Software development skills: Mobile – Learning Diary

Marko Jutila

Setting up the environment - 10.08.2023

I started to work on the course by installing the proper environment for developing an Android application. The IDE I chose was JetBrains IDEA, which is a paid IDE, but has a free license for student use. I have used Android Studio before in a different course, so I wanted to try out a new environment for a change. I have also heard that JetBrains IDE:s are good and IDEA should have all the features that Android Studio has and should be even better. At first I had a small problem with creating a new project and empty activity, as the IDE defaulted to using Kotlin as the main language. With a bit of tinkering I managed to create an empty project using Java, and was ready to start foing through the first module.

The computer I'm using is running Archlinux. I chose to use the default API, JDK and JRE given by IDEA. The minimum API was 31, JDK (Java Development Kit) was Java OpenJDK 17. For running the app I created I used a Nexus S using API 33. The empty activity was running fine, and I was ready to go through the module 1 tutorial.

Module 1 - 13.08.2023

I started going through the tutorial for the basics of the project structure. The basic application consists of activities, each of which have a separate layout file in XML-format. The layout file can be modified to add elements, such as plain text, buttons, sliders etc., to the activity. The activity file is mainly responsible for running application logic, such as calculations, fetching data online, etc. AndroidManifest.xml file has some basic settings and information about the app. Such as the target API level, themes, icons, label, imports and the default activity, which is launched when the app is first opened.

In the res (resources) folder there is also the values folder, which is used to store the values used in the app, such as the strings that are shown in the app, color keys and hexadecimal color code values for the colors etc. A good practice is to store all the text in the application to these values files and load them to the layout xml:s through them. Avoiding hardcoded text like this makes it easier to keep track of all the strings and makes the localization much easier in future. For example, if we wanted to add Swedish localization to the app, we would only need to make the application load a different strings.xml, which would contain the Swedish translation for all the text. This is much easier than manually editing every activity's layout's strings.

Next the tutorial goes through basic manipulation of elements in the acitivity layout file. Elements, such as TextView, EditText and button can be easily added to the activity by dragging from the sidebar, and the attributes can be edited from a different sidebar. All the elements have different attributes that give the developer the ability to manipulate their proterties, such as text, input format, and size. Also, the location of the element on the screen can be changed by adding constraints to the element. Constraints basically mean that the element is positioned in reference to other elements, or the sides of the display.

This way the element is placed correctly on different kinds of devices, though in some cases modifications have to be made for portrait and landscape modes, or tablet sized devices. The constraint lengths can be modified, and the element can be offset to left or right to create an application, that looks neat.

After adding the basic input fields for the user, the addition button and the results TextView I coded the addition logic to the MainActivity.java file. As instructed in the tutorial, I found the correct elements from the layout file using findViewByID() method and was able to add a click listener to it. I made the OnClick method to addition the 2 input numbers and show the result in the results view. I got this working with no problems, and my app ranjust as it was supposed to. This demonstrated how the activity file is responsible for all the programming logic, and layout file is used to put the elements in their correct locations with the attributes. They communicate with each other together they make the app interactable.

The final thing in the tutorial was debugging. This was not a new thing for me. Using breakpoints and stepping over the code one line at a time, or stopping at different parts of the code to look at the variables is used to look for what the program is doing wrong in some parts of the code. A handy tool for the developer to search for faults in their code.

Module 2 - 13.08.2023

The second module goes through launching another acitivty and intents. In Android development Intent is an action requested that the device should try to perform, in this exmple it's opening the browser with the given address. The tutorial also goes a bit more in depth about the values folder and constraints, but I didn't learn anythign new from that.

Intents can also have extras, which are basically key value pairs to pass information from the activity that gives the intent to the acitivity that gets the intent. This allows for the activities to pass data to other activities. The Intents are handled in IntentService, but the tutorial didn't really go in depth about further uses for this. The tutorial also mentions BroadcastReceivers, which receives intents from sendBroadcasts mehtod, usually indicating some work has been completed. This wasn't really elaborated further either.

I added the 2 new buttons and created a new secondActivity as in the tutorial, and got the app working as it should. Pressing the first button opens the secondActivity, and sends a "HELLO WORLD!" string to the second activity through the intent extras. The second activity sends an intent to the device to open URI http://www.google.com. I had a slight problem with this, as the tutorial is 6 years old and I was using a newer Android API. After API 30 the application cannot directly see or interact with most external packages without requesting allowance. I resolved this by adding a <queries> element to the AndroidManifest, which sets up the request for the application intents to open a link in browser. Other than that, I didn't really have any problems and got through this tutorial quite quickly.

Module 3 - 14.08.2023

The third and final module is mostly about the ListView and ImageView elements. In the tutorial we create a simple app which populates a ListView element with given string-array information about fruits, with their prices and descriptions. ListView element is an element that can be used to list items on the screen in a list. The items can be customized as their own xml-file to give each list item for example images, multiple texts and all the other elements. In this tutorial we create a simple layout file for a list item called my_listview_detail, which consists of 3 different plain text fields. The data for the text fields is fetched from string arrays in the project's string resources.

An ItemAdapter is used to get all the correct data for the listview items. ItemAdapter is an adapter that extends from BaseAdapter. The ItemAdapter has the necessary methods to get the correct data on the product from the string arrays using indexes. For example, all the info on tomato are in index 2 of all the string arrays. After getting the proper data on an item, the ItemAdapter uses LayoutInflater service to inflate the ListView element with the data.

The app also introduces the option to click on a product and switch to a different activity called DetailActivity. The DetailActivity uses Intent extras from the MainActivity to get the data on what item was clicked, and then displays the corresponding image on the screen using switch case with the item index that was clicked. The DetailActivity also scales the image correctly in ratio to the screen using a BitmapFactory object, and also sets some other options to make the shown image look neat.

The tutorial Is a bit outdated, from my understanding ListView element is nowadays deprecated. When developing for never API:s, the more modern and flexible RecyclerView should be used. I didn't want to take this approach this time, as tutorial had all the ready material for using ListView, but I'm going to try to use the more modern approaches when creating my project for the course.

Project

Work in progress, requesting extra time to complete