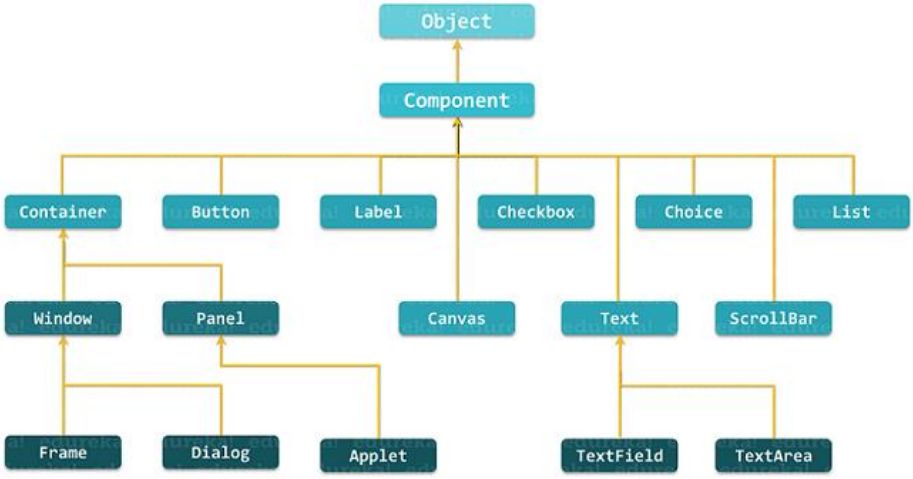
AWT (Abstract Window Toolkit): Abstract Window Toolkit acronymed as AWT is a toolkit of classes in Java which helps a programmer to develop Graphics and Graphical User Interface components. It is a part of JFC (Java Foundation Classes) developed by Sun Microsystems. The AWT API in Java primarily consists of a comprehensive set of classes and methods that are required for creating and managing the Graphical User Interface (GUI) in a simplified manner. It was developed for providing a common set of tools for designing the cross-platform GUIs. One of the important features of AWT is that it is platform dependent. This means that the AWT tools use the native toolkits of the platforms they are being implemented. This approach helps in preserving the look and feel of each platform. But as said everything comes with a price, there is a major drawback of this approach. When executed on various platforms because of platform dependency it will look different on each platform. This hampers the consistency and aesthetics of an application.

Apart from being platform-dependent, there are several other features of AWT classes about which I will be talking in the next section of this Java AWT Tutorial.



Features of awt in GUI (Graphical User Interface):

UI elements: These refers to the core visual elements which are visible to the user and used for interacting with the application. AWT in Java provides a comprehensive list of widely used and common elements.

Layouts: These define how UI elements will be organized on the screen and provide the final look and feel to the GUI.

Behavior: These define the events which should occur when a user interacts with UI elements.

There are two ways to create a GUI using Frame in AWT.

- By extending Frame class (inheritance).
- By creating the object of Frame class (association).

Container: Containers are integral part of AWT GUI components. A container provides a space where a component can be located. A Container in AWT is a component itself and it adds the capability to add

component to itself. Following are noticable points to be considered. Sub classes of Container are called as Containter. For example Panel, Frame and Window. Container can add only Component to itself.

A default layout is present in each container which can be overridden using setLayout method.

For Example: Note(To create simple awt example, you need a frame. There are two ways to create a frame in AWT.)

- By extending Frame class (inheritance): Let's see a simple example of AWT where we are inheriting Frame class. Here, we are showing Button component on the Frame.

  For Example:

```
import java.awt.*;
public class AWTExample1 extends Frame {
   AWTExample1() {
 Button b = new Button("Click Me!!");

      b.setBounds(30,100,80,30);
      add(b);
      setSize(300,300);
      setTitle("This is our basic AWT example");
      setLayout(null);
      setVisible(true);
}
public static void main(String args[]) {
AWTExample1 f = new AWTExample1();

}

}
```

AWT Example by Association: Let's see a simple example of AWT where we are creating instance of Frame class. Here, we are creating a TextField, Label and Button component on the Frame.

For Example:

```
import java.awt.*;
class AWTExample2 {
   AWTExample2() {
      Frame f = new Frame();
      Label l = new Label("Employee id:");
```

```
      Button b = new Button("Submit");
      TextField t = new TextField();

      l.setBounds(20, 70, 80, 30);
      t.setBounds(20, 100, 80, 30);
      b.setBounds(100, 100, 80, 30);
      f.add(b);
      f.add(l);
      f.add(t);
      f.setSize(400,300);
      f.setTitle("Employee info");
      f.setLayout(null);
      f.setVisible(true);
}
public static void main(String args[]) {
AWTExample2 awt_obj = new AWTExample2();
}
}
```

For Example:

Program:Note(In this program print the button in class)

```
import java.awt.*;
 class First extends Frame{
 First()
{
 Button b=new Button("click me");
 b.setBounds(30,100,80,30);       // setting button position
 add(b);                         //adding button into frame
 setSize(300,300);          //frame size 300 width and 300 height
 setLayout(null);           //no layout manager
 setVisible(true);          //now frame will be visible, by default not visible
 }
public static void main(String args[])
{
 First f=new First();
 }
}
```

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

For Example:

```
import java.awt.*;
public class PanelExample {
     PanelExample()
        {
        Frame f= new Frame("Panel Example");
        Panel panel=new Panel();
        panel.setBounds(40,80,200,200);
        panel.setBackground(Color.gray);
        Button b1=new Button("Button 1");
        b1.setBounds(50,100,80,30);
        b1.setBackground(Color.yellow);
        Button b2=new Button("Button 2");
        b2.setBounds(100,100,80,30);
        b2.setBackground(Color.green);
        panel.add(b1); panel.add(b2);
        f.add(panel);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
        }
        public static void main(String args[])
        {
        new PanelExample();
        }
}
```

Button with ActionListener : we are handling the button click events by implementing ActionListener Interface.

For Example:

```
import java.awt.*;
import java.awt.event.*;
 class But {
public static void main(String[] args) {
```

```
    Frame f = new Frame("Button Example");
    final TextField tf=new TextField();
    tf.setBounds(50,50, 150,20);

    Button b=new Button("Click Here");
        b.setBounds(50,100,60,30);
    b.addActionListener(new ActionListener()
    {
    public void actionPerformed (ActionEvent e)
    {
            tf.setText("Welcome to Advance Java Practice.");
        }
    });

    f.add(b);
    f.add(tf);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```

Label: The object of the Label class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by a programmer but a user cannot edit it directly. It is called a passive control as it does not create any event when it is accessed. To create a label, we need to create the object of Label class.

For Example:

```
import java.awt.*;

class Label1 {
public static void main(String args[])
{

    Frame f = new Frame ("Label example");
    Label l1, l2;
```

```
    // initializing the labels
    l1 = new Label ("First Label.");
    l2 = new Label ("Second Label.");

    // set the location of label
    l1.setBounds(50, 100, 100, 30);
    l2.setBounds(50, 150, 100, 30);

    // adding labels to the frame
    f.add(l1);
    f.add(l2);

    // setting size, layout and visibility of frame
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```

TextArea: The object of a TextArea class is a multiline region that displays text. It allows the editing of multiple line text. It inherits TextComponent class. The text area allows us to type as much text as we want. When the text in the text area becomes larger than the viewable area, the scroll bar appears automatically which helps us to scroll the text up and down, or right and left.

For Example:

```
import java.awt.*;
 class textarea1
{

     textarea1() {

        Frame f = new Frame();

          TextArea area = new TextArea("Welcome to javatpoint");

        area.setBounds(10, 30, 300, 300);
```

```
        f.add(area);

        f.setSize(400, 400);
        f.setLayout(null);
        f.setVisible(true);
     }

public static void main(String args[])
{
   new textarea1();
}
}
```

CheckBox: The Checkbox class is used to create a checkbox. It is used to turn an option on (true) or off (false). Clicking on a Checkbox changes its state from "on" to "off" or from "off" to "on".

For Example:

```
import java.awt.*;
public class chb
{
       chb(){
       Frame f= new Frame("CheckboxGroup Example");
        CheckboxGroup cbg = new CheckboxGroup();
        Checkbox checkBox1 = new Checkbox("C++", cbg, false);
        checkBox1.setBounds(100,100, 50,50);
        Checkbox checkBox2 = new Checkbox("Java", cbg, true);
        checkBox2.setBounds(100,150, 50,50);
        f.add(checkBox1);
        f.add(checkBox2);
        f.setSize(400,400);
        f.setLayout(null);
        f.setVisible(true);
     }
public static void main(String args[])
{
    new chb();
```

```
}
}
```

Choice: we are creating a choice menu using Choice() constructor. Then we add 5 items to the menu using add() method and Then add the choice menu into the Frame.

For Example:

```
import java.awt.*;
public class che {


        che() {
         Frame f = new Frame();
         Choice c = new Choice();

         c.setBounds(100, 100, 75, 75);


         c.add("Item 1");
         c.add("Item 2");
         c.add("Item 3");
         c.add("Item 4");
         c.add("Item 5");
         f.add(c);
         f.setSize(400, 400);
         f.setLayout(null);
         f.setVisible(true);
      }
public static void main(String args[])
{
   new che();
}
}
```

List: The object of List class represents a list of text items. With the help of the List class, user can choose either one item or multiple items. It inherits the Component class.

For Example:

```
import java.awt.*;
```

```
public class Lt
{
    // class constructor
     Lt() {
     // creating the frame
        Frame f = new Frame();
       // creating the list of 5 rows
        List l1 = new List(5);

        // setting the position of list component
        l1.setBounds(100, 100, 75, 75);

        // adding list items into the list
        l1.add("Item 1");
        l1.add("Item 2");
        l1.add("Item 3");
        l1.add("Item 4");
        l1.add("Item 5");

        // adding the list to frame
        f.add(l1);

        // setting size, layout and visibility of frame
        f.setSize(400, 400);
        f.setLayout(null);
        f.setVisible(true);
     }

// main method
public static void main(String args[])
{
   new Lt();
}
}
```

Dialog Box: The Dialog control represents a top level window with a border and a title used to take some form of input from the user. It inherits the Window class. Unlike Frame, it doesn't have maximize and minimize buttons.

For Example:

```
import java.awt.*;
import java.awt.event.*;
 class de {
     static Dialog d;
    de() {
        Frame f= new Frame();
        d = new Dialog(f , "Dialog Example", true);
        d.setLayout( new FlowLayout() );
        Button b = new Button ("OK");
        b.addActionListener ( new ActionListener()
        {
            public void actionPerformed( ActionEvent e )
            {
               de.d.setVisible(false);
            }
        });
        d.add( new Label ("Click button to continue."));
        d.add(b);
        d.setSize(300,300);
        d.setVisible(true);
    }
    public static void main(String args[])
    {
        new de();
    }
}
```