AsReader Finger-Type

Android SDK Reference Guide

ASR-023B

Asterisk, Inc.

2020-07-10

Revision History

Ver.	Description	Revised by	Date
V1.0	Initial version	Trinity Liu	2020/9/3
V1.1	1. Delete getMacAddress 2. Delete receivedMacAddress	Trinity Liu	2020/9/3
V1.2	Fixed the error value of AsFingerConnectionType.	Trinity Liu	2024/7/1

Contents

Introduction

Main purposes of this paper:

- Guide developers to build the development environment so that developers can use the AsReader SDK library to develop Android applications.
- Explain the SDK library to the users.

Development tools:

- Android Studio 3.5.2+
- Android SDK 10.0
- Android Gradle 5.4 - 2.1

System requirements:

- Android 9.0+

1. Import and Usage of SDK

1.1. Import SDK

1. Click the project file "libs" in the app folder, and right-click "Reveal in Finder" (as shown in FIG. 1-1-1).

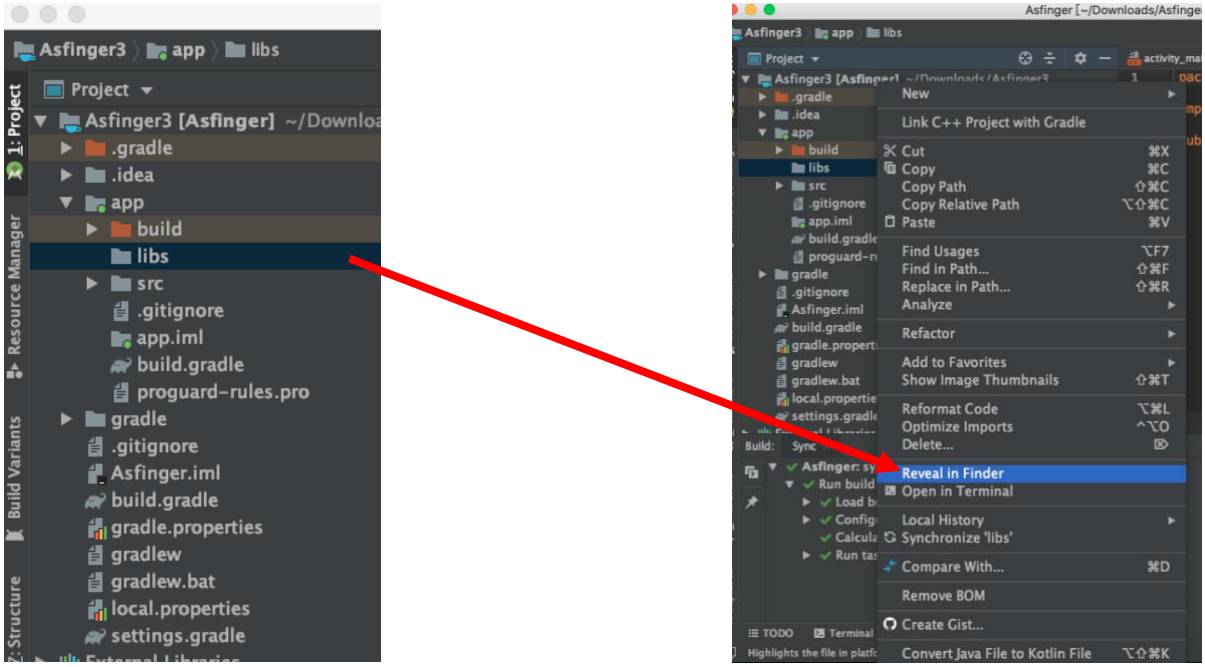

FIG. 1-1-1

2. In the pop-up window, select the "libs" directory, and paste "asfingersdk.aar" into it (as shown in FIG. 1-1-2). After the above operation, "asfingersdk.aar" will appear under "libs" of this project (as shown in FIG. 1-1-3).

