






## Table of contents

第1章	Environment configuration instructions
第2章	SDK Tools
第3章	SDK Bluetooth Interface Description
第4章	MicroLifeDevice Interface Description
第5章	Temperature Interface Description
第6章	Blood Pressure Interface Description
第7章	Weight Interface Description
第8章	Oxygen Interface Description
第9章	WatchBP O3 Interface Description
第10章	WatchBP Home Interface Description
第11章	WatchBP Office Interface Description
第12章	Peak flowInterface Description
第13章	V1.x Migration V2.x Instructions
第14章	Dome Code Description
第15章	Operating Instructions
第16章	Update History

第1章 Environment configuration instructions:

- 1.1. Minimum supported SDK version: **iOS 16**
- 1.2. Drag MicroLifeDeviceSDK.framework into the project.
- 1.3. 在TARGETS / General / Frameworks,Libraries,and Embedded Content 中加入MicroLifeDeviceSDK.framework。
- 1.4. **To use the body fat meter, you need to add BelterScaleInfo.framework in the MicroLifeDeviceSDK to Frameworks, Libraries, and Embedded Content and set it to "Do Not Embed".**

▼ Frameworks, Libraries, and Embedded Content	
Name	Embed
 BelterScaleInfo.framework	Do Not Embed ↕
 ECGProcess.framework	Do Not Embed ↕
 MessageUI.framework	Do Not Embed ↕
 MicroLifeDeviceSDK.framework	Do Not Embed ↕
 ZipArchive.framework	Embed & Sign ↕
+ -	

- 1.5. In TARGETS / Build Settings / Other Linker Flags add -all\_load.
- 1.6. Where you need to use it, just import the header file.

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

## 第2章 SDK Tool Description:

### 2.1. Log Management: Local Log Recording Tool

```
//Log Management
#import <MicroLifeDeviceSDK/LogManager.h>
```

### 2.2. Bluetooth Core:

- 2.2.1. BLESDK: Bluetooth low-level manager, responsible for all Bluetooth general operations.
- 2.2.2. BLEDeviceInfo: Bluetooth device tool, Bluetooth scan response basic object.
- 2.2.3. MicroLifeDevice: A universal device tool for Bailue. Responsible for transmitting and parsing commands from various Bailue devices.
- 2.2.4. MicroLifeDataModel: A general data object.
- 2.2.5. MicroLifeDeviceInfo: MicroLife device object.
- 2.2.6. MicroLifeUserInfo: MicroLife user object.

```
//Bluetooth Core
#import <MicroLifeDeviceSDK/BLESDK.h>
#import <MicroLifeDeviceSDK/BLEDeviceInfo.h>
#import <MicroLifeDeviceSDK/MicroLifeDevice.h>
#import <MicroLifeDeviceSDK/MicroLifeDataModel.h>

//Data Model
#import <MicroLifeDeviceSDK/MicroLifeDeviceInfo.h>
#import <MicroLifeDeviceSDK/MicroLifeUserInfo.h>
```

### 2.3. Temperature:

- 2.3.1. MicroLifeTemperature: MicroLifeTemperature device tool.
- 2.3.2. MicroLifeTemperatureMeasureData: Measure Data

```
//Temperature
#import <MicroLifeDeviceSDK/MicroLifeTemperature.h>
#import <MicroLifeDeviceSDK/MicroLifeTemperatureMeasureData.h>
```

## 2.4. Blood Pressure:

- 2.4.1. MicroLife3GBP: 3G BP device tool.
- 2.4.2. MicroLife4GBP: 4G BP device tool.
- 2.4.3. MicroLifeBloodPressureDRecord: DRecord。
- 2.4.4. MicroLifeBloodPressureCurrentAndMData: Current、MData。

```
//Blood Pressure
#import <MicroLifeDeviceSDK/MicroLife3GBP.h>
#import <MicroLifeDeviceSDK/MicroLife4GBP.h>
#import <MicroLifeDeviceSDK/MicroLifeBloodPressureDRecord.h>
#import <MicroLifeDeviceSDK/MicroLifeBloodPressureCurrentAndMData.h>
```

## 2.5. Weight:

- 2.5.1. MicroLifeWeight: Weight device tool.
- 2.5.2. MicroLifeBodyFat: Weight data。
- 2.5.3. BelterScaleInfo.framework: Body composition analysis tool.

```
//Weight
#import <MicroLifeDeviceSDK/MicroLifeWeight.h>
#import <MicroLifeDeviceSDK/MicroLifeBodyFat.h>
```

