

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

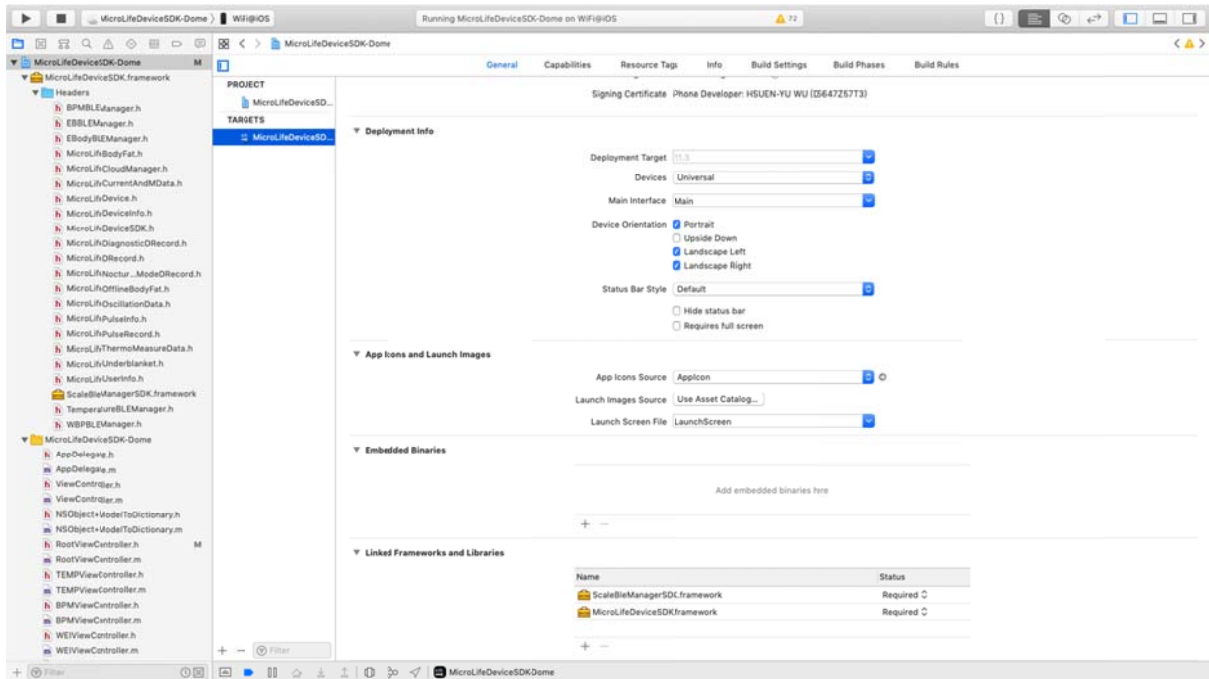
## **Table of Contents**

<b>Chapter 1</b>	<b>Development Environment</b>
<b>Chapter 2</b>	<b>Operation Sequence of BLEManager</b>
<b>Chapter 3</b>	<b>APIs of BLEManager</b>
<b>Chapter 4</b>	<b>Instruction of Demo App</b>
<b>Chapter 5</b>	<b>Functionality of Demo App</b>

## Chapter 1 Development Environment

This user manual serves as a quick introduction to MicroLifeDeviceSDK / APIs and shows how to integrate into an iOS Demo App.

- 1.1. First of all, add MicroLifeDeviceSDK.framework into a development project. The minimum supported version is iOS 10. The compatible version of Xcode IDE is 10.2.1.
- 1.2. Under TARGETS / General / Linked Frameworks and Libraries, add MicroLifeDeviceSDK.framework.



- 1.3. Under TARGETS / Build Settings / Enable, set Bitcode to NO.
- 1.4. Under TARGETS / Build Settings / Other Linker Flags, use the linker flag “-all\_load”.
- 1.5. Import header file as bellows.

```
#import <MicroLifeDeviceSDK/MicroLifeDeviceSDK.h>
```

## Chapter 2    **Operation Sequence of BLEManager**

The BLEManager is responsible for managing the Bluetooth communication.

- 2.1.    Initiate a BLEManager and register the delegation of BLEManager. The DataResponseDelegate is dedicated for response messages of BLEManager.
- 2.2.    Once the BLEManager is initiated, the BLEManager checks the status of Bluetooth by  
BLEManagerCellPhoneBluetoothDidUpdateState.
- 2.3.    While turning on Bluetooth, BLEManager runs the scan/discovery procedure automatically by  
BLEManagerDidDiscoverBluetoothDeviceMacAddress to get devices' information in the vicinity.
- 2.4.    Call the bindingDevice:(NSData \*)macAddress to bind an activated device. The status of binding can be found by the following section 2.5. The BLEManager will automatically connect to a bound device while it is activated.
- 2.5.    Connection status :
  - 2.5.1.    Connection / binding :
    - (void) \* BLEManagerDidConnectDevice;
  - 2.5.2.    Fail of connection :
    - (void) \* BLEManagerDidFailToConnectDevice;
  - 2.5.3.    Disconnection :
    - (void) \* BLEManagerDidDisconnectDevice;

## Chapter 3 APIs of BLEManager

By utilizing the APIs of BLEManager, can transfer data between device and App via Bluetooth. The APIs includes two parts: one is for fundamental Bluetooth and another is dedicated for communication with a designated device / BPM (Blood Pressure Monitor) with specific commands and protocol.

