CHAPTER 12

APPLET PROGRAMMING

This chapter introduces the classes of the java.applet package and explains how applets are integrated within Web documents. When you use a Java technology-enabled browser to view a page that contains an applet, the applet's code is transferred to your system and executed by the browser's Java Virtual Machine (JVM). An applet is a Java program that can operate only within a compatible Web browser, such as Netscape Navigator or Microsoft Internet Explorer. Java applets can run in a Web browser using a Java Virtual Machine (JVM), using a java interpreter or in Sun's AppletViewer included in the java development kit, a stand-alone tool for testing applets. In order to run an applet it must be included in a web page, using HTML tags. When a user browses a web page containing an applet, the browser downloads the applet from the web server and runs it on the user's system.

12.1. INTRODUCTION

Java applets were introduced in the first version of the Java language in 1995. The applet programs are also written in Java. The first browser that could show applets was introduced in 1994, as "WebRunner" - later known as "The HotJava Browser". Java applications are stand-alone Java programs that can be run by using just the Java interpreter, for example, from a command line. Applets are not executed by the console-based Java run-time interpreter. Rather, they are executed by either a Web browser or an applet viewer. Java applets, however, are run from inside a World Wide Web browser. A reference to an applet is embedded in a Web page using a special HTML tag. All applets are subclasses of Applet. Thus, all applets must import java.applet. Applets do not have main() method. All applets must be declared public.

12.2. TYPES OF AN APPLET

An applet is a special kind of Java program which is used for internet programming. Basically applet has two types:

(i) Local Applet

(ii) Remote Applet

12.2.1. Local Applets

A Local applet is the one which is stored on our local computer system. When browser try to access the applet, it is not necessary for our computer to be connected to The Internet. When our Web page finds a local applet, it doesn't need to retrieve information from the Internet; in fact, your web browser doesn't even need to be connected to the Internet at that time.

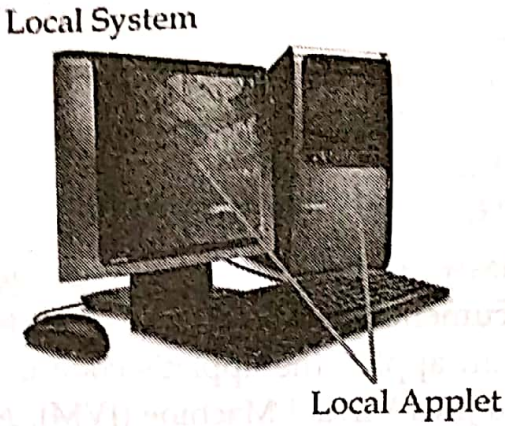


Fig 12.1: Local Applet

As you can see in the following example, a local applet is specified by a path name and a file name.

Syntax to specifying a Local Applet

```
<APPLET   CODEBASE="localApplet "
          CODE="localApplet.class"
          WIDTH=300
          HEIGHT=400>
</APPLET>
```

Where codebase specifies a path name on your system for the local applet and code specifies the name of the byte-code file (.class) that contains the applet's code. The path specified in codebase is relative to the folder containing the HTML document to references the applet.

12.2.2 Remote Applets

A Remote applet is the one which is not stored on our computer system but located on another computer system. We are required to be connected to the Internet. This computer system may be located in the internet. No matter, where the remote applet is located, it's downloaded onto our computer via the Internet. Our browser must be connected to the Internet at the time it needs to display the remote applet. To display the remote applet in your Web browser, you must know the URL of Applet (location of applet on Web) and the parameters and attributes are needed to supply in order to display the applet correctly. If you are not the author of applet then you will need to find the document that describes the applet's parameters and attributes. This document is usually written by the author of applet. The following example shows how to compose an HTML <applet> tag for accessing the remote applet.

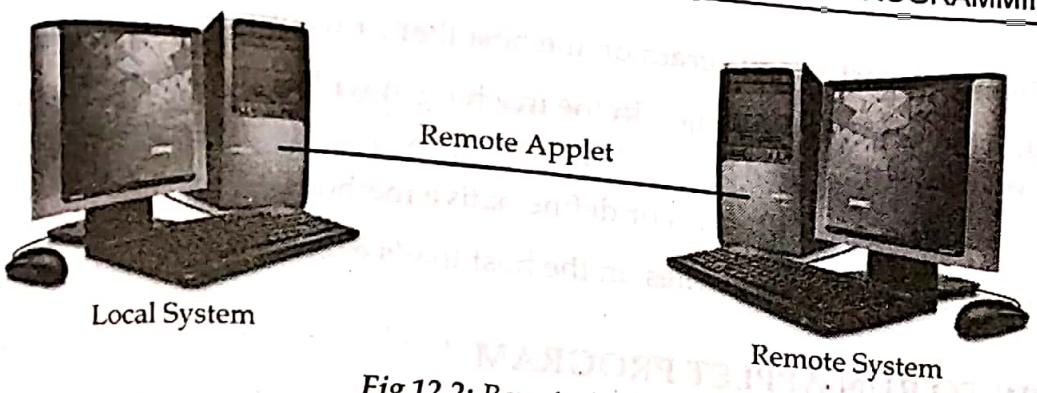


Fig 12.2: Remote Applet

Syntax to specifying a Remote Applet.

```
<APPLET  CODEBASE=" http://www.myapplet.com/remoteapplets/"
         CODE="remoteApplet.class"
         WIDTH=300
         HEIGHT=400>
</APPLET>
```

The only difference between local applet and remote applet is the value of the codebase attribute. In the local applet, codebase specifies a local folder and in the remote applet, it specifies the URL, where the applet is located.

