🌟 Java GUI Notes (Swing-Based)

📌 What is a GUI?

A **GUI (Graphical User Interface)** allows users to interact with programs using graphical elements like **windows, buttons, text fields**, etc., instead of just typing commands.

In Java, GUI development is commonly done using **Swing**, which is part of the javax.swing package.

🧱 1. Getting Started with Swing

🧰 Important Swing Components:

Component	Description
JFrame	Main window
JLabel	Displays text/images
JButton	Clickable button
JTextField	Single-line input field
JTextArea	Multi-line input field
JCheckBox	Checkbox
JRadioButton	Radio button
JComboBox	Drop-down list
JPanel	Groups multiple components

✅ First GUI Program

🎯 Goal: Display a simple window with a label

```java
import javax.swing.*;

public class HelloGUI {
    public static void main(String[] args) {
        JFrame frame = new JFrame("My First GUI");
        JLabel label = new JLabel("Hello, World!", SwingConstants.CENTER);

        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.add(label); // add label to frame
        frame.setVisible(true); // show frame
    }
}
```

🔍 Code Explanation:

- **JFrame**: The main application window.
- **JLabel**: Used to show a line of text.
- **setSize()**: Sets width and height of the frame.
- **setDefaultCloseOperation()**: Closes app when "X" is clicked.
- **add()**: Adds the label to the frame.
- **setVisible(true)**: Displays the window on screen.

👆 2. Adding Interactivity with JButton

🎯 Goal: Show a message when a button is clicked

```java
import javax.swing.*;
import java.awt.event.*;

public class ButtonExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Button Example");
        JButton button = new JButton("Click Me");

        button.setBounds(100, 100, 120, 40);

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(null, "You clicked the button!");
            }
        });

        frame.setSize(300, 300);
        frame.setLayout(null); // absolute positioning
        frame.add(button);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

🔍 Code Explanation:

- **JButton**: Creates a clickable button.
- **addActionListener()**: Handles click events.
- **JOptionPane**: Pops up a message dialog box.
- **setBounds()**: (x, y, width, height) – position and size.

🧱 3. Layout Managers

📦 **Common Layouts**

Layout	Description
FlowLayout	Left-to-right, wraps around
BorderLayout	NORTH, SOUTH, EAST, WEST, CENTER
GridLayout	Equal-sized grid layout
BoxLayout	Line up components in a row/column

🧪 Example: BorderLayout

```java
import javax.swing.*;
import java.awt.event.*;

public class ButtonExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Button Example");
        JButton button = new JButton("Click Me");

        button.setBounds(100, 100, 120, 40);

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(null, "You clicked the button!");
            }
        });

        frame.setSize(300, 300);
        frame.setLayout(null); // absolute positioning
        frame.add(button);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setVisible(true);
    }
}
```

📋 4. Input Fields (JTextField)

🎯 Goal: Read input from a text field and show it in a dialog.

```
 1. import javax.swing.*;
 2. import java.awt.event.*;
 3.
 4. public class TextFieldExample {
 5.     public static void main(String[] args) {
 6.         JFrame frame = new JFrame("Input Example");
 7.         JLabel label = new JLabel("Enter your name:");
 8.         JTextField textField = new JTextField();
 9.         JButton button = new JButton("Submit");
10.
11.         label.setBounds(50, 50, 120, 30);
12.         textField.setBounds(180, 50, 150, 30);
13.         button.setBounds(130, 100, 100, 30);
14.
15.         button.addActionListener(e -> {
16.             String name = textField.getText();
17.             JOptionPane.showMessageDialog(null, "Hello, " + name + "!");
18.         });
19.
20.         frame.add(label);
21.         frame.add(textField);
22.         frame.add(button);
23.         frame.setLayout(null);
24.         frame.setSize(400, 200);
25.         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
26.         frame.setVisible(true);
27.     }
28. }
29.
```

✅ 5. Checkboxes and Radio Buttons

```
 1. import javax.swing.*;
 2.
 3. public class CheckRadioExample {
 4.     public static void main(String[] args) {
 5.         JFrame frame = new JFrame("Options Example");
 6.
 7.         JCheckBox checkBox = new JCheckBox("Subscribe to newsletter");
 8.         checkBox.setBounds(50, 50, 200, 30);
 9.
10.         JRadioButton male = new JRadioButton("Male");
11.         JRadioButton female = new JRadioButton("Female");
12.         male.setBounds(50, 90, 100, 30);
13.         female.setBounds(150, 90, 100, 30);
14.
15.         ButtonGroup genderGroup = new ButtonGroup();
16.         genderGroup.add(male);
17.         genderGroup.add(female);
18.
19.         frame.add(checkBox);
20.         frame.add(male);
21.         frame.add(female);
22.         frame.setLayout(null);
23.         frame.setSize(300, 200);
24.         frame.setVisible(true);
25.         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
26.     }
27. }
```

🔍 Explanation:

- JCheckBox: Allows selecting multiple options.
- JRadioButton + ButtonGroup: Allows selecting only one option at a time.

🎯 6. ComboBox (Drop-down list)

```
 1. import javax.swing.*;
 2.
 3. public class ComboBoxExample {
 4.     public static void main(String[] args) {
 5.         JFrame frame = new JFrame("ComboBox Example");
 6.
 7.         String[] countries = {"India", "Canada", "USA", "Germany"};
 8.         JComboBox<String> comboBox = new JComboBox<>(countries);
 9.
10.         comboBox.setBounds(100, 50, 150, 30);
11.         frame.add(comboBox);
12.
13.         frame.setSize(300, 200);
14.         frame.setLayout(null);
15.         frame.setVisible(true);
16.         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
17.     }
18. }
19.
```

🧠 7. Events and Listeners

Common Listeners:

- **ActionListener** – Used with buttons
- **ItemListener** – For checkboxes/comboboxes
- **MouseListener** – Mouse clicks, enter/exit
- **KeyListener** – Keyboard input

Listeners let you **react to user actions**.

🎨 8. Custom Drawing with JPanel

```java
import javax.swing.*;
import java.awt.*;

class MyPanel extends JPanel {
    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.drawString("This is custom drawing!", 50, 50);
    }
}

public class CustomDrawing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Custom Drawing");
        MyPanel panel = new MyPanel();

        frame.add(panel);
        frame.setSize(300, 200);
        frame.setVisible(true);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}

```

📚 9. Advanced GUI: Multi-Window GUI

```java
import javax.swing.*;
import java.awt.event.*;

public class MultiWindow {
    public static void main(String[] args) {
        JFrame mainFrame = new JFrame("Main Window");
        JButton openButton = new JButton("Open New Window");

        openButton.setBounds(100, 100, 200, 40);

        openButton.addActionListener(e -> {
            JFrame newWindow = new JFrame("Second Window");
            JLabel label = new JLabel("Hello from second window");
            label.setBounds(50, 50, 200, 30);
            newWindow.add(label);
            newWindow.setSize(300, 200);
            newWindow.setLayout(null);
            newWindow.setVisible(true);
        });

        mainFrame.add(openButton);
        mainFrame.setSize(400, 300);
        mainFrame.setLayout(null);
        mainFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        mainFrame.setVisible(true);
    }
}

```

10. Good Practices for GUI in Java

- **Use Layout Managers instead of null layout for dynamic resizing.**
- **Group related components using JPanel.**
- **Always run GUI using:**

```
1. SwingUtilities.invokeLater(() -> {
2.     // your code here
3. });
4.
```