Chapter Two

Applets

Definition: An applet is a program written in the Java programming language that runs on a web page. Applets can be run within any modern browser.

Packages and classes:

To implement applet, Java supplies a huge library of pre-written “code,” ready for you to use in your programs. One way to use this code is to import it. You can import a single class, or all the classes in a package.

The JApplet class must be the superclass of any applet that is to be embedded in a Web page or viewed by the Java Applet Viewer (appletviewer.exe).

Applets Structure

- There is no main method.
- Two methods that are called automatically- init() and paint()
- The init method initializes variables and objects; if you don't have one you will inherit one from the JApplet class.
- Use paint to draw screen
- A lot of methods exist in JApplet class so the "extends" keyword inherits everything that the class has. In the above example JApplet is parent class and shellapplet is the subclass so use the keyword "extends" to create inheritance.
- You have to import JApplet and java.awt.Graphics (abstract windowing toolkit) to get Graphics to paint.
- All applets must inherit JApplet

Graphics Class

- Browser or appletviewer sends a Graphics object to the paint method.
- The Graphics object represents the applet window, current font, and current color and provides methods to draw shapes (rectangles, triangles, circles, etc.) and text on the window.

The Graphics Class Coordinate System

- Never use 0,0 because the applet itself uses it for the title bar.

The `java.awt.Graphics` package includes classes for:

Drawing lines and shapes

Drawing letters

Setting colors

Choosing fonts

If it's drawn on the screen, then `java.awt.Graphics` is probably involved!

Since you may want to use many classes from the java.awt package, simply import them all by using * as:

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

Example

```
import javax.swing.JApplet;
import java.awt.Graphics;
public classMyFirstApplet extends JApplet
{
      // declare variables here

    public void init( )
    {
          // data initialization goes here
    }

   public void paint( Graphics g )
  {
          g.drawstring(“This is my first applet code”, 30,50);
   // other  code goes here
   }
}
```

- The window gets drawn by invoking the paint class of the applet, which calls the superclass paint which draws the actual window you've given the size for.

- Other applet methods

- **start()** called when the applet is visible. Start() is called by the browser or applet viewer to inform this applet that it should start its execution. It is called after the init method and each time the applet is revisited in a Web page. A subclass of JApplet should override this method if it has any operation that it wants to perform each time the Web page containing it is visited.
- **stop()** called when applet is no longer visible. Stop () is called by the browser or applet viewer to inform this applet that it should stop its execution. It is called when the Web page that contains this applet has been replaced by another page, and also just before the applet is to be destroyed.
- **destroy()** when hosting window is closed (exit). Destory() method is called by the browser or applet viewer to inform this applet that it is being cultivated and that it should destroy any resources that it has allocated. The stop method will always be called before destroy.

Running an Applet

we need to use html code to run applets. The minimum html required to run applet with a browser (java host) is as follows:

```
<HTML>
<HEAD>
<TITLE>TitleName</TITLE>
</HEAD>
<BODY>
<APPLET>
   CODE = Classname.class
   CODEBASE = . directory of class file
   WIDTH = 50 width of window in pixels
   HEIGHT = 50 height of window in pixels
</APPLET>
</BODY>
</HTML>
```

In Netbeans the html code is automatically generated and you can find it under File→Your applet project name→Build→your applet file html version→right click→view.

More Examples:

Example 1:

This example will show you the structure of an applet, the basic drawing methods on the applet viewer screen.

```
// required when you create an applet
import javax.swing.JApplet;

// required to paint on screen
import java.awt.Graphics;


// the start of an applet - HelloWorld will be the executable class
// Extends applet means that you will build the code on the standard Applet class
```

```
public class HelloWorld extends Applet
{

// The method that will be automatically called  when the applet is started
     public void init()
     {
 // It is required but does not need anything.
     }


// This method gets called when the applet is terminated
// That's when the user goes to another page or exits the browser.
     public void stop()
     {
     // no actions needed here now.
     }


// The standard method that you have to use to paint things on screen
// This overrides the empty Applet method, you can't called it "display" for example.

     public void paint(Graphics g)
     {
 //method to draw text on screen
 // String first, then x and y coordinate.
      g.drawString("This is my first Applet code",20,20);
      g.drawString("Hellow World",20,40);

     }

}
```

Example 2 :

```
// That's it. Next is drawing special shapes on screen and using fonts.


/*
Applet will paint special shapes and use colors and fonts
Only new methods are explained
*/

import java.awt.*;
import javax.swing.JApplet;
public class DrawExample extends Applet
{
```

```
// Specify variables that will be needed everywhere, anytime here
// The font variable
   Font bigFont;

// The colors you will use
    Color redColor;
    Color weirdColor;
    Color bgColor;

    public void init()
    {
 // Here we will define the varibles further
 // Will use Arial as type, 16 as size and bold as style
 // Italic and Plain are also available
         bigFont = new Font("Arial",Font.BOLD,16);

 // Standard colors can be named like this
         redColor = Color.red;

 // lesser known colors can be made with R(ed)G(reen)B(lue).
         weirdColor = new Color(60,60,122);

         bgColor = Color.blue;

 // this will set the backgroundcolor of the applet
     setBackground(bgColor);

    }

    public void stop()
    {
    }

// now lets draw things on screen
    public void paint(Graphics g)
    {
 // tell g to use your font
     g.setFont(bigFont);
     g.drawString("Shapes and Colors",80,20);

 // Now we tell g to change the color
     g.setColor(redColor);

 // This will draw a rectangle (xco,yco,xwidth,height);
     g.drawRect(100,100,100,100);

 // This will fill a rectangle
     g.fillRect(110,110,80,80);
```

```
  // change colors again

     g.setColor(weirdColor);

  // a circle (int x, int y, int width, int height,int startAngle, int arcAngle);
  // ovals are also possible this way.

     g.fillArc(120,120,60,60,0,360);

     g.setColor(Color.yellow);

  // Draw a line (int x1, int y1, int x2, int y2)

     g.drawLine(140,140,160,160);

  // reset the color to the standard color for the next time the applets paints
  // an applet is repainted when a part was'nt visible anymore
  // happens most often because of browser minimizing or scrolling.

     g.setColor(Color.black);

    }

}
```

Summary on drawing figures:

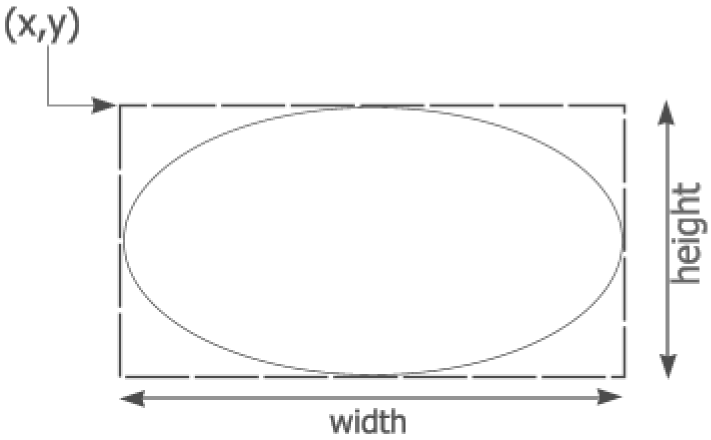
- Draw a line, g.drawLine(xStart, yStart, xEnd, yEnd);

- Draw a rectangle g.drawRect(x, y, width, height);

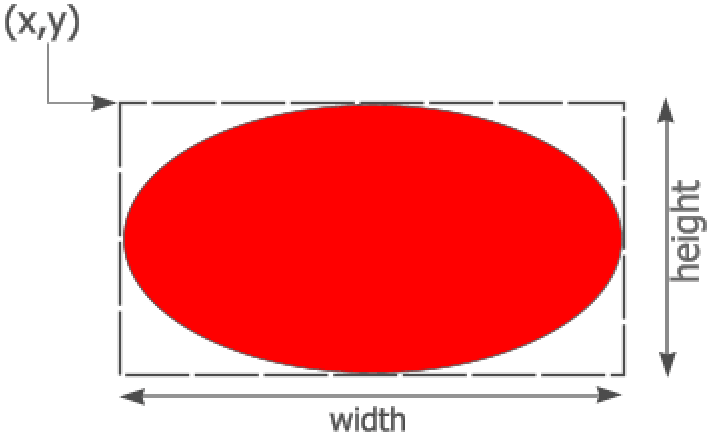
- Draw a solid recta ngle: g.fillRect(x, y, width, height);

- Draw am oval: g.drawOval(x, y, width, height)

-Draw circle:g.drawarc(int x, int y, int width, int height,int startAngle, int arcAngle)



- Draw a filled (solid) oval: g.fillOval(x, y, width, height);



<u>GUI and database connection of Applet</u>

Apply your previous general database connection concept; the only difference is using JAppletForm rather than JFrameForm