12.3. ADVANTAGES OF JAVA APPLET

(i) Applets run in a sandbox, so that user does not need to trust the code, so it can work without security approval.

(ii) Applets are platform independent and can run on Windows, Macintosh OS and Linux platform.

(iii) Applets are supported by most web browsers.

(iv) Applets running within a Web browser can easily cause HTML documents to be displayed

(v) Applets can invoke public methods of other applets on the same page.

(vi) Applets can make network connections to the host they came from

(vii) Applets can work on all the versions of Java Plug-Ins.

(viii) Applets are cached in most web browsers, so will be quick to load when returning to a web page

12.4 DISADVANTAGES OF JAVA APPLET

(i) Java plug-in is required to run any applet.

(ii) Applets can't make network connections except to the host that it came from.

(iii) Designing and build good user interfaces are difficult in applets compared to HTML.

(iv) Applet can't start any program on the host that's executing it.

(v) If applet is not already cached in the machine, it will be downloaded from internet and will take time

(vi) An applet can't load libraries or define native methods.

(viii) Applet can't read or write files on the host that's executing it.

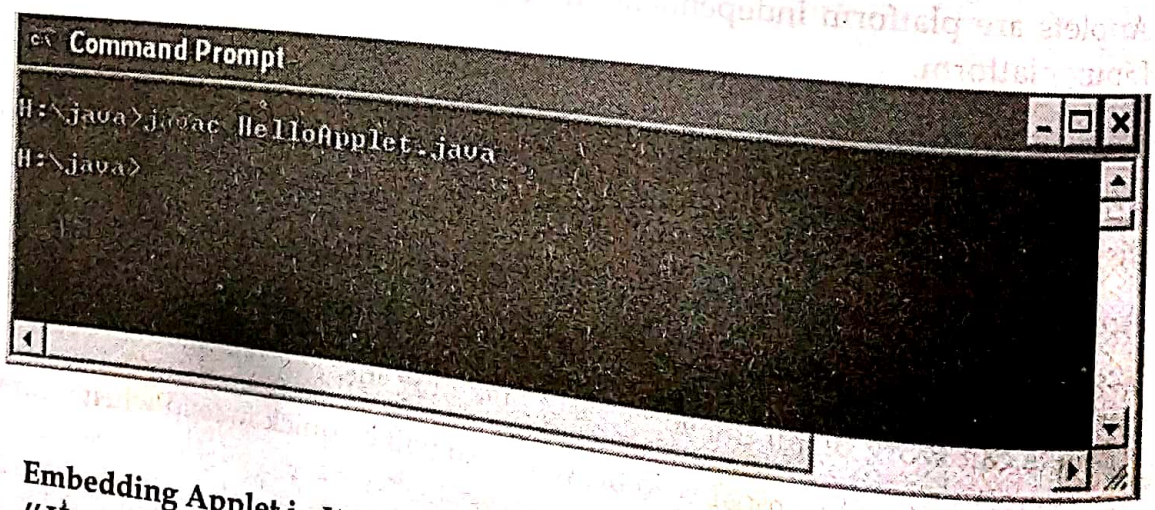
12.5 HOW TO RUN APPLET PROGRAM

To run the applet program, we need to load the HTML file into an application that is used to run Java applet program. This application might be a java compatible web browser or appletviewer which is provided in the JDK.

Building Applet Code: An Example

```
//HelloApplet.java

import java.awt.* ;
import java.applet.*;
public class HelloApplet extends Applet
{
public void paint(Graphics g)
{
g.drawString ("My First Applet",50, 50);
}
}
```



