Event Handling in Java Using AWT & Swing Components

In Java, **Abstract Window Toolkit (AWT)** and **Swing** provide various components to create Graphical User Interfaces (GUIs). These components include Frame, Panel, Button, Label, Checkbox, TextField, and more.

Component in Java (AWT & Swing)

What is a Component in Java?

- In Java, **Component** is the base class for all graphical user interface (GUI) elements that can be displayed on the screen.
- It is part of the **Abstract Window Toolkit (AWT)** package.
- All GUI elements such as **Button, Label, TextField, Checkbox, Panel, List, Choice, and Canvas** are subclasses of Component.
- It provides methods to set size, color, font, visibility, and event handling.

Key Features of Component

- **Graphical Representation:** Can be displayed on the screen.
- **User Interaction:** Can accept user inputs (e.g., Button click, text input).
- **Event Handling:** Listens to user actions like mouse clicks or keyboard presses.
- **Hierarchy:** It is the parent class for all AWT components.
- **Container Dependency:** Must be added inside a Container like Frame or Panel.

Real-Life Example of a Component

Example Scenario: "Order Coffee in a Café"

- Imagine you visit a café where you order coffee.
- The café's **touchscreen kiosk** (like a McDonald's self-ordering machine) displays:
    - A **Label** saying "Select Your Coffee."
    - A **Button** named "Order Now."
    - A **TextField** to enter your name.
    - A **Checkbox** for "Add Sugar."

Java Program Example of a AWT Component

```
import java.awt.*;


public class CoffeeOrderApp {
    public static void main(String[] args) {
        // Create a Frame (Container)
        Frame frame = new Frame("Coffee Order");


        // Create Components
        Label label = new Label("Select Your Coffee:");
        Button orderButton = new Button("Order Now");
        TextField nameField = new TextField("Enter your name here", 20);
        Checkbox sugarCheckbox = new Checkbox("Add Sugar");


        // Set Layout and Add Components
        frame.setLayout(new FlowLayout());
        frame.add(label);
        frame.add(nameField);
        frame.add(sugarCheckbox);
        frame.add(orderButton);


        // Frame Properties
        frame.setSize(300, 200);
        frame.setVisible(true);
    }
}
```

Container in Java (AWT & Swing)

What is a Container in Java?

- A **Container** in Java is a special type of Component that **can hold other components** inside it.
- It is an essential part of the **Abstract Window Toolkit (AWT)**.
- Containers help in **grouping components** (like buttons, labels, text fields) to **manage layout** efficiently.

Types of Containers

1. **Top-Level Containers** (can exist independently):
   - Frame
   - Window
   - Dialog
2. **General-Purpose Containers** (must be placed inside another container):
   - Panel
   - ScrollPane

Key Features of a Container

- **Holds multiple components** inside it.
- **Manages layout** of the components (FlowLayout, GridLayout, BorderLayout, etc.).
- **Can be nested** (one container inside another).
- **Supports event handling** for user interactions.

Real-Life Example of a Container

Example Scenario: "Self-Checkout Kiosk at a Supermarket"

- Imagine you are at a **self-checkout machine** at a supermarket.
- The touchscreen UI contains:
  - A **Panel** displaying **Item Selection** (buttons for bread, milk, eggs).
  - Another **Panel** displaying the **Total Price & Checkout Button**.
  - The **Panels** are inside the **Main Frame** (the screen).

Window in Java (AWT & Swing)

What is a Window in Java?

- A **Window** in Java is a **top-level container** in **Abstract Window Toolkit (AWT)**.
- It is a subclass of Container and does **not** have a title bar, menu bar, or border by default.
- Unlike Frame, a Window cannot exist independently; it requires another Frame or Window as its **owner**.

Key Features of a Window

✓ **Top-level container** (it does not need to be placed inside another container).
✓ **No title bar or borders** (unlike Frame).
✓ **Used for splash screens, popup windows, and custom dialogs.**
✓ **Cannot be minimized or maximized** (unless it's a subclass like Frame or Dialog).
✓ **Requires an existing Frame or Window as a parent.**

Real-Life Example of a Window

Scenario: "ATM Withdrawal Pop-up Confirmation"

- You use an **ATM** to withdraw cash.
- After entering the amount, a **pop-up appears**, asking:
  - **"Do you want to continue with this transaction?"**
  - **Two buttons:** ✅ Yes | ❌ No
- This pop-up does **not** have a title bar or minimize/maximize buttons—just a confirmation dialog.

A Window in Java behaves similarly—it **creates a temporary, borderless pop-up**.

When to Use Window?

