Applet :-

Applets a re small java program that are used for the internet. They are stored on an Internet Server, transported over the Internet and automatically installed and executed as a part of a Webpage on the client machine. Applets can perform arithmetic operation, display graphics, play sounds, accept user inputs, create animations and support interactive games. Before the introduction of applets, web pages were not as interactive and dynamic as today.

After an applet arise the client machine via a web page, it has a very limited access to the resources of that machine and so, it can run several GUI applications without introducing the risk of viruses.

What is HTML :-

HTML (Hyper Text Markup Language) describes the layout of a webpage. HTML is simply a vehicle to indicate the elements of a webpage. An HTML file is written using either notepad or edit or any ASCII text editor. Web pages contain several HTML tags which reflect how the webpage will look like.

Applets and HTML :-

The basic idea of how to use an applet in a webpage is very simple. The HTML tags tell the web browser which applets to load and which to display them on the webpage.

Creating an Applet :-

Step 1

Building the applet code :-

```
import java.awt.*;
import java.applet.*;
public class MyFirstApplet extends Applet {
public void paint (Graphics g) {
g.drawString ("Hello world", 20, 20);
}       }
```

Analysis of the above Applet :-

- In java any GUI based program should import the awt package because the awt package contains the relevant classes for creating the GUI.
- Any applet should extend in the applet package and hence we have to imports that package.
- Since the HTML page which embeds the applet may not remain in the same package where the applet is, the applet is to be declared as public.
- When ever we want to write or draw something on an applet window we need to override the paint() method. The paint() takes as arguments on a object of the Graphics class.
- drawingstring which written a text on the applet window and is a method of the Graphics class. Within the drawstring() method we need to specify the co-ordinates on the applet window from where the screen is to be written.

Step 2

Create the executable applet :-

Save the above applet code as MyFirstApplet.java. Compile the file to get MyFirstApplet.class.

Step 3

Design the webpage and the applet to it.

```
<html>
        <head>
                <title>This is my first applet</title>
        </head>
                <body><center>
                        <H1>Welcome to applet</H1>
                        </Center>
                        <BR><center>
                        <applet
                        code="MyFirstApplet"
                        width=400
```

```
                hight=400>
                </applet>
                </center>
            </body>
</html>
```

Save this file as MyFirstApplet.html in the same folder where MyFirstApplet.java and MyFirstApplet.class are.

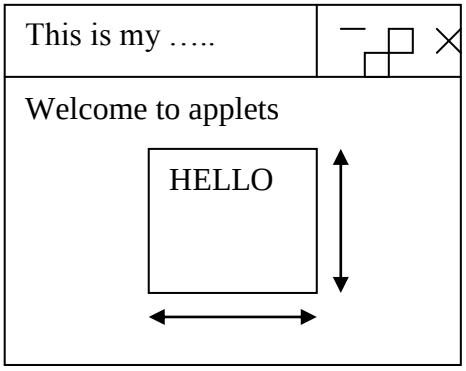
Step 4

Extending the Applet :-

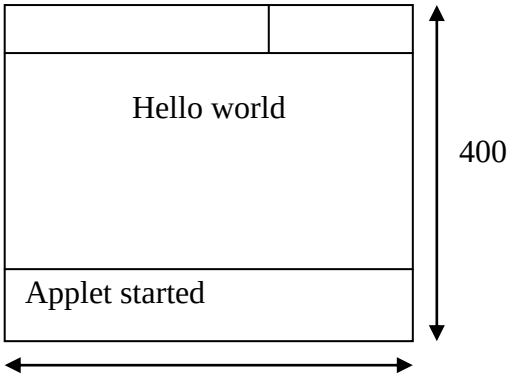
We can execute the applet in two different ways

1. By using Java enabled web browser and
2. By using applet viewer.

1. When ever we save the html file with html extension, the file will be saved as a webpage. We can view (or execute) the applet by climbing on the html file the outcome will be —



2. We can execute the applet by writing
   C:\Myjava>appletviewer MyFirstApplet.html.



Setting foreground and background colors :-

The background color of an applet can be changed by —
setBackground(Color.c);

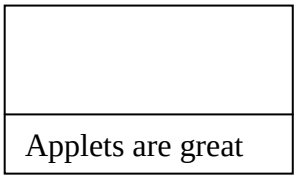
The foreground color (color of the text on the applet) can be enhanced by —
setForeground(Color.c);

where c is any valid color constant of the color class.

