Chapter 2
Java Application and Applet

Applet

- An applet is a special kind of java program that a browser enabled with java technology can download from the internet and run.
- An applet is also window based event driven program that runs in web browser.
- An applet is a Java program that can be embedded into a web page. It runs inside the web browser and works at client side.
- An applet is embedded in an HTML page using the APPLET or OBJECT tag and hosted on a web server.
- The applet class are contained in the **java.applet** package so the applet must import it.
- It also must import **java.awt** for graphics**.**
- Applet are executed by either a **web browser or applet viewer** provided by JDK**.**
- An applet is subclass of **Applet** class or **JApplet class.**

Types of Applet

1. AWT Applet
   The first type of applets uses the Abstract Window Toolkit (AWT) to provide the graphic user interface (or use no GUI at all). This style of applet has been available since Java was first created.
2. Swing Applet
   The second type of applets is those based on the Swing class JApplet. Swing applets use the Swing classes to provide the GUI.
   Swing offers a richer and often easier-to-use user interface than does the AWT. Thus, Swing-based applets are now the most popular. However, traditional AWT-based applets are still used, especially when only a very simple user interface is required.

Characteristics of Applet

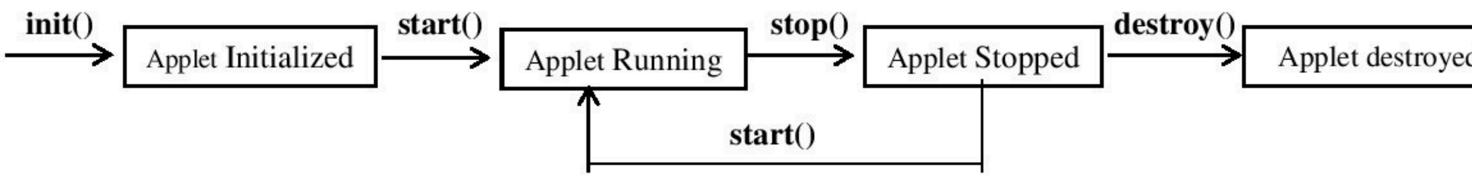
1. An applet is a Java class that extends the java.applet.Applet class.
2. A main () method is not invoked on an applet, and an applet class will not define main().
3. Applets are designed to be embedded within an HTML page. When a user views an HTML page that contains an applet, the code for the applet is downloaded to the user's machine.
4. An applet can't read or write file to the local computer.
5. An applet can capture mouse input and also have controls like buttons, text fields.
6. Unlike application, it is not necessary to write methods such as setVisible() or show() to display the applet.
7. We define the height and width of the applet in the HTML file not in applet program.

Application vs Applet

Application	**Applet**
1. Applications are the java program that can run directly on our machine.	1. Applets are the java program that can run over the internet using client/server architecture.
2. Application do have main() method.	2. Applet do not have main() method.
3. Application can read or write to the file in the local computer.	3. Applet can not read or write to a file in the local computer.
4. GUI components are added to the content pane of its Container class such as JFrame or Frame.	4. GUI components are added directly to the Applet.
5. Java applications are compiled suing the **javac** command and run using the **java** command.	5. Java Applet are also compiled using the **javac** command but are run either with a web browser or with the **appletviewer** command
6. Constructors are used for initialization.	7. Application constructor is replaced by start() and init() method.

8. Java applications are mostly used to write commercial application.	9. With the limitation of disk and network access, it would be difficult to write commercial applications.

<u>Life Cycle of an Applet</u>



The life cycle of an applet consists of four states and has four methods to change the state of an applet. The four states are explained below.

1. **New or initialization state**: Applet enters the new state when it is first loaded. This is achieved by calling the init() method . init() method is used to initialize the applet each time it is loaded. We define the action such as setting color, initial value, loading images etc. within the init() method.
2. **Running State:** An applet enters the running state when the system calls the start () method. The start () method is automatically called after the init() method. It is also called whenever the user returns to the page containing the applet after having gone off to other pages.
3. **Idle or Stopped State:** An applet come to stopped state when it is stopped from execution. We can stop the execution of applet by invoking the stop () method however the applet automatically goes to stopped state whenever we go away from the page containing that applet.
4. **Dead State:** An applet comes to dead state when it is removed from the memory. The destroy () method is automatically invoked when we quit the browser to leave resources behind after a user leaves the page that contains the applet.

In addition to these four method defined by Applet, another method **paint()** method is defined by the AWT component. This method is invoked immediately after the start () method, and also any time the applet needs to repaint itself in the browser. So paint () method is called each time your output must be redrawn in the case like applet widow is minimized and then restored. The paint () method has one parameter of Graphics for describing the graphical environment in which the applet is running.
Also AWT define another method Repaint () for updating a small portion of window because it takes much time to update the entire window.
when an applet begins, following sequence of method call take place

1. init()
2. start()
3. paint()

when an applet is terminated, following sequence of method call takes place

1. stop()
2. destroy();

Note

- **Applete do not need to override those method they don't use.**
- **init() ,destroy() method are called only once, start() and stop() method are called one or more times in the life cycle of applet.**
- **getSize().width returns the width of the applet, getSize().height returns the height of the applet.**
- **showStatus(String txt) method is used to output a message to the status window of the browser or applet viewer on which it is running.**

Applet Architecture

- An applet is a GUI-based program. As such, its architecture is different from the console-based programs.
- First, applets are event driven. An applet resembles a set of interrupt service routines. Here is how the process works. An applet waits until an event occurs. The run-time system notifies the applet about an event by calling an event handler that has been provided by the applet. Once this happens, the applet must take appropriate action and then quickly return
- Second, the user initiates interaction with an applet—not the other way around. As you know, in a console-based program, when the program needs input, it will prompt the user and then call some input method, such as readLine( ). This is not the way it works in an applet. Instead, the user interacts with the applet as he or she wants, when he or she wants. These interactions are sent to the applet as events to which the applet must respond. For example, when the user clicks the mouse inside the applet's window, a mouse-clicked event is generated. If the user presses a key while the applet's window has input focus, a keypress event is generated. An applet can contain various controls, such as push buttons and check boxes. When the user interacts with one of these controls, an event is generated.

Use of Html APPLET tag

The APPLET tag is used to start an applet from both HTML document and from an applet viewer provided by JDK.

Note: Bracket items are optional

```
<APPLET
        [CODEBASE=”codeBaseURL”]
        CODE=”appletfile”
        [ALT=”Alternate text”]
        [NAME=”appletInstanceName”]
        WIDTH=pixels
        HEIGHT=pixels
        [ALIGN=alignment]
        [VSPACE=pixels]
        [HSPACE=pixels]
>
[<PARAM NAME=AttributeName1 VALUE=AttributeValue1>]
[<PARAM NAME=AttributeName2 VALUE=AttributeValue2>]

……………….
</APPLET>
```

Where,

1. **CODEBASE** is an optional attribute that specifies the base URL for the applet code, which is directory that will be searched for the applet executable class file. If not used, the HTML document's URL directory is used as the CODEBASE.
   Example: **CODEBASE=**"http://amrood.com/applets"
2. **CODE** is required attribute that gives th name of the file containing your applet's compiled .class file.
   Example: **CODE=**"Demo.class";
3. **ALT** tag is optional and is used to specify a short text message that should be displayed of the browser understands the APPLET tag but can't currently run java applets.
   Example: **ALT="Menu Applet"**
4. **Name** is an optional attribute used to specify a name for the applet instance. We can obtain an applet by name using getApplet() method.
   Example: NAME="myapplet"
5. **WIDTH and HEIGHT** are required attribute that give the size in pixels of the applet display area.
   Example: **WIDTH=300**
   HEIGHT=600

6. **ALIGN** is an optional attribute that specifies the alignment of the applet. Some mostly used alignments are LEFT, RIGHT, TOP, BOTTOM, MIDDLE etc.
   Example: **ALIGN="Right"**
7. **VSPACE and HSPACE** are optional attributes that are used to specify the space in pixels. VSPACE specifies space in pixels above and below the applet while HSPACE specify the space in pixels on each side of the applet.
   Example: **HSPACE=20**
   VSPACE=20
8. **PARAM NAME and VALUE** are the optional tag that allow us to specify the applet specific argument name and its value in an HTML page. The attributes are accessed with the getParameter() method.
   Example:

```
<PARAM NAME="Name" VALUE="Bhesh">
<PARAM NAME="Roll" VALUE=5>
```

