Applet Basics

* **What is an Applet?**

Applet is a special type of program that is embedded in the webpage.It runs inside the browser and works at client side. Applets are run on web browser.

**Applet** is a predefined class in **java.applet** package.  An applet must be a subclass of the java.applet.Applet class

**Advantages of Applet :**

* Applets are supported by most web browsers.
* Applets works at client side so less response time.
* **Secured:**No access to the local machine and can only access the server it came from.
* Easy to develop applet, just extends applet class.

### Drawback of Applet :

* Plugin is required at client browser to execute applet.
* **Life Cycle of an Applet :-**

1. Applet is initialized.
2. Applet is started.
3. Applet is painted.
4. Applet is stopped.
5. Applet is destroyed.

#### 

#### Lifecycle methods for Applet:

For creating any applet java.applet.Applet class must be inherited. It provides 4 life cycle methods of applet.

1. **public void init():** is used to initialized the Applet. It is invoked only once.
2. **public void start():** is invoked after the init() method or browser is maximized. It is used to start the Applet.
3. **public void stop():** is used to stop the Applet. It is invoked when Applet is stop or browser is minimized.
4. **public void destroy():** is used to destroy the Applet. It is invoked only once. It will be executed whenever applet window or browser is going to be closed .

### How to run an Applet?

There are two ways to run an applet

1. By html file.
2. By appletViewer tool (for testing purpose).

## Design applet program

**class**className**extends**Applet

{

......

// override lifecycle methods

......

}

## Running of applet using html :-

Html support a predefined tag called <applet> to load the applet program on the browser window.

### Simple example of Applet by html file:

//First.java

**import** java.applet.Applet;

**import** java.awt.Graphics;

**public** **class** First **extends** Applet

{

**public** **void** paint(Graphics g)

{

g.drawString("welcome",150,150);

}

}

### Now, compile this program by d:\>javac First.java

### myapplet.html

<html>

<body>

<applet code="First.class" width="300" height="300">

</applet>

</body>

</html>

Now,run the above html file in browser.

### Simple example of Applet by appletviewertool:

//First.java

**import** java.applet.Applet;

**import** java.awt.Graphics;

**public** **class** First **extends** Applet

{

**public** **void** paint(Graphics g)

{

g.drawString("welcome to applet",150,150);

}

 }

/\*

<applet code="First.class" width="300" height="300">

</applet>

\*/

* How to run :

D:\ >javac First.java

D:\>appletviewer First.java

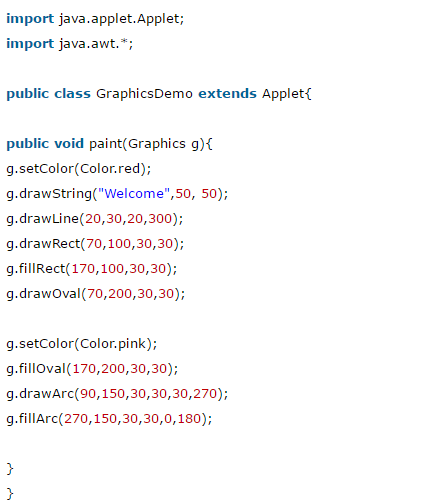
### Draw Graphics in Applet :-

java.awt.Graphics class provides many methods for graphics programming.

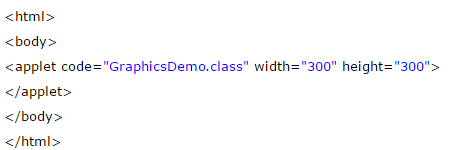
Commonly used methods of Graphics class:

1. **public abstract void drawString(String str, int x, int y):** is used to draw the specified string.
2. **public void drawRect(int x, int y, int width, int height):** draws a rectangle with the specified width and height.
3. **public abstract void fillRect(int x, int y, int width, int height):** is used to fill rectangle with the default color and specified width and height.
4. **public abstract void drawOval(int x, int y, int width, int height):** is used to draw oval with the specified width and height.
5. **public abstract void fillOval(int x, int y, int width, int height):** is used to fill oval with the default color and specified width and height.
6. **public abstract void drawLine(int x1, int y1, int x2, int y2):** is used to draw line between the points(x1, y1) and (x2, y2).
7. **public abstract booleandrawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.
8. **public abstract void drawArc(int x, int y, int width, int height, intstartAngle, intarcAngle):** is used draw a circular or elliptical arc.
9. **public abstract void fillArc(int x, int y, int width, int height, intstartAngle, intarcAngle):** is used to fill a circular or elliptical arc.
10. **public abstract void setColor(Color c):** is used to set the graphics current color to the specified color.
11. **public abstract void setFont(Font font):** is used to set the graphics current font to the specified font.

