COMPX202/Y05853

Assignment One: Introduction to Android

Due: Friday 7th of November 11:59pm

Introduction

This assignment has four objectives:

1. Successfully create your first Android application;
2. Use Git from within Android Studio;
3. Learn about and develop the layout of a responsive Android application;
4. Interact with the Android UI through Java.

The assignment builds on your Android UI skills – specifically using XML to design flexible layouts with ViewGroups and Views – and introduces working with multiple activities and passing data between them. It also makes use of the values, drawables, and layout folders, as well as arrays.xml. Additionally, this assignment will further develop your familiarity with Android Java development.

"University of Waikato is exploring the development of a mobile companion app for students. The concept app, `CampusLife`, will showcase four key student service areas: `Facilities`, `Events`, `Clubs`, and `Support`. The app must be responsive, look good across a variety of screen sizes and orientations, and provide information about each area in a clean and readable format."

Resources required for completing this assignment to the specifications have been included on Canvas. Pay close attention to the screenshots used throughout these assignment instructions as they will demonstrate how the application should look and function.

You are required to use GitLab for assignment submission. During marking, I will be looking to ensure that there is at least one commit and push per assignment part. Failure to do so will result in a reduction of marks even if the final application otherwise meets the specifications.

Note, if any of these instructions are unclear to you, you are expected to ask for clarification. Start the assignment early and make use of the labs for extra guidance and assistance.

Figure 1: XML layout

Part A: Setup – creating an Android app and using Git

1. In GitLab, create a new repository called "AssignmentOne". *(DO NOT include a readme file. The repository needs to be empty – Android Studio requires an empty repo).*
2. Make sure that the visibility of your project is set to Private. *(If not: Settings > expand Permissions > Project visibility: Private; Save changes).*
3. Create a new Android application in Android Studio. *(see the lecture content for creating a new Android application).*
4. Run your application to make sure it works. This can either be done using your personal device or using an emulator.
5. Use Git from inside Android Studio to add, commit, and push your new Android application to GitLab. *(see the lecture content for using Git inside Android Studio, don't forget to remove .iml from .gitignore).*
6. Check that your Android application has successfully been pushed to GitLab – the "AssignmentOne" repository should now contain your Android application.

Note, when setting up your Android application, you should create it using an Empty Views Activity. It should be named "AssignmentOne", should use Java, and should use a minimum API level of API 24: Android 7.0 (Nougat).

Part B: Android XML

Alter the activity_main.xml file to duplicate the UI layout shown in Figure 1. There are good and bad ways to create this layout. Use what you have learned in lectures to create a clean, responsive layout using ViewGroups and Views. Your application must be responsive and must not use any hard-coded sizes (except for the banner title text size of 40sp). You can find the required images on Canvas.

Don't forget to add, commit and push your work to GitLab.

Figure 2: Launcher icon *Figure 3: Portrait Layout* *Figure 4: Landscape orientation*

Part C: Android resources

While Part B focused on using XML to describe the UI layout, this section utilises some of the other resources found in the `res` folder. Complete the following steps:

1. Change the name of the application to be "CampusLife".
2. Create a new launcher icon using the logo image (available on Canvas).
3. Add the following colours to your colors.xml file:
    a. Green: #1B5E20
    b. Light Green: #66BB6A
4. Create a button selector resource that colours the buttons green when not pressed and light green when pressed.

Your application UI should now reflect that shown in Figure 3, while your launcher icon should look like the one shown in Figure 2.

Don't forget to add, commit, and push your changes to GitLab.

Part D: Orientation

Your current layout works well when in portrait orientation, but may not be as well suited for landscape. Create a new activity_main.xml file for landscape layout and alter the code to display the four image buttons next to each other, rather than in a two-by-two grid, as shown in Figure 4.

Note, if you have done Part B correctly, you should be able to copy and paste your XML layout from your portrait activity_main.xml file and only change a small portion of the bottom half of the XML to make it work.

Don't forget to add, commit, and push your changes to GitLab.

