

# Java2D – Moving Shapes – One Trace

## Use Concept Trace Slicing

Supporting Applications : [GeoSoft]

```
import java.awt.color.ColorSpace;
import java.awt.image.ImageProducer;
import java.awt.Adjustable;
import java.awt.Color;
import java.awt.Image;
import java.awt.Point;
import java.awt.BasicStroke;
import java.awt.Polygon;
import java.awt.event.MouseListener;
import java.awt.event.AdjustmentListener;
import java.awt.event.ComponentListener;
import java.awt.BorderLayout;
import java.awt.Component;
import java.awt.RenderingHints$Key;
import java.awt.Dimension;
import java.awt.Rectangle;
import java.util.EventObject;
import java.awt.LayoutManager;
import java.awt.event.ActionListener;
import java.awt.event.InputEvent;
import java.awt.event.MouseMotionListener;
import java.awt.print.Printable;
import java.awt.geom.Area;
import java.awt.geom.RectangularShape;
import java.awt.event.MouseEvent;
import java.awt.Paint;
import java.awt.Graphics;
import java.awt.Container;

public class AppomponentListener
implements ComponentListener {
}

public class AppouseMotionListener
implements MouseMotionListener {

    public void mouseDragged(MouseEvent) {
        int app_int2 = MouseEvent.getModifiers();
        int app_int1 = MouseEvent.getX();
        int app_int = MouseEvent.getY();
        Component.setLocation(app_int1, app_int);
    }

    public void mouseMoved(MouseEvent) {
        int app_int1 = MouseEvent.getX();
        int app_int = MouseEvent.getY();
        Component.setLocation(app_int1, app_int);
    }
}

public class AppouseListener
implements MouseListener {

    public void mouseClicked(MouseEvent) {
    }

    public void mouseEntered(MouseEvent) {
        int app_int1 = MouseEvent.getX();
        int app_int = MouseEvent.getY();
    }

    public void mouseExited(MouseEvent) {
        int app_int1 = MouseEvent.getX();
        int app_int = MouseEvent.getY();
    }

    public void mousePressed(MouseEvent) {
        int app_int2 = MouseEvent.getModifiers();
        int app_int1 = MouseEvent.getX();
        int app_int = MouseEvent.getY();
    }

    public void mouseReleased(MouseEvent) {
        int app_int2 = MouseEvent.getModifiers();
        int app_int1 = MouseEvent.getX();
        int app_int = MouseEvent.getY();
    }
}

public class AppayoutManager
implements LayoutManager {

    public void addLayoutComponent(String, Component) {
    }

    public void layoutContainer(container) {
        Component[] componentArray = container.getComponents();
        int app_int5 = Component.getX(); // REPEATED!
        int app_int6 = Component.getY(); // REPEATED!
    }

    public Dimension minimumLayoutSize(container) {
        int app_int3 = container.getHeight();
        int app_int4 = container.getWidth();
        Dimension dimension = new Dimension(app_int4, app_int3);
    }

    public Dimension preferredLayoutSize(container) {
        int app_int3 = container.getHeight();
        int app_int4 = container.getWidth();
        Dimension dimension = new Dimension(app_int4, app_int3);
    }

    public void removeLayoutComponent(Component) {
    }
}
```

```

public class AppdjustmentListener
implements AdjustmentListener {

    public AppdjustmentListener() {
        /*          Cyclic Statements          */
        Adjustable.setVisibleAmount(int);
        Adjustable.setUnitIncrement(int);
        Adjustable.setMaximum(int);
        Adjustable.addAdjustmentListener(appdjustmentListener);
        Adjustable.removeAdjustmentListener(appdjustmentListener);
        Adjustable.setValue(int);
        Adjustable.setBlockIncrement(int);
        Adjustable.setMinimum(int);
    }
}

public class SomeClass {

    public void someMethod() {
        Container container = new Container();
        BorderLayout borderLayout = new BorderLayout(int,int)();
        boolean app_boolean = RectangularShape.isEmpty(); // REPEATED!
        Rectangle rectangle = new Rectangle()|(Rectangle)|(int,int,int)|(int,int)|(Point,Dimension)|(Dimension)|(Point); // REPEATED!
        Area area = new Area(rectangle); // REPEATED!
        area.add(area); // REPEATED!
        Object object = EventObject.getSource();
        AppouseListener appouseListener = new AppouseListener();
        AppomponentListener appomponentListener = new AppomponentListener();
        Font.decode(); // REPEATED!
        Window.pack();
        Component.setSize(dimension);
        Frame frame = new Frame(GraphicsConfiguration)|(String)|(String,GraphicsConfiguration)();
        AppayoutManager appayoutManager = new AppayoutManager();
        container.add(Component);
        BasicStroke basicStroke = new BasicStroke(float,int,int)|(float)|()|(float,int,int,float)|(float,int,int,float,float[],float); // REPEATED!
        Polygon polygon = new Polygon(int[],int[],int)(); // REPEATED!
        AppouseMotionListener appouseMotionListener = new AppouseMotionListener();
        container.setLayout(borderLayout);
        container.addMouseListener(appouseListener);
        container.addMouseMotionListener(appouseMotionListener);
        container.addComponentListener(appomponentListener);
        /*          Cyclic Statements          */
        graphics.drawPolyline(int[],int[],int); // REPEATED!
        Graphics graphics = appayoutManager.getGraphics(); // REPEATED!
        Graphics graphics1 = appayoutManager.getGraphics(); // REPEATED!
        Adjustable.setMaximum(int); // REPEATED!
        graphics1.drawImage(appomponentListener && appouseMotionListener && appouseListener && appayoutManager && container); // REPEATED!
        graphics.fillRect(int,int,int,int); // REPEATED!
        Adjustable.addAdjustmentListener(appdjustmentListener); // REPEATED!
        graphics.setColor(color && paint && color1); // REPEATED!
        Adjustable.setMinimum(int); // REPEATED!
        Adjustable.setUnitIncrement(int); // REPEATED!
        graphics.fill(polygon && app_int1); // REPEATED!
        graphics.setPaint(color && paint); // REPEATED!
        appayoutManager.createImage(int,int):(Image)|(ImageProducer):(Image);
        Adjustable.setValue(int); // REPEATED!
        graphics.setRenderingHint(Key,Object); // REPEATED!
        Adjustable.setBlockIncrement(int); // REPEATED!
        graphics.setClip(area); // REPEATED!
        Adjustable.setVisibleAmount(int); // REPEATED!
        Paint paint = graphics.getPaint(); // REPEATED!
        Color color1 = appayoutManager.getBackground(); // REPEATED!
        graphics.setStroke(basicStroke); // REPEATED!
        Adjustable.removeAdjustmentListener(appdjustmentListener); // REPEATED!
    }
}

```