Applets

Applets- Applet class, Applet structure, an example Applet program, Applet life cycle, paint() and repaint().

Java Applet

- ✓ Applet is a special type of program that is embedded in the webpage to generate the dynamic content. It runs inside the browser and works at client side.

- ✓ An Applet does not have a main method. An Applet is a program that is loaded and executed by another program – usually a browser or appletviewer.

- ✓ An applet is a special kind of Java program that runs in a Java enabled browser. This is the first Java program that can run over the network using the browser. Applet is typically embedded inside a web page and runs in the browser.

- ✓ In other words, we can say that Applets are small Java applications that can be accessed on an Internet server, transported over Internet, and can be automatically installed and run as apart of a web document.

- ✓ After a user receives an applet, the applet can produce a graphical user interface. It has limited access to resources so that it can run complex computations without introducing the risk of viruses or breaching data integrity.

- ✓ An Applet class does not have any main() method. It is viewed using JVM. The JVM can use either a plug-in of the Web browser or a separate runtime environment to run an applet application(Appletviewer)

There are two types of the applet -

- Applets based on the AWT(Abstract Window Toolkit) package by extending its Applet class.
- Applets based on the Swing package by extending its JApplet class

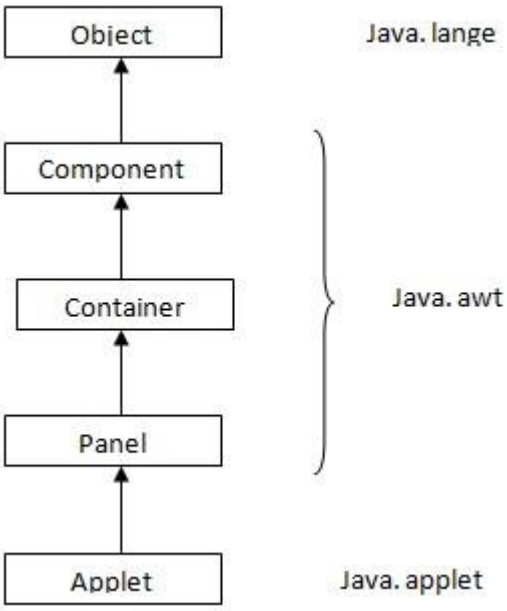
Advantage of Applet

- ✓ 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

- ✓ Plugin is required at client browser to execute applet.

Class Hierarchy of Applet



- ✓ The AWT allows us to use various graphical components. When we start writing any applet program we essentially import two packages namely – java.awt and java.applet.

- ✓ The java.applet package contains a class Applet which uses various interfaces such as AppletContext, AppletStub and AudioClip. The applet class is an extension of Panel class belonging to java.awt package.

- ✓ To create an user friendly graphical interface we need to place various components on GUI window. There is a Component class from java.awt package which derives several classes for components. These classed include Check box, Choice, List, buttons and so on. The Component class in java.awt is an abstract class.

- ✓ The class hierarchy for Applets is as shown in Fig. As displayed in the above diagram, Applet class extends Panel. Panel class extends Container which is the subclass of Component.

Lifecycle of Java Applet

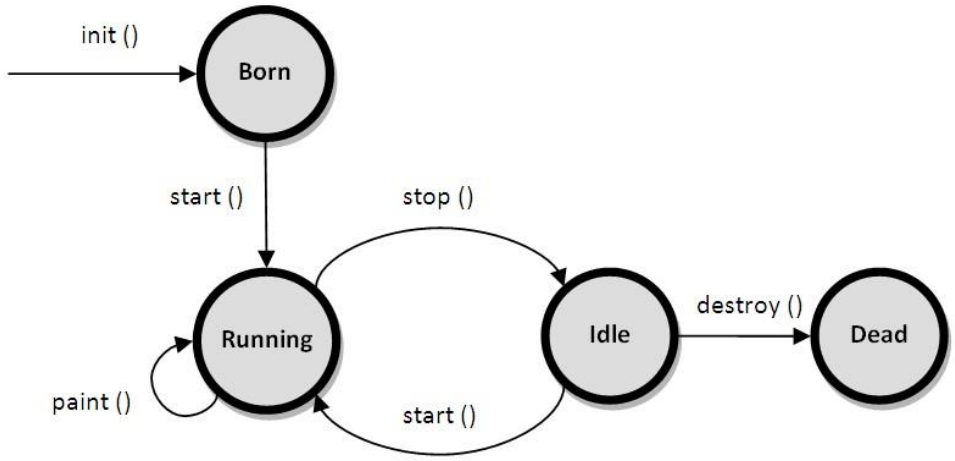
Applet life cycle defined as how the object created, started, stopped and destroyed during the entire execution of the application is said to applet life cycle. Applet life cycle has 5 methods as follows