Embedding Applet in Web Page

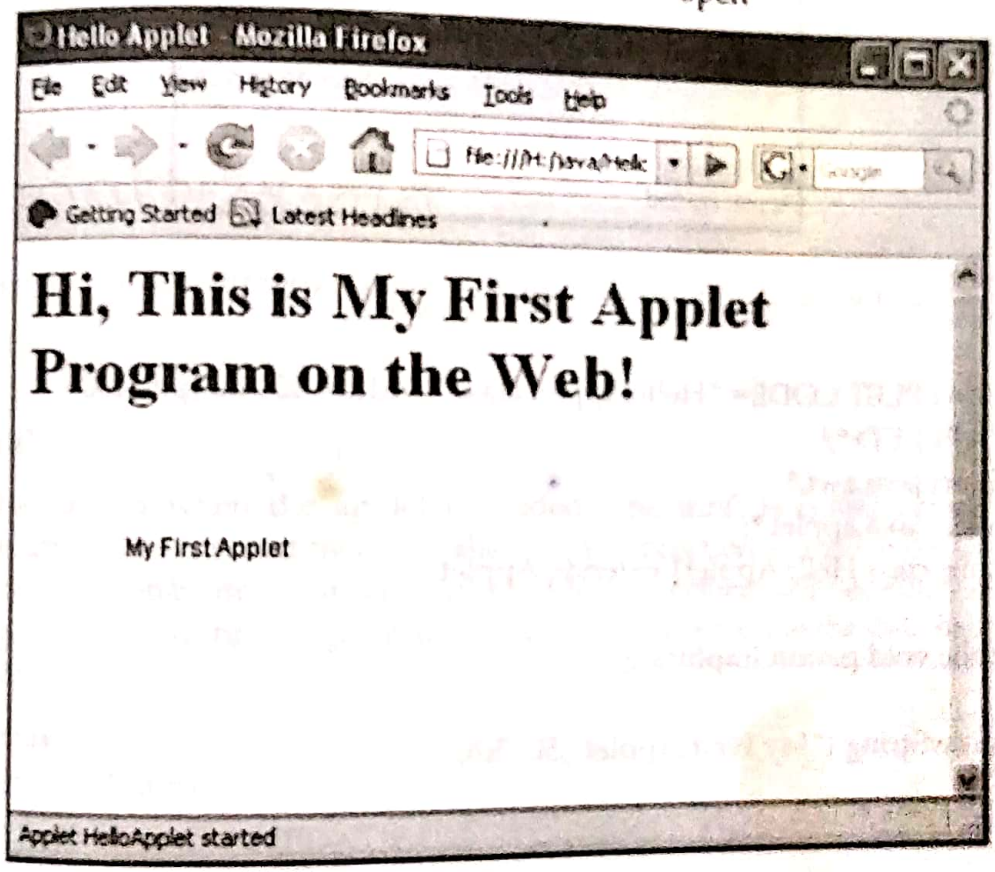
```
// HelloApplet.html
<HTML>
<HEAD>
<TITLE>
Hello Applet
</TITLE>
```

```
</HEAD>
<BODY>
<H1>Hi, This is My First Applet Program on the Web! </H1>
<APPLET CODE= "HelloApplet.class" width= 400 height=400>
</APPLET>
</BODY>
</HTML>
```

Accessing Web page (runs Applet)

Start internet explorer.

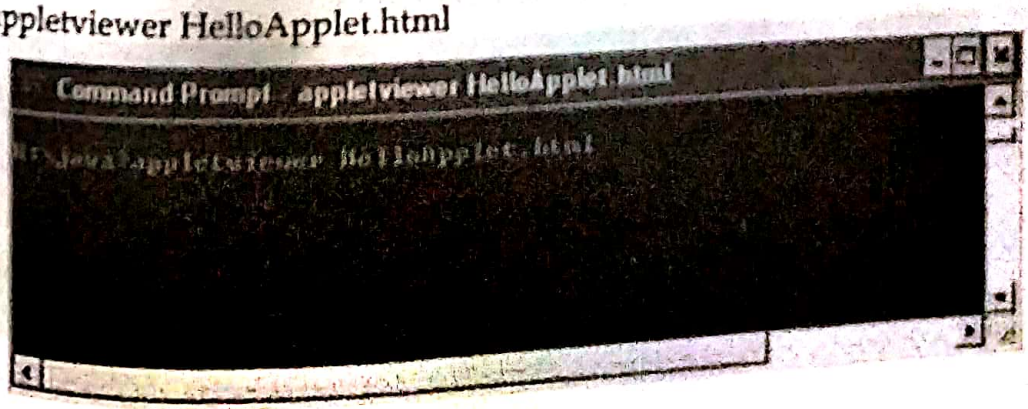
Click file -> open -> Browse... -> select the file -> open