### 3.1. Device searching :

#### 3.1.1.

	<code>+(instancetype)shareInstanceWhithAuthorizationkey:(NSString *)key Target * Names:(NSArray *)target * Names</code>
Definition	Searching for device with default / customized name
Parameter	Key: Authorization code target * Names: device name which can be included single or multiple
	<pre>self.aBPMBLEManager = [BPMBLEManager     sharedInstanceWhithAuthorizationkey:SDKkey TargetBPMNames:@[@"A6 BT",@"A6     BASIS PLUS BT",@"A7 TCUCH BT",@"B3 BT",@"B6 Connect",@"A6",@"Progress"]]; self.aBPMBLEManager.dataResponseDelegate = self;</pre>

#### 3.1.2.

	<code>-(void) *</code> <code>BLEManagerCellPhoneBluetoothDidUpdateState:(MicroLifeBLEState)state;</code>
Definition	Response for the sharedInstanceWhithAuthorizationkey (). This is to manage Bluetooth status of cellphone.
Parameter	status: Bluetooth status of cellphone

	<pre> * @enum MicroLifeBLEState * * @discussion Represents the current state of a BLE. * * @constant MicroLifeBLEStateUnknown=CBManagerStateUnknown      State unknown, update imminent. * @constant MicroLifeBLEStateResetting=CBManagerStateResetting    The connection with the system service was momentarily lost, update imminent. * @constant MicroLifeBLEStateUnsupported=CBManagerStateUnsupported  The platform doesn't support the Bluetooth Low Energy Central/Client role. * @constant MicroLifeBLEStateUnauthorized=CBManagerStateUnauthorized The application is not authorized to use the Bluetooth Low Energy role. * @constant MicroLifeBLEStatePoweredOff=CBManagerStatePoweredOff   Bluetooth is currently powered off. * @constant MicroLifeBLEStatePoweredOn=CBManagerStatePoweredOn     Bluetooth is currently powered on and available to use. * */ typedef NS_ENUM(NSUInteger, MicroLifeBLEState) {     MicroLife3LEStateUnknown = 0,     MicroLife3LEStateResetting,     MicroLife3LEStateUnsupported,     MicroLife3LEStateUnauthorized,     MicroLife3LEStatePoweredOff,     MicroLife3LEStatePoweredOn, } </pre>
--	---

	<p>- (void) *</p> <p>BLEManagerDidDiscoverBluetoothDeviceMacAddress:(NSData *)macAddress Name:(NSString *)name RSSI:(NSNumber *)RSSI;</p>
Definition	Response for the sharedInstanceWhithAuthorizationkey (). This is to manage information of devices that are discovered in the vicinity.
Parameter	<p>macAddress: MAC address of device</p> <p>name: device name</p> <p>RSSI: RSSI</p>

### 3.2. Binding / Pairing :

#### 3.2.1.

	- (void)bindingDevice:(NSData *)macAddress
Definition	Binding a specified device by MAC
Parameter	macAddress: MAC address of device

#### 3.2.2.

	- (void) * BLEManagerDidConnectDevice;
Definition	Response for the bindingDevice() with the status Connection

	- (void) * BLEManagerDidFailToConnectDevice
Definition	Response for the bindingDevice() with the status Fail of Connection

	- (void) * BLEManagerDidDisconnectDevice;
Definition	Response for the bindingDevice() with the status Disconnection

### 3.3. MAC address of binding device :

#### 3.3.1.

	- (NSData *)getBindingDevice;
Definition	Get MAC address of a binding device or nil.

### 3.4. Unbinding :

#### 3.4.1.

	- (void)unBindingDevice;
Definition	Unbinding device

#### 3.4.2.

	- (void) * BLEManagerDidFailToConnectDevice;
Definition	Response for the unBindingDevice()

### 3.5. Disconnection :

#### 3.5.1.

	- (void)disconnectDevice;
Definition	Disconnect with device

#### 3.5.2.

	- (void) * BLEManagerDidFailToConnectDevice;
Definition	Response for the disconnectDevice ()

### 3.6. Searching for devices :

#### 3.6.1.

	- (void)reScan;
Definition	Search for devices repeatedly

#### 3.6.2.

	- (void) * BLEManagerDidDiscoverBluetoothDeviceMacAddress:(NSData *)macAddress Name:(NSString *)name RSSI:(NSNumber *)RSSI;
Definition	This is to get information of devices which are discovered in the vicinity.
Parameter	macAddress: MAC of device name: device name RSSI: RSSI

### 3.7. Stop searching for devices :

#### 3.7.1.

	- (void)stopScan;
Definition	Stop searching

#### 3.7.2. Null.

### 3.8. Read usual mode history data from BPM :

#### 3.8.1.

	- (void)readUsualModeHistoryData;
Definition	Read the history data of usual mode from BPM

#### 3.8.2.

	- (void)WBPBLEManagerResponseReadUsualModeHistoryData:(MicroLifeDRecord *)data
--	---



	IsNoData:(BOOL)isNoData;
Definition	Response for the readUsualModeHistoryData()
Parameter	data : history data of usual mode isNoData : the status True or False

### 3.9. Read diagnostic mode history data from BPM :

#### 3.9.1.

	- (void)readDiagnosticModeHistoryData;
Definition	Read diagnostic mode history data from BPM

#### 3.9.2.

	- (void)WBPBLEManagerResponseReadDiagnosticModeHistoryData:(MicroLifeDiagnosticDRecord *)data IsNoData:(BOOL)isNoData;
Definition	Response for the readDiagnosticModeHistoryData()
Parameter	data : history data of diagnostic mode isNoData : the status True or False

