Unit III Part A

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.

Advantage of Applet

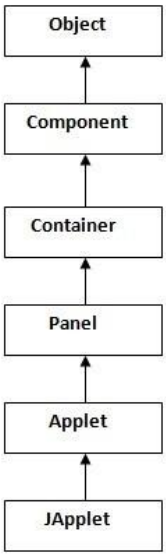
There are many advantages of applet. They are as follows:

- 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.

Hierarchy of Applet



As displayed in the above diagram, Applet class extends Panel. Panel class extends Container which is the subclass of Component.

Lifecycle of Java Applet

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.

java.applet.Applet class

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

1. **public void init():** is used to initialized the Applet. It is invoked only once.
2. **public void start():** is invoked after the init() method or browser is maximized. It is used to start the Applet.
3. **public void stop():** is used to stop the Applet. It is invoked when Applet is stop or browser is minimized.
4. **public void destroy():** is used to destroy the Applet. It is invoked only once.

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.

//AppletExample1.java

```
import java.applet.Applet;
import java.awt.Graphics;
public class AppletExample1 extends Applet
{
        public void paint(Graphics g)
        {
                g.drawString("welcome",150,150);
        }
}
```

Save as myapplet.html

```
<html>
        <body>
                <applet code="AppletExample1.class" width="300" height="300">
                </applet>
        </body>
</html>
```

Combining both file to execute in appletviewer

```
import java.applet.Applet;
import java.awt.Graphics;
public class AppletExample1 extends Applet
{
        public void paint(Graphics g)
        {
                g.drawString("welcome",150,150);
        }
}
/*
<applet code="AppletExample1.class" width="300" height="300">
</applet>
*/
```

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

```
import java.applet.Applet;
import java.awt.*;
public class GraphicsDemo 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);
```

```
    }
}
```

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

Displaying Image in Applet

Applet is mostly used in games and animation. For this purpose image is required to be displayed. The java.awt.Graphics class provide a method drawImage() to display the image.

Syntax of drawImage() method:

1. **public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer):** is used draw the specified image.

How to get the object of Image:

The java.applet.Applet class provides getImage() method that returns the object of Image.

Other required methods of Applet class to display image:

1. **public URL getDocumentBase():** is used to return the URL of the document in which applet is embedded.
2. **public URL getCodeBase():** is used to return the base URL.

In most of the applets, it is required to load text and images explicitly. Java enables loading data from two directories. The first one is the directory which contains the HTML file that started the applet (known as the **document base).** The other one is the directory from which the class file of the applet is loaded (known as the **code base).** These directories can be obtained as URL objects by using getDocumentBase ()and getCodeBase ()methods respectively. You can concatenate these URL objects with the string representing the name of the file that is to be loaded.

```
import java.applet.Applet;
import java.awt.Graphics;
import java.net.URL;
import java.awt.*;
import java.awt.event.*;
public class GetDocumentBaseExample extends Applet
{
        public void paint(Graphics g)
        {
                String message;
                //getCodeBase() method gets the base URL of the directory in which contains
this applet.
                URL appletCodeDir=getCodeBase();
```

```
                message = "Code Base : "+appletCodeDir.toString();
                g.drawString(message,10,90);

               // getDocumentBase() Returns an absolute URL of the Document
               URL appletDocDir = getDocumentBase();
               message="Document Base : "+appletDocDir.toString();
               g.drawString(message,10,120);
        }
}
/*
   <applet code="GetDocumentBaseExample" width=350 height=250>
   </applet>
*/
```

Example

```
import java.awt.*;
import java.applet.*;
public class DisplayImage extends Applet
{
   Image picture;
   public void init()
        {
                picture = getImage(getDocumentBase(),"Desert.jpg");
   }
   public void paint(Graphics g)
        {
                g.drawImage(picture, 30,30, this);
   }
}

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

In the above example, drawImage() method of Graphics class is used to display the image. The 4th argument of drawImage() method of is ImageObserver object. The Component class implements ImageObserver interface. So current class object would also be treated as ImageObserver because Applet class indirectly extends the Component class.

Animation in Applet

Applet is mostly used in games and animation. For this purpose image is required to be moved.

Example of animation in applet:

```
import java.awt.*;
import java.applet.*;
public class AnimationExample extends Applet
{
   Image picture;
   public void init()
        {
      picture =getImage(getDocumentBase(),"bike.jpg");
   }
   public void paint(Graphics g)
        {
      for(int i=0;i<500;i++)
                {
                        g.drawImage(picture, i,30, this);
        try
                        {
                                Thread.sleep(50);
                        }
                        catch(Exception e)
                        {
                        }
      }
   }
}
/*
<applet code="AnimationExample.class" width="300" height="300">
</applet>
*/
```

repaint() in Java Applet Example

The paint () method is called automatically by the environment (usually a web browser) that contains the applet whenever the applet window needs to be redrawn. This happens when the component is first displayed, but it can happen again if the user minimizes the window that displays the component and then restores it or if the user moves another window over it and then move that window out of the way. In addition to these implicit calls to the paint() method by the environment, one can also call the paint () method explicitly whenever the applet window needs to be redrawn, using the repaint () method.

The repaint () method causes the AWT runtime system to execute the update () method of the Component class which clears the window with the background color of the applet and then calls the paint () method. For example: Suppose you want to display the current x and y coordinates of the location where the mouse button is clicked in the applet window. As the applet needs to update information displayed in its window (i.e. redraw the window), each time the mouse is being clicked so this is possible with the use of repaint () method. To sum up, the repaint() method is invoked to refresh the viewing area i.e. you call it when you have new things to display.

```
import java.awt.*;
import java.applet.Applet;
import java.awt.event.*;
/*<applet code="RepaintJavaExample.class" width="350" height="150"> </applet>*/
public class RepaintJavaExample extends Applet implements MouseListener
{
   private int mouseX, mouseY;
   private boolean mouseclicked = false;
   public void init()
   {
      setBackground(Color.CYAN);
      addMouseListener(this);
   }
   public void mouseClicked(MouseEvent e)
   {
      mouseX = e.getX();
      mouseY=e.getY();
      mouseclicked = true;
      repaint();
   }
   public void mouseEntered(MouseEvent e){};
   public void mousePressed(MouseEvent e){};
   public void mouseReleased(MouseEvent e){};
```

```
    public void mouseExited(MouseEvent e){};
    public void paint( Graphics g )
    {
        String str;
        g.setColor(Color.RED);
        if (mouseclicked)
        {
            str = "X="+ mouseX + "," + "Y="+ mouseY ;
            g.drawString(str,mouseX,mouseY);
            mouseclicked = false;
        }
    }
}
```

Use of update() method. When is it invoked?

Overriding update( )

In some situations, applet may need to override another method defined by the AWT, called update( ).This method is called when your applet has requested that a portion of its window be redrawn. The default version of update( ) first fills an applet with the default background color and then calls paint( ). If you fill the background using a different color in paint( ), the user will experience a flash of the default background each time update( ) is called—that is, whenever the window is repainted.

One way to avoid this problem is to override the update( ) method so that it performs all necessary display activities. Then have paint( ) simply call update( ). Thus, for some applications, the applet skeleton will override paint( ) and update( ), as shown here:

```
public void update(Graphics g) {
// redisplay your window, here.
}

public void paint(Graphics g) {
update(g);
}
```

- An update() method is called on calling the repaint method.
- The default implementation of the update() method clears the screen and calls the paint() method.
- The graphics instance is valid only within the context of the update method() returns.
- This method is called in response to repaint() request.
- The default implementation is provided by the component class which erases the background and calls the paint() method.