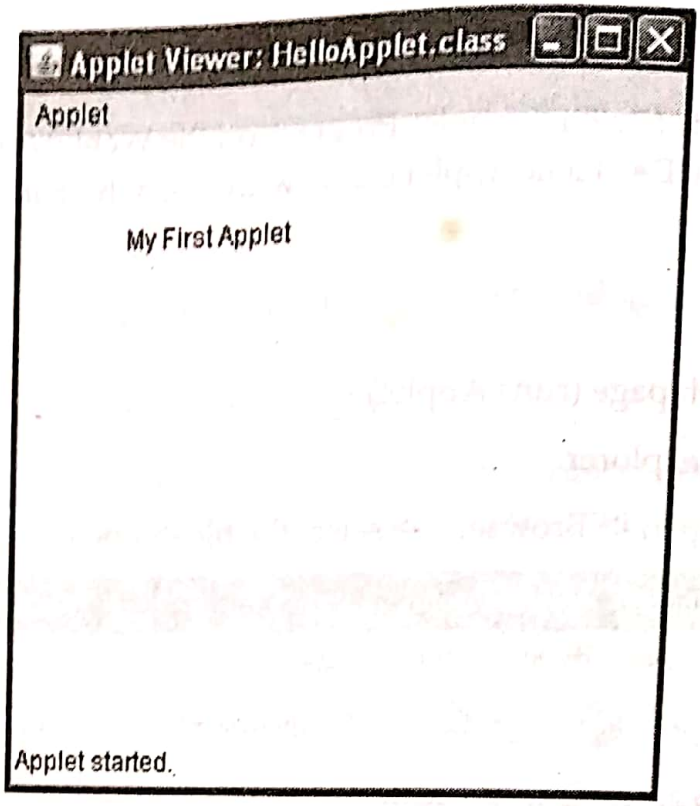


Embedding Applet in AppletViewer

Write the following command to the command prompt

>appletviewer HelloApplet.html



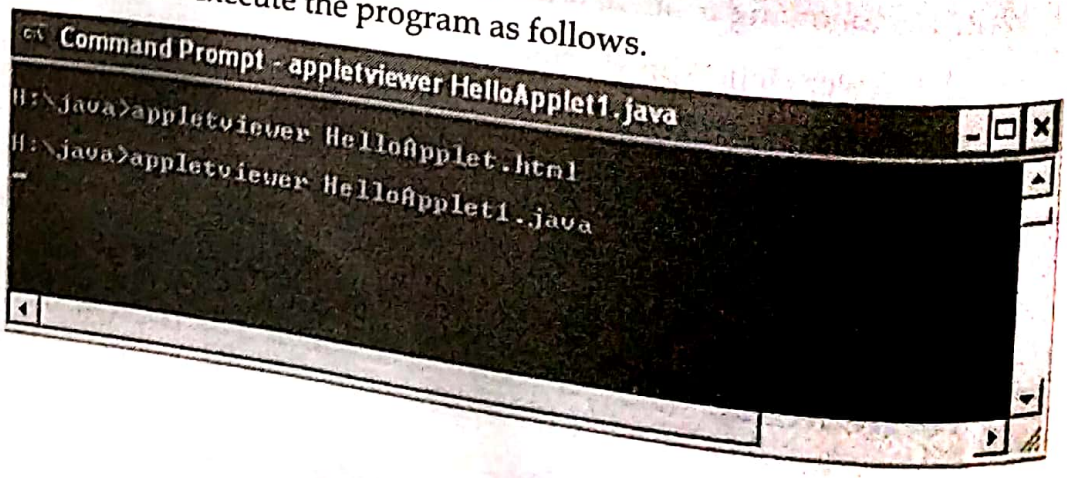


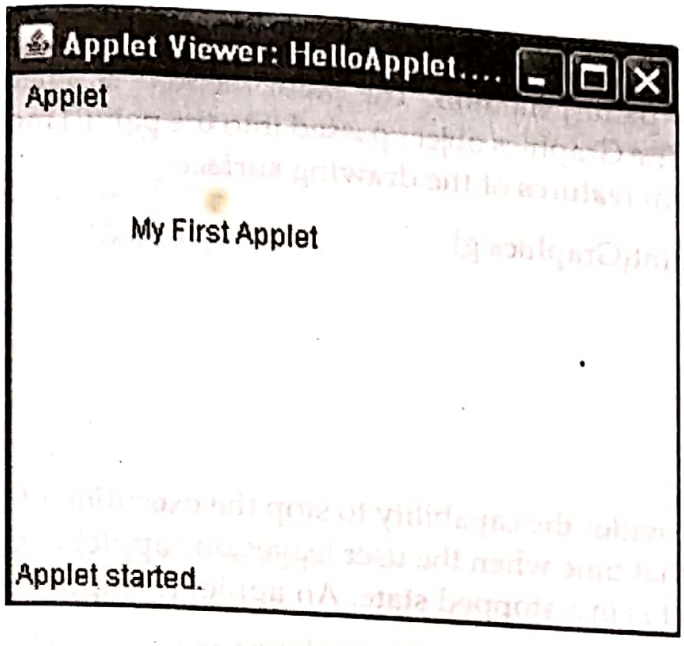
Or write the following code with the main program using comment line as follows.

```
/* <APPLET CODE= "HelloApplet.class" width= 400 height=400>
</APPLET>*/
import java.awt.* ;
import java.applet.*;
public class HelloApplet1 extends Applet
{
public void paint(Graphics g)
{
g.drawString ("My First Applet",50, 50);
}
}
```

Save the above program with HelloApplet1.java

Now compile and execute the program as follows.





12.6 LIFECYCLE OF AN APPLET

Applets are automatically constructed by the runtime environment associated with a Web browser or the applet viewer. Applets have four lifecycle methods: init(), start(), paint(), stop(), and destroy().

12.6.1 init()

Initialization occurs when the applet is loaded. The init() is called before the applet begins execution, it is the first method called by an applet once it has been loaded by the AppletViewer or web browser., it can be overridden to perform any necessary initialization processing, such as loading images, establishing the connection to the data base, set layout or set Fonts.

Syntax:

```
public void init()
{
...
}
```

12.6.2 start()

After initialization an applet is started by the start() method. start( ) is called automatically to begin the actual execution of the applet. The start() method serves as the execution entry point for an applet when it is initially executed and restarted as the result of a user returning to the Web page that contains the applet.

Syntax:

```
public void start()
{
...
}
```

12.6.3 paint()

The applet overrides paint() method. The paint method is where the real work of this applet really occurs. The Graphics object passed into the paint() method holds the graphics state that is, the current features of the drawing surface.

```
public void paint(Graphics g)
{
 ...
}
```

12.6.4 stop()

The stop() method provides the capability to stop the execution of an applet. It is called by the web browser on that time when the user leaves one applet to go another applet. start() can occur if the applet is in a stopped state. An applet is stopped when the user goes to a different web page and comes back to the applet page.