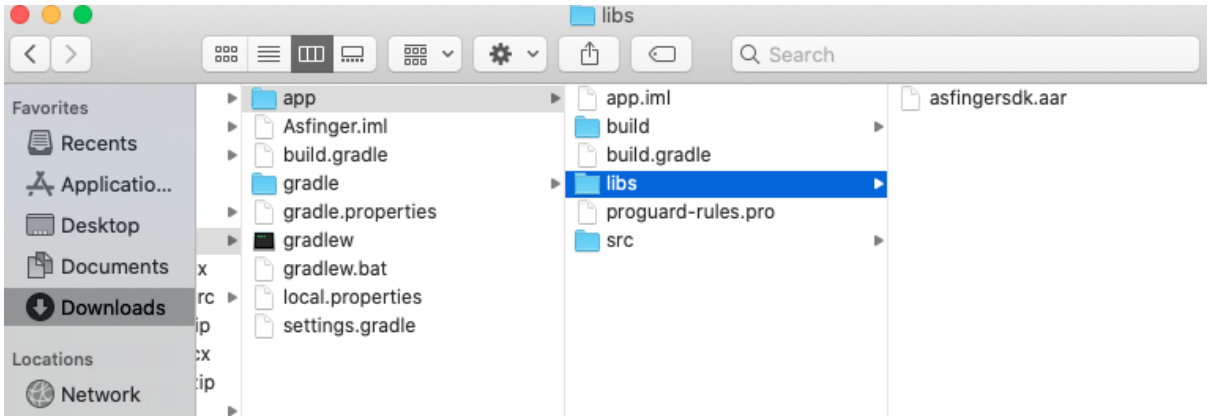

FIG. 1-1-2

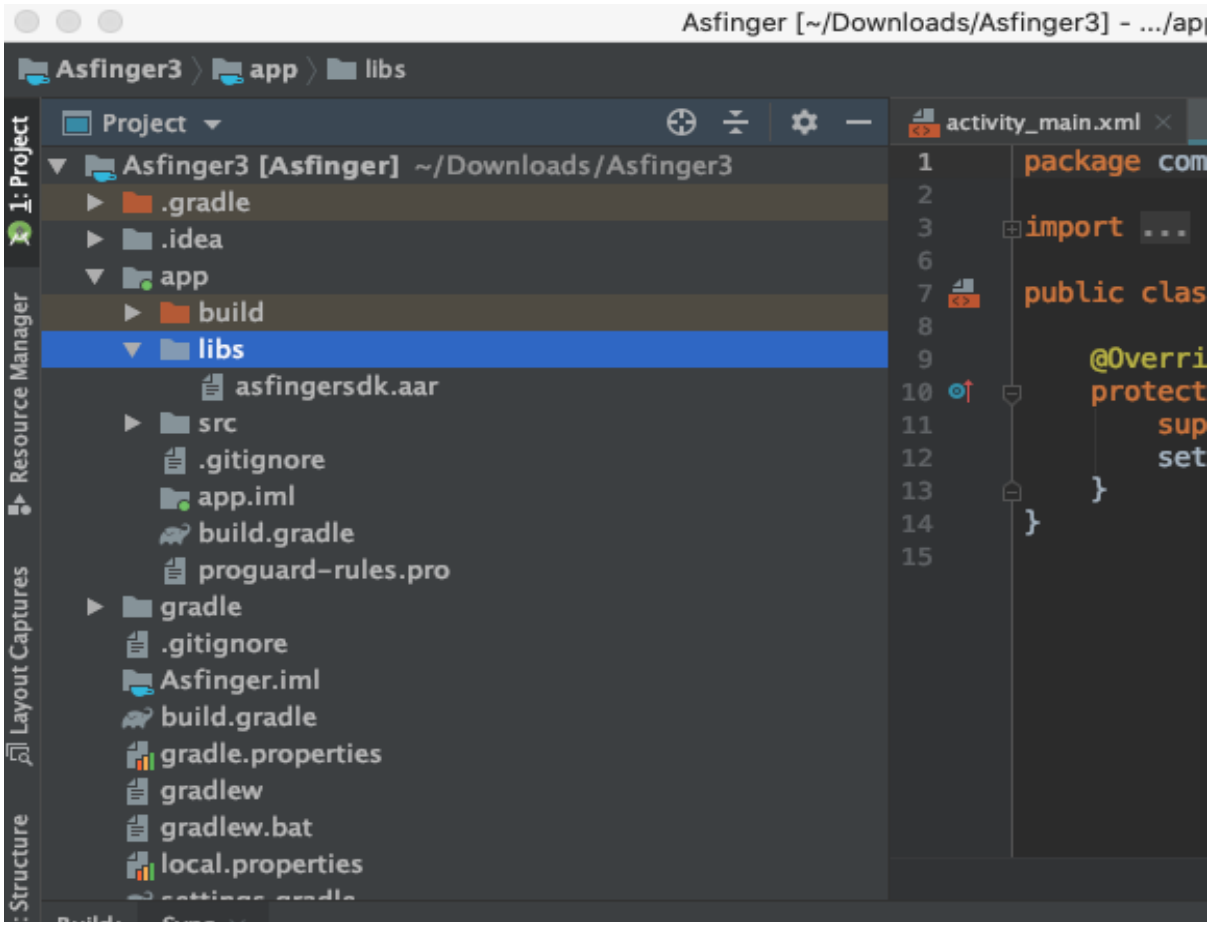

FIG. 1-1-3

3. Double-click to open "build.gradle" in the project (as shown in FIG. 1-1-4). Add the repositories and dependencies as shown in step 1,2 of FIG. 1-1-5, and click 'Sync Now' as shown in step 3.

