

# MicroLifeDeviceSDK (3G BPM - Android)

## Table of Contents

- Chapter 1    Development Environment**
- Chapter 2    Entry Point and Bluetooth LE Protocol**
- Chapter 3    BPM APIs**
- Chapter 4    User Interface of Demo App**
- Chapter 5    Functionality of Demo App**

## Chapter1 Development Environment

1.1 The supported SDK version is as follow:

```
compileSdkVersion 26
buildToolsVersion '26.0.3'

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

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

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

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

## Chapter2 Entry Point and Bluetooth LE Protocol

The “ChoseActivity” is the entry point of the sample application. The “BPMTestActivity” is dedicated to the device of general Blood Pressure Monitor (Bluetooth LE).

```
<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=".WBPTTestActivity"
    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 “bpmProtocol”. This is to fulfill Bluetooth LE features and connection sequence.

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

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

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

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

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

2.2 The “isEnabledBt()” 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.

## Chapter3    BPM 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 BPM
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
	<pre>//Initialize the connection SDK Global.bpmProtocol = BPMProtocol.getInstance ( aty: this, isSimulation: false, isPrintLog: true, Global.sdkid);</pre>

### 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 isEnabled)
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	<pre> public enum ConnectState {     ScanFinish,           //Scan finish     Connected,            //Connect success     Disconnect,           //Disconnect     ConnectTimeout,       //Connection timeout     ScaleWake,            //Scale Wake [EBodyProtocol limited]     ScaleSleep            //Scale Sleep [EBodyProtocol limited] } </pre>

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

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

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

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

## 3.7. Read all history data from the BPM and synchronization :

### 3.7.1. Interface :

	void readHistorysOrCurrDataAndSyncTiming ()
Definition	Read all history data from the BPM and synchronization

### 3.7.2. Delegate :

	void onResponseReadHistory(DRecord dRecord)
Parameter	DRecord : History Measurement Data Model. CurrentAndMData : data.currentData & data.MData Model.

## 3.8. Clear all history data from BPM :

### 3.8.1. Interface :

	void clearAllHistorys()
Definition	Clear all history data from BPM

### 3.8.2. Delegate :

	public void onResponseClearHistory(boolean isSuccess)
--	---



Parameter	isSuccess : True or False
-----------	---------------------------

### 3.9. Disconnect the Bluetooth with BPM :

#### 3.9.1. Interface :

	public void disconnectBPM()
Definition	Disconnect the Bluetooth with BPM

#### 3.9.2. Delegate :

	void onConnectionState(ConnectState state)
Definition	The “onConnectionState()” is to monitor the status of connection with BPM.
Parameter	<pre> public enum ConnectState {     ScanFinish,           //Scan finish     Connected,            //Connect success     Disconnect,           //Disconnect     ConnectTimeout,       //Connection timeout     ScaleWake,            //Scale Wake [EBodyProtocol limited]     ScaleSleep            //Scale Sleep [EBodyProtocol limited] } </pre>

### 3.10. Read last one data from BPM :

#### 3.10.1. Interface :

	void readLastData ()
Definition	Read last one data from BPM

#### 3.10.2. Delegate :

	void onResponseReadLastData(CurrentAndMData dRecord,int historyMeasuremeNumber,int userNumber,int MAMState,boolean isNoData)
Parameter	<p>CurrentAndMData dRecord : last 1 data</p> <p>historyMeasuremeNumber : History Measurement Number: The range of Memory Set = 0~255</p> <p>userNumber : User Number : User1 = 1, User2 = 2, Guest = 3</p>

	MAMState : MAM state : 0=MAM disable, 1=Weight off, 2=Weight on, 3=Light off, 4=Light on. isNoData : True or False
--	--

### 3.11. Write a new user ID & Age to BPM :

#### 3.11.1. Interface :

	void writeUserData(String ID, int age)
Definition	Write a new user ID & Age to BPM
Parameter	ID : User ID with 11 bytes (maximal) Age : User age from 18 to 80

#### 3.11.2. Delegate :

	void onResponseWriteUser(boolean isSuccess)
Parameter	isSuccess : True or False

### 3.12. Clear last one data of BPM :

#### 3.12.1. Interface :

	void clearLastData ()
Definition	Clear last one data of BPM

#### 3.12.2. Delegate :

	void onResponseClearLastData(boolean isSuccess)
Parameter	isSuccess : True or False

### 3.13. Read user ID and version data from BPM :

#### 3.13.1. Interface :

	public void readUserAndVersionData ()
Definition	Read user ID and version data from BPM

#### 3.13.2. Delegate :

	<code>void onResponseReadUserAndVersionData(User user, VersionData versionData)</code>
Parameter	User : User Info VersionData : Device Info

## Chapter4 User Interface of Demo App

### 4.1. Getting Started :

Start the app and then select the button “Blood Pressure” / “A” to

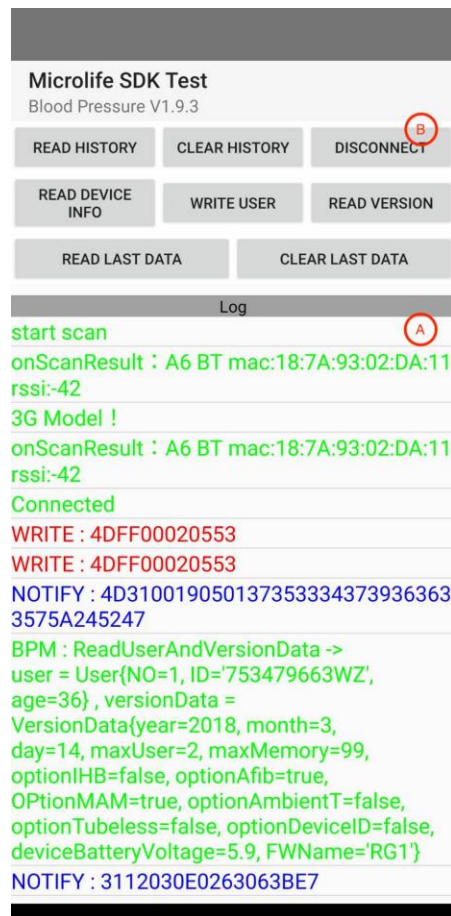


communicate with the designate device of general Blood Pressure.

### 4.2. Operation Sequence :

- 4.2.1. The scanning (discovery) is automatically run to discover devices in the vicinity.
- 4.2.2. If the pairing is Just Works, it will be connected accordingly. If not, it is a Numeric Comparison process.

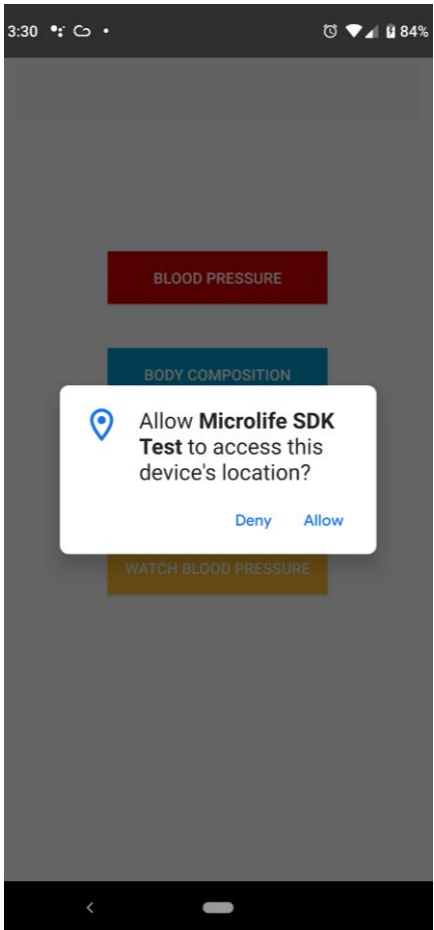
#### 4.3. Operating Interface and Sequence :



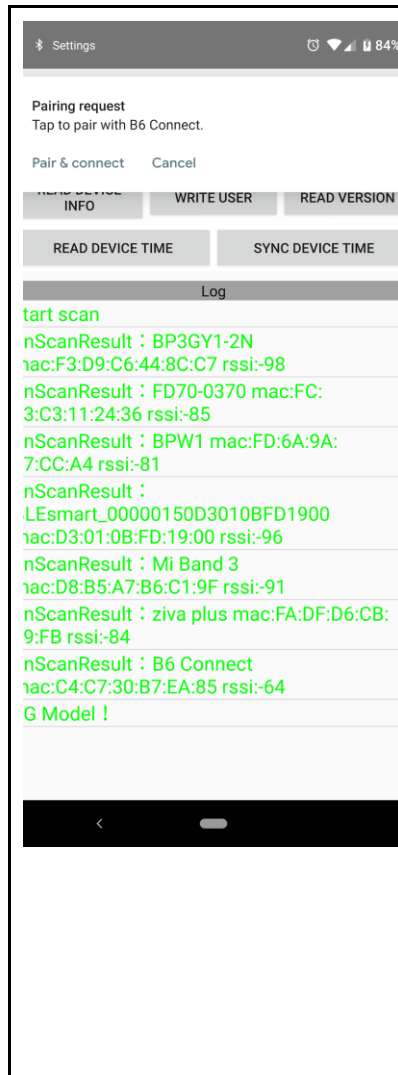
- 4.3.1. Region A : The log window is used to display information about communication handshake between App and device.
- 4.3.2. Region B : This part is to communicate with the device of general Blood Pressure Monitor by different functions / commands such as data transferring, synchronization and so on.
- 4.3.3. Refer to “BPMTTestActivity” from the demo application (sample code) to get more detailed.

## Chapter5    Functionality of Demo App

### 5.1.     Bluetooth authorization :

 A screenshot of an Android application interface. At the top, the status bar shows the time 3:30, signal strength, Wi-Fi, and battery at 84%. The app's main screen has a grey background with three buttons: a red 'BLOOD PRESSURE' button, a teal 'BODY COMPOSITION' button, and a brown 'WATCH BLOOD PRESSURE' button. A white system dialog box is centered on the screen, asking 'Allow Microlife SDK Test to access this device's location?' with 'Deny' and 'Allow' options. The bottom of the screen shows the Android navigation bar with a back arrow and a home button.	<p>1. Request for Bluetooth permission.</p>
---	---

## 5.2. Pairing / Bonding :



1. The pairing is Just Works.

2. There is a message to confirm the pairing bonding procedure between device and cellphone if they haven't bonded yet.

3. Once the procedure is done, choose any function/ command to do communication with general Blood Pressure Monitor.

4. The **green** part is from "onScanResult".

### 5.3. Command: Write a new user ID to BPM



1. The command “WRITE USER” is to write a new user ID to BPM.

2.The log “WRITE : write user “ and “Write : userID:753479663WZ” are indicated that the App sends a command with an user ID. The ID is made up of ASCII code.

3. The log “BPM:WriteUser -> isSuccess = true” means that the writing/ sending procedure is successful.

4. The **red** part is the command and communication protocol that is sent to device. The **blue** part is notification with the raw data from BPM via Bluetooth.



#### 5.4. Command: Read user ID and version data from BPM

 <p><b>Microlife SDK Test</b> Blood Pressure V1.9.3</p> <p>READ HISTORY CLEAR HISTORY DISCONNECT</p> <p>READ DEVICE INFO WRITE USER READ VERSION</p> <p>READ LAST DATA CLEAR LAST DATA</p> <p>Log</p> <pre> start scan onScanResult : A6 BT mac:18:7A:93:02:DA:11 rssi:-50 3G Model ! onScanResult : A6 BT mac:18:7A:93:02:DA:11 rssi:-50 Connected WRITE : 4DFF00020553 WRITE : 4DFF00020553 NOTIFY : 4D310019050133303739313930333 8564B005247 BPM : ReadUserAndVersionData -&gt; user = User{NO=1, ID='307919038VK', age=0} , versionData = VersionData{year=2018, month=3, day=14, maxUser=2, maxMemory=99, optionIHB=false, optionAfib=true, OPtionMAM=true, optionAmbientT=false, optionTubeless=false, optionDeviceID=false, deviceBatteryVoltage=5.9, FWName='RG1'} NOTIFY : 3112030E0263063BA9 </pre>	<p>1. The command “Read user ID and version data” is to get user ID and device information as below.</p> <p>2.The log “BPM :</p> <p>ReadUserAndVersionData -&gt;</p> <p>user = User{NO=1, ID='307919038VK', age=0}</p> <p>VersionData :</p> <p>VersionData{year=2018, month=3, day=14, maxUser=2, maxMemory=99, optionIHB=false, optionAfib=true, optionMAM=true, optionAmbientT=false, optionTubeless=false, optionDeviceID=false, deviceBatteryVoltage=5.9, FWName='RG1'}” is included ID, Age and Device information (Battery voltage, Firmware version and so on).</p>
--	--

## 5.5. Command: Read history data from BPM

	<p>1. The button “READ HISTORY” is to get history data.</p> <p>2. The <b>red</b> part is the command and communication protocol that is sent to device. The <b>blue</b> part is notification with the raw data from BPM via Bluetooth. The <b>green</b> part is the result after decoding with the raw data.</p> <p>3. The log “BPM : ReadHistory -&gt; DRecord{mode=0, noOfCurrentMeasurement=3, historyMeasuremeNumber=11, userNumber=1, MAMState=4, currentData=[null, null, null], MData=[CurrentAndMData{systole=118, dia=77, hr=75, day=19, hour=17, month=1, minute=36, MAM=0, arr=false, year=2021, cuffokr=2, IHB=false, AFib=false, isFor3G=true, resultCode=0,... }]}” is included BP readings.</p> <p>3. Each BP reading has its own measurement date &amp; time, systolic, diastolic and pulse. For instance, the above-mentioned reading is DateTime 2021/1/19,</p>
--	---

<div><div></div><div><div><div>Microlife SDK Test</div><div>Blood Pressure V1.9.3</div></div><div><div>READ HISTORY</div><div>CLEAR HISTORY</div><div>DISCONNECT</div></div><div><div>READ DEVICE INFO</div><div>WRITE USER</div><div>READ VERSION</div></div><div><div>READ LAST DATA</div><div>CLEAR LAST DATA</div></div><div>Log</div><div>resultCode=0, usual=false, diagnostic=false, AM=false, PM=false, deviceMode=49), CurrentAndMData{systole=116, dia=75, hr=73, day=15, hour=11, month=4, minute=38, MAM=0, arr=false, year=2021, cuffokr=2, IHB=false, AFib=false, isFor3G=true, resultCode=0, usual=false, diagnostic=false, AM=false, PM=false, deviceMode=49), CurrentAndMData{systole=123, dia=71, hr=71, day=18, hour=15, month=5, minute=59, MAM=0, arr=false, year=2021, cuffokr=2, IHB=false, AFib=false, isFor3G=true, resultCode=0, usual=false, diagnostic=false, AM=false, PM=false, deviceMode=49), CurrentAndMData{systole=123, dia=73, hr=70, day=18, hour=16, month=5, minute=4, MAM=3, arr=true, year=2021, cuffokr=2, IHB=false, AFib=false, isFor3G=true, resultCode=0, usual=false, diagnostic=false, AM=false, PM=false, deviceMode=49}], measureMode=false}</div><div>NOTIFY : 4B26157B4747524F3B157B4946525004D560</div></div></div>	Sys 118, Dia 77, Pulse 75.
--	----------------------------