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

Lifecycle methods for Applet:

The **java.applet.Applet** class 4 life cycle methods and **java.awt.Component** class provides 1 life cycle methods for an applet.

The life cycle of an applet is as shown in the figure below:



As shown in the above diagram, the life cycle of an applet starts with init() method and ends with destroy() method. Other life cycle methods are start(), stop() and paint().

The methods to execute only once in the applet life cycle are init() and destroy(). Other methods execute multiple times.

Below is the description of each applet life cycle method:

1.public void init(): is used to initialized the Applet. It is invoked only once.

The init() method is the first method to execute when the applet is executed. Variable declaration and initialization operations are performed in this method.

2.public void start(): is invoked after the init() method or browser is maximized. It is used to start the Applet.

The start() method contains the actual code of the applet that should run. The start() method executes immediately after the *init()* method. It also executes whenever the applet is restored, maximized or moving from one tab to another tab in the browser.

3.public void stop(): is used to stop the Applet. It is invoked when Applet is stop or browser is minimized.

4.stop(): The stop() method stops the execution of the applet. The stop() method executes when the applet is minimized or when moving from one tab to another in the browser.

5.**public void destroy():** is used to destroy the Applet. It is invoked only once.

The destroy() method executes when the applet window is closed or when the tab containing the webpage is closed. *stop()* method executes just before when destroy() method is invoked. The destroy() method removes the applet object from memory.

6.paint(): The paint() method is used to redraw the output on the applet display area. The paint() method executes after the execution of *start()* method and whenever the applet or browser is resized.

The method execution sequence when an applet is executed is:

- init()
- start()
- paint()

The method execution sequence when an applet is closed is:

- stop()
- destroy()

java.applet.Applet class

Applet is a Java program that can be transported over the internet and executed by a Javaenabled web-browser(if a browser is supporting the applets) or an applet can be executed using the **appletviewer** utility provided with JDK. An appLet us created using the **Applet** class, which is a part of **java.applet** package. Applet class provides several useful methods to give you full control over the execution of an applet.

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

Note :

At present, some modern versions of browsers like Google Chrome and Mozilla Firefox have stopped supporting applets, hence applets are not displayed or viewed with these browsers although some browsers like Internet Explorer and Safari are still supporting applets. **Some methods of Applet class**

Methods	Description
void init()	The first method to be called when an applet begins its execution.
void start()	Called automatically after **init()** method to start the execution of an applet.
void stop()	Called when an applet has to pause/stop its execution.
void destoy()	Called when an appLet us finally terminated.
String getParameter(String ParameterName)	Returns the value of a parameter defined in an applet
Image getImage(URL *url*)	Returns an Image object that contains an image specified at location, *url*
void play(URL *url*	Plays an audio clip found at the specified at location, *url*
showStatus(String str)	Shows a message in the status window of the browser or appletviewer.

These include some more methods that do the following in an applet−

- Get applet parameters
- Get the network location of the HTML file that contains the applet
- Get the network location of the applet class directory
- Print a status message in the browser
- Fetch an image
- Fetch an audio clip
- Play an audio clip
- Resize the applet

Additionally, the Applet class provides an interface by which the viewer or browser obtains information about the applet and controls the applet's execution. The viewer may −

- Request information about the author, version, and copyright of the applet
- Request a description of the parameters the applet recognizes
- Initialize the applet
- Destroy the applet
- Start the applet's execution
- Stop the applet's execution

The Applet class provides default implementations of each of these methods. Those implementations may be overridden as necessary.

java.awt.Component class

The Component class provides 1 life cycle method of applet.