## 2.6. Oxygen:

- 2.6.1. MicroLifeOxygen: Oxygen device tool.
- 2.6.2. MicroLifeOxygenData: Oxygen data。

```
//Oxygen
#import <MicroLifeDeviceSDK/MicroLifeOxygen.h>
#import <MicroLifeDeviceSDK/MicroLifeOxygenData.h>
```



## 2.7. WatchBP:

- 2.7.1. MicroLifeWatchBPHome: WatchBP Home device tool.
- 2.7.2. MicroLifeWatchBP03: WatchBP O3 device tool.
- 2.7.3. MicroLifeWatchBPOffice: WatchBP Office device tool.
- 2.7.4. MicroLifeWatchBPDRecord: DRecord。
- 2.7.5. MicroLifeWatchBPCurrentAndMData: Current、MData。
- 2.7.6. MicroLifeWatchBPcBPdataAndCalCBP: CBP data、CalCBP。
- 2.7.7. MicroLifeWatchBPSettingValues: Setting Values。
- 2.7.8. MicroLifeWatchBPFunctionSettingValues: Function Setting Values。
- 2.7.9. MicroLifeWatchBPDiagnosticDRecord: Diagnostic DRecord。
- 2.7.10. MicroLifeWatchBPNocturnalModeDRecord: Nocturnal Mode DRecord。
- 2.7.11. MicroLifeWatchBPRemoteMeasurement: Remote Measurement。
- 2.7.12. MicroLifeWatchBPM1: WatchBP M1 device tool.

```
//WatchBP
#import <MicroLifeDeviceSDK/MicroLifeWatchBPHome.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBP03.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPOffice.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPDRecord.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPCurrentAndMData.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPcBPdataAndCalCBP.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPSettingValues.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPFunctionSettingValues.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPDiagnosticDRecord.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPNocturnalModeDRecord.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPRemoteMeasurement.h>
#import <MicroLifeDeviceSDK/MicroLifeWatchBPM1.h>
```

## 2.8. Peak flow:

- 2.8.1. MicroLifePFM: Respiratory flow measurement function
- 2.8.2. MicroLifePFMDRecord: respiratory flow data recording
- 2.8.3. MicroLifePFMCurrentAndMData: current and average respiratory data
- 2.8.4. MicroLifePFMWaveformRecord: respiratory waveform data recording

```
// 呼吸流量測量 (Peakflow Measurement)
#import <MicroLifeDeviceSDK/MicroLifePFM.h> // 呼吸流量測量功能 (Peakflow Measurement Functions)
#import <MicroLifeDeviceSDK/MicroLifePFMDRecord.h> // 呼吸流量數據記錄 (Peakflow Data Record)
#import <MicroLifeDeviceSDK/MicroLifePFMCurrentAndMData.h> // 當前與平均呼吸數據 (Current and Mean Peakflow Data)
#import <MicroLifeDeviceSDK/MicroLifePFMWaveformRecord.h> // 呼吸波形數據記錄 (Peakflow Waveform Record)
```

## 第3章 SDK Bluetooth interface description:

## 3.1. Singleton:

	+ (instancetype)shareOne;
Definition	Singleton

## 3.2. Instantiation:

	+ (instancetype)share;
Definition	Instantiation

## 3.3. Show Log:

	- (void)showLog:(BOOL)showLog;
Definition	Display SDK running log
Parameter	showLog: whether to display the log

## 3.4. Set MicroLife Device:

	- (void)device:(NSArray *)bluetooth;
Definition	After setting up the Bailue device, when the Bluetooth of the mobile phone is turned on, it will automatically scan the surrounding devices and automatically connect to the device when it finds a matching device.
Parameter	bluetooth: MicroLifeDevice Class
	<pre>[self.sdk     device:@     [self.bp3g,self.bp4g,self.temperature,self.oxygen,self     .weight,self.bloodSugar]];</pre>

## 3.5. Set Auto Scan:

	- (void)autoScan:(BOOL)autoScan;
Definition	When CBManagerStatePoweredOn automatically opens scanning [Default: Open]
Parameter	autoScan: Auto Scan

## 3.6. Set Sacn stop Time:

	- (void)stopTime:(NSInteger)time;
Definition	Set the scan stop time
Parameter	time: Scan stop time

## 3.7. Set RSSI:

	- (void)rssi:(NSInteger)rssi;
Definition	Set the scan RSSI strength
Parameter	rssi: Scan RSSI strength