### 3.10. Clear selected mode history data of the BPM :

#### 3.10.1.

	- (void)clearHistoryDataWithSelectedUsualMode:(BOOL)clearUsualMode DiagnosticMode:(BOOL)clearDiagnosticMode NocturnalMode:(BOOL)clearNocturnalMode;
Definition	Clear history data by selected mode
Parameter	clearUsualMode : clear data for Usual Mode clearDiagnosticMode : clear data for Diagnostic Mode clearNocturnalMode : clear data for Nocturnal Mode

#### 3.10.2.

	- (void)WBPBLEManagerResponseClearSelectedModeHistoryData:(BOOL)isSuccess;
Definition	Response for the clearHistoryDataWithSelectedUsualMode()
Parameter	isSuccess : the status True or False

### 3.11. Clear current mode history data of the BPM :

#### 3.11.1.

	- (void)clearCurrentModeHistoryData;
Definition	Clear data for current mode

#### 3.11.2.

	- (void)WBPBLEManagerResponseClearCurrentModeHistoryData:(BOOL)isSuccess;
Definition	Response for the clearCurrentModeHistoryData()
Parameter	isSuccess : the status True or False

### 3.12. Disconnect the Bluetooth with BPM :

#### 3.12.1.

	- (void)disconnect;
Definition	Disconnect the Bluetooth with BPM

#### 3.12.2.

	- (void)WBPBLEManagerDidDisconnectDevice;
Definition	Response for the disconnect ()

### 3.13. Write device Time to BPM :

#### 3.13.1.

	- (void)writeDeviceTime;
--	--------------------------

Definition	Synchronization between device and cellphone
------------	--

## 3.13.2.

	- (void)WBPBLEManagerResponseWriteDeviceTime:(BOOL)isSuccess;
Definition	Response for the writeDeviceTime()
Parameter	isSuccess : the status True or False

## 3.14. Write a new user ID to BPM :

## 3.14.1.

	- (void)writeUserID:(NSString *)ID
Definition	Write a new user ID to BPM
Parameter	ID : User ID

## 3.14.2.

	- (void)WBPBLEManagerResponseWriteUserID:(BOOL)isSuccess;
Definition	Response for the writeUserID ()
Parameter	isSuccess : the status True or False

## 3.15. Read nocturnal mode setting :

## 3.15.1.

	- (void)readNocturnalModeSetting;
Definition	Read the setting of Nocturnal mode

## 3.15.2.

	- (void)WBPBLEManagerResponseReadNocturnalModeSetting:(MicroLifeDeviceInfo *)deviceInfo;
--	---

Definition	Response for the readNocturnalModeSetting()
Parameter	deviceInfo : device information (setting) of nocturnal mode

### 3.16. Change nocturnal mode setting :

#### 3.16.1.

	- (void)changeNocturnalModeSettingOn:(BOOL)open StartYear:(NSInteger)year StartMonth:(NSInteger)month StartDay:(NSInteger)day StartHour:(NSInteger)hour;
Definition	Change the setting of nocturnal mode Note: This command has to match with specific device. It can be used the following: "readDeviceIDAndInfo(MicroLifeDeviceInfo.openNocturnalMode)" to check if the device supports it or not.
Parameter	open : ON/OFF for Nocturnal mode  year : year  month : month  day : day  hour : hour

#### 3.16.2.

	- (void)WBPBLEManagerResponseChangeNocturnalModeSetting:(BOOL)isSuccess;
Definition	Response for the changeNocturnalModeSettingOn()
Parameter	isSuccess : the status True or False

### 3.17. Erase all of device measure & error times :

#### 3.17.1.

	- (void)eraseAllOfDeviceMeasureAndErrorTimes;
Definition	Erase all of device measure & error times

## 3.17.2.

	- (void)WBPBLEManagerResponseEraseAllOfDeviceMeasureAndErrorTimes:(BOOL)isSuccess;;
Definition	Response for the eraseAllOfDeviceMeasureAndErrorTimes ()
Parameter	isSuccess : the status True or False

## 3.18. Read device ID and info from BPM :

## 3.18.1.

	- (void)readDeviceIDAndInfo;
Definition	Read device ID and info from BPM

## 3.18.2.

	- (void)WBPBLEManagerResponseReadDeviceIDAndInfo: (MicroLifeDeviceInfo *)deviceInfo;
Definition	Response for the readDeviceIDAndInfo ()
Parameter	deviceInfo : device ID and info

## 3.19. Read device Time from BPM :

## 3.19.1.

	- (void)readDeviceTime;
Definition	Read device Time from BPM

## 3.19.2.

	- (void)WBPBLEManagerResponseReadDeviceTime:(MicroLifeDeviceInfo *)deviceInfo;
Definition	Response for the readDeviceTime ()
Parameter	deviceInfo : Time of device

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

## 3.20.1.

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

## 3.20.2.

	- (void)WBPBLEManagerResponseReadUserAndVersionData:(MicroLifeUserInfo *)user VersionData:(MicroLifeDeviceInfo *)verData;
Definition	Response for the readUserAndVersionData ()
Parameter	user : user ID  verData : version data

## 3.21. Read Nocturnal mode history data from BPM :

## 3.21.1.

	- (void)readNocturnalModeHistoryData;
Definition	Read the history data of Nocturnal mode from BPM

## 3.21.2.

	- (void)WBPBLEManagerResponseReadNocturnalPatternHistoryData:(MicroLifeNocturnalModeDRecord *)data IsNoData:(BOOL)isNoData;
Definition	Response for the readNocturnalModeHistoryData ()
Parameter	data : Nocturnal pattern history data  isNoData : the status True or False

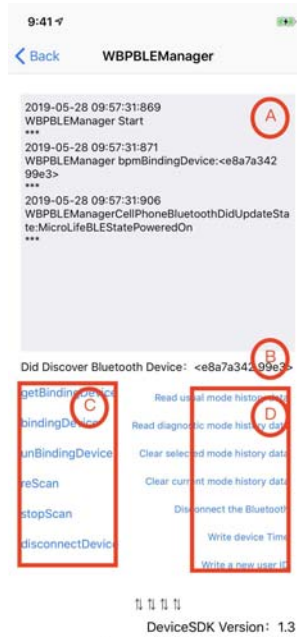
## Chapter 4 Instruction of Demo App

### 4.1. Getting Started :

Start the app and then select the button “WatchBP HOME SDK” / “D” to communicate with the designate device.