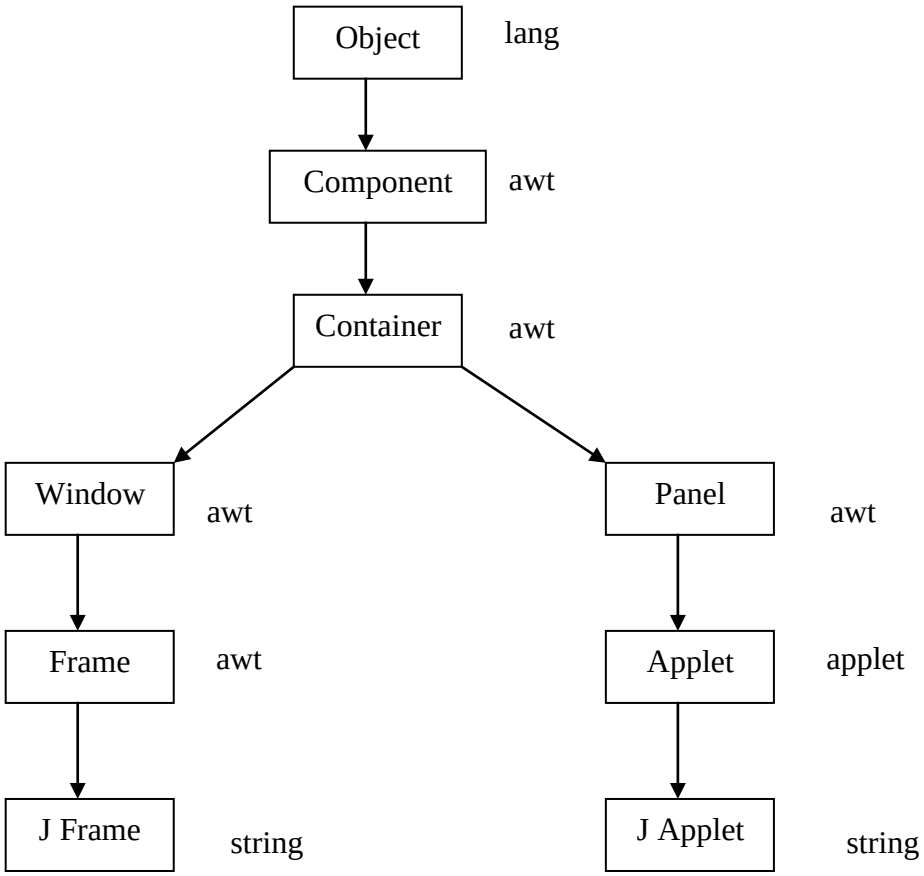
For ex.
Color.blue; Color.red; Color.cyan; Color.pink; Color.black; etc.

