LabW01 – Android Basics

Objectives:

1. Get familiar with Android Studio
2. Understand basic Android UI elements

Tasks:

1. Create your first Android project for mobile and wear apps
2. Build a simple mobile app: ToDoList
3. Add time information to your ToDoList app

This tutorial is based on Android Studio. If you haven't set up your environment for Android development, please refer to https://developer.android.com/studio/install. You can download the latest version 4.0.1 of Android Studio from: https://developer.android.com/studio/.

For Mac users, please visit the following link to download Android Studio: https://redirector.gvt1.com/edgedl/android/studio/install/4.0.1.0/android-studio-ide-193.6626763-mac.dmg

Almost everybody today has a smartphone and is using apps on their smartphone. Some use wearables with apps. There are apps to check email, weather, play games, and so on.

In this tutorial, we will learn how to develop our first apps in the Android platform.

Please backup your codes to your U: drive or USB key before you leave the lab.

Task 1: Create your first Android project

1. Launch Android Studio.

2. In the 'Welcome to Android Studio' window, click 'Start a new Android Studio project'.

3. In the 'Create New Project' window:
   - Under "Phone and Tablet" tab, create a new empty activity by selecting "Empty Activity"
   - Click Next

4. In the 'Configure Your Project' window, enter the following:
   - Name: HelloWorld
   - Package name: comp5216.sydney.edu.au.helloworld
   - Save location: (leave as is or change to your preferred location)
   - Language: Java
   - Minimum SDK: API 23: Android 6.0 (Marshmallow)
   - Click Finish

5. After some processing, Android Studio opens the IDE. Now take a moment to review the most important files. First, be sure the Project window is open (select View > Tool Windows > Project) and the Android view is selected from the drop-down list at the top of that window. You can then see the following files:

- app > java > comp5216.sydney.edu.au.helloworld > MainActivity
- app > res > layout > activity_main.xml
- app > manifests > AndroidManifest.xml
- Gradle Scripts > build.gradle

6. Open the Android Virtual Device Manager by selecting Tools > AVD Manager. Click Create Virtual Device to create an emulator.

7. In the Select Hardware screen, select a phone device, such as Pixel 2, and then click Next.

8. In the System Image screen, select the version with the highest API level e.g. Release Name: “R”. If you don't have that version installed, a Download link is shown, so click that and complete the download.

9. Click Next.

10. On the Android Virtual Device (AVD) screen, leave all the settings alone and click Finish. Close the Android Virtual Device Manager.

11. We will run the app on the emulator we have just created. In Android Studio, click the app module in the Project window and then select Run > Run ‘app’ (or click Run ▶ in the toolbar, or click ‘Shift + F10’).

12. Android Studio installs the app on the emulator and starts it. You should now see "Hello World!" displayed in the app running on the emulator.

13. Next, we will create our first wearable app using Android Studio's New Project wizard. Click File > New > New Project.

14. In the ‘Create New Project’ window:
    - Under “Wear OS” tab, create a new blank activity by selecting “Blank Activity”
    - Click Next

15. On the Configure your new project screen, enter the following:
    - Name: HelloWorldWearApp
    - Package name: comp5216.sydney.edu.au.helloworldwearapp
    - Save location: (leave as is or change to your preferred location)
    - Language: Java
    - Minimum API level: API 23 Android 6.0 (Marshmallow)

- Click Finish

16. Open the Android Virtual Device Manager by selecting Tools > AVD Manager.

17. Click Create Virtual Device.

18. In the Category pane, select ‘Wear OS’ and choose a hardware profile, such as ‘Android Wear Square’. Click Next.

19. In the ‘Select a system image’ screen, select an image with the Release Name of Oreo, the API Level of 26, and the Target of "Android 8.0 (Android Wear)". If you don't have that version installed, a Download link is shown, so click that and complete the download.