✅ **Pop-ups or Confirmation Boxes:** ATM confirmations, exit prompts.
✅ **Splash Screens:** Introductory screens before opening the main app.
✅ **Custom Dialogs:** Borderless alerts.

Conclusion

- A Window is a **top-level container** without a **title bar**.
- It **requires a parent** (Frame or another Window).
- Used for **pop-ups, confirmation messages, and splash screens**.
- Can be **customized with event handling** (like Yes/No buttons).

Frame and Panel in Java (AWT) –

1. Frame in Java

What is a Frame?

- A **Frame** is a top-level window in **Java AWT** that contains a **title bar, border, and buttons** (minimize, maximize, close).
- It is the **main window** of a Java GUI application.
- It **inherits from Window**, meaning it does not need a parent container.

Key Features of Frame

✓ **Can exist independently** (unlike Window).
✓ **Supports title bars, menu bars, and borders.**
✓ **Can contain multiple components (buttons, labels, text fields, etc.).**
✓ **Can be resized, minimized, maximized, and closed.**
✓ **Uses layouts to arrange components (FlowLayout, GridLayout, etc.).**

Real-Life Example of a Frame

Example Scenario: "Banking Application Window"

- When you **open a banking app**, the main screen appears with:
  - A **title bar** showing the bank name.
  - A **menu bar** with options like **Home, Transactions, Logout**.
  - Buttons to **Deposit, Withdraw, Transfer**.
- This entire **application window** is a **Frame** in Java.

Comparison: Frame vs Window

Feature	**Frame**	**Window**
Title Bar	✅ Yes	❌ No
Can Exist Independently?	✅ Yes	❌ No
Has Minimize, Maximize, Close?	✅ Yes	❌ No
Used For	Main application window	Pop-ups, splash screens

2. Panel in Java

What is a Panel?

- A **Panel** is a general-purpose container used inside a Frame or another Container.
- It **groups components** together.
- It does **not** have a **title bar, borders, or menu**.

Key Features of Panel

✓ **Used to organize components** inside a Frame.
✓ **Cannot exist independently** (must be placed inside another container).
✓ **Supports different layouts (FlowLayout, GridLayout, etc.).**
✓ **Used for creating sections inside a GUI.**

Real-Life Example of a Panel

Example Scenario: "ATM Screen Layout"

- In an **ATM machine**, different sections are displayed:
  - **Account Info Panel** (Balance, Account Number).
  - **Transaction Panel** (Deposit, Withdraw buttons).
  - **Receipt Panel** (Transaction Summary).

Each **section is a Panel** in Java.

Comparison: Frame vs Panel

Feature	**Frame**	**Panel**
Top-Level Container?	✅ Yes	❌ No
Has Title Bar?	✅ Yes	❌ No
Can Contain Other Components?	✅ Yes	✅ Yes
Used For	Main application window	Grouping UI elements

Use of AWT Controls in Java

AWT (Abstract Window Toolkit) provides various GUI components (controls) to create interactive applications. These include **Labels, Buttons, Checkboxes, Checkbox Groups, TextFields, and TextAreas**. Let's understand each in detail with **real-life examples** and Java **code examples**.

---

1. Label in Java AWT

What is a Label?

- A **Label** is a non-editable text component used to display **information or instructions**.
- It cannot be modified by the user.

Real-Life Example of Label

✓ **Online Forms:** "Enter your Name", "Password", "Email".
✓ **ATMs:** "Available Balance: $5000".
✓ **Mobile Apps:** "Welcome to WhatsApp".

2. Button in Java AWT

What is a Button?

- A **Button** is an interactive component that **triggers an action** when clicked.

Real-Life Example of Button

✓ **Websites:** "Submit", "Login", "Sign Up" buttons.
✓ **ATMs:** "Withdraw", "Deposit", "Check Balance".
✓ **Mobile Apps:** "Send Message" in WhatsApp.

3. Checkbox in Java AWT

What is a Checkbox?

- A **Checkbox** is a component that allows users to **select or deselect** an option.
- Multiple checkboxes can be selected.

Real-Life Example of Checkbox

✓ **Online Forms:** "I accept Terms & Conditions".
✓ **Shopping Websites:** "Filter by Brand - Samsung, Apple, Sony".
✓ **Mobile Apps:** "Enable Notifications", "Dark Mode".

4. Checkbox Group in Java AWT

What is a Checkbox Group?

- A **CheckboxGroup** creates **radio buttons**, allowing **only one selection**.
- Unlike checkboxes, users **cannot select multiple options**.

Real-Life Example of Checkbox Group

✓ **Online Forms:** "Select Gender - Male / Female / Other".
✓ **Pizza Order:** "Select Size - Small / Medium / Large".
✓ **ATMs:** "Select Account Type - Savings / Current".

