Homework 5, CS 220X D-term 2015
GUI for a Student Database

Due: Thursday, April 23 at 5pm

Outcomes

After successfully completing this assignment, you will…

- be able to read, understand, and modify a program written by someone else
- utilize the Java Swing classes to implement a GUI

Before Starting

Review Chapter 10 (events); read Chapter 19 (GUIs). Read this Java Tutorial on How to use Lists. Work on understanding as much of the tutorial as you can.

The Assignment

In Homework 4 you developed an application that modeled a student database, such as might be used in a Registrar's office. In this assignment, we'll continue with the idea of a student database, but the underlying model will be simpler than the one developed for Homework 4. The focus on this assignment is to provide a "view" (GUI) for our underlying "model" (student database). As you know, programming even a simple GUI means paying a lot of attention to detail. Given the limitations of time, you'll implement a GUI for a student database that keeps track only of each student's name and ID.

The Swing component *JTable* is good choice for displaying information in a database, such as we have with our ArrayList<Student>. However, programming with JTable is more advanced than we're ready for at this point. Instead, we'll display the database information in a simpler kind of component called a *JList*.

So, in summary, we'll simplify as follows:

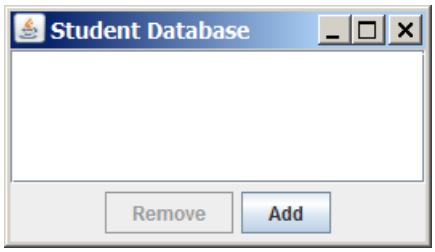
- the model will consist of an ArrayList<Student>, where the Student class maintains fields only for a last name, first name, and student ID. All fields in the Student class will be of type String (even the ID). The class that holds the ArrayList<Student> will need methods for adding a student and for removing a student.
- the view will present the information in the student database using a JList (even though a JTable would be a better choice in a real-world situation)

What you should do

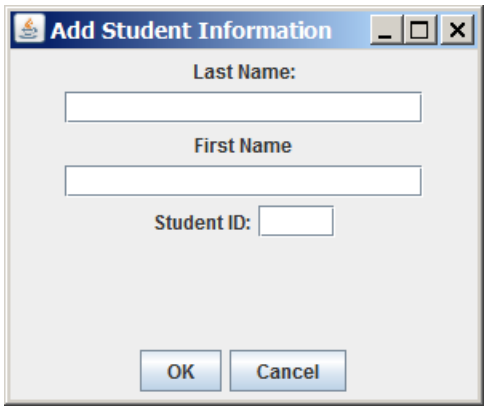
Despite the simplifications outlined above, there's still quite a bit of coding to do. The tutorial on How to Use Lists provides code for a ListDemo program. Download the code, run the program, and play with it a little bit. Try to understand as much of the program as you can. You may use the code in the ListDemo program as a starting point for this assignment. The copyright allows you download the code and modify it.

Here's how your Student Database GUI should behave:

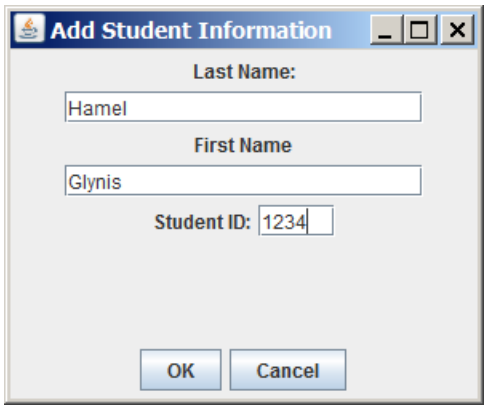
1. When the program starts up, a frame for the Student Database should be displayed. There should be two buttons, an Add button, which is enabled, and a Remove button, which is initially disabled (because there aren't any students to remove yet):