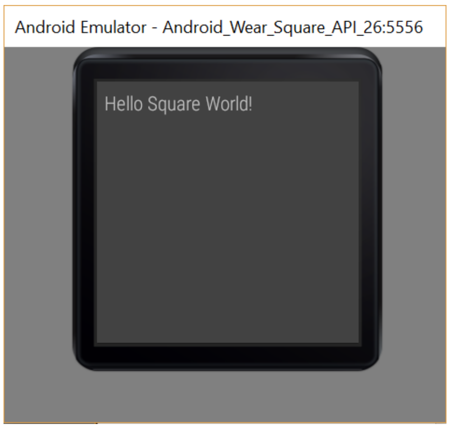
20. Click Next and then click Finish.

21. Close the Android Virtual Device Manager.

22. In Android Studio, click the Run ‘app’ button (or click ‘Shift + F10’).

23. Select the new AVD just created, and click OK.

24. The AVD starts and, after a few moments, runs your app. A "Hello Square World!" message is displayed.



Android Emulator - Android_Wear_Square_API_26:5556

Hello Square World!

Task 2: Build a simple ToDoList App

1. Start a new Android Studio project.

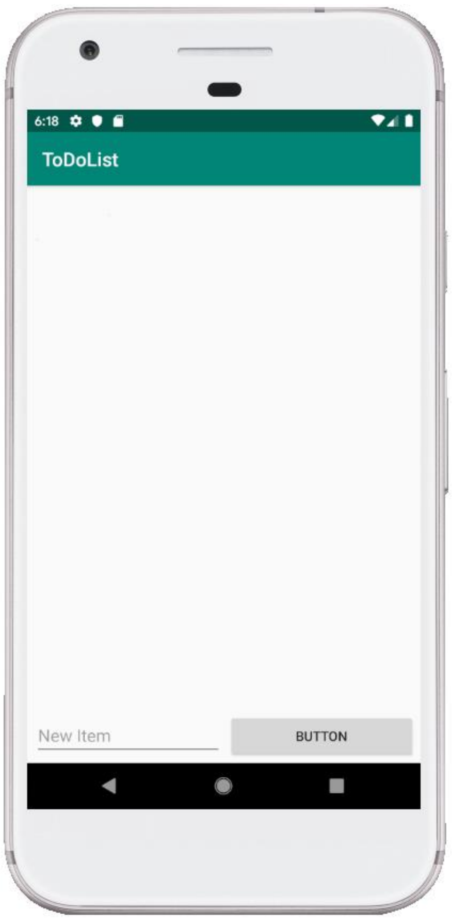
2. In the ‘Create New Project’ window:
    - Under “Phone and Tablet” tab, create a new empty activity by selecting “Empty Activity”
    - Click Next

3. In the ‘Target Android Devices’ screen, select:
    - Name: ToDoList
    - Package name: comp5216.sydney.edu.au.todolist
    - Save location: (leave as is or change to your preferred location)
    - Language: Java
    - Minimum API level: API 23: Android 6.0 (Marshmallow)
    - Click Finish

4. In ‘activity_main.xml’:
    - Remove the “Hello World!” TextView
    - Drag these widgets onto the main_layout (please refer to the screenshot below)
        - ➢ Text → Plain Text
        - ➢ Buttons → Button
        - ➢ Legacy → ListView
        - ➢ Open the XML layout file and edit the following:
            - o Assign hint to EditText: New Item
            - o Assign ID to views:
                - ▪ ListView -> lstView
                - ▪ EditText -> txtNewItem
                - ▪ Button -> btnAddItem
    - Specify the necessary constraint for each of the widget to position a given widget relative to another one e.g. app:layout_constraintBottom_toBottomOf.

Refer to https://developer.android.com/reference/androidx/constraintlayout/widget/ConstraintLayout to learn more about ConstraintLayout which allows you to position and size widgets in a flexible way.

