CS 158 Lab 11 February 23, 2010
Displaying Drawings in Swing

1. It will take some work to develop a good way of displaying the asteroid game. Today we will work on developing the component that will be repeatedly redrawn. This component will have to contain all of the objects of the game. Begin by adding a class to the **gui** package called **MyComponent** and make it a descendent of **JComponent**. Add a **paint** method that does not return anything and takes an argument of the **Graphics** class. We will fill in an implementation for this method, but we will not call it directly. It overrides a method of the parent class and gets called automatically as a result of calling other methods. In the **paint** method, send to the **Graphics** object a **setColor** message, and supply as an argument any color you like from the **Color** class (which has a bunch of static colors for you to choose from). Then send the **Graphics** object a **fillOval** message, and as arguments use 100, 200, 20, and 20 for the x and y location of circle of size 20.

2. Create another class in the **gui** package called **GameWindow** and give it a **main** method. Make this class inherit from **JFrame** and implement the **DisplayObjects** interface. Today we will not worry about actually making it implement that interface, so just create the required method and don't put anything into it. Create static constants in this class called **FRAME_WIDTH** and **FRAME_HEIGHT**, and set them to 500. Add a constructor that calls the following inherited methods: **setTitle** and supply a string argument "Asteroids" (or whatever you want), **setSize** and supply the two constants you just created, and **setDefaultCloseOperation(EXIT_ON_CLOSE)**. In the **main** method instantiate a **GameWindow** object, send it an **add** message and supply a new **MyComponent** object as an argument. Then send the **GameWindow** object a **setVisible** message and supply **true** as its argument. Run this and see that you get a circle appearing somewhere in a new window.

3. You can have some fun with putting the ball on other places in the window, or try using the **fillRectangle** method or other methods of the **Graphics** class. You can try placing several drawings on your component and then seeing what they look like.

4. Here is the main task for today. Your component object, **MyComponent**, must display all of the asteroids at once. To do this, it will have to get them from the **Board** one at a time and add their locations and sizes to its own variables. This will take all of the remaining steps. Create parallel arrays for x and y values and sizes as instance variables. You will need to save the **Graphics** object in your component as an instance variable as well, so that you can use it from other methods. The **paint** method receives the **Graphics** object, so that's where you will be able to set the instance variable.

5. Add a constructor to **MyComponent** that takes an integer argument for the number of asteroids. Have the constructor store that number, and instantiate all of the parallel arrays. This constructor will cause an error to appear in the **GameWindow** class, because it was called without a parameter last time. We will fix that later. Still in **MyComponent** create a method called **setAsteroid** that takes 4 arguments: the asteroid number it's setting (which will be the subscript of the parallel arrays), the

asteroid's x and y location, and the asteroid's size. Set the appropriate elements of the parallel arrays. See if you can figure out how to "scale" the locations so that the asteroids end up proportionately in the window where they were on the **Board**.

6. Create a method in your component object called **drawAsteroids** that will draw all the asteroids by repeatedly calling the **fillOval** method of the **Graphics** object and supplying it with the appropriate information from the parallel arrays. Add a call to **drawAsteroids** in the **paint** method.

7. Make this work from the **main** in **GameWindow** by creating the board and about 10 asteroids. You can now fix the error with the **MyComponent** constructor, since you know how many asteroids you are adding. Now you can repeatedly call the **setAsteroid** method to add the asteroids to your component object. Finally, add your component to the window and make it visible.