```
 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 Shape
11      {
12          private Color _color;
13          private float _x, _y;
14          private int _width, _height;
15          private bool _selected;
16
17          public Shape()
18          {
19              _color = Color.Green;
20              _x = 0;
21              _y = 0;
22              _width = 100;
23              _height = 100;
24          }
25          public Color Color
26          {
27              get { return _color; }
28              set { _color = value; }
29          }
30          public float X
31          {
32              get { return _x; }
33              set { _x = value; }
34          }
35          public float Y
36          {
37              get { return _y; }
38              set { _y = value; }
39          }
40
41          public bool Selected { get; internal set; }
42
43          public void DrawOutline()
44          {
45              SplashKit.FillRectangle(Color.Black, _x - 2, _y - 2, _width +
                  4, _height + 4);
46          }
47
48          public void Draw()
```

```
49          {
50              if (Selected)
51              {
52                  DrawOutline();
53              }
54              SplashKit.FillRectangle(_color, _x, _y, _width, _height);
55          }
56
57          public bool IsAt(Point2D pt)
58          {
59              return SplashKit.PointInRectangle(pt, SplashKit.RectangleFrom
                  (X, Y, _width, _height));
60          }
61      }
62  }
63
```

```
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  }
74
```

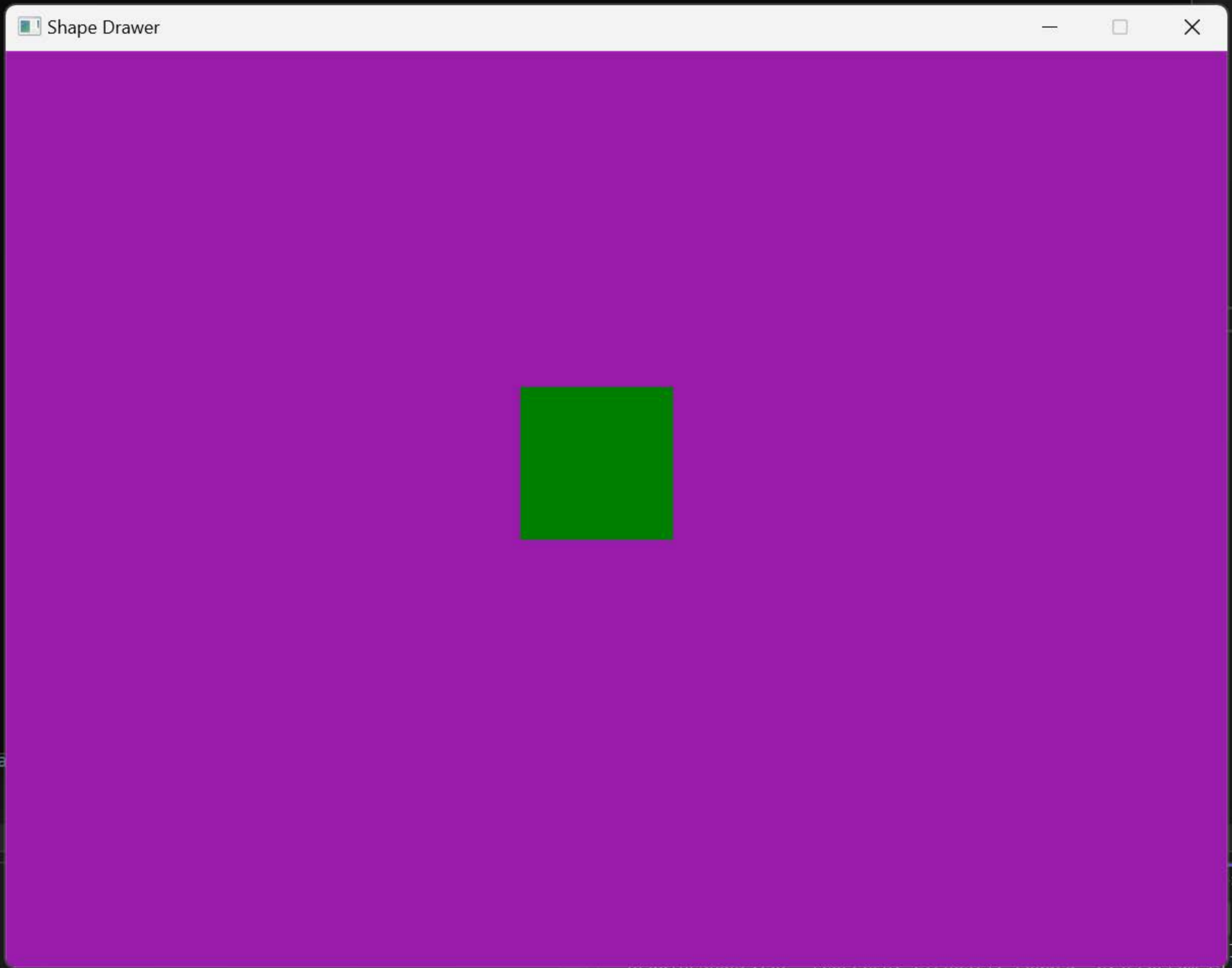
```
using System;
using System.Collections.Generic;
using SplashKitSDK;

namespace DrawingShape
{
    public class Program
    {
        public static void Main()
        {
            Window window = new Window("Shape Drawer", 800, 600);
            Drawing myDrawing = new Drawing();
            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();
                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    Shape myShape = new Shape();
                    myShape.X = SplashKit.MouseX();
                    myShape.Y = SplashKit.MouseY();
                    myDrawing.AddShape(myShape);
                }
                Point2D pt = SplashKit.MousePosition();
                if(SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    myDrawing.Background = SplashKit.RandomRGBColor(255);
                }
                if(SplashKit.MouseClicked(MouseButton.RightButton))
                {
                    foreach(Shape s in myDrawing.Shapes)
                    {
                        if(s.IsAt(pt))
                        {
                            s.Selected = !s.Selected;
                        }
                    }
                }
                if(SplashKit.KeyTyped(KeyCode.DeleteKey)||
                  SplashKit.KeyTyped(KeyCode.BackspaceKey))
                {
                    for (int i = myDrawing.ShapeCount - 1; i >= 0; i--)
                    {
                        if (myDrawing.Shapes[i].Selected)
                        {
                            myDrawing.Shapes.RemoveAt(i);
                        }
                    }
                }
```

```
49                  myDrawing.Draw();
50                  SplashKit.RefreshScreen();
51              } while (!window.CloseRequested);
52          }
53      }
54  }
55
```