Syntax:

```
public void stop()
{
 ...
}
```

12.6.5 destroy()

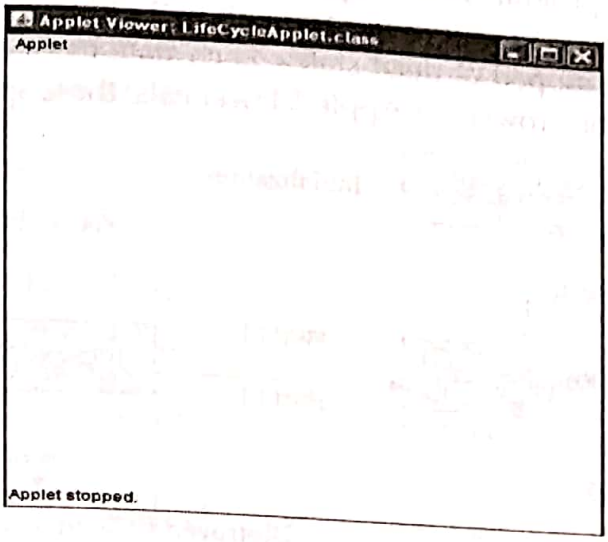
Like init() method, the destroy() method is called only one time in the life cycle of Applet. destroy( ) is called when an applet has completely finished the execution process, or when the browser needs to Shut down. This method is used at the end of an applet's life cycle to perform the process of termination of applet from web browser.

Syntax:

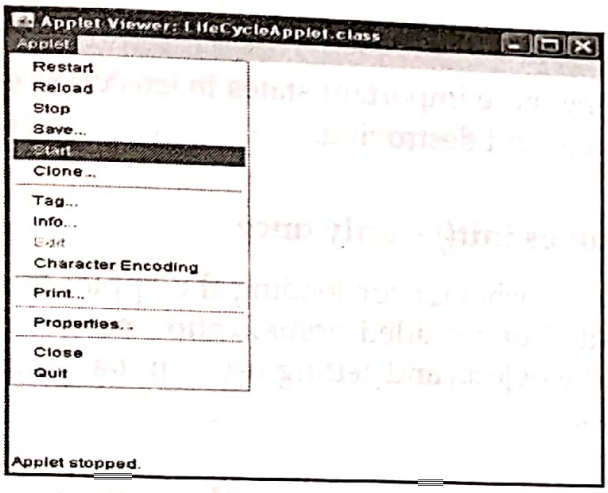
```
public void destroy()
{
 ...
}
```

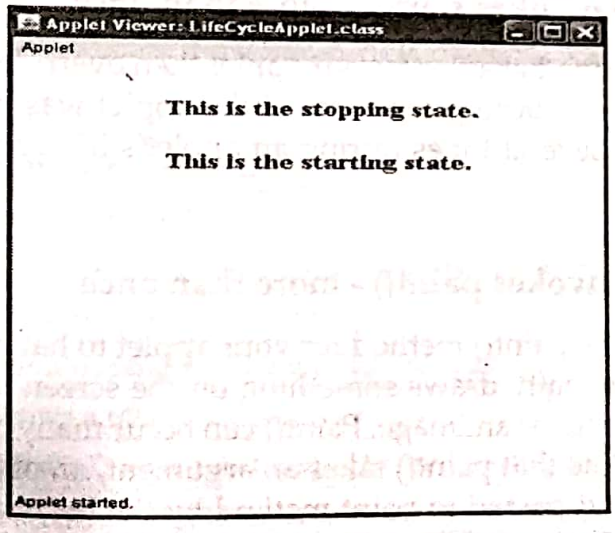
Example: *LifeCycleApplet*

```
import java.awt.*;
import java.applet.*;
import java.applet.*;
public class LifeCycleApplet extends Applet
{
  Font f = new Font("times new roman", Font.BOLD, 20);
  int i;
  String S1, S2;
```



Now again click on start.





12.7 STATES TRANSITION DIAGRAM OF AN APPLET

If we have to create a basic Java application then our class must have one method, main(). When our application starts up then main() method is executed and from main() we can set

up the behavior that our program needs. Applets are similar to Java application. Applets have many states that correspond to various major events in the life cycle of an applet, for example, initialization, painting, stopped or dead states. Each state has a corresponding method, so when an event occurs, the browser or AppletViewer calls those specific methods.

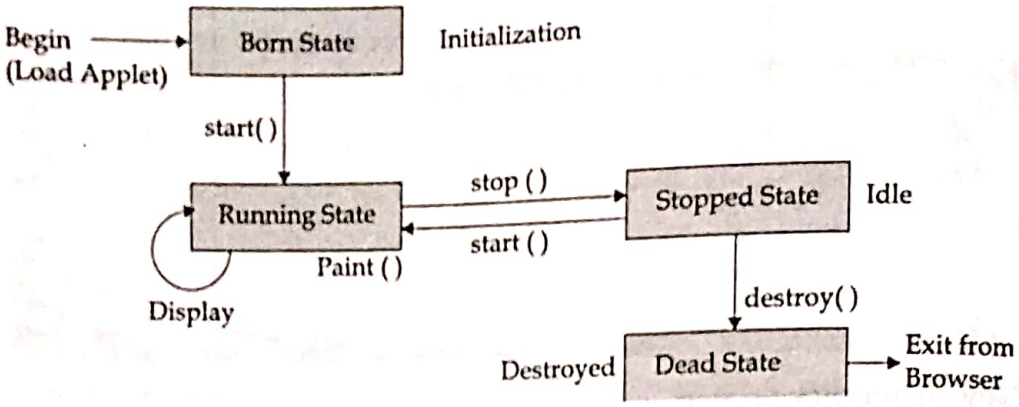


Fig 12.3: Life cycle of Applet

Here are five of the more important states in execution of an applet: initialization, starting, painting, stopping and destroying.

