Swinburne University of Technology

COS20007 Object Oriented Programming

3.3P - Drawing Program - A Drawing Class

PDF generated at 15:42 on Monday 3rd April, 2023

```
using System;
using SplashKitSDK;
using DrawingShape;

namespace DrawingShape
{
    public class Program
    {
        public static void Main()
        {
            Drawing myDrawing = new Drawing();

            new Window("Drawing Shape", 800, 600);
            do
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen();

                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    Shape shape = new Shape();
                    shape.X = SplashKit.MouseX();
                    shape.Y = SplashKit.MouseY();
                    myDrawing.AddShape(shape);
                }

                if (SplashKit.MouseClicked(MouseButton.RightButton))
                {
                    myDrawing.SelectShapesAt(SplashKit.MousePosition());
                }


                if (SplashKit.KeyTyped(KeyCode.BackspaceKey) ||
↪   SplashKit.KeyTyped(KeyCode.DeleteKey))
                {
                    foreach (Shape s in myDrawing.SelectedShapes())
                    {
                        myDrawing.RemoveShape(s);
                    }
                }

                if (SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    myDrawing.Background = SplashKit.RandomRGBColor(255);
                }

                myDrawing.Draw();

                SplashKit.RefreshScreen();

            }
            while (!SplashKit.WindowCloseRequested("Drawing Shape"));
        }
```

```
53      }
54
55  }
```

