

## Lab 7: assigned 10/19 , due 10/31

**Simple Graphics on your computer** - The purpose of this lab is to demonstrate JFrame use, the concepts of inheritance, the Color class, the Graphics helper class, and the paint method.

*Please read the following explanation of inheritance:*

### A. Extending a class:

Up until now, we have used JFrame as a library class, and created an object from it to draw a window on screen, like this:

```
JFrame myWindow = new JFrame();
```

with methods such as:

```
myWindow.setLocation( 100, 100 ); // etc...
```

This week we are learning about INHERITANCE, in which we construct a class that not only uses, but BECOMES JFrame, by inheriting, or "extending" all of its capabilities. If we define our class like this:

```
public class Lab7Class extends JFrame
{
```

then Lab7Class inherits all of JFrame's methods directly, and we can use JFrame methods as if they are copied right into Lab7Class, with no object necessary:

```
setSize( 500, 500 );
setDefaultCloseOperation( EXIT_ON_CLOSE );
setVisible(true);
```

(note, no objects or dot notation).

### B. The paint( ) method and the graphics helper class:

The author of the original JFrame class, provides a method that we can't see, but that we can make use of... called the **paint ( )** method. Since we are creating a class that is in essence a copy of the JFrame class itself, we can re-write the paint method to perform our own graphics functions. We just have to follow the author's rules, which is to define the paint method like this:

```
public void paint( Graphics g )
{
    // write stuff using the g object here
}
```

the purpose of the "Graphics g" parameter is to provide an object ("g") that itself has useful methods for drawing. We are over-riding, or rewriting the paint method ourselves to create graphics on-screen. The paint method is called automatically by the computer (much like main). We just put stuff in it, and the computer does the drawing.

## Lab 7: Do the following 5 steps:

**Step 1:** Type in the program example below. This is the basic framework for a graphics program. Make sure that it runs and behaves.

```
import java.awt.*;
import javax.swing.*;

public class shapesClass extends JFrame {

    public shapesClass()
    {
        setSize( 500, 500 );
        setDefaultCloseOperation( EXIT_ON_CLOSE );
        setVisible(true);
    }

    public void paint(Graphics g)
    {
        g.setColor( Color.white ); // what does this do?
        g.fillRect( 0, 0, 500, 500 );

        g.setColor( Color.red );
        g.fillRect( 50, 50, 100, 75 );
    }

    public static void main(String args [] )
    {
        shapesClass app = new shapesClass();
    }
} // end class
```

imports

class declaration, extends JFrame

constructor

paint method

graphics

main method

Note that there are 5 parts to any graphics program. Make sure that you can identify them in the code:

- the imports
- the class declaration, which extends JFrame
- the constructor, which sets up a window
- the paint method, which uses Graphics to draw things
- the main method, which instantiates the class and runs the constructor

**Step 2:** Now, do some research. Look up the Graphics class (Google Graphics class, and look for pages from the Sun Java web site). Find a Graphics method to draw a line. Place a method call in your paint method to draw a black line anywhere on the frame.

**Step 3:** Do some more research. Find a way to draw a pink Oval.

**Step 4:** Do some more research. Draw 5 different shapes on your window, all of them different colors.

**Step 5: You must comment your code to answer the following questions:**

- a. What import statement allows us to use JFrame?
- b. What import statement allows us to use the Graphics class?
- c. Where is the JFrame drawn?
- d. Where is the rectangle drawn?
- e. What is the purpose of the constructor?
- f. What is the purpose of the main method?
- g. What are the arguments in the call to drawLine?

**Submit the java file using submit\_cse115c**