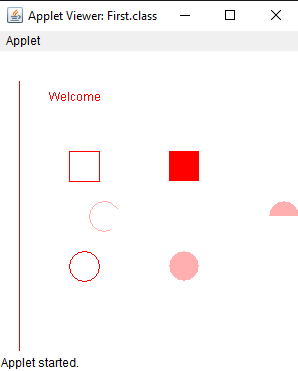
**Example :**

****

### myapplet.html

****

**Output :**

****

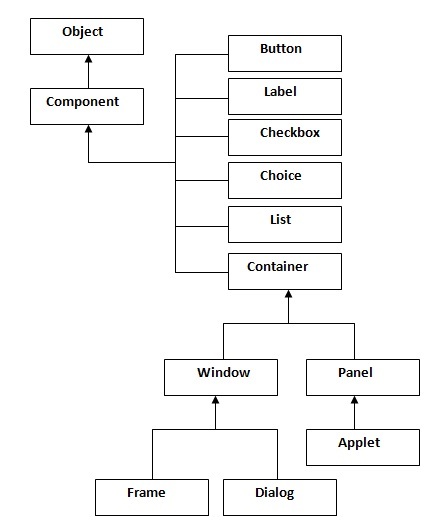
### AWT Class :-

Java AWT (Abstract Window Toolkit) is an API to develop GUI or window-based applications in java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavyweight i.e. its components are using the resources of OS.

The java.awt [package](https://www.javatpoint.com/package) provides [classes](https://www.javatpoint.com/object-and-class-in-java) for AWT api such as [TextField](https://www.javatpoint.com/java-awt-textfield), [Label](https://www.javatpoint.com/java-awt-label), [TextArea](https://www.javatpoint.com/java-awt-textarea), RadioButton, [CheckBox](https://www.javatpoint.com/java-awt-checkbox), [Choice](https://www.javatpoint.com/java-awt-choice), [List](https://www.javatpoint.com/java-awt-list) etc.

**AWT Hierarchy :**



**Container**

The Container is a component in AWT that can contain another components like buttons, textfields, labels etc. The classes that extends Container class are known as container such as Frame, Dialog and Panel.

**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.

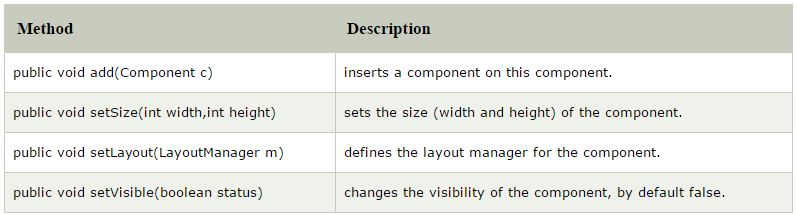
**Panel**

The Panel is the container that doesn't contain title bar and menu bars. It can have other components like button, textfield etc.

**Frame**

The Frame is the container that contain title bar and can have menu bars. It can have other components like button, textfield etc.

### Useful Methods of Component class :

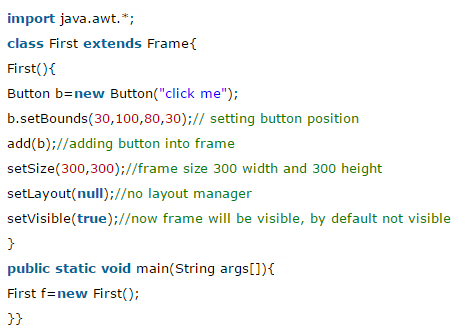
****

### Java AWT Example:

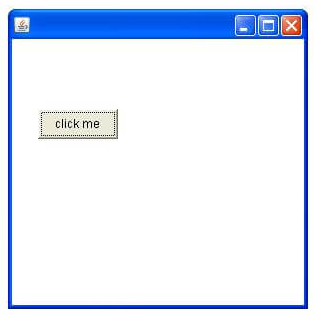
To create simple awt example, you need a frame. There are two ways to create a frame in AWT.

