Lab 8 UI for the Guitar

Due:

See your lab Blackboard for Due date.

Marks:

- 7 marks for demonstration, and explanation of GuitarPanel code to the lab instructor.
- 3 marks for proper Java code on PlayGuitar,java and GuitarPanel.java

Recommended Reading:

Chapter 14 of Deitel and Deitel, Java How to Program (10th Edition)

Study Figure 14.34: PaintPanel.java, and Figure 14.35: Painter.java. You can find these files in the folder for this Lab.

Purpose:

To gain further knowledge of Java Swing, Event Handling, and to illustrate, in a basic way, one strategy for putting a User Interface on a program like our guitar simulation.

Demonstration:

To demonstrate your finished program, you will export your program as a jar file, and run it from the command line.

Description:

We will adapt an existing Java program that implements a mouse handler to act as primitive User Interface for our guitar simulation. Specifically, we'll use the painting program (PaintPanel.java and Painter.java) from the Deitel and Deitel textbook. That program draws a line as the mouse is dragged, and we'll adapt the program to pluck the strings of our guitar as the mouse is dragged over them.

Detailed Steps:

Study (Run) the PaintPanel.java and Painter.java files, to understand how that program uses an event handler to keep track of which black ovals should be drawn on the drawing area as the mouse is dragged. In our guitar UI, we **will not** need points or black ovals and we will not need to draw black ovals on the screen (repaint()), so the lines

```
 private int pointCount = 0; // count number of points

// array of 10000 java.awt.Point references
private Point[] points = new Point[ 10000 ];
```

should be removed. We **will** need to keep track of the previous position of the mouse, which, along with the current position of the mouse, will allow us to determine when the mouse has crossed the position of a string. The PaintPanel class shows how Java code can retrieve the current position of the mouse. The background will be changed to the image of a guitar, and because our program won't change that image, we don't need to explicitly repaint().

1. Start with a working Guitar simulation project (make a copy of the project) and Import the PaintPanel.java and Painter.java files to that project but change their names to GuitarPanel.java and PlayGuitar.java respectively.
2. An image file has been provided for you. Save it in the Eclipse project folder, so that it is *directly under* the src directory, and add the following as properties of GuitarPanel.java:
   private Image img = new ImageIcon(getClass().getResource("/guitar.jpg")).getImage();
   private int[] y = {0,408,381,356,327,300,272};
   The array of integers contains the y coordinates of the strings in the image which is 1024x680 pixels. We ignore y[0] as we ignored string 0. The strings in the image start at (roughly – good enough for our purposes) an x coordinate of 100. Remember that in Java the origin (0,0) is the top left corner, and y increases as you go down, x increases as you go right.

   You will also need to add a guitar property and instantiate it. Thereafter in your event handler you can pluck strings on your guitar.

3. Unlike the paint program, our program does not change what's displayed to the user as they use the program. Our program makes sounds instead, depending on where they drag the mouse. Locate the paintComponent method in GuitarPanel.java and replace the code that does the drawing of ovals with
   g.drawImage(img,0,0,null);
   This will display the image from Step 3 in the window.

4. Locate the mouseDragged method of the anonymous inner class, and change it so that when the mouse is dragged over a string according to the y array we added earlier, it will cause the corresponding string of the guitar to be plucked. Consider this boolean expression:
   previousY < y[6] && currentY >= y[6] || previousY > y[6] && currentY <= y[6]
   This expression is true when the mouse has crossed String 6 in either direction. You will need devise code to implement previousY and currentY (it's not as difficult as it may sound). The mouseDragged method is called many times as the user drags the mouse. Every time the mouseDragged method is invoked by a user dragging the mouse, you will need to update the previousY integer.

Notice that because we are not changing the image when we pluck, we do not need the call to repaint.

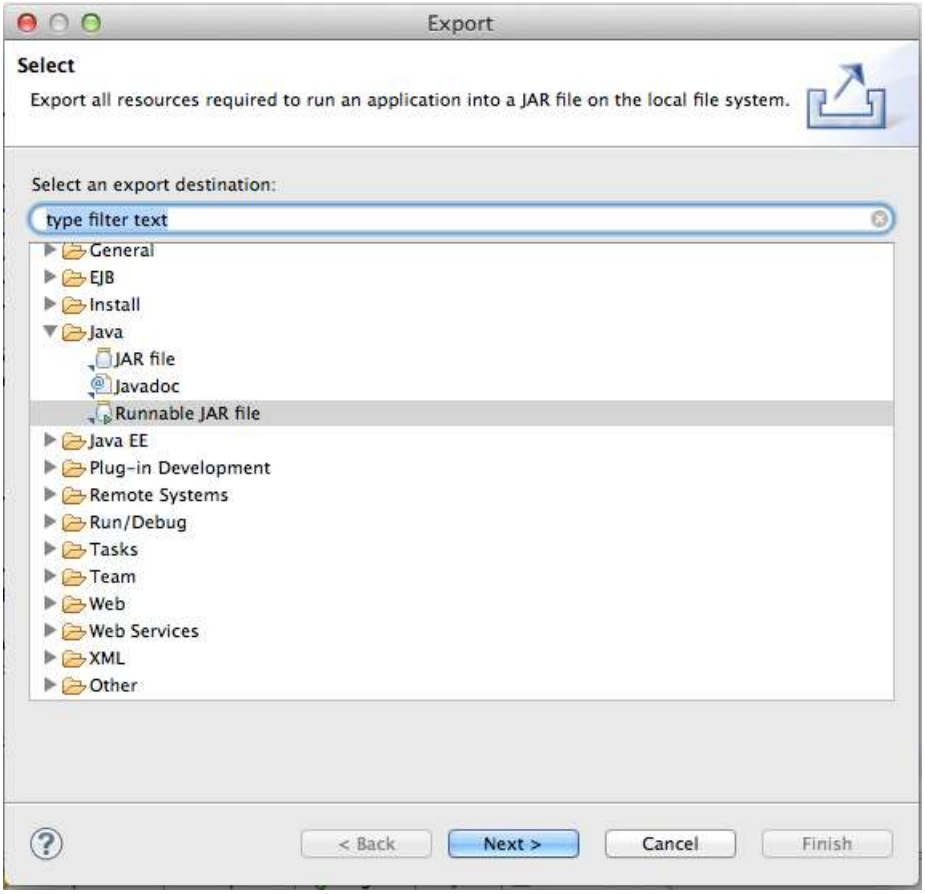
5. Make appropriate changes to PlayGuitar.java, including changing the dimension to 1024x680 to accommodate the size of our guitar picture.

