Experiment No. 11
Implement a program on Applet or AWT Controls
Date of Performance:
Date of Submission:

Aim: Implement a program on Applet or AWT Controls

Objective: To develop application like Calculator, Games, Animation using AWT Controls.

Theory:

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.

1. A general interface between Java and the native system, used for windowing, events and layout managers. This API is at the core of Java GUI programming and is also used by Swing and Java 2D. It contains the interface between the native windowing system and the Java application1.

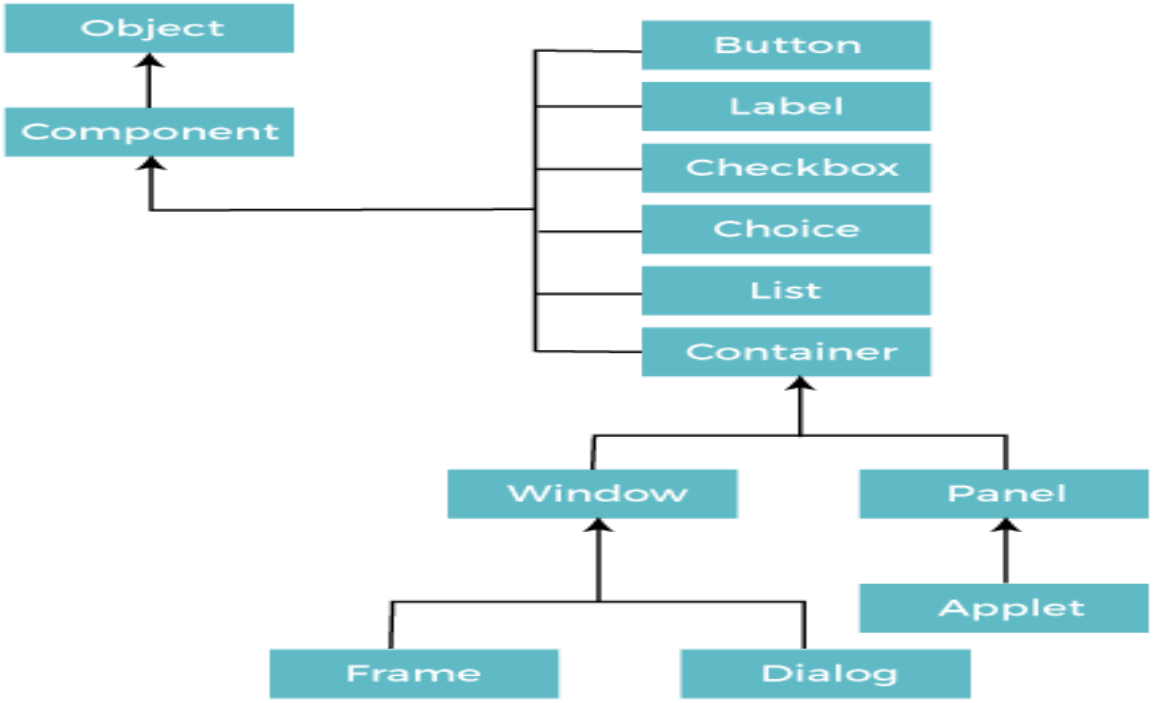
2. A basic set of GUI widgets such as buttons, text boxes, and menus1. AWT also provides Graphics and imaging tools, such as shape, color, and font classes2. AWT also avails layout managers which helps in increasing the flexibility of the window layouts2

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

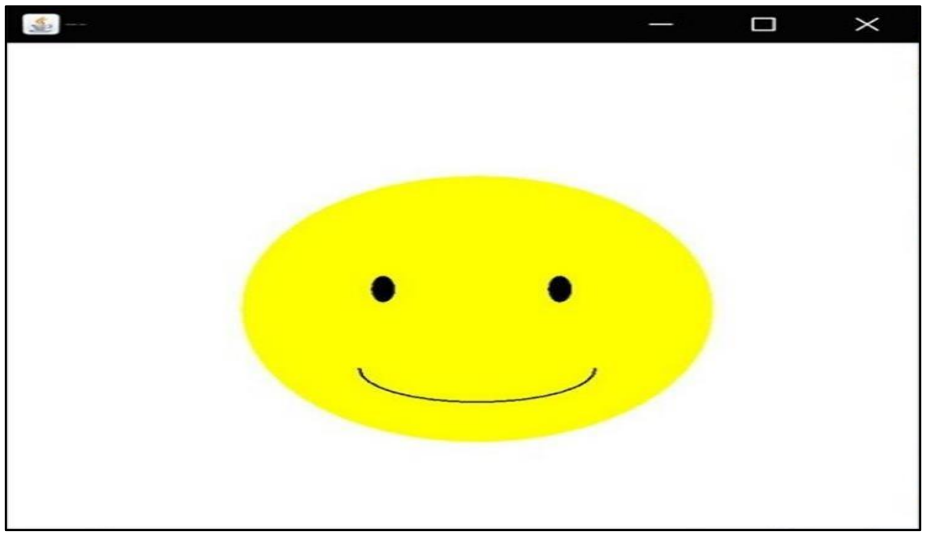


Code:

```
import java.applet.Applet;import java.awt.*;
 public class SmileyApplet extends Applet {public void paint(Graphics g) {
setBackground(Color.white); g.setColor(Color.yellow);
g.fillOval(100, 100, 200, 200);

g.setColor(Color.black); g.fillOval(155, 175, 10, 20);
g.fillOval(230, 175, 10, 20);

g.drawArc(150, 220, 100, 50, 180, 180);

}

public static void main(String[] args)

{
```

```
Frame frame = new Frame("SmileyApplet Frame");SmileyApplet applet = new
SmileyApplet(); frame.add(applet);
applet.init(); frame.setSize(400, 400);frame.setVisible(true);
frame.addWindowListener(new java.awt.event.WindowAdapter()
{
 @Override public void windowClosing(java.awt.event.WindowEventwindowEvent)
 {
System.exit(0);
 }
 });
 }
 }
```

Output:



Conclusion:

AWT controls are platform-dependent, which can make it difficult to develop applications that need to run on multiple operating systems.

AWT controls can be difficult to use, especially for complex layouts.

AWT controls are not as flexible as some of the newer GUI APIs, such as Swing and JavaFX.Overall, AWT is a good choice for developing simple GUI applications. However, it is not the best choice for developing complex or cross-platform applications. it is important to note that AWT is a legacy API, and it is no longer the preferred choice for developing new GUI applications. Swing and JavaFX are newer GUI APIs that offer more flexibility and features than AWT.