Unit 8 Applets

Structure:

8.1 Introduction

In the last unit, we have discussed various stream classes in Java. In this unit, we will discuss Java applets. Internet as of today has become an inseparable part of life used widely for accessing libraries, getting information on latest happenings, transferring information, sending email and communicating with others. Java programs written to run on World Wide Web (WWW) are called as **applets**. Since applets are run on different machines, security becomes a critical issue. Another probable disadvantage of applets would be the slight flickering that would be inevitable in animations but there is light at the end of the tunnel since a separate buffer can be used for the images which can then be superimpose on to the window or canvas.

Objectives:

After studying this unit, you should be able to:

- explain applet and applet classes
- explain the Life Cycle of an applet
- discuss the Graphics class
- paint the applet

- create user interfaces for applets
- add components to user interfaces
- create AWT (Abstract Windowing Toolkit) controls

8.2 What are Applets?

An applet is a Java program that can be embedded in a web page. Java applications are run by using a Java interpreter. Applets are run on any browser that supports Java. Applets can also be tested using the ***appletviewer*** tool included in the Java Development Kit. In order to run an applet it must be included in a web page, using HTML tags. When a user browses a web server and it runs applets on the user's system. Applets have certain restrictions put on them.

- They can not read or write files on the user's system.
- They can not load or run any programs stored on the user's system.

All applets are subclasses of the **Applet** class in the ***java.applet*** package. Applets do not have main() method. All applets must be declared **public**. An applet displays information on the screen by using the paint method. This method is available in ***java.awt.Component*** class. This method takes an instance of the class ***Graphics*** as parameter. The browser creates an instance of Graphics class and passes to the method paint(). Graphics class provides a method ***drawString*** to display text. It also requires a position to be specified as arguments.

8.3 The Applet Class

The ***java.applet*** package is the smallest package in the Java API. The Applet class is the only class in the package. An applet is automatically loaded and executed when you open a web page that contains it. The Applet class has over 20 methods that are used to display images, play audio files, and respond when you interact with it.

The applet runs in a web page that is loaded in a web browser. The environment of the applet is known as the context of the applet. You can retrieve the context using the ***getAppletContext()*** method. The life cycle of an applet is implemented using ***init(), start(), stop(),*** and ***destroy()*** methods.

Use ***javac*** to compile applets and ***appletviewer*** to execute them. You can view applets in any browser that is Java-enabled.

Self Assessment Questions

1. Java programs written to run on World Wide Web (WWW) are called _____.
2. Applets can be tested using the _____ tool included in the Java Development Kit.

8.4 The Applet and HTML

The Applet tag is used to embed an applet in an HTML document. The Applet tag takes zero or more parameters.

The Applet Tag

The Applet tag is written in the **body** tag of an HTML document.

Syntax:

<APPLET

CODE = "name of the class file that extends java.applet.Applet"

CODEBASE = "path of the class file"

HEIGHT = "maximum height of the applet, in pixels"

WIDTH = "maximum width of the applet, in pixels"

VSPACE = "vertical space between the applet and the rest of the HTML"

HSPACE = "horizontal space between the applet and the rest of the HTML"

ALIGN = "alignment of the applet with respect to the rest of the web page"

ALT = "alternate text to be displayed if the browser does not support applets"

>

<PARAM NAME="parameter_name" value="value_of_parameter">

……..

</APPLET>

The most commonly used attributes of the Applet tag are **CODE, HEIGHT, WIDTH, CODEBASE** and **ALT**. You can send parameters to the applet using the **PARAM** tag. The PARAM tag must be written between <APPLET> and </APPLET>

Example:

```
<applet
        Code = "clock. class"
        Height = 200
        Width = 200 >
</applet>
```

8.5 Life Cycle of an Applet

You can describe the life cycle of an applet through four methods. These methods are:

- The ***init()*** method.
- The ***start()*** method.
- The ***stop()*** method.
- The ***destroy()*** method.

The init() method

The init() method is called the first time an applet is loaded into the memory of a computer. You can initialize variables, and add components like buttons and check boxes to the applet in the init() method.

The start() method

The start() method is called immediately after the init() method and every time the applet receives focus as a result of scrolling in the active window. You can use this method when you want to restart a process, such as thread animation.

The stop() method

The stop() method is called every time the applet loses the focus. You can use this method to reset variables and stop the threads that are running.

The destroy() method

The destroy() method is called by the browser when the user moves to another page. You can use this method to perform clean-up operations like closing a file.

