PROG24178

Assignment #3

(Due Date: See SLATE)

Georg Feil / Winter 2015

Android Student Database (20 marks / 4%)

In this assignment you'll create an Android application that allows the user to enter and sort a student database. You'll be creating your first Android GUI, and get some experience using abstract classes and interfaces. You program will be able to run on the Android emulator or on an Android phone/tablet.

Student Class

Implement a class called Student to store student information. Your class should be a subclass of the abstract class Person provided with this assignment. In addition to the inherited fields and methods, include the following fields/methods:

- A field to store the student's GPA
- An implementation for the abstract method currentStatus. This could simply return the string "Student".
- A toString method that includes the GPA and follows the format of Person.toString. Hint: You could call the superclass to do most of the work.
- Any required getters or setters

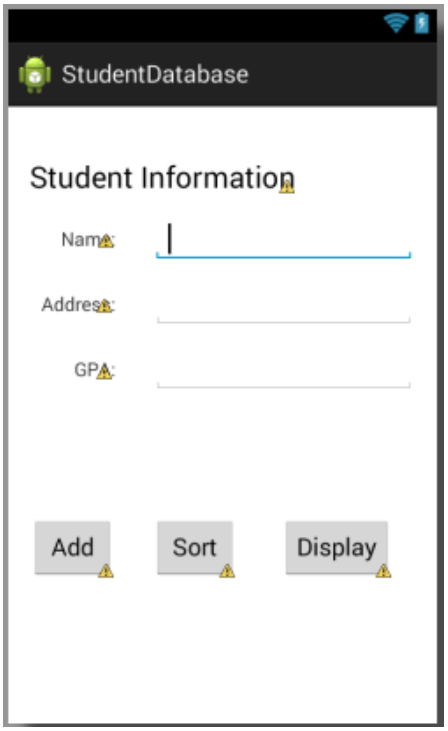
You'll be using your Student class in a list of students created by your GUI.

Android GUI

Create an Android application which allows you to enter student information. Your GUI should have one activity with the following components:

- Text fields to enter the student's name, address, and GPA
- Buttons to add a new student, sort the list of students, and display the list of students.
- Any required labels (TextView) or other items to make your GUI look good and be easy to use.

The following image shows a possible GUI design, but this is just a fairly boring example – design your GUI in the way you think is best! Your GUI class should be a subclass of Activity (the Android tools will automatically create this for you, follow the project creation steps demonstrated in class).



Use an ArrayList of Student objects to store your list of students. The Add button adds a new student to the end of the list, the sort button sorts your list of students by name, and the display button shows a simple popup dialog containing a list of all students' information (student info is provided by the toString method).

For text entry fields and buttons (but not simple labels) you'll have to create field variables and associate them with the widget as in the following example:

(At the top of the class) private EditText nameField;

(In the onCreate method) nameField = (EditText)findViewById(*R.id.editText1*);

You can find the ID information (e.g. *R.id.editText1*) by right-clicking on the text field or button in the GUI editor graphical view and selecting "Edit ID" (look beside where it says "Java"). You may want to change the ID name to make more sense than the one that's automatically generated.

Popup Dialog

Here is an example of how to show a popup dialog from a method of your activity. The message string could be a long string containing the list of student information.

```
AlertDialog dlg = new AlertDialog.Builder(this).create();
dlg.setTitle("Title");
dlg.setMessage("Message");
dlg.setButton(AlertDialog.BUTTON_POSITIVE, "OK", (OnClickListener)null);
dlg.show();
```

Sorting Your ArrayList

To sort your list of students you can use the Collections.sort method from the Java library. For this to work you must implement the Comparable interface for the objects being sorted. A good place to put the Comparable interface implementation is in the Person class, since we are sorting by name (a field of the Person class). Update the class I provided.

Bonus (1 mark)

Add the capability to store “fees paid” status (yes/no) for each student. One way to show this on the GUI might be a checkbox. Update your currentStatus method to include whether fees have been paid, e.g. "Student – fees not paid"

Assignment Submission

Your submission must follow all the submission requirements outlined below.

Do your work individually. This is not a group assignment. Submit your assignment using SLATE (click on the Dropbox tab, then click on the correct assignment link). **ZIP and attach your entire Eclipse project** (do not use RAR).

Your assignment must be submitted before the due date/time. Any assignments submitted after this time are considered late. See our class plan on SLATE for my rules governing late assignments. If your assignment is more than a week late I’ll be happy to read it and give feedback but you will receive a mark of zero.

Evaluation

Your submission will be evaluated based on the following criteria:

Efficient Code: Program doesn’t use too much repetitive code (e.g. uses methods instead of copy/pasting code). Program uses variables where and only when necessary; program doesn't define variables that are never used, nor does it use too

many variables for unnecessary tasks. Program logic is written concisely and is not cluttered with unnecessary code.

Functionality: Program functions according to specifications.

Programming Style: Proper indentation and spacing, use of comments; all identifiers are descriptive and follow our coding standards; variables are defined with appropriate types and converted when required.

External Documentation (if any): Good organization and clarity of written communication.

Other: All instructions regarding submissions and program specifications have been followed; submission was completed as requested in a timely fashion. Techniques discussed in class have been used as appropriate.