Using the Status Bar :-

We can use the status bar to write user defined messages out there by using— showStatus("Applets are great");



Applet class hierarchy :-

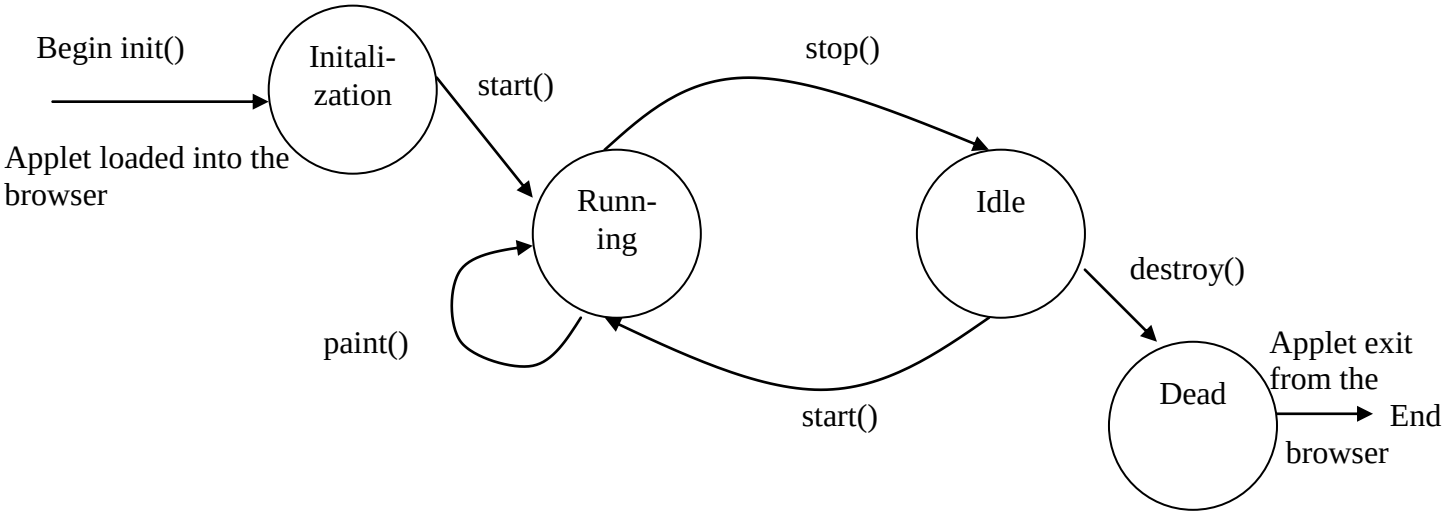


Applet life cycle :-

Thought out its life time an applet passes through the following stage.

1. Initialization.
2. Running.
3. Idle.
4. Dead.

1. **Initialization Stage :-** An applet is initialized when it is first loaded in to a web browser. It is achieved by the init() method.
2. **Running Stage :-** An applet enters the running stage when the stage method is called. The start method is called automatically by the init() method. The start() is also called when an applet is reloaded in the web browser. That is when an applet returns to running stage from an Idle stage.
3. **Idle Stage :-** An applet enters into the idle stage when the stop() method is called. The stop() is called when the web browser leaves the applet temporarily.
4. **Dead Stage :-** An applet enters the dead stage when the destroy() method is called. The destroy() method is called when the web browser quits the applet.



Applet Method :-

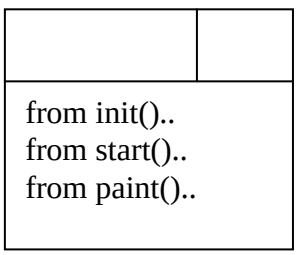
An applet contains several methods line init(), start(), paint(), destroy(), repaint(). Among these simple applets does not need to override init(), start(), stop() and destroy(). The only method that ernoy applet should override is the paint(). But complex real life applets need to override all of the above method.
The above methods are called in the following sequence :-

