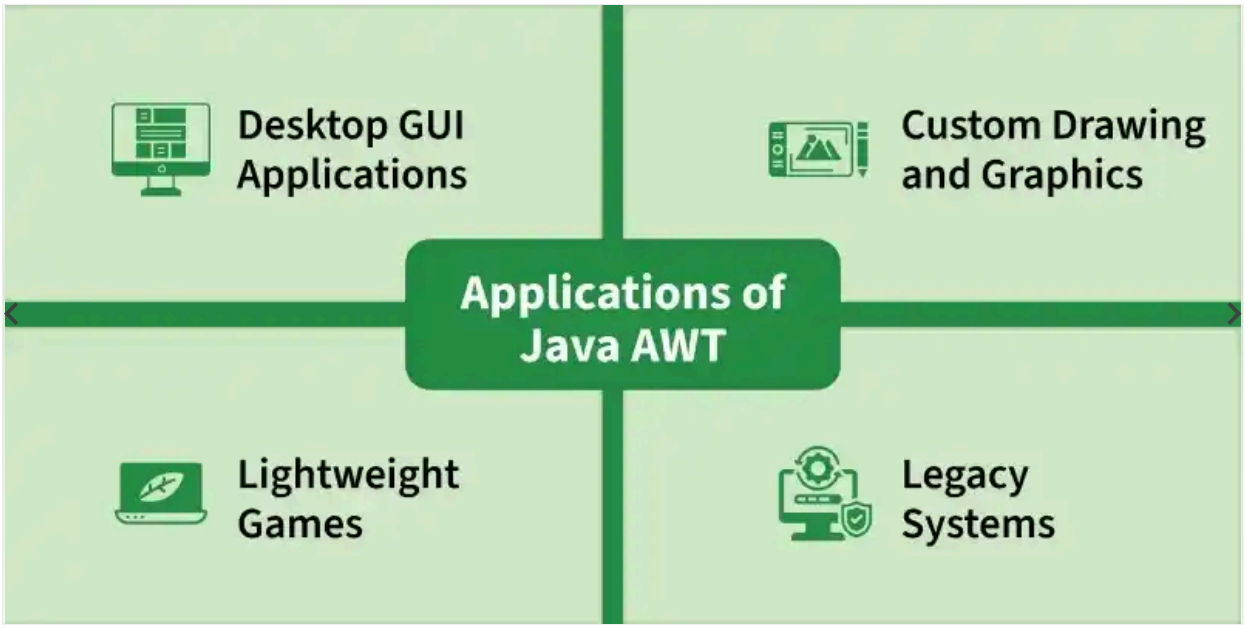
Java AWT Tutorial

Last Updated : 11 Jun, 2026

Java AWT (Abstract Window Toolkit) is a GUI toolkit provided by Java for creating window-based applications. It is a part of the Java Foundation Classes (JFC) and is included in the java.awt package. AWT provides a set of components such as buttons, labels, text fields, checkboxes, and menus that help developers build graphical user interfaces.

- Part of the java.awt package.
- Provides GUI components such as Button, Label, TextField, Checkbox, Choice, List, and Menu.
- Uses native operating system resources for rendering.



Types of Containers in Java AWT

Containers are special AWT components that can hold and organize other GUI components such as buttons, labels, text fields, and checkboxes. Java AWT provides the following main types of containers:

1. Window

border, or menu bar. It serves as the base class for other top-level containers such as Frame and Dialog.

2. Panel

A Panel is a lightweight container used to group and organize related components within a window or frame. It is commonly used to divide a user interface into sections.

3. Frame

A Frame is a top-level container that includes a title bar, border, and optional menu bar. It is the most commonly used container for creating standalone AWT applications.

4. Dialog

A Dialog is a temporary pop-up window used to interact with the user, such as displaying messages, warnings, confirmations, or collecting user input. It is usually associated with a parent frame.

Common AWT Components & Constructors

Java AWT provides a variety of GUI components that help developers build interactive desktop applications. Some commonly used AWT components are:

- **Label** : Displays a single line of read-only text.
- **Button** : Represents a clickable button that performs an action when pressed.
- **TextField** : Allows users to enter and edit a single line of text.
- **Checkbox** : Enables users to select or deselect an option.
- **CheckboxGroup** : Groups multiple checkboxes so that only one option can be selected at a time.
- **Choice** : Provides a drop-down list from which users can select an item.
- **List :** Displays a list of items and allows single or multiple selections.
- **Canvas** : Provides a blank area for custom drawing and graphics.
- **Scrollbar** : Allows users to scroll content vertically or horizontally.
- **MenuItem & Menu** : Used to create menus and menu items in a menu bar.
- **PopupMenu** : Displays a context-sensitive menu when triggered by the user.
- **Panel** : Acts as a container for organizing and grouping components.
- **Toolkit** : Provides access to platform-specific GUI resources and utility methods.

3. Layout Managers in Java AWT

Layout managers (FlowLayout, BorderLayout, GridLayout, CardLayout) control component arrangement in containers.

- FlowLayout : Arranges components in a left-to-right sequence and moves them to the next line when space runs out.
- BorderLayout: Divides the container into five regions: North, South, East, West, and Center.
- GridLayout : Arranges components in a grid of equally sized rows and columns.
- CardLayout: Displays one component at a time, allowing users to switch between multiple screens or views.

