❖ Applet:

- Applet is a small Java program which runs on internet.
- An applet is a Java program that runs in a Web browser.
- Applet is a like application program which can perform Arithmetic operations, display graphics, play sounds, accept user inputs, create animation and play interactive games.
- An Applet class does not have any main() method.
- We can create the applet by extending the java.applet.Applet class.
- To run the applet on internet, we need to add it within HTML page.
- There are two types of Applet:
  1. Local Applet
  2. Remote Applet

1. Local Applet:

- Applet which is developed locally and stored in a local machine is known as Local Applet.
- Execution of local applets doesn't require any internet connection.
- The web page will search the local system directories, find the local applet and execute it.
- Specifying a Local Applet:

```
<applet codebase="path" code="NewApplet.class" width=120 height=120 >
</applet>
```

- In above example the codebase attribute specifies a path name on your system for the local applet where that applet's code is present.

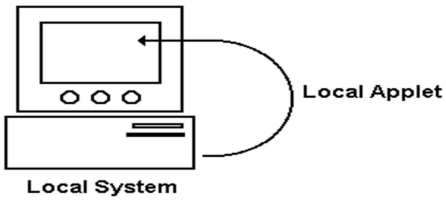


Fig. Loading local Applet from Local computer

2. Remote Applet:

- An applet which is developed by someone else and stored on a remote computer is called as Remote Applet.
- Execution of Remote applets internet connection is required.
- For locating and loading the remote applet, the URL must be specified in the codebase attributes of the HTML code.
- Specifying a Remote Applet:

```
<applet codebase="URL path" code="NewApplet.class" width=120 height=120 >
</applet>
```

- Below diagram shows, how the remote applet is loading and executing from remote system.

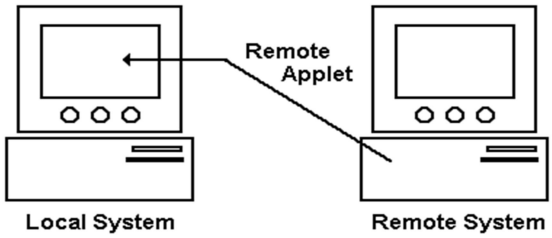


Fig. Loading Remote Applet from Remote computer

Advantages of Applet:

1. Applet works at client side so less response time.
2. Applets are platform independent.
3. Applets increase interactivity for users.
4. Database integration is another important advantage of applets.
5. Quick Execution

Disadvantages of Applet:

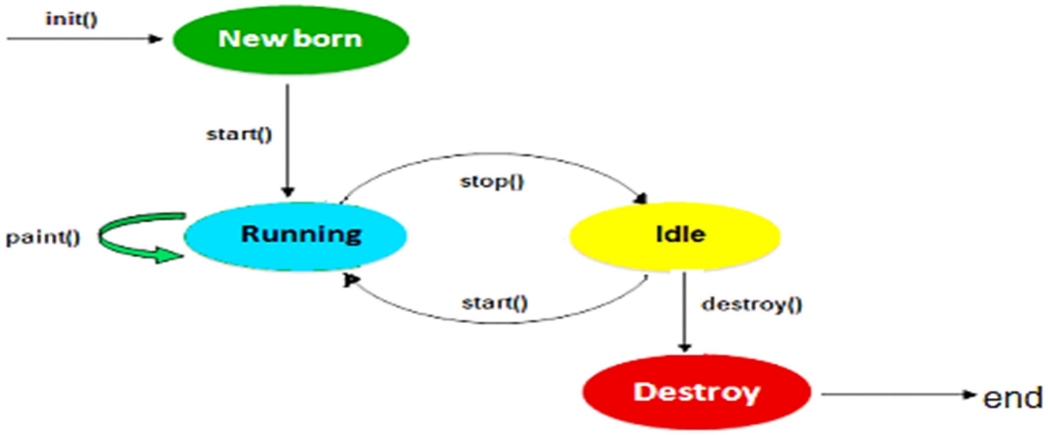
1. Applet cannot run independently.
2. Browser is required to run an applet

