Lappeenrannan teknillinen yliopisto
School of Business and Management

Sofware Development Skills Mobile, Online course

Václav Tůma, 002528535

LEARNING DIARY, MOBILE MODULE

LEARNING DIARY

27.11.2024

I have enrolled for this course because I was interested in the basics of mobile application development. As I already finished the front-end course, I was familiar with the course's style. I just looked at each part of this course to check if it is the same style or differs. I was already familiar with Git from the introduction of web programming and front-end courses.

28.11.2024

What I didn't have was an Android Studio. I have never used it before, I downloaded the program, walked through the installation process, and created a new repository for the video learning task. I started by watching the introductory lecture video on Android Studio and its basic setup. Since I didn't have Java installed, I downloaded it as well. However, I couldn't find Java SDK on the website. I tried without it as the video is old and maybe this part is no longer necessary.

07.12.2024

Tutorial Part1

After basic installation I began an example project with the video tutorial and took exactly the same steps as in the video to create my first project. The only thing that didn't show there was the version of android. When I selected **Empty activity** the Activity and Layout name were missing. After setup it didn't work and there was an error. After some time, it downloaded some other parts and errors were solved. Starting the virtual device took at least 7 minutes and that's why I thought that installation was incorrect, but luckily everything was just fine.

Firstly, I discovered **.xml** and **.java** files. In the top on the left side there is a menu where I can find Android, packages and much more, I think this is important to remember.

But as I want to be in Android the next was discovery what can I find here. Java is where the java code will be. There are 3 packages. The first is official and then testing versions. **Resources** for pictures and more. The most important thing mentioned in the video is **layout**, but I didn't have this.

In part1 we wanted to improve **MainActivity.java** and **Activity_main.xml**, but I still had a problem with finding or creating Activity_main.xml. After searching on google I found a

solution (1). The problem was that in the current version of Android Studio it is not possible to use Empty Activity, but it is necessary to use **Empty Views Activity**.

Next thing I made was setting preference of auto import in **preferences menu**: **Optimize import on the fly** and **add unambiguous imports on the fly**. Then I changed my view to **designer**. First, we started with deleting the hello world text in the center to have truly blank design. Started with two text boxes that can be found in the Palette menu and select **Plain text**. Really interesting feature is dragging blue button from right and then from left and that center my text bar, and I can just make offset in settings. Here it is possible to change from text to **number** which will help later with input numbers by the user. The same was done with the second number input. Now it was necessary to add button and that is located in **Common.** Change of buttons Id and the text was the next step. The last part of this first demo was adding TextView for the result also from the Common tab. In the default the text is really small, and it is good to change **text size** to 48sp (screen independent pixels), new unit for me. Now I learned how to start the app. For this part it was necessary to create a virtual device that I will use. I have chosen Pixel 9 Pro with android 15. It was the first try it took a while. Everything worked, but it was only visual and no other functionality. For adding the functionality, the MainActivity.java was necessary to improve. But before improving Java file I made my first commit to the GIT from Android Studio. As all buttons and inputs now have their independent ID it is necessary to use **findViewById**. R is also important, and it stands for **resource object**. I created a variable addBtn with type of Button. setOnClickListener Is used for remembering that each time the button is clicked do this and in this part we say what each part must do. After that converting the string to int and sum these number in res and display the result. But res is int not string and fastest way is to add “”. Then I learned how to debug in Android Studio (Run > debug). As I already have a lot of experience with debugging, this was not new for me, only the way to do that in Android Studio was new for me.

Tutorial Part2

In part2 the main goal was to create an app that has **multiple screens (activities)**. Here I learned that android development has 2 main parts. **Activity** is the area that displays something (mainly buttons), and **Intent** is activity, that the device tries to perform (tap or swipe). Because the showcase was different from the first one and I wanted to have part1 available I created a new project for tutorial Part2 and created new Git repository for this

project. A new thing I learned here was why there is drawable. It is used as a folder where I can drag and drop images, and Android Studio will remember what I put there. The option for center vertically in parent was also new for me. Because I wanted to add a second activity it was created by adding new empty view activity by clicking the first file in java – new – activity. The first part of the FundById and setOnClickListener part was the same as in part1. The new part was usage of startActivity which starts my created intent. **putExtra** put info to the bundle and send it to other activity, it is done with key value pairs, mostly there are used package name. First is a key name and second is actual value.

08.12.2024

After finishing the function of the first button next was creation of functionality for **google button** that will move user outside this app to the browser. The first part was the same as with the first button with findViewById and setOnClickListener. Then it was important to make variable with the address where I want to go. But here is used different intent ACTION_VIEW, this will help to go outside of the application and launch that and I need that webAddress here. It will send broadcast, and it says: is there any app, that can do this? But if there is no app that can do it if is necessary to have **if**. If gotoGoogle can resolve activity that handle this request then get packet manager and make sure, that it is null, only if this if statement if true, then it can start the activity. After coding all important parts I tried if it works, but it didn't. After clicking the button nothing happened. In one forum they said that it is important to have
<uses-permission android:name="android.permission.INTERNET" />, but it didn't help. It looks like this emulator can't find say, that it has preinstalled browser and returned null. When I tried different page then google it worked, so maybe was a problem here.

Tutorial Part3

Again, I started a new project with a new repository, because I might want to use this part later. A new thing I learned here was usage of container instead of just button or text field. ListView from the video was replaced by RecycleView. I created an ID as myRecycleView. Another new part was usage of Strings in Values category, because it is good to have all string (items of my list) in one place. Here I created 3 arrays with names, prices and description by using string-array. Because we need to present a list somewhere I created

layout file that will merge two created files together. Layout – new – layout resource file. I named this file as my_listview_detail. After creating the file, it was necessary to go to the design and change the text size. After trying if the basic works it crashed. The problem was in activity_main where I didn't have ListView and it crashed the app every time. After correcting this error, it worked.

09.12.2024

Because my_list_detail was just for a test I deleted it and created it again for prices and description. Name I left the same, but this time I changed to relative layout instead of text view layout. After creating the file, I added TextView in graphical editor. I placed it in a relative position and changed the text size. Next was the addition of another text view and aligned with the first one. This wasn't anything new as it was in the first 2 tutorials and the last added text view was for the price. Back in the MainActivity it was important to create arrays for the prices and descriptions. New was for me creation of new java class in top package in java folder. This was created as **ItemAdapter.** A really interesting feature was that if it needs abstract methods, it is possible just click left and it will create them for me without typing. After I created layout adapter here. I had an error here, but the solution was easy as I didn't have one ID right in my_listview_detail. String name has item's current position. Then put information to the name, description and price TextView with using setText. A new ItemAdapter was created in MainActivity, but then I had to use it, because this was only referencing. I wanted to use it in myListView. After the first try the app crashed again. The problem was as this time again with ID, where two different things had one ID, this was easy to fix. What I noticed was the weird look of the price part, where all prices were in right corner, this was caused by attachement to the left side of name and as the first name was large it pushed prices to the right. I detached prices from the name and attached it to the right screen side that fixed this bug.

But now it was uninteractive and after clicking it didn't do anything. To create this functionality, I made a second screen that will show images. New – activity – Empty Views Activity. A new thing for me was adding images. In contrast with the video, it was on a different menu. Common – ImageView. I have chosen the picture as background as this also changed from older version shown in video.

Back In mainActivity I created setOnItemClickListener where I say what I want to happen after clicking the item. New was adding of images to folder here. It is done by drawable

folder where I had to choose **Find in Files** and copied pictures to the folder. After this I created in detailActivity private method which will return image depending on given index. Then usage of BitmapFactory given access to java library that scales images. It again ended with an error.

10.12.2024

After a long time, I found out that in file that I closed (activity_details) the ID didn't save, so the same problem as above there were 2 same IDs and when the program tried to load this screen these two IDs lead to crash of the app. In this moment I finished given materials.