1. init() ->
    - Called once for all when the applet is loaded.
    - Create objects needed by the applet, initialized variables, set background and foreground colors, load images etc. within init().
2. start() ->
    - Called after init() to start the applet.
    - Also called to restart an applet after it has been stopped.
    - Create threads that will control the applet.
3. paint() ->
    - Calls after the applet being execution.
    - Called when ever the output of the applet is to be redrawn or up dated.
4. repaint ->
    - repaint() is actually called to invoke the paint method because paint can't be called explicitly.
5. stop() ->
    - Called when the browser leaves the webpage containing the applet.
    - Threads that don't need to run when the applet has been stopped are stopped rare.
6. destroy() ->
    - Called when the applet is removed from the memory.
    - Terminates all threads rare.
    - The stop() is called every time the destroy() is called.

```
/*Programs to exhibit the sequence in which the applet methods are called*/
import java.awt.*;
import java.applet.*;
/* <applet
    lode="Method Sequence"
    width=600
     hight=600>
  </applet>
*/
public class MethodSequence extends Applet {
String msg = " ";
Public void init() {
msg =msg+"from init()..";
        }
        public void start() {
        msg = msg+ "from start..()";
        public void paint(Graphics g){
        msg = msg+"from paint()..";
        g.drawString(msg,50,50);
}       }
```

Save this as MethodSequence.java
Compiled as javac MethodSequence.java
Execute this as
C:>applet viewer MethodSequence.java

O/P->



Example :-

```
/*Applet and MultiThreading*/
import java.awt.*;
import java.aplet.*;
import java.utill.*;
/* <applet
     code = "Threaded Applet"
     width=700
      height=700>
   </applet>
*/
public class ThreadedApplet extends Applet implements Runnable {
Thread t;
string msg="  Deptof C.S";
public void init() {
          setBackground(Color.gray);
          setForeground(Color.red);
          setFont(new Font ("Sudip",Font.PLAIN,20));
          }
public void start () {
          t= new Thread (this);
          t.start();
          }
public void run() {
          char ch;
          while(true){
          repaint();
          try {
                    Thread.sleep(200);
               }
          Catch (InteruptedException ie) {
                                                  }
          ch = msg.charAt(0);
          msg= msg.Substring(1,msg.length());
          msg=msg+ch;
          }
}
public void paint (Graphics g){
          g.drawString("Date="+new Date(),250,50);
          g.drawString(msg,100,300);
          }
public void stop () {
          t.stop();
          }
}
```

1. While an applet which shows an array of integers and prints the maximum and minimum of that array on the same applet.
2. Write an applet that shows a new array often every 4 sec and prints its minimum and maximum on the same array.
3. Write an applet that shows the system date and time on the top sign hand(??) cornatr of the applet and a random number along with the cumulative sum of the random number.
4. Do the problem of PING-PONG through an applet.

Example –
To determine whether a thread has finished
isAlive() :- gen1.final Boolean isAlive()—returns if the thread which is called is still running false otherwise.
wait() :- tells the calling thread to give up the monitor and go to sleep until some other thread enters the some monitor and calls notshy().
notshy() :- works up the first thread that called wait() on the same object.
sleep() :- causes the thread from which it is called to suspend execution for the specific period of milliseconds.

static void yield () :- Causes the currently executing thread to yield. If there are other runnable threads whose priority is at least as high as the priority of this thread, they will be scheduled next.

The highest priority runnable thread keeps running until it yields by calling yield() method.

Native code :- Constructor will be executed in order in which they appear in the duration of derived class that is will be executed in the order of inheritance.

Interface Vs class

Similarity

1. Both have variable and methods in them.
2. Both name represents valid java identifier.
3. Both supports inheritance.

Dissimilarity

1. A class defines what an object is ought to do and how to do, but an interface indicates what to do but not how to do. This way we can fully abstract the class name form its implementation.
2. In interface variables are final and static, methods are abstract. In class those things are not must for variable and methods.
3. Interface supplies multiple inheritances. Class supports it with the help of interface.
4. In interface access is either public or friendly. It is not mandatory to give access specifier in class each time its defines (friendly by default).

Difference class and object

1. A class is a blue print or prototype that defines variables and methods common to all objects.
   An object is a basic runtime entity of a class.
2. Class may be thought of as a data type and object as its variable.
3. Class is a logical construct that is, no memory is allotted to it when it is defined and object has a physical reality (mean allotted).