Chapter - 3

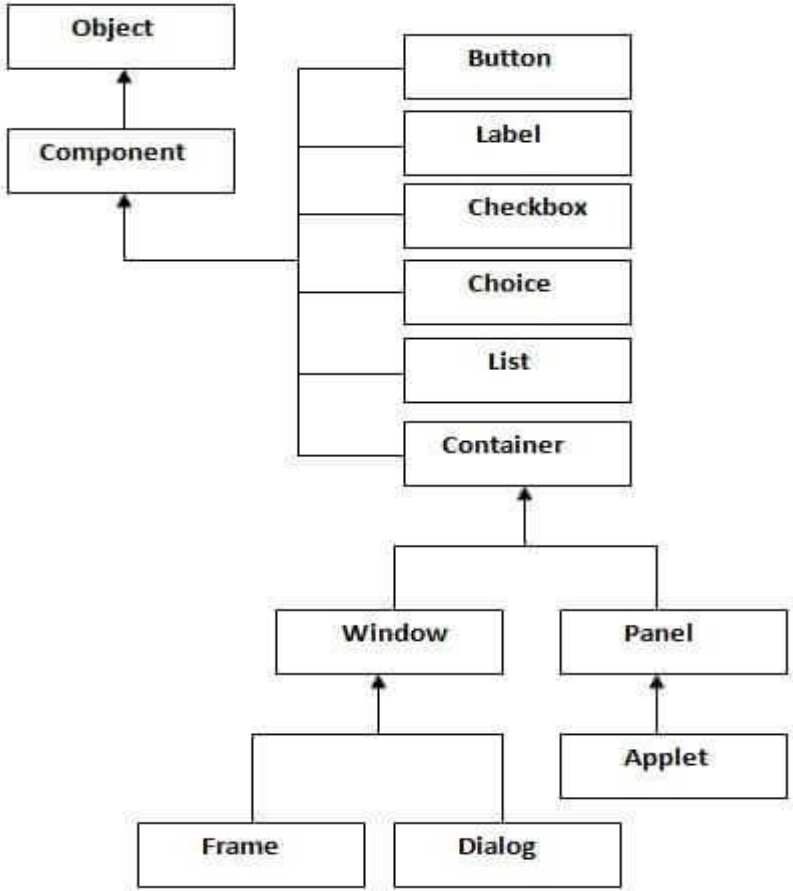
Abstract Window ToolKit (AWT)

Introduction

- The Abstract Window Toolkit (AWT) was Java's first GUI framework, and it has been part of Java since version 1.0.
- It contains numerous classes and methods that allow you to create windows and simple controls.
- Java AWT (Abstract Windowing Toolkit) is an API to develop GUI or window-based application in java.
- Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system.
- AWT is heavyweight i.e. its components uses the resources of system.
- The java.awt package provides classes for AWT api such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

AWT Classes

- The AWT classes are contained in the java.awt package.
- It is one of Java's largest packages.
- It contains:
  - Event Class
  - EventListeners
  - AdapterClass
  - Components (TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.)
  - Layouts
  - Panel
  - Frame
  - And many more


Object
Component
Button
Label
Checkbox
Choice
List
Container
Window
Panel
Applet
Frame
Dialog

Component

- At the top of the AWT hierarchy is the Component class.
- Component is an abstract class that encapsulates all of the attributes of a visual component.
- Except for menus, all user interface elements that are displayed on the screen and that interact with the user are subclasses of Component.
- It defines over a hundred public methods that are responsible for managing events, such as mouse and keyboard input, positioning and sizing the window, and repainting.
- A Component object is responsible for remembering the current foreground and background colors and the currently selected text font.

Container

- The Container class is a subclass of Component.
- It has additional methods that allow other Component objects to be nested within it.
- The Container is a component in AWT that can contain another components like buttons, textfields, labels etc.
- The classes that extends Container class are known as container such as Frame, Dialog and Panel.
- A container is responsible for laying out (that is, positioning) any components that it contains

Window

- The Window class creates a top-level window.
- A top-level window is not contained within any other object; it sits directly on the desktop.
- Generally, you won’t create Window objects directly. Instead, you will use a subclass of Window called Frame, described next.
- The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window.

Frame

- Frame encapsulates what is commonly thought of as a “window.”
- It is a subclass of Window and has a title bar, menu bar, borders, and resizing corners.
- The precise look of a Frame will differ among environments.
- A number of environments are reflected in the screen captures shown throughout this book.
- The Frame is the container that contain title bar and can have menu bars. It can have other components like button, textfield etc.

Panel

- The Panel class is a concrete subclass of Container.
- Panel is the superclass for Applet.
- Panel is a window that does not contain a title bar, menu bar, or border.
- Other components can be added to a Panel object by its
- add( ) method (inherited from Container).
- Once these components have been added, you can position and resize them manually using the setLocation( ), setSize( ), setPreferredSize( ), or setBounds( ) methods defined by Component.

Useful Methods of Component Class

Method	Description
public void add(Component c)	inserts a component on this component.
public void setSize(int width,int height)	sets the size (width and height) of the component.
public void setLayout(LayoutManager m)	defines the layout manager for the component.
public void setVisible(boolean status)	changes the visibility of the component, by default false.

Working with Frame Windows

- Here are two of Frame's constructors:

  Frame( ) throws HeadlessException

  Frame(String title) throws HeadlessException

- The first form creates a standard window that does not contain a title.
- The second form creates a window with the title specified by title.
- Notice that you cannot specify the dimensions of the window. Instead, you must set the size of the window after it has been created.
- A HeadlessException is thrown if an attempt is made to create a Frame instance in an environment that does not support user interaction. Usually this happens on a virtualbox unix system.

Setting windows dimensions

- The setSize( ) method is used to set the dimensions of the window. Its signature is shown here:

  void setSize(int newWidth, int newHeight)

  void setSize(Dimension newSize)

  The new size of the window is specified by newWidth and newHeight, or by the width and height fields of the Dimension object passed in newSize. The dimensions are specified in terms of pixels.

- The getSize( ) method is used to obtain the current size of a window. One of its forms is shown here:

  Dimension getSize( )

  This method returns the current size of the window contained within the width and height fields of a Dimension object.

Hiding and Showing a Window

- After a frame window has been created, it will not be visible until you call setVisible( ). Its signature is shown here:

  void setVisible(boolean visibleFlag)

- The component is visible if the argument to this method is true. Otherwise, it is hidden.

Closing a Frame Window

- When using a frame window, your program must remove that window from the screen when it is closed, by calling setVisible(false).
- To intercept a window-close event, you must implement the windowClosing( ) method of the WindowListener interface.
- Inside windowClosing( ), you must remove the window from the screen.

Creating a Frame Window in an AWT-Based Applet

- First, create a subclass of Frame.
- Next, override any of the standard applet methods, such as init( ), start( ), and stop( ), to show or hide the frame as needed.
- Finally, implement the windowClosing( ) method of the WindowListener interface, calling setVisible(false) when the window is closed.
- Once you have defined a Frame subclass, you can create an object of that class.
- This causes a frame window to come into existence, but it will not be initially visible. You make it visible by calling setVisible( ).
- When created, the window is given a default height and width.
- You can set the size of the window explicitly by calling the setSize( ) method.