```
using SplashKitSDK;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DrawingShape
{
    internal class Drawing
    {
        private readonly List<Shape> _shapes;
        private Color _background;

        public Color Background
        {
            get { return _background; }
            set { _background = value; }
        }

        public List<Shape> Shapes
        {
            get { return _shapes; }
        }

        public Drawing(Color background)
        {
            _shapes = new List<Shape>();
            _background = background;
        }

        public Drawing() : this(Color.White)
        {

        }

        public int ShapeCount
        {
            get { return _shapes.Count; }
        }

        public void AddShape(Shape s)
        {
            _shapes.Add(s);
        }

        public void Draw()
        {
            SplashKit.ClearScreen(_background);
            foreach (Shape s in _shapes)
            {
                s.Draw();
            }
            SplashKit.RefreshScreen();
        }

        public List<Shape> SelectedShapes
        {
            get
            {
                List<Shape> result = new List<Shape>();
                foreach (Shape s in _shapes)
                {
                    if (s.Selected)
                    {
                        result.Add(s);
                    }
                }
                return result;
            }
        }

        public void SelectShapesAt(Point2D pt)
        {
            foreach (Shape s in _shapes)
            {
                if(s.IsAt(pt))
                {
                    s.Selected = true;
                }
                else
                {
                    s.Selected = false;
                }
            }
        }

        public void Save(string filename)
        {
            StreamWriter writer = new StreamWriter(filename);
            try
            {
                writer.WriteColor(_background);
                writer.WriteLine(ShapeCount);
                foreach (Shape s in _shapes)
                {
                    s.SaveTo(writer);
                }
            }
            finally
            {
                writer.Close();
            }
        }

        public void Load(string filename)
        {
            StreamReader reader = new StreamReader(filename);
            try
            {
                _background = reader.ReadColor();
                int count = reader.ReadInteger();
                for (int i = 0; i < count; i++)
                {
                    string kind = reader.ReadLine();
                    Shape s;
                    switch (kind)
                    {
                        case "Line":
                            s = new MyLine();
                            break;
                        case "Rectangle":
                            s = new MyRectangle();
                            break;
                        case "Circle":
                            s = new MyCircle();
                            break;
                        default:
                            throw new InvalidDataException("Unknown shape kind: " + kind);
                    }
                    s.LoadFrom(reader);
                    _shapes.Add(s);
                }
            }
            finally
            {
                reader.Close();
            }
        }
    }
}
```