Project

I decided to continue with the same style as in the front-end course, so transport in the Czech Republic. I started with choice of the photos for each railway carrier and gave these photos to the Android studio. I created new project and recreated what was done in Tutorial 3, but it didn't work. This time there was a problem with AndroidManifest, where I forgot to add. detailActivity. The first problem I faced was, when I tried to change the layout of the menu. But this problem was caused by too small windows for description and text had weird formatting.

11.12.2024

Next weird thing was text. Description changed position randomly. It was caused by the position change in my_listview_detail and wrong string texts. In this moment I wasn't sure how I want the app to look like, but in this moment, I changed my mind, and the topic of this project changed to Best Castles in the Czech Republic, because railway carriers didn't fit in my vision of the project. I decided to start with the title, so I added a title describing the purpose of the app: Best Castles in Czechia. The title was created by adding a new TextView to the activity_main. But it didn't look good so I made the text larger (20sp) and centered the text by the android:gravity="center". Next thing was large gap between TextView and ListView. I changed this in Design view. When I was happy with the title I thought that the description needs improvements as it was just basic information and it wasn't right. This change was done in Strings – description.

I wanted to add button with link to: **Visit Czechia** (official turism page of the Czech Republic). I started with the layout change as in activity_main was the myListView linked

to the bottom and after size change of this field I added button with ID **linkButton**. I remembered from the tutorial how this can be done, so I swiched to MainActivity and started by finding the button in the layout using findViewById. This helps me to do next steps with that. To make the button functional, I used setOnClickListener, which listens for when the user clicks on the button and defines what should happen after. I chose the official website for travel information and defined a string with the web address. Then, I converted the String into a **Uri**, which is required for opening of the browser. Next, I created an **Intent** with the action ACTION_VIEW, signalling that user wants to open a webpage. Finally, I added start of the activity.

At this moment I was satisfied with the button, and I wanted to add another feature which was **toggle** switching between light and dark mode. Because I never done anything similar in the Android Studio I found really good tutorial for implementing of the dark mode (https://www.youtube.com/watch?v=94gvVpGsap8). I started with implementation concretely in MainActivity. For this I needed to create title variable linked to ID of the title, variable for listView called **list** and the same for the switch. Then, how I had prepared switchMode I could use setOnCheckedChangeListener and by **if statement** say if it will be a light or dark mode depending in position of the switch. Only thing where I had a problem was with text of the myListView as it is a little bit different than regular title. So, I started with giving information to the detailActivity as I wanted to have a dark mode even after showing picture of the castle. For make this functionality I added parameter **checked** to the itemAdapter. This new parameter is changed in setOnCheckedChangeListener where I gave actual information of the toggle switch (if switched or not) to **checked**. The next step was adding of the showDetailActivity Methode. This transmit the information if the toggle is switched from MainActivity to detailActivity. In detailActivity I made basic if else condition and changing the background colour.

But there was still a problem with listView's text dark mode. First what I thought was adding it to the MainActivity, but it didn't work. During changes I got completely lost, because I change almost all files, and I had no idea how to fix it. I used ai for error check and if it helps me to finding way out. My code wasn't completely bad as the structure and idea was right, but addressing in itemAdapter was not. It helped me with itemAdapter.setDarkMode(isChecked); Because I tried to use only itemAdapter in MainActivity and it is probably not possible to do it that way.

12.12.2024

Next thing I didn't like was the basic app icon. This change I learned from video tutorial (2), and it is really simple. The only thing was creation of res – new - Vector asset and choice of the right picture. But it is better to change background from green to something else if some users have different shaped icon. That can be found in Background_layer and change from image to color as I have colorful picture I changed to dark gray as it looks the best from options. After clicking finished it created a folder called mipmap.

The last functionality I wanted to add was a map showing the position of these castles. First of all, I visited the official google website for google maps (3). In this step I needed to create an API key. This took me a while, because this process changed from the time when the tutorial was written. In Google Cloud I created a new project called Mobile. Then it was important to enable APIs and services with the next step Maps SDK for Android. First it wanted from me a registration and credit card, but it is a free trial. I didn't use any restriction because the project will be checked, and I wasn't sure if it would work for a lecturer repairing the assignment. In module Gradle file I copied the dependency line and paste it to the Gradle Scripts – Build.gradle app where are already dependencies and click the sync now button. In step 9 of the google tutorial it says, that I need meta data and gave it to the Manifest. I can change the Api key right here, but I wanted to make it more clear so I created string called ApiKey in Values – Strings.xml. Into this string I copied API key from that google provided me and rename back in Manifest value to ApiKey. Back on google's webside I found XML layout file that I need for implementation of the map. I added this fragment into the activity_main right between the list and Button, but this needs some graphical changes later. Id I let as the map, because it wasn't necessary to change it. For a map is used **SupportMapFragment** and **getSupportFragmentManager**, but after trying to add findFragmentById I faced a problem with fragment providing (error). This was solved by clicking on red bulb on left and cast expression to com.google and it added (SupportMapFragment). Then inicializing of the Map and creation of the LatLng for Karlštejn castle with coordinates. I used premade Marker where I gave these coordinates and name that will be shown. Because I didn't want to have camera view right above this castle but approximately in the middle of the Czech Republic, I had to change camera view by moveCamera and CameraUpdateFactory I imported this earlier. It ended with an error. I searched where the problem could be, and I found that I forgot to implement

OnMapReadyCallback. Even after this change, there still was a problem and after a longer time I found, that I changed in android:name="com.google.android.geo.API_KEY" the API_KEY for my api key and it caused the app crash. Because camera wasn't where I expected I changed **NewLatLng** to **NewLatLngZoom** and that provides zoom level. Now was the map functioning, but the design needs a lot of attention, because it was still small, zoom level wasn't perfect and I wanted to add all castles, not only the first one. Adding of other castles was easy as it is just needed to add other Markers with their specific coordinates. When I tried to change the design, project for some reason don't work to run at all. Helped **Build - Clean Project** and than **Build - Rebuild Project.**

I still wasn't satisfied with the design of the app as the possition of the map was too low. After some time in activity_main I changed the design into the form I liked. The last part was design of the StatusBar (6) where it was easier than I expexted. This was new for me, bacause the change was made in res – themes – themes.xml. For change it is used colorPrimaryDark and StatusBarColor. Problem with button was, that I tried to set background colour for the button, but it is not way how to do that. I had to change **backgroundTint** and after that was button in right design. After finishing the project I wanted to clear unnecessary code and make it more clear.

Lastly, I would like to say, that I tried to make a video of that project, but my laptop is too slow for handling both Android Studio and OBS, so the video is extremely slow and there are almost no animations visible.

References

1. Stack Overflow. (2023, April 24). *No layout folder in Android Studio*. Stack Overflow. Available at: https://stackoverflow.com/questions/76095138/no-layout-folder-in-androidstudio
2. YouTube. (2022, January 3). *How to Change App Icon in Android Studio | Beginner Tutorial. Available at:* [Video]. YouTube. https://www.youtube.com/watch?v=bJjHgWjiAKw
3. Google. (n.d.). *Maps SDK for Android Quickstart*. Google Developers. https://developers.google.com/maps/documentation/android-sdk/start
4. Google. (n.d.). *Get started with the Android Publisher API*. Google Developers. Availavle at: https://developers.google.com/android-publisher/getting_started
5. Google. (n.d.). *Maps SDK for Android Quickstart*. Google Developers. Availavle at: https://developers.google.com/maps/documentation/android-sdk/start#maps_activity_file
6. Page Pixels. (2023, Fabruary 18). *How to Change Status Bar Color in Android Studio | 2024 | JAVA, KOTLIN* [Video]. YouTube. Available at: https://www.youtube.com/watch?v=35-2raGroRs&ab_channel=PagePixels