Ch – 4
Applets & Layout Manager

Basic Introduction

- Applet is a Java program that is embedded in an HTML document and runs in the context of a Java-capable browser.

- The Applet class is contained in the java.applet package. Applet contains several methods that give you detailed control over the execution of your applet.

Methods of Applet Class

- ❖ **1) void init() :** This method is called when an applet is initialized. Its first for any applet.

- ❖ **2) void start() :** This method is called when starts or resume its execution.

- ❖ **3) void stop() :** This method is called to suspend the execution of the applet.

- ❖ **4) void destroy() :** This method is called by the browser just before an applet terminates its execution.

Methods of Applet Class

- ❖ **5) AppletContext getAppletContext() :** This method is returns an AppletContext object that is associated with the applet.

- ❖ **6) string getAppletInfo() :** It returns a string containing the Applet information.

- ❖ **7) URL getCodeBase() :** It returns the URL os the applet class file.

- ❖ **8) void showStatus() :** It display the status in the status bar of the browser or Applet Viewer.

Simple Applet Program

```
import java.applet.Applet;
import java.awt.Graphics;
/*<applet code="FirstApplet" width=300
   height=50>
</applet>
public class FirstApplet extends Applet
{
   public void paint(Graphics g)
   { g.drawString("This is my first applet.",20,100);
   }
}
```

Compile : javac FirstApplet.java
Run : Appletviewer FirstApplet.java

Applet Life Cycle

- ❖Each applet has a life cycle. So every time an applet is initialized it goes through four states and finally after its execution the applet is destroyed.

- ❖This methods are described below :

- ❖**1) The init() Method :** The init() method is first method called by applet. In this method the initialization process is done. The variable, fonts, colors etc. are initialized in this method.

  - ❖**Syntax : public void init()**

    { // initialization process….}

Applet Life Cycle

- **2) The start() Method :** The start() method is start the applet. It is also called to restart the applet if it is stopped.

  - **Syntax : public void start()**

    { //code to start the applet….}

- **3) The paint() Method :** The paint() method is component class which is in the java.awt.package. This method is called when the applet execution starts or ontents of applet redraw.
  - **Syntax : public void paint(Graphics g)**

    { //code to display an applet…. }

Applet Life Cycle

- ❖ **4) The stop() Method :** The stop() method is called when you minimize the applet window or leave the applet page and go to another page in the browser.

  - ❖ **Syntax : public void stop()**

    { // statements….}

- ❖ **5) The destroy() Method :** The destroy() method is called when your applet is to be yterminated and nedds to remove from the memory.

  - ❖ **Syntax : public void destroy()**

    { //code for finalization…. }

Example of Applet Life Cycle Methods

```
import java.applet.Applet;
import java.awt.Graphics;
/*<applet code="AppletLifeCycle" width=300
   height=50>
</applet>
*/ public class AppletLifeCycle extends Applet
{
   String str="";
   public void init()
   { str+="init";
   }
   public void start()
   {  str+="start“ }
   public void stop()
   {  str+="stop"; }
   public void destroy()
   {  System.out.println("destroy");
public void paint(Graphics g)
   {
        g.drawString(str,10,25);
   } }
```

The Graphics class

- The Graphics class in the java.awt package and it contains serveral methods that can help to perform graphics related operations in the applet.

- This methods include drawing, text, lines and many different shapes. **This methods are :**

  - **1) Void drawString(String str, int x, int y) :** Draws str at location of x and y Cordinates.

  - **2) Void drawLine(int x0,int y0,int x1, int y1) :** Draws a line between the points at x0,y0 and x1,y1.

The Graphics class

- **3) Void drawRect(int x, int y, int w, int h) :** Draws a rectangle with upper-left corner at coordinates x and y width w and height h.

- **4) Void fillRect(int x, int y, int w, int h) :** Fills a rectangle with upper-left corner at coordinates x and y, width w and height h.

- **5) Void drawOval(int x, int y, int w, int h) :** Draws an oval.

- **6) Void fillOval(int x, int y, int w, int h) :** Fills an oval.

The Graphics class

- **7) Void drawArc(int x, int y, int w, int h, int degree0, int degrees1) :** Draws an arc between degrees0 and degrees1.

- **8) Void fillArc(int x, int y, int w, int h, int degrees0, int degrees1) :** Fills an arc between degrees0 and degrees1.

- **9) Void drawPolyline(int x[], int y[], int n) :** Draws a polyline with n points.

- **10) Void fillPolygon(int x[], int y[], int n) :** Fills a polygon with n corners. The coordinates are given by the elements of x and y.

