

```
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Point;
public class Shape
{
    private int x;
    private int y;
    private Color color;

    public Shape(int x, int y, Color color)
    {
        this.x = x;
        this.y = y;
        this.color = color;
    }

    public int getX()
    {
        return x;
    }

    public int getY()
    {
        return y;
    }

    public Color getColor()
    {
        return color;
    }

    public void setEndPoint(Point start, Point end)
    {
        x = Math.min(start.x, end.x);
        y = Math.min(start.y, end.y);
    }

    public void draw(Graphics g)
    {
        g.setColor(color);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Point;
public class IrregularShape extends Shape
{
    private int width;
    private int height;

    public IrregularShape(int x, int y, int width, int height, Color color)
    {
        super(x, y, color);
        this.width = width;
        this.height = height;
    }

    public int getWidth()
    {
        return width;
    }

    public int getHeight()
    {
        return height;
    }

    public void setEndPoint(Point start, Point end)
    {
        width = Math.abs(start.x - end.x);
        height = Math.abs(start.y - end.y);

        super.setEndPoint(start, end);
    }

    public void draw(Graphics g)
    {
        super.draw(g);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Point;
public class RegularShape extends Shape
{
    private int side;

    public RegularShape(int x, int y, int side, Color color)
    {
        super(x, y, color);
        this.side = side;
    }

    public int getSide()
    {
        return side;
    }

    public void setEndPoint(Point start, Point end)
    {
        side = Math.min(Math.abs(start.x - end.x), Math.abs(start.y - end.y));
        super.setEndPoint(start, end);
    }

    public void draw(Graphics g)
    {
        super.draw(g);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
public class Circle extends RegularShape
{
    public Circle(int x, int y, int side, Color color)
    {
        super(x, y, side, color);
    }

    public void draw(Graphics g)
    {
        super.draw(g);

        int x = getX();
        int y = getY();
        int s = getSide();

        g.drawOval(x, y, s, s);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
public class Ellipse extends IrregularShape
{
    public Ellipse(int x, int y, int width, int height, Color color)
    {
        super(x, y, width, height, color);
    }

    public void draw(Graphics g)
    {
        super.draw(g);

        int x = getX();
        int y = getY();
        int w = getWidth();
        int h = getHeight();

        g.drawOval(x, y, w, h);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
public class Rectangle extends IrregularShape
{
    public Rectangle(int x, int y, int width, int height, Color color)
    {
        super(x, y, width, height, color);
    }

    public void draw(Graphics g)
    {
        super.draw(g);

        int x = getX();
        int y = getY();
        int w = getWidth();
        int h = getHeight();

        g.drawRect(x, y, w, h);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
public class Square extends RegularShape
{
    public Square(int x, int y, int side, Color color)
    {
        super(x, y, side, color);
    }

    public void draw(Graphics g)
    {
        super.draw(g);

        int x = getX();
        int y = getY();
        int s = getSide();

        g.drawRect(x, y, s, s);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
public class Triangle extends IrregularShape
{
    public Triangle(int x, int y, int width, int height, Color color)
    {
        super(x, y, width, height, color);
    }

    public void draw(Graphics g)
    {
        super.draw(g);

        int x = getX();
        int y = getY();
        int w = getWidth();
        int h = getHeight();

        g.drawLine(x, y + h - 1, x + w / 2 - 1, y);
        g.drawLine(x + w / 2 - 1, y, x + w - 1, y + h - 1);
        g.drawLine(x + w - 1, y + h - 1, x, y + h - 1);
    }
}
```



```
import java.awt.Graphics;
import java.util.ArrayList;
public class Shapes
{
    private ArrayList<Shape> alShapes = new ArrayList<>();

    public void addShape(Shape e)
    {
        alShapes.add(e);
    }

    public void clear()
    {
        alShapes.clear();
    }

    public void draw(Graphics g)
    {
        for (int i = 0; i < alShapes.size(); i++)
            alShapes.get(i).draw(g);
    }
}
```

```
import java.awt.Color;
import java.awt.Graphics;
public class DrawPanel extends javax.swing.JPanel
{
    private Shapes shapes = null;

    public void setShapes(Shapes shapes)
    {
        this.shapes = shapes;
    }

    @Override
    public void paintComponent(Graphics g)
    {
        g.setColor(Color.WHITE);
        g.fillRect(0, 0, getWidth(), getHeight());

        if (shapes != null)
            shapes.draw(g);
    }

    public DrawPanel()
    {
        initComponents();
    }
    // Skipped: ... initComponents { ... }
    // Skipped: ... graphic attributes
}
```

```

import java.awt.Color;
import java.awt.Point;
import java.awt.event.MouseEvent;
public class MainFrame extends javax.swing.JFrame
{
    private Shapes shapes = new Shapes();
    private Shape newShape = null;
    private Point start = null;

    public MainFrame()
    {
        initComponents();
        drawPanel.setShapes(shapes);
    }

    public void updateView()
    {
        // Rien d'autre à faire...
        repaint();
    }
// Skipped: ... initComponents { ... }
    private void drawPanelMousePressed(java.awt.event.MouseEvent evt)//GEN-FIRST:event_drawPanelMousePressed
    {//GEN-HEADEREND:event_drawPanelMousePressed
        start = evt.getPoint();

        if (evt.getButton() == MouseEvent.BUTTON1)
        {
            if (evt.isShiftDown())
                newShape = new Square(start.x, start.y, 0, Color.ORANGE);
            else
                newShape = new Rectangle(start.x, start.y, 0, 0, Color.RED);
        }
        else if (evt.getButton() == MouseEvent.BUTTON2)
            newShape = new Triangle(start.x, start.y, 0, 0, Color.BLUE);
        else
        {
            if (evt.isShiftDown())
                newShape = new Circle(start.x, start.y, 0, Color.CYAN);
            else
                newShape = new Ellipse(start.x, start.y, 0, 0, Color.GREEN);
        }

        shapes.addShape(newShape);
        updateView();
    }//GEN-LAST:event_drawPanelMousePressed

    private void drawPanelMouseDragged(java.awt.event.MouseEvent evt)//GEN-FIRST:event_drawPanelMouseDragged
    {//GEN-HEADEREND:event_drawPanelMouseDragged
        newShape.setEndPoint(start, evt.getPoint());

        updateView();
    }//GEN-LAST:event_drawPanelMouseDragged

    private void drawPanelMouseReleased(java.awt.event.MouseEvent evt)//GEN-FIRST:event_drawPanelMouseReleased
    {//GEN-HEADEREND:event_drawPanelMouseReleased
        newShape.setEndPoint(start, evt.getPoint());

        updateView();
    }//GEN-LAST:event_drawPanelMouseReleased
// Skipped: ... Look & Feel
// Skipped: ... graphic attributes
}

```