4. Event Handling Components in Java AWT

Event handlers such as ActionListener, MouseListener, ItemListener, KeyListener and WindowListener are used to capture user actions and execute the corresponding response in GUI applications.

- ActionListener: Handles action events such as button clicks and menu item selections.
- Mouse and MouseMotion Listener :Detect mouse events such as clicks, presses, releases, movement, and dragging
- ItemListener: Responds to item state changes in components like checkboxes and choice menus.
- KeyListener: Captures keyboard events when keys are pressed, released, or typed
- WindowListener: Handles window-related events such as opening, closing, minimizing, and activating a window

5. Java AWT Examples

1. Hello World in Java AWT

Hello, World is was the first step in learning Java. So, let us program our first Program in Java AWT as Hello World using Labels and Frames.

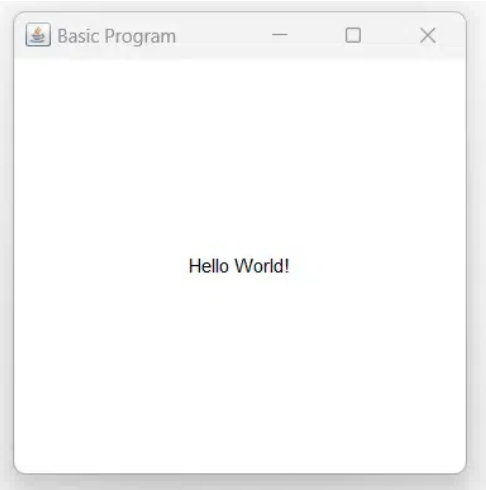
Below is the implementation of the above method:

```
import java.awt.*;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

// Driver Class
public class AWT_Example {
    // main function
    public static void main(String[] args)
    {
        // Declaring a Frame and Label
        Frame frame = new Frame("Basic Program");
        Label label = new Label("Hello World!");

        // Aligning the label to CENTER
        label.setAlignment(Label.CENTER);

        // Adding Label and Setting the Size of the Frame
        frame.add(label);
        frame.setSize(300, 300);

        // Making the Frame visible
        frame.setVisible(true);

        // Using WindowListener for closing the window
        frame.addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }
}
```

Running:

javac AWT_Example.java
java AWT_Example

Output:



2. Java AWT Program to create Button

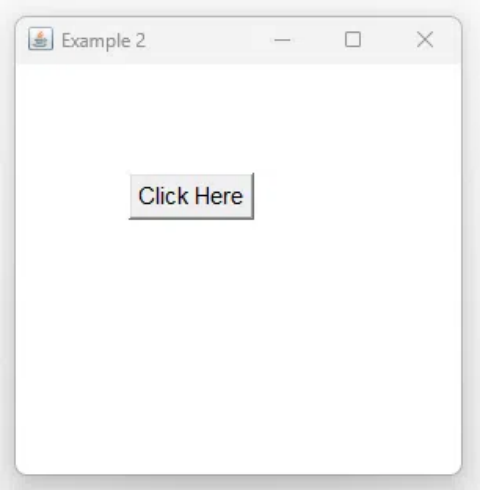
Below is the implementation of the Java AWT Program to create a Button:

```
import java.awt.*;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Button_Example {
    // main function
    public static void main(String[] args)
    {
        // Creating instance of frame with the label
        Frame frame = new Frame("Example 2");

        // Creating instance of button with label
        Button button = new Button("Click Here");

        // Setting the position for the button in frame
        button.setBounds(80, 100, 64, 30);

        // Adding button to the frame
        frame.add(button);

        // setting size, layout and visibility of frame
        frame.setSize(300, 300);
        frame.setLayout(null);
        frame.setVisible(true);

        // Using WindowListener for closing the window
        frame.addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }
}
```

Run

javac Button_Example.java
java Button_Example

Output:



Advantages of Java AWT

- Simple and easy to learn for beginners.
- Provides a rich set of basic GUI components.
- Uses native operating system controls, giving a familiar look and feel.
- Supports event handling through listener interfaces.
- Built into Java, so no additional libraries are required.
- Lightweight in terms of API complexity compared to modern GUI frameworks.

AWT vs Swing

Feature	AWT	Swing
Package	java.awt	javax.swing
Components	Heavyweight	Lightweight
Look and Feel	Platform-dependent	Platform-independent
Customization	Limited	Highly customizable
Performance	Uses native controls	Pure Java implementation
Modern Usage	Limited	More widely used

Comment

K kartik 38

GeeksforGeeks
Sanchhaya Education Private Limited

Corporate & Communications Address:

A-143, 6th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305)

Registered Address:

K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

Company
About Us
Legal
Privacy Policy
Careers
Contact Us
Corporate Solution
Campus Training

Explore
POTD
Practice Problems
Blogs
Upskill Courses
Connect

Tutorials
Programming Languages
DSA
Web Technology
AI, ML & Data Science
DevOps
CS Core Subjects
GATE
School Subjects
Software and Tools

Courses
ML and Data Science
DSA and Placements
Web Development
Data Science
Programming Languages
DevOps & Cloud
GATE
MongoDB
Certifications

Preparation Corner
Interview Corner
Aptitude
Puzzles
GfG 160
System Design

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved