

## Assignment No. 8

**Assignment Title:** Create simple application of Java AWT in which show an awt component button by setting its placement and window frame size.

**Aim:** Write a Java program that accepts four integers from the user and prints equal if all four are equal, and not equal otherwise.

**Pre-Requisites:** C/C++

**Objective:** The objective is to impart fundamentals of Java AWT (Abstract Window Toolkit) an API to develop Graphical User Interface (GUI) or windows-based applications in Java.

**Outcomes:** After learning this concept students will be able to,

1. Analyze and Work with Frame class, Colour, Fonts and layout managers
2. Design and develop interface components- Labels, Button, Text Components, Check Box, Check Box Group, Choice, List Box, Panels – Scroll Panel, Menu, Scroll Bar.

**Theory:**

### JAVA AWT

Java AWT (Abstract Window Toolkit) is an API to develop Graphical User Interface (GUI) or windows-based applications in Java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavy weight i.e. its components are using the resources of underlying operating system (OS).

The java.awt package provides classes for AWT API such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

The AWT tutorial will help the user to understand Java GUI programming in simple and easy steps. Java Operator Precedence

### Why AWT is platform independent?

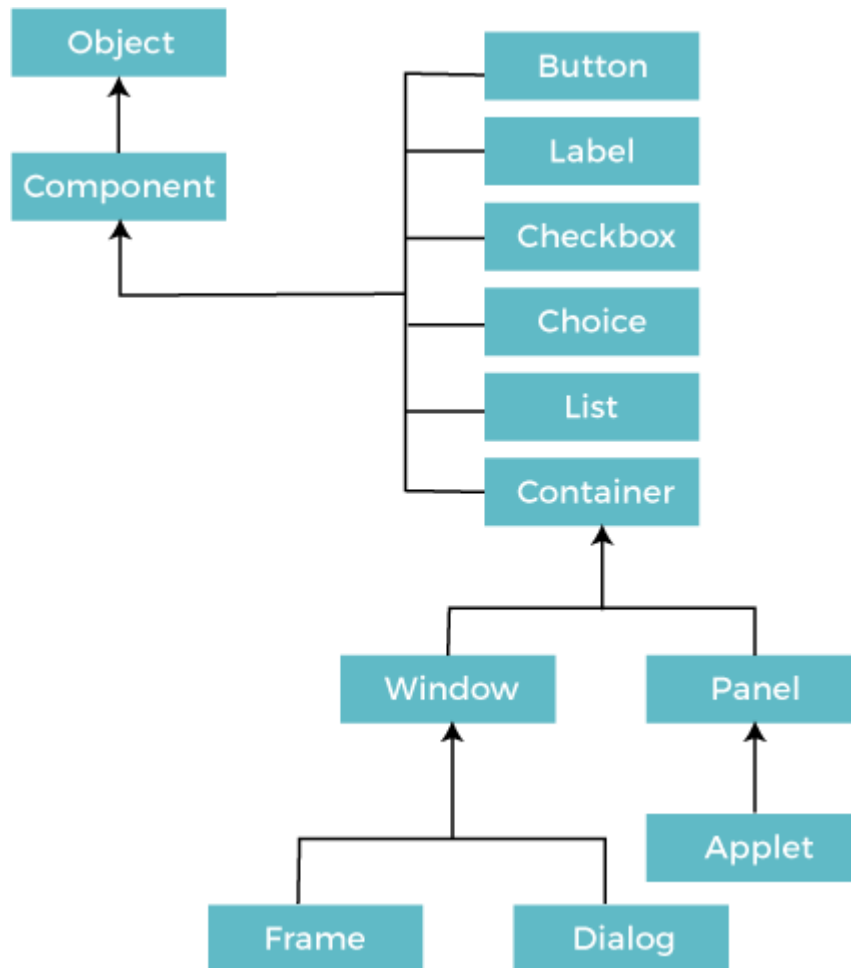
Java AWT calls the native platform calls the native platform (operating systems) subroutine for creating API components like TextField, ChechBox, button, etc.

For example, an AWT GUI with components like TextField, label and button will have different look and feel for the different platforms like Windows, MAC OS, and Unix. The reason for this is the platforms have different view for their native components and AWT directly calls the native subroutine that creates those components.

In simple words, an AWT application will look like a windows application in Windows OS whereas it will look like a Mac application in the MAC OS.

## JAVA AWT HIERARCHY

The hierarchy of Java AWT classes are given below.



There are four types of containers in Java AWT:

Window

Panel

Frame

Dialog

## WINDOW

The window is the container that have no borders and menu bars. You must use frame, dialog or another window for creating a window. We need to create an instance of Window class to create this container.

## PANEL

The Panel is the container that doesn't contain title bar, border or menu bar. It is generic container for holding the components. It can have other components like button, text field etc. An instance of Panel class creates a container, in which we can add components.

## FRAME

The Frame is the container that contain title bar and border and can have menu bars. It can have other components like button, text field, scrollbar etc. Frame is most widely used container while developing an AWT application.

## 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.

## Code:

```
import java.awt.event.*;
import javax.swing.*;
import java.util.*;

public class MyStationery extends JFrame implements ActionListener
{
    HashMap<String, String> product_and_price;

    DefaultListModel<String> product_name_bill, product_quantity_bill, product_price_bill;

    JLabel product_name, price_per_unit, quantity_label, price_of_product,
    price_per_unit_label, total, total_amount, bill_product, bill_quantity, bill_price;
    JComboBox product_list, quantity_list;
    JList product_bill, product_quantity, product_price;
    JButton add_product;

    public MyStationery()
    {
        product_and_price = new HashMap<String, String>();

        product_and_price.put("Book", "80");
        product_and_price.put("Pen", "10");
        product_and_price.put("Pencil", "7");
        product_and_price.put("Marker", "40");
        product_and_price.put("Eraser", "5");

        product_name_bill = new DefaultListModel<>();
        product_price_bill = new DefaultListModel<>();
        product_quantity_bill = new DefaultListModel<>();

        product_name = new JLabel("Product Name");
        quantity_label = new JLabel("Quantity");
        price_per_unit = new JLabel("Price Per Unit(₹)");
        price_per_unit_label = new JLabel("80");

        product_list = new JComboBox();
        quantity_list = new JComboBox();

        product_and_price.forEach((key,value)->
        {
            product_list.addItem(key);
        });

        for(int i=1; i<=10; i++)
        {
            quantity_list.addItem(Integer.toString(i));
        }

        bill_product = new JLabel("Item");
        bill_quantity = new JLabel("Quantity");
        bill_price = new JLabel("Cost");
    }
}
```

```
product_bill = new JList(product_name_bill);
product_price = new JList(product_price_bill);
product_quantity = new JList(product_quantity_bill);

add_product = new JButton("Add Product");

total = new JLabel("Total Amount : ");
total_amount = new JLabel();

product_name.setBounds(10, 30, 100, 20);
price_per_unit.setBounds(160, 30, 100, 20);
quantity_label.setBounds(310, 30, 100, 20);
price_per_unit_label.setBounds(160, 50, 100, 20);

product_list.setBounds(10, 50, 100, 20);
quantity_list.setBounds(310, 50, 100, 20);

add_product.setBounds(10, 80, 400, 50);

bill_product.setBounds(10, 150, 150, 20);
bill_quantity.setBounds(170, 150, 90, 20);
bill_price.setBounds(270, 150, 140, 20);

product_bill.setBounds(10, 175, 150, 400);
product_quantity.setBounds(170, 175, 90, 400);
product_price.setBounds(270, 175, 140, 400);

total.setBounds(140, 575, 200, 20);
total_amount.setBounds(310, 575, 90, 20);

add(product_name);
add(price_per_unit);
add(quantity_label);

add(price_per_unit_label);
add(product_list);
add(quantity_list);

add(add_product);

add(bill_product);
add(bill_quantity);
add(bill_price);

add(product_bill);
add(product_quantity);
add(product_price);

add(total);
add(total_amount);

product_list.addActionListener(this);
add_product.addActionListener(this);
```

```

        setSize(435,650);
        setLayout(null);
        setVisible(true);
        setTitle("My Stationery");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==add_product)
        {
            product_name_bill.addElement((String) product_list.getSelectedItemAt());
            product_quantity_bill.addElement((String) quantity_list.getSelectedItemAt());
            int unit_price =
Integer.parseInt(product_and_price.get(product_list.getSelectedItemAt()));
            int product_quantity = Integer.parseInt((String)
quantity_list.getSelectedItemAt());
            int total_for_product = unit_price*product_quantity;
            int total_cost = 0;
            product_price_bill.addElement(Integer.toString(total_for_product));

            for(int i=0;i<product_price_bill.getSize();i++)
            {
                total_cost += Integer.parseInt(product_price_bill.elementAt(i));
            }

            total_amount.setText(Integer.toString(total_cost));
        }
        else if(e.getSource()==product_list)
        {
            price_per_unit_label.setText(product_and_price.get(product_list.getSelectedIt
m()));
        }
    }

    public static void main(String[] args)
    {
        MyStationery ms = new MyStationery();
    }
}

```

**Output:**

[illegible]