# Programming Assignment 6 ( 100 Points )

## Due: 11:59pm Thursday, February 21th

## README ( 10 points )

You are required to provide a text file named **README,** NOT Readme.txt, README.pdf, or README.doc, with your assignment in your pa6 directory. There should be no file extension after the file name "**README**". Your README should include the following sections:

**Program Description ( 5 points ) :** Provide a high level description of what your program does and how you can interact with it. Make this explanation such that your grandmother or uncle or someone you know who has no programming experience can understand what this program does and how to use it. For example, if you intentionally add padding to a line so it is easier to grab, say it here so the grader knows it was intentional.

Write your READMEs as if it was intended for a 5 year old. Do not assume your reader is a computer science major. **The more detailed the explanation, the more points you will receive.**

**Short Response ( 5 points ) :** Answer the following questions:

1. Using vim/gvim, how can you open a file from the Linux command line so that you are taken directly to a specified line in the file?
2. What run time structure holds the local variables and formal parameters with each method invocation?
3. What are the only things that can be declared in a Java interface? Be specific with full access modifiers and keywords.
4. How do you turn off special coloring of keywords and syntax in vim (this is most evident in gvim)? For example, by default your gvim colors comments differently from other parts of code (comments are colored blue by default). If you turn off syntax coloring then all the code in your file will be of same color, black by default. How do you turn the coloring back on?
5. What is the command that you type in to view more information on how to use vim/gvim? The specific command that we're looking for is the one you type in command mode in an open vim/gvim file, not on the terminal. Typing the command will bring up a help manual for vim.

6. How do you jump to a specific subject in the help manual for vim? (This is easily answered using the answer to question 5).

## STYLE ( 25 points )

Please see preview programming assignments.

You will be specifically graded on commenting, file headers, class and method headers, meaningful variable names, judicious use of blank lines, not having more than 80 characters on a line, perfect indentation, no magic numbers/hard-coded numbers other than zero, using setters and getters, etc.

## CORRECTNESS ( 65 points )

For this assignment, you will extend your PA4 ResizableBall applet to include standard Java GUI components (JButtons, JLabel, JScroller, JPanels) and standard Java Event Handling. You will want to reference Ch 11 and the notes for Ch 11!

You will first want to make your new pa6 directory and copy the appropriate pa4 files over into this new pa6 directory:

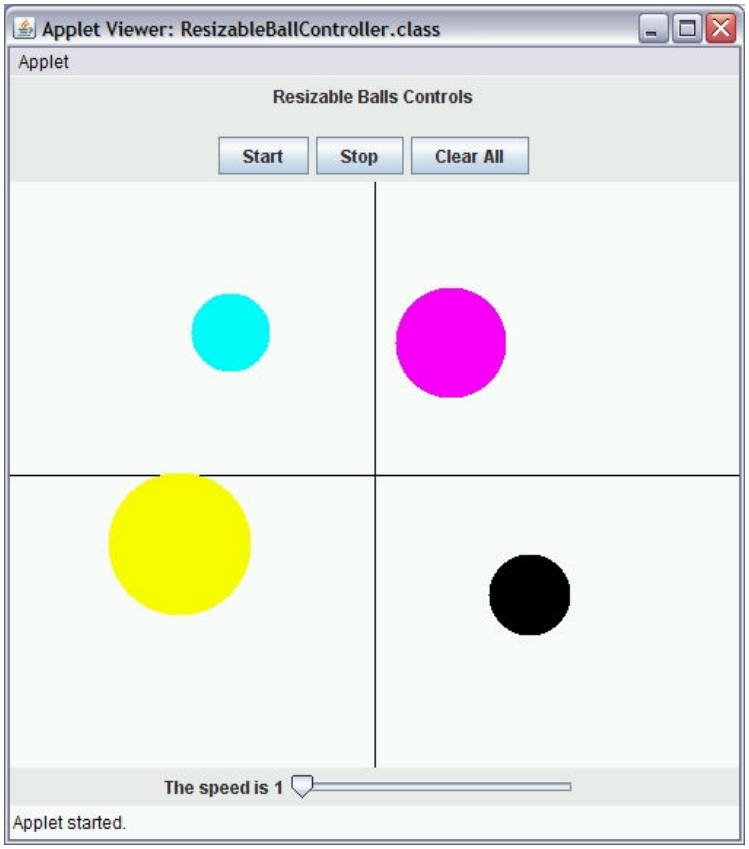
```
> cd
> mkdir pa6
> cp pa4/Resizable*.java pa6
> cp pa4/objectdraw.jar pa6
> cp pa4/*.html pa6
```

If it is not already, you should probably name the html file ResizableBallController.html

Now you can cd into pa6 and get started.