❖ Difference between Applet and Application

Applet	Application
1) Applet is a Small Java Program.	Application is a Large Java Program.
2) Applet is not a fully featured application programs.	Application is a fully featured program.
3) Applet program runs on client Browser	Application program can be executed on standalone computer system.
4) Applet programs are portable and It can be executed by any JAVA supported browser.	Need JDK, JRE, JVM installed on client machine.
5) Applet program do not contain main() method	Every application program contain main() method
6) Applet program cannot run independently	Application program can run independently
7) Applet Requires highest security.	Does not require any security.
8) Applet cannot access anything on the system except browser's services.	Can access any data or software available on the system

❖ Applet Life Cycle:

- Following diagram shows the life cycle of Applet.



Applet life cycle consists of 4 states which are given below:

1. New Born/Initialization State
2. Running State
3. Idle/stopped State
4. Dead/Destroy state

New Born State:

- When applet is first loaded then it enters into New Born /Initialization state by calling init() method.
- When Applet is born then it can do creation of objects, setting up initial values, Loading images or fonts, set up colors, etc.
- The Initialization of an applet is occurs only once in the life cycle of an Applet.
- Syntax of init() method:

```
public void init()
{
        //body of init method
}
```

Running State:

- When system calls the start() method of an applet class then it enters into **Running state.**
- This occurs automatically after initialization of an applet.
- Syntax of start() method:

```
public void start()
{
        //body of start method
}
```

- In this state, Applet actually is in running mode. Applet can calls to paint() method to draw any output on the screen. paint() method can takes Graphics class object as arguments.
- Syntax of paint() method:

```
public void paint(Graphics g)
{
        //body of paint method
}
```

Idle/Stopped State:

- When applet is stopped from running then it becomes Idle.
- Stopping of the applet is done automatically when we leaving the web page or we can done explicitly by calling stop() method.
- Idle state applet again come to the running state by calling start() method.
- Syntax of stop() method:

```
public void stop()
{
        //body of destroy method
}
```

Dead/Destroyed State:

- When applet is removed from the memory then it becomes dead.
- This will happen automatically when we exit from the web browser or we can done explicitly by calling destroy() method.
- Destroying of the applet is happen only once in the lifetime of the Applet.
- Syntax of destroy() method:

```
public void destroy()
{
        //body of destroy method
}
```

❖ How to run an Applet:

There are two different ways to run an applet code in Java Programming language.

1. Executing the Applet by using Java-compatible web browser.
2. Using an Applet viewer

1. Executing the Applet by using Java-compatible web browser:

Following steps are required for executing the Applet by using Java-compatible web browser:

Step 1:

- First, we need to create the applet Java code and compile it by using normal **javac** command.
- Example of Applet code:

```
//AppletDemo.java file

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

public class AppletDemo extends Applet
{
          public void paint(Graphics g)
          {
                    g.drawString("Welcome to the world of Applet",150,150);
          }

}
```

Step 2:

- After that we need to create HTML file in the same directory where the applet Java code is present. Inside the body tag of the HTML file, we need to include the applet tag for loading the applet class file.
- Example of HTML file and saved as MyApplet.html:

```
<html>
<body>
<applet code="AppletDemo.class" width="300" height="300">
</applet>
</body>
</html>
```

Step 3: Open the MyApplet.html file by double clicking on it and then you will able to see the output of the applet code.