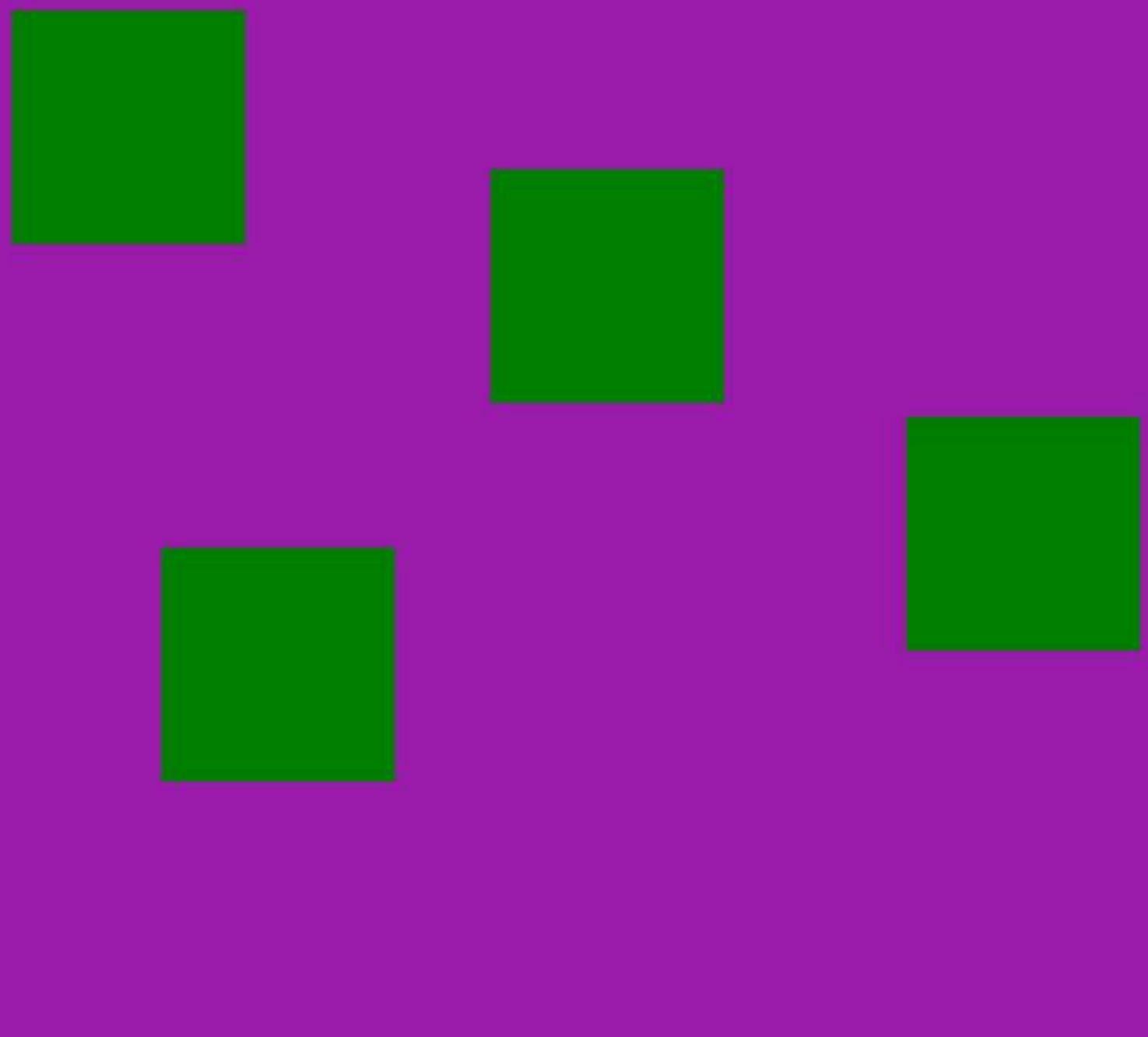
Shape Drawer

'DrawingShape.exe' (CoreCLR: clrhost): Loaded 'C:\Program File

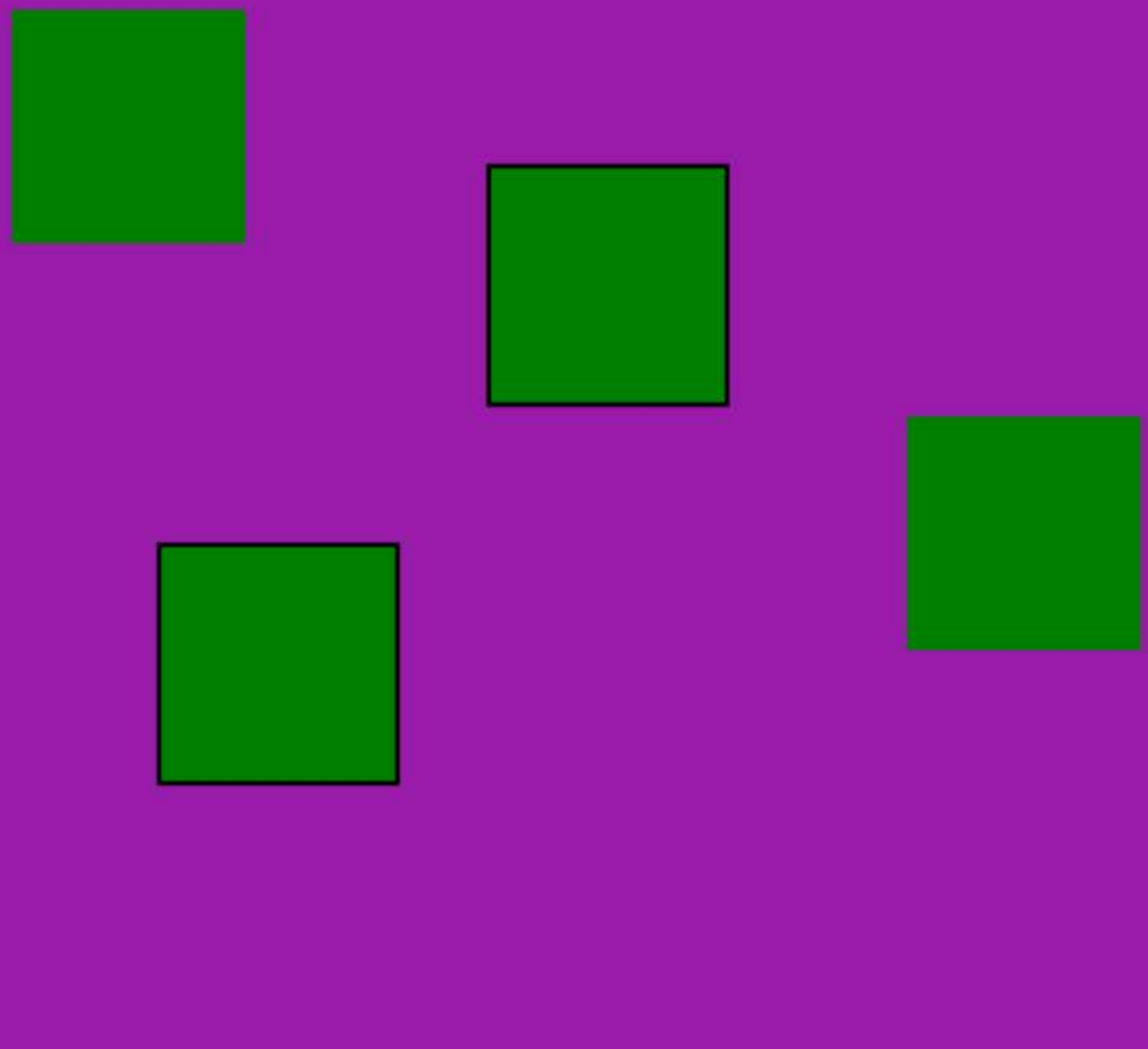


'DrawingShape.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\
'DrawingShape.exe' (CoreCLR: clrhost): Loaded 'C:\Program Files\

Shape Drawer



Shape Drawer



Shape Drawer