1. **public void paint(Graphics g):** is used to paint the Applet. It provides Graphics class object that can be used for drawing oval, rectangle, arc etc.

Who is responsible to manage the life cycle of an applet? Java Plug-in software.

How to run an Applet?

There are two ways to run an applet

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

Simple example of Applet by html file:

To execute the applet by html file, create an applet and compile it. After that create an html file and place the applet code in html file. Now click the 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);
}
 }
```

> *Note: class must be public because its object is created by Java Plugin software that resides on the browser.*

myapplet.html

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

Simple example of Applet by appletviewer tool:

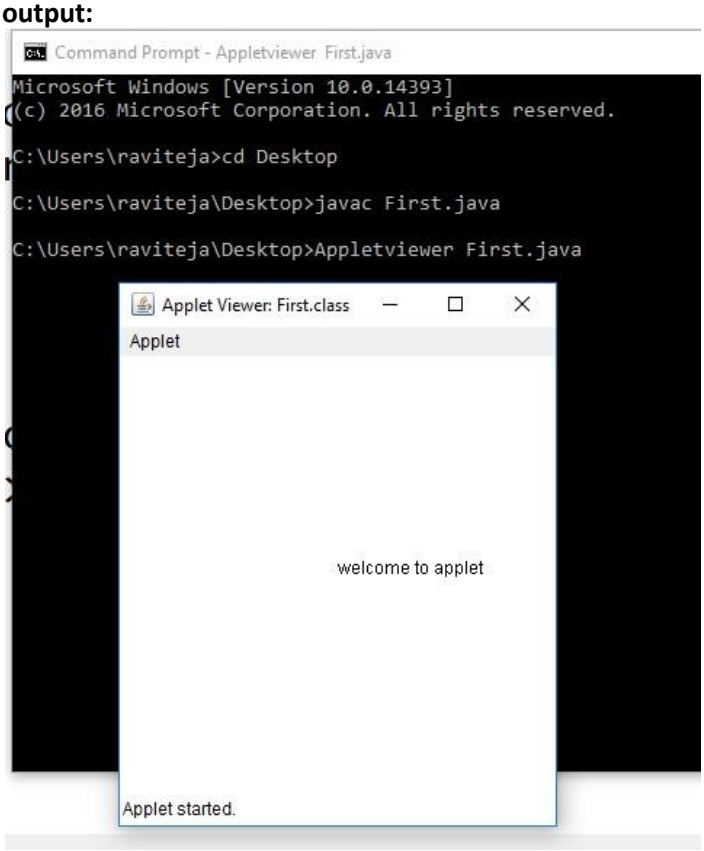
To execute the applet by appletviewer tool, create an applet that contains applet tag in comment and compile it. After that run it by: appletviewer First.java. Now Html file is not required but it is for testing purpose only.

```java
//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>
*/
```

To execute the applet by appletviewer tool, write in command prompt:

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

output:



Passing Parameter in Applet

User-define Parameter can be applied in applet using <PARAM…> tags. Each <PARAM…> tag has a name and value attribute. **Example:**

name = color Value = red **Syntax:**

<PARAM name = …… Value = "……" >

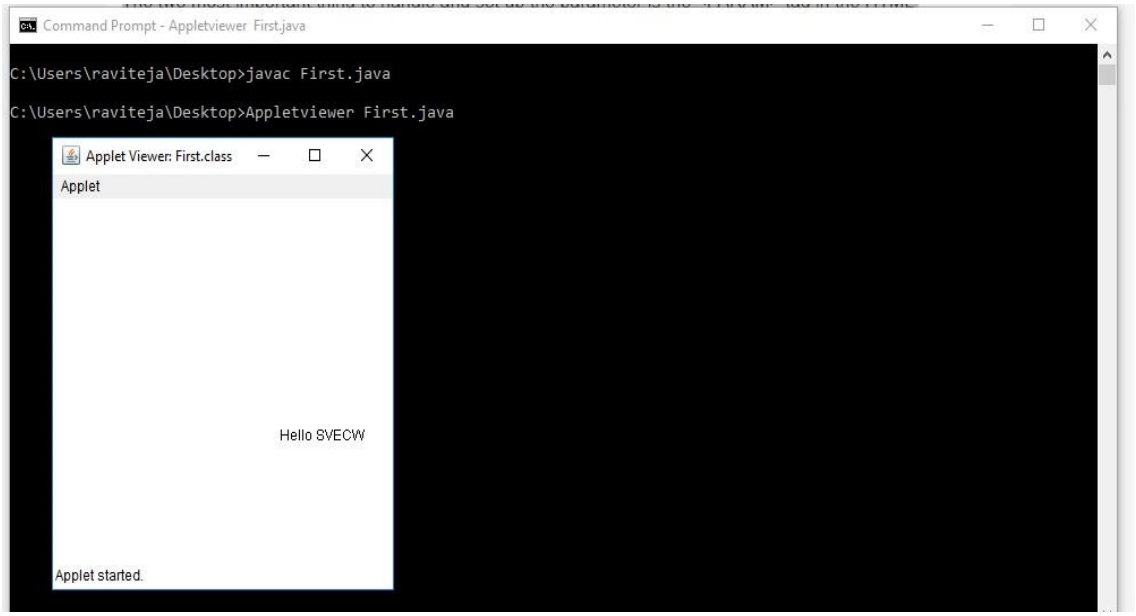
In an applet code, applet can refer to a parameter by its name and then find its value.

- ✓ The two most important thing to handle and set up the parameter is the <PARAM> tag in the HTML document and an applet code to parse this parameter.
- ✓ init() method is used to get hold of the parameters which is defined in the <PARAM> tags. And getParameter() method is used for getting the parameters.

In Applet, Parameters are passed on applet when it is loaded.

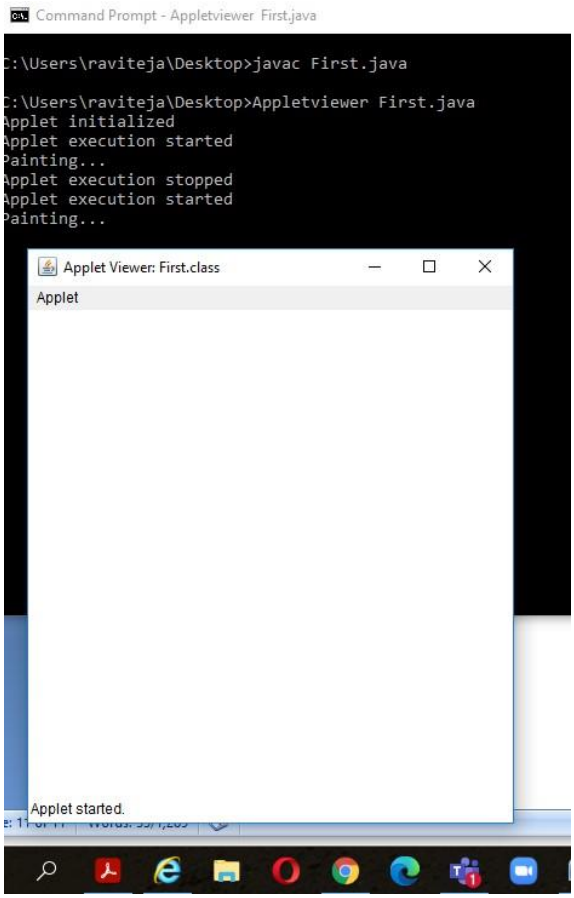
```
import java.applet.*; import
java.awt.*;

/*<applet code=First.class height=300 width=300>
<param Name="pname" value="SVECW">
</applet>*/
public class First extends Applet
{
 String          str;
public void init()
 {
  str=getParameter("pname");
  if (str == null)
  str = "Welcome to SVECW";

  str = "Hello " + str;
 }
  public void paint(Graphics g)
  {
   g.drawString(str, 200, 200);
  }
}
```

Output:



Example Applet program

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

/*<applet code=First.class height=400 width=400>

</applet>*/
public class First extends Applet
{
        public void init()
        {
                System.out.println("Applet initialized");
        }
        public void start()
        {
                System.out.println("Applet execution started");
        }
        public void stop()
        {
                System.out.println("Applet execution stopped");
        }
        public void paint(Graphics g)
        {
                System.out.println("Painting...");
        }
        public void destroy()
        {
                System.out.println("Applet destroyed");
        }
}
```