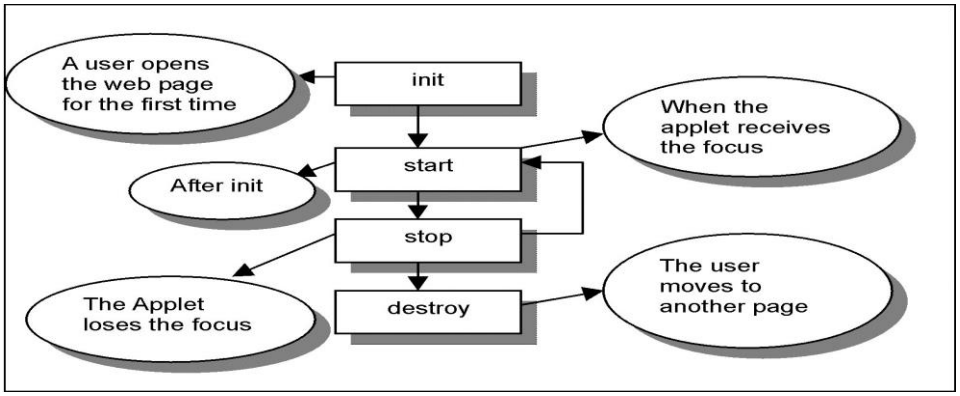

Figure 8.1: Diagram depicting the life cycle of an applet.

It is not mandatory to use any or all the methods of the applet. These methods are called automatically by the Java environment, and hence, must be declared public. None of the methods accept parameters.

Example:

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

All but the most trivial applets override a set of methods that provides the basic mechanism by which the browser or appletviewer interfaces to the applet and controls its execution. Four of these methods —**init()**, **start()**, **stop()**, and **destroy() –** are defined by **Applet**. Another, **paint()** method, is defined by the AWT **Component** class. Default implementations for all of these methods are provided. Applets do not need to override those methods that they do not use. However, only very simple applets will not need to define all of them. These five methods can be assembled into the skeleton shown here:

```
// An Applet skeleton.
import java.awt.*;
import java.applet.*;
/*
<applet code="AppletSkel" width=300 height=100>
```

```
</applet>
*/
public class AppletSkel extends Applet {
// Called first.
public void init() {
// initialization
}
/* Called second, after init(). Also called whenever
the applet is restarted. */
public void start() {
// start or resume execution
}
// Called when the applet is stopped.
public void stop() {
// suspends execution
}
/* Called when applet is terminated. This is the last
method executed. */
public void destroy() {
// perform shutdown activities
```

Self Assessment Questions

3. The Applet tag is written in the _____ tag of an HTML document.
4. Parameters can be sent to the applet using _____ tag.

8.6 The Graphics Class

The Graphics class is an abstract class that represents the display area of the applet. It is a part of the ***java.awt*** package. It is used for drawing on the applet's display.

The Graphics class provides methods to draw a number of graphical figure including

- Text
- Lines

- Circle and ellipse.
- Rectangle and polygons.
- Images.

A few of the methods are given below:

public abstract void drawString (String text, int x, int y)

public abstract void drawLine (int x1, int y1, int x2, int y2)

public abstract void drawRect (int x1, int y1, int width, int height)

public abstract void fillRect (int x, int y1, int width, int height)

public abstract void clearRect (int x, int y1, int width, int height)

public abstract void draw3DRect (int x1, int y1, int width, int height, boolean raised)

public abstract void drawRoundRect (int x1, int y1, int width, int height, int arcWidth, int arcHeight)

public abstract void fillRoundRect (int x1, int y1, int width, int height, int arcWidth, int arcHeight)

public abstract void drawOval (int x1, int y1, int width, int height)

public abstract void fillOval (int x, int y1, int width, int height)

You cannot create an object of the Graphics class since it is abstract. You can use the method ***getGraphics()*** to obtain an object of the class.

8.7 Painting the Applet

When you scroll an applet, the screen has to be refreshed to show the new content. Windows handles this by marking the area (rectangle) that has to be refreshed. The area is then painted to display the result of scrolling. This is handled by the ***update()*** and ***paint()*** methods.

The update() method

The ***update()*** method takes a Graphics class object as a parameter. When the applet area needs to be redrawn, the Windows system starts the painting process. The update() method is called to clear the screen and calls the paint() method. The screen is then refreshed by the system.

The paint() method

The ***paint()*** method draws the graphics of the applet in the drawing area. The method is automatically called the first time the applet is displayed on

the screen and every time the applet receives the focus. The paint() method can be triggered by invoking the ***repaint()*** method.

The paint() method of the applet takes an object of the Graphics class as a parameter.

Example:

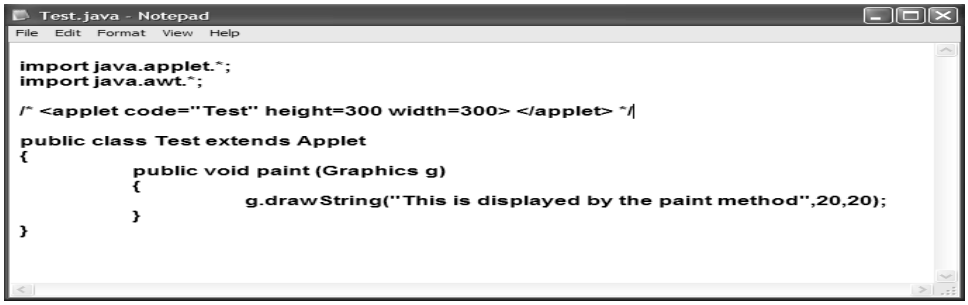

Figure 8.1: Coding (Step 1)

Figure 8.2: Compiling and Executing (Step 2)

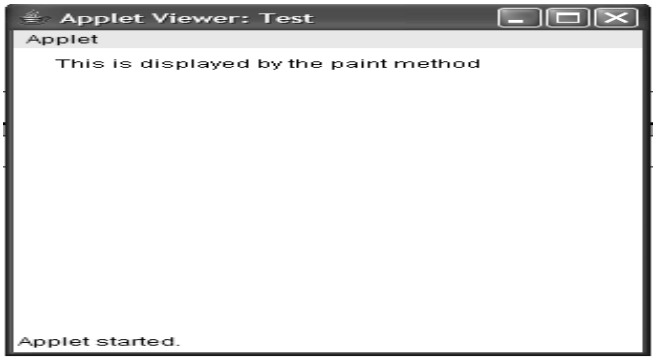

Figure 8.3: Output (Step 3)

The repaint() method

You can call the repaint() method when you want the applet area to be redrawn. The repaint() method calls the update() method to signal that the applet has to be updated. The default action of the update () method is to clear the applet area and call the paint() method. You can override the update() method if you do not want the applet area to be cleared.

The following program uses the paint() and repaint() methods to check when the init(), start(), and stop() methods of an applet are called.

As a general rule, an applet writes to its window only when its **update( )** or **paint( )** method is called by the AWT. This raises an interesting question: How can the applet itself cause its window to be updated when its information changes? For example, if an applet is displaying a moving banner, what mechanism does the applet use to update the window each time this banner scrolls? Remember, one of the fundamental architectural constraints imposed on an applet is that it must quickly return control to the AWT run-time system. It cannot create a loop inside **paint( )** that repeatedly scrolls the banner, for example. This would prevent control from passing back to the AWT. Given this constraint, it may seem that output to your applet's window will be difficult at best. Fortunately, this is not the case. Whenever your applet needs to update the information displayed in its window, it simply calls **repaint( )**.

The **repaint( )** method is defined by the AWT. It causes the AWT run-time system to execute a call to your applet's **update( )** method, which, in its default implementation, calls **paint( )**. Thus, for another part of your applet to output to its window, simply store the output and then call **repaint( )**. The AWT will then execute a call to **paint( )**, which can display the stored information. For example, if part of your applet needs to output a string, it can store this string in a **String** variable and then call **repaint( )**. Inside **paint( )**, you will output the string using **drawString( )**.

The **repaint( )** method has four forms. Let's look at each one, in turn. The simplest version of **repaint( )** is shown here:

void repaint( );

This version causes the entire window to be repainted. The following version specifies a region that will be repainted:

void repaint(int left, int top, int width, int height);

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

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

public class Test extends Applet
{
        int initcounter=0;
        int startcounter=0;
        int stopcounter=0;
        int destroycounter=0;

        public void init()
        {
                initcounter++;
                repaint ();
        }
        public void start()
        {
                startcounter++;
                repaint ();
        }
        public void stop()
        {
                stopcounter++;
                repaint ();
        }
        public void destroy()
        {
                destroycounter++;
                repaint ();
        }

        public void paint (Graphics g)
        {
                g.drawString ("init has been invoked " +
                String.valueOf(initcounter) +"times",20,20);
```

```
            g.drawString ("start has been invoked " +
            String.valueOf(startcounter) +"times",20,35);

            g.drawString ("stop has been invoked " +
            String.valueOf(stopcounter) +"times",20,50);

            g.drawString ("destroy has been invoked " +
            String.valueOf(destroycounter) +"times",20,65);
        }
}
```

Self Assessment Questions

5. The Graphics class is a part of _____ package.
6. The _____ method draws the graphics of the applet in the drawing area.

8.8 User Interfaces for Applet

The **Abstract Windowing Toolkit**, also called as AWT is a set of classes, enabling the user to create a user friendly, **Graphical User Interface (GUI).** It will also facilitate receiving user input from the mouse and keyboard. The AWT classes are part of the java.awt package. The user interface consists of the following three:

- **Components** – Anything that can be put on the user interface. This includes buttons, check boxes, pop-up menus, text fields, etc.
- **Containers** – This is a component that can contain other components.
- **Layout Manager** – These define how the components will be arranged in a container.

The statement ***import java.awt.*;*** imports all the components, containers and layout managers necessary for designing the user interface.

The AWT supplies the following components.

- **Labels (java.awt.Label)**
- **Buttons (java.awt.Button)**
- **Checkboxes (java.awt.Checkbox)**
- **Single- line text field (java.awt.TextField)**
- **Larger text display and editing areas (java.awt.TextArea)**
- **Pop-up lists of choices (java.awt.Choice)**
- **Lists (java.awt.List)**
- **Sliders and scrollbars (java.awt.Scrollbar)**

- **Drawing areas (java.awt.Canvas)**
- **Menus (java.awt.Menu, java.awt.MenuItem, java.awt.CheckboxMenuItem)**
- **Containers (java.awt.Panel, java.awt.Window and its subclasses)**

8.9 Adding Components to user interface

Because all applets are containers, components can be added directly to the applet windows. The following steps add a component:

- Create the component by creating an object of that class.
- Call the Container's adds method to add the component.

Labels

Labels display non editable text. To create a label use one of the following:

- **label() –** Creates an empty label.
- **label(String) –** Creates a label with the given string.
- **label(String, int) –** Creates a label with the given string and right, left or center alignment.

Buttons

Clickable buttons can be created from the Button class. You can create a button by using either of the following:

- **Button() –** Creates a button without any label. Use ***setLabel(String)*** to display a text on the button.
- **Button(String) –** Creates a button with the given string as the label.

Checkboxes

They are labeled or unlabeled boxes that can be either checked off or empty. They are used for selecting some option. Use one of the following to create them.

- **Checkbox() –** Creates a checkbox without any label.
- **Checkbox(String) –** Creates a labeled checkbox.

When many checkboxes are there any number of them can be checked or unchecked. Checkboxes can be organized into groups so that only one of them can be checked at a time. The following statements a checkbox group:

CheckboxGroup mygroup = new CheckboxGroup();

To create a checkbox, belonging to this group, use the following statement.

Checkbox c1 = new Checkbox ("Manipal", mygroup, true);

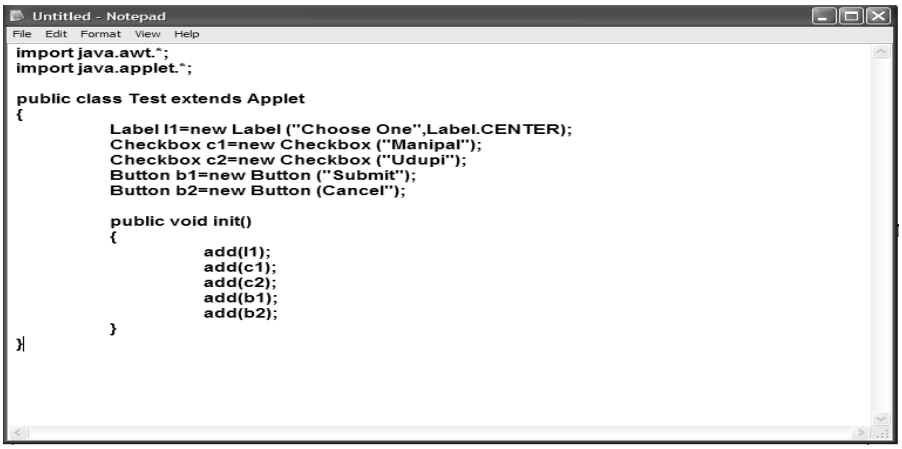

Figure 8.4: Program that creates label, checkbox and buttons

TextFields

TextField is an editable component. Text field can be created by using one of the following:

- **TextField() –** Creates an empty text field.
- **TextField(String) –** Creates a text field with the specified string.
- **TextField(String, int) –** Creates a text field with the specified String and specified width.

The TextField class has several useful methods:

- The ***getText()*** method returns the text in the field.
- The ***setText(String)*** method fills the field with the string.
- The ***setEditable(boolean)*** methods decides whether the filed should be editable or not depending upon the Boolean value.
- The ***isEditable()*** method returns a Boolean value indicating whether the field is editable or not.

Text Areas

These are editable text fields having more than one line of input. Use one of the following to create a text area.

- **TextArea()** – Creates an empty text area.
- **TextArea(rows, character)** – Creates a text area of rows specified and width to accommodate the character specified.
- **TextArea(String)**
- **TextArea(String, rows, character)**

The methods available in case of Text field can also be used here. In addition to that the following methods could be used:

- The ***insert(String, charindex)*** method inserts the indicated string at the position indicated by the second argument.
- The ***replace(string, startpos, endpos)*** method replaces the text between the given integer positions with the indicated string.

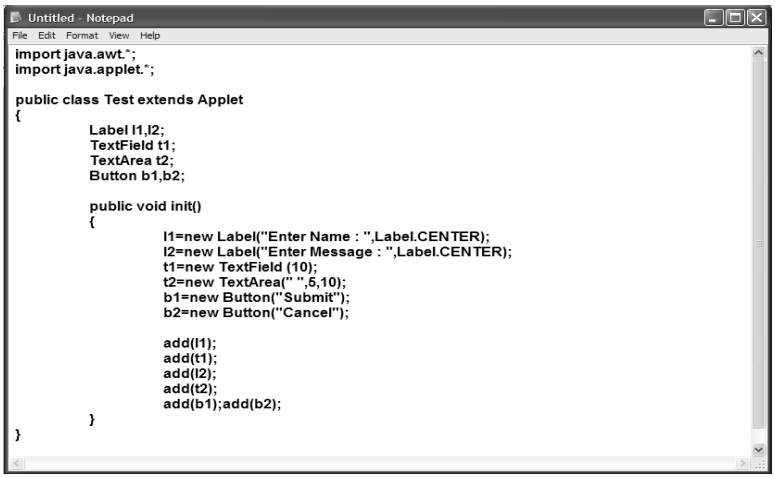

Figure 8.5: Example showing the use of TextField and TextArea.

8.10 AWT (Abstract Windowing Toolkit) Controls

Following AWT (Abstract Windowing Toolkit) controls are available in Java.

Table 8.1: Various AWT Controls

GUI Category	Control	AWT Class Name
Basic Controls	Button Combo box List Menu Slider Toolbar Text Field	Button ComboBox List Menu, MenuBar, MenuItem Slider Toolbar TextField, PasswordField, TextArea
Uneditable Display	Label Tooltip Progress bar	Label ToolTip ProgressBar

Editable Display	Table Text Tree Color chooser File chooser	Table TextPane, TextArea Tree ColorChooser FileChooser
Space-Saving Containers	Scroll pane Split pane Tabbed pane	ScrollPane, ScrollBar SplitPane TabbedPane
Top-Level Containers	Frame Applet Dialog	Frame Applet Dialog

Self Assessment Questions

7. AWT stands for _____.
8. _____ defines how the components will be arranged in a container.

8.11 Summary

- The Applet class is the only class of the ***java.applet*** package.
- An applet runs in a ***Webpage***.
- An applet can be executed using the ***appletviewer*** or a Java-enabled browser.
- The Applet tag is used to embed an applet in a Web page.
- The ***init()*** method is called the first time the applet is loaded into the memory of a computer.
- The ***start()*** method is called immediately after the ***init()*** method and every time the applet receives focus as a result of scrolling in the active window.
- The ***stop()*** method is called every time the user moves on to another web page.
- The ***destroy()*** method is called just before the browser is shut down.
- You can use the method ***getGraphics()*** to obtain an object of the Graphics class.
- The ***update()*** method clears the screen and calls the ***paint()*** method.
- The ***paint()*** method draws the graphics of the applet in the drawing area.
- The ***repaint()*** method calls the ***update()*** method to signal that the applet's redrawn in the drawing area.

8.12 Terminal Questions

1. What are the restrictions on an applet?
2. What is the life cycle of an applet class?
3. What is AWT?
4. What are the components available in AWT?

8.13 Answers

Self Assessment Questions

1. applets.
2. appletviewer.
3. body
4. PARAM
5. java.awt
6. paint()
7. Abstract Windowing Toolkit
8. Layout Manager

Terminal Questions

1. Since applets are run on different machines, security becomes a critical issue. Another probable disadvantage of applets would be the slight flickering that would be inevitable in animations. (Refer Section 8.1)
2. The life cycle of an applet can be described through four methods viz. init(), start(), stop() and destroy(). (Refer Section 8.5)
3. AWT (Abstract Windowing Toolkit) is a set of classes, enabling the user to create a user friendly, Graphical User Interface. (Refer Section 8.8)
4. Label, Button, Checkbox, TextField, Choice, List etc. are various components available in AWT. (Refer Section 8.8)