Swinburne University of Technology

COS20007 Object Oriented Programming

Drawing Program - Multiple Shape Kinds

PDF generated at 05:25 on Thursday 24th August, 2023

Produced by OOP Submission Formatter

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class Program
    {

        private enum ShapeKind
        {
            Rectangle,
            Circle,
            Line
        }
        public static void Main()
        {
            Drawing myDraw = new Drawing();
            Window window = new Window("Shape Drawer", 800, 600);

            ShapeKind kindToAdd = ShapeKind.Circle;

            do
            {
                SplashKit.ProcessEvents();
                myDraw.Draw();

                if (SplashKit.KeyDown(KeyCode.RKey))
                {
                    kindToAdd = ShapeKind.Rectangle;
                }

                if (SplashKit.KeyDown(KeyCode.CKey))
                {
                    kindToAdd = ShapeKind.Circle;
                }

                if (SplashKit.KeyDown(KeyCode.LKey))
                {
                    kindToAdd = ShapeKind.Line;
                }



                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    Shape newShape;

                    if (kindToAdd == ShapeKind.Circle)
                    {

                        newShape = new MyCircle();
                        newShape.Color = SplashKit.RandomRGBColor(255);
                    }
```

```
54                      else if (kindToAdd == ShapeKind.Rectangle)
55                      {
56
57                          newShape = new MyRectangle();
58                      }
59                      else
60                      {
61
62                          newShape = new MyLine();
63                      }
64
65                      newShape.X = SplashKit.MouseX();
66                      newShape.Y = SplashKit.MouseY();
67
68                      myDraw.AddShape(newShape);
69
70                  }
71
72                  if (SplashKit.MouseClicked(MouseButton.RightButton))
73                  {
74                      myDraw.SelectedShapesAt(SplashKit.MousePosition());
75                  }
76
77                  if (SplashKit.KeyDown(KeyCode.SpaceKey))
78                  {
79                      myDraw.Background = SplashKit.RandomRGBColor(255);
80                  }
81
82                  if (SplashKit.KeyDown(KeyCode.DeleteKey) ||
    ↪  SplashKit.KeyDown(KeyCode.BackspaceKey))
83                  {
84                      foreach (Shape s in myDraw.SelectedShapes)
85                      {
86                          myDraw.RemoveShape(s);
87                      }
88                  }
89
90                  SplashKit.RefreshScreen();
91              } while (!window.CloseRequested);
92          }
93      }
94  }
```

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

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

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

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

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

        public Color Background
        {
            get
            {
                return _background;
            }

```

```
54              set
55              {
56                  _background = value;
57              }
58          }
59
60          public void Draw()
61          {
62              SplashKit.ClearScreen(_background);
63              foreach (Shape shape in _shapes)
64              {
65                  shape.Draw();
66              }
67          }
68          public void SelectedShapesAt(Point2D pt)
69          {
70              foreach (Shape s in _shapes)
71              {
72                  if (s.IsAt(pt))
73                  {
74                      s.Selected = true;
75                  }
76                  else
77                  {
78                      s.Selected = false;
79                  }
80              }
81          }
82
83          public void AddShape(Shape s)
84          {
85              _shapes.Add(s);
86          }
87
88          public void RemoveShape(Shape s)
89          {
90              _shapes.Remove(s);
91          }
92
93      }
94  }
```

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public abstract class Shape
    {
        private Color _color;
        private bool _selected;
        private float _x;
        private float _y;
        public Shape(Color color)
        {
            _color = color;
            _x = 0;
            _y = 0;
            _selected = false;
        }

        public Shape() : this(Color.Yellow) { }

        public Color Color
        {
            get
            {
                return _color;
            }

            set
            {
                _color = value;
            }
        }

        public bool Selected
        {
            get
            {
                return _selected;
            }

            set
            {
                _selected = value;
            }
        }

        public float X
        {
            get
            {
                return _x;
            }
```

