EE447 Mobile Internet Lab 1

Zhou Litao 518030910407 F1803016

April 16, 2021, Spring Semester

1 Introduction

In this lab, we will get used to the WiFi system and accomplish the sampling and measuring of WiFi signal strength through programming in Android on smartphone. Based on the skeleton codes, we will compile them and run on a actual Android phone. We will collect the signal strength of the WiFi AP nearby and analyze them. Furthermore, we will implement the function of locating user's position.

2 Environment Setup

The environment of this experiment is listed as follows. I could not run Android Studio on my computer and used Intellij IDEA as an alternative.

- Development Envrionment OS: macOS 11.2
- IDE: Intellij IDEA Ultimate 2020.2
- Android SDK 26

Note that a few classes (such as `android.support.v4.app.ActivityCompat`) used in `MainActivity.java` are deprecated. Following the hints of Intellij IDEA, we port them to the current environment (changed to `androidx.core.app.ActivityCompat`). Furthermore, we also changed the layout of the `activity_main.xml` so that the current SDK could compile it.

3 WiFi Signal Strength

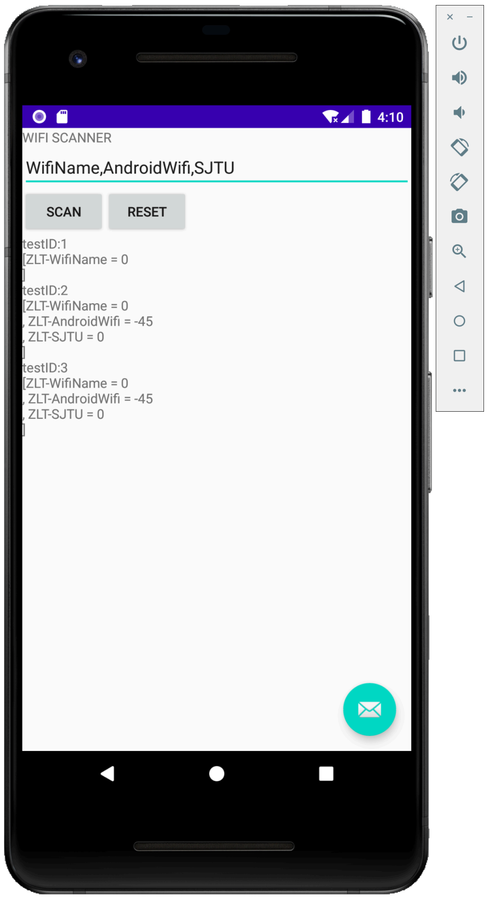
The skeleton has implemented the following function:

1. User click the `SCAN` button. `MainActivity.java` will respond to the click event by creating a `SuperWifi.java` class.
2. SuperWifi.java obtain the SSID groups by `mWiFiManager.startScan()` function.
3. SuperWifi.java will select the user expected APs and record their signal strength 10 times.
4. SuperWifi.java write the results into an output file. The scanning ends.
5. MainActivity.java notifies users by showing a message in the TextView area of the application.

In order to port to SDK 26, we rewrite the user interface `activity_main.xml`. To help with debugging, we allow users to specify which APs should be measured in the user interface, so that the APs are no longer hard-encoded into the code. We also change some codes in `SuperWifi.java` in order to fit our new design.

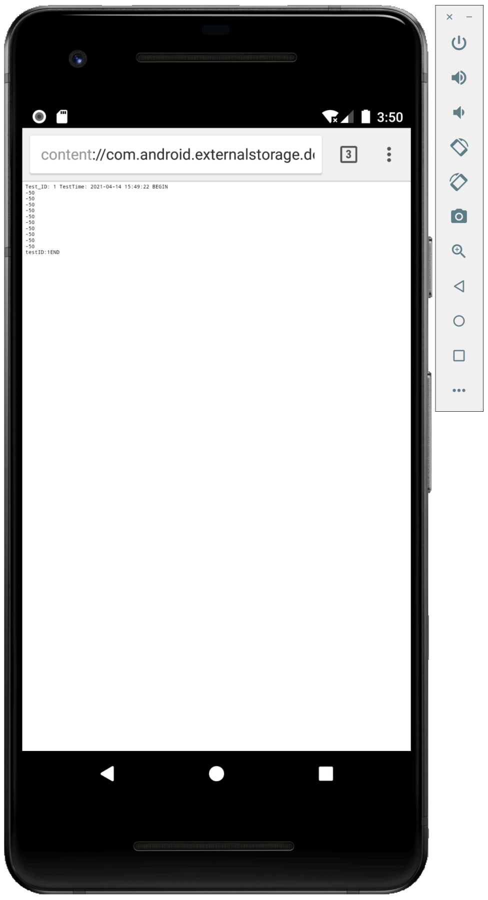
We test the basic functions on Android Emulator, the results are shown in Figure 3 and 3. Note that the emulator only simulates a Wifi AP called `AndroidWifi`, so we get constant results.

