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

```
50                  return result;
51               }
52          }
53      public  int ShapeCount
54          {
55              get { return _shapes.Count; }
56          }
57          public void AddShape(Shape shape)
58          {
59              _shapes.Add(shape);
60          }
61          public void RemoveShape(Shape shape)
62          {
63              _shapes.Remove(shape);
64          }
65          public void Draw()
66          {
67              SplashKit.ClearScreen(_background);
68              foreach (Shape shape in _shapes)
69              {
70                  shape.Draw();
71              }
72          }
73
74          public void SelectShapesAt(Point2D pt)
75          {
76              foreach (Shape shape in _shapes)
77              {
78                  if (shape.IsAt(pt))
79                  {
80                      shape.Selected = true;
81                  }
82                  else { shape.Selected = false; }
83              }
84          }
85
86      }
87  }
88
```