

## Lab 10: Swing – Swing ...Mr. DJ, Mr. DJ ... (sing along now....)

0. This lab is due by **4 p.m. on Friday, July 8<sup>th</sup>**. You should be checked off by 4 p.m. Do not wait till 3:45 p.m. on Friday before trying to get checked off. You have all day today and tomorrow to work on this lab. Take advantage of it.
1. In this lab, you will explore the concepts and procedures involved in using swing to construct and use GUIs.
2. Create a class called `SwingEmployees`. In this class, you are to design a GUI that allows you to add employee names to an `ArrayList`, print the names in the `ArrayList` and clear the contents in the text display area.
3. The GUI consists of three buttons created using `JButtons` and one text area created using `JTextArea`. Implement the following functionalities for the buttons:

-‘*Add Employee*’ button

This should add employee names (`Strings`) to an `ArrayList`. This is very similar to your task in Lab 9, but this time, you are adding the names of the employees and not the employees themselves to the `ArrayList`. Clicking this button should prompt an input dialog box for the name and then add the name to the `ArrayList`.

-‘*Print Employee Names*’ button

This button should print all the names in the `ArrayList` in the `JTextArea` created.

-‘*Clear*’ button

This button should clear the text in the `JTextArea`.

**Note:** You need the following methods in the `SwingEmployee` class to implement the functionalities of the respective buttons:

```
addPressed();
printPressed();
clearPressed();
```

4. In order to fully implement the functionalities of the buttons above, you need to create a Listener class ‘*EmployeeListener*’. This class should initialize a `SwingEmployees` object as done in class yesterday. The class should have an `actionPerformed` method. Part of the class is written for you. This part includes the portion for the ‘*Add Employee*’ button. Complete the method for the ‘*Print Employee Names*’ and ‘*Clear*’ buttons.

```
import java.awt.event.*;
/**
 * This class listens to buttons that have been
 * implemented in the SwingEmployees application
 */
```

```

class EmployeeListener implements ActionListener {

    // FIELDS
    // private field that keeps a handle on the
    // SwingEmployee that is using this listener
    private SwingEmployee semp;

    // Constructor
    public EmployeeListener(SwingEmployee s) {
        semp = s;
    }

    // the actionPerformed method
    public void actionPerformed(ActionEvent e) {
        // get the identity of the button that was pressed
        String buttonPressed = e.getActionCommand();

        // dispatch to the right method call based on the
        // button pressed
        if (buttonPressed.equals("Add Employee")){
            // do the addition
            semp.addPressed();
            // the method in SwingEmployee is therefore
            // addPressed() for this button
        }else if (.....)

            //Finish up the method for the two other buttons
        }
    }
}

```

## 5. Checkoff

Compile and test your program on your own. Call a staff member to your computer and demonstrate a successful run of the code. We may modify your test code and quiz you on various aspects of your lab.