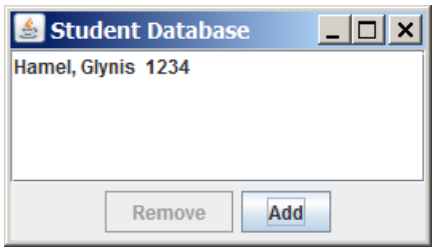
2. When the Add button is pressed, a new frame should appear. The "Add Student Information" frame provides text fields for a Student's last name, first name, and ID. It also provides two buttons (OK and Cancel):



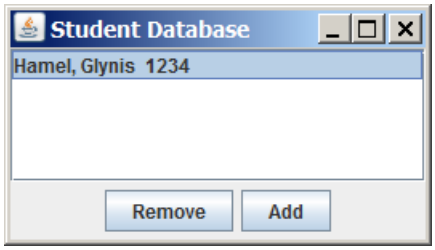
The user can enter the appropriate information for a new student, then press OK to have the student added to the database, or press Cancel.



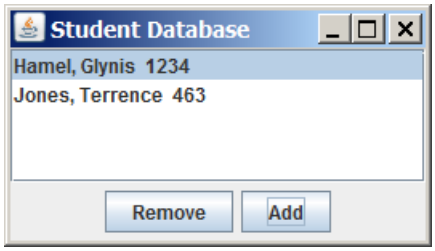
3. If Cancel is pressed, the window is disposed of without adding the student to the view (or to the database). If OK is pressed, a new student is created with the given information, the "Add Student Information" window is disposed of, and the main window now shows the new student's information:



4. Once an item in the list is selected (as shown below), the Remove button is enabled.

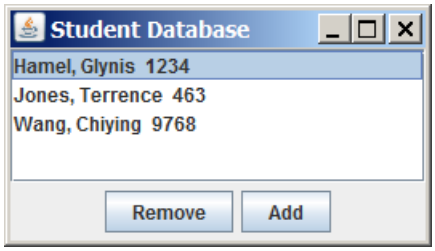


5. This is what the window looks like after a second student has been added.

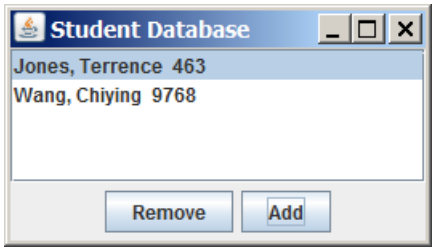


Whenever student information is added to / removed from the view (GUI), the underlying model should also be changed. So if the Student Database window shows information about two students named Glynis Hamel and Terrence Jones, then the ArrayList<Student> should also contain those two students. New Students will always be added to the end of the list. So no matter which item is currently selected in the Student Database window, adding new student information puts the new student at the end of the list.

6. Pressing Remove removes the selected item (and that student is also removed from the underlying ArrayList). Here's a list with several items, where Hamel is selected:

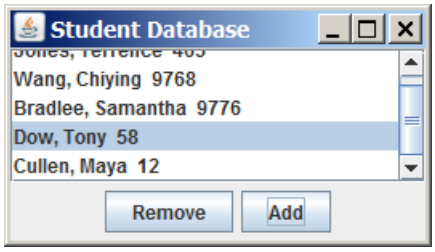


Pressing Remove results in the following display:



If all students are removed from the database, the Remove button should be disabled.

7. The list should be part of a scroll pane (as it is in the ListDemo program). Once the window is filled with student information, you should see the scroll bar appear:



That's it. Your GUI needs to provide Add/Remove capabilities as described above. The view and the model are kept in synch with each other, with respect to the number of items displayed and the number of items stored, and with respect to each item's position in the list. Much of what you need to do is already provided in the given ListDemo code.

Deliverables

Export your Eclipse project to a zip file and submit the zip file via web-based turnin. To export a project, go to File | Export… and choose Archive File from the General folder. Check off the entire project in the top left window and make sure the format is .zip and not .tar. Finally, give the archive file a name and click finish.

The name of the turnin project is Homework 5.

Programs submitted after 5pm on April 23 will be tagged as late, and will be subject to the late homework policy.

Grading

Your program must compile without errors in order to receive any credit. It is suggested that before you submit your program, you should compile it one more time to make sure that it compiles correctly.