Java Applets - Programming Examples For more examples please refer following url https://www.tutorialspoint.com/javaexamples/java_applets.htm

Displaying Graphics in Applet java.awt.Graphics class provides many methods for graphics programming.

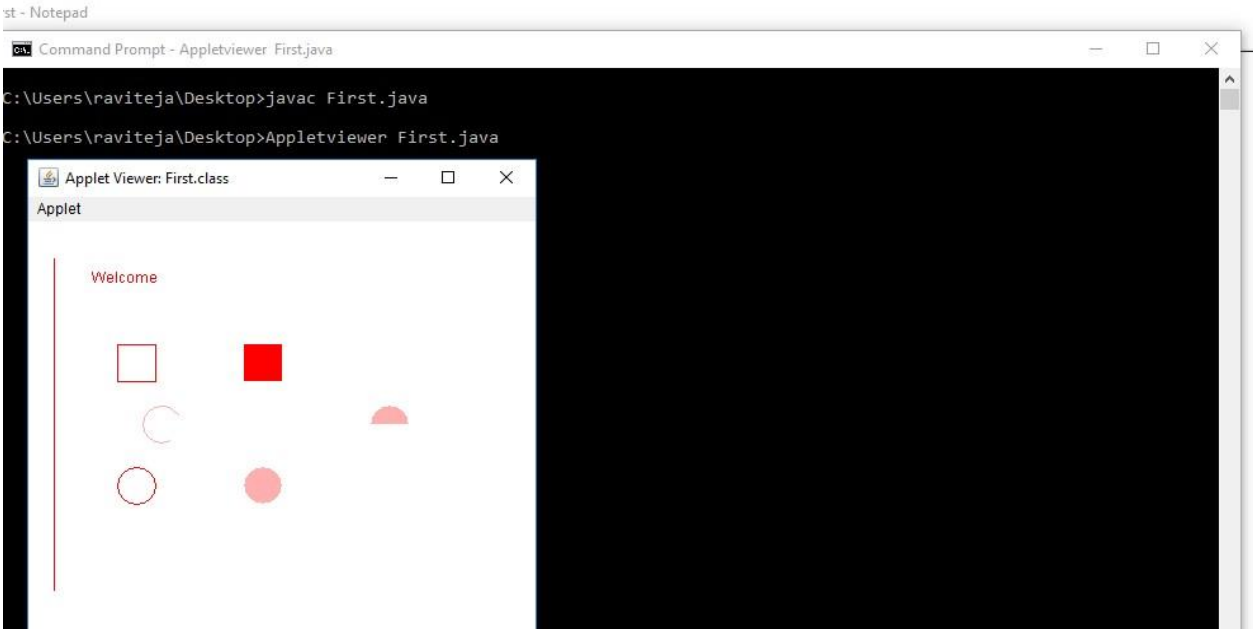
Commonly used methods of Graphics class:

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 boolean drawImage(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, int startAngle, int arcAngle):** is used draw a circular or elliptical arc.
9. **public abstract void fillArc(int x, int y, int width, int height, int startAngle, int arcAngle):** 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 of Graphics in applet:

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

/*<applet code=First.class height=400 width=400>
</applet>*/

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

public class First extends Applet{

public void paint(Graphics g){   g.setColor(Color.red);
g.drawString("Welcome",50, 50);
g.drawLine(20,30,20,300);
g.drawRect(70,100,30,30);
g.fillRect(170,100,30,30);
g.drawOval(70,200,30,30);

g.setColor(Color.pink);
g.fillOval(170,200,30,30);
g.drawArc(90,150,30,30,30,270);
g.fillArc(270,150,30,30,0,180);

