IoTSSC LAB #5
ANDROID SERIAL APP

OVERVIEW

In this lab, you will develop an app, which will facilitate the communication between a phone and the embedded device via Bluetooth. This communication will involve the HC-06 Bluetooth module and the Mbed's serial communication API from the previous lab.

The goal of this lab is to gain some experience with Android development and establish a baseline implementation for your project.

SETTING UP THE PROJECT

Install Android Studio if you have not already done so: installation_instructions.

Unpack the zip file (IoTSSC_App.zip) provided and import the skeleton code into Android Studio by opening from an existing Android Studio project.

In this lab you will extend the skeleton code to implement Bluetooth serial communication with your embedded device and provide a user interface to facilitate the communication.

USER INTERFACE

The app has a simple pre-existing user interface for detecting and displaying Bluetooth devices, which revolves around a single (main) activity.[1] You will be developing the functionality of this activity through the relevant files: MainActivity.class and activity_main.xml.

The code and purpose of the app should largely be self explanatory. However, you may wish to read about the RecyclerView class, ButterKnife view bindings or the BroadcastReceiver class if you are unfamiliar with any of these terms and wish to learn more about the existing code.

1 Excluding the SplashScreenActivity, which is just for fun visualization purposes and irrelevant to this lab

BLUETOOTH CONNECTION

ENABLING BLUETOOTH

Before we can connect to a Bluetooth device, we need discover it using a Bluetooth scan. The Android API has provided a BluetoothAdapter class for this purpose (see android.bluetooth.BluetoothAdapter). To receive an instance of this class, use the following snippet:

```
bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
```

The OS needs to be informed that the user has given permission for the app to access various Bluetooth functions on the phone, otherwise a java.lang.SecurityException will be thrown when these functions are called. The following permissions are required, and must be added to the AndroidManifest.xml for the rest of the code to work:

```
android.permission.BLUETOOTH
android.permission.BLUETOOTH_ADMIN
android.permission.ACCESS_COARSE_LOCATION
```

It is possible to check if Bluetooth is enabled with the `BluetoothAdapter.isEnabled()` function. If this returns false, we can fire an Intent with action `BluetoothAdapter.ACTION_REQUEST_ENABLE`.

COMMUNICATING OVER SERIAL

Taking a look at the documentation contained in the app code, we can see the connection to a device is triggered on user request after they select a scanned device from the presented list (implemented as a RecyclerView).

Navigating to the bottom of the MainActivity.class, your job is to implement the communication functionality of the app using the default methods presented.[2] The app should be able to display the latest message received from the device alongside its time of arrival to the user. The app should also allow a user to send messages to the device.

You will be mainly using the message layout (*connection_message_layout.xml)* where you should add the necessary view to implement this functionality. Implement the views such that the app user interface looks broadly similar to Figure 1.

It will also be necessary to update the views in the main activity layout (*activity_main.xml)* to ensure the views you've added work with existing ones (making use of the existing views when necessary). The user should always be up to date about the state of the app.

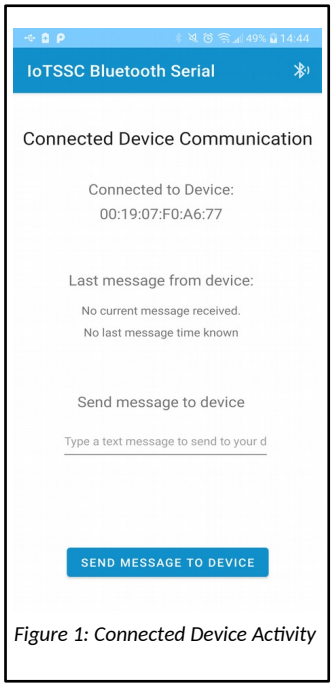

Figure 1: Connected Device Activity

2 Methods which interface with the android-bluetooth-serial library.

YOUR APPLICATION CODE

In this lab, you need to perform the following tasks:

- Learn about and become familiar with Android app programming
- Write code to interface with your device, which should be able to communicate over Bluetooth – to both read from and send messages to this device
- Implement the user interface of the Android app, such that you allow users to view previous messages and send new custom ones.
- If you have completed the above tasks, describe what the purpose of the ApplicationData.class might be (in the context of multiple activites) and find any improvements you think would make it more useful for users.
- If you have completed the above tasks, you may also wish to plan and implement the means for how your app will interface with your chosen cloud stack.