Due: Tuesday, 3 October 2023, 4:00 PM

Learning Objectives

- Solve a problem by constructing a simple, interactive application using Android and Java.
- Document an object-oriented design in Unified Modeling Language.

Problem Description

Consider the situation of someone who needs to manage their monthly expenses. What is needed is a simple, attractive, intuitive, mobile app to track this information. Let us call this app: ExpBook.

Specifically, each expense has a record with the following fields:

- name (textual, up to 15 characters)
- month started (presented yyyy-mm format)
- monthly charge (non-negative currency value, in Canadian dollars)
- comment (textual, up to 20 characters)

Only the comment field might be left blank for an expense. The app should allow the user to:

- show a list of current monthly expenses, along with a summary of the total monthly charge for them
- add a new expense (which always appends to the bottom end of the list)
- view and edit the details of an existing expense
- delete an existing expense

The list need not show all the information for an expense if space is limited. Minimally, each record in the list should show the name, date, and monthly charge.

The app must assist the user in proper data entry. For example, use appropriate user interface controls to enforce particular data types and prevent invalid input from being entered.

For this assignment, the app does not need to be persistent or save between runs.

Use your campus computing ID in the app name. Specifically, the app name must show up as `*YOURCCID*-ExpBook` (e.g., `kennyw-ExpBook`).

Deliverables

1. **Code Base:**
   Your complete source code and compiled binary, implementing the working app and its user interface, will be inspected and run by the TA. The Android Studio project and APK (Android Package Kit) binary must be included in the submission. Each class must contain comments describing its purpose, design rationale, and any outstanding issues.
2. **Video:**
   The video is a demonstration of the app. The video file must be included in the submission. The video is meant to show that the demonstration actions below actually work. No audio is needed. Maximum duration is 3 minutes. Focus on just the screen of the app, not your whole desktop. For visual clarity, do not use a handheld camera.
3. **System Documentation:**
   Describe the structure of your app's object-oriented design using UML class diagram(s), saved as non-lossy image file(s). Focus on the most important classes that you designed and implemented. Add notes to describe the main responsibilities of these classes.

Demonstration Actions

1. Open the app from the launcher.
2. Show the list of expenses, with none so far. (This should be the initial screen.)
3. Add a monthly expense with name: Phone, month: 2023-09, monthly charge: 40.00, no comment.
4. Show the list and summary (total monthly charge), with the one expense.
5. View/edit the Phone expense with monthly charge: 42.00, comment: "text/data plan".
6. Show the list and summary, with the updated expense.
7. Add a monthly expense with name: Stream, month: 2023-08, monthly charge: 8.99, comment: "basic".
8. Show the list and summary, with the two expenses.
9. Add a monthly expense with name: Rent, month: 2023-09, monthly charge: 1100.00, comment: "no parking".
10. Show the list and summary, with the three expenses.

?

11. View/edit the Rent expense to have monthly charge 1150.00, comment: "with parking".
12. Show the list and updated summary.
13. Delete the Stream expense.
14. Show the list, with the two remaining expenses.
15. View the details of the Phone expense.
16. View the details of the Rent expense.
17. Delete the Phone expense.
18. Delete the Rent expense.

Hints

This is a description of the core functionality. Often, problem statements from users lack details. As you are prototyping a design, you may uncover other behaviors that have not been specified, but make sense in the context and intent of the application. For example, think about how someone might effectively use your application. You should decide what functions your design will need, based on the given problem description and valid assumptions, in discussion with your users (the TAs and instructor). You should consider asking the customer (the instructor) what they want to see.

While you may discuss your design with other students, the code and documentation must be your own work. Code from publicly available sources may be used within reason (i.e., does not essentially solve major sections of the app) and only if their licenses permit so. Always fully cite to give proper credit to the original developers in the source code and in the system documentation. For example, in citing a work, at least state: from whom, the date of publication, license, and URL. Do what is required by its license. When citing generative AI, at least state the tool used, date of generation, relevant prompt(s), and explain your revisions. You do not need to cite standard language or API references.

The TAs will be inspecting your code, so you should follow Java coding conventions, have useful comments, and "major" commented-out experiments should be cleaned up so that the code is readable.

Do not skimp on the UML class diagrams in the system documentation. For neatness and readability, diagrams should be created or drawn using a vector graphics editing tool, exported in a common, non-lossy graphics format, and employ a readable layout.

Besides addressing the problem correctly, your software design will be evaluated on its proper use of object-oriented design concepts, such as abstraction, separation of concerns, and information hiding.

Losing Marks

You may lose marks for any of the following:

- files not in properly named subdirectories
- missing compiled binary APK file for the app
- cannot run the app after install
- cannot distinguish CCID from the app name
- cannot view files without specialized tools
- lossy compression used in image file(s) for UML (e.g., JPEG)
- inadequate or improper citations
- missing requirements

Submission Procedure

Create an assignment directory called `YOURCCID-ExpBook` (e.g., `kennyw-ExpBook`), and within it have three subdirectories: `code`, `video`, `doc`.

Your entire Android Studio project directory structure goes within `code`. The compiled binary APK file `app-debug.apk` must be built and included. The video file goes in the `video` subdirectory. The UML documentation goes in the `doc` subdirectory.

Zip the assignment directory and upload to eClass as a single `.zip` file.

The TA will test your app from the submitted code and provided APK file. Please make sure all the required files are included to rebuild the app if needed.

The app name must show up as `YOURCCID-ExpBook` (e.g., `kennyw-ExpBook`), so that it can be easily distinguished from other submissions.

Submission status

Submission status	Submitted for grading
Grading status	Not graded
Time remaining	Assignment was submitted 6 mins 29 secs early
Last modified	Tuesday, 3 October 2023, 3:53 PM
File submissions	mhossai6-ExpBook.zip ✚ 3 October 2023, 3:53 PM Export to portfolio
Submission comments	▸ Comments (0)

Feedback

Grade	7.00 / 8.00
Graded on	Monday, 23 October 2023, 10:11 AM