12.7.1 Born State -invokes init() – only once

When we want to provide behavior for loading the applet then override init() method. When the applet is loaded or reloaded, initialization occurs. In this state Initialization might include creating the objects and setting up an initial state, loading images or fonts or setting the parameters.

12.7.2 Running State invokes start() – more than once

We have to provide startup behavior for our applet then override the start() method. After initialization, it is started. Starting can occur if the applet was previously stopped. Note that starting can occur several times during an applet's life cycle, whereas initialization happens only once.

12.7.3 Display State invokes paint() - more than once

Display state override the paint() method for your applet to have an actual appearance on the screen. An applet actually draws something on the screen using paint(), be it text, a line, a colored background, or an image. Paint() can occur many hundreds of times during an applet's life cycle. Note that paint() takes an argument, an object of the class Graphics. This object is created and passed to paint method by the browser. The java.awt package has the class Graphics. So import this package in your Applet program.

12.7.4 Stopped State invokes stop() - more than once

Stopping state occurs when we leaves the page that contains a currently running applet, or we can stop the applet by calling stop() method. By overriding stop(), we can suspend execution of applet and then restart them if the applet is viewed again.

12.7.5 Dead State invokes destroy() - only once

Dead state occurs when the web browser exits. To provide clean up behavior for your applet, override the destroy() method. Destroying enables the applet to clean up before it is freed.

12.8 APPLET'S METHODS

Applet provides all of the necessary support and functionality for window-based programming. Applet provides more methods for applet execution, such as starting and stopping, load and display images, load and play audio clips. Applet extends the Applet class which exists in applet package. These classes provide support for Java's window-based, graphical user interface.

Table 12.1 : Applet's Methods

Method	Result
Graphics( )	To constructs a new Graphics object.
create( )	To creates a new Graphics object.
dispose( )	To disposes of the current graphics context.
getColor( )	Gets the current color value.
setColor( )	Sets the current color value.
getFont( )	Gets the current font name.
setFont( )	Sets the font for all text operations.
drawRoundRect( )	Draws a rounded corner rectangle using the current color.
draw3DRect( )	Draws a 3-D rectangle.
drawOval( )	Draws an oval using the current color.
drawPolygon( )	Draws a polygon using an array of x points and y points.
drawString( )	Draws a string using the current font and color.
drawLine( )	Draws a line between two points(x1,y1) and (x2,y2).
drawRect( )	To draws a rectangle using the current color.
clipRect( )	Clips to a rectangle.
clearRect( )	Clears the specified rectangle using the current background color.
fillPolygon( )	To fills a polygon with the current color.
fillRect( )	To fills a rectangle using the current color.
fillRoundRect( )	To draws a filled rounded rectangle.
fill3DRect( )	Paints a 3-D rectangle filled with the current color.
fillOval( )	Fills an oval using the current color.

12.9 The <APPLET> Tag

Use the <APPLET> tag, to include an applet on a Web page. <APPLET> is a special extension to HTML for including applets in Web pages. In order to view the applet, an HTML code has to be written. It has three attributes.

(i) **CODE** – Specifies the name of the applet class including the .class extension. For example code="applet1.class". In this case, the class file must be in the same directory as this HTML file.

(ii) **WIDTH** – Specifies the width of the applet window on the Web Page or the width of appletviewer window.

(iii) **HEIGHT** – Specifies the height of the applet window on the Web Page or the width of appletviewer window.

Other attribute that could be added to the <applet> tag are as follows.

(i) **ALIGN** – This could be set to LEFT,RIGHT,MIDDLE,TOP,BOTTOM

(ii) **HSPACE** – Controls the horizontal space to the left and right of the applet.

(iii) **VSPACE** – Controls the vertical space above and below the applet.

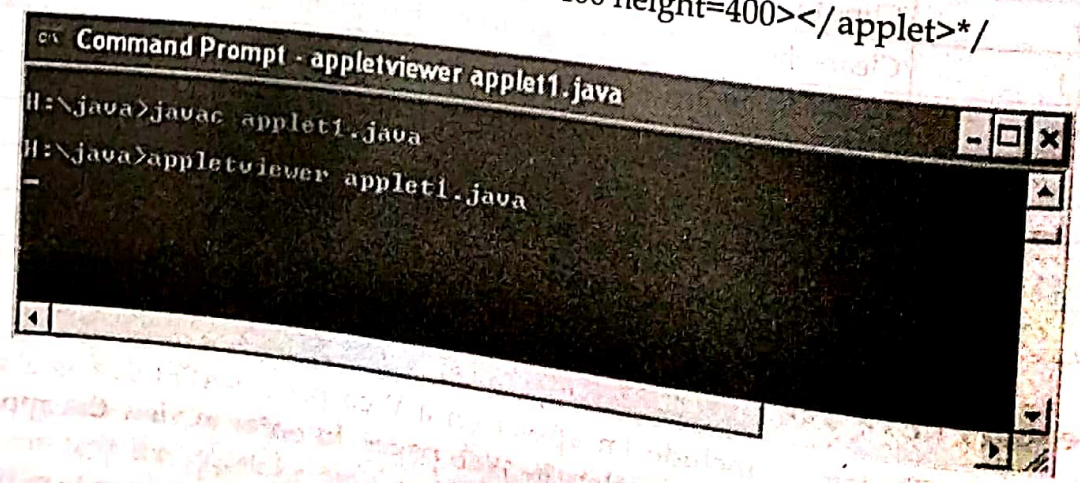
(iv) **CODEBASE** – Indicates an alternate web site for the web browser to specify the location of the applet.

