**ASSIGNMENT NO – 5**

**TITLE :** Designing a java application using Applet.

**DATE:**

**REMARK :**

**ASSIGNMENT – 5**

**TITLE :** Implement an application using Applet

**OBJECTIVE :**

Java **Applets** are usually **used** to add small, interactive components or enhancements to a webpage. These may consist of buttons, scrolling text, or stock tickers, but they can also be **used** to display larger programs like word processors or games.

**PROBLEM STATEMENT :**

Design a system by using Applet in Java and enhance the system using advanced data structures.

**THEORY :**

A Java applet is a small dynamic Java program that can be transferred via the Internet and run by a Java-compatible Web browser. The main difference between Java-based applications and applets is that applets are typically executed in an AppletViewer or Java-compatible Web browser. All applets import the java.awt package.

Advantage of Applet

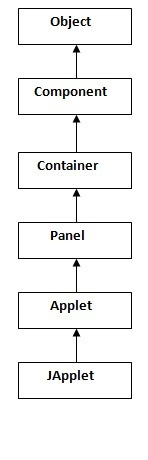
There are many advantages of applet. They are as follows:

* It works at client side so less response time.
* Secured
* It can be executed by browsers running under many plateforms, including Linux, Windows, Mac Os etc.

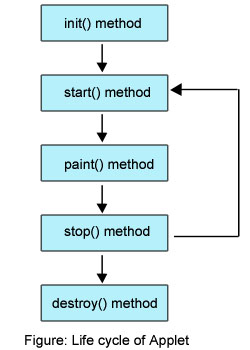
Drawback of Applet

* Security: Java resolves the security issue by restricting applets to Java’s execution environment and preventing access to system resources.
* Portability: Portability is defined as the applet’s ability to run on different computers and operating systems.
* Plugin is required at client browser to execute applet.

**DIAGRAMATIC REPRESENTATION**



As displayed in the above diagram, Applet class extends Panel. Panel class extends Container which is the subclass of Component.



Various states, an applet, undergoes between its object creation and object removal (when the job is over) is known as **Applet Life Cycle**. Each state is represented by a method. There exists 5 states represented by 5 methods. That is, in its life of execution, the applet exists (lives) in one of these 5 states.

These methods are known as "**callback methods**" as they are called automatically by the browser whenever required for the smooth execution of the applet. Programmer just write the methods with some code but never calls.

Following are the methods.

1. init() method
2. start() method
3. paint() method
4. stop() method
5. destroy() method

These methods are known as Applet Life Cycle methods. These methods are defined in **java.applet.Applet**class except **paint()** method. The paint() method is defined in **java.awt.Component** class, an indirect super class of Applet.

Following is the brief description of the above methods.

1. **init():** The applet's voyage starts here. In this method, the applet object is created by the browser. Because this method is called before all the other methods, programmer can utilize this method to instantiate objects, initialize variables, setting background and foreground colors in GUI etc.; the place of a constructor in an application. It is equivalent to [**born state**](http://way2java.com/multithreading/life-cycle-of-thread/) of a thread.
2. **start():** In init() method, even through applet object is created, it is in **inactive**state. An inactive applet is not eligible for microprocessor time even though the microprocessor is idle. To make the applet active, the init() method calls start() method. In start() method, applet becomes active and thereby eligible for processor time.
3. **paint():** This method takes a **java.awt.Graphics** object as parameter. This class includes many methods of drawing necessary to draw on the applet window. This is the place where the programmer can write his code of what he expects from applet like animation etc. This is equivalent to [**runnable state**](http://way2java.com/multithreading/life-cycle-of-thread/) of thread.
4. **stop():** In this method the applet becomes temporarily inactive. An applet can come any number of times into this method in its life cycle and can go back to the active state (paint() method) whenever would like. It is the best place to have cleanup code. It is equivalent to the [**blocked state**](http://way2java.com/multithreading/life-cycle-of-thread/) of the thread.
5. **destroy():** This method is called just before an applet object is garbage collected. This is the end of the life cycle of applet. It is the best place to have cleanup code. It is equivalent to the [**dead state**](http://way2java.com/multithreading/life-cycle-of-thread/) of the thread.

HOW TO RUN AN APPLET

There are two ways to run an applet

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

**SYNTAX / CODE :**

package cgg;

import java.applet.Applet;

import java.awt.Color;

import java.awt.Dimension;

import java.awt.Graphics;

import javax.swing.JFrame;

public class Pendulum extends Applet{

public Pendulum() {

// TODO Auto-generated constructor stub

}

public void init(){

}

public void paint(Graphics g){

int x1,y1,x2,y2;

int k1,j1,k2,j2;

int px,py;

double theta=3\*Math.PI/2;

x1=450;

y1=100;

x2=450;

y2=300;

px=425;

py=275;

this.setSize(new Dimension(1000,600));

//g.setColor(Color.red);

//g.drawLine(450,100, 450, 300);

//g.drawOval(425, 275, 50, 50);

for(int i=0;i<1024;i++){

Dimension d = getSize();

g.setColor(Color.WHITE);

g.fillRect(0, 0, d.width, d.height);

g.setColor(Color.red);

if(i<256||i>768)

{

k2= (int) (x2\*Math.cos(theta));

j2= (int) (y2\*Math.sin(theta));

g.drawLine(450,100,450-k2,200-j2);

g.setColor(Color.black);

g.fillOval(425-k2,175-j2,50,50);

g.setColor(Color.red);

theta=theta+0.001;

}

else if(i<768){

k2= (int) (x2\*Math.cos(theta));

j2= (int) (y2\*Math.sin(theta));

g.drawLine(450,100,450-k2,200-j2);

g.setColor(Color.black);

g.fillOval(425-k2,175-j2,50,50);

g.setColor(Color.red);

theta=theta-0.001;

}

try {

Thread.sleep(1);

} catch (InterruptedException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

}

public static void main(String[] args, Graphics g){

Pendulum a=new Pendulum();

a.setSize(2000,2000);

a.setVisible(true);

a.paint(g);

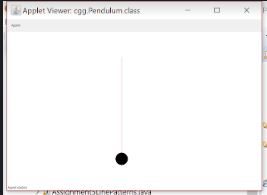
// a.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

}

**OUTPUT:**

Implementation of an Applet in Java



**PLATFORM:**

Net Beans and JDK8.0.

**Conclusion:**

In this Experiment we Learned that how Applets can be simply implemented and make your system more attractive for the user.