Week 1: GUI Programming - Swing & AWT Components

Introduction to GUI Programming

In Java, **Graphical User Interface (GUI)** programming allows developers to create interactive applications where users can interact with graphical elements like buttons, text fields, labels, and more. There are two main libraries used for GUI programming in Java:

1. **AWT (Abstract Window Toolkit)**: Java's original platform-independent windowing, graphics, and user-interface toolkit.
2. **Swing**: A more modern, versatile, and lightweight toolkit built on AWT, which provides a richer set of GUI components.

AWT vs Swing

AWT	Swing
Platform-dependent (uses native OS widgets)	Platform-independent (uses Java to render components)
Limited features and flexibility	Provides more components and features (e.g., tables, trees, sliders)
Components are heavyweight (depend on the OS)	Components are lightweight (handled by Java itself)
Examples: Button, TextField, Label	Examples: JButton, JTextField, JLabel
Poorer look and feel	Supports pluggable look-and-feel mechanisms

Why Use Swing?

- **Lightweight**: Swing components are not reliant on platform-specific code, making them portable.
- **Customizable**: You can change the appearance of Swing components by customizing or subclassing them.
- **Rich Component Set**: Swing includes components not found in AWT, such as JTable, JTree, and JSlider.
- **Pluggable Look-and-Feel**: The look and feel of Swing applications can be changed at runtime without changing the code.

Core Swing Components

1. JFrame: The main window that holds other components.

- Example: A window where all GUI components will be displayed.

2. JButton: A button that performs an action when clicked.

- Example:

```java
JButton button = new JButton("Click Me");
```

3. JLabel: A component that displays a short string or an image.

- Example:

```java
JLabel label = new JLabel("Hello, World!");
```

4. JTextField: A single-line text input field.

- Example:

```java
JTextField textField = new JTextField("Enter your name");
```

5. JPanel: A container that can group other components.

- Example:

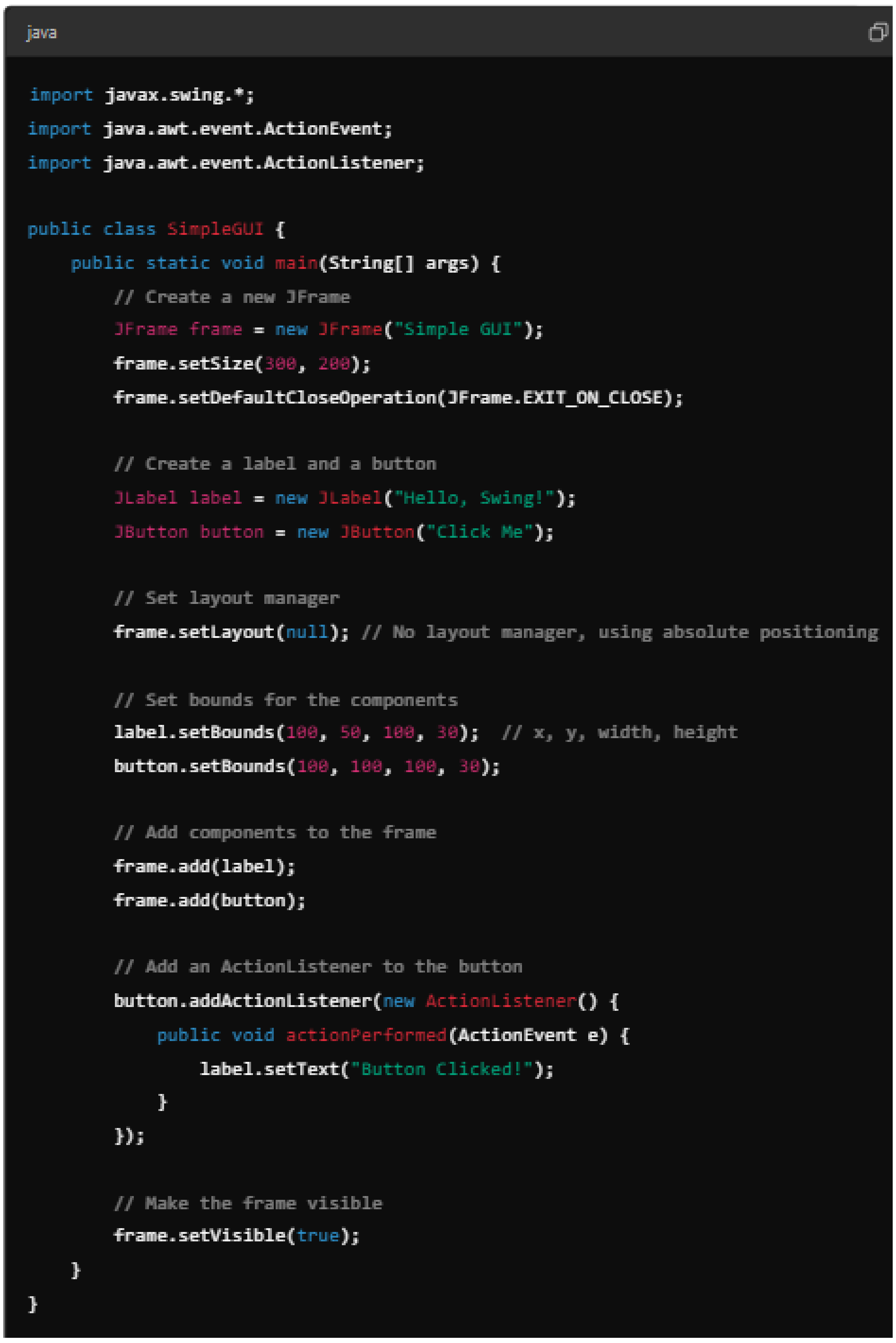
```java
JPanel panel = new JPanel();
```