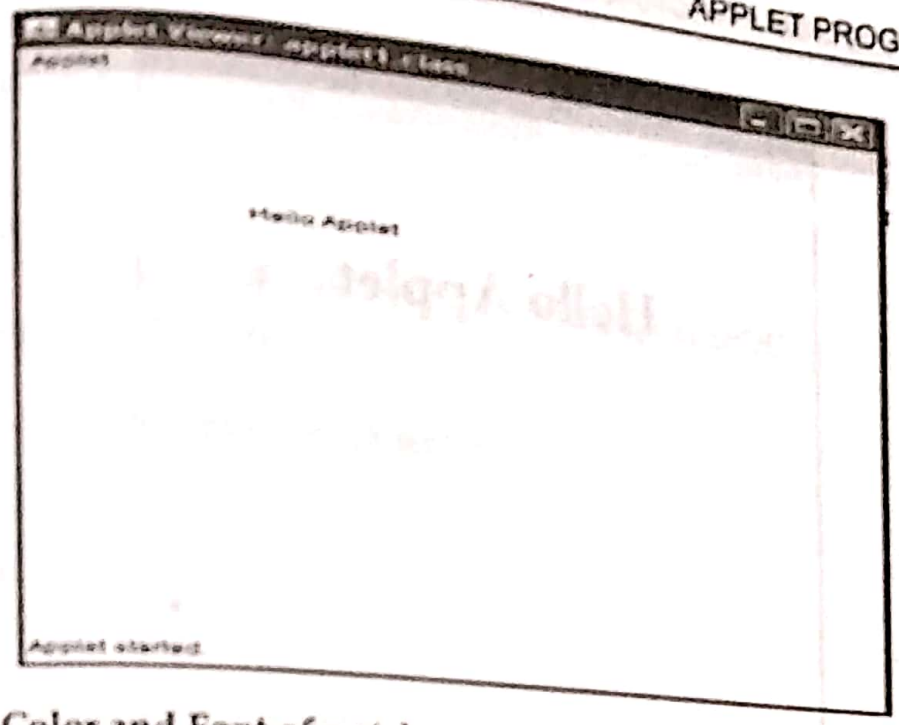
(v) **ALT** – Specifies the text to be display if the web browser understands the <applet> tag, but can't run Java applets.

12.10 SOME EXAMPLES OF APPLET

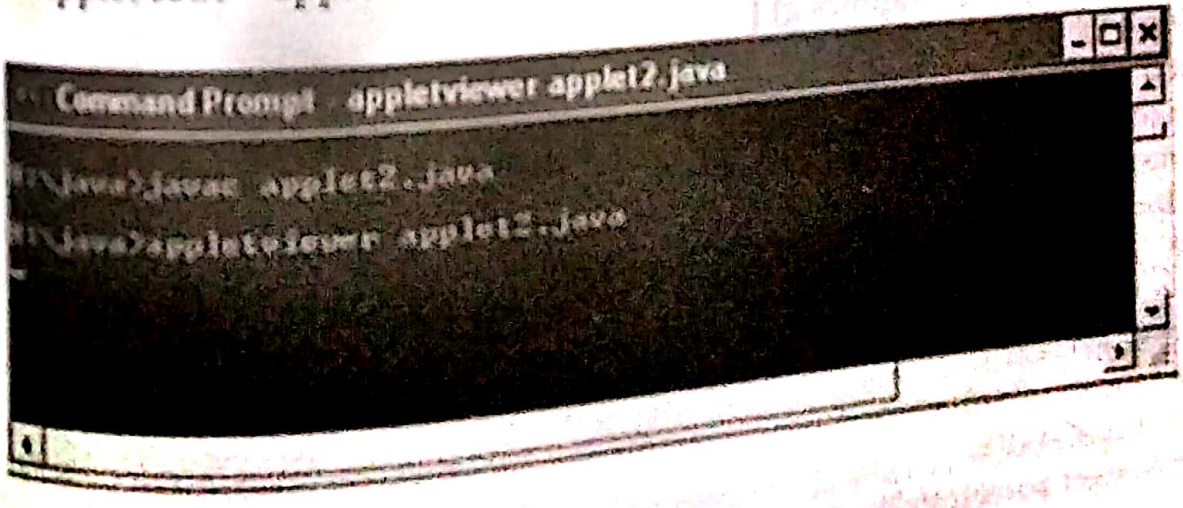
// Draw a string "Hello Applet" to web browser/AppletViewer

```
import java.awt.*;
import java.applet.*;
public class applet1 extends Applet
{
public void paint(Graphics g)
{
g.drawString("Hello Applet", 100, 100);
}
}
/*<applet code="applet1.class" width=400 height=400></applet>*/
```






```
// Set the Color and Font of a string
import java.awt.*;
import java.applet.*;
public class applet1 extends Applet
{
Font f;
Public void init()
{
f = new Font("TimesRoman", Font.BOLD, 36);
}
public void paint(Graphics g)
 {
 g.setFont(f);
 g.setColor(Color.red);
 g.drawString("Hello Applet", 100, 100);
 }
}
/*<applet code="applet1.class" width=400 height=400></applet>*/
```



Syntax
```
Public void stop()
{
}
```

destroy(): It is called only once in the one time of life cycle of applet. It is called when an applet has completely finished the execution process or when the browser needs to shut down.

