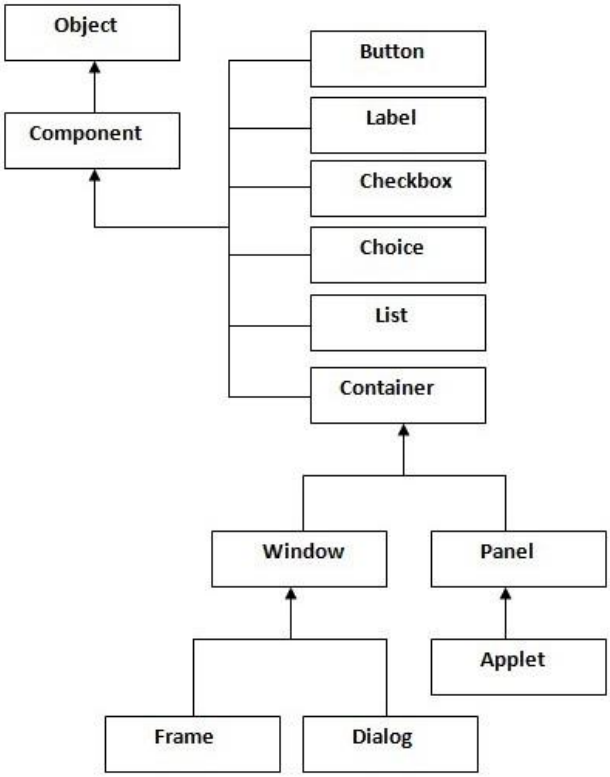
Java AWT

The **Abstract Window Toolkit (AWT) is Java's original platform-independent windowing, graphics, and user-interface widget toolkit.** The AWT classes are contained in the java.awt package.

- Contains all of the classes for creating user interfaces and for painting graphics and images.
- *an API to **develop GUI or window-based applications*** in java.

The hierarchy of Java AWT classes are shown below.



Component

A component is an object having a graphical representation that can be displayed on the screen and that can interact with the user.

Examples :

buttons, checkboxes, and scrollbars

The Component class is the abstract superclass of all user interface elements that are displayed on the screen. A Component object remembers current text font, foreground and background color.

Container

The Container class is the subclass of Component. The container object is a component that can contain other AWT components. It is responsible for laying out any components that it contains.

Window

The class Window is a top level window with no border and no menubar. The default lay- out for a window is `BorderLayout`. A window must have either a frame, dialog, or another window defined as its owner when it's constructed.

Panel

The class Panel is the simplest container class. It provides space in which an application can attach any other component, including other panels. The default layout manager for a panel is the FlowLayout layout manager

Frame

A Frame is a top-level window with a title and a border. It uses BorderLayout as default layout manager.

Dialog

A Dialog is a top-level window with a title and a border that is typically used to take some form of input from the user.

Canvas

A Canvas component represents a blank rectangular area of the screen onto which the application can draw or from which the application can trap input events from the user. An application must subclass the Canvas class in order to get useful functionality such as creat- ing a custom component. The paint method must be overridden in order to perform custom graphics on the canvas. It is not a part of hierarchy of Java AWT.

java.awt.Graphics class

The java.awt.Graphics class provides many methods for graphics programming. A graph- ics context is encapsulated by the Graphics class and is obtained in two ways:

- It is passed to an applet when one of its various methods, such as paint( ) or update( ) is called.
- It is returned by the getGraphics( ) method of Component.

Graphics Methods

The commonly used methods of Graphics class are as follows

Method	Description
abstract Graphics create()	Creates a new Graphics object that is a copy of this Graphics object
abstract void drawString(String str, int x, int y)	Draws the text given by the specified string
void drawRect(int x, int y, int width, int height)	draws a rectangle with the specified width and height
void draw3DRect(int x, int y, int width, int height, boolean raised)	Draws a 3-D highlighted outline of the specified rectangle.
abstract void drawRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)	Draws an outlined round-cornered rect- angle using this graphics context's current color

abstract void fillRect(int x, int y, int width, int height)	fill rectangle with the default color and specified width and height.
abstract void drawPolygon(int[] xPoints, int[] yPoints, int nPoints)	Draws a closed polygon defined by arrays of x and y coordinates.
abstract void fillPolygon(int[] xPoints, int[] yPoints, int nPoints)	Fills a closed polygon defined by arrays of x and y coordinates.
abstract void drawOval(int x, int y, int width, int height)	draw oval with the specified width and height.
abstract void fillOval(int x, int y, int width, int height)	fill oval with the default color and specified width and height.
abstract void drawLine(int x1, int y1, int x2, int y2)	draw line between the points(x1, y1) and (x2, y2).
abstract boolean drawImage(Image img, int x, int y, ImageObserver observer)	draw the specified image.
abstract void drawArc(int x, int y, int width, int height, int startAngle, int arc Angle)	draw a circular or elliptical arc.
abstract void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)	fill a circular or elliptical arc.
abstract void setColor(Color c)	set the graphics current color to the specified color.
abstract void setFont(Font font)	set the graphics current font to the specified font.

Example:

GraphicsDemo.java import

java.applet.Applet; import

java.awt.*;

public class GraphicsDemo extends Applet{ public

void paint(Graphics g){ g.setColor(Color.red); //

set font color g.drawString("Welcome",50, 50); //

display text

g.drawLine(120,120,200,300); // draw a line

// draw and fill rectangle

g.drawRect(170,100,60,50);

g.fillRect(170,100,60,50);

// draw and fill rounded rectangle

g.drawRoundRect(190, 10, 60, 50, 15, 15);

g.fillRoundRect(190, 10, 60, 50, 15, 15);

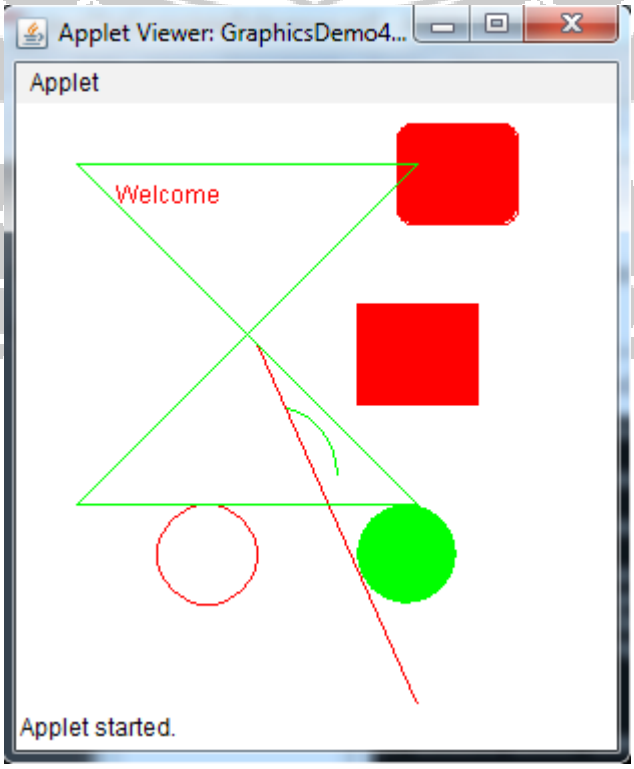
// draw and fill oval

```
g.drawOval(70,200,50,50);
g.setColor(Color.green);
g.fillOval(170,200,50,50);
// draw and fill arc
g.drawArc(90,150,70,70,0,75);
g.fillArc(270,150,70,70,0,75);
// draw a polygon
int xpoints[] = {30, 200, 30, 200, 30};
int ypoints[] = {30, 30, 200, 200, 30};
int num = 5;
g.drawPolygon(xpoints, ypoints, num);
}
}
```

Test.html

```
<html>
<body>
<applet code="GraphicsDemo4.class" width="300" height="300">
</applet>
</body>
</html>
```

Sample Output:



ROHINI COLLEGE OF ENGINEERING AND TECHNOLOGY
PALKULAM, KANYAKUMARI
OBSERVE OPTIMIZE OUTSPREAD