## 4.2. Operating Interface and Sequence :



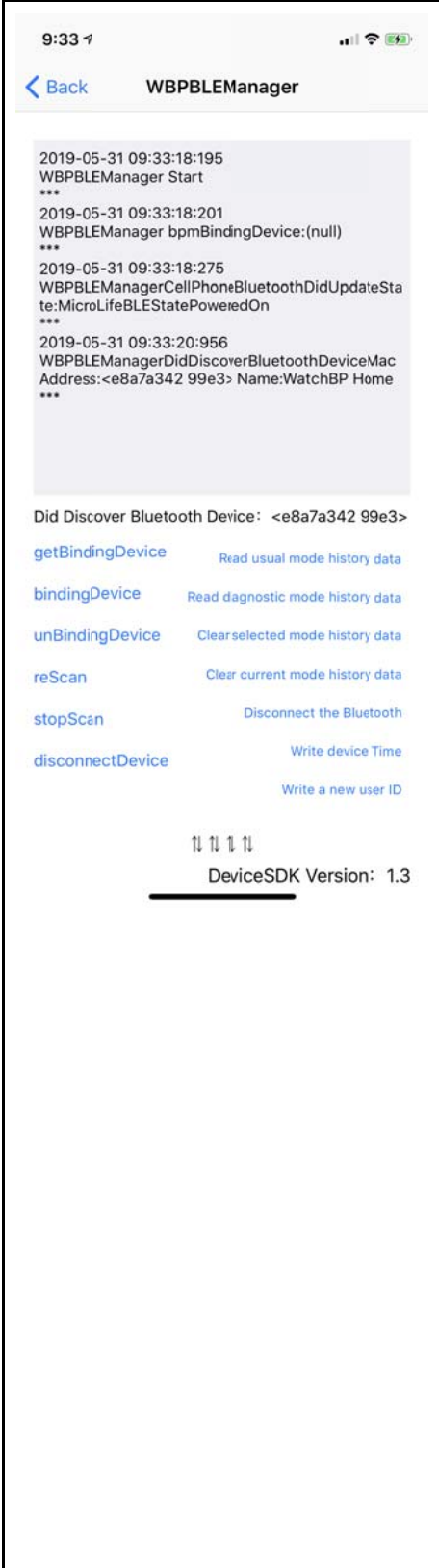
- 4.2.1. Region A : The log window is used to display information about communication handshake between App and device.
- 4.2.2. Region B : The display is for current device with MAC address.
- 4.2.3. Region C : This part is included fundamental Bluetooth tasks such as Connection, Disconnection and Bonding/ Pairing.
- 4.2.4. Region D : This part is to communicate with the device WatchBP HOME by different functions / commands such as data transferring, synchronization and so on.
- 4.2.5. Refer to WBPViewController from the demo code (sample code) to get more detailed.
- 4.2.6. Operation Sequence :
  - 4.2.6.1. The scanning (discovery) is automatically run to discover devices in the vicinity.
  - 4.2.6.2. If a device is bonded, it will be connected accordingly. If not, the “bindingDevice” can be used to run bonding process.



- 4.2.6.3. Once the device is connected, select each function / command from Region D to manage Read and Write.

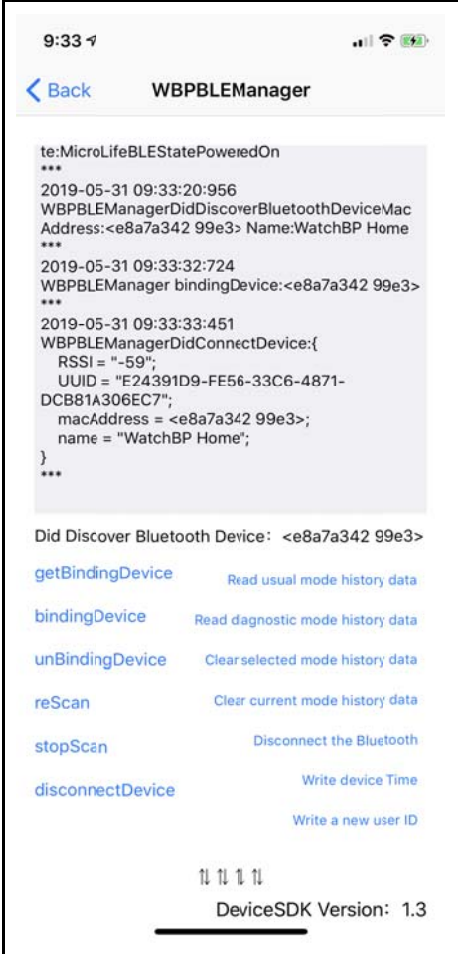
## Chapter 5    Functionality of Demo App

### 5.1.    Scanning / Discovery :

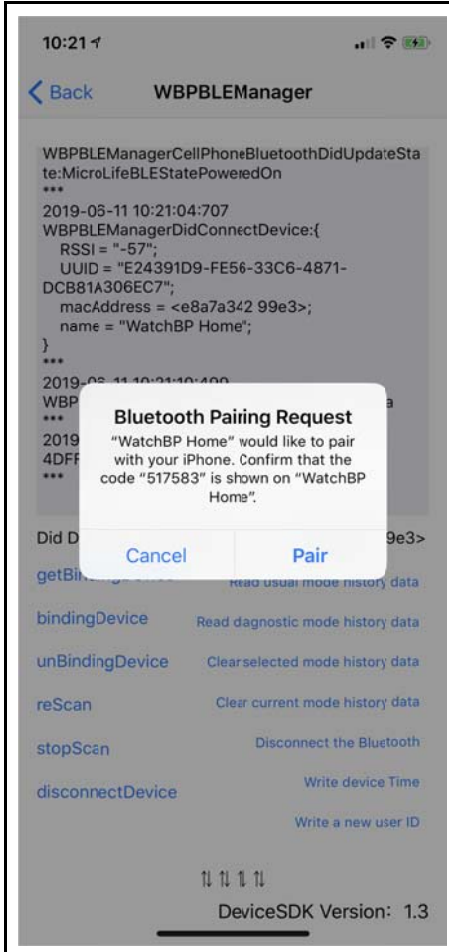
 <p>The screenshot shows the WBPBLEManager app interface. At the top, the status bar displays the time 9:33 and signal/battery icons. Below the title bar, there is a log area with the following text:</p> <pre> 2019-05-31 09:33:18:195 WBPBLEManager Start *** 2019-05-31 09:33:18:201 WBPBLEManager bpmBindingDevice:(null) *** 2019-05-31 09:33:18:275 WBPBLEManagerCellPhoneBluetoothDidUpdateState:MicroLifeBLEStatePoweredOn *** 2019-05-31 09:33:20:956 WBPBLEManagerDidDiscoverBluetoothDeviceMacAddress:&lt;e8a7a342 99e3&gt; Name:WatchBP Home *** </pre> <p>Below the log, there is a section titled "Did Discover Bluetooth Device: &lt;e8a7a342 99e3&gt;". Under this section, there are several actions listed with corresponding labels:</p> <ul style="list-style-type: none"> <li>getBindingDevice (Read usual mode history data)</li> <li>bindingDevice (Read diagnostic mode history data)</li> <li>unBindingDevice (Clear selected mode history data)</li> <li>reScan (Clear current mode history data)</li> <li>stopScan (Disconnect the Bluetooth)</li> <li>disconnectDevice (Write device Time)</li> <li>Write a new user ID</li> </ul> <p>At the bottom, there is a section titled "DeviceSDK Version: 1.3" with a horizontal line underneath.</p>	<ol style="list-style-type: none"> <li>1. First of all, the WBPBLEManager starts Scanning / Discovery procedure. The detailed information will be displayed in Region A.</li> <li>2. The WBPBLEManager bpmBindingDevice: * : (null) indicates that the remote device wasn't bonded yet.</li> <li>3. The WBPBLEManagerCellPhoneBluetoothDidUpdateState:MicroLifeBLEStatePoweredOn indicates that the power status of Bluetooth from cell phone is On.</li> <li>4. The WBPBLEManagerDidDiscoverBluetoothDeviceMacAddress:&lt;e8a7a342 99e3&gt; Name:WatchBP Home indicates that the device with MAC address &lt;e8a7a342 99e3&gt; is discovered nearby.</li> <li>5. In Region B, the "Did Discover Bluetooth Device : &lt;e8a7a342 99e3&gt;" stands for that the discovered/ selected device can be used in next steps such as Connection or Bonding.</li> <li>6. In Region C, click the</li> </ol>
--	---

	function “bindingDevice” to run bonding process.
--	--

## 5.2. Connection :

 <p>The screenshot shows the WBPBLEManager app interface. At the top, there's a status bar with the time 9:33 and signal indicators. Below it is a navigation bar with a 'Back' button and the title 'WBPBLEManager'. The main content area displays a log of events:         <ul style="list-style-type: none"> <li>te:MicroLifeBLEStatePoweredOn</li> <li>***</li> <li>2019-05-31 09:33:20:956</li> <li>WBPBLEManagerDidDiscoverBluetoothDevice/Mac Address:&lt;e8a7a342 99e3&gt; Name:WatchBP Home</li> <li>***</li> <li>2019-05-31 09:33:32:724</li> <li>WBPBLEManager bindingDevice:&lt;e8a7a342 99e3&gt;</li> <li>***</li> <li>2019-05-31 09:33:33:451</li> <li>WBPBLEManagerDidConnectDevice:{</li> <li>    RSSI = "-59";</li> <li>    UUID = "E24391D9-FE56-33C6-4871-DCB81A306EC7";</li> <li>    macAddress = &lt;e8a7a342 99e3&gt;;</li> <li>    name = "WatchBP Home";</li> <li>}</li> <li>***</li> </ul>         Below the log, there are several control buttons:         <ul style="list-style-type: none"> <li>getBindingDevice (Read usual mode history data)</li> <li>bindingDevice (Read diagnostic mode history data)</li> <li>unBindingDevice (Clear selected mode history data)</li> <li>reScan (Clear current mode history data)</li> <li>stopScan (Disconnect the Bluetooth)</li> <li>disconnectDevice (Write device Time)</li> <li>(Write a new user ID)</li> </ul>         At the bottom, there are three vertical sliders and the text 'DeviceSDK Version: 1.3'.       </p>	<p>1 The information about the device bonding will be shown “WBPBLEManager bindingDevice:&lt;e8a7a342 99e3&gt;” in Region A.</p> <p>2.The following indicates that the device is connected. WBPBLEManagerDidConnect Device:{</p> <pre>         RSSI = "-68";         UUID = "E24391D9-FE56-33C6-4871-DCB81A306EC7";         macAddress = &lt;e8a7a342 99e3&gt;;         name = "WatchBP Home";       </pre>
---	---

### 5.3. Bonding / Pairing :

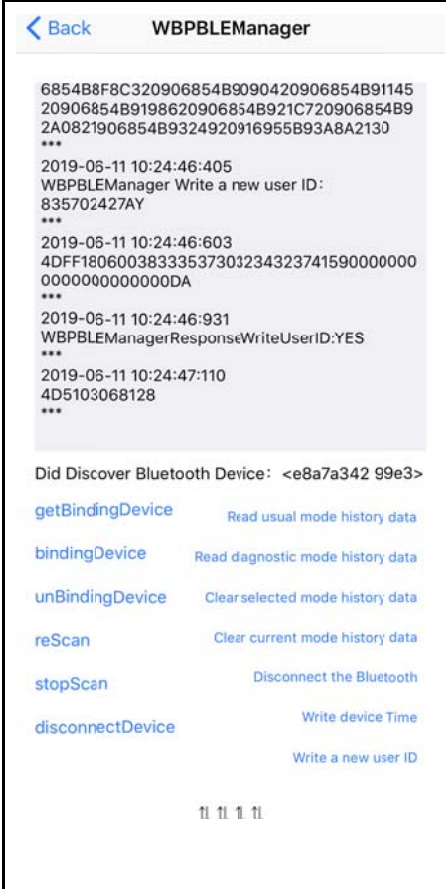


The screenshot shows the WBPBLEManager app interface. At the top, there is a status bar with the time 10:21 and signal indicators. Below the status bar is a navigation bar with a back arrow and the title "WBPBLEManager". The main content area displays a log of events, including a timestamp "2019-05-11 10:21:04:707" and a dictionary of device information: `WBPLEManagerDidConnectDevice:{ RSSI = "-57"; UUID = "E24391D9-FE56-33C6-4871-DCB81A306EC7"; macAddress = <e8a7a342 99e3>; name = "WatchBP Home"; }`. A modal dialog titled "Bluetooth Pairing Request" is overlaid on the screen, containing the text: "WatchBP Home" would like to pair with your iPhone. Confirm that the code "517583" is shown on "WatchBP Home". The dialog has two buttons: "Cancel" and "Pair". Below the dialog, the app interface shows a list of actions: "Did D", "getBin", "bindingDevice", "unBindingDevice", "reScan", "stopScan", and "disconnectDevice". At the bottom, there is a footer with the text "DeviceSDK Version: 1.3".

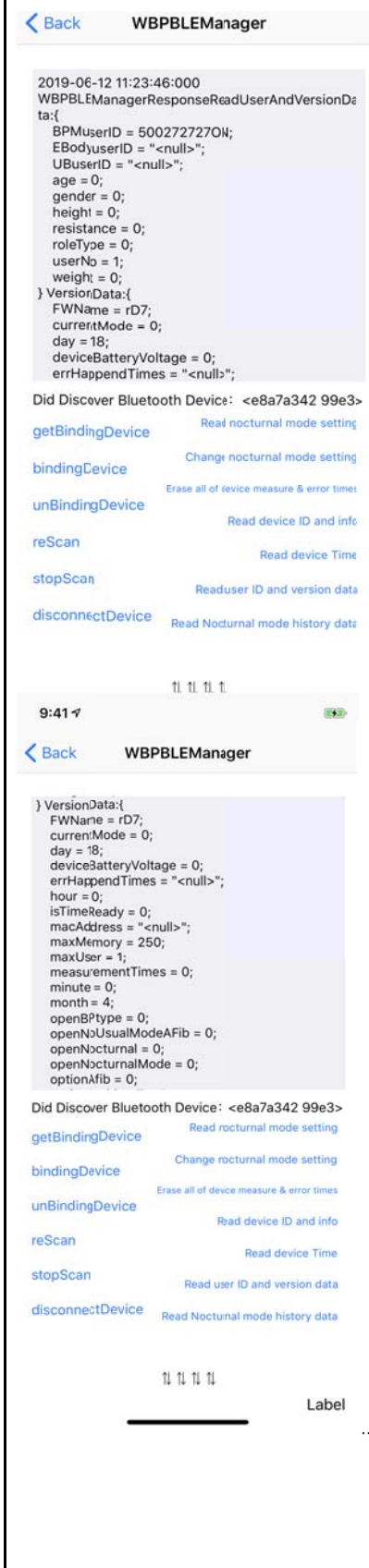
1. There is a message to confirm the bonding between device and cellphone if they haven't bonded yet.

2. Click "Pair" to run the bonding process.

#### 5.4. Testing Command : Write a new user ID to BPM

	<ol style="list-style-type: none"> <li>1. The command “Write a new user ID” is to write a new user ID to device.</li> <li>2. The following “WBPBLEManager Write a new user ID : 835702427AY” indicates that the writing value is “835702427AY” made up of ASCII code.</li> <li>3. The response is as below: WBPBLEManagerResponse WriteUserID:YES. It means that the process is successful.</li> <li>4. It can be checked by the command “Read user ID and version data”.</li> </ol>
--	--

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



The screenshot shows the WBPBLEManager app interface. At the top, there's a 'Back' button and the title 'WBPBLEManager'. Below it, a log entry shows the timestamp '2019-06-12 11:23:46:000' and the command 'WBPBLEManagerResponseReadUserAndVersionData:'. The response is a JSON object containing user data and version data. The user data includes BPMUserID, EBodyuserID, UUserID, age, gender, height, resistance, roleType, userNo, and weight. The version data includes FWName, currentMode, day, deviceBatteryVoltage, and errHappendTimes. Below the log entry, there's a list of commands and their descriptions: getBindingDevice (Read nocturnal mode setting), bindingDevice (Change nocturnal mode setting), unBindingDevice (Erase all of device measure & error times), reScan (Read device ID and info), stopScan (Read device Time), and disconnectDevice (Read Nocturnal mode history data). At the bottom, there's a status bar showing the time '9:41' and a battery icon.

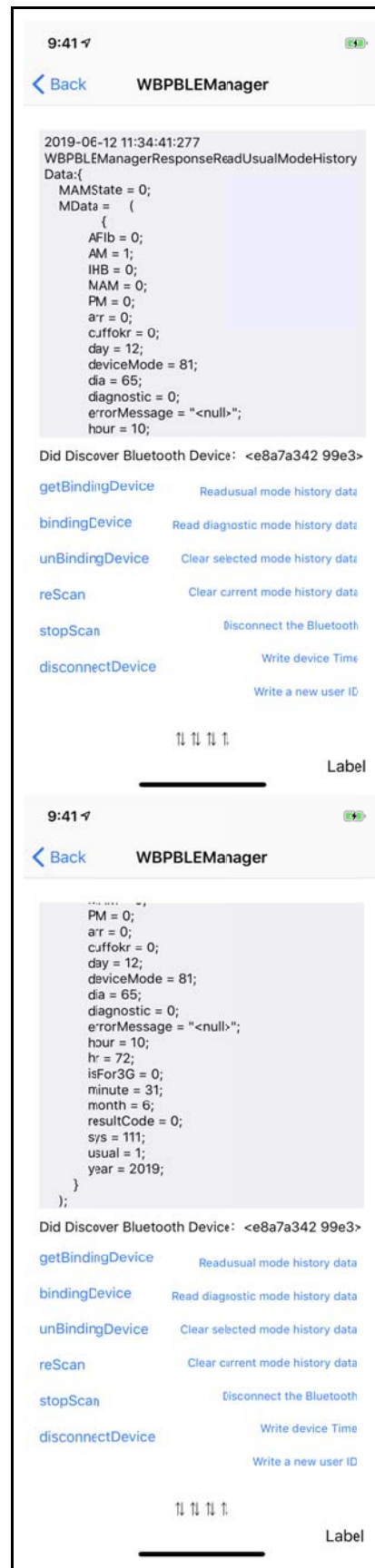
1. The command “Read user ID and version data” is to receive the information of user ID and device version.

2. The response is as below:  
WBPBLEManagerResponseReadUserAndVersionData:{  
  BPMUserID =  
500272727ON;  
  EBodyuserID = "<null>";  
  UUserID = "<null>";  
  age = 0;  
  gender = 0;  
  height = 0;  
  resistance = 0;  
  roleType = 0;  
  userNo = 1;  
  weight = 0;  
} VersionData:{  
  FWName = rD7;  
  currentMode = 0;  
  day = 18;  
  deviceBatteryVoltage = 0;  
  errHappendTimes =  
"<null>";  
  hour = 0;  
  isTimeReady = 0;  
  macAddress = "<null>";  
  maxMemory = 250;  
  maxUser = 1;  
  measurementTimes = 0;  
  minute = 0;  
  month = 4;  
  openBPtype = 0;  
  openNoUsualModeAFib = 0;  
  openNocturnal = 0;  
  openNocturnalMode = 0;  
  optionAfib = 0;  
  openNocturnal = 0;  
  openNocturnalMode = 0;  
  optionAfib = 0;  
  optionAmbientT = 1;  
  optionDeviceID = 0;  
  optionDiagnosticModeAFib  
= 0;  
  optionIHB = 1;  
  optionMAM = 1;

	<pre>optionTubeless = 0; protocolVersion = 0; second = 0; year = 2019; }</pre> <p>3. The “BPMuserID” stands for user ID and the value is “500272727ON”.</p> <p>4. The “VersionData” stands for device version included name of firmware “FWName” and its building date “day”, “month” and “year”.</p>
--	---



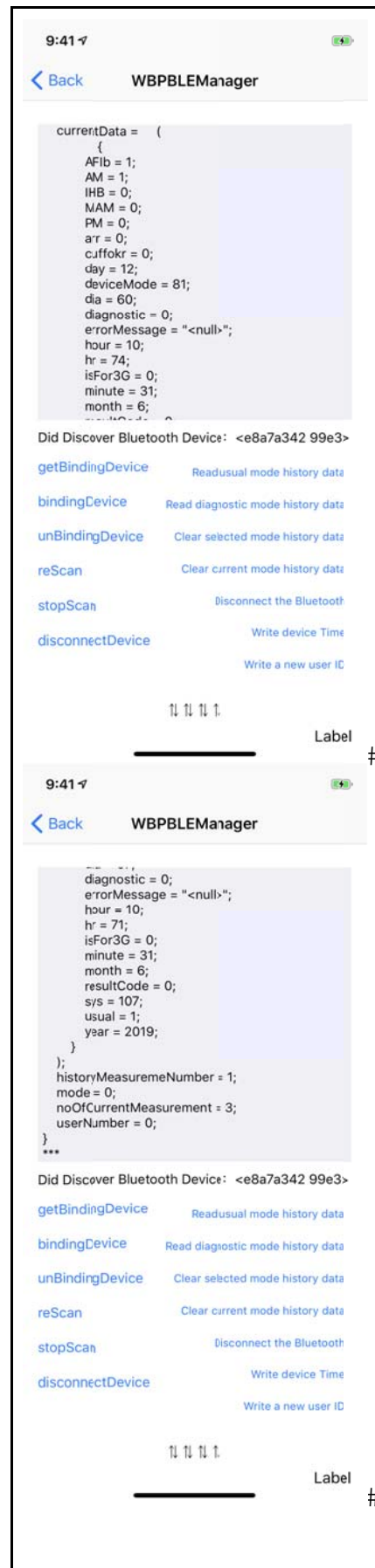
## 5.6. Testing Command : Read usual mode history data from BPM



1. The command "Read usual mode history data" is to read the data of usual mode.

2. The response is as below:  
WBPBLEManagerResponseReadUsualModeHistoryData:

```
{
  MAMState = 0;
  MData = (
    {
      AFib = 0;
      AM = 1;
      IHB = 0;
      MAM = 0;
      PM = 0;
      arr = 0;
      cuffokr = 0;
      day = 12;
      deviceMode = 81;
      dia = 65;
      diagnostic = 0;
      errorMessage =
"<null>";
      hour = 10;
      hr = 72;
      isFor3G = 0;
      minute = 31;
      month = 6;
      resultCode = 0;
      sys = 111;
      usual = 1;
      year = 2019;
    }
  );
  currentData = (
    {
      AFib = 1;
      AM = 1;
      IHB = 0;
      MAM = 0;
      PM = 0;
      arr = 0;
      cuffokr = 0;
      day = 12;
      deviceMode = 81;
      dia = 60;
```



```

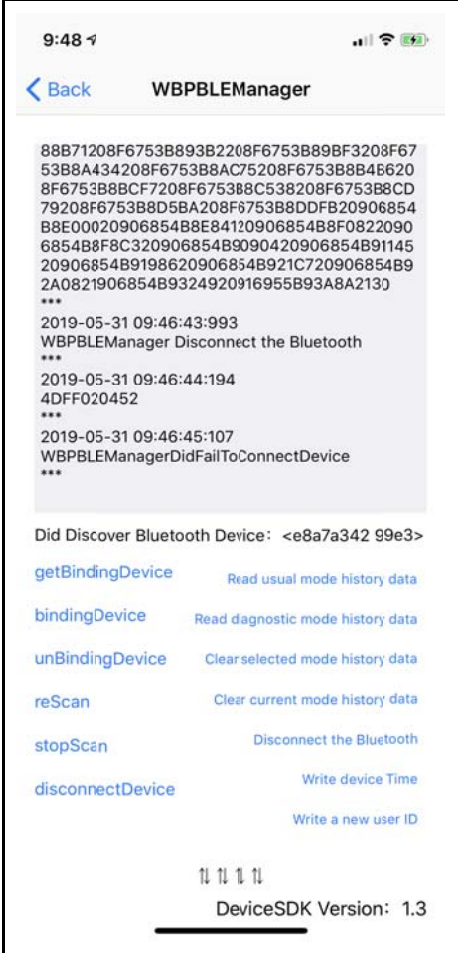
        diagnostic = 0;
        errorMessage =
"<null>";
        hour = 10;
        hr = 74;
        isFor3G = 0;
        minute = 31;
        month = 6;
        resultCode = 0;
        sys = 115;
        usual = 1;
        year = 2019;
    },
    {...},
    {...}
);
historyMeasuremeNumber
= 1;
mode = 0;
noOfCurrentMeasurement
= 3;
userNumber = 0;
}

```

3. Both “MData” & “currentData” are made up of blood pressure reading systolic (sys), diastolic (dia), pulse (hr), and date & time (year, month, day, hour, minute) respectively. The “MData” stands for the “history” data. The “currentData” stands for the “current” measurement data.

4. For instance, the above-mentioned MData stands for “2019/6/12 10:31 sys=111 dia=62 hr=72...”.

## 5.7. Disconnection

 <p>The screenshot shows the WBPBLEManager app interface. At the top, there's a status bar with the time 9:48 and signal indicators. Below the title bar, there's a log area with the following text:</p> <pre> 88B71208F6753B893B2208F6753B89BF3208F67 53B8A434208F6753B8AC75208F6753B8B46620 8F6753B8BCF7208F6753B8C538208F6753B8CD 79208F6753B8D5BA208F6753B8DDFB20906854 B8E00020906854B8E84120906854B8F0822090 6854B8F8C320906854B9090420906854B91145 20906854B9198620906854B921C720906854B9 2A0821906854B9324920916955B93A8A213D *** 2019-05-31 09:46:43:993 WBPBLEManager Disconnect the Bluetooth *** 2019-05-31 09:46:44:194 4DFF020452 *** 2019-05-31 09:46:45:107 WBPBLEManagerDidFailToConnectDevice *** </pre> <p>Below the log, there's a message: "Did Discover Bluetooth Device: &lt;e8a7a342 99e3&gt;". At the bottom, there's a list of actions with their corresponding descriptions:</p> <ul style="list-style-type: none"> <li>getBindingDevice: Read usual mode history data</li> <li>bindingDevice: Read diagnostic mode history data</li> <li>unBindingDevice: Clear selected mode history data</li> <li>reScan: Clear current mode history data</li> <li>stopScan: Disconnect the Bluetooth</li> <li>disconnectDevice: Write device Time</li> <li>Write a new user ID</li> </ul> <p>At the very bottom, there's a footer: "DeviceSDK Version: 1.3".</p>	<p>1. In order to disconnect the device completely, either “disconnectDevice” (Region C) or “Disconnect the Bluetooth” ( Region D ) is available.</p> <p>2. The response is the following “WBPBLEManagerDidFailToConnectDevice” once the disconnection command is executed.</p>
--	---