

# **Demo APP for MicroLifeDeviceSDK (Android)**

## **Table of Contents**

<b>Chapter 1</b>	<b>Development Environment</b>
<b>Chapter 2</b>	<b>Entry Point and Bluetooth LE Protocol</b>
<b>Chapter 3</b>	<b>Oxygen APIs</b>
<b>Chapter 4</b>	<b>User Interface of Demo App</b>
<b>Chapter 5</b>	<b>Functionality of Demo App</b>

## Chapter1 Development Environment

1.1 The supported SDK version is as follow:

```
android {  
    compileSdkVersion 32  
    buildToolsVersion '31.0.0'  
  
    defaultConfig {  
        minSdkVersion 21  
        //noinspection OldTargetApi  
        targetSdkVersion 32  
    }  
}
```

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

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

```
dependencies {  
  
    implementation fileTree(dir: 'libs', include: ['*.jar'])  
    implementation (name:'sdk-release', ext:'aar')  
}
```

## Chapter2 Entry Point and Bluetooth LE Protocol

The “ChoseActivity” is the entry point of the sample application. The “SPO2TestActivity” is dedicated to the device oxygen (Bluetooth LE).

```
<activity
    android:name=".BPMTestActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="stateHidden"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".WeightTestActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="stateHidden"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".BtTestActivity"
    android:screenOrientation="portrait"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".SPO2TestActivity"
    android:screenOrientation="portrait"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".WBPTTestActivity"
    android:screenOrientation="portrait"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".WB03TestActivity"
    android:screenOrientation="portrait"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".WB0TestActivity"
    android:screenOrientation="portrait"
    tools:ignore="LockedOrientationActivity" />
<activity
    android:name=".ChoseActivity"
    android:screenOrientation="portrait"
    android:exported="true"
    tools:ignore="LockedOrientationActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

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

```
//Initialize the connection SDK
Global.oxygenProtocol = OxygenProtocol.getInstance(
    this, false, true, Global.sdkid_SP02);
Global.oxygenProtocol.setOnDataResponseListener(this);
Global.oxygenProtocol.setOnConnectStateListener(this);
Global.oxygenProtocol.setOnNotifyStateListener(this);
Global.oxygenProtocol.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 responded 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 Oxygen 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 Oxygen SPO2 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
	<pre>//Initialize the connection SDK Global.oxygenProtocol = OxygenProtocol.getInstance(     this, false, true, Global.sdkid_SPO2);</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 }</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 }</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 }</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 } </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 } </pre>

## 3.7. Read data from Oxygen :

## 3.7.1. Interface :

	public void dataReslut(byte[] data)
Definition	Read raw data bytes array from Oxygen.

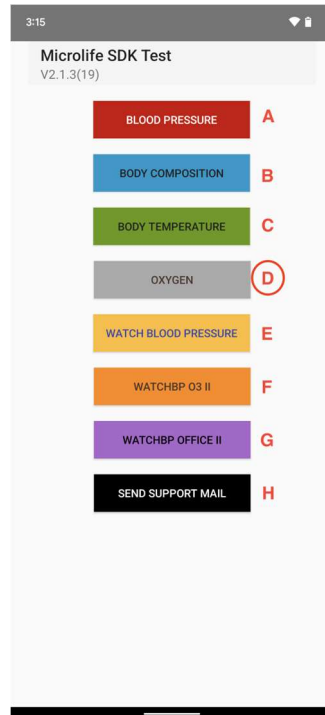
## 3.7.2. Delegate :

	void onResponseReadOxygenData(Oxygen oxygen)
Parameter	oxygen : Oxygen result data.

## Chapter4 User Interface of Demo App

### 4.1. Getting Started :

Start the app and then select the button “OXYGEN “ \ ” D” to communicate with the designate device Oxygen SPO2.

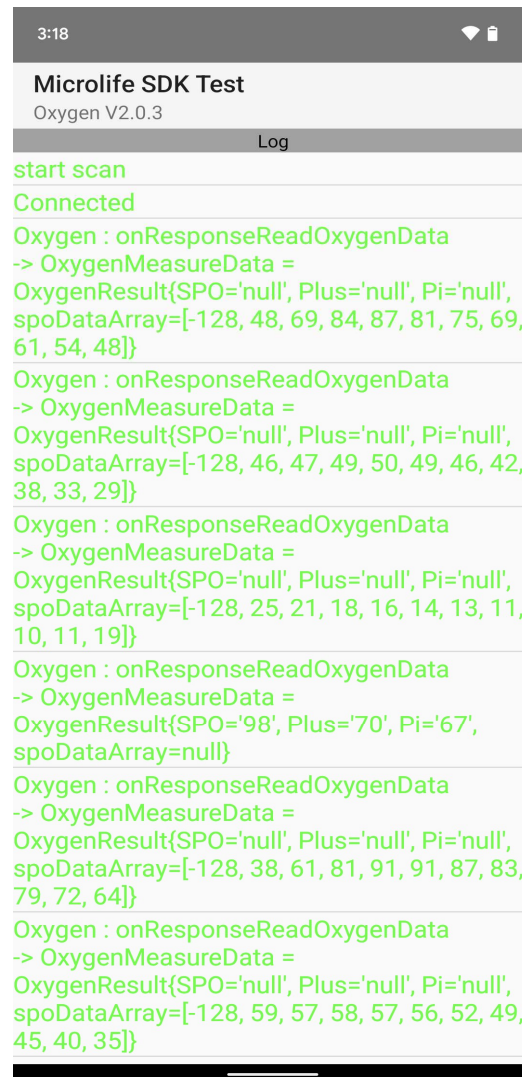


### 4.2. Operation Sequence :

- 4.2.1. The scanning (discovery) is automatically run to discover devices in the vicinity.
- 4.2.2. If a device is bonded, it will be connected accordingly. If not, the “bindingDevice” can be used to run bonding process.

### 4.3. Operating Interface and Sequence :






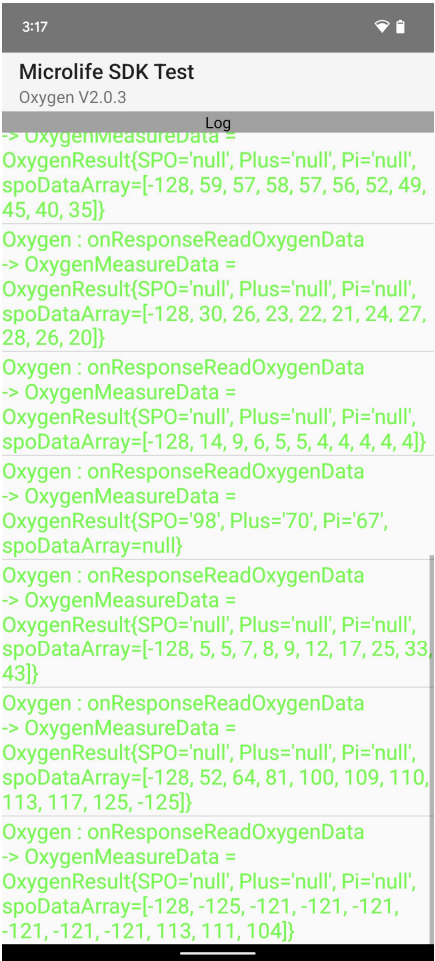
- 4.3.1. The log window is used to display result data information about communication handshake between App and communicate with the device Oxygen SPO2 by such as data transferring, synchronization and so on.
- 4.3.2. Refer to “SPO2TestActivity.java” from the demo application (sample code) to get more detailed.

## Chapter5    Functionality of Demo App

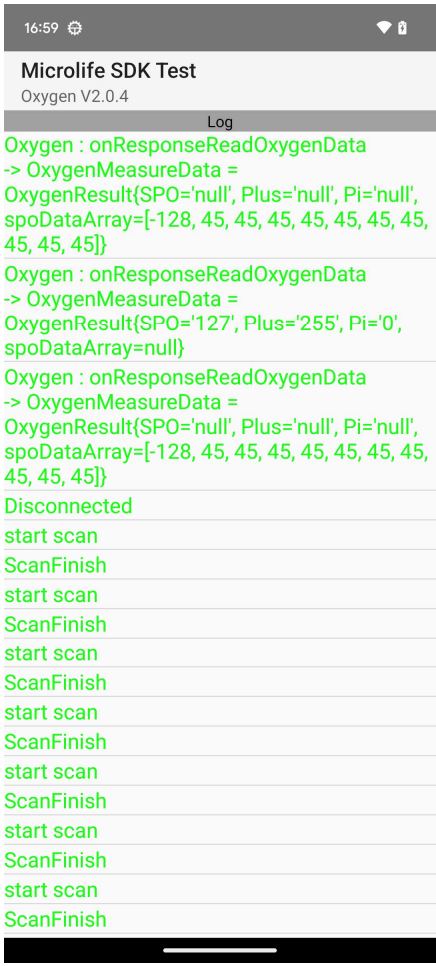
### 5.1.    Bluetooth authorization :

	<p>1. Request for Bluetooth permission.</p>
--	---

## 5.2. Command: Read data from SPO2

 <p>3:17</p> <p><b>Microlife SDK Test</b> Oxygen V2.0.3</p> <p>Log</p> <pre> -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 59, 57, 58, 57, 56, 52, 49, 45, 40, 35]} Oxygen : onResponseReadOxygenData -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 30, 26, 23, 22, 21, 24, 27, 28, 26, 20]} Oxygen : onResponseReadOxygenData -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 14, 9, 6, 5, 5, 4, 4, 4, 4]} Oxygen : onResponseReadOxygenData -&gt; OxygenMeasureData = OxygenResult{SPO='98', Plus='70', Pi='67', spoDataArray=null} Oxygen : onResponseReadOxygenData -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 5, 5, 7, 8, 9, 12, 17, 25, 33, 43]} Oxygen : onResponseReadOxygenData -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 52, 64, 81, 100, 109, 110, 113, 117, 125, -125]} Oxygen : onResponseReadOxygenData -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, -125, -121, -121, -121, -121, -121, -121, 113, 111, 104]} </pre>	<p>1. onResponseReadOxygenData :</p> <pre> -&gt; OxygenMeasureData = OxygenResult{SPO='98', Plus='70', Pi='67', spoDataArray=null}  -&gt; OxygenMeasureData = OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 52, 64, 81, 100, 109, 110, 113, 117, 125, -125]} </pre>
---	--

### 5.3. Command: Disconnect the Bluetooth

 <p>The screenshot shows the Microlife SDK Test app interface. At the top, the status bar displays the time 16:59 and battery level. The app title is 'Microlife SDK Test' and the version is 'Oxygen V2.0.4'. Below the title is a 'Log' section. The log contains several entries: 'Oxygen : onResponseReadOxygenData' followed by 'OxygenMeasureData =' and 'OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 45, 45, 45, 45, 45, 45, 45, 45, 45, 45, 45]}'. This is followed by another 'Oxygen : onResponseReadOxygenData' entry with 'OxygenMeasureData =' and 'OxygenResult{SPO='127', Plus='255', Pi='0', spoDataArray=null}'. Then, another 'Oxygen : onResponseReadOxygenData' entry with 'OxygenMeasureData =' and 'OxygenResult{SPO='null', Plus='null', Pi='null', spoDataArray=[-128, 45, 45, 45, 45, 45, 45, 45, 45, 45, 45, 45]}'. Finally, the log shows 'Disconnected' followed by a series of 'start scan' and 'ScanFinish' events.</p>	<ol style="list-style-type: none"><li>1. Disconnect the Bluetooth</li><li>2. If the oxygen device leaves the finger for a period of time, the bluetooth of the oxygen device will automatically disconnect.</li><li>3. The result is able to observe by the “onConnectionState(ConnectState state)” and its status can be found by the “ConnectState = Disconnect”.</li></ol>
--	---