DUE DATES:
Monday recitations: 9:00 PM on 10/09
Wednesday recitations: 9:00 PM on 10/11
Thursday recitations: 9:00 PM on 10/12
Friday recitations: 9:00 PM on 10/13
Saturday recitations: 9:00 PM on 10/14

Ready!

We've been talking about instance variables (how to declare and intitialize them) and also mutator methods (which allow us to change the values of instance variables). In this lab you will get some practice writing a sequence of programs which let you draw pictures with lines of different colors and widths.

If you are unsure how to do these things you should review your lecture notes, slides posted on the course website, and readings from the textbook.

Before describing the required lab work we need to discuss a few points needed for this (and future) labs.

IMPORTS

Rather than always fully qualifying a class name from another package, you can "import" it. Importing a fully qualified name allows you to use it in an unqualified way throughout a file. This can make both writing and reading the code easier. Rather than writing this code:

```
package part3;

public class Farm {
     public Farm() {
          example1.BarnYard b;
          b = new example1.BarnYard();
          example1.Pig p1;
          p1 = new example1.Pig();
          b.addPig(p1);
          p1.start();
          example1.Pig p2;
          p2 = new example1.Pig();
          b.addPig(p2);
          p2.start();
     }
}
```

we can use imports to make the names example1.BarnYard and example1.Pig available in their

unqualified forms, BarnYard and Pig:

```
package part3;

import example1.BarnYard;
import example1.Pig;

public class Farm {
     public Farm() {
          BarnYard b;
          b = new BarnYard();
          Pig p1;
          p1 = new Pig();
          b.addPig(p1);
          p1.start();
          Pig p2;
          p2 = new Pig();
          b.addPig(p2);
          p2.start();
     }
}
```

STATIC VARIABLES
Instances of the class java.awt.Color represent colors. Some color objects are "pre-defined". References to these objects are stored in static variables of the Color class. A static variable is often public, and is associated with the class rather than an instance. For example, Color.BLUE is a static variable which holds a reference to a Color object representing the color blue.

REMEMBER: do **NOT** label any of **your** variables as static.

PUBLIC VARIABLES
In this lab you will use instances of the class java.awt.Point and java.awt.Dimension.

Point objects, which represent points on a plane, have two instance variables representing the x and y coordinates of the point. These instance variables are public rather than private. If the variable p is of type Point then p.x and p.y are the two instance variables.

Dimension objects, which represent dimensions, have two instance variables representing the width and height of the dimension. These instance variables are public rather than private. If the variable dim is of type Dimension then dim.width and dim.height are the two instance variables.

REMEMBER: do **NOT** label any of **your** instance variables public.

Set!

1. Log in
2. Start Eclipse
3. Switch to the CVS Repository Exploring perspective
4. Check out the CSE115-Lab4 project from the Labs repository
5. Switch to the Java perspective

Go!

For this lab we have written some code that creates a simple graphical user interface. You will be able to do this on your own soon enough, but we need to cover a few more topics first. For now our code will handle user inputs through the user interface (UI) and your job is to write the code that responds correctly to the user input.

Part 1

Edit the definition of the Lab4 class in the part1 package so that it has an association relationship with the part1.support.Canvas class. Establish the association via the constructor (the simple version we learned first in lecture).

A Canvas object consists of a region on the screen on which we can draw. You will define the Lab4 class so that when a user releases the mouse button in the Canvas a circle with diameter 40 is drawn at the point where the user released the mouse button. The point determines the upper left corner of the bounding box of the cirlcle:

When this happens the support code we wrote will attempt to call a method named 'drawCircle' on a Lab4 object.

You must define this 'drawCircle' method. It must be a void method that has one java.awt.Point parameter. This method needs to get a Graphics2D object from the Canvas object that Lab4 is associated with. Canvas defines a parameterless accessor method called getGraphics2D you need to call for this purpose.

The Graphics2D object in turn defines a drawOval method that takes four values, x, y, w, and h. It draws an oval at (x,y) with a width of w and a height of h. Since we want to draw a circle with diameter 40 you need to make sure that the last two arguments of the drawOval method call are both 40.

See the discussion above about how to extract the x and y coordinates from a java.awt.Point.

Once you have completed part 1 you can run it (use the Driver provided). When the window opens you should be able to draw thin black circles in the Canvas.

Part 2

In this part of the lab we will add the ability to change the color of the circle that's drawn. The support code now includes several buttons labelled with different colors. Clicking on a button will cause the support code to call a setColor mutator method on Lab4. You must define this mutator, in addition to making some other changes in your drawCircle method.

First edit the definition of the Lab4 class in the part2 package so that it does everything that the Lab4 class in part 1 did.

Because **selecting a color** for a circle and **drawing a circle** are distinct activities, you must add code to Lab4 so that it remembers the last color selected until such time as the user draws a circle.

To do this you need to declare an instance variable of type java.awt.Color in the Lab4 class. Initialize your instance variable to Color.GREEN in the constructor. Define the setColor mutator method. Remember that this will be a public void method whose parameter list declares a variable of type java.awt.Color. The body of the method will simply assign the value of the parameter to the instance variable.

Finally, you need to modify the drawCircle method so that just before the circle is drawn the color of the Graphics2D object is set to the last selected color. Do this by calling the setColor mutator method on that Graphics2D object.

Note that the setColor mutator method you define for Lab4 is different from the setColor mutator method already defined in the Graphics2D object.

Once you have completed part 2 you can run it (use the Driver provided). When the window opens you should be able to draw thin lines in various colors on the Canvas.

Part3

In this part of the lab we will add the ability to change the diameter of the circle that's drawn. The support code in part 3 also includes several buttons labelled with different diameters. Clicking on a button will cause the support code to call a setDimension mutator method on Lab4. You must define this mutator, in addition to making some other changes in your drawCircle method.

First edit the definition of the Lab4 class in the part3 package so that it does everything that the Lab4 class in part 2 did.

Because **selecting a diameter** for a circle and **drawing a circle** are distinct activities, you must add code to Lab4 so that it remembers the last dimension selected until such time as the user draws a circle.

To do this you need do what you did in part 2 (to handle color), but now to handle a java.awt.Dimension value. Initialize this instance variable to a new java.awt.Dimension(80,80) value in the constructor, and add the setDimension mutator method.

Finally, you need to modify the drawCircle method so that the last two arguments of the drawOval method call are the width and height components of the java.awt.Dimension object. 'width' and 'height' are public instance variables of the Dimension object, just like 'x' and 'y' are public instance variable of a Point object. (Remember, do **NOT** label any of **your** instance variable public!)

Once you have completed part 3 you can run it (use the Driver provided). When the window opens you should be able to circle in various colors and diameters on the Canvas. Have fun – get creative and make some cool pictures!

Part4

In this part of the lab we will add the ability to change the width of the line used to draw circles. The support code in part 4 includes several buttons labelled with different line widths. Clicking on a button will cause the support code to call a setStroke mutator method on Lab4. You must define this mutator, in addition to making some other changes in your drawCircle method.

First edit the definition of the Lab4 class in the part4 package so that it does everything that the Lab4 class in part 3 did.

Because **selecting a stroke** width for a line and **drawing a circle** are distinct activities, you must add code to Lab4 so that it remembers the last stroke selected until such time as the user draws a circle.

To do this you need do what you did in part 2 (to handle color) and part 3 (to handle diameter), but now to handle a java.awt.BasicStroke value. Initialize this instance variable to a new BasicStroke(5) value in the constructor, and add the setStroke mutator method.

Finally, you need to modify the drawCircle method so that just before the circle is drawn not only is the color of the Graphics2D object set but also the stroke. Do this by calling the setStroke mutator method on that Graphics2D object, passing along the value stored in the new instance variable.

Once you have completed part 4 you can run it (use the Driver provided). When the window opens you should be able to draw thin lines in various colors and thicknesses on the Canvas. Have fun – get creative and make some cool pictures!

Submitting your project to Web-CAT

Make sure you submit your work on time; due dates are listed at the beginning of this lab description. This lab will be automatically graded by Web-CAT. You may submit as many times as you wish. Your last submission is the one that counts (so consider carefully whether you want to make any late submissions, as the late penalty is 20 points per day or portion thereof late).