The testing data are listed in Figure 3. Since they are very stable, we don't need to plot them out.



```
WIFI SCANNER
WifiName,AndroidWifi,SJTU
testID:1
[ZLT-WifiName = 0
]
testID:2
[ZLT-WifiName = 0
, ZLT-AndroidWifi = -45
, ZLT-SJTU = 0
]
testID:3
[ZLT-WifiName = 0
, ZLT-AndroidWifi = -45
, ZLT-SJTU = 0
]
```

Figure 1: Emulation Demo



```
content://com.android.externalstorage.d
Test_ID: 1 TestTime: 2021-04-14 15:49:22 BEGIN
-50
-50
-50
-50
-50
-50
-50
-50
-50
-50
testID:1END
```

Figure 2: Output Result

4 Questions

1. Why is necessary to record all the measured value rather than only the average value?

   From the experiment we know that the WiFi strength we test at a particular moment is very unstable. However, user usually expects a stable value. Therefore, it is important that our application can identify the outliers that are extremely off the average. By taking the record of all measured value, we can identify the irregular measurement out and evaluate the confidence of the result of a scanning.

2. Besides the WiFi signal strength, what other information of the Routers can be got in the test?

   We may refer to the system API document to identify what attributes we can get. See Figure 4. We list a few as follows:

   - `SSID`: The network name.
   - `BSSID`: The address of the access point.
   - `level`: The detected signal level in dBm, also known as the RSSI.
   - `capabilities`: Describes the authentication, key management, and encryption schemes supported by the access point.
   - `frequency`: The primary 20 MHz frequency (in MHz) of the channel over which the client is communicating with the access point.

3. Why does the scanning need to be operated in thread “scanThread”?

   ScanThread takes a long time because we need to repeatedly record the WiFi signal strength. If we don’t let the scanning be a separate thread from the main process, every time user launches a scan command, the main process will be blocked, which will degrade user’s experience.

5 Indoor Location

The WiFi signal strength has a relation to the device’s distance to the AP as follows.

$$d = 10^{\frac{|RSSI|-A}{10n}} \tag{1}$$

where A and n are hyperparameters.

We implement the Indoor Location algorithm in its simplest form - triangle location[1]. To be specific, we use the following equation sets to solve the user’s location.

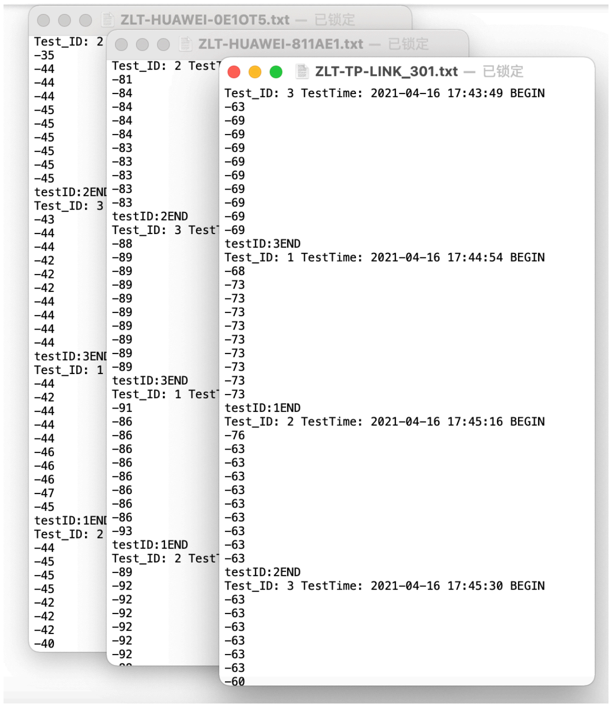
$$\begin{cases} (x-x_1)^2 + (y-y_1)^2 &= d_1^2 \\ \dots \\ (x-x_i)^2 + (y-y_i)^2 &= d_i^2 \end{cases} \tag{2}$$

Note that due to measurement errors, the equation should be solved in the form of least squares. However, to simplify the implementation, we choose to measure the signal strength of three APs and obtain two linear equations to solve an approximation solution. The equations are listed as follows.