2. Executing the Applet by using an Applet viewer:

Following steps are required for executing the Applet by using an Applet viewer

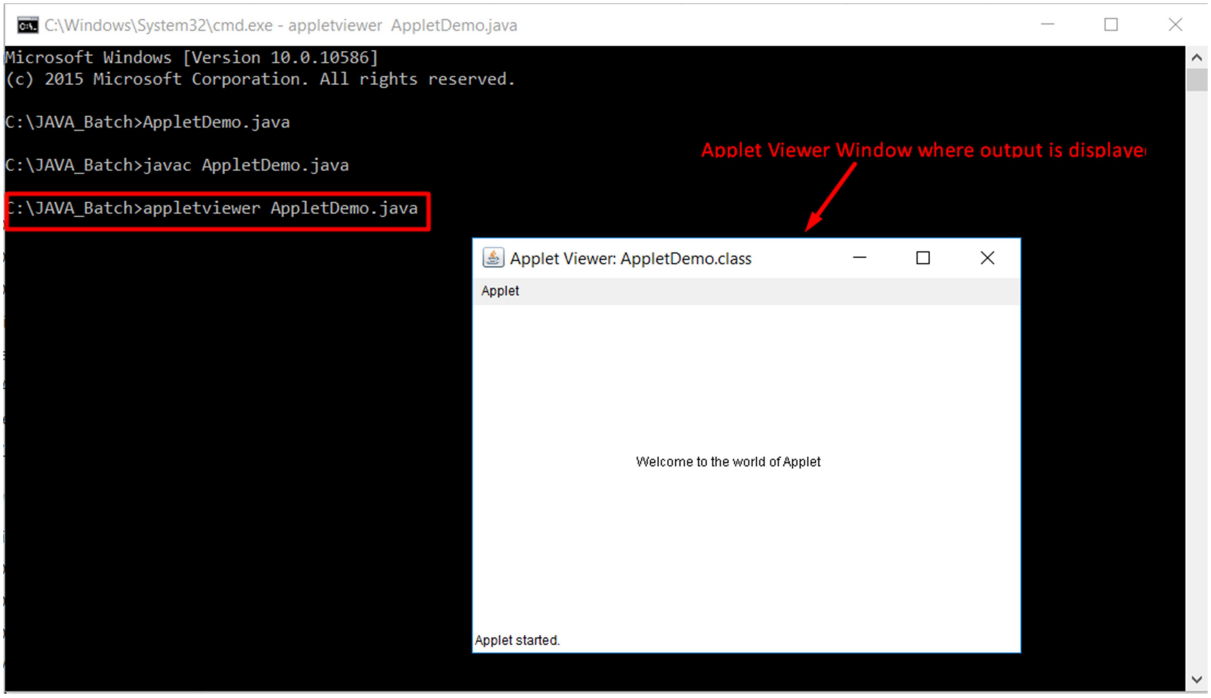
Step 1: First, we need to create an applet that contains applet tag in comment and compile it

Example of Applet code:

```
//AppletDemo.java file

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

public class AppletDemo extends Applet
{
            public void paint(Graphics g)
            {
                        g.drawString("Welcome to the world of Applet",150,150);
            }

}
/*
<applet code="AppletDemo.class" width="300" height="300">
</applet>
*/
```

Step 2: After that run it by using appletviewer command: **appletviewer First.java**

```
c:\>javac AppletDemo.java
c:\>appletviewer First.java
```



❖ **Applet Tag:**

- The pair of <APPLET> and </APPLET> tag is included into the body section of the HTML code.
- The <APPLET> tag is used to mention the name of the applet to be loaded and it tells the browser how much space is required to applet.
- **Syntax:**

```
< APPLET
    [CODEBASE = codebaseURL]
    CODE = Applet_File_Name
    [ALT = alternate_Text]
    [NAME = appletInstance_Name]
    WIDTH = pixels
    HEIGHT = pixels
    [ALIGN = alignment]
    [VSPACE = pixels]
    [HSPACE = pixels]
>
. . .

[< PARAM NAME = appletParameter1 VALUE = value1 >]
[< PARAM NAME = appletParameter2 VALUE = value2 >]
. . .

</APPLET>
```

Where:

CODEBASE = codebaseURL

This optional attribute specifies the base URL of the applet -- the directory or folder that contains the applet's code. If this attribute is not specified, then the document's URL is used.

CODE = Applet_File_Name

This required attribute gives the name of the file that contains the applet's compiled Applet code.

ALT = alternate_Text

This optional attribute specifies any text that should be displayed if the browser understands the APPLET tag but can't run Java applets.

NAME = appletInstance_Name

This optional attribute specifies a name for the applet instance, which makes it possible for applets on the same page to find (and communicate with) each other.

WIDTH = pixels
HEIGHT = pixels

These required attributes give the initial width and height (in pixels) of the applet display area.

