Unit 3: Introducing the AWT: LH 3

- AWT classes
- Window fundamentals:
  - component,
  - container,
  - panel,
  - window,
  - frame,
  - console
- Working with frame windows:
  - setting windows dimensions,
  - hiding and showing and closing windows
- Creating a frame window in an applet

AWT Class

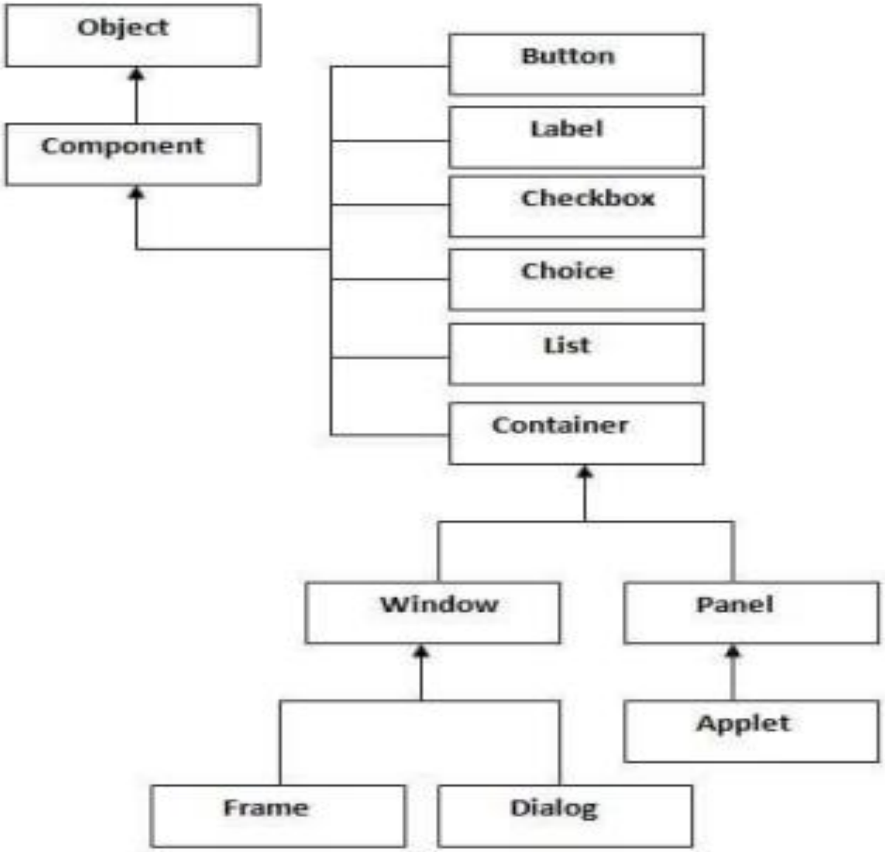
- The Java programming language class library provides a user interface toolkit called the Abstract Windowing Toolkit, or the AWT

What is a user interface

- The user interface is that part of a program that interacts with the user of the program.
- User interfaces take many forms.
- These forms range in complexity from simple command-line interfaces to the point-and-click graphical user interfaces provided by many modern applications.

- At the lowest level, the operating system transmits information from the mouse and keyboard to the program as input, and provides pixels for program output.
- The AWT was designed so that programmers don't have worry about the details of tracking the mouse or reading the keyboard, nor attend to the details of writing to the screen.
- The AWT provides a well-designed object-oriented interface to these low-level services and resources.

Hierarchy of AWT classes:



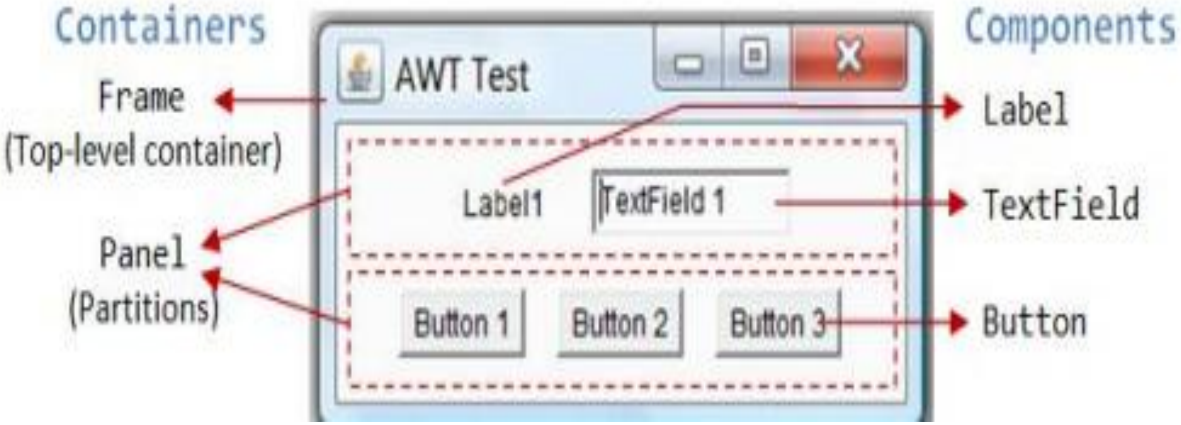
Components and containers

Components

- A graphical user interface is built of graphical elements called components.
- Typical components include such items as buttons, scrollbars, and text fields.
- Components allow the user to interact with the program and provide the user with visual feedback about the state of the program.
- In the AWT, all user interface components are instances of class Component or one of its subtypes.
- Spatially, components must fit completely within the container that contains them.
- This nesting of components (including containers) into containers creates a tree of elements, starting with the container at the root of the tree and expanding out to the leaves, which are components such as buttons.

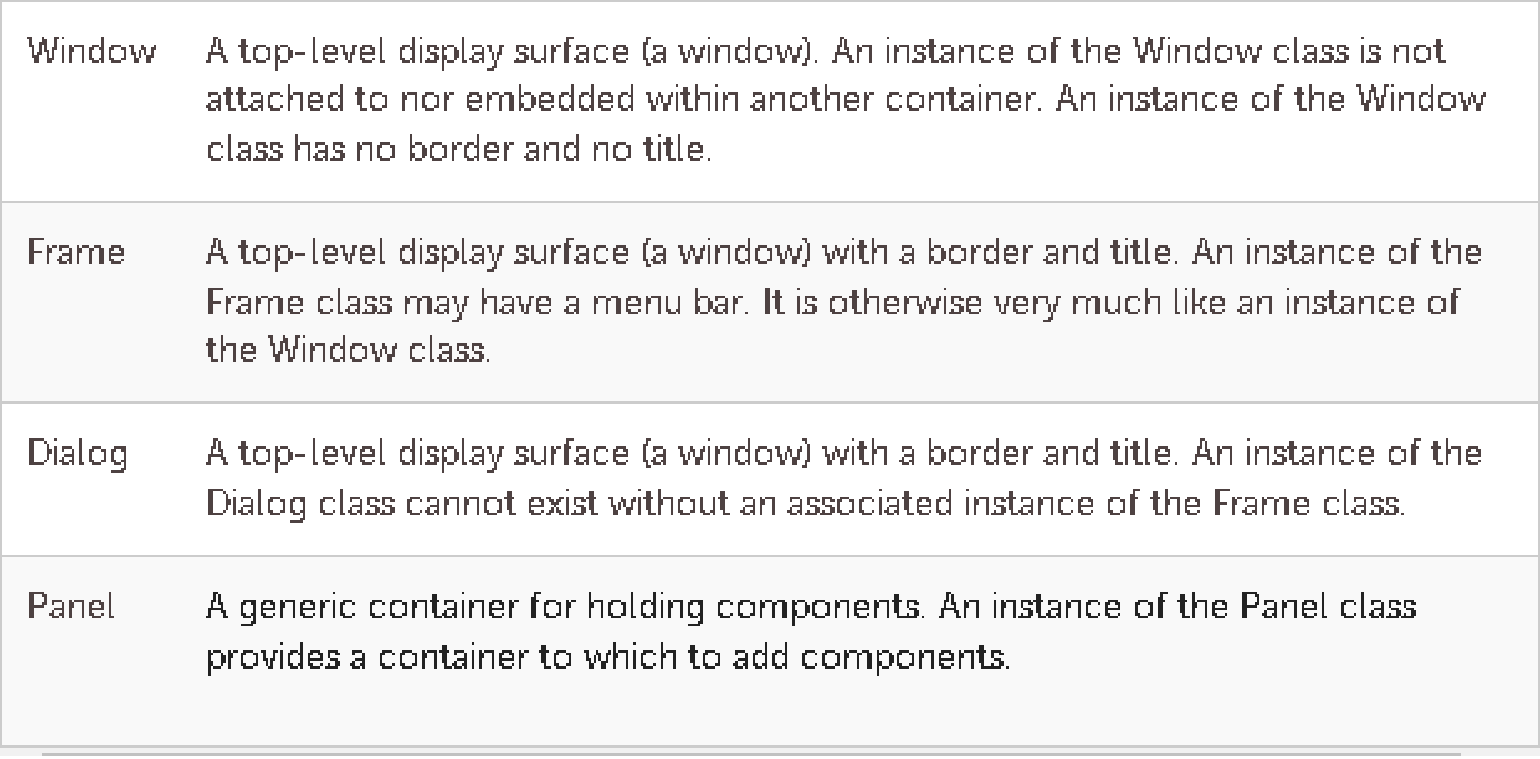
Containers

- Components do not stand alone, but rather are found within containers.
- Containers contain and control the layout of components. Containers are themselves components, and can thus be placed inside other containers.
- In the AWT, all containers are instances of class Container or one of its subtypes.


Containers
Frame
(Top-level container)
Panel
(Partitions)
AWT Test
Label1
TextField 1
Button 1
Button 2
Button 3
Components
Label
TextField
Button

Types of containers

- The AWT provides four container classes. They are class Window and its two subtypes –
- class Frame and class Dialog -- as well as the Panel class.
- In addition to the containers provided by the AWT, the Applet class is a container -- it is a subtype of the Panel class and can therefore hold components.
- Brief descriptions of each container class provided by the AWT are provided below.



Window	A top-level display surface (a window). An instance of the Window class is not attached to nor embedded within another container. An instance of the Window class has no border and no title.
Frame	A top-level display surface (a window) with a border and title. An instance of the Frame class may have a menu bar. It is otherwise very much like an instance of the Window class.
Dialog	A top-level display surface (a window) with a border and title. An instance of the Dialog class cannot exist without an associated instance of the Frame class.
Panel	A generic container for holding components. An instance of the Panel class provides a container to which to add components.

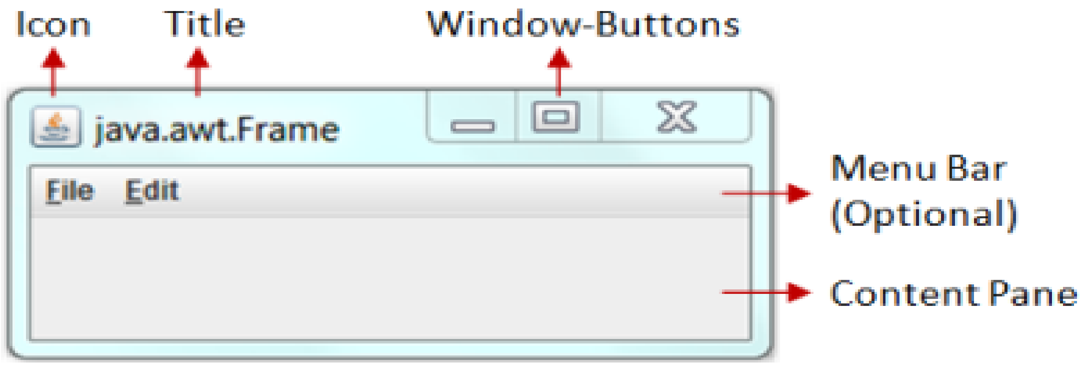
- The AWT provides nine basic non-container component classes from which a user interface may be constructed. (Of course, new component classes may be derived from any of these or from class Component itself.)
- These nine classes are class Button, Canvas, Checkbox, Choice, Label, List, Scrollbar, TextArea, and TextField

Window

- 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.

Frame

- 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.
- A Frame provides the "main window" for your GUI application.
- It has a title bar (containing an icon, a title, the minimize, maximize/restore-down and close buttons), an optional menu bar, and the content display area.



Working With Frame Window

Setting the Window’s 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.

Setting a Window’s Title

You can change the title in a frame window using **setTitle( )**, which has this general form:

void setTitle(String *newTitle*)

Here, *newTitle* is the new title for the window.

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.
- NOTE: Example Program related to Frame are already provided.

Canvas

- Although it is not part of the hierarchy for applet or frame windows, there is one other type of window that we will find valuable: Canvas.
- Derived from Component, Canvas encapsulates a blank window upon which we can draw.
- Syntax to create: **C*anvas c=new Canvas();***

Panel

- The Panel class is a concrete subclass of Container.
- Panel is the superclass for Applet.
- When screen output is directed to an applet, it is drawn on the surface of a Panel object.
- In essence, a Panel is a window that does not contain a title bar, menu bar, or border.

Syntax to create: ***Panel p=new Panel();***

Creating frames in applets

- 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.

Example:

```
import java.awt.*;
import java.applet.*;
public class FrameApplet extends Applet{
    Frame f;
    public void init()
    {
        f=new Frame("Frame Window");
        f.setSize(400,400);
    }
    public void start()
    {
        f.setVisible(true);

            }
    public void stop()
    {
        f.setVisible(false);
    }
    public void paint(Graphics g)
    {
        g.drawString("this is in applet window",50,80);
    }

}
```