$$\begin{cases} 2x(x_1-x_3) + 2y(y_1-y_3) &= d_3^2 - d_1^2 + y_1^2 - y_3^2 + x_1^2 - x_3^2 \\ 2x(x_2-x_3) + 2y(y_2-y_3) &= d_3^2 - d_2^2 + y_2^2 - y_3^2 + x_2^2 - x_3^2 \end{cases} \tag{3}$$

As for the implementation, we add an `EqnSolver.java` module to solve the above equation. We add an extra input row where users can input the reference locations of the three known APs, indicated in Figure 5. We augment the `SuperWifi.java` module so that it can calculate the user’s location when exactly three APs are measured.

[1] `https://m.hanspub.org/journal/paper/30812`


ZLT-HUAWEI-0E1OT5.txt — 已锁定

```
Test_ID: 2
-35
-44
-44
-44
-45
-45
-45
-45
-45
-45
testID:2EN
Test_ID: 3
-43
-44
-44
-42
-42
-42
-44
-44
-44
-44
testID:3EN
Test_ID: 1
-44
-42
-44
-44
-44
-46
-46
-46
-47
-45
testID:1EN
Test_ID: 2
-44
-45
-45
-45
-42
-42
-42
-40
```

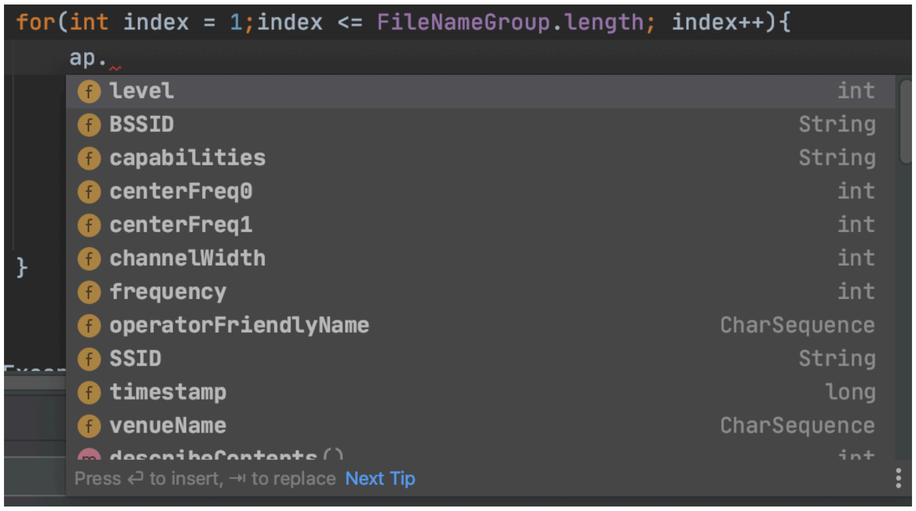
ZLT-HUAWEI-811AE1.txt — 已锁定

```
Test_ID: 2 Test
-81
-84
-84
-84
-84
-83
-83
-83
-83
-83
testID:2END
Test_ID: 3 Test
-88
-89
-89
-89
-89
-89
-89
-89
-89
-89
testID:3END
Test_ID: 1 Test
-91
-86
-86
-86
-86
-86
-86
-86
-86
-93
testID:1END
Test_ID: 2 Test
-89
-92
-92
-92
-92
-92
-92
```

ZLT-TP-LINK_301.txt — 已锁定

```
Test_ID: 3 TestTime: 2021-04-16 17:43:49 BEGIN
-63
-69
-69
-69
-69
-69
-69
-69
-69
-69
testID:3END
Test_ID: 1 TestTime: 2021-04-16 17:44:54 BEGIN
-68
-73
-73
-73
-73
-73
-73
-73
-73
-73
testID:1END
Test_ID: 2 TestTime: 2021-04-16 17:45:16 BEGIN
-76
-63
-63
-63
-63
-63
-63
-63
-63
-63
testID:2END
Test_ID: 3 TestTime: 2021-04-16 17:45:30 BEGIN
-63
-63
-63
-63
-63
-63
-60
```