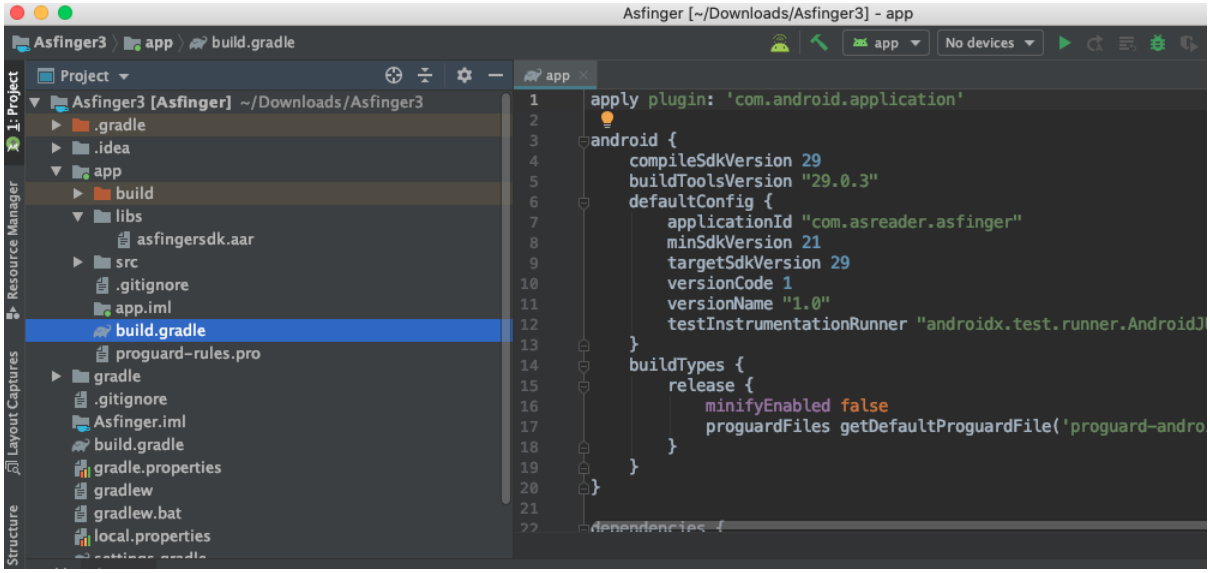

FIG. 1-1-4

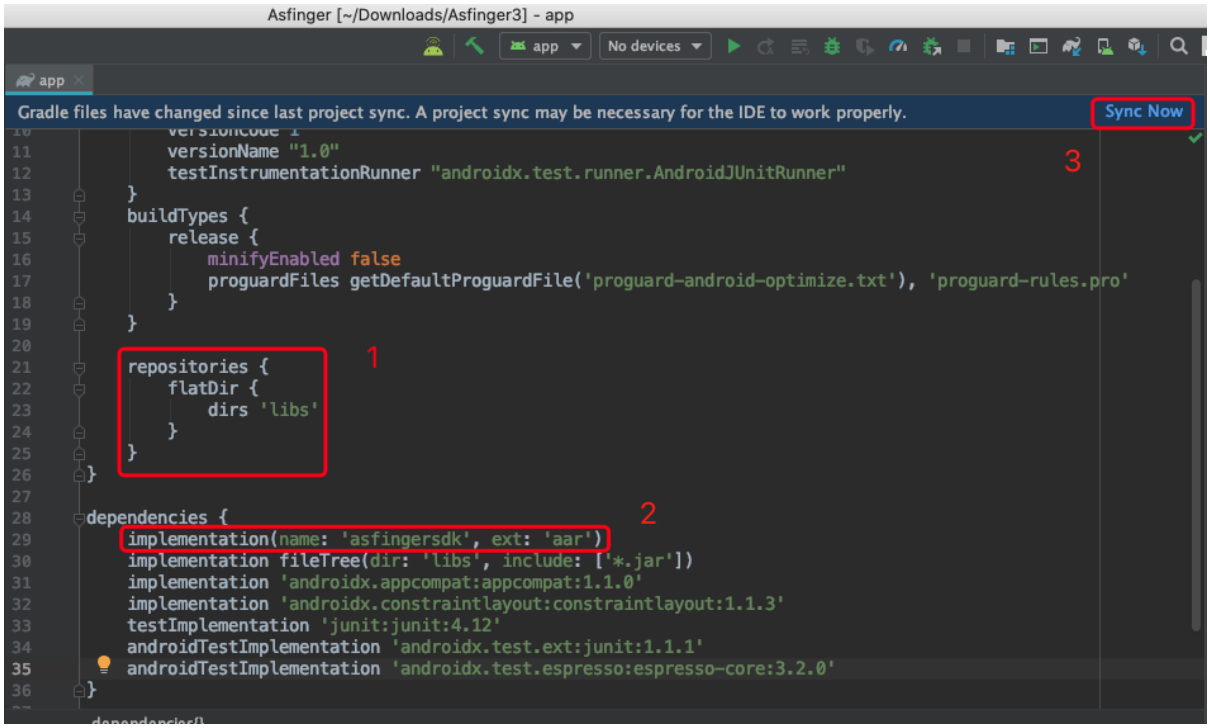

FIG. 1-1-5

4. Successful synchronization is shown in the red box below. Thus, SDK import is successful.

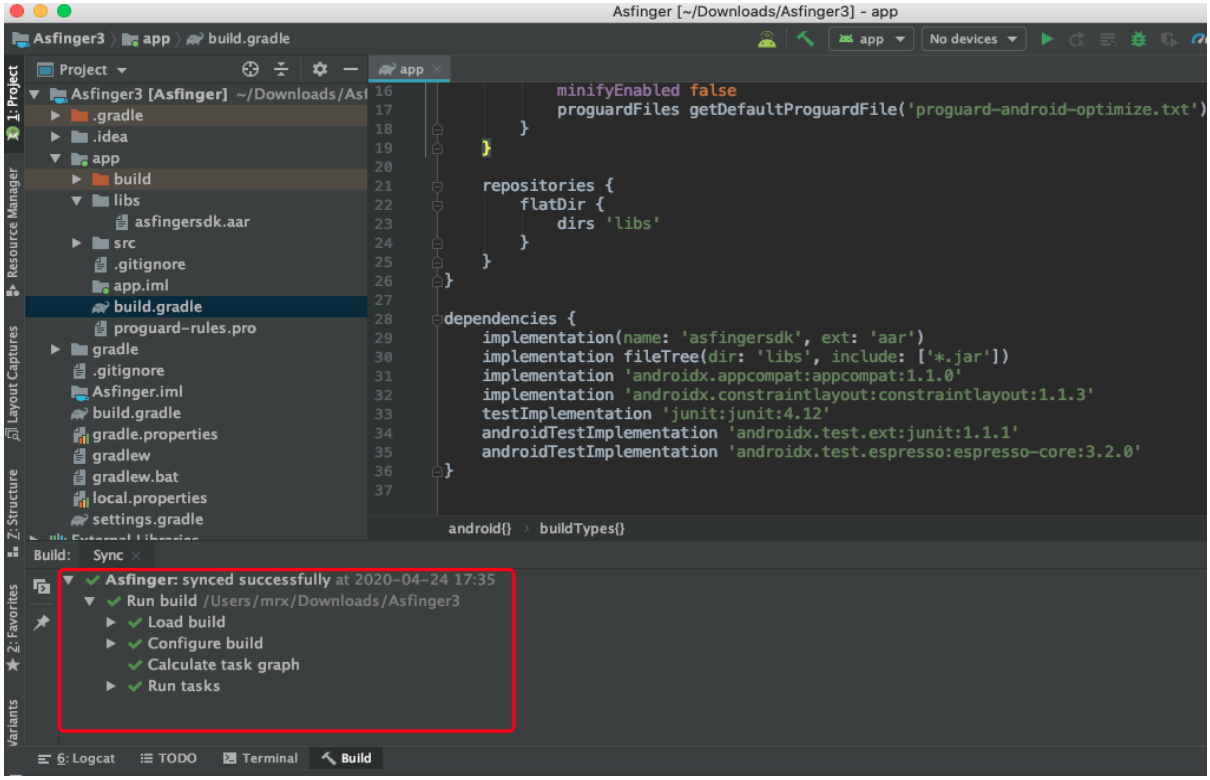

FIG. 1-1-6

1.2. SDK Usage

1. In the class to use the SDK, use the "import" statement to reference the AsFingerSDK (as shown in FIG. 1-2-1).

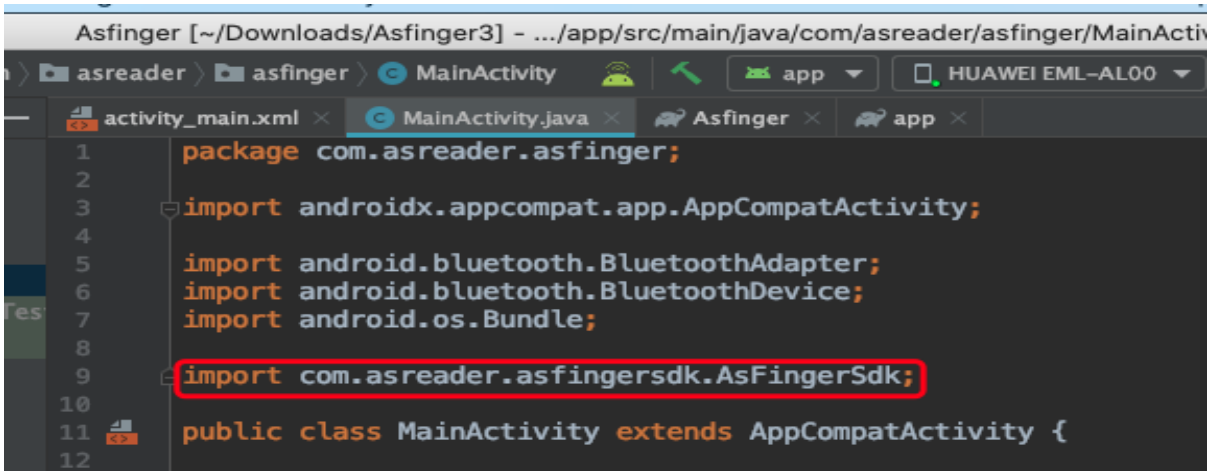

FIG. 1-2-1

2. Follow the steps below:
   Create an object for AsFingerSDK and pass in the Activity object (this) for it.