```
54
55              set
56              {
57                  _x = value;
58              }
59          }
60
61          public float Y
62          {
63              get
64              {
65                  return _y;
66              }
67
68              set
69              {
70                  _y = value;
71              }
72          }
73
74          public abstract void Draw();
75          public abstract void DrawOutline();
76
77          public abstract bool IsAt(Point2D point);
78
79
80      }
81  }
```

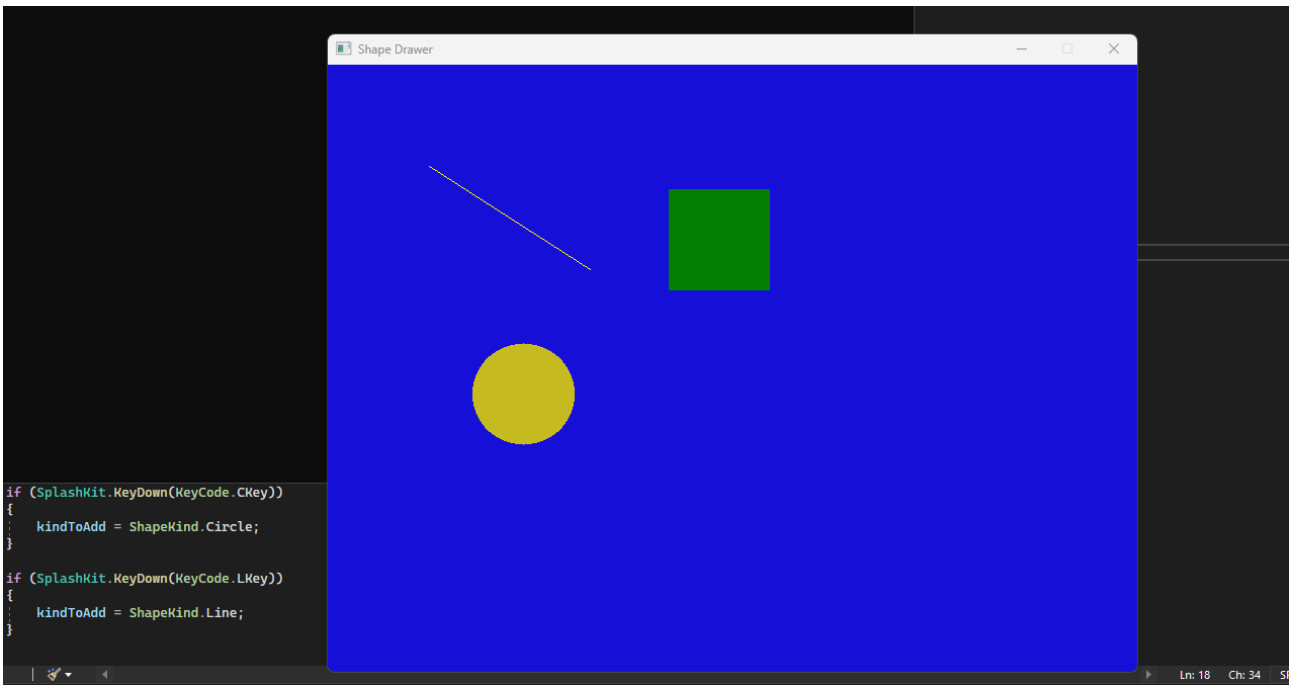
```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class MyRectangle : Shape
    {
        private int _width, _height;

        public MyRectangle(Color clr, float x, float y, int width, int height) :
    ↪  base(clr)
        {
            X = x;
            Y = y;
            Width = width;
            Height = height;
        }

        public MyRectangle() : this(Color.Green, 0, 0, 100, 100) { }

        public int Width
        {
            get
            {
                return _width;
            }

            set
            {
                _width = value;
            }
        }

        public int Height
        {
            get
            {
                return _height;
            }

            set
            {
                _height = value;
            }
        }

        public override void Draw()
        {
            if (Selected)
                DrawOutline();
            SplashKit.FillRectangle(Color, X, Y, Width, Height);
        }

```

```
53          public override void DrawOutline()
54          {
55              SplashKit.DrawRectangle(Color.Black, X - 2, Y - 2, _width + 4, _height +
    ↪  4);
56          }
57
58          public override bool IsAt(Point2D point)
59          {
60              if ((((point.X >= X) && (point.X <= (X + _width))) && (point.Y >= Y) &&
    ↪  (point.Y <= (Y + _height))))
61                  return true;
62              else
63                  return false;
64          }
65
66      }
67
68  }
```

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class MyCircle : Shape
    {
        private int _radius;

        public MyCircle(Color clr, float x, float y, int radius) : base(clr)
        {
            X = x;
            Y = y;
            _radius = radius;
        }

        public MyCircle() : this(Color.Blue, 0, 0, 50) { }

        public int Radius
        {
            get
            {
                return _radius;
            }

            set
            {
                _radius = value;
            }
        }

        public override void Draw()
        {
            if (Selected)
                DrawOutline();
            SplashKit.FillCircle(Color, X, Y, _radius);
        }

        public override void DrawOutline()
        {
            SplashKit.DrawCircle(Color.Black, X, Y, _radius + 2);
        }

        public override bool IsAt(Point2D point)
        {
            return SplashKit.PointInCircle(point, SplashKit.CircleAt(this.X, this.Y,
    _radius));
        }

    }
}
```

```
using System;
using SplashKitSDK;

namespace ShapeDrawer
{
    public class MyLine : Shape
    {
        private float _endX, _endY;

        public MyLine(Color clr, float startX, float startY, float endX, float endY)
    ↪   : base(clr)
        {
            X = startX;
            Y = startY;
            EndX = endX;
            EndY = endY;
        }

        public MyLine() : this(Color.Yellow, 0, 0, 100, 100) { }
        public float EndX
        {
            get
            {
                return _endX;
            }

            set
            {
                _endX = value;
            }
        }

        public float EndY
        {
            get
            {
                return _endY;
            }

            set
            {
                _endY = value;
            }
        }

        public override void Draw()
        {
            if (Selected)
                DrawOutline();
            SplashKit.DrawLine(Color, X, Y, EndX, EndY);
        }

        public override void DrawOutline()
```

```
53          {
54              SplashKit.DrawLine(Color.Black, X - 2, Y - 2, EndX - 2, EndY - 2);
55          }
56
57          public override bool IsAt(Point2D point)
58          {
59              return SplashKit.PointOnLine(point, SplashKit.LineFrom(X, Y, EndX, EndY),
    ↪  10.0F);
60          }
61      }
62  }
```


Shape Drawer
if (SplashKit.KeyDown(KeyCode.CKey))
{
kindToAdd = ShapeKind.Circle;
}
if (SplashKit.KeyDown(KeyCode.LKey))
{
kindToAdd = ShapeKind.Line;
}
Ln: 18
Ch: 34