

Lappeenranta teknillinen yliopisto
School of Business and Management

Software Development Skills

Antti Keronen 001855050

LEARNING DIARY, MOBILE MODULE

LEARNING DIARY

Declaration of AI: Chatgpt was used to figure out error messages by pasting the error message to Chatgpt and seeing what is wrong with the code.

17.2.2026

Today I started the course. I think this is the final of these so-called self-service courses. I have completed Full-stack, Frontend and DevOps courses already. I also have completed the “Olio-ohjelmointi” course which I think is similar to this course. The objective is to deepen my understanding of Java (Just a guess the course is about Java didn’t check yet 😊). I’ll start by going through what I have to install to my laptop since just yesterday I emptied my laptop completely.

Android studio and Java installed. My studio looks a bit different but it doesn’t matter the video is just 8 years old. I watched all three of the videos

The intro video explained how to set up a new Android Studio project and understand the basic structure of an Android application. The video shows how Android Studio can be used to create a simple app and use debugging tools to find and fix errors. The second video was about the core elements of Android development. It explains what activities are and how they represent different screens in an application. It also covers intents and how they are used for communication between components, intent services for handling background tasks, and broadcast receivers that respond to system or application events. In the last and the longest video, it is explained how to use lists to display lists of data in an application. The video also explains how to create custom layout components to design the user interface and how to incorporate images into the app using images to improve the visual appearance.

My app will have

1. Functionality with components (ex. buttons, text fields, togglers etc.)
2. Multiple views
3. A component to display information (ex. listView)

I will create a simple workout listing app, where you can add the workout and the time amount used. You can also toggle a switch whether the workout is done or not. There has to be two views so the first one will be basically a plank screen with add a workout and then the second screen will be the view where you add the workout. The workout will be listed in the plank screen.

Since the videos were short and not a lot was covered, I'm afraid this won't be learning diary. More like just an explanation of what I did.

18.2.2026

The coding started. In the "Olio-ohjelmointi" course I have created similar apps so I'll try to use a similar approach.

Don't know if I created the project with correct attributes, but it said that the app will work on 99,2% of devices so maybe I'm fine. I chose the Groovy DSL instead of Kotlin since I'm using Java.

The Android Studio created it anyways with Kotlin+Java folder but there are no Kotlin files so it doesn't matter I'll just leave everything as they are.

I created Java files:

MainActivity.java which is the main screen of the app. It shows the list of workouts and has the button to open the add workout screen.

AddWorkoutActivity.java which is the second screen. It lets the user enter workout name, minutes and choose whether it is done. When saved, it stores the workout and goes back to the main screen.

Workout.java a simple data class. It represents one workout with name, minutes, and done status.

WorkoutStore.java it handles saving and loading workouts using SharedPreferences. It stores the workouts so they stay even after closing the app.

WorkoutAdapter.java connects the workout data to the ListView. It controls how each workout row looks and updates the done switch.

After those I had to create the .xml files:

activity_main.xml has the layout for the main screen. Contains the Add button and the ListView.

activity_add_workout.xml has layout for the add workout screen. Contains text fields, switch, and save button.

item_workout.xml has layout for a single row inside the ListView. Contains image, workout text, and done switch.

AndroidManifest.xml registers your activities and defines which activity starts when the app launches.

strings.xml stores all text used in the app so text is not hardcoded.

Doing this project actually reminded me a lot of how Android development works, since it's already been two years since Olio-ohjelmointi course. At first it felt very basic, but while building it I started remembering how everything connects together. This project refreshed my knowledge of how Java works in practice. Even though this app was simple, the structure was still the same as in more complex apps.

This project didn't necessarily teach completely new concepts, but it refreshed my knowledge and reinforced the fundamentals. It reminded me that I still have a very long road to go to be a successful Java coder.

Last but not least I added a delete button for the workouts since I added some inaudible content and realized I cannot delete it.

If using Android Studio virtual device while evaluating. At least with these specs the app works: Medium Phone: Android 7.0 Nougat x86

Medium Phone API 24