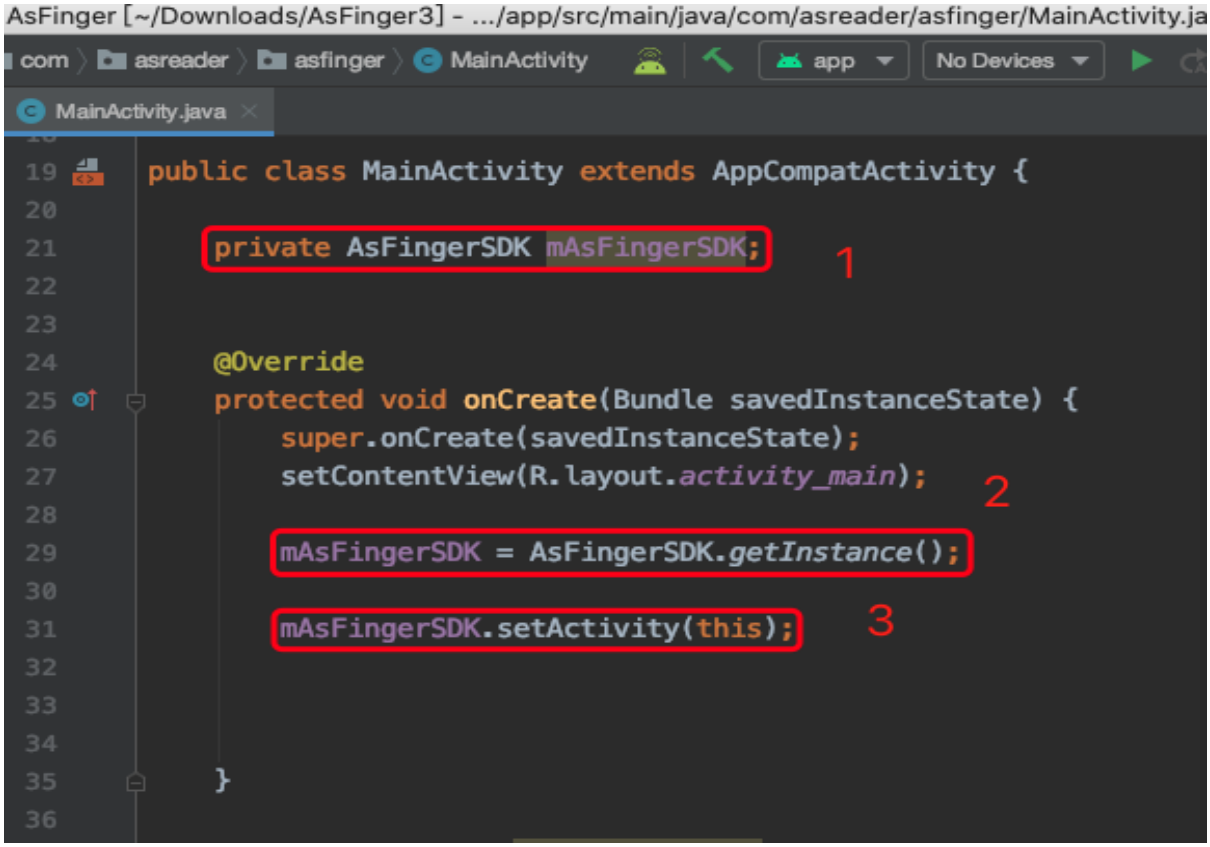

FIG. 1-2-2

3. Call API: Take connect(BluetoothDevice device, DeviceType type) as an example, follow the steps in FIG. 1-2-3:

1) Reference BluetoothAdapter, BluetoothDevice, and DeviceType library files by the "import" statement (Mark 1).
2) Get local Bluetooth adapter, BluetoothAdapter objects. (Mark3)
3) Define the Mac address that is used to get the BluetoothDevice object. (Mark4)
4) Call the function mAsFingerSDK.connect (device, DeviceType.spp) to connect. (Mark 5)