Example of Graphics class methods

```
import java.applet.Applet;
import java.awt.Graphics;
/* <applet code="DrawShapes" width=400
   height=400>
</applet> */
public class DrawShapes extends Applet
{  public void paint(Graphics g)
   {    g.drawArc(20,20,160,160,0,135);
        g.drawOval(50,50,60,30);
        g.drawString("All shapes",10,15);
        g.fillRect(100,100,70,40); }  }
```

The Color class

- The Color class in the java.awt package and to deal with colors.

- The color class lets you make any color you want. **Its Constructor are :**

  - **1) Color(float red, float green, float blue):** These values should be between 0 and 255.
  - **for example:**
    - **Color c=new Color(0,255,0); // green color**

  - **2) Color(int RGB value):** Here RGB is mix of red, green, blue.

The Color class

- **Some methods defined by the color class:**
  - **1) int getRed():** It returns the value of red component of color object.
  - **2) int getGreen():** It returns the value of green component of color object.
  - **3) int getBlue():** It returns the value of blue component of color object.
  - **4) int getRGB():** It returns an int number that represent the current colors.

Example of Color Class

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;
/*<applet code="BlueString" width=300
   height=100>
</applet>*/
public class BlueString extends Applet
{  public void paint(Graphics g)
   {    g.setColor(Color.blue);
        g.drawString("Blue string",100,50);
   }  }
```

The Font class

- The font class allows you to specify the size and type of the font displayed in the applet window. The font class Is in java.awt package.

- **Its Constructor are :**

  - **Font(String name, int style, int size):** Specify the name of font. E.g. Arial, Verdana etc.

  - After creating a font object you have to set it by the setFont() method.

    - **Syntax : void setFont(Font obj)**

Example of Font class

```
import java.applet.Applet;  import java.awt.*;
/* <applet code="FontDemo" width=200 height=200>
</applet>*/
public class FontDemo extends Applet
{ public void paint(Graphics g)
   {    int baseline=100;
        g.setColor(Color.blue);
        g.drawLine(0,baseline,200,baseline);
        g.setFont(new Font("serif",Font.BOLD,36));
        g.setColor(Color.black);
        g.drawString("hello",5,baseline);  }
```

The Dimension class

- **The Dimension class allows us to get the dimensions of the applet window. If has method getSize() that is used to obtain the current size of applet window.**

- **Example :**

```
import java.applet.Applet;
import java.awt.*;
/*<applet code="DimensionDemo" width=200
   height=200>
</applet> */
public class DimensionDemo extends Applet
{
```

The Dimension class

```
    public void paint(Graphics g)
    {
        Dimension d=getSize();
        int x=d.width/2;
        int y=d.height/2;
        int radius=(int) ((d.width<d.height) ?
        0.4*d.width:0.4*d.height);

        g.drawOval(x-radius, y-radius,2*radius,2*radius);
    }
}
```

The FontMetrics class

- **The FontMetrics class is used to get various mesures of the font. All the fonts do not have the same dimensions. Thus we manage the text output. The following terms are used to describe fonts :**

  - **1) Baseline :** It is the line to which the bottoms of characters are aligned to.

  - **2) Ascent :** It is the distance from the baseline to the top of a character.

  - **3) Descent** : It is the distance from the baseline to the buttom of a character.

The FontMetrics class

- ❖**Some methods defined by the FontMetrics class are :**
- ❖**1) Int getAscent() :** Returns the ascent.
- ❖**2) Int getDescent() :** Returns the descent.
- ❖**3) Int getHeight() :** Returns the height
- ❖**4) Int getLeading() :** Returns the leading
- ❖**5) Int charWidth(char c) :** Returns the width of c.

Example of FontMatrics class

```
import java.applet.Applet;  import java.awt.*;
/* <applet code="FontMetricsDemo" width=200
   height=200>
</applet>*/
public class FontMetricsDemo extends Applet
{  public void paint(Graphics g)
   {    int baseline=100;
        g.setColor(Color.blue);
        g.drawLine(0,baseline,300,baseline);
        Font font=new Font("serif",Font.BOLD,36);
        g.setFont(font);
        g.setColor(Color.black);
```

Example of FontMatrics class

```
        g.drawString("hello",5,baseline);
        g.setColor(Color.blue);
        //get fontMetrics
        FontMetrics fm=g.getFontMetrics(font);
        //draw line at baseline-ascent
        int ascent=fm.getAscent();
        int y=baseline-ascent;
        g.drawLine(0,y,300,y);
    }
}
```

AppletContext Interface

❖ The AppletContext interface is used to get information of other documents from an applet.

❖ To get an object of AppletContext, **getAppletContext()** method is used.The followiong method of AppletContext interface is used to get information of another file :

❖ **1) void showDocument(URL path) :** This method display the file which is located at the specified path.

❖ **2) void showDocument(URL path, String where)**

The Applet tag of HTML

- As we saw that the applet tag is used to start an apple from both appletviewer and web browser.

- The general form of a standard applet is show below. The attributes in square brackets are optional attributes.

```
<applet
   [codebase=url]
   Code=clsName
   [alt=text]
   [name=appName]
   Width=wpixels
```

The Applet tag of HTML

```
    Height=hpixels
    [align=alignment]
    [vspace=vspixels]
    [hspace=hspixels] >
    [<param name=pname1 value=value1>]
    [<param name=pname2 value=value2>]
    …
    [<param name=pnameN value=valueN>]
</applet>
```

The Applet tag of HTML

- ❖**Each attributes are explained here :**

- ❖**1) Codebase :** This attribute specify the URL of the directory which contains the class defination of your applet.(Optional attribute)

- ❖**2) Code :** In this attribute specify the name of the class file of your applet program. (Compulsory)

- ❖**3) Alt :** It is used to display an alternate text in the browser. (Optional)

The Applet tag of HTML

- **4) Name :** It specify the name of the applet . (Optional)

- 5) Width and Height : Width and Height are compulsory attributes. They specify the width and height of applet window in pixels.

- 6) Align : It specify the Alignment of the applet. The alignment value can be LEFT,RIGHT,TOP, BOTTOM, MIDDLE etc. (Optional)

The Applet tag of HTML

- **7) VSPACE and HSPACE :** This attribute specify the Vertical and Horizontal spaces of the applet. (Optional)

- **8) Param name and Value:** It allows us to pass parameters to an applet. The name specify the name of the parameter and value specify its value. (Optional)

```
<img src=”photo1.jpg” width=200 height=200>
<br> <applet code=”example1” codebase=”host2”
  width=300 height=300>
</applet>
import java.applet.*;
import java.awt.*;
public class ParamTest extends Applet {
  public void paint(Graphics g) {
    String myFont   = getParameter("font");
    String myString = getParameter("string");
int mySize  =
   Integer.parseInt(getParameter("size"));
      Font f = new Font(myFont, Font.BOLD,
        mySize);
       g.setFont(f);
      g.setColor(Color.red);
      g.drawString(myString, 20, 20);
   }
}
```

```
<HTML>
<HEAD> <TITLE>Passing params </TITLE></HEAD>
<BODY>
<APPLET CODE="ParamTest.class" WIDTH="400"
   HEIGHT="50">
    <PARAM NAME="font"    VALUE="Chiller">
    <PARAM NAME="size"    VALUE="24">
    <PARAM NAME="string"  VALUE="Hello, world ... it's
   me.   :)">
</APPLET>
</BODY>
</HTML>
```

15) Layout Managers

- The layout manager decided how the control should be arranged or positioned within a container or a window.

- **1) FlowLayout :** The FlowLayout is the default layout manager i.e. if no layout manager is set by the setLayout() mehtod , this layout is used to arrange the controls.
- **To set Flow layout following constants :**
  - FlowLayout.CENTER
  - FlowLayout.LEFT
  - FlowLayout.RIGHT

- **2) BorderLayout :** The BorderLayout arranges the components to five position. These positions are known as north, south, east, west and center.

- **Following constant:**
  - BorderLayout.NORTH, BorderLayout.SOUTH, BorderLayout.EAST, BorderLayout.WEST, BorderLayout.CENTER.

- **The constructors for BorderLayout are:**
  - BorderLayout()
  - BorderLayout(int hspace, int vspace)

- **3) CardLayout :** The CardLayout has some special capabilities the other layout do not have. The CardLayout creates a layout like the playing cards.

- **The constructors for CardLayout are:**
  - CardLayout()
  - CardLayout(int hspace, int vspace)

- **4) GridLayout :** The GridLayout class creates a layout which has a grid of rows and columns.
- **The constructors for GridLayout are:**
  - GridLayout()
  - GridLayout(int row, int columns)

- **5) BoxLayout:** The BoxLayout is a general purpose layout manager which is an extensible FlowLayout.
- **The constructors for BoxLayout are:**
  - BoxLayout.PAGE.AXIS
  - BorderLayout.LINE.AXIS

- **5) SpringLayout :** The SpringLayout is a very flexible layout than has many features for specifying the size of the components.

- **The constructors for SpringLayout are:**
  - SpringLayout()

- **Its Method :**

  - Void putConstraint(string edge1, Componet obj1, int distance, String edge2, Compontnt obj2)