5. TextField in Java AWT

What is a TextField?

- A **TextField** is a one-line text input field.
- Used for **user input** (name, email, password, etc.).

Real-Life Example of TextField

✓ **Login Forms:** "Enter Username", "Enter Password".
✓ **Online Search Boxes:** "Search for Products".
✓ **Mobile Apps:** "Enter Mobile Number".

6. TextArea in Java AWT

What is a TextArea?

- A **TextArea** is a **multi-line text input field**.
- Used for **comments, descriptions, or feedback**.

Real-Life Example of TextArea

✓ **Feedback Forms:** "Write your feedback".
✓ **Messaging Apps:** "Type your message".
✓ **Code Editors:** Writing large pieces of text.

AWT Control	**Used For**
Label	Displaying text (not editable)
Button	Performing actions on click
Checkbox	Selecting multiple options
CheckboxGroup	Selecting **only one** option (radio buttons)
TextField	One-line text input
TextArea	Multi-line text input

Java AWT Program: Designing a User Registration Form

```java
import java.awt.*;

import java.awt.event.*;


public class RegistrationForm {

    public static void main(String[] args) {

        // Create Frame

        Frame frame = new Frame("User Registration Form");


        // Set Layout

        frame.setLayout(new GridLayout(8, 2, 10, 10)); // 8 rows, 2 columns, spacing


        // Create Components

        Label nameLabel = new Label("Full Name:");

        TextField nameField = new TextField(20);


        Label emailLabel = new Label("Email:");

        TextField emailField = new TextField(20);


        Label genderLabel = new Label("Gender:");

        CheckboxGroup genderGroup = new CheckboxGroup();

        Checkbox male = new Checkbox("Male", genderGroup, false);

        Checkbox female = new Checkbox("Female", genderGroup, false);


        Label languageLabel = new Label("Known Languages:");

        Checkbox java = new Checkbox("Java");

        Checkbox python = new Checkbox("Python");

        Checkbox cpp = new Checkbox("C++");


        Label countryLabel = new Label("Country:");
        Choice countryChoice = new Choice();
        countryChoice.add("India");
        countryChoice.add("USA");
        countryChoice.add("UK");
        countryChoice.add("Australia");

        Label addressLabel = new Label("Address:");
        TextArea addressArea = new TextArea(3, 20);

        Button submitButton = new Button("Submit");
        Label messageLabel = new Label("");

        // Event Handling for Submit Button
        submitButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                String name = nameField.getText();
                String email = emailField.getText();
                String gender = genderGroup.getSelectedCheckbox() != null ?
genderGroup.getSelectedCheckbox().getLabel() : "Not Selected";
                String languages = (java.getState() ? "Java " : "") + (python.getState() ? "Python " : "") +
(cpp.getState() ? "C++" : "");
                String country = countryChoice.getSelectedItem();
                String address = addressArea.getText();

                messageLabel.setText("Registration Successful!");
                System.out.println("Registration Details:");
                System.out.println("Name: " + name);
                System.out.println("Email: " + email);
                System.out.println("Gender: " + gender);
                System.out.println("Languages: " + languages);
                System.out.println("Country: " + country);
                System.out.println("Address: " + address);
            }
        });

        // Add Components to Frame
        frame.add(nameLabel);       frame.add(nameField);
        frame.add(emailLabel);      frame.add(emailField);
        frame.add(genderLabel);     frame.add(male); frame.add(new Label("")); frame.add(female);
        frame.add(languageLabel);   frame.add(java); frame.add(new Label("")); frame.add(python);
frame.add(new Label("")); frame.add(cpp);
        frame.add(countryLabel);    frame.add(countryChoice);
        frame.add(addressLabel);    frame.add(addressArea);
        frame.add(submitButton);    frame.add(messageLabel);

        // Set Frame Properties
        frame.setSize(400, 400);
        frame.setVisible(true);

        // Close Frame on Window Close
        frame.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                frame.dispose();
            }
        });
    }
}
```

Output

```
+-----------------------------------+
| User Registration Form          |
|-----------------------------------|
| Full Name:    [ ________________ ] |
| Email:        [ ________________ ] |
| Gender:      ( ) Male  ( ) Female |
| Known Languages:                |
| [ ] Java  [ ] Python  [ ] C++      |
| Country:    [ India ▼ ]           |
| Address:                        |
| [ ________________________ ]        |
| [ ________________________ ]        |
| [ ________________________ ]        |
| [ Submit ]  Registration Successful! |
+----------------------------------+
```