5. Run it. You should only see an empty list, text input and button. Clicking the button does not have any effect.



```
<Button
    android:id="@+id/btnAddItem"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="4dp"
    android:layout_marginEnd="5dp"
    android:text="Button"
    android:onClick="onAddItemClick"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/txtNewItem"
/>

<ListView
    android:id="@+id/lstView"
    android:layout_width="390dp"
    android:layout_height="560dp"
    android:layout_marginBottom="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<EditText
    android:id="@+id/txtNewItem"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="6dp"
    android:layout_marginEnd="6dp"
    android:layout_marginStart="8dp"
    android:ems="10"
    android:hint="New Item"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/btnAddItem"
    app:layout_constraintStart_toStartOf="parent" />
```

6. Add the code in MainActivity.java (**keep the original imports and package**). Read the comments inside the code.

```
package comp5216.sydney.edu.au.todolist;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ListView;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    // Define variables
    ListView listView;
    ArrayList<String> items;
    ArrayAdapter<String> itemsAdapter;
    EditText addItemEditText;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Use "activity_main.xml" as the layout
        setContentView(R.layout.activity_main);

        // Reference the "listView" variable to the id "lstView" in the layout
        listView = (ListView) findViewById(R.id.lstView);
        addItemEditText = (EditText) findViewById(R.id.txtNewItem);

        // Create an ArrayList of String
        items = new ArrayList<String>();
        items.add("item one");
        items.add("item two");

        // Create an adapter for the list view using Android's built-in item layout
        itemsAdapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1,
items);

        // Connect the listView and the adapter
        listView.setAdapter(itemsAdapter);
    }
}
```

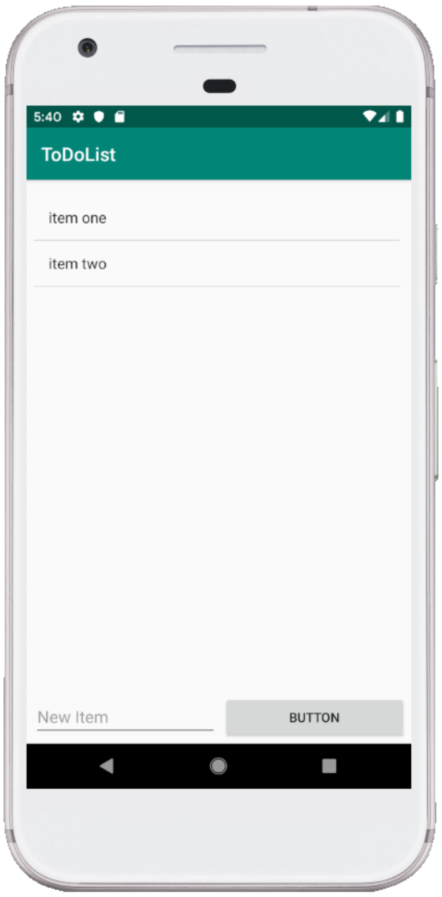
7. Run the project

8. Add button click handler as follows:

```
public void onAddItemClick(View view) {
    String toAddString = addItemEditText.getText().toString();
    if (toAddString != null && toAddString.length() > 0) {
        itemsAdapter.add(toAddString); // Add text to list view adapter
        addItemEditText.setText("");
    }
}
```

9. Link the method to the button. Edit the activity_main.xml layout file:
    1) Update the button text to ‘Add’
    2) Add ‘android:onClick’ attribute to the method name ‘onAddItemClick’

10. Enjoy running your ToDoList Android app! If you did it right, you should see something like below. Add as many new items as you like to test it.



Task 3: Add time information to your ToDoList app

In this task, you are required to add time information and display it every time you add a ToDoItem to your ListView. In order to achieve this, you may need to use the "Date" class. For detailed information of this Date class, please refer to https://developer.android.com/reference/java/util/Date.html

References

The tutorial is partially based on the materials from Android Developer:
https://developer.android.com/training/basics/firstapp/
https://developer.android.com/training/basics/firstapp/running-app.html
https://developer.android.com/training/wearables/apps/creating

If you wish to set up Android app development environment, please refer to:
https://developer.android.com/studio/install.html