}
}
```



repaint Method in Java

Repaint method in java is available in java.applet.Applet class and is a final method which is used whenever we want to call update method along with the call to paint method, call to update method clears the current window, performs update, and afterwards calls paint method.

How repaint Works in Java?

- ✓ Repaint method is a final method available in Applet class and that's why it cannot be overridden. Whenever repaint method is to be used it should be directly called from subclasses of Applet class.

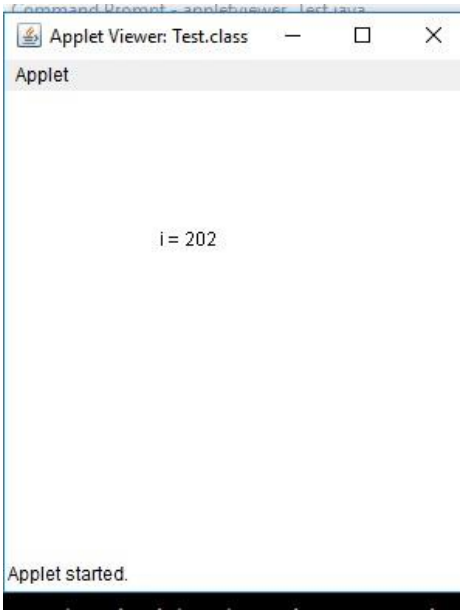
- ✓ Repaint method is responsible for handling update to paint cycle of applet. Whenever we want a component to repaint itself we need to call repaint method.

- ✓ In case we have made changes to appearance of a component but have not made any changes to its size than we can call repaint method to update new appearance of component on the graphical user interface. Repaint method is an asynchronous method of applet class. When call to repaint method is made, it performs request to erase and perform redraw of the component after a small delay in time.

- ✓ Whenever repaint method is invoked from a component, a request is sent to the graphical user interface which communicates to the graphical user interface to perform some action at a future instance of time. The whole idea behind repaint method is to avoid call to paint () method directly.

```
import java.applet.Applet;
import java.awt.Graphics;

public class Javaapp extends Applet {

   int i;

   public void paint(Graphics g)
   {
      g.drawString("i = "+i, 100, 100);

      try{
         Thread.sleep(1000);
      }catch(InterruptedException ex){}
          i++;
repaint();
   }
}

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

Output:



Paint() method in Applet

In this article, we are going to understand how to print a message in an applet window by implementing the method **paint()** of the Applet class. Within the method **paint()** we will call the **drawString()** method to print a text message in the applet window.

when the paint() method is invoked?

The method **paint()** is automatically called whenever there is a need to display an applet window, this need could arise in certain situations -

- When an applet window is brought up on the screen for the first time.
- When an applet window is brought up on the screen from a minimized state, this leads the redrawing of the applet window and hence the **paint()** method is automatically called.
- When an applet window is stretched to a new size, this leads to redrawing of the applet window to a new size and hence the **paint()** method is automatically called.

signature of paint() method

```
public void paint(Graphics g)
```

The method **paint()** gives us access to an object of type **Graphics** class. Using the object of the Graphics class, we can call the **drawString()** method of the Graphics class to write a text message in the applet window.

Signature of drawString() method

```
public
void drawString(String, int x, int y)
```

The method **drawString()** takes a String which will be printed in the applet window, starting from left-top corner at the coordinates *x* and *y*.

Writing a message on the applet window.

- In the upcoming code, we are implementing the **paint()** method in our applet class.
- Within the **paint()** method, the **drawString()** method is called to write a message in the applet window.

```
import java.awt.*; import
java.applet.*;
/*
<applet  code="Applet1" width=400 height=200>
</applet>
*/

public class Applet1 extends Applet
{
String str ="";

public void init()
{
str="init-";
}

public void start()
{
str=str+"start-";
```

```
}

public void stop()
{
str=str+"stop-";
}

public void paint(Graphics g)
{
str=str+"paint-";;
g.drawString(str,40,100);
g.drawString("Hello from the Applet.", 40,40);
g.drawString("How are you doing?", 40, 60);
g.drawString("We wish you a pleasant day today.", 40, 80);
}
}
```

Output: