Chapter 7

Applet and Graphics Programming

Dulal Acharjee

Before starting concepts of Applet, we shall describe some tools of graphics programming. Now-a-days, most of the software, apps, interfaces are designed with user to machine interfacing tools. These interfacing tools are, icon, pointer, fonts with different sizes, pictures, boxes, lines, drawing tools as brush, sprays, color plates, drop down box, check box etc. all are used to make the comfort of using computer. Many technologies and programming arts are related to these tools. Gradually, some theoretical and practical concepts will be discussed in this chapter.

Applet is a separate programming environment especially used for designing graphic features. Every Java applet program should start with a subclass of the `Applet` or `JApplet` class. *Applet* class contains all methods to develop graphics program, it has a hierarchy like:-

```
java.applet.Applet
```

Another version of it is *JApplet*, which is a class of *swing* package and its hierarchy is as:

```
javax.swing.Japplet
```

To work with applet, a subclass must be created from *Applet* or *JApplet*. Probably, upto jdk11 applet is available, after that, applet tool is not kept within Java. *Appletviewer* is an application residing within \bin\ folder of JAVA and translates all graphical statements of an applet application. Applet application can’t be run by *java* interpreter/JVM but through the embedded code within HTML file.

Applet is a simple java file without *main()* module. It should be compiled as ‘class’ file and invoked from a HTML file as shown below the program file TestString.java.

Applet program does not contain *main()* module within the applet class. It is used by an <applet> tag within the HTML file. *Appletviewe*r is an application which can interpret/execute a ***.html*** file having embedded applet code.

Basics of HTML programming:

Full name of HTML is Hiper Text Markup Language. What is Hyper Text? In any website if you click a link it display a new page or some other documents; that means that link is connected with some other webpage; the portion of text which is clicked is known as Hyper Text. We are clicking these types of Hyper Texts every day in mobile phone or in computer.

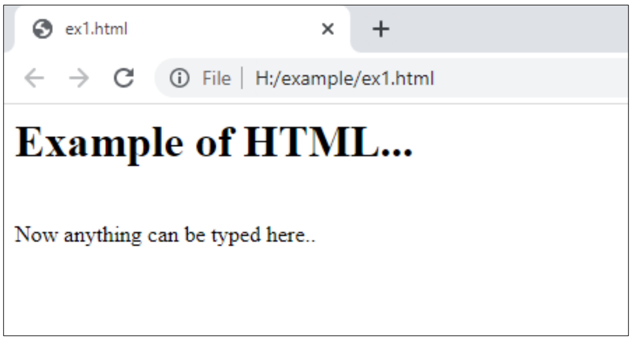
HTML is a scripting language used to design a webpage. Any html file or program can be run by any browser. If you open the xyz.html file within any of browser like: Google Chrome, explorer of Microsoft, Edge and Firefox etc. the file will be executed or interpreted by the browser instantly. The .html file is not required to compile like programs of C/C++/JAVA, it is only interpreted line by line and does not give any error messages, only the erroneous lines are bypassed. In any text editor .html file can be created.

```
//ex1.html: an example of html program
<html>
<body>

<h1> Example of HTML...</h1>
<br> Now anything can be typed here..</br>

</body>
</html>
```

Output:



```
Example of HTML...

Now anything can be typed here..
```

Fig. 7.1. output of a html program to show how to print heading of a webpage.

Discussion: there are different ways to run a html file. Easiest way is to select the file> right chick on mouse>open with>Google Chrome(or other browser).

Program will run with results/outputs as shown above. In above example, only 6 lines are there and meanings of these lines are:

<html>…</html> : program starts from html Tag and closes by the same Tag as right slash symbol '/'.

<body>…</body> : anything to be displayed to be written within this area.

<h1>…</h1> : large font based title of the page; there are h2, h3,… different subtitle Tags. These are the basics of learning webpage designing.

Below, another html program is shown with Hyper Text markings and how to link with any picture. It is to be noted that by default it will search the picture within the default folder i.e. from where the .html program is opened. Else, path of the picture should be mentioned with the name of the picture file.

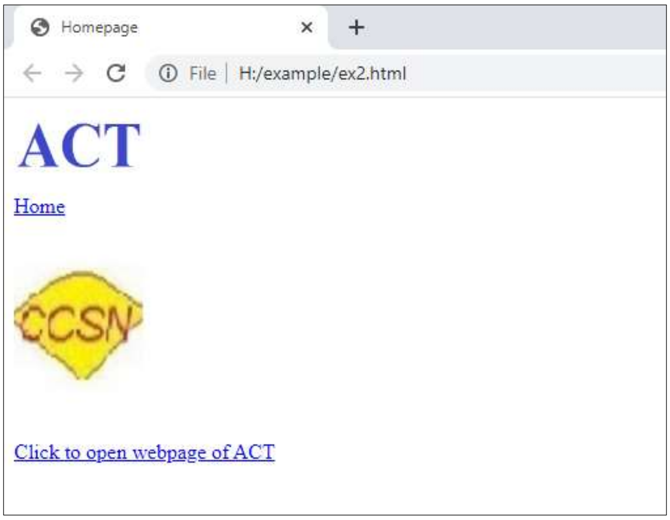
//**ex2.html**: shows how to open picture and to link a hyper text.

```
<html>
<title>Homepage</title>

<body>

<img src="/photo/ACT-logo.png" height=50 width=100 ></img>
<p> <a href="index.html" >Home</a></p>
<br> <img src="/photo/ccsn-logo.png" height=100 width=100 ></img></br>
<br> <br><a href="https://actsoft.org">Click to open webpage of ACT</a>
</body>
</html>
```

Output:



Home

Click to open webpage of ACT

Fig. 7.2. output of a html program to show how to display any picture and how to make hyper test to link of a URL(Unified Resource Locator).

Discussion: in this example first is <title> as ‘Homepage’ is displayed at top of left side of the window, then, within the <body> Tag we have written what to be displayed within the webpage. First, we have opened a picture file ‘ACT-logo.png’, mentioned height and width of the area where the picture will be displayed. There are two Hyper Texts, first is “Home” and the second is “Click to open webpage of ACT”. There are created by <a>…</a> Tag as:
<a href="index.html" >Home</a>

As this book is not for HTML programming, so, we shall not give more examples of this programing/scripting language. Only a simple knowledge is given, as because, it will be required to embed applet program within html program. How to write and run .html file- to demonstrate that concept, here, we have given these two examples.

Steps to execute an applet based program:

a. Write an applet program and compile as a ***.class*** file.
b. Write a ***.html*** program having <applet> tag and call the *.class* file.
c. Execute the *.html* script file by '*appletviewer*'

(a) Below, a class file '*TestString.java*' is created; there is no *main()* module; '*TestString*' subclass is created from the class '*Applet*'.

```
// TestString.java: this class will be read from a .html file
// this java program can't execute by 'java' as there is no 'main()'
//method in this program.

import java.awt.*;
import java.applet.*;
public class TestString extends Applet
{
        String st1, st2;
        public void init()
        {
            st1="INDIA";
            st2="I am from :" + st1;
        }
        public void paint(Graphics g)
        {
            g.drawString("Halo ! " + st2, 10, 100);
        }
}
```

Discussions: an user defined method '*paint()*' is created in this program and it has an argument of a class *Graphics* ; at the same time an object '*g*' is created from the class *Graphics;* so, it is better to say that argument of the method 'paint()' is an object '*g*'. The class 'Grahics' is an component of the package '*awt*', so, we have imported the package as: *import java.awt.*;* at the beginning of the program.

(b) Now, *.html* file will be created. What is a scripting languate? *.html* is the Hiper Text Markup Language(HTML) program which is used for writing web pages. On any link if we put the cursor- it shows a link indicating a hand/pointer symbol meaning to mark the text with a link- so its name is Markup Language. It was the basic and first webpage designing language. After that, many other scripting languages have been developed as: javaScript, xml, PHP, typeScript, CSS, Ruby etc. Files written as *.html* are executed with the help of any web browser as: internet explorer, Google Chrome, Microsoft Edge, Mozilla Firefox etc. The .html files are written under any text editor and are given the extension name as: *xyz.html*. To run any .html program, only double click to be given by mouse over the filename- it will run under the default browser. (For understanding .html program- how to develop a web page- many examples are written in Appendix section of this book). An example of writing a *.html* file is shown below:

```
//tes2.html: Applet class is used by an <applet> tag within  the HTML file.
<html>
<body>
    <applet code= TestString.class width= 400 height=300> </applet>
</body>
</html>
```

Discussions: a html is a tag based program, all instructions are coded under a < > tag as:

<html> ….</html> : starting and ending of a *.html* file.
<body>….</body> : any text or other works will be written within this section.
<applet> ….</applet> : to invoke applet code or *.class* file which is written following applet rules.

Here, the meaning of the line shown below is that it will invoke/call the '*TestString.class*' file from the default folder; we have mentioned the window size width=400 and height=300 pixels.

```
<applet code= TestString.class width= 400 height=300>
```

Any text or graphics code written within the *.class* file will be displayed within this new window area as mentioned by width and height. Within the graphics area '*.println(), printf (), print()* etc.' methods will not work for displaying any text. For that graphics method 'drawString()' is used for printing any text at a particular (x, y) point. Under graphics, if any text is displayed- that is also considered as graphics elements, texts are nothing but combinations of some dots(pixels) on the screen.

(c) Execute the Applet program: in *jdk* old versions, applet viewer is available, we have used *jdk8* for making demonstrations of '*Applet*' features. For higher versions of *jdk* Applet is not available, but advanced graphics programming features have been included. For all other examples of this book, jdk16 version is used for testing programs. To run the '*test2.html*' file, we have to give command from terminal/command prompt as:

```
E:\example>appletviewer test2.html
```

Problem: write an Applet program to demonstrate mouse events to catch row-column of click point and can draw a line at the next clicked point.

```
//Draw1.java:
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class Draw1 extends JApplet implements MouseListener
{
        private int lastX = 0, lastY = 0;
        public void init()
        {
          this.addMouseListener(this);
        }
        public void mousePressed(MouseEvent  e)
        {
                Graphics g = getGraphics();
                Int  x = e.getX(), y = e.getY();
                g.drawLine(lastX, lastY, x, y);
                lastX = x;
                lastY = y;
        }
}
```

Discussions: here, the class Draw1 is inherited from two classes, JApplet and MouseListener. But, careful observation will detect that *Draw1 implements MouseListener* meaning that *MouseListener* is an interface and Draw1 is also inherited from this class of type interface. Interface class is behaves like a class but it has only the declarations of body of methods without constructions. That means, if methods of interface-class is to be used, that to be done within the main class body. Details of interface have been discussed in the chapter-5 of ‘Object Oriented programming’.

In this example, mousePressed() method is constructed within the Draw1 class. This method is a member of MouseListener class and constructed in this place. Argument of this method is created in this way: MouseEvent e; here argument of the method is the object ‘*e*’ which is created from the class *MouseEvent*. The two methods getX() and getY() read the x-y coordinates of the point where mouse is clicked. These points are required to draw a line on the graphic screen.

```
//Draw1.html
<html>
<h1> Applet Testing</h1>
<applet code=Draw1.class height=300 width=400>
</applet>
</html>
```

Run the program:

```
E:\example>appletviewer Draw1.html
```

Output: drawing within Applet screen.

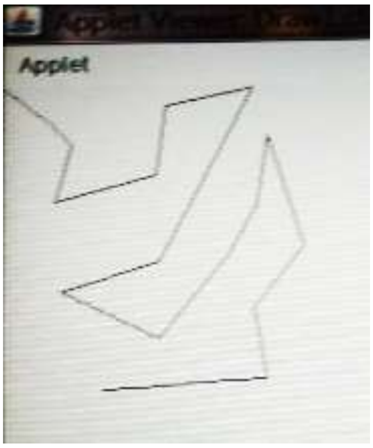


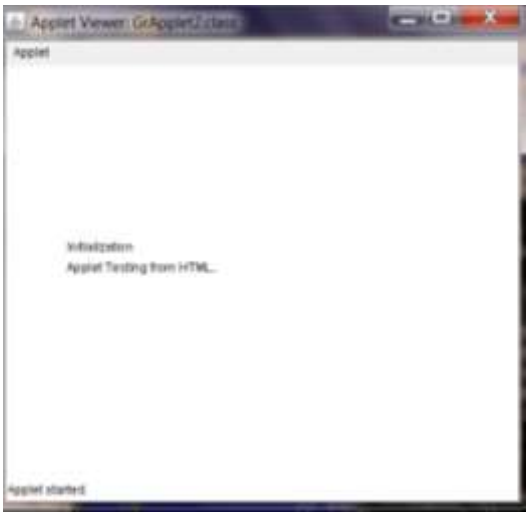
Fig. 7.3. output of a applet program to show how to create a drawing panel.

```
//GrApplet2.java:
import java.awt.*;
import java.applet.*;
public class GrApplet2 extends Applet
{
       String str="", str1="", str2="";
            public void init()
            {
             str1="Initialization";
            }
            public void start()
            {
             str2="Starting....";
            }
            public void stop()
            {
             str="Stop Program now...";
            }
            public void paint(Graphics g)
            {
             // str=str+"Painting....";
            // g.drawString(str,60,120);
             // g.drawString(str2, 60,130);
              g.drawString(str,60,180);
              g.drawString("Applet Testing from HTML..", 80, 150);
            }
}
```

```
//GrApplet2.html:
<html>
<applet code= GrApplet.class width=500 height=400>
</applet></html>
```

Run the programmer:

```
E:\example>appletviewer GrApplet2.html
```



```
Initialization
Applet Testing from HTML..
```