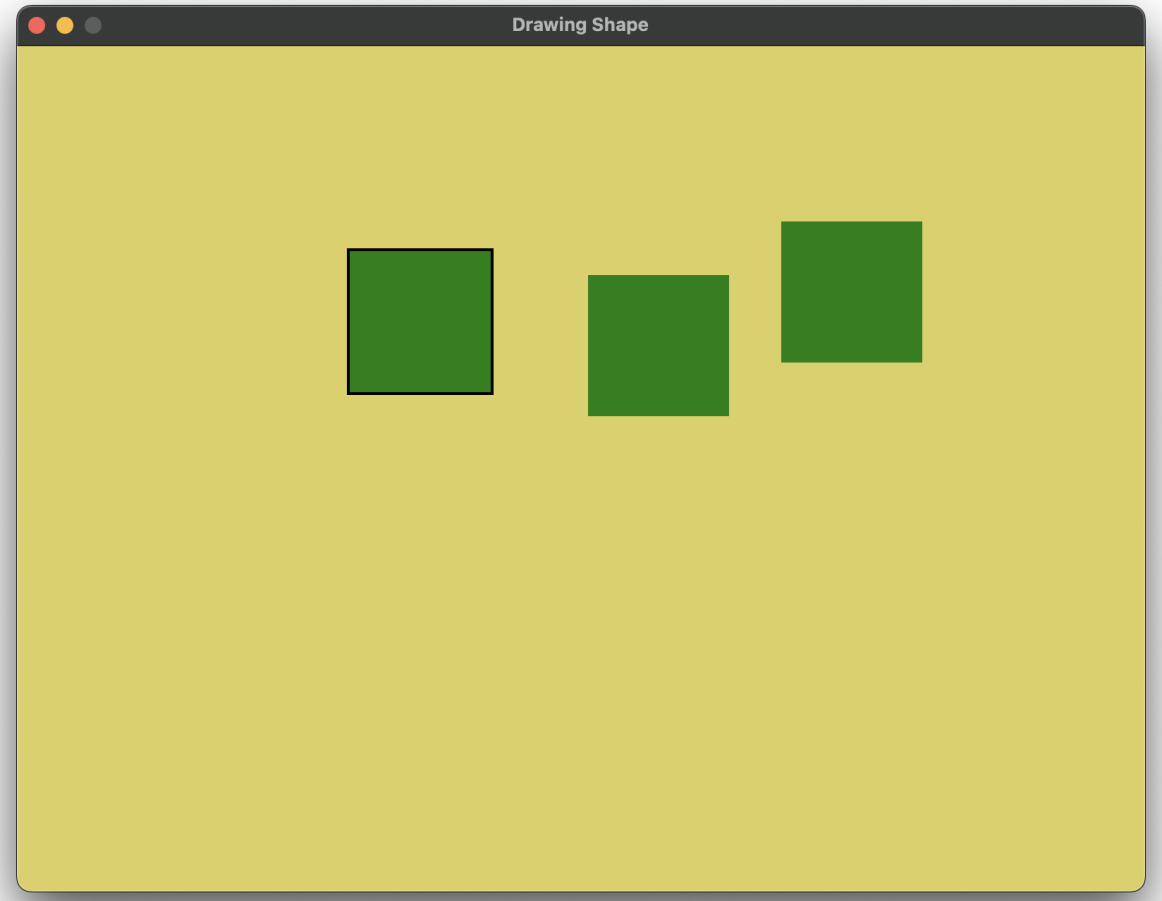
```
using System;
using SplashKitSDK;
using System.Linq;
using System.Collections.Generic;

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


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

        //default constructor
        public Drawing() : this(Color.White)
        {

        }

        //list of currently selected shapes
        public List<Shape> SelectedShapes()
        {
            List<Shape> result = new List<Shape>();
                foreach (Shape s in _shapes)
                {
                    if (s.Selected == true)
                    {
                        result.Add(s);
                    }
                }
                return result;
        }

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

        //background color
        public Color Background
        {
            get
            {
                return _background;
```

```
54              }
55              set
56              {
57                  _background = value;
58              }
59          }
60
61
62
63          public void Draw()
64          {
65              SplashKit.ClearScreen(_background);
66
67              foreach (Shape shape in _shapes)
68              {
69                  shape.Draw();
70              }
71          }
72
73          public void ShapeColor()
74          {
75              foreach (Shape s in _shapes)
76              {
77                  if (s.Selected)
78                  {
79                      s.Colors = Color.RandomRGB(255);
80                  }
81              }
82          }
83
84          public void SelectShapesAt(Point2D pt)
85          {
86              foreach (Shape s in _shapes)
87              {
88                  if (s.IsAt(pt))
89                  {
90                      s.Selected = true;
91                  }
92                  else
93                  {
94                      s.Selected = false;
95                  }
96              }
97          }
98
99
100         public void AddShape(Shape shape)
101         {
102             _shapes.Add(shape);
103         }
104
105         public void RemoveShape(Shape shape)
106         {
```

```
107                 _shapes.Remove(shape);
108             }
109         }
110 }
```

```
using System;
using SplashKitSDK;


namespace DrawingShape
{
    public class Shape
    {
        private Color _color;
        private float _x, _y;
        private int _width, _height;
        private bool _selected;

        public Shape()
        {
            _color = Color.Green;
            _x = 0;
            _y = 0;
            _width = 100;
            _height = 100;
        }

        public Color Colors
        {
            get
            {
                return _color;
            }
            set
            {
                _color = value;
            }
        }

        public float X
        {
            get
            {
                return _x;
            }
            set
            {
                _x = value;
            }
        }

        public float Y
        {
            get
            {
                return _y;
            }
            set
```

```
54              {
55                  _y = value;
56              }
57          }
58
59          public void Draw()
60          {
61              if (Selected)
62              {
63                  DrawOutline();
64              }
65              SplashKit.FillRectangle(_color, _x, _y, _width, _height);
66          }
67
68          public bool IsAt(Point2D p)
69          {
70              return SplashKit.PointInRectangle(p, SplashKit.RectangleFrom(X, Y,
    ↪  _width, _height));
71          }
72
73          public bool Selected
74          {
75              get
76              {
77                  return _selected;
78              }
79              set
80              {
81                  _selected = value;
82              }
83          }
84
85          public void DrawOutline()
86          {
87              SplashKit.FillRectangle(Color.Black, _x - 2, _y - 2, _width + 4, _height
    ↪  + 4);
88          }
89      }
90  }
```


Drawing Shape