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

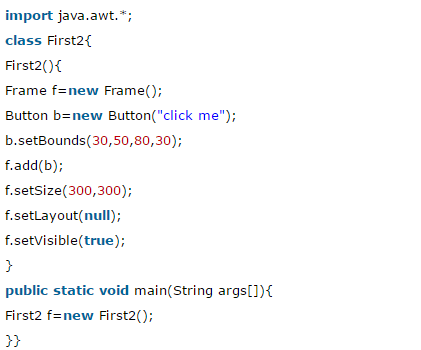
1. **Example By Inheritance :-**

****

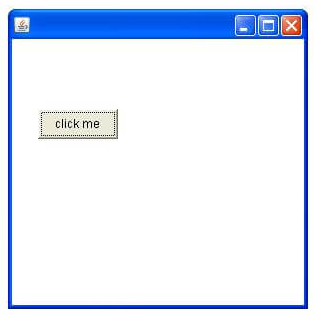
**Output :-**

****

1. **Example By Association :-**

****

**Output :-**

****

### LayoutManagers:-

The LayoutManagers are used to arrange components in a particular manner. LayoutManager is an interface that is implemented by all the classes of layout managers. There are following classes that represents the layout managers:

1. java.awt.BorderLayout
2. java.awt.FlowLayout
3. java.awt.GridLayout
4. java.awt.CardLayout
5. java.awt.GridBagLayout
6. javax.swing.BoxLayout
7. javax.swing.GroupLayout
8. javax.swing.ScrollPaneLayout
9. javax.swing.SpringLayout etc.

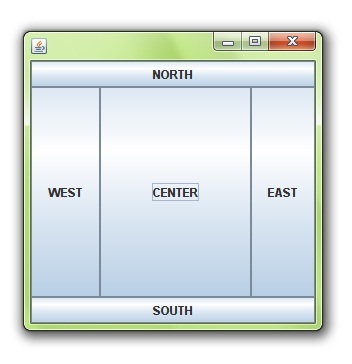
### BorderLayout:

The BorderLayout is used to arrange the components in five regions: north, south, east, west and center. Each region (area) may contain one component only. It is the default layout of frame or window. The BorderLayout provides five constants for each region:

1. public static final int NORTH
2. public static final int SOUTH
3. public static final int EAST
4. public static final int WEST
5. public static final int CENTER

Constructors of BorderLayout class:

* **BorderLayout():** creates a border layout but with no gaps between the components.
* **JBorderLayout(inthgap, intvgap):** creates a border layout with the given horizontal and vertical gaps between the components.

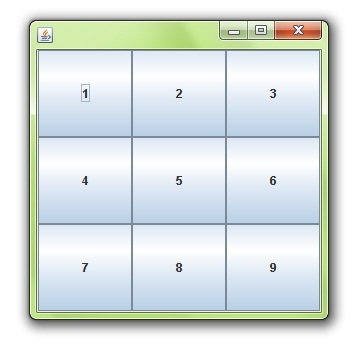


# GridLayout

|  |
| --- |
| The GridLayout is used to arrange the components in rectangular grid. One component is displayed in each rectangle. |

### Constructors of GridLayout class:

|  |
| --- |
| 1. **GridLayout():** creates a grid layout with one column per component in a row. 2. **GridLayout(int rows, int columns):** creates a grid layout with the given rows and columns but no gaps between the components. 3. **GridLayout(int rows, int columns, inthgap, intvgap):** creates a grid layout with the given rows and columns alongwith given horizontal and vertical gaps. |



# FlowLayout

|  |
| --- |
| The FlowLayout is used to arrange the components in a line, one after another (in a flow). It is the default layout of applet or panel. |

### Fields of FlowLayout class:

|  |
| --- |
| 1. **public static final int LEFT** 2. **public static final int RIGHT** 3. **public static final int CENTER** 4. **public static final int LEADING** 5. **public static final int TRAILING** |

### Constructors of FlowLayout class:

|  |
| --- |
| 1. **FlowLayout():** creates a flow layout with centered alignment and a default 5 unit horizontal and vertical gap. 2. **FlowLayout(int align):** creates a flow layout with the given alignment and a default 5 unit horizontal and vertical gap. 3. **FlowLayout(int align, inthgap, intvgap):** creates a flow layout with the given alignment and the given horizontal and vertical gap. |