6. Layout Managers: Swing provides various layout managers to control how components are arranged, such as:

- **FlowLayout** (default for JPanel): Aligns components in a row.
- **BorderLayout** (default for JFrame): Arranges components in five regions (North, South, East, West, Center).

Basic Example: Creating a Simple GUI

Here’s a basic example of a Swing application that creates a window with a button and a label.



java

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

public class SimpleGUI {
    public static void main(String[] args) {
        // Create a new JFrame
        JFrame frame = new JFrame("Simple GUI");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Create a label and a button
        JLabel label = new JLabel("Hello, Swing!");
        JButton button = new JButton("Click Me");

        // Set layout manager
        frame.setLayout(null); // No layout manager, using absolute positioning

        // Set bounds for the components
        label.setBounds(100, 50, 100, 30);  // x, y, width, height
        button.setBounds(100, 100, 100, 30);

        // Add components to the frame
        frame.add(label);
        frame.add(button);

        // Add an ActionListener to the button
        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                label.setText("Button Clicked!");
            }
        });

        // Make the frame visible
        frame.setVisible(true);
    }
}
```

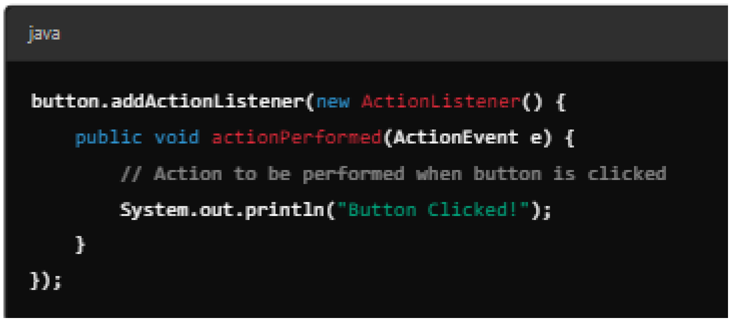
Event Handling in Swing

In Swing, events occur when users interact with the GUI (e.g., clicking a button). Event handling is a mechanism that catches these events and processes them.

Steps for Event Handling:

1. **Source**: The object on which the event occurs (e.g., a button).
2. **Event Object**: Encapsulates information about the event.
3. **Listener**: Receives the event and processes it. It must implement the appropriate interface (e.g., ActionListener for button clicks).

Example of handling button clicks:



```java
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        // Action to be performed when button is clicked
        System.out.println("Button Clicked!");
    }
});
```

Exercises for Week 1

Exercise 1: Create a Simple Swing Application

- **Objective**: Create a Java application with a JFrame containing a JLabel and a JButton. When the button is clicked, the label text should change.
- **Steps**:
    1. Create a JFrame and set its size.
    2. Add a JButton and a JLabel to the frame.
    3. Use an ActionListener to detect when the button is clicked and update the label's text.
- **Expected Output**: A window with a button and a label. Clicking the button changes the label's text.

Exercise 2: Adding Multiple Components to a JPanel

- **Objective**: Create a GUI application where you have a JPanel containing multiple components (a label, a text field, and a button).
- **Steps**:
    1. Create a JFrame and set its layout to BorderLayout.
    2. Create a JPanel and add a JLabel, JTextField, and JButton to it.
    3. Add the JPanel to the frame.
    4. Add an event handler that changes the text of the JLabel when the button is clicked.

- **Expected Output**: A window with a panel. The panel contains a label, text field, and button. Clicking the button changes the label’s text.

Exercise 3: Experiment with Layout Managers

- **Objective**: Modify the layout of a simple Swing application using different layout managers.
- **Steps**:
    1. Create a JFrame with FlowLayout, BorderLayout, and GridLayout and observe the differences.
    2. Add at least three components to each layout and see how they arrange.
- **Expected Output**: A better understanding of how different layout managers control the position of components in a Swing application.

Advanced Reading

- **Java Swing Tutorial (Official)**: https://docs.oracle.com/javase/tutorial/uiswing/
- **Java Documentation on JFrame, JButton, JLabel**: Java API Docs

Recommended Books

- Holzner Steven (2005), *JAVA 2 Programming Black Book*, DreamTech
- Wigglesworth and Lumby (2002), *JAVA Programming*, NCC

Conclusion

In Week 1, we have introduced GUI programming using Java's Swing and AWT toolkits. You have learned how to create simple GUI applications, use basic components like JLabel, JButton, and JTextField, and handle user interactions using event listeners. You are encouraged to practice the exercises to solidify your understanding.