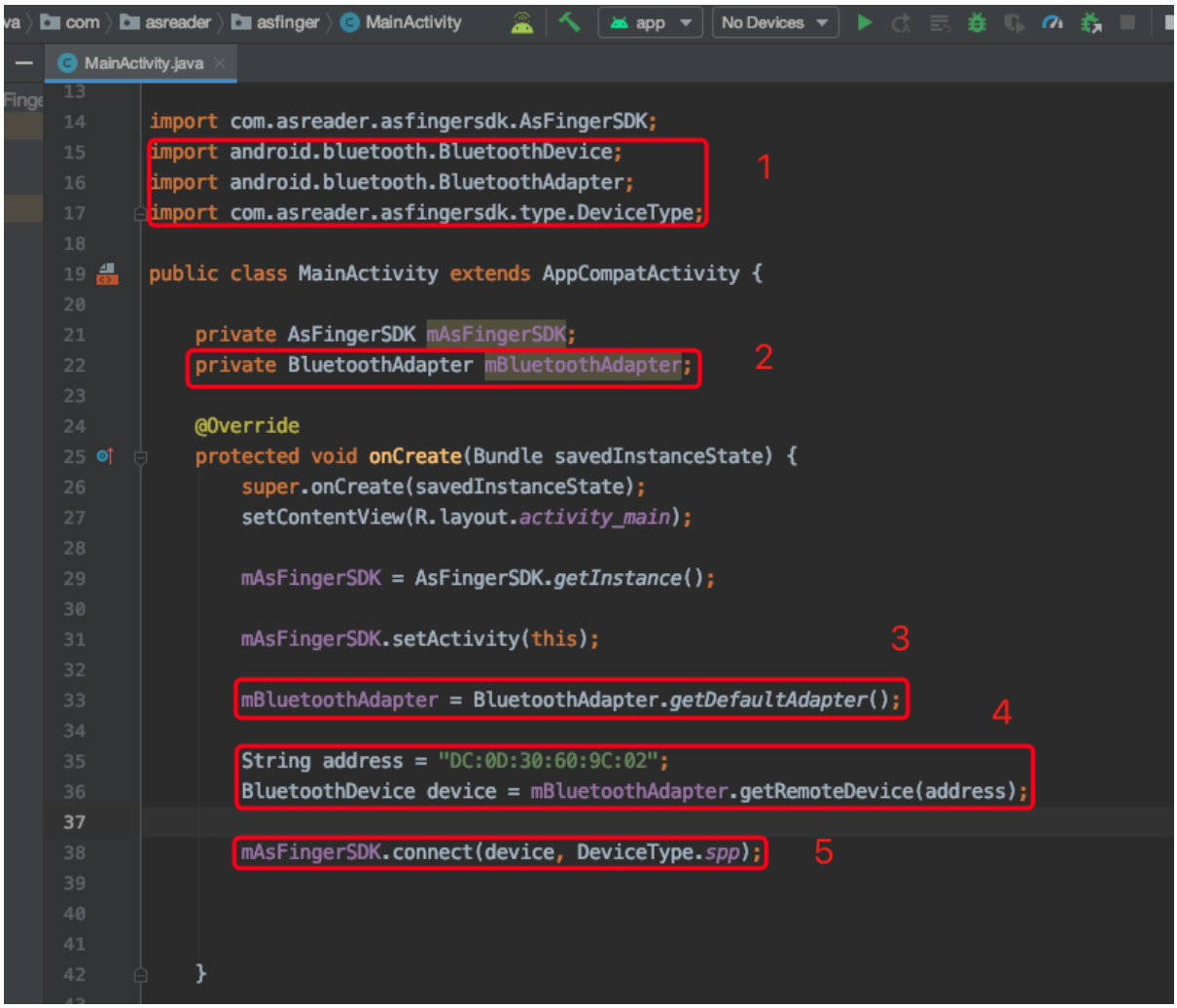

FIG. 1-2-3

Note: *Create the AsFingerEventListener object and override the function in the AsFingerEventListener interface.* FIG. 1-2-4*).*

```
private AsFingerEventListener mAsFingerEventListener = new AsFingerEventListener() {
    @Override
    public void onStateChanged(AsFingerConnectionType asFingerConnectionType) {}

    @Override
    public void receivedBarcodeData(byte[] bytes) {}

    @Override
    public void receivedBattery(int i) {}

    @Override
    public void receivedMacAddress(String s) {}

    @Override
    public void receivedFirmwareVersion(String s) {}

    @Override
    public void receivedData(byte[] bytes) {}

    @Override
    public void receivedDevice(BluetoothDevice bluetoothDevice) {}

    @Override
    public void receivedFoundDeviceFinished() {}
};
```

FIG. 1-2-4

Note: *Before connecting to the Bluetooth device, the function setAsFingerEventListener(AsFingerEventListener asFingerEventListener) must be called and passed in the AsFingerEventListener object to listen for the status of the Bluetooth connection and to receive data returned from the AsReader Finger-Type. (see the mark in FIG. 1-2-5)*

```
public class MainActivity extends AppCompatActivity {

    private AsFingerSDK mAsFingerSDK;
    private BluetoothAdapter mBluetoothAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mAsFingerSDK = AsFingerSDK.getInstance();

        mAsFingerSDK.setActivity(this);

        mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

        String address = "DC:0D:30:60:9C:02";
        BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(address);

        mAsFingerSDK.setAsFingerEventListener(mAsFingerEventListener);

        mAsFingerSDK.connect(device, DeviceType.spp);

    }
```

FIG. 1-2-5

Note: *Take the functions onStateChanged(AsFingerConnectionType connectionType) and receivedBarcodeData(byte[] barcodeData) for example: (see FIG. 1-2-6).*

```
@Override
public void onStateChanged(BluetoothConnectionType bluetoothConnectionType) {
    switch (bluetoothConnectionType){//Determine the connection status
        case Connected://Connected

            break;
        case Disconnected://Disconnected

            break;
    }
}

@Override
public void receivedBarcodeData(byte[] barcodeData) {
    // Once the function startScan() is called, the device returns the barcode data scanned.
    // Encoding
    // Charset mCharset = Charset.forName("ASCII");
    // Charset mCharset = Charset.forName("Shift_JIS");
    // Charset mCharset= Charset.forName("UTF-8");
    // String str = "";
    // try {
    //    str = new String(barcodeData, mCharset);
    //} catch (Exception e) {}
    // str = "6921734976727"
}
```

FIG. 1-2-6

1.3. Add project permissions

You need to define project permissions in the APP's AndroidManifest.xml file before using the SDK (see FIG. 1-3-1):

1. Allows applications to connect to paired Bluetooth devices. (Mark 1)
2. Allows applications to discover and pair Bluetooth devices. (Mark 2)
3. Allows applications to access the exact location (set only when APP adds "Scan Bluetooth Device" function). (Mark 3)
4. Allows the application to access the approximate location (set only when the APP adds the "Scan Bluetooth Device" function). (Mark 4)
5. Allows applications to write to external storage.
6. Allows applications to read from external storage. (Mark 6)

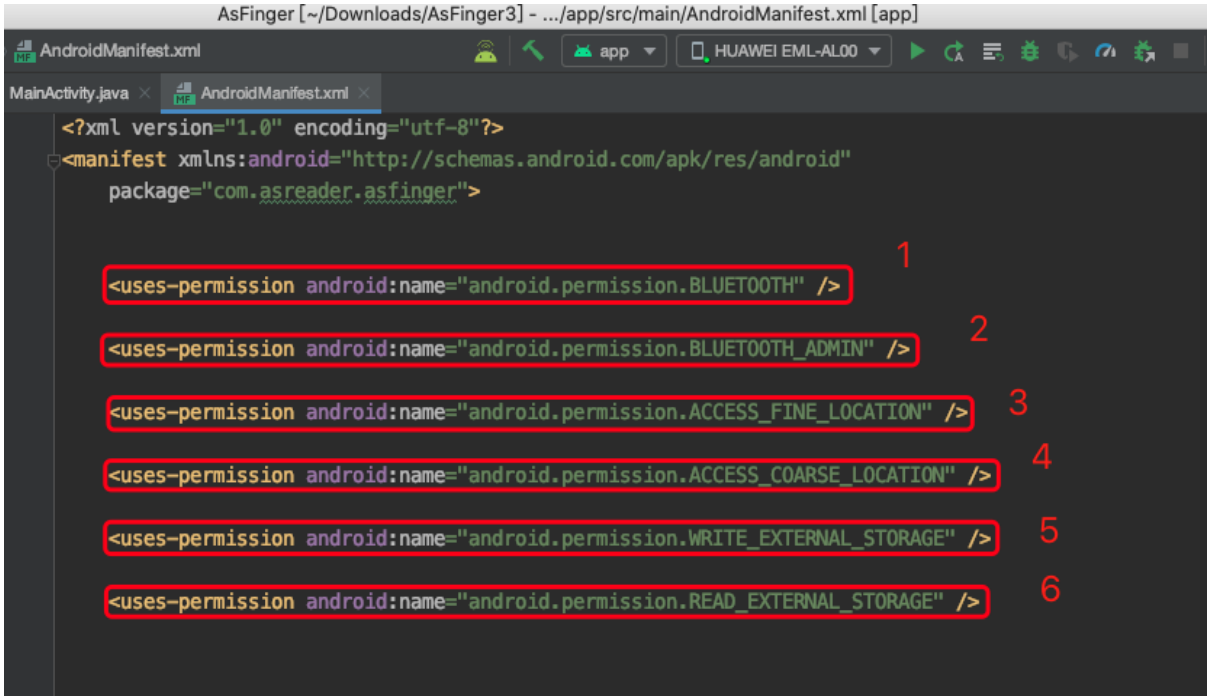

FIG. 1-3-1

2. Function Instructions

2.1. AsFingerSDK

2.1.1. getInstance

Function	AsFingerSDK *getInstance()*;		
Parameters	IN/OUT	Types	Descriptions
	OUT	AsFingerSDK	AsFingerSDK

■Function Description:
Gets the object of AsFingerSDK.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();

■Sample Code:
//Create and initialize an object of AsFingerSDK.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();

2.1.2. setAsFingerEventListener

Function	void setAsFingerEventListener(AsFingerEventListener asFingerEventListener);		
Parameters	IN/OUT	Types	Descriptions
asFingerEventListener	IN	AsFingerEventListener	AsFingerEventListener object (see 2.2).

■Function Description:
Sets AsFingerEventListener.
Note: This function needs to be called before the connect(BluetoothDevice device, DeviceType type) function is called.

■Sample Code:

```
//Create and initialize an object of AsFingerSDK.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//this is Activity type.
mAsFingerSDK.setActivity(this);
// Instantiate AsFingerEventListener
private AsFingerEventListener mAsFingerEventListener = new AsFingerEventListener() {

    @Override
    public void onStateChanged(AsFingerConnectionType connectionType) {
        For details, see 2.2.1
        }
    }

    @Override
    public void receivedBarcodeData(byte[] barcodeData) {
       For details, see 2.2.2
    }

    @Override
    public void receivedBattery(int battery) {
        For details, see 2.2.3
    }
```

```
@Override
public void receivedFirmwareVersion(String firmwareVersion) {
    For details, see 2.2.4
}

@Override
public void receivedData(byte[] data) {
    For details, see 2.2.5
}

@Override
public void receivedDevice(BluetoothDevice bluetoothDevice) {
For details, see 2.2.6
}

@Override
public void receivedFoundDeviceFinished() {
For details, see 2.2.7

}

};
//Sets AsFingerEventListener.
mAsFingerSDK.setAsFingerEventListener(mAsFingerEventListener);
```

2.1.3. connect

Function	void connect(BluetoothDevice device, DeviceType type);		
Parameters	IN/OUT	Types	Descriptions
device	IN	BluetoothDevice	The object of BluetoothDevice
type	IN	DeviceType	Connection mode, Enum type (see 3.1).

■Function Description:
This function is used to connect AsReader Finger-Type (via Bluetooth).
Once this function is called, the connection status of the AsReader Finger-Type is returned via invoking function onStateChanged(AsFingerConnectionType connectionType). (see 2.2.1).

■Sample Code:

```
//GAsFingerSDK.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//Get local Bluetooth adapter
BluetoothAdapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
//Create and initialize a Bluetooth address for the device.
String address = "DC:0D:30:60:9C:02";
//Get the BluetoothDevice object, device.
BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(address);
//Call the function connect(BluetoothDevice device, DeviceType type); to connect the device.
mAsFingerSDK.connect(device, DeviceType.spp);
```

2.1.4. setActivity

Function	void setActivity(Activity activity)		
Parameters	IN/OUT	Types	Descriptions
activity	IN	Activity	Context

■Function Description:
Set up a same context as the application lifecycle (such as MainActivity).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity (this);
```

2.1.5. disconnect

Function	void disconnectDevice();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
This function is used to disconnect from the AsReader Finger-Type.
Once this function is called, the connection status of the AsReader Finger-Type is returned via invoking function onStateChanged(AsFingerConnectionType connectionType). (see 2.2.1).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//Disconnect the Bluetooth connection from the AsReader Finger-Type.
mAsFingerSDK.disconnect();
```

2.1.6. getSdkVersion

Function	String getSdkVersion();		
Parameters	IN/OUT	Types	Descriptions
	OUT	String	The SDK version

■Function Description:
This function is used to get the SDK version of the connected AsReader Finger-Type.

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//sdkVersion is the SDK version of AsFingerSDK.
String sdkVersion= mAsFingerSDK.getSdkVersion();
```

2.1.7. getFirmwareVersion

Function	void getFirmwareVersion();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
This function is used to get the firmware version of the connected AsReader Finger-Type.
Once this function is called, the firmware version of the AsReader Finger-Type is returned via invoking function receivedFirmwareVersion(String firmwareVersion). (see 2.2.4).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//firmwareVersion Indicates the firmware version number of the AsReader Finger-Type.
String firmwareVersion=mAsFingerSDK. getFirmwareVersion ();
```

2.1.8. getBattery

Function	void getBattery();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
Get the battery remaining of the connected AsReader Finger-Type.
Once this function is called, the battery remaining of the AsReader Finger-Type is returned via invoking function receivedBattery(int battery). (see 2.2.3).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//Get the battery remaining of the AsReader Finger-Type.
mAsFingerSDK.getBattery();
```

2.1.9. startScan

Function	void startScan();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
Start to scan barcodes.
Once this function is called, the barcode data scanned by the AsReader Finger-Type is returned via invoking function receivedBarcodeData(byte[] barcodeData). (see 2.2.2).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//Start to scan.
mAsFingerSDK.startScan();
```

2.1.10. stopScan

Function	void stopScan();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
Stop scanning.

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//Stop scanning.
mAsFingerSDK.stopScan();
```

2.1.11. sendData

Function	void sendData(byte[] data);		
Parameters	IN/OUT	Types	Descriptions
data	IN	byte[]	Command data

■Function Description:
Send custom data to the AsReader Finger-Type.
Once this function is called, the data returned from the AsReader Finger-Type is returned via invoking function receivedData(byte[] data). (see 2.2.5).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
//Create and initialize byte[] type command.
byte[] data = new byte[]{0x03};
// Send command data to the AsReader Finger-Type.
mAsFingerSDK.sendData(data);
```

2.1.12. startDiscovery

Function	void startDiscovery();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
Start to search for the Bluetooth devices.
Once this function is called, the Bluetooth devices that be found by the AsReader Finger-Type is returned via invoking function receivedDevice(BluetoothDevice device). (see 2.2.6).

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// Search for the Bluetooth devices.
mAsFingerSDK.startDiscovery();
```

2.1.13. stopDiscovery

Function	void stopDiscovery();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
Stop searching for Bluetooth devices.
Once this function is called, the function receivedDevice(BluetoothDevice device) will be invoked to notify that the search is stopped. (see 2.2.7).

■Sample Code:

```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// Stop searching for Bluetooth devices.
mAsFingerSDK.stopDiscovery();
```

2.1.14. getPairedDevices

Function	Set<BluetoothDevice> getPairedDevices();		
Parameters	IN/OUT	Types	Descriptions
	OUT	Set<Bluetooth Device>	The list of the Bluetooth devices

■Function Description:
Get the list of the paired Bluetooth devices stored in the phone.
■Sample Code:

```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// The list of the paired Bluetooth devices
Set<BluetoothDevice> pairedDevices = mAsFingerSDK.getPairedDevices();
```

2.1.15. deviceType

Function	DeviceType deviceType();		
Parameters	IN/OUT	Types	Descriptions
	OUT	DeviceType	Connection mode, Enum (see 3.1).

■Function Description:
Get the Bluetooth connection mode of the current connected AsReader Finger-Type.

■Sample Code:

```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// The Bluetooth connection mode of the current connected device.
DeviceType mDeviceType = mAsFingerSDK.deviceType();
```

2.1.16. connectedDevice

Function	BluetoothDevice connectedDevice();		
Parameters	IN/OUT	Types	Descriptions
	OUT	BluetoothDevice	Bluetooth device

■Function Description:
Get the object of the AsReader Finger-Type that is currently connected.

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// Get the connected AsReader Finger-Type object.
BluetoothDevice mBluetoothDevice = mAsFingerSDK.connectedDevice();
```

2.1.17. autoConnect

Function	boolean autoConnect();		
Parameters	IN/OUT	Types	Descriptions
	OUT	boolean	true: Automatic connection enable false: Automatic connection disable

■Function Description:
Get the status value of whether the automatic connection to AsReader Finger-Type is enabled.

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// Get the status value of whether the automatic connection to AsReader Finger-Type is enabled.
boolean isAutoConnect = mAsFingerSDK. autoConnect();
```

2.1.18. setAutoConnect

Function	setAutoConnect(boolean isAutoConnect);		
Parameters	IN/OUT	Types	Descriptions
isAutoConnect	IN	boolean	true: Automatic connection enable false: Automatic connection disable

■Function Description:
Sets whether automatic connection to the AsReader Finger-Type is enabled.

■Sample Code:
```
//Create and initialize an AsFingerSDK object.
AsFingerSDK mAsFingerSDK = AsFingerSDK.getInstance();
//”this” is Activity type.
mAsFingerSDK.setActivity(this);
// Sets whether automatic connection to the AsReader Finger-Type is enabled.
mAsFingerSDK.setAutoConnect(true);
```

2.2. AsFingerEventListener

2.2.1. onStateChanged

Function	void onStateChanged(AsFingerConnectionType connectionType);		
Parameters	IN/OUT	Types	Descriptions
connectionType	IN	AsFingerConnectionType	Connection status, Enum (see 3.2).

■Function Description:
Listen to the change of the connection status of the AsReader Finger-Type.
This function will be called when function connect(BluetoothDevice device, DeviceType type) is called or disconnect the Bluetooth connection from the AsReader Finger-Type, or the Bluetooth connection is disconnected due to turning off the phone's Bluetooth. (see 2.1.3 and 2.1.5)

■Sample Code:

```
 @Override
 public void onStateChanged(AsFingerConnectionType connectionType) {
        //Determine the connection status
        switch (connectionType){
//Connected
            case Connected:

                break;
            //Disconnected
            case Disconnected:

                break;
        }
    }
```

2.2.2. receivedBarcodeData

Function	void receivedBarcodeData(byte[] barcodeData);		
Parameters	IN/OUT	Types	Descriptions
barcodeData	IN	byte[]	The scanned barcode data

■Function Description:
Receive the barcode data from the AsReader Finger-Type.
This function will be called when any barcode be scanned or function startScan() is called and any barcode be scanned. (see 2.1.9)

■Sample Code:

```
    @Override
    public void receivedBarcodeData(byte[] barcodeData) {
        // Once the function startScan() is called, the device returns the barcode data //scanned.
        // Encoding
        // Charset mCharset = Charset.forName("ASCII");
        // Charset mCharset = Charset.forName("Shift_JIS");
        // Charset mCharset= Charset.forName("UTF-8");
        // String str = "";
        // try {
        // str = new String(barcodeData, mCharset);
        //} catch (Exception e) {}
        //Sample data: str = “6921734976727”
    }
```

2.2.3. receivedBattery

Function	void receivedBattery(int battery);		
Parameters	IN/OUT	Types	Descriptions
battery	IN	Int	Battery remaining value of the AsReader Finger-Type. Range: 1-100

■Function Description:
Receive the remaining battery value of the AsReader Finger-Type.
This delegate function will be called when function getBattery() is called. (see 2.1.8)

■Sample Code:

```
    @Override
    public void receivedBattery(int battery) {
        // Once the function getBattery() is called, the device returns the data for the
          // battery's remaining.
        // The sample of the returned data: battery = 60
    }
}
```

2.2.4. receivedFirmwareVersion

Function	void receivedFirmwareVersion (String firmwareVersion);		
Parameters	IN/OUT	Types	Descriptions
firmwareVersion	IN	String	The firmware version of AsReader Finger-Type

■Function Description:
Receive the firmware version of the AsReader Finger-Type.
This function will be called when function getFirmwareVersion() is called. (see 2.1.7)

■Sample Code:

```
    @Override
    public void receivedFirmwareVersion(String firmwareVersion) {
        // Once the function getFirmwareVersion() is called, the device returns the data //for
        thefirmware version of the device.
        //The sample of the returned data: firmwareVersion = "FW_1.0.0"
    }

}
```

2.2.5. receivedData

Function	void receivedData(byte[] data);		
Parameters	IN/OUT	Types	Descriptions
data	IN	byte[]	The whole data returned from AsReader Finger-Type

■Function Description:
Receive the whole data returned from AsReader Finger-Type.
This function will be called when any of the functions sendData(byte[] data); (see 2.1.11), getFirmwareVersion(); (see 2.1.7), getBattery(); (see 2.1.8), is called. This function also will be called when the function startScan() (see 2.1.9) is call and there are some barcodes be scanned.

■Sample Code:

```
    @Override
    public void receivedData(byte[] data) {
        // Once the function One of all the interfaces to send commands is called or
        //sendData(byte[] data) is called the device returns
        // the Corresponding data
        // Encoding
        // Charset mCharset = Charset.forName("ASCII");
        // Charset mCharset = Charset.forName("Shift_JIS");
        // Charset mCharset= Charset.forName("UTF-8");
        // String str = "";
        // try {
        //      str = new String(data, mCharset);
        //} catch (Exception e) {}
        //Sample data: str = “6921734976727”

    }
};
```

2.2.6. receivedDevice

Function	void receivedDevice(BluetoothDevice bluetoothDevice);		
Parameters	IN/OUT	Types	Descriptions
bluetoothDevice	IN	BluetoothDevice	Bluetooth devices

■Function Description:
Received the Bluetooth devices that be searched for by AsFingerSDK.
This function will be called when function startDiscovery() is called or any Bluetooth device is searched for. (see 2.1.12)

■Sample Code:

```
    @Override
    public void receivedDevice(BluetoothDevice bluetoothDevice) {
        // Once the function One of all the interfaces to send commands is called or
        // startDiscovery () is called the device returns BluetoothDevice
        // bluetoothDevice are the Bluetooth devices that be searched for.
    });

}
```

2.2.7. receivedFoundDeviceFinished

Function	void receivedFoundDeviceFinished();		
Parameters	IN/OUT	Types	Descriptions

■Function Description:
Receive the message that searching for Bluetooth device is stopped.
This function will be called when function stopDiscovery() is called. (see 2.1.13)

■Sample Code:

```
@Override
public void receivedFoundDeviceFinished(){
    // Once the function One of all the interfaces to send commands is called or
      //stopDiscovery () is called
});
```

3. Enum Type

3.1. DeviceType

Definition	Descriptions
unknow = 0	no connection
ble = 1	ble connection
spp = 2	spp connection

3.2. AsFingerConnectionType

Definition	Descriptions
connected = 0	connected
connecting = 1	connecting
disconnected = 2	disconnected