ALIGN = alignment

This required attribute specifies the alignment of the applet. The possible values of this attribute are the same (and have the same effects) as those for the IMG tag: left, right, top, middle, bottom.

VSPACE = pixels
HSPACE = pixels

These optional attributes specify the number of pixels above and below the applet (VSPACE) and on each side of the applet (HSPACE).

< PARAM NAME = appletParameter1 VALUE = value >

PARAM tag is used to retrieve the user inputs to the applet code by using getParameter() method.

Example:

Following applet tag specifies the minimum requirements to put applet code on a web page.

```
< APPLET
    CODE = HelloJava.class
    WIDTH = 300
    HEIGHT = 300 >

</APPLET>
```

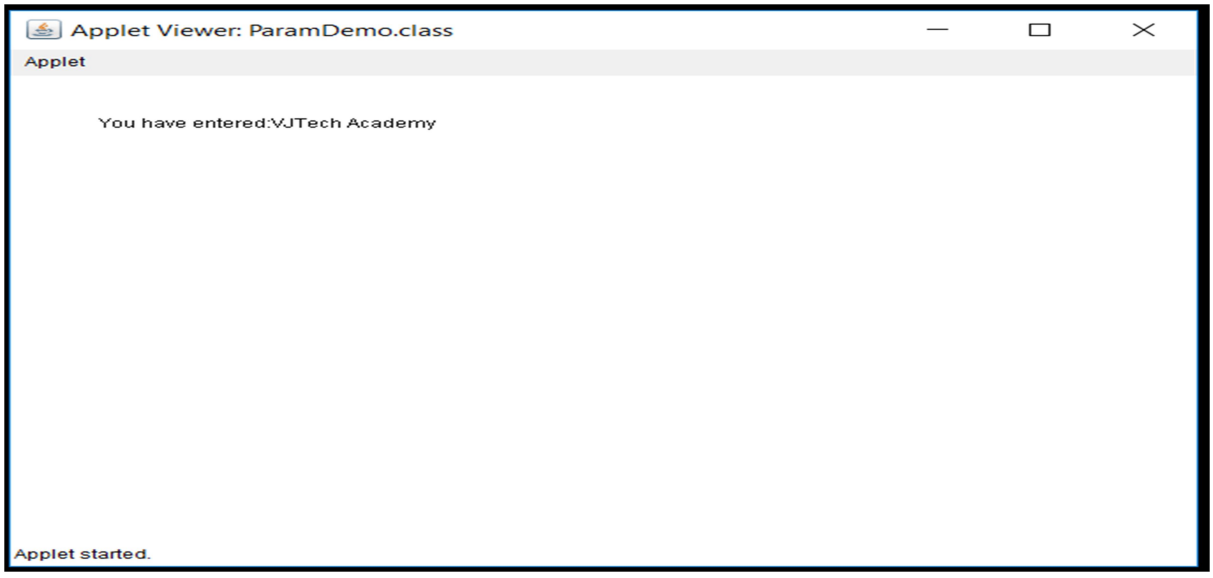

❖ **Parameter Passing to Applet or PARAM Tag:**

- The HTML <param> tag is used for passing parameters to an applet code.
- By using <param> tag we can pass the input values to the Java applet code.
- <param> tag contains two main attributes name and value.
- Applet code can retrieve the value which is associated with the name by using getParameter() method.
- Parameters are passed to the applet code when its loaded and inside the init() method we can retrieve the values of param tag.
- <param> tag needs to be included inside the html code with the <applet> tag.

