Demo APP for MicroLifeDeviceSDK (Body Temperature / Thermometer) - Android

Table of Contents

Chapter 1 Development Environment

Chapter 2 Entry Point and Bluetooth LE Protocol

Chapter 3 Bluetooth LE Protocol & APIs

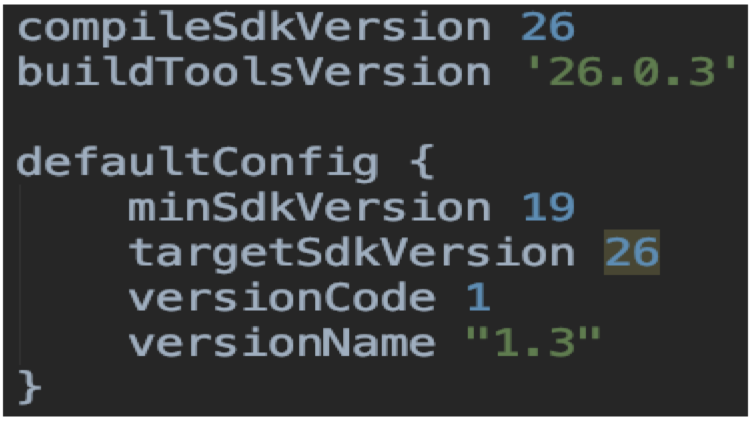
Chapter 4 Thermometer APIs

Chapter 5 User Interface and Functionality

Chapter 6 Demo App

Chapter 1 Development Environment

The supported SDK version is as follow:

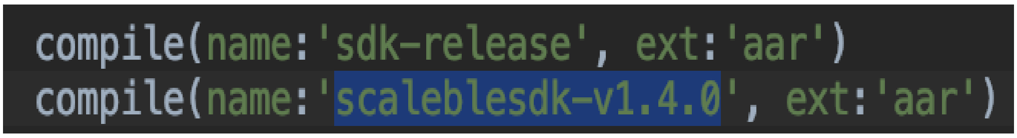


```
compileSdkVersion 26
buildToolsVersion '26.0.3'

defaultConfig {
    minSdkVersion 19
    targetSdkVersion 26
    versionCode 1
    versionName "1.3"
}
```

1.1. Add the library “sdk-release.arr” into the “libs” directory.

1.2. In the “build.gradle”, add the description as bellows:

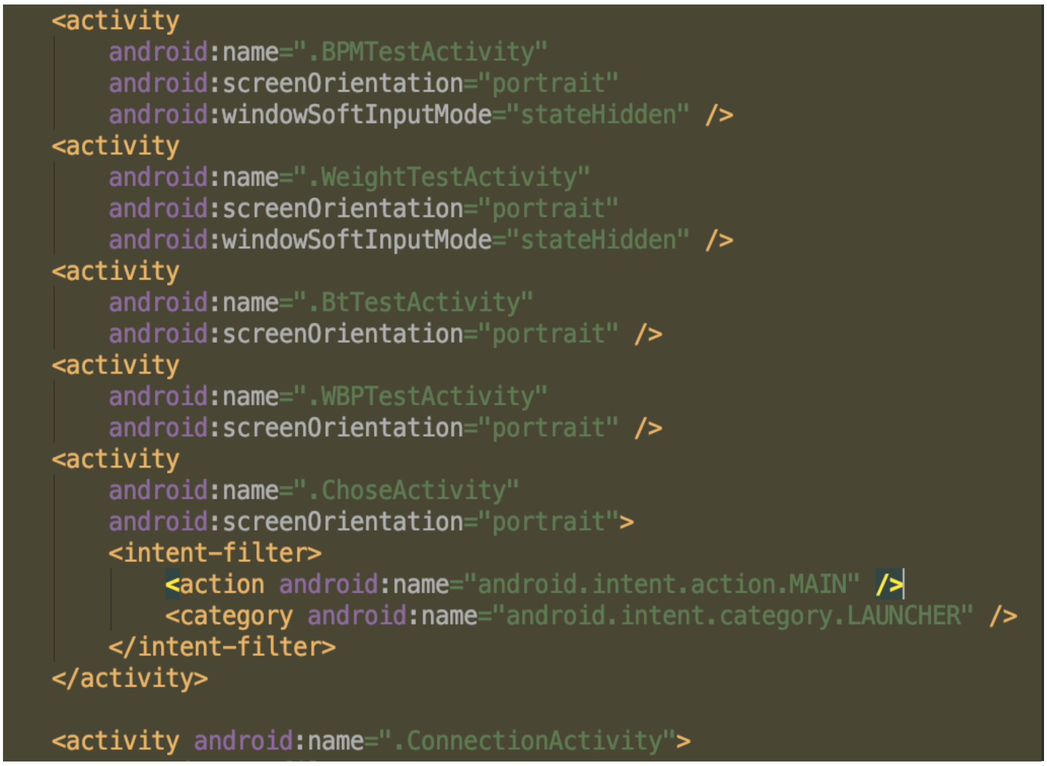


```
compile(name:'sdk-release', ext:'aar')
compile(name:'scaleblesdk-v1.4.0', ext:'aar')
```

Chapter 2 Entry Point and Bluetooth LE Protocol

The “ChoseActivity” is the entry point of the sample application.

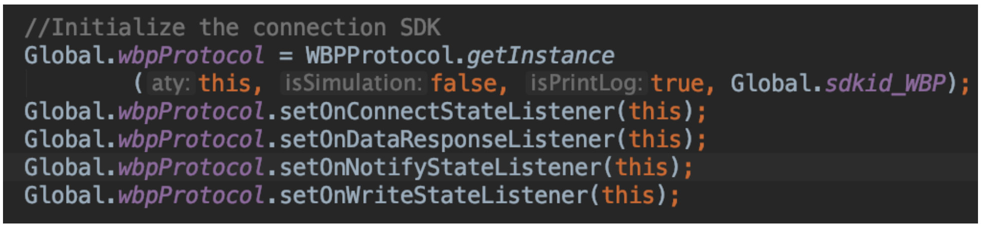
The “BtTestActivity” is dedicated to Thermometer.



```
<activity
    android:name=".BPMTestActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="stateHidden" />
<activity
    android:name=".WeightTestActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="stateHidden" />
<activity
    android:name=".BtTestActivity"
    android:screenOrientation="portrait" />
<activity
    android:name=".WBPTestActivity"
    android:screenOrientation="portrait" />
<activity
    android:name=".ChoseActivity"
    android:screenOrientation="portrait">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity android:name=".ConnectionActivity">
```

2.1 Initialize the instance “wbpProtocol”. This is to fulfill Bluetooth LE features and connection sequence.



```
//Initialize the connection SDK
Global.wbpProtocol = WBPProtocol.getInstance
        ( aty: this,  isSimulation: false,  isPrintLog: true, Global.sdkid_WBP);
Global.wbpProtocol.setOnConnectStateListener(this);
Global.wbpProtocol.setOnDataResponseListener(this);
Global.wbpProtocol.setOnNotifyStateListener(this);
Global.wbpProtocol.setOnWriteStateListener(this);
```

2.1.1 The “setOnConnectStateListener()” is to get the connection status of device.

2.1.2 The “setOnDataResponseListener()” is to get the response from device.

2.1.3 The “setOnNotifyStateListener()” is to get the data which is response from device.

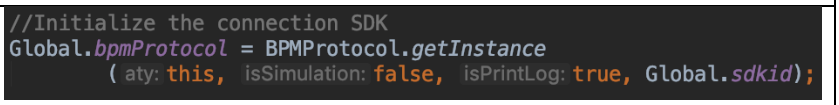
2.1.4 The “setOnWriteStateListener()” is to get the data which is sent to device.

2.2 The “isEnableBt()” or “ isSupportBluetooth() is to check if the smartphone’s Bluetooth is enabled or not. The “isSupportBluetooth()” will prompt a warning message to inform user to turn on Bluetooth if it is disabled.

Chapter 3 Bluetooth LE Protocol & APIs

3.1. Instance of Bluetooth LE Protocol：

3.1.1. Interface：

	public static * Protocol getInstance(Activity aty, boolean isSimulation, boolean isPrintLog, String sdkid)
Definition	Initialize Bluetooth LE Protocol for WatchBP Home A device
Parameter	Activity aty：name of activity or this boolean isSimulation： is simulator or device boolean isPrintLog：is printing log or not。 String sdkid：SDK ID of designated device
	

```
//Initialize the connection SDK
Global.bpmProtocol = BPMProtocol.getInstance
        ( aty: this,  isSimulation: false,  isPrintLog: true, Global.sdkid);
```

3.2. Connection State and Result：

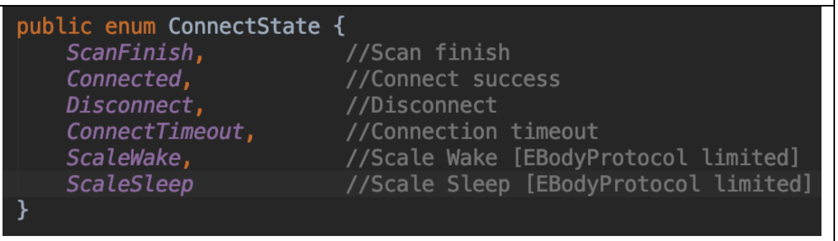
3.2.1. Interface：

	public void setOnConnectStateListener(OnConnectStateListener l)
Definition	The “setOnConnectStateListener()” is to get the connection status of device.

3.2.2. Delegate：

	void onBtStateChanged(boolean isEnable)
Definition	The “onBtStateChanged()” is to monitor the state of Enabled or Disabled.

	void onScanResult(String mac, String name, int rssi)
Definition	This is to get Bluetooth information of devices which discovered in the vicinity.
Parameter	macAddress: MAC of device name: device name RSSI: RSSI

	void onConnectionState(ConnectState state)
Definition	The “onConnectionState()” is to monitor the status of connection.
Parameter	

```
public enum ConnectState {
    ScanFinish,               //Scan finish
    Connected,                //Connect success
    Disconnect,               //Disconnect
    ConnectTimeout,           //Connection timeout
    ScaleWake,                //Scale Wake [EBodyProtocol limited]
    ScaleSleep                //Scale Sleep [EBodyProtocol limited]
}
```

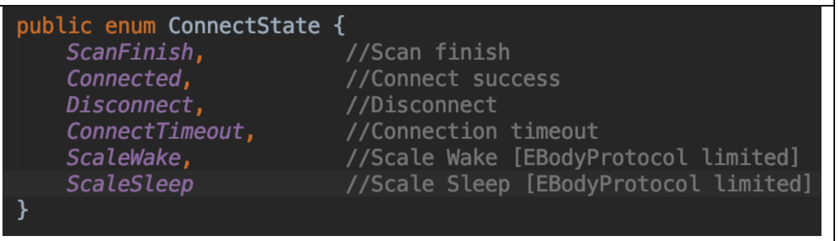
3.3. Device scanning or discovery :

3.3.1. Interface :

	public void startScan(int timeout)
Definition	The “startScan()” is for device scanning or discovery. The result will be shown with the “onScanResult”.
Parameter	int timeout

	public void stopScan()
Definition	Terminate the scanning process.

3.3.2. Delegate :

	void onConnectionState(ConnectState state)
Definition	The “onConnectionState()” is to monitor the status of scanning.
Parameter	

```
public enum ConnectState {
    ScanFinish,               //Scan finish
    Connected,                //Connect success
    Disconnect,               //Disconnect
    ConnectTimeout,           //Connection timeout
    ScaleWake,                //Scale Wake [EBodyProtocol limited]
    ScaleSleep                //Scale Sleep [EBodyProtocol limited]
}
```

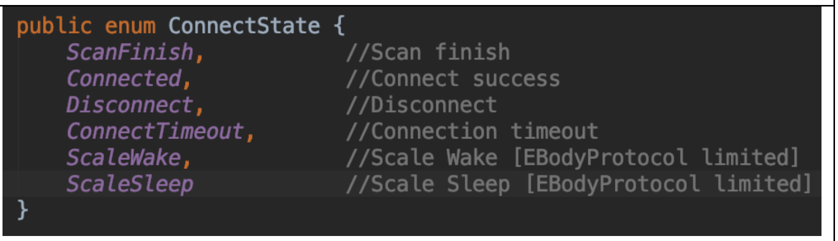
3.4. Connection :

3.4.1. Interface :

	public void connect(String macAddress)

Definition	Connect to device with MAC address.
Parameter	macAddress: MAC of device

3.4.2. Delegate：

	void onConnectionState(ConnectState state)
Definition	The “onConnectionState()” is to monitor the status of connection.
Parameter	

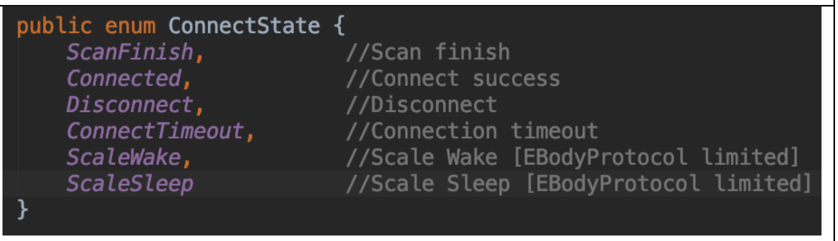
```
public enum ConnectState {
    ScanFinish,           //Scan finish
    Connected,            //Connect success
    Disconnect,           //Disconnect
    ConnectTimeout,       //Connection timeout
    ScaleWake,            //Scale Wake [EBodyProtocol limited]
    ScaleSleep            //Scale Sleep [EBodyProtocol limited]
}
```

3.5. Bonding：

3.5.1. Interface：

	public void bond(String macAddress)
Definition	Binding specified device by MAC
Parameter	macAddress: MAC of device

3.5.2. Delegate：

	void onConnectionState(ConnectState state)
Definition	The “onConnectionState()” is to monitor the status of connection.
Parameter	

```
public enum ConnectState {
    ScanFinish,           //Scan finish
    Connected,            //Connect success
    Disconnect,           //Disconnect
    ConnectTimeout,       //Connection timeout
    ScaleWake,            //Scale Wake [EBodyProtocol limited]
    ScaleSleep            //Scale Sleep [EBodyProtocol limited]
}
```

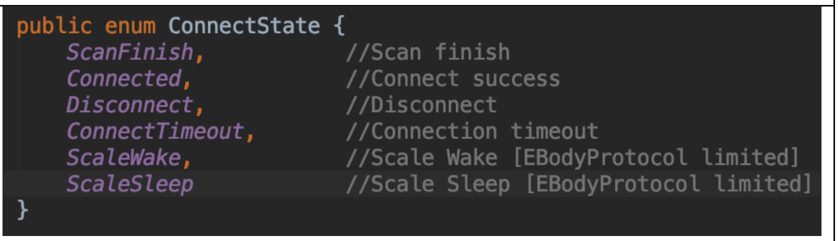
3.6. Disconnection：

3.6.1. Interface：

	public void disconnect()

Definition	Disconnect device.

3.6.2. Delegate：

	void onConnectionState(ConnectState state)
Definition	The “onConnectionState()” is to monitor the status of disconnection.
Parameter	

```
public enum ConnectState {
    ScanFinish,           //Scan finish
    Connected,            //Connect success
    Disconnect,           //Disconnect
    ConnectTimeout,       //Connection timeout
    ScaleWake,            //Scale Wake [EBodyProtocol limited]
    ScaleSleep            //Scale Sleep [EBodyProtocol limited]
}
```

Chapter 4 Thermometer APIs

4.1. Received device information：Once Bluetooth connects to the thermometer successfully, the first time the thermometer sends the information.

	void onResponseDeviceInfo(String macAddress, int workMode, float batteryVoltage)
Parameter	macAddress：macAddress description workMode：workMode description batteryVoltage：batteryVoltage description

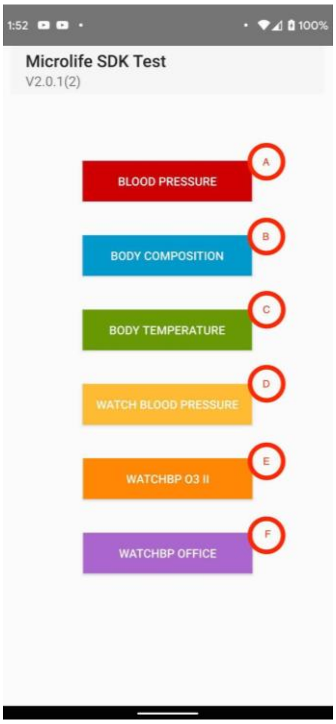
4.2. Received the current measurement result / data：

	void onResponseUploadMeasureData(ThermoMeasureData data)
Parameter	data：measurement result

Chapter 5 User Interface and Functionality

5.1 Getting Started：

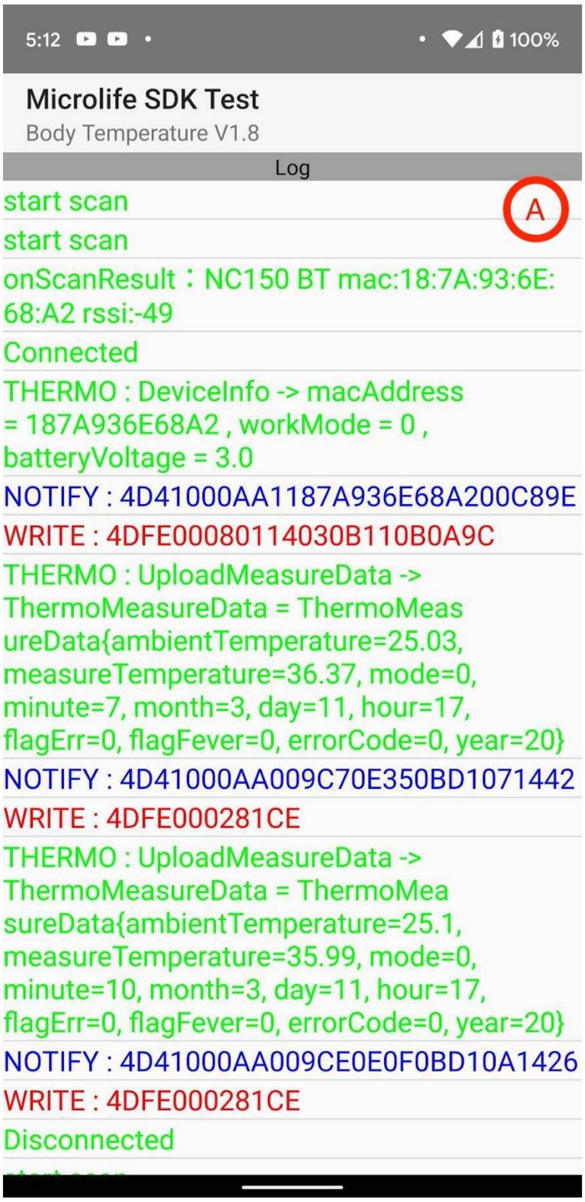
Start the app and then select the button “BODY TEMPERATURE” / “C” to communicate with the designate device Thermometer.



5.2 Operation Sequence：

The scanning (discovery) is automatically run to discover devices in the vicinity. If a device is bonded, it will be connected accordingly.

5.3 GUI Layout :



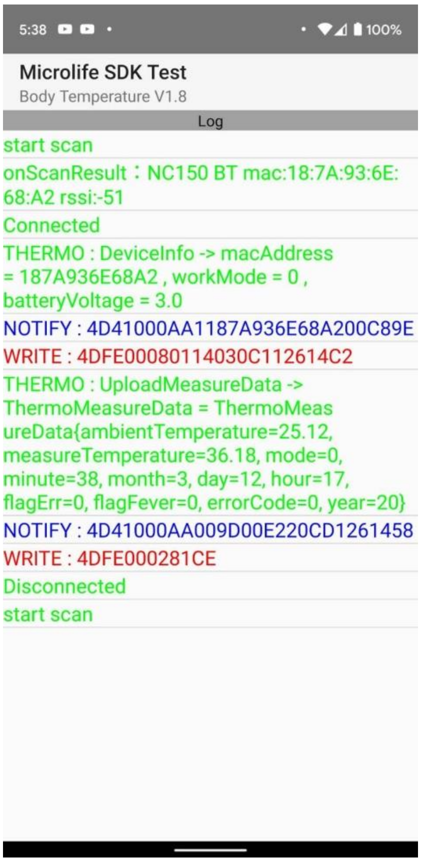
```
start scan
start scan
onScanResult：NC150 BT mac:18:7A:93:6E:
68:A2 rssi:-49
Connected
THERMO : DeviceInfo -> macAddress
= 187A936E68A2 , workMode = 0 ,
batteryVoltage = 3.0
NOTIFY : 4D41000AA1187A936E68A200C89E
WRITE : 4DFE00080114030B110B0A9C
THERMO : UploadMeasureData ->
ThermoMeasureData = ThermoMeas
ureData{ambientTemperature=25.03,
measureTemperature=36.37, mode=0,
minute=7, month=3, day=11, hour=17,
flagErr=0, flagFever=0, errorCode=0, year=20}
NOTIFY : 4D41000AA009C70E350BD1071442
WRITE : 4DFE000281CE
THERMO : UploadMeasureData ->
ThermoMeasureData = ThermoMea
sureData{ambientTemperature=25.1,
measureTemperature=35.99, mode=0,
minute=10, month=3, day=11, hour=17,
flagErr=0, flagFever=0, errorCode=0, year=20}
NOTIFY : 4D41000AA009CE0E0F0BD10A1426
WRITE : 4DFE000281CE
Disconnected
```

5.3.1 Region A：The log window is used to display information about communication handshake between App and device.

5.4 Refer to “BtTestActivity” from the demo application (sample code) to get more detailed.

Chapter 6 Demo App

6.1. Scanning (Discovery), Connection, Data (Readings) :



```
start scan
onScanResult：NC150 BT mac:18:7A:93:6E:
68:A2 rssi:-51
Connected
THERMO : DeviceInfo -> macAddress
= 187A936E68A2 , workMode = 0 ,
batteryVoltage = 3.0
NOTIFY : 4D41000AA1187A936E68A200C89E
WRITE : 4DFE00080114030C112614C2
THERMO : UploadMeasureData ->
ThermoMeasureData = ThermoMeas
ureData{ambientTemperature=25.12,
measureTemperature=36.18, mode=0,
minute=38, month=3, day=12, hour=17,
flagErr=0, flagFever=0, errorCode=0, year=20}
NOTIFY : 4D41000AA009D00E220CD1261458
WRITE : 4DFE000281CE
Disconnected
start scan
```

1. Once starts the demo App, it will be conducting a scanning / discovery process. The device named NC150 BT is displayed in the vicinity. The connection can be created by the MAC address “18:7A:93:6E:68:A2”.

2. The received data (included Device information & Readings) is as Blue part, and it can be decode in Green part. The following are the

(a) Device information:

THERMO : DeviceInfo -> macAddress = 187A936E68A2 , workMode = 0 , batteryVoltage = 3.0.

(b) Readings:

THERMO : UploadMeasureData -> ThermoMeasureData = ThermoMeasureData{ambientTemperature=25.12, measureTemperature=36.18, mode=0, minute=38, month=3, day=12, hour=17, flagErr=0, flagFever=0, errorCode=0, year=20}。

3. The Red part is ACK to inform device the process is complete.