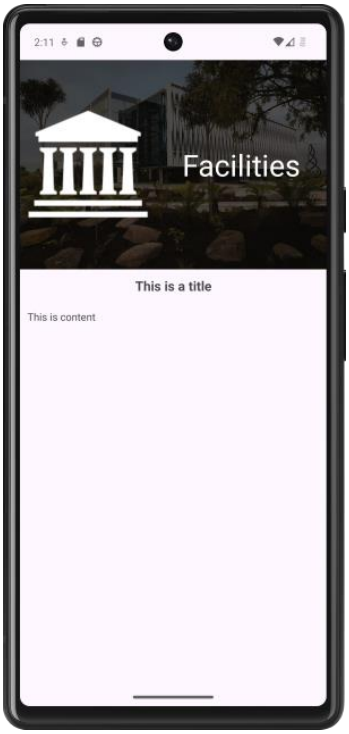

Figure 5: Empty Detail Activity

Part E: Onclick Events

Your next task is to add `"onclick"` events to the four ImageButtons. The Main Activity of the application is shown in Figures 3 and 4, where the ImageButtons are the four icons in the lower half of the page. Please note, you will need to make changes to both the portrait and landscape activity_main.xml files.

Add a Toast to each onclick event to check that they are working:

i.e. Toast.makeText(this, "Facilities clicked", Toast.LENGTH_SHORT).show();

Don't forget to add, commit, and push your changes to GitLab.

Part F: Creating an Activity

Your final application will provide users with information about each of the topics when they click the ImageButtons. To facilitate this, your next step is to create a new Activity and set up the XML layout for that Activity.

Your new Activity should be called "DetailActivity" and should contain the layout illustrated in Figure 5. Remember, there are good and bad ways to create this layout. Use what you have learned in the lectures to create a good layout using ViewGroups and Views. Your application must be responsive and must not use any hard coded sizes. You can find the required images on Canvas.

Don't forget to add, commit and push your work to GitLab.

Figure 6: Detail Activity (Facilities)

Figure 7: Detail Activity (Events)

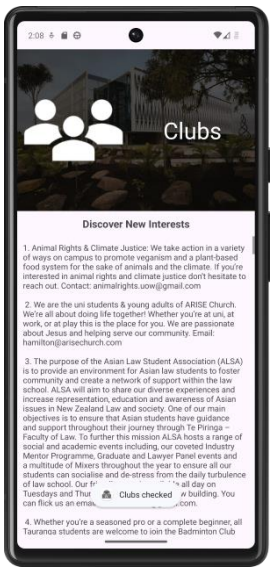

Figure 8: Detail Activity (Clubs)

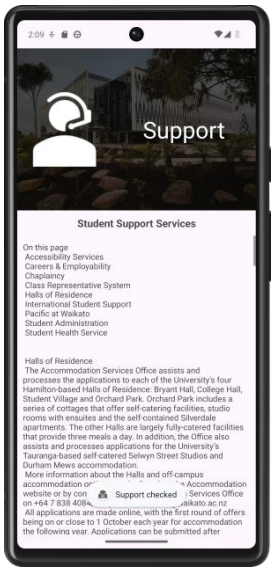

Figure 9: Detail Activity (Support)

Part G: Using an Intent

Next create an explicit Intent that will be called inside each of your MainActivity "onclick" methods. The intent should start the DetailActivity.

Make sure you pass some sort of unique identifier through the Intent, you will need to use this in the DetailActivity to determine which content to display.

Don't forget to add, commit, and push your changes to GitLab.

Part H: Updating the UI using Java

Until now, your DetailActivity should be populated with place-saver text, as shown in Figure 5. You are required to add code to its onCreate() method that populates it with the dummy data stored in arrays.xml. (Note, you will need to copy the arrays.xml file into the "values" folder).

The arrays.xml file contains three arrays, one that contains the titles for each of the topics, one that contains the detail text for each, and one that contains the names of the corresponding images. You will need to retrieve the title, content, and image name that corresponds to the ImageButton that was clicked (this is where your unique identifier will come in handy).

Once you have retrieved the content from arrays.xml, you can use it to populate the DetailActivity. Figures 6 to 9 show what the DetailActivity should look like once it has been populated with dummy data for each of the topics.

Please note, you should not use any 'if statements' or 'switch cases'.

Don't forget to add, commit, and push your changes to GitLab.

Part I: Code quality

Make sure you structure and comment your code appropriately.

Submitting

Make sure you have pushed all of your changes to your GitLab repository. Ensure that you can see your changes on GitLab. It is usually a good idea to clone the repository back to a new location on your local machine and re-run it to check that it was uploaded correctly.

Grading

Weighting	*Allocated to*
10%	Part A – Creating your app and using Git
20%	Part B – Android XML
5%	Part C – Android resources
5%	Part D – Orientation
5%	Part E – Onclick events
20%	Part F – Creating a new Activity
5%	Part G – Using an intent
20%	Part H – Updating the UI using Java
10%	Part I – Code quality