Example:

```
import java.applet.*;
import java.awt.*;
public class ParamDemo extends Applet
{
  String msg;
  public void init()
  {
          msg=getParameter("String");
  }
  public void paint(Graphics g)
  {
          g.drawString("You have entered:"+msg,50, 50);
  }
}
/*
<applet code="ParamDemo.class" width="300" height="300">
<param name="String" value="VJTech Academy">
</applet>
*/
```

Output:



❖ **drawString() method of Graphics Class in Java:**

- drawstring() method is one of the most useful method of Graphics class to display the text messages on the window.
- **Syntax:**

```
drawString(String msg, int X ,int Y);
```

Where:

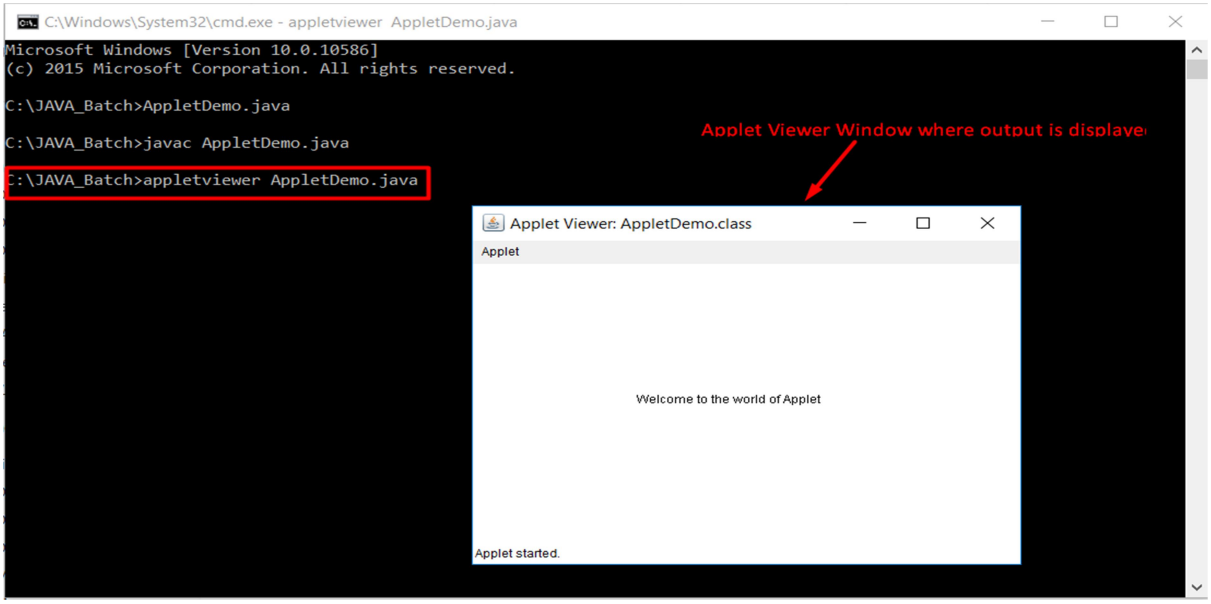
msg = msg is the string that can be displayed on the screen.

X, Y = X and y axis parameter value of the screen.

Example:

```
//AppletDemo.java file

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

public class AppletDemo extends Applet
{
                  public void paint(Graphics g)
                  {
                           g.drawString("Welcome to the world of Applet",150,150);
                  }

}
/*
<applet code="AppletDemo.class" width="300" height="300">
</applet>
*/
```

Output:



❖ drawLine() method of Graphics Class in Java:

- This method is used to draw the line on the output window.
- The drawLine() method takes four parameters of starting point x and y coordinates and ending point x and y coordinates.
- **Syntax:**

```
drawLine(int X1, int Y1, int X2, int Y2);
```

