

Swing and MVC

To simplify these problems, I have made a class in my swing-mvc-exercises project called JListBase. Copy this class and WindowUtilities to your new project as a starting point. Make a subclass of JListBase and implement these methods:

- **makeJList.** Have this return the JList you want. This is the only method you need to override for problems 1, 2, 4, and 5. Note that the base code takes this return value and puts it into a protected instance variable call jList.
- **addStufftoListPanel.** Have this call “add” on the JPanel argument, if you want to add something extra to the top window. For problem 3, you will use this method to make a JButton, attach a listener, and then call listPanel.add(yourNewJButton).

Once you have done this, just make a “main” method that instantiates your class. E.g.:

```
public class JList1 extends JListBase {
    @Override
    protected JList makeJList() {
        // You have to write this part
    }

    public static void main(String[] args) {
        new JList1();
    }
}
```

1. Make an Employee class that stores a first name, last name, and salary. Make an array of Employee objects. Make a JList that shows the names. Make an array of Strings by looping-down the list of Employees and looking up the name. Display the Strings in a JList.
2. Repeat the previous problem, but this time don't make an array of Strings first. (Hint: give your class a toString method and note that you can pass an Object[] (e.g., the Employee[]) to the JList constructor.)
3. Add a push button that, when pressed, pops up a dialog box showing the salary of the currently selected name. Hint1: call listBox.getSelectedValue() and cast the result to Employee. Hint2: assuming “this” refers to the subclass of JListBase, pop up a dialog box with JOptionPane.showMessageDialog(this, someMessage).
4. Make a List<Employee>. Implement the ListModel interface in order to put the employees into a JList. (Hint: as in the class example, you can have empty bodies for addListDataListener and removeListDataListener).
5. **[If you are feeling inspired]** If the employee's salary is above \$50,000, display the name in red. Otherwise use blue. If you feel inspired, change the font for the two cases as well. (Hint: use getListCellRendererComponent and grab the original JLabel as in the class example.)