Steps to create and execute a simple applet to display your name

1. Open any text editor like notepad and write java applet program as below.

```
import java.applet.Applet;
import java.awt.Graphics;
import java.awt.Color;
public class Demo extends Applet
{
        String name="";
        public void init()
        {
                name="Bhesh Bahadur Thapa";
                setBackground(Color.RED);
        }
        public void paint(Graphics g)
        {
                g.setColor(Color.WHITE);//sets the font color to white
                g.drawString(name,10,10);// write the string at pixel position (10,10)
        }
}
```

2. After writing the above code, save it as java file i.e. Demo.java
3. Open command prompt and navigate to folder you saved the Demo.java file, say Desktop

```
C:\Desktop>
```

4. Compile the java file using the command 'javac' as below

```
C:\Desktop> javac Demo.java
```

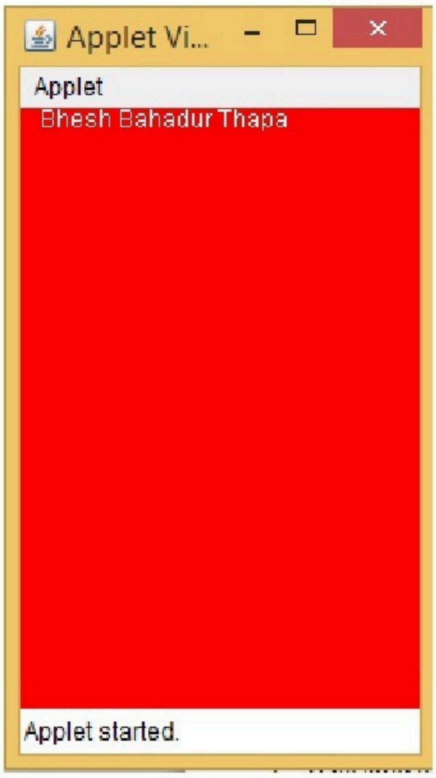
5. If no error occurs during the compilation, Demo.class file will be created.
6. Again, Open any new text editor like notepad and write the APPLET tag as below.

```
<applet code="Demo.class" width=200 height=300> </applet>
```

7. After writing the above tag, save with .html extension i.e. Demo.html
8. Runt the applet using one of the following tool.
   i. Web browser: Double click the html file to open in java enabled web browser.
   ii. Applet Viewer: To run the applet in applet viewer provided by JDK, we use the following command

```
C:\Desktop> appletviewer Demo.html
```

Following output is observed:



```
Bhesh Bahadur Thapa
```

```
Applet started.
```

<u>Passing Parameters to applets</u>

We use <PARAM> tag to define user defined parameters. A <PARAM> tag has two attributes NAME and VALUE. The general syntax is

<PARAM Name=”attributeName” Value=”attributeValue”>

The getParameter() method is used to access the attribute value.

Example:

Write an applet program to input name, roll and marks in three different subject. The program must display all the information along with total mark obtained using green background and Black font. The applet size is (200,300).

Program file Demo.java

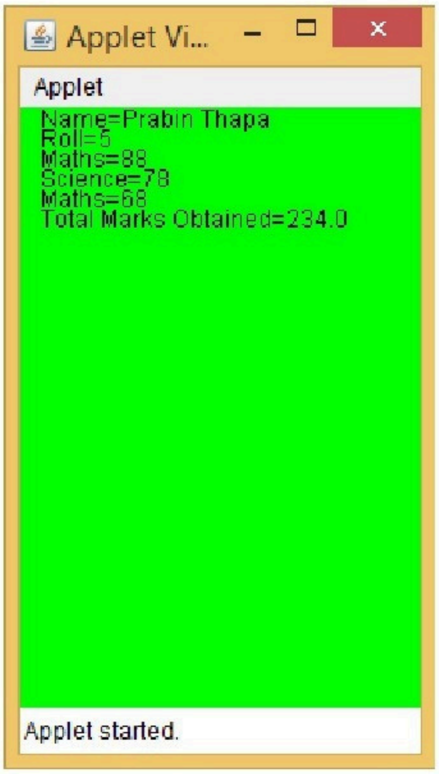
```
import java.awt.*;
import java.applet.*;
public class Demo extends Applet
{
        String name,roll,s1,s2,s3;
        double total;
        public void init()
        {
                setBackground(Color.GREEN);
                name=getParameter("Stdname");
                roll=getParameter("Stdroll");
                s1=getParameter("Maths");
                s2=getParameter("Science");
                s3=getParameter("English");
                total=Double.parseDouble(s1)+Double.parseDouble(s2)+Double.parseDouble(s3);
        }
```

```
                public void paint(Graphics g)
                {
                        g.setColor(Color.BLACK);
                        g.drawString("Name="+name,10,10);
                        g.drawString("Roll="+roll,10,20);
                        g.drawString("Maths="+s1,10,30);
                        g.drawString("Science="+s2,10,40);
                        g.drawString("Maths="+s3,10,50);
                        g.drawString("Total Marks Obtained="+total,10,60);
                }
        }
```

HTML file Demo.html

```
                <applet Code="Demo.class" Width=200 Height=300>
                        <Param Name=Stdname Value="Prabin Thapa">
                        <Param Name=Stdroll Value="5">
                        <Param Name=Maths Value="88">
                        <Param Name=Science Value="78">
                        <Param Name=English Value="68">
                </applet>
```

Output:



```
Name=Prabin Thapa
Roll=5
Maths=88
Science=78
Maths=68
Total Marks Obtained=234.0
```

```
Applet started.
```

<u>**Taking input from the users though Applet Window**</u>
We can use various AWT and Swing components such as TextField, Button etc to take input from the users.

Write an applet program to find the sum and difference of two number using AWT components.

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

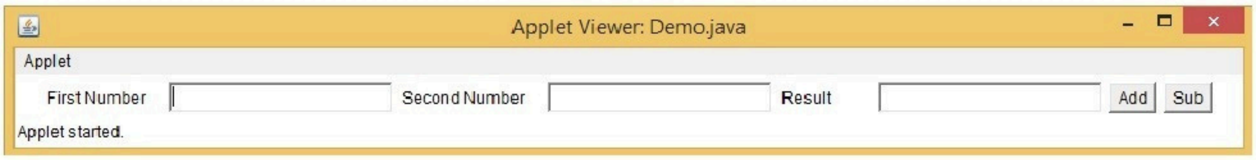
```
import java.awt.event.*;
public class Demo extends Applet implements ActionListener
{
        Button btn_add,btn_sub;
        Label lbl_fn,lbl_sn,lbl_res;
        TextField txt_fn,txt_sn,txt_res;
        public void init()
        {
                setLayout(new FlowLayout());
                lbl_fn=new Label("First Number");
                add(lbl_fn);
                txt_fn=new TextField(20);
                add(txt_fn);
                lbl_sn=new Label("Second Number");
                add(lbl_sn);
                txt_sn=new TextField(20);
                add(txt_sn);
                lbl_res=new Label("Result ");
                add(lbl_res);
                txt_res=new TextField(20);
                add(txt_res);
                btn_add=new Button("Add");
                btn_add.addActionListener(this);
                add(btn_add);
                btn_sub=new Button("Sub");
                btn_sub.addActionListener(this);
                add(btn_sub);
        }

        public void actionPerformed(ActionEvent e)
        {
                double fn=Double.parseDouble(txt_fn.getText());
                double sn=Double.parseDouble(txt_sn.getText());
                if(e.getSource()==btn_add)
                {
                        double res=fn+sn;
                        txt_res.setText(Double.toString(res));
                }
                else if(e.getSource()==btn_sub)
                {
                        double res=fn-sn;
                        txt_res.setText(Double.toString(res));
                }
        }
}
```

HTML file

```
<applet Code="Demo.java" Width="900" Height="30"></applet>
```

Output:



Applet started.