PaintApp.java — Full Code with Explanations

Import required Java Swing and AWT classes

```
import javax.swing.*;              // For GUI components like JFrame, JPanel, JButton
import java.awt.*;                 // For drawing (Graphics, Color, etc.)
import java.awt.event.*;           // For handling mouse events and actions
import java.util.ArrayList;        // For storing a list of drawing points
```

PaintApp Class (Main Window)

```
public class PaintApp extends JFrame {
```

- This defines the main class PaintApp.
- It extends JFrame, which is a top-level window in Swing.

```
private DrawArea drawArea = new DrawArea();         // Custom panel where the user draws
 private JButton clearButton = new JButton("Clear");  // A button to clear the drawing
```

- drawArea is where the drawing happens (a custom panel).
- clearButton is a button to erase everything on the canvas.

```
 public PaintApp() {
```

- Constructor for the main window

```
setTitle("Simple Paint App");       // Sets the title of the window
setSize(800, 600);                  // Sets the window size (width x height)
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // Exit app when window is closed
setLayout(new BorderLayout());      // Use BorderLayout to arrange components

    JPanel topPanel = new JPanel();     // Create a panel to hold the Clear button
    topPanel.add(clearButton);          // Add the button to the top panel

    add(topPanel, BorderLayout.NORTH);  // Place the topPanel at the top of the window
    add(drawArea, BorderLayout.CENTER); // Place the drawing area in the center

  clearButton.addActionListener(e -> drawArea.clear()); // When button is clicked, clear canvas

    setVisible(true);                   // Make the window visible on screen
  }

  public static void main(String[] args) {
        SwingUtilities.invokeLater(PaintApp::new); // Start the GUI safely on the Event Dispatch
Thread
    }
}
```

DrawArea Class (Canvas for Drawing)

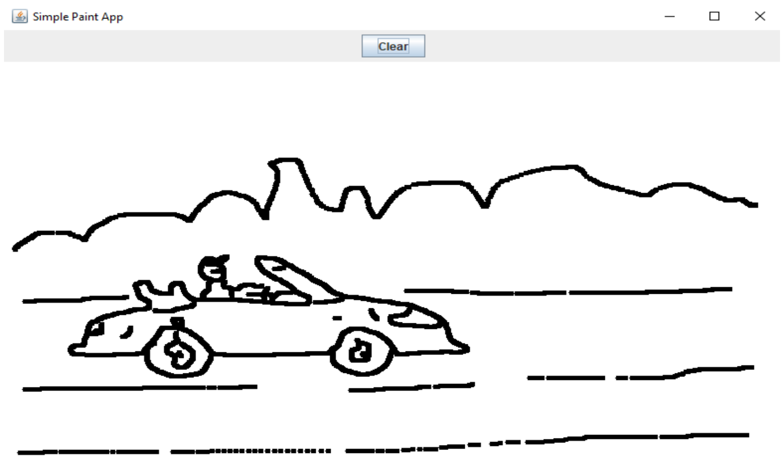
```
class DrawArea extends JPanel {
    private ArrayList<Point> points = new ArrayList<>(); // Stores all the points the user draws

    public DrawArea() {
        setBackground(Color.WHITE);   // Set the background color of the canvas to white

        addMouseMotionListener(new MouseMotionAdapter() {
            public void mouseDragged(MouseEvent e) {
                points.add(e.getPoint()); // Add current mouse point to the list
                repaint();              // Refresh the screen to draw the new point
            }
        });
    }
```

- This listens for mouse drag events and stores the cursor location each time it's dragged

```
    public void clear() {
        points.clear();    // Clear all stored points
        repaint();         // Repaint the screen (it will be empty now)
    }

    protected void paintComponent(Graphics g) {
        super.paintComponent(g);    // Clear the previous drawings

        g.setColor(Color.BLACK);    // Set color to black

        for (Point p : points) {    // For every point stored
            g.fillOval(p.x, p.y, 6, 6); // Draw a small black circle at that point
        }
    }
}
```

Concept	Purpose
JFrame	Main application window
JPanel	Custom panel for drawing
MouseMotionListener	Detects mouse drag to track drawing points
Graphics	Used to draw shapes (circles, lines, etc.)
ArrayList<Point>	Stores all the dots drawn
repaint()	Requests the panel to refresh/redraw

screenshot



Refence

Official & Trusted Resources

1. **Oracle Official Java Tutorials** 🌐 https://docs.oracle.com/javase/tutorial/uiswing/
2. **w3schools – Java Swing Tutorial** 🌐 https://www.w3schools.com/java/java_swing.asp
3. **GeeksforGeeks – Java Swing Tutorials** 🌐 https://www.geeksforgeeks.org/java-swing/
4. **Tutorials Point – Java Swing** 🌐 https://www.tutorialspoint.com/swing/index.htm

YouTube Channels

1.Programming with Mosh (YouTube)

2. thenewboston 🌐 https://www.youtube.com/user/thenewboston