## 3.8. Start Sacn:

	- (void)startSacn;
Definition	Start Sacn

## 3.9. Cancel Scan :

	- (void)cancelScan;
Definition	Cancel Scan

## 3.10. Cancel All Connect:

	- (void)cancelAllConnect;
Definition	Cancel All Connect

## 3.11. Get the currently connected devices :

	- (NSArray *)findConnectedDevices;
Definition	Get the currently connected devices

## 3.12. When CentralManager state changed :

	- (void)getDidUpdateStateBlock:(didUpdateStateBlock)block;
Definition	Monitor mobile phone Bluetooth status
Parameter	void(^didUpdateStateBlock) (CBManagerState state)

## 3.13. Scan Device :

	- (void)getScanDeviceBlock:(scanDeviceBlock)scanDeviceBlock CancelScanBlock:(cancelScanBlock)cancelScanBlock;
Definition	Monitor scanning device status
Parameter	void(^scanDeviceBlock) (id device)
Parameter	void(^cancelScanBlock) (void)

## 3.14. Connect Device State :

	- (void)getConnectDeviceStateBlock:(connectDeviceStateBlock) connectDeviceStateBlock CancelAllDevicesConnectionBlock:(cancelAllDevicesConnectionBlock)cancelAllDevicesConnectionBlock;
Definition	Monitor the status of connected devices
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)

## 第4章 MicroLifeDevice interface description:

## 4.1. Singleton:

	+ (instancetype)shareWhithAuthorizationkey:(NSString *)key;
Definition	Singleton
Parameter	key: Authorization code

## 4.2. Add Device Model:

	- (void)addDeviceModel:(NSArray *)models;
Definition	Search for custom device models
Parameter	models: device name which can be included single or multiple
	<pre>[self.watchBP03 addDeviceModel:@[@"Watch BP 03"]];</pre>

## 4.3. Set Auto Scan:

	- (void)autoScan:(BOOL)autoScan;
Definition	When CBManagerStatePoweredOn automatically opens scanning [Default: Open]
Parameter	autoScan: Auto Scan

## 4.4. Auto Connect [Default: Open]:

	- (void)setAutoConnect:(BOOL)autoConnect;
Definition	Set Auto Connect [Default: Open]
Parameter	autoConnect: Auto Connect

## 4.5. ReConnect Time [Default: 0]:

	- (void)setReConnectTime:(NSInteger)reConnectIndex;
Definition	Set ReConnect Time [Default: 0], reconnection interval
Parameter	reConnectIndex: ReConnect Time

## 4.6. Set Sacn stop Time:

	- (void)stopTime:(NSInteger)time;
Definition	Set the scanning stop time, use in Bluetooth independent management mode
Parameter	time: Scan stop time

## 4.7. Set RSSI:

	- (void)rssi:(NSInteger)rssi;
Definition	Set the scan RSSI strength for Bluetooth standalone management mode
Parameter	rssi: Scan RSSI strength

## 4.8. Start Sacn:

	- (void)startSacn;
--	--------------------

Definition	Start Sacn, use Bluetooth independent management mode
------------	---

## 4.9. Cancel Scan:

	- (void)cancelScan;
Definition	Cancel Sacn, use Bluetooth independent management mode

## 4.10. Connect:

	- (void)connectDevice;
Definition	Connect

## 4.11. Disconnect:

	- (void)disconnectDevice;
Definition	Disconnect

## 4.12. Set Auto Reconnect:

	- (void)setAutoReconnect;
Definition	Set up automatic reconnection and enable it by default

## 4.13. Cancel Auto Reconnect:

	- (void)cancelAutoReconnect;
Definition	Disable automatic reconnection

## 4.14. Device Connected:

	- (void)deviceConnectedBlock:(deviceConnectedBlock)block;
Definition	The monitoring device has registered all preset services
Parameter	void(^deviceConnectedBlock) (void);

## 4.15. Device independent bluetooth:

	- (void)getDidUpdateStateBlock:(didUpdateStateBlock)didUpdateStateBlock ScanDeviceBlock:(scanDeviceBlock)scanDeviceBlock CancelScanBlock:(cancelScanBlock)cancelScanBlock ConnectDeviceStateBlock:(connectDeviceStateBlock)connectDeviceStateBlock CancelAllDevicesConnectionBlock:(cancelAllDevicesConnectionBlock)cancelAllDevicesConnectionBlock;
Definition	Use Bluetooth independent management mode to monitor mobile phone Bluetooth status, device scanning and connection status
Parameter	void(^didUpdateStateBlock) (CBManagerState state)
Parameter	void(^scanDeviceBlock) (id device)
Parameter	void(^cancelScanBlock) (void)

Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)

## 4.16. Read RSSI:

	- (void)getReadRSSIBlock:(readRSSIBlock)block;
Definition	Monitor the RSSI status of the device.
Parameter	typedef void(^readRSSIBlock) (NSNumber *RSSI, NSError * error);

## 4.17. Read Data Value :

	- (void)getReadDataValueBlock:(readDataValueBlock)block;
Definition	Monitor response status (raw value).
Parameter	void(^readDataValueBlock) (NSInteger CMD, NSData * value, NSError * error);

## 4.18. Read Data Model :

	- (void)getReadDataModelBlock:(readDataModelBlock)block;
Definition	Monitor response status.
Parameter	void(^readDataModelBlock) (NSInteger CMD, id model, NSError * error);

## 4.19. Read Valify item Error :

	- (void)getValidationErrorBlock:(validationErrorBlock)block;
Definition	Monitors error response status.
Parameter	void(^validationErrorBlock) (NSString *item, NSString *info, NSData * value);

## 第5章 Temperature Interface Description

- 5.1. Temperature is active, and the SDK only provides parsing functions.
- 5.2. Set the monitoring response status and obtain the corresponding data according to CMD

```
[self.temperature getReadDataModelBlock:^(NSInteger CMD, id
    _Nonnull model, NSError * _Nonnull error) {
    [weakSelf log:weakSelf.temperature CMD:CMD Model:model
        Error:error];
    switch (CMD) {
        case CMDTemperatureReadDeviceInfo:
        case CMDTemperatureReadMeasureData:
            break;
        default:
            break;
    }
}];
```

### 5.3. Read Device Info:

Definition	RRead Device Info
CMD	CMDTemperatureReadDeviceInfo
model	MicroLifeDeviceInfo

### 5.4. Read Measure Data:

Definition	Read Measure Data
CMD	CMDTemperatureReadMeasureData
model	MicroLifeTemperatureMeasureData

## 第6章 Blood Pressure Interface Description

- 6.1. Construct a corresponding manager based on the device type (3G/4G), set the monitoring response status, and obtain the corresponding data based on CMD

```
[self.bp3g getReadDataModelBlock:^(NSInteger CMD, id _Nonnull
model, NSError * _Nonnull error) {
[weakself log:weakself.bp3g CMD:CMD Model:model Error:error];
switch (CMD) {
case CMD3GReadHistorys:
case CMD3GClearAllHistorys:
case CMD3GReadUserAndVersionData:
case CMD3GWriteUser:
case CMD3GReadLastData:
case CMD3GClearLastData:
case CMD3GReadDeviceInfo:
break;
default:
break;
}
}];
```

```
[self.bp4g getReadDataModelBlock:^(NSInteger CMD, id _Nonnull
model, NSError * _Nonnull error) {
[weakself log:weakself.bp4g CMD:CMD Model:model Error:error];
switch (CMD) {
case CMD4GReadHistorys:
case CMD4GClearAllHistorys:
case CMD4GReadUserAndVersionData:
case CMD4GWriteUser:
case CMD4GReadDeviceInfo:
case CMD4GReadDeviceTime:
case CMD4GSyncTiming:
case CMD4GReadSerialNumber:
break;
default:
break;
}
}];
```



## 6.2. Read all history or current data from BPM:

	- (void)readAllHistorys;
Definition	Read all history or current data from BPM
CMD	CMD3GReadHistorys CMD4GReadHistorys
model	MicroLifeBloodPressureDRecord
	<pre> case CMD3GReadHistorys:     [weakSelf BPMBLEManagerResponseReadHistory:model];     break;  case CMD4GReadHistorys:     [weakSelf BPMBLEManagerResponseReadHistory:model];     break; </pre>

## 6.3. Clear all history data of the BPM:

	- (void)clearAllHistorys;
Definition	clear all history data of the bpm
CMD	CMD3GClearAllHistorys CMD4GClearAllHistorys
model	MicroLifeDataModel
	<pre> case CMD3GClearAllHistorys: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseClearHistory:data.success         .boolValue];     }     break;  case CMD4GClearAllHistorys: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseClearHistory:data.success         .boolValue];     }     break; </pre>

## 6.4. Disconnect the Bluetooth with BPM:

	- (void)disconnect;
Definition	Disconnect the Bluetooth with BPM
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)
	<pre> [self.sdk getConnectDeviceStateBlock:^(id _Nonnull device,     CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull     peripheral, CBPeripheralState state, NSError * _Nonnull error)     {         [weakSelf connectDeviceInfo:device PeripheralState:state];     } CancelAllDevicesConnectionBlock:^(         [weakSelf addLogWhitText:@"Cancel All Devices Connection"];     }]; </pre>

