Mini-assignment 2: Moving a circle with the mouse

Handed out in Week 6: 3rd March 2014

Introduction

This mini-assignment is designed to give you more experience of playing with a Swing program. You will be modifying the code from Topic05 Interactive Graphics to allow you to program the mouse dragging a circle.

This mini assignment (worth 10% of your mark) is to encourage your familiarity with Swing:

YOUR WORK MUST BE SIGNED OFF BY YOUR DEMONSTRATOR IN YOUR ASSIGNED PRACTICAL SLOT. YOU NEED TO HAVE THIS SHEET SIGNED OFF AT THE LATEST DURING WEEK 8 THE **WEEK BEGINNING 17th March.**

Worksheet Steps

There are several tasks to complete in this worksheet. Work through each task in turn.

Task 1: Download the code and load into Eclipse or equivalent

On Blackboard, you will find the starting code for this project. This is the code from Lecture05 for Interactive Graphics basicGraphics-code-with-comments.zip.

Note: you can use a different IDE, or none, if you want to. The key issue is that you can compile and run the code you have just downloaded and that you can then modify the code to complete this worksheet. **If you use an IDE then do NOT use a drag and drop Swing GUI builder.** We want you to understand how Swing works when you build GUIs from scratch. Next year you may use GUI builders.

Task 2: Look at the existing code

Run the program and be sure you understand what is happening.

Then look at the code. In particular look at the MyMouseListener and its actionPerformed() method, which includes the code:

```
if(SwingUtilities.isLeftMouseButton(e)) {
     mousePane.addCircle(e.getX(),e.getY());
}
else {
     mousePane.removeNearestCircle(e.getX(),e.getY());
}
```

So, this adds a Circle if you left click and removes the nearest one if you right click. Follow the code into MouseEventPanel for addCircle and removeNearestCircle.

Also look at the code in MyMousePositionListener – not much there yet! You are going to be getting mouseDragged() to work.

Task 3: You need to modify mouseDragged() of MyMousePositionListener so that it finds the nearest Circle and redraws it.

At the end of every step compile and test.

Step 1: Get ready for mouseDragged() to find the nearest Circle.

Start by adding this code to the mouseDragged() method:

```
    Circle thisOne=mousePane.findNearestCircle(e.getX(),e.getY());
```

Compile – it will fail. It needs a few things to compile.

- The first problem is that there is no mousePane variable in MyMousePositionListener. You need to pass in the MouseEventPanel (from the MouseEventFrame to the MyMousePositionListener's constructor) just like you pass in the StatusPanel and assign it to an instance variable called mousePane. This is so that you can even make the call above.
- The second problem is you need a method of MouseEventPanel called findNearestCircle(). To start with write a one line method that returns null:

```
public Circle findNearestCircle(int x, int y) {
            return null;
}
```

Your code should now compile from MouseEventFrame again once you have done these things.

Step 2: Now you have to actually write the proper code for findNearestCircle().

Look at the code for removeNearestCircle(). It is going to be like that but return a Circle. (This will involve writing a method of VectorsOfCircles called findNearestCircle() again very similar to removeNearestCircle() there (it would be good to refactor this code at some point to remove the duplication!))
Return null if there is no near Circle. Compile everything.

Step 3: Now back to mouseDragged. When you are dragging a Circle with the mouse you need to update that Circle on the MouseEventPanel (which means updating the underlying Circle).

So far you have:

```
    Circle thisOne=mousePane.findNearestCircle(e.getX(),e.getY());
```

Now add an if statement to update the coordinates in thisOne

```
    if (thisOne!=null)
                thisOne.update(e.getX(),e.getY());
```

Compile. It will fail on the last line because you have to write a method for the Circle class called update() that changes the Circle's coordinates to the new ones . Do that.

Task 4: Try running the program. Try dragging a circle. It won't move ☹
But then minimise the window then bring it back – Look the circle did move ☺!

This is because you are not repainting after updating the Circle (minimise does that for you – so it is often a useful thing to do.)

Look through your code to see how and where to call repaint().
Hint: The MouseEventPanel is the thing that needs repainting and you have a link to it you can use right there in MyMousePositionListener.

Task 5: Stop displaying the statusPanel

Just comment out the line in the MouseEventFrame constructor. Don't bother deleting.

Task 6: Make the circle be a different colour while it is being moved.

You will need to add a Color attribute of Circle that gets updated (to red perhaps) along with the position when you are dragging.

Hint: you will need to import java.awt.* to any class that uses Color. The way you refer to a color is as Color.red or Color.black and the way you display a Circle of some color is to say:

```
g.setColor(color);
g.fillOval((int)Math.round(x-r),(int)Math.round(y-r),
       (int)Math.round(2.0*r),(int)Math.round(2.0*r));
```

You'll also need to add the guts of the mouseReleased method in MouseMotionListener to find the nearest Circle and then update it back to black.
(It might have been nice to have just one class that implemented both Mouse Interfaces.)

Task 6: Glory

If you complete all of the above, add a menu that allows you to load and save the drawing (Serialize the Vector of Circles). There are no marks for this Task, but it is a great way to expand your skills by modifying code to add functionality.

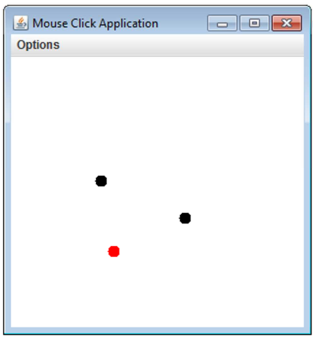
Assessment

You must demonstrate what you have to a demonstrator AND submit your folder via BlackBoard so we have a record of people's accomplishments:

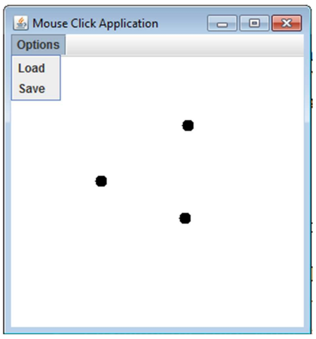
• Moving a circle around the screen minimise to see the move	(4%)
• Dragging properly so you can see it move as you drag	(2%)
• Changing the colour while you are dragging	(2%)
• When you release the circle goes back to black	(2%)

Finally – demonstrate this in your practical and upload to Blackboard so we have a record!

While moving :



While loading:



-END-