Java Color System

Syntax:- Color(int green, int red, int blue)

```
Color c = new Color(192, 196, 200);
                or
          (192, 192, 192); // light Gray
```

Color	Red	Green	Blue
Red	255	0	0
Green	0	255	0
Blue	0	0	255
Light Gray	192	192	192
Dark Gray	128	128	128
Black	0	0	0
White	255	255	255
Yellow	255	255	0
Purple	255	0	255

Setting the Current Graphic Color

Set Color(): By default graphics object are drawn current foreground color. You can change this color by calling graphics method set color().

```
void set-Color (Color c)
```

we can obtain the current color by calling get-color method.

ex:-

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

public class SetColor extends Applet
{
  public void Paint (Graphics g)
  {
    Color x = new Color(255, 0, 0);
    Color Y = new Color(0, 255, 0);
    Color z = new Color(0, 0, 255);
    g.setColor(x);
    g.drawLine(10, 30, 100, 300);
    g.setColor(Y);
    g.drawLine(0, 50, 300, 10);
    g.setColor(z);
    g.drawLine(0, 80, 400, 80);
```

```
        g.setColor(Color.red);
        g.drawOval(10,10,100,100);
    }
}
/* <applet code="setColor.class" height=200
                width=200>
    </applet> */
```

Working With Fonts

Fonts are encapsulated by the Font class.
Awt provides flexibility by abstracting fonts.

Syntax:

Font(String Font name, int style, int size) → Create new font

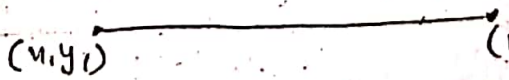
Methods Used:

setFont(Font f)
getFont();
getFontName();
getName();
getSize();
getStyle();
isBold();
isItalic();

ex:
```
import java.awt.*;
import java.applet.*;
public class setfont1 extends Applet
{
  Font f;
  public void init()
  {
    f = new Font("Times New Roman", Font.BOLD, 20);
  }
  public void Paint(Graphics g)
  {
    g.setFont(f);
    g.setColor(Color.BLUE);
    g.drawString("JFTM", 100, 200);
  }
}
/*<applet code="setfont1.class" width=300 height=200>
</applet>*/
```

Drawing & Filling: Filling a shape is just as easy as drawing it.

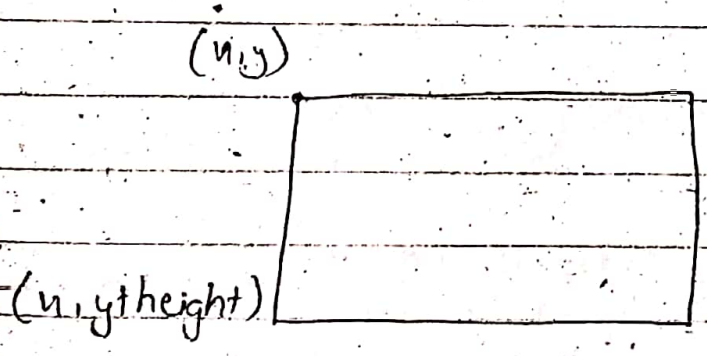
1. Draw line: drawLine()



Ex:- public class Line extends Applet
{
public void paint (Graphics g)
{
g.drawLine(0, 0, 100, 100);
g.drawLine(0, 100, 100, 100);
}
}

/* <applet code = "Line.class" width = 200 height = 200>
</applet> */

2. Drawing Rectangle :- drawRect()



Ex:- public class Rectangle extends Applet
{
public void paint (Graphics g)
{
g.drawRect(50, 50, 100, 100);
g.fillRect(90, 90, 100, 100);
}
}

/* <applet code = "Rectangle" width = 200 height = 200>
</applet> */

3. Drawing Oval & ellipse : drawOval()

drawOval(int top, int left, int width, int height

ex:- public class Oval extends Applet

```
{
  public void paint(Graphics g)
  {
    g.drawOval(100,20,100,30);
    g.setColor(50,50,100,150);
    g.drawOval(55,50,100,100);
  }
}
```

4. Drawing Arcs : drawArc(); , fillArc()

ex:- public class Arc extends Applet

```
{
  public void paint(Graphics g)
  {
    g.setColor(Color.Yellow);
    g.fillArc(10,40,150,150,0,75);
  }
}
```

5. Drawing Polygon :-

```
Eg:- public class Polygon extends Applet
     {
     public void paint(Graphics g)
     {
     SetBackground(Color.RED);
     g.setColor(color.WHITE);
     int x[]={30,200,30,200,30};
     int y[]={30,30,200,200,30};
     int n=5;
     g.drawPolygon(x,y,n);
     }
}
```