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

	- (void)readUserAndVersionData;
Definition	Read user ID and version data from BPM: After the device is successfully connected, the user ID and version data will be automatically read.
CMD	CMD3GReadUserAndVersionData CMD4GReadUserAndVersionData
model	Dictionary:{ UserInfo = "<MicroLifeUserInfo: User Information>"; Version = "<MicroLifeDeviceInfo: Device Information>"; }
	<pre> case CMD3GReadUserAndVersionData: {     MicroLifeUserInfo *userInfo = [model         valueForKey:@"UserInfo"];     MicroLifeDeviceInfo *version = [model         valueForKey:@"Version"];     [weakSelf         BPMBLEManagerResponseReadUserAndVersionData:userInfo         fo VersionData:version]; } break;  case CMD4GReadUserAndVersionData: {     MicroLifeUserInfo *userInfo = [model         valueForKey:@"UserInfo"];     MicroLifeDeviceInfo *version = [model         valueForKey:@"Version"];     [weakSelf         BPMBLEManagerResponseReadUserAndVersionData:userInfo         fo VersionData:version]; } break; </pre>

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

	- (void)writeUserID:(NSString *)ID Age:(NSInteger)age;
Definition	Write a new user ID to BPM
Parameter	ID: User ID。 age: age
CMD	CMD3GWriteUser CMD4GWriteUser
model	MicroLifeDataModel
	<pre> case CMD3GWriteUser: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseWriteUserID:data.success         .boolValue]; } break; </pre>

	<pre> case CMD4GWriteUser: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseWriteUserID:data.success         .boolValue];     }     break; </pre>
--	---

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

	- (void)readDeviceIDAndInfo;
Definition	Read device ID and info from BPM
CMD	CMD3GReadDeviceInfo CMD4GReadDeviceInfo
model	MicroLifeDeviceInfo
	<pre> case CMD3GReadDeviceInfo:     [weakSelf BPMBLEManagerResponseReadDeviceInfo:model];     break;  case CMD4GReadDeviceInfo:     [weakSelf BPMBLEManagerResponseReadDeviceInfo:model]; </pre>

#### 6.8. [4G Dedicated] Read device Time from BPM:

	- (void)readDeviceTime;
Definition	Read device Time from BPM
CMD	CMD4GReadDeviceTime
model	MicroLifeDeviceInfo
	<pre> case CMD4GReadDeviceTime:     [weakSelf BPMBLEManagerResponseReadDeviceTime:model];     break; </pre>

#### 6.9. [4G Dedicated] Write device Time to BPM:

	- (void)syncTiming;
Definition	Write device Time to BPM: After the device is successfully connected, the time will be automatically synchronized.
CMD	CMD4GSyncTiming
model	MicroLifeDataModel
	<pre> case CMD4GSyncTiming: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseWriteDeviceTime:data.success         .boolValue];     }     break; </pre>

## 6.10. Read last 1 data from the BPM: [Attention] will not respond!

	- (void)readLastData;(3G) - (BOOL)readLastData;(4G)
Definition	Read last 1 data from the BPM [Attention] will not respond!
CMD	CMD3GReadLastData CMD4GReadLastData
model	There is measurement data: MicroLifeBloodPressureDRecord No measurement data: MicroLifeDataModel
	<pre> case CMD3GReadLastData: {     if ([model         isKindOfClass:[MicroLifeBloodPressureDRecord             class]]) {         MicroLifeBloodPressureDRecord *dRecord = model;         [weakSelf             BPMBLEManagerResponseReadLastData:                 [dRecord.MData firstObject]             HistoryMeasurementNumber:dRecord                 .historyMeasuremeNumber.intValue             UserNumber:dRecord.userNumber.intValue             MAMState:dRecord.MAMState.intValue             IsNoData:YES];     } else {         // reply Null ACK         [weakSelf             BPMBLEManagerResponseReadLastData:nil             HistoryMeasurementNumber:nil             UserNumber:nil MAMState:nil IsNoData:NO];     } }  case CMD4GReadLastData: {     if ([model         isKindOfClass:[MicroLifeBloodPressureDRecord             class]]) {         MicroLifeBloodPressureDRecord *dRecord = model;         [weakSelf             BPMBLEManagerResponseReadLastData:                 [dRecord.MData firstObject]             HistoryMeