeAt(Point2D pt)
71          {
72              foreach (Shape s in _shapes)
73              {
74                  if (s.IsAt(pt))
75                  {
76                      s.Selected = true;
77                  }
78                  else
79                  {
80                      s.Selected = false;
81                  }
82              }
83          }
84
85          public void AddShape(Shape s)
86          {
87              _shapes.Add(s);
88          }
89
90          public void RemoveShape(Shape s)
91          {
92              _ = _shapes.Remove(s);
93          }
94
95          public void Save(String filename)
96          {
97              StreamWriter writer = new StreamWriter(filename);
98
```

```
 99              try
100              {
101                  writer.WriteColor(BackGround);
102                  writer.WriteLine(ShapeCount);
103
104                  foreach (Shape s in _shapes)
105                  {
106                      s.SaveTo(writer);
107                  }
108              }
109              finally
110              {
111                  writer.Close();
112              }
113          }
114
115          public void Load(string filename)
116          {
117              StreamReader reader = new StreamReader(filename);
118
119              try
120              {
121                  int count;
122                  Shape s;
123                  string kind;
124
125                  BackGround = reader.ReadColor();
126                  count = reader.ReadInteger();
127                  _shapes.Clear();
128
129                  for (int i = 0; i < count; i++)
130                  {
131                      kind = reader.ReadLine();
132                      switch (kind)
133                      {
134                          case "Rectangle":
135                              s = new MyRectangle();
136                              break;
137                          case "Circle":
138                              s = new MyCircle();
139                              break;
140                          case "Line":
141                              s = new MyLine();
142                              break;
143                          default:
144                              throw new InvalidDataException("Unknown shape
                        kind: " + kind);
145                      }
146                      s.LoadFrom(reader);
```

```
147                         AddShape(s);
148                     }
149                 }
150                 finally
151                 {
152                     reader.Close();
153                 }
154             }
155         }
156 }
157
```