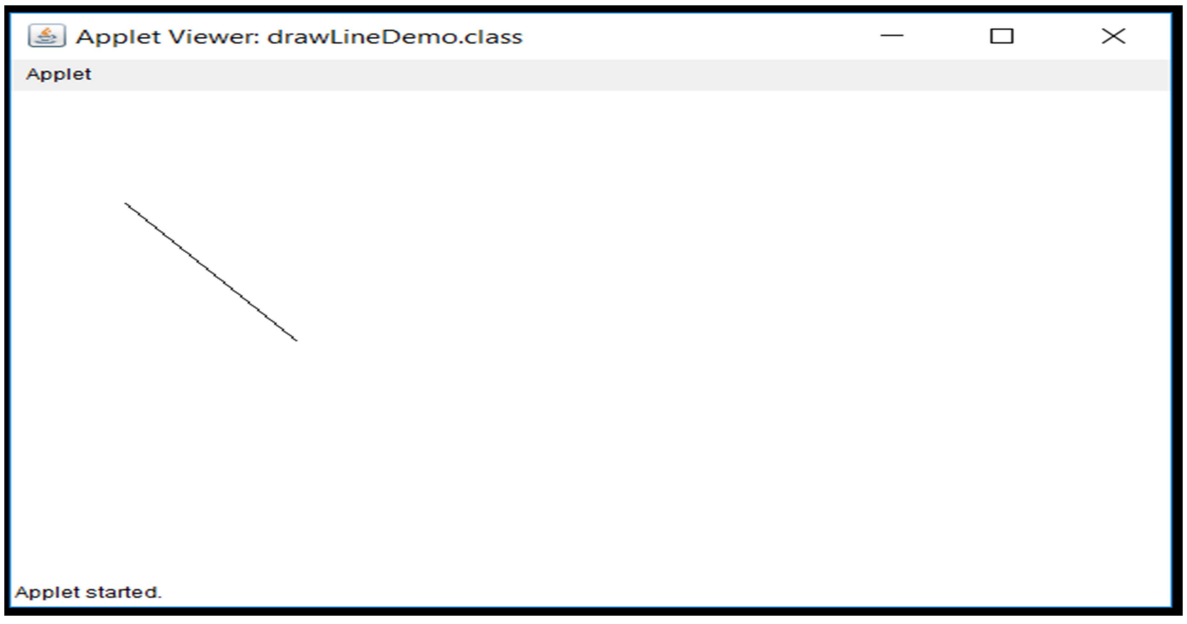
Where:

X1 and Y1 are starting point coordinates and X2 and Y2 are ending point coordinates of the line.

Example:

```
//drawLineDemo.java file

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

public class drawLineDemo extends Applet
{
            public void paint(Graphics g)
            {
                        g.drawLine(60,90,150,200);
            }

}
/*
<applet code="drawLineDemo.class" width="300" height="300">
</applet>
*/
```

Output:



This method takes four parameters, the starting coordinates (60, 90) and ending coordinates (150, 200) of the line to be drawn.

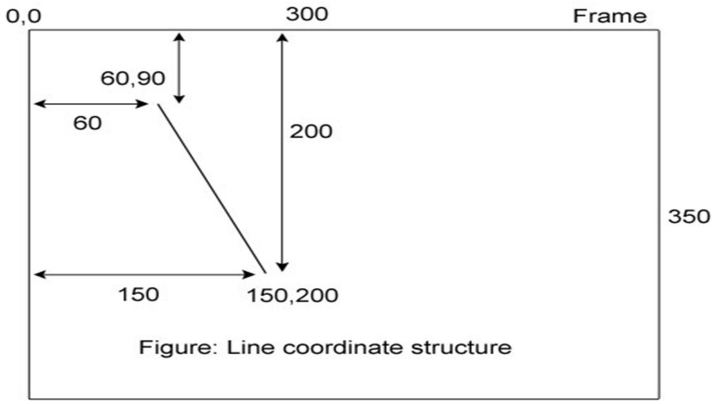


Figure: Line coordinate structure

❖ drawOval() method of Graphics Class in Java:

- This method is used to draw the circle and ellipse on the output window.
- The drawOval() method takes four parameters of starting point x and y coordinates and width and height of the circle or ellipse.
- If width and height parameters values are same then drawOval() method will draw the circle otherwise it will draw the ellipse.
- fillOval() method also draw the circle and ellipse but it's interior area is filled up.
- **Syntax:**

```
drawOval(int X, int Y, int Width, int Height);
fillOval(int X, int Y, int Width, int Height);=
```

Example:

```
//drawOvalDemo.java file

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

public class drawOvalDemo extends Applet
{
                    public void paint(Graphics g)
                    {
                              g.drawOval(50,80,150,100);
                              g.fillOval(230,80,150,100);

                    }

}
```

```
/*
<applet code="drawOvalDemo.class" width="300" height="300">
</applet>
*/
```

Output: