

```
public class Shape
{
    private Point from;
    private Point to;
    private Color color;

    public Shape(int pX1, int pY1, int pX2, int pY2, Color pColor)
    {
        from = new Point(pX1, pY1);
        to = new Point(pX2, pY2);
        color = pColor;
    }

    public Shape(Point pFrom, Point pTo, Color pColor)
    {
        from = pFrom;
        to = pTo;
        color = pColor;
    }

    public Point getFrom()
    {
        return from;
    }

    public void setFrom(Point pFrom)
    {
        from = pFrom;
    }

    public Point getTo()
    {
        return to;
    }

    public void setTo(Point pTo)
    {
        to = pTo;
    }

    public void setColor(Color pColor)
    {
        color = pColor;
    }

    public Color getColor()
    {
        return color;
    }

    public int getTop()
    {
        return Math.min(from.y, to.y);
    }

    public int getBottom()
    {
        return getTop() + getHeight() - 1;
    }

    public int getLeft()
    {
        return Math.min(from.x, to.x);
    }

    public int getRight()
    {
        return getLeft() + getWidth() - 1;
    }

    public int getWidth()
    {
        return Math.abs(from.x - to.x) + 1;
    }

    public int getHeight()
    {
        return Math.abs(from.y - to.y) + 1;
    }

    public void draw(Graphics g)
    {
        // Ne rien faire
    }

    public String toString()
    {
        return "[" + from.x + "," + from.y + "] -> [" + to.x + "," + to.y + "];"
    }
}
```

```
public class Ellipse extends Shape
{
    public Ellipse(int pX1, int pY1, int pX2, int pY2, Color pColor)
    {
        super(pX1, pY1, pX2, pY2, pColor);
    }

    public Ellipse(Point pFrom, Point pTo, Color pColor)
    {
        super(pFrom, pTo, pColor);
    }

    @Override
    public void draw(Graphics g)
    {
        g.setColor(getColor());
        g.drawOval(getLeft(), getTop(), getWidth(), getHeight());
    }

    @Override
    public String toString()
    {
        return "Ellipse " + super.toString();
    }
}
```

```
public class IsoTriangle extends Shape
{
    public IsoTriangle(int pX1, int pY1, int pX2, int pY2, Color pColor)
    {
        super(pX1, pY1, pX2, pY2, pColor);
    }

    public IsoTriangle(Point pFrom, Point pTo, Color pColor)
    {
        super(pFrom, pTo, pColor);
    }

    @Override
    public void draw(Graphics g)
    {
        int x = getLeft();
        int y = getTop();
        int w = getWidth();
        int h = getHeight();

        g.setColor(getColor());

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

    @Override
    public String toString()
    {
        return "IsoTriangle " + super.toString();
    }
}
```

```
public class Line extends Shape
{
    public Line(int pX1, int pY1, int pX2, int pY2, Color pColor)
    {
        super(pX1, pY1, pX2, pY2, pColor);
    }

    public Line(Point pFrom, Point pTo, Color pColor)
    {
        super(pFrom, pTo, pColor);
    }

    @Override
    public void draw(Graphics g)
    {
        g.setColor(getColor());
        g.drawLine(getFrom().x, getFrom().y, getTo().x, getTo().y);
    }

    @Override
    public String toString()
    {
        return "Line " + super.toString();
    }
}
```

```
public class Rectangle extends Shape
{
    public Rectangle(int pX1, int pY1, int pX2, int pY2, Color pColor)
    {
        super(pX1, pY1, pX2, pY2, pColor);
    }

    public Rectangle(Point pFrom, Point pTo, Color pColor)
    {
        super(pFrom, pTo, pColor);
    }

    @Override
    public void draw(Graphics g)
    {
        g.setColor(getColor());
        g.drawRect(getLeft(), getTop(), getWidth(), getHeight());
    }

    @Override
    public String toString()
    {
        return "Rectangle " + super.toString();
    }
}
```

```
public class RightTriangle extends Shape
{
    public RightTriangle(int pX1, int pY1, int pX2, int pY2, Color pColor)
    {
        super(pX1, pY1, pX2, pY2, pColor);
    }

    public RightTriangle(Point pFrom, Point pTo, Color pColor)
    {
        super(pFrom, pTo, pColor);
    }

    @Override
    public void draw(Graphics g)
    {
        int x = getLeft();
        int y = getTop();
        int w = getWidth();
        int h = getHeight();

        g.setColor(getColor());

        g.drawLine(x + w, y + h, x, y + h);
        g.drawLine(x, y + h, x, y);
        g.drawLine(x, y, x + w, y + h);
    }

    @Override
    public String toString()
    {
        return "RightTriangle " + super.toString();
    }
}
```

```
public class Shapes
{
    private ArrayList<Shape> alShapes = new ArrayList<>();

    public void add(Shape s)
    {
        alShapes.add(s);
    }

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

    public void draw(Graphics g)
    {
        // Chaque figure se dessine elle-même
        for (int i = 0; i < alShapes.size(); i++)
            alShapes.get(i).draw(g);
    }

    public Object[] toArray()
    {
        return alShapes.toArray();
    }
}
```

```
public class DrawPanel extends javax.swing.JPanel
{
    private Shapes shapes = null;

    public DrawPanel()
    {
        initComponents();
    }

    public void setShapes(Shapes s)
    {
        shapes = s;
    }

    @Override
    public void paintComponent(Graphics g)
    {
        // Effacer le dessin
        int w = getWidth();
        int h = getHeight();
        g.setColor(Color.WHITE);
        g.fillRect(0, 0, w, h);

        // Dessiner les figures si présentes
        if (shapes != null)
            shapes.draw(g);
    }
}
// Skipped: ... initComponents { ... }
// Variables declaration - do not modify//GEN-BEGIN:variables
// End of variables declaration//GEN-END:variables
```



```

public class MainFrame extends javax.swing.JFrame
{
    private Shapes shapes = new Shapes();
    private Shape newShape = null;

    // Couleur à utiliser pour de nouvelles lignes
    private Color drawColor = Color.BLUE;

    public MainFrame()
    {
        initComponents();
        // La liste des shapes est fixée directement dans la JList typeList:
        // 0=Line / 1=Rectangle / 2=Ellipse / 3=IsoTriangle / 4=RightTriangle
        // au début on sélectionne directement la ligne
        typeList.setSelectedIndex(0);
        drawPanel.setShapes(shapes);
        updateView();
    }

    public void updateView()
    {
        shapesList.setListData(shapes.toArray());
        repaint();
    }
// Skipped: ... initComponents { ... }
    private void drawPanelMouseDragged(java.awt.event.MouseEvent evt) {GEN-FIRST:event_drawPanelMouseDragged
        if (newShape == null)
            return;

        newShape.setTo(evt.getPoint());
        updateView();
    }GEN-LAST:event_drawPanelMouseDragged

    private void drawPanelMousePressed(java.awt.event.MouseEvent evt) {GEN-FIRST:event_drawPanelMousePressed
        int i = typeList.getSelectedIndex();
        if (i == -1)
            return; // plus rien de sélectionné!

        if (i == 0)
            newShape = new Line(evt.getPoint(), evt.getPoint(), drawColor);
        else if (i == 1)
            newShape = new Rectangle(evt.getPoint(), evt.getPoint(), drawColor);
        else if (i == 2)
            newShape = new Ellipse(evt.getPoint(), evt.getPoint(), drawColor);
        else if (i == 3)
            newShape = new IsoTriangle(evt.getPoint(), evt.getPoint(), drawColor);
        else
            newShape = new RightTriangle(evt.getPoint(), evt.getPoint(), drawColor);

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

    private void drawPanelMouseReleased(java.awt.event.MouseEvent evt) {GEN-FIRST:event_drawPanelMouseReleased
        if (newShape == null)
            return;

        newShape.setTo(evt.getPoint());
        newShape = null;
        updateView();
    }GEN-LAST:event_drawPanelMouseReleased

    private void changeColorButtonActionPerformed(java.awt.event.ActionEvent evt){GEN-FIRST:event_changeColorButtonActionPerformed
    {GEN-HEADEREND:event_changeColorButtonActionPerformed
        drawColor = JColorChooser.showDialog(this, "Choix d'une couleur", drawColor);
        updateView();
    }GEN-LAST:event_changeColorButtonActionPerformed

    private void drawPanelMouseClicked(java.awt.event.MouseEvent evt){GEN-FIRST:event_drawPanelMouseClicked
    {GEN-HEADEREND:event_drawPanelMouseClicked
        if (evt.getClickCount() >= 2)
        {
            shapes.clear();
            updateView();
        }
    }GEN-LAST:event_drawPanelMouseClicked
// Skipped: ... Look & Feel
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton changeColorButton;
    private DrawPanel drawPanel;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JList shapesList;
    private javax.swing.JList typeList;
    // End of variables declaration//GEN-END:variables
}

```