SENG 440: Topics in Mobile Computing

Project 1

1 Overview

Your responsibility in this project is to get acquainted with the Android user interface system and its event-driven programming model. You will do this by writing an app that makes use of the features that we discuss in term 1 of the semester. The design and purpose of the app is mostly unspecified. You are encouraged to make something that interests you or a community that you care about.

2 Requirements

The requirements for this project fall into two categories: mandatory requirements and grade-bearing requirements. For your project to be eligible for marking, you must satisfy all of the mandatory requirements by the end of the due date—which is the last Friday of term 1. If these requirements are met, the grade you receive will be calculated based on how many of the grade-bearing requirements you meet.

2.1 Mandatory Requirements

Complete the following mandatory requirements by the due date:

- ☐ Create an app that has a particular and meaningful purpose for some audience. It should not just be a sandbox app in which you cobble together disconnected features.
- ☐ Use Kotlin for the program logic, not Java. Create your user interface declaratively with XML, not imperatively through Kotlin.
- ☐ Maintain and submit your project in a Git repository on the departmental GitLab server.
- ☐ Document your app with a post on Slack. In your post, tell us a story about the app's purpose and development process. Include screenshots. At the end of your post, enumerate which of the grade-bearing requirements you believe you have met and how. Be brief but specific in your enumeration.

2.2 Grade-bearing Requirements

The more grade-bearing requirements you complete, the better your grade. There are 20 Pakipaki available in this project. Each of the following requirements is worth 1 Pakipaki:

- ☐ Compose your app out of at least three screens, where a screen is either an `Activity` spawned via an explicit `Intent`, or a `Fragment` with a fullscreen layout.
- ☐ Invoke at least one other app on the system via an implicit `Intent`.
- ☐ Include a list view, preferably using `RecyclerView`.
- ☐ Compose your list view using a custom adapter whose view creation method uses a custom layout.
- ☐ Include at least five different kinds of widgets besides a list view (buttons, textboxes, checkboxes, labels, and so on) in the user interface, and handle their interactions with event listeners.
- ☐ Use at least two different layout groups (e.g., `ConstraintLayout` and `LinearLayout`) to organize your widgets.
- ☐ Support both landscape and portrait orientations in all views. In other words, all widgets should be able to be made fully visible in either orientation. This may happen automatically given your layout manager, or you may use a `ScrollView`, or you may specify two separate layouts.
- ☐ Provide separate landscape and portrait layout resources for at least one of the views.
- ☐ Use string resources for all static text on the user interface.
- ☐ Provide default definitions for your string resources in English. Provide definitions for one other language. (Use your favorite online translator if necessary.)
- ☐ Use a `Toast` message or dialog to alert or interact with the user.
- ☐ Use an `AsyncTask` to trigger some computation without blocking the user interface.
- ☐ Share a plan for your app before Saturday of week 2 in a post on `#project1` in Slack—before you've written any code or created any layouts. Include hand-drawn sketches or wireframes.
- ☐ Share an update of your work before Saturday of week 3 in a post on `#project1` in Slack. Include screenshots.
- ☐ Share an update of your work before Saturday of week 4 in a post on `#project1` in Slack. Include screenshots.
- ☐ Share an update of your work before Saturday of week 5 in a post on `#project1` in Slack. Include screenshots.
- ☐ Share an update of your work before Saturday of week 6 in a post on `#project1` in Slack. Include screenshots.

- ☐ Incorporate an animation into your UI, preferably one specified in XML. We will not discuss these in lecture.
- ☐ Incorporate some other Android feature not mentioned above into your app.
- ☐ Incorporate some other Android feature not mentioned above into your app.

If you have questions about any of these, communicate with your instructor. Do so early and often. Please share your questions in `#general`.

3 Submission

To submit your homework for grading, complete the following before the due date:

- ☐ Commit and push your app to your Git repository.
- ☐ Verify that the push succeeded by visiting your repository via your provider's web interface.
- ☐ Publish your post mortem on Slack in the `#project1` channel.