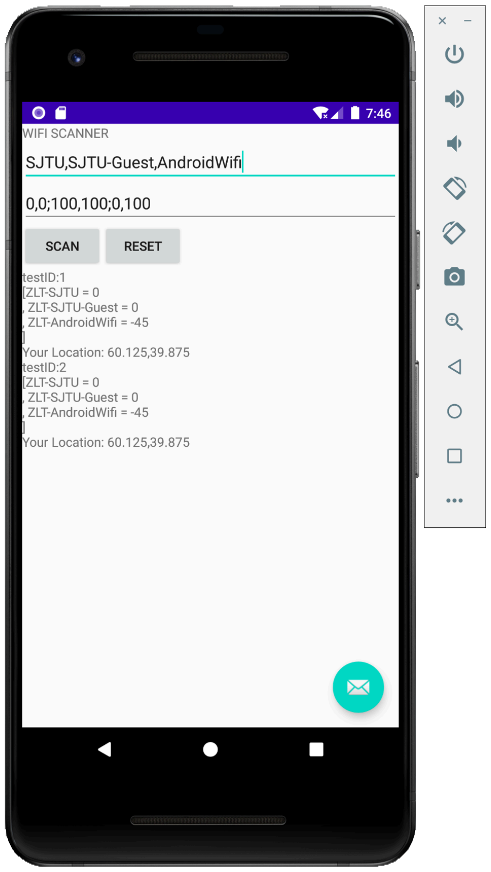
Figure 3: Wifi Strength Data



```
for(int index = 1;index <= FileNameGroup.length; index++){
    ap.
}
```

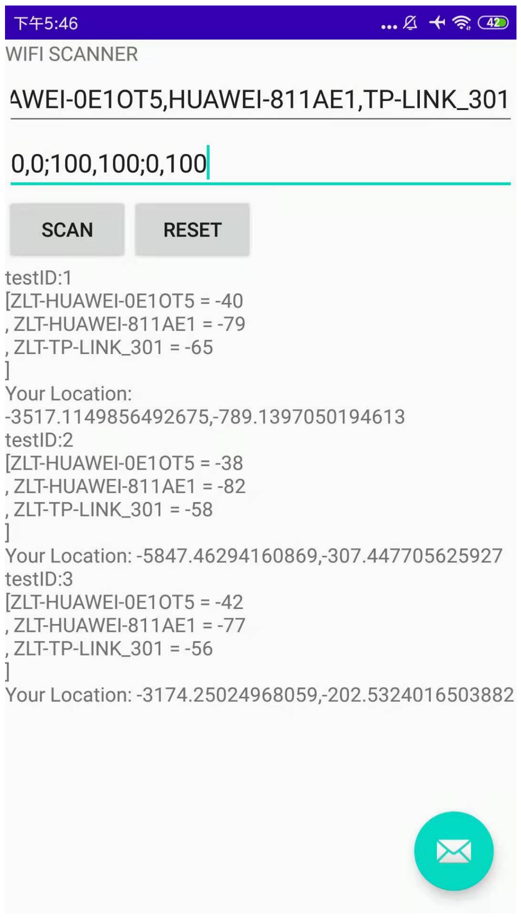
```
level                       int
BSSID                       String
capabilities                String
centerFreq0                 int
centerFreq1                 int
channelWidth                int
frequency                   int
operatorFriendlyName        CharSequence
SSID                        String
timestamp                   long
venueName                   CharSequence
```

Figure 4: AP attributes



```
SJTU,SJTU-Guest,AndroidWifi
0,0;100,100;0,100
testID:1
[ZLT-SJTU = 0
, ZLT-SJTU-Guest = 0
, ZLT-AndroidWifi = -45
]
Your Location: 60.125,39.875
testID:2
[ZLT-SJTU = 0
, ZLT-SJTU-Guest = 0
, ZLT-AndroidWifi = -45
]
Your Location: 60.125,39.875
```

Figure 5: User interface for indoor location



```
AWEI-0E1OT5,HUAWEI-811AE1,TP-LINK_301
0,0;100,100;0,100
testID:1
[ZLT-HUAWEI-0E1OT5 = -40
, ZLT-HUAWEI-811AE1 = -79
, ZLT-TP-LINK_301 = -65
]
Your Location:
-3517.1149856492675,-789.1397050194613
testID:2
[ZLT-HUAWEI-0E1OT5 = -38
, ZLT-HUAWEI-811AE1 = -82
, ZLT-TP-LINK_301 = -58
]
Your Location: -5847.46294160869,-307.447705625927
testID:3
[ZLT-HUAWEI-0E1OT5 = -42
, ZLT-HUAWEI-811AE1 = -77
, ZLT-TP-LINK_301 = -56
]
Your Location: -3174.25024968059,-202.5324016503882
```

Figure 6: Location on Real Machine, the location result is not correct because the hyperparameters A and n have not been adjusted