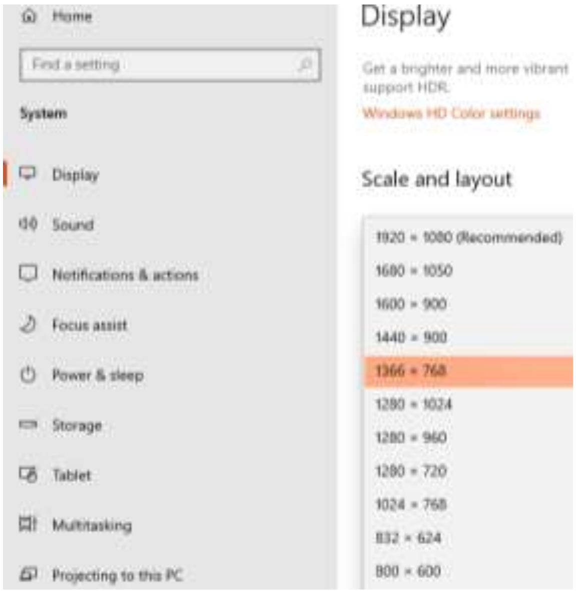
Applet started

Fig. 7.4. output of a applet program to show how to create a window and how to display some texts with that area.

Discussion: In this program we have created a window of size width=500 height=400 and within that graphics area it has drawn the text “initialization” and “Applet Testing from HTML”; but remember that these are drawn as figure not as text. The class “Graphics” has all related methods to operate with graphics mode.

It is a graphics programming; all text, data and figures are displayed under graphics mode. What is Graphics mode? To understand computer display or monitoring technology, first of all it is required to understand what a display device is. It may be a CRT(Cathode Ray Tube) based monitor which display illuminate the cross point of Horizontal and Vertical scanning signals. It creates a dot and many dots create a character.

In LCD(Liquid Crystal Display) there are a grid of horizontal and vertical lines. The base points of these horizontal and vertical points are connected with Graphics processor and circuits. It can illuminate and radiate some light of small area which we call a dot. These dots create a character.
For general programming it is not required to know the hardware attached with the computer, but, for graphics programming, controlling quality of graphics some concept of Graphics Card, Graphics processor, Cash memory of graphics card, speed of graphics card, different mode of graphics supporting by the card should be learnt. Generally, in windows, there are options to setup personal display resolution sizes of display. Some computer can display very fine graphics and some computers can’t produce fine qualities of pictures. The reason is that graphics card (now a days attached with mother board of computer) and supports different graphics resolutions as:



Display

Get a brighter and more vibrant support HDR.

Scale and layout

1920 × 1080 (Recommended)
1680 × 1050
1600 × 900
1440 × 900
1366 × 768
1280 × 1024
1280 × 960
1280 × 720
1024 × 768
832 × 624
800 × 600

Fig. 7.5. graphics resolution selection from window personalization of display.

This display setup is taken from windows 10, it may be different for different versions of windows. The present setup of this computer is shown as 1366 x 768. It is known as resolution of the screen and different resolution supported by this computer are shown by different numbers. Resolution is the unit of display quality and counted by numbers of dots in one square inch area. The numbers 1366 means in one inch length it can produce 1366 numbers of dots and in vertical direction in one inch it can produce 768 dots. So many higher dots, picture quality will be so good.

Generally we work with text processing, but, when windows starts it works with graphics mode, all components of graphics remain active.
Applet works under user defined graphics environments available within the hardware features of the computer.

```
//grApp.java: and grApp.html: this program creates circle, line and rectangle.
// run this program: E:\example>appletviewer grApp.html
import java.awt.*;
import java.io.*;
import java.lang.*;
import javax.swing.*;
import java.applet.*;
public class grApp extends Applet
{
        public void init()
        {
                setBackground(Color.white);
                setForeground(Color.red);
        }
        public void paint(Graphics g)
        {
                g.drawRect(0,0,50,70);//two corners; x1,y1, x2,y2
                g.drawOval(100,150, 40, 120);//x1, y1, width, height
                g.drawLine(0,0,400,70);//x1,y1,x2,y2
                g.drawOval(100,200,50,10);
        }
}
```

To run this program, a HTML program will be required and shown below:

```
//grApp.html : a html file which calls grApp.class file to show graphic drawings.
<html>
<applet code=grApp.class width=500 height=400>
</applet>
</html>
```

To Run the program:

```
E:\example>appletviewer grApp.class
```

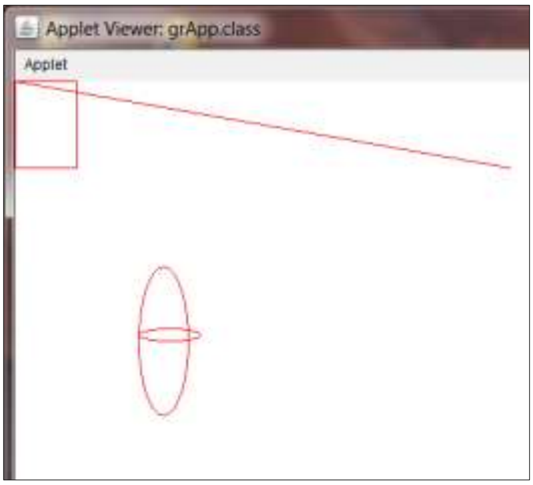


Fig. 7.6. output of an applet program to show how to draw different shapes required for graphics designing.

Discussions: in this example different graphics methods are used like:
g.drawLine(0,0,400,70); here, x1=0, y1=0, x2=400 and y2=70; it starts from zero coordinates, moves towards x direction 400 pixels and then down 70 pixels which is shown as straight line above.
Other two methods are g.drawRect() and g.drawOval() used in this example;
drawRect(0,0,50,70); this will start from 0,0 coordinate where width= 50 pixels and height=70 pixels; but remember that second two points are not coordinates. A rectangle has four right angles, 90^0, at four corners; front sides are same in size. But sizes of sides should be dissimilar. If all sides are similar and angles are 90^0 that will become square. So, in the method of rectangle if we put the last two numbers same as:
drawRect(0,0,50,50); it is a rectangle
or,
drawRect(150,100,70,70); it is square starting point 150,100 where with and heights are same meaning that sizes of two sides are similar.
only the coordinates are required to understand, i.e, where to start drawing and at which point it will end. Coordinate values of end corner can be calculated as example:
x2 = 150+70 and y2 = 100+70; So, we can write in this way also. It is the way to calculate coordinate values of corners.
More concepts of coordinates and drawing fundamentals are discussed below.
An oval shape is drawn within a rectangle area and when an oval is drawn on screen, the equivalent rectangle area is occupied on the screen. Below, an example of drawing an Oval is shown:

```
drawOval(100,150, 40, 120);//x1, y1, width, height
```

Here, 100,150 is the starting coordinates of x, y of the rectangle within which the Oval will be drawn and 40, 120 are width and height respectively of the rectangle; but, noted that last two numbers are not the coordinate of the other cornet.

In Java, there is no method by name ‘circle()’ because, if width and height are same then it becomes a circle. By the same instruction of drawOval() the shape of circle is created putting width and height in same size. Below, it is shown how a circle is drawn in Java.

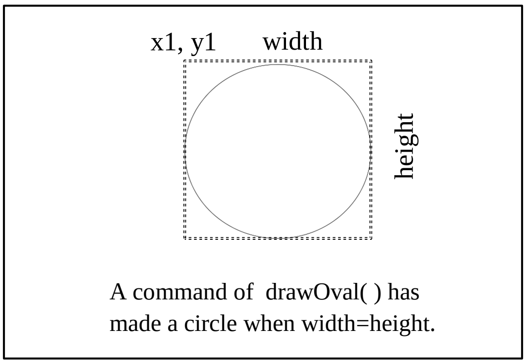

x1, y1
width
height
A command of drawOval( ) has
made a circle when width=height.

Fig. 7.7. figure shows concept of drawing an oval within a rectangle area. It also shows that if width and height are same then an oval becomes a circle.

Graphic design and programming

Some theories of Graphic design:

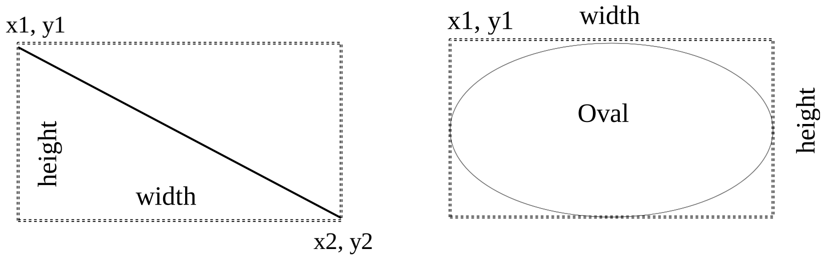
Many JAVA components used for graphic program are swing, awt, JFrame, applet, Graphics which have all required methods and components line Class, constructor to develop all graphics design related user required jobs.

The top left corner coordinate value is considered as (0,0) which is for (column, row) or (x, y) values known as starting point on the screen or called as zero point. All drawings will be within the range as mentioned in previous examples: width=500 and height=400. Some examples of using coordinate values are shown below:

Syntax of Line is : drawLine(x1, y1, x2, y2);

Drawing a rectangle :

drawRect(x1,y1,width, height); starting from at the coordinate(x1, y1) to ending to coordinate(x2,y2) is the corner of the rectangle. Where x2 = x1+ width and y2= y1+ height. It is shown below:



Coordinates of a rectangle.
(i)

A Oval within the same rectangle.
(ii)

Fig. 7.8. figure shows (i) concept of a rectangle, to draw a rectangle it requires starting coordinate(x1,y1), height and width (ii) to draw an oval starting coordinate(x1,y1), height and width are required. From this figure it is clear how an oval is drawn in Java.

From this concept, we see if the width and height are same of an Oval, then it will be a circle. For this reason, in Java, there is no method by name 'circle' but it is oval.

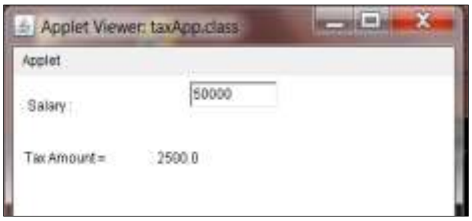
Problem: WAP an applet program to draw a window area and within that input salary amount of a person, compute 5% tax amount and display the tax amount in another place within the window.

```
//taxApp.java: in this example, input Salary, then calculate tax amount as considering
//5% of salary.
import java.awt.*;
import java.applet.*;
public class taxApp extends Applet
{
    TextField tx1;
    public void init()
    {
        tx1=new TextField(8);
        add(tx1);
        tx1.setText("");
    }
    public void paint(Graphics g)
    {
        int x=0,y=0;
        String s1,s2,s;
        g.drawString(" Salary :", 10, 30);
        try
        {
            s1=tx1.getText();
            x=Integer.parseInt(s1);
        }
        catch(Exception ex){}
                double z=x*(0.05);//calculating tax amount
        s=String.valueOf(z);
        g.drawString("Tax Amount = ", 10, 75);
        g.drawString(s,130,75);
    }
    public boolean action(Event event, Object ob)
    {
        repaint();
        return true;
     }
}
```

```
//taxApp.html:
<html>
<applet code=taxApp.class height=500 width=400>
</applet>
</html>
```

Run the program:

```
E:\example>appletviewer taxApp.html
```



```
Salary          50000
Tax Amount =    2500.0
```

Fig. 7.9. figure shows using applet how to create a display panel like window having fixed height and width. It also shows how to input and display at particular row-column position.

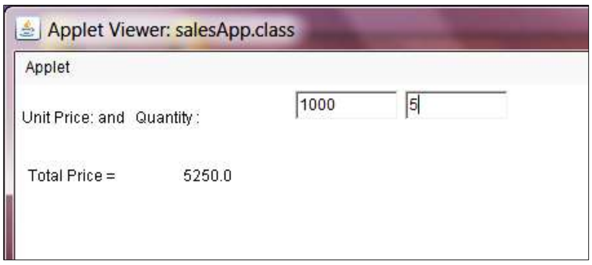
Discussions: the above Java program creates a new class from Applet class and it is done through 'extends' keywords. So, the new class 'taxApp' will inherit all properties of Applet class without private properties. Another class 'TextField' is used for creating objects 'txt1' and then different methods of 'TextField' class are invoked as tx1.setText("") and tx1.getText(); size of textfield is assigned by the statement TextField(8) which will give scope to input maximum 8 characters/digits.

```
//salesApp.java
import java.awt.*;
import java.applet.*;
public class salesApp extends Applet
{
    TextField tx1, tx2;
    public void init()
    {
        tx1=new TextField(8);
        tx2=new TextField(8);
        add(tx1);
        add(tx2);
        tx1.setText("");
        tx2.setText("");
    }
    public void paint(Graphics g)
    {
        int x = 0, y = 0;
        String s1, s2, s;
        double z;
        g.drawString("Unit Price: and ", 5, 30);
        g.drawString(" Quantity :", 90, 30);
        try
        {
            s1= tx1.getText();
            x = Integer.parseInt(s1);
            s2 = tx2.getText();
            y = Integer.parseInt(s2);
        }
        catch(Exception ex){}
        z = x * y;
              double tax=z*(0.05);
              z = z + tax;
        s = String.valueOf(z);
        g.drawString("Total Price = ", 10, 75);
        g.drawString(s,130,75);
    }
    public boolean action(Event event, Object ob)
    {
        repaint();
        return true;
     }
}
```

Study issues: in this program some points should be observed carefully, as: declaring input field as TextField(8); and setting it as: tx1.setText(""); with empty date or blanks. Another important point is that ‘paint()’ is a user defined method and argument of this method is (Graphins g) meaning that ‘g’ is an object created from the Class ‘Graphics’. Instead of creating object ‘g’ in a separate line, here, in same place of argument of ‘paint(Graphics g)’ is created, so, in a same statement both jobs are performed here. Another user defined method is ‘action()’ is of type Boolean and two arguments are there as: ‘Event event’ – it looks that both are same but, careful observation will show you that first one has capital ‘E’ meaning that it is inbuilt class ‘Event’ from which we are creating an object ‘event’ having first letter ‘e’ small letter. In the same way, ‘ob’ is the object created from the class ‘Object’.

```
//salesApp.html
<html>
<applet code=salesApp.class height=400 width=600>
</applet>
</html>
```

Output:



```
Unit Price: and  Quantity :        1000    5

Total Price =          5250.0
```

Fig. 7.10. it shows how to create working panel as window and then how to input and display data at particular row-column position.

Discussion: in this example two text fields are created for inputting data of unit price and quantity of sales. Then it computes tax amount 5% and finally display the text ‘Total Price=’ at (10,75) which is the (x, y) values. Then, as shown in the program, the numerical value of ‘z’ is converted to string as: s = String.valueOf(z); Finally, the graphics object ‘g’ could print the ‘s’ at (130, 75) as:

```
g.drawString(s,130,75);
tx1.setText("");

tx1.getText();
```

It is seen that row values are same as 75, so, both ‘Total Price’ and ‘s’ are printed in same row. Many methods are used and they are:

```
g.drawString(..)
TextField(8);
Integer.parseInt(s1);
String.valueOf(z);
repaint();
```
etc.

Designing graphics using JFrame:

Before, many examples of designing graphics were shown using Applet or JApplet. Now, it will be discussed how graphics can be designed within a single java program using JFrame, JPanel, swing, awt etc. inbuilt components.

The component ‘awt’ is an older one while ‘swing’ is a newer one. Then latest component is JavaFX. The ‘awt’ components have ‘Frame’, ‘Button’ classes but in ‘swing’ more advanced components are designed as ‘JFrame’, ‘JButton’ etc. These are the advanced features of JAVA and will be discussed in next other chapters also.

Hierarchy of these components are shown below:

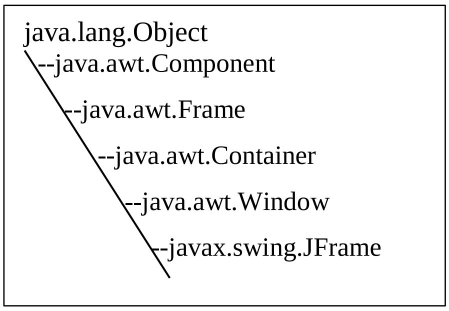


Fig. 7.11. Hierarchy of JFrame and other graphic design components.

Problem: wap to display all drawing components required to design any graphics job as: rectangle, square, line, arc, circle, fill the rectangle, fill the circle etc. that basic components of graphics designing are demonstrated.

```
//graphicDemo.java: to demonstrate different graphics components
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

  public class graphicDemo extends JFrame {
  public static final int width  = 550;
  public static final int height = 450;
   int x=100, y=70, w=300,h=250, angle1=30,angle2=60;
  private DrawCanvas cn;
  public graphicDemo() {
    cn = new DrawCanvas();
    cn.setPreferredSize(new Dimension(width, height));
    Container con = getContentPane();
    con.add(cn);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    pack();
    setTitle("Drawing Sheet");
    setVisible(true);
  }
  private class DrawCanvas extends JPanel
 {
    public void paintComponent(Graphics g)
   {
      super.paintComponent(g);
      setBackground(Color.WHITE);
      g.setColor(Color.BLACK);
      g.drawString("Demo of graphics tools", 100, 50);
      g.drawLine(x, y, 400, 70);  //X1,Y1,X2,Y2
      g.drawRect(x, y+20, 50, 50);  //x1,y1,width,height
      g.fillRect(x+50, y+80, 100, 50);
      g.setColor(Color.BLUE); //
      g.fillOval(x+100, 220, 130, 50);
      g.setColor(Color.BLACK);
      g.drawOval(x+50, 280, 130, 130);//circle
      g.drawLine(100, 430, 450, 430);
      g.drawString("Line before Translate of coordinate", 100, 440);
      g.drawArc(x, y+300, w, h, angle1, angle2);
      g.translate(50,130);
      g.drawString("Line after Translate of coordinate", 100, 415);
      g.drawLine(x, 420, 450, 420);//new: x=100+50; y=420+130
    }
  }
  public static void main(String[] args)
  {
    SwingUtilities.invokeLater(new Runnable()
    {
        public void run()
        {
           new graphicDemo();
        }
    }
                              );
  }//end of main
}
```

Output:

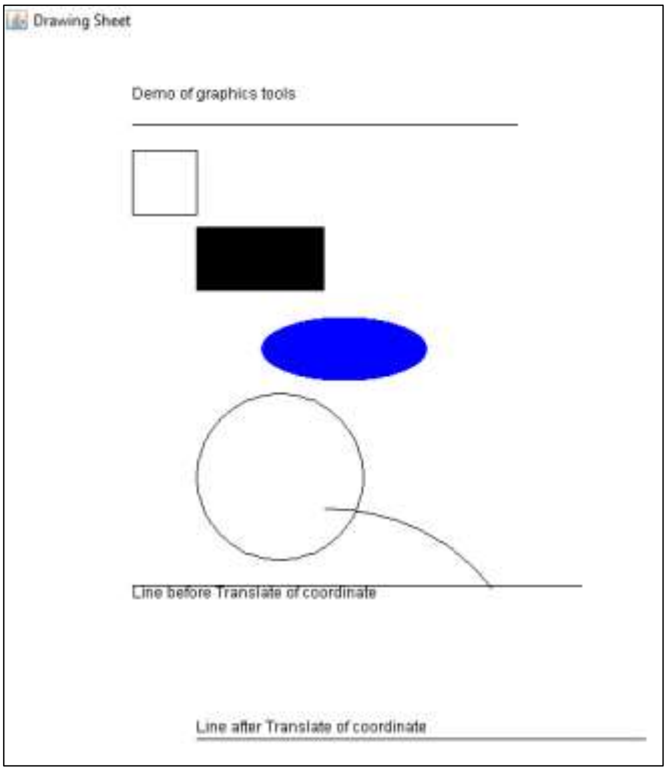

Drawing Sheet
Demo of graphics tools
Line before Translate of coordinate
Line after Translate of coordinate

Fig. 7.11. output of a graphic single program having main() module. It shows how to create all components required to design any graphics.

Discussions: in this program, or a basic template of graphic, all features required to design any graphic tool are demonstrated. There are many components with different shapes with different features, so, program size is increased.

Here, in this program, many classes are used as: **JFrame, DrawCanvas, Container, JPanel** and **Graphics**. Three interfaces like: **awt, event** and **swing** are imported at beginning of the program.
The graphicDemo() constructor, initializes drawing a canvas, holds the canvas within a container, set title of the drawing pane, define close conditions etc.

This program may be considered as a standard template for writing graphics programs and all required components are demonstrated.

Details of documentations on graphics design are available in: https://docs.oracle.com/javase/8/docs/index.html

-------X------

Questions:

1. Describe Applet.
2. Why does Applet not contain main() module?
3. What are the differences of Applet and JApplet?
4. What is the application of HTML?
5. How .html file is executed?
6. What do you understand by markup language?
7. How a applet program is executed?
8. What is an appletviewer?
9. What is the role of the keyword 'extends'
10. Say something about the package 'swing'.

Select for TRUE/FALSE as T/F

i. 'awt' is a package (T/F) :
ii. 'event' is a class (T/F) :
iii. The keyword 'extends' creates an object(T/F)
iv. In JAVA pointer concept is present(T/F)
v. 'this' is a operator which connects to a memory(T/F)
vi. 'implements' keyword is used for interfacing(T/F)
vii. 'mousePressed' is a catcher of mouse event(T/F)
viii. <applet > tag is used within .html file(T/F)
ix. Extension of a HTML file is .java(T/F)
x. drawRect(0,0,50,50); two 50,50 are last coordinate(T/F)

Ans: i (T), ii(F), iii(F), iv(F), v(T), vi(T),
vii(T), viii(T), ix(F), x(F).

Multiple choices of Questions:

a. In this statement drawOval(100,150, 40, 120); what is 100?
i. Horizontal length of the oval
ii. Vertical length of the oval
iii. X coordinate value of the rectangle within which oval is placed.
iv. Distance of the oval from the right end of the monitor

b. JFrame is a class under which package?
i. awt
ii. swing
iii. Window
iv. lang

c. Frame is a class under which package?
i. awt
ii. Container
iii. swing
iv. lang

d. In the method, drawLine(x1, y1, x2, y2); what is y2?
i. Distance from right corner
ii. Distance from the left corner
iii. y coordinate value of the second coordinate.
iv. Width of the rectangle

e. In drawString(..), the text will be displayed is as:
i. Text data
ii. As graphics data
iii. As integer data
iv. As ASCII data

f. In catch(..) method, what is its role?
i. To catch error values
ii. To catch exceptions error messages if any
iii. To store exceptions in a text file
iv. To write alternate code in this section

g. tx1.getText(); it will input text from where?
i. From any text file of hard disk
ii. From any scanner attached with computer
iii. from keyboard
iv. None of above

h. Integer.parseInt(s2); here parseInt() is a method of:
i. integer package
ii. Java package
iii. Integer type of data package
iv. Integer class

i. As hierarchy of 'JPanel' which one is correct?
i. javax.swing.JPanel
ii. javax.awt.JPanel
iii. java.swing.JPanel
iv. java.awt.JPanel

j. As hierarchy of 'Button' which one is correct?
i. java.swing.Button
ii. javax.awt.Button
iii. javax.swing.Button
iv. java.awt.Button

Ans.
a(iii), b(ii), c(i), d(iii), e(ii), f(ii), g(iii)
h(iv), i(i), j(iv)

Link of PPT of this book and all source codes:

https://drive.google.com/drive/folders/1bMxMCaqPe0W35COAdn_-BrnV_uzQ-tNO?usp=sharing

(Note: to get access, copy and paste this link to your browser. Both PPT and PDF version of presentation of this book are uploaded in Google Drive.)