6. Notice that when you move the mouse across any strings without dragging, and after that you then drag the mouse a little (not even over a string), the program may pluck the strings you previously moved across without dragging. The program considers the old position of the mouse to be on the other side of those strings, and new position looks as if the mouse was dragged across, when in fact it was only moved across. Fix this problem (hint: print currentY values and see the pattern when the problem happens).

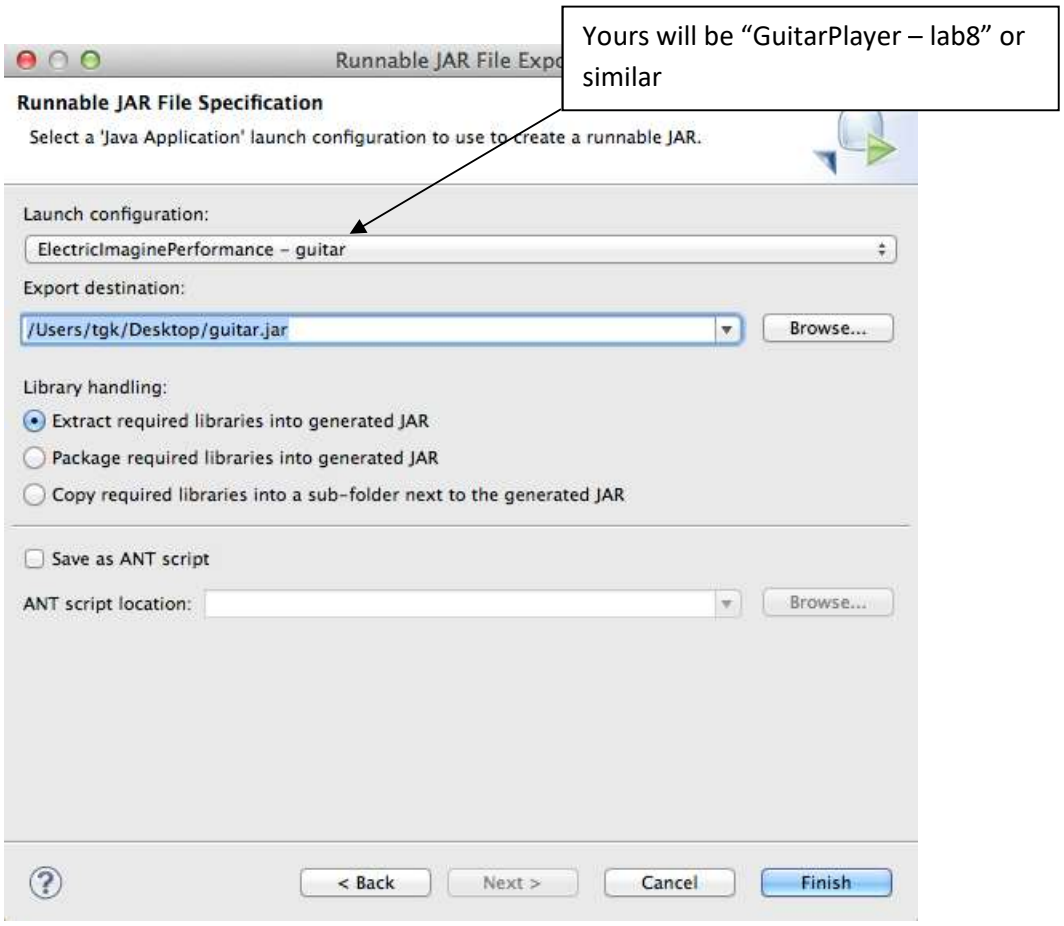
7. Make any other changes needed. For example, update the title of the window – it's not a Painting program anymore.

8. Demonstrate your program to your lab instructor. First export your program as a jar file:

    a. In Eclipse, right-click on the java file that contains the main method ("GuitarPlayer.java"), and select Java->Runnable JAR file, as shown in the screenshot below.



b. On the next window, click Finish:



c. Now you can run your program on the command line with

   java –jar guitar.jar

Submission

The submission process for this assignment is the same as before:

1. demonstrate your program to your lab instructor.
2. submit a zip archive of your deliverables folder: Lastname_Firstname_CST8132_Lab8.zip