### Begin the Coding

Let's take a look at a screen shot of what your new applet should look like:




As you can see, the basic layout of the canvas with the horizontal and vertical lines and the

resizable balls are the same along with their same functionality from PA4. But notice the control panel at the top with a centered label and three buttons and the bottom panel with a label indicating the current speed of the resizable balls and a slider to control the speed (the smaller the speed value the slower the balls resize -- MIN_SPEED is 1 and MAX_SPEED is 100).

You should probably get the new layout with the panels and labels and buttons and scroller done first without worrying about the event handling on these GUI components. Ch 11 in the textbook explains what the default layout manager is for a WindowController. We will still be extending WindowController and using the drawing canvas for the balls. Resizing the applet should still keep the lines proportional but the balls don't move (same functionality as PA4).

**The book is your best friend for this assignment.**

Once you get the layout looking good, time to add some standard Java event handling. You will need to change from using the objectdraw helper event handlers to using the standard Java event handlers. For example, change from using

```
public void onMouseClick( Location point )
```

to using (and defining)

```
public void mouseClicked( MouseEvent evt )
```

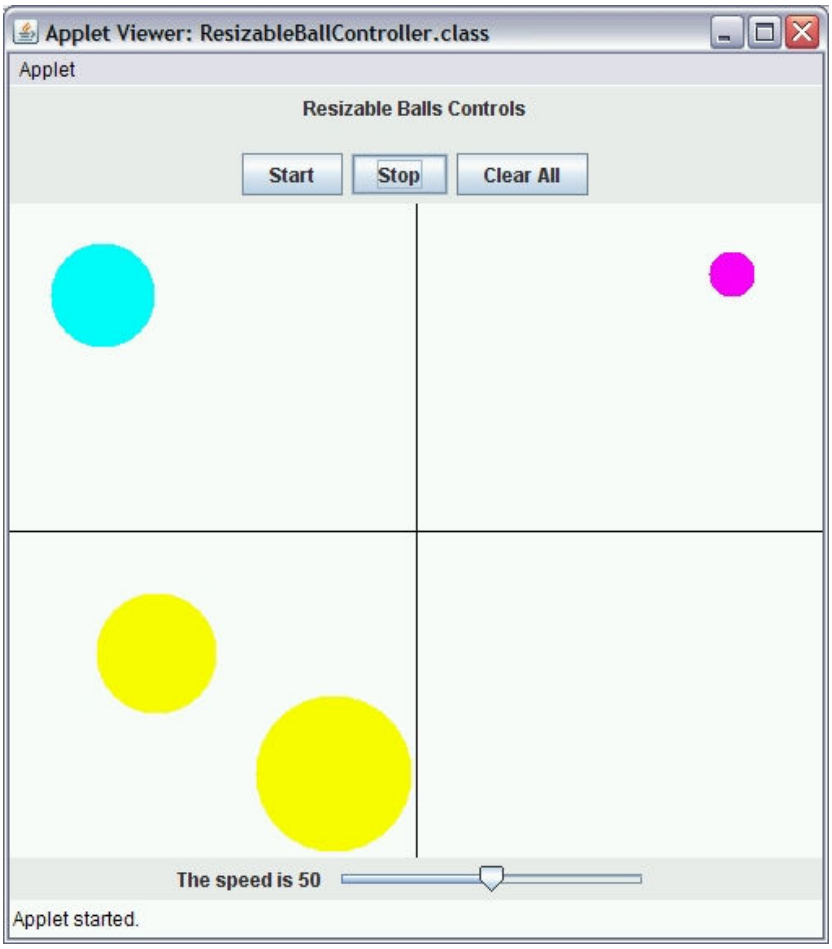
modifying any code that used a Location object to using a MouseEvent object and add the appropriate interfaces. Again **Ch 11 and the notes should be excellent guides**. Remember you will need to provide an implementation for each abstract method declaration in the interfaces you implement. You will need to do this in both your ResizableBallController.java and ResizableBall.java source files. There should be no onMouse*() methods. However, we are still using objectdraw library objects like Line and FilledOval and their methods like contains() which takes a Location object. So when necessary, you will need to create Location objects from the MouseEvent's x and y - for example, evt.getX() and evt.getY() - see the Java API docs for MouseEvent and the objectdraw docs for Location.

Each ResizableBall object will need to handle the events from each of the buttons and the scroller. The ResizableBallController will also need to respond to the scroller to change the value in the label indicating the current speed and respond to the start and stop buttons to pass an indication of whether a new ResizableBall when created and initialized should start off resizing or not. Clicking the mouse to create a new ResizableBall after the stop button has been pressed should create a new ball but the ball should not be resizing until the start button is pressed.

**You will probably need to make heavy use of the on-line docs for the objectdraw library and the standard Java API libraries.**

> **Tip**: Bring the book and notes to the lab. They have pretty much everything you need to implement this assignment.

Once you get all that going where you can start and stop the balls resizing and add more balls and start and clear all and add and stop and clear some more and add ... etc., then you want to add the ability to grab a ball and drag/move it around the canvas. The ball should change color as it is getting dragged from one quadrant to another (based on the center of the ball). This should work whether the ball is resizing or not. If balls overlap each other you can grab multiple balls in the overlapping area and drag them together around the canvas. If a ball overlaps one or both lines you can grab the ball(s) and the line(s) at the same time and move them all together.




Note: Do not allow a ball to be dragged off the canvas area. Use the same 5 pixels margins you used in PA4 to prevent either line from being dragged off the canvas and do not allow the center of a ball to be dragged past that margin.

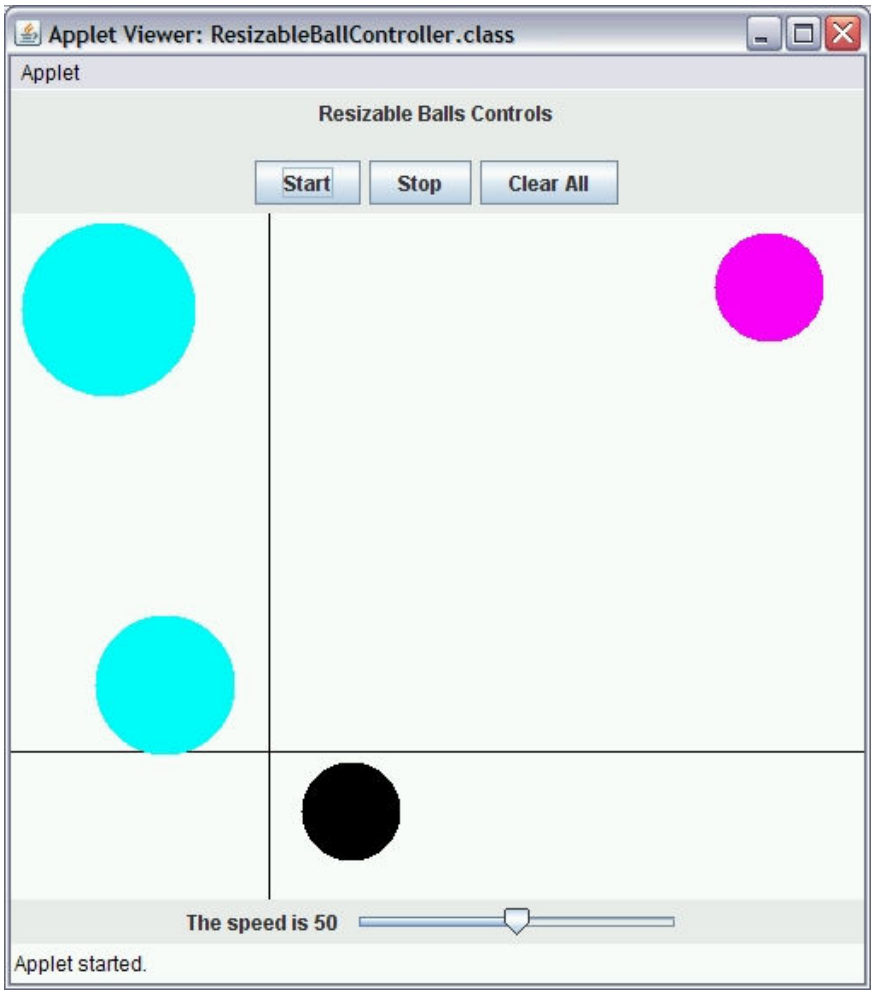
Some of you may want to implement dragging the balls around before the buttons and scroller event handling. It is up to you.

You will probably need to modify the ResizableBall constructor to deal with new information that needs to be sent to each ball so each ball can operate independently. There should be no mechanism in the controller object that knows anything about how many balls there are or where they are or if they have been cleared from the canvas, etc. Each ball handles the various events (mouse, button, scroller) on its own and makes its own decisions based on things like where its center is located and how big it is in its resizing cycle.

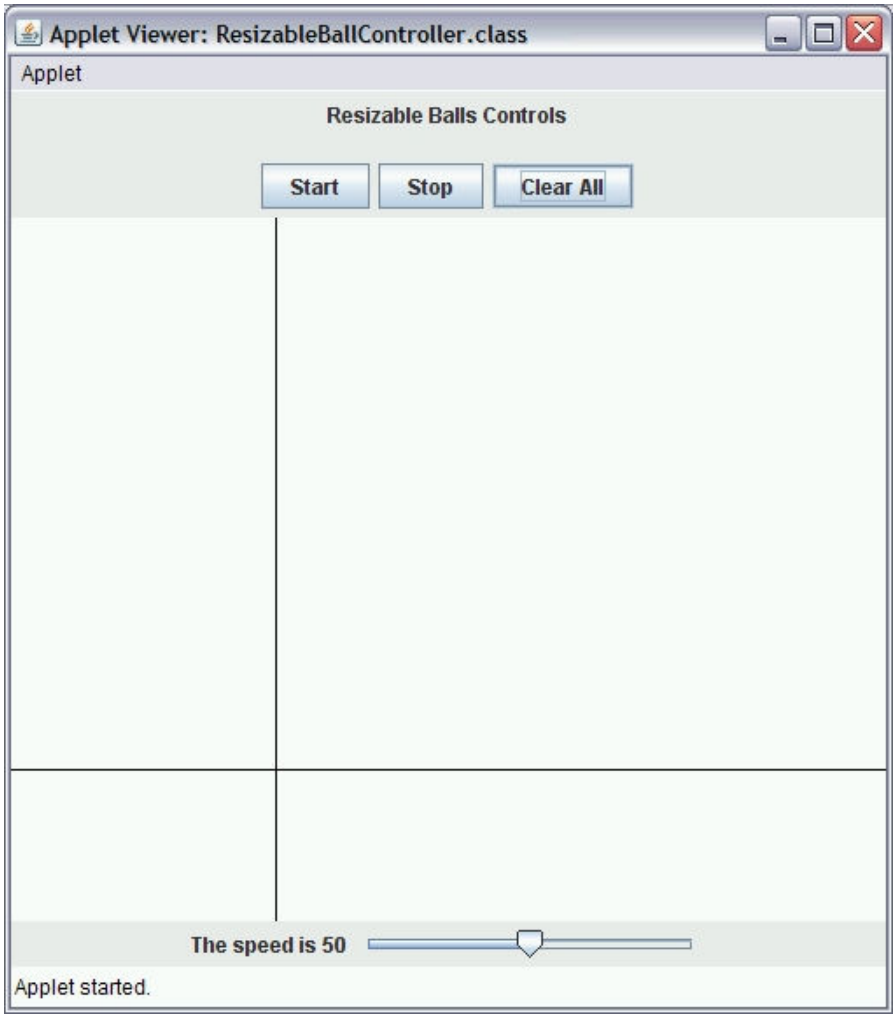
When a ball is cleared from the canvas, it's run() method running in its own thread should end (change the forever loop in run() to have a condition that will fail when the ball is cleared from the canvas).

The screen shot on the previous page is after moving the balls around a bit, changing the resizing speed, and hitting the stop button.

And here is a screen shot after moving the lines around a bit and hitting the start button.



And then after clicking on the Clear All button. The lines should stay in the same place.




**Running**

To run (and view) your applet, first create an html file named **ResizableBallController.html**, which contains the following code:

```
<html>
   <body>
          <applet code="ResizableBallController.class"
          archive="objectdraw.jar" width="500" height="500"></applet>
   </body>
</html>
```

Then use the appletviewer command specifying this html file:

> **appletviewer ResizableBallController.html**

You must have all the files in the pa6 directory and run appletviewer in the pa6 directory for it to display correctly.

**Turnin**

To turnin your code, navigate to your home directory and run the following command:

> **turnin pa6**

You may turn in your programming assignment as many times as you like. The last submission you turn in before the deadline is the one that we will collect.

**Verify**

To verify a previously turned in assignment,

> **verify pa6**

If you are unsure your program has been turned in successfully, use the verify command. We will not take any late files you forgot to turn in. Verify will help you check which files you have successfully submitted. **It is your responsibility to make sure you properly turned in your assignment.**

**Files to be collected:**

- README
- objectdraw.jar
- ResizableBall.java
- ResizableBallController.html
- ResizableBallController.java

**NO LATE ASSIGNMENTS ACCEPTED!**

**START EARLY!**

**and … HAVE FUN!**

## Extra Credit ( 5 points )

Currently, the balls should not move with the lines when the window is resized. When the applet is resized, have the center of the balls move like the lines to stay in the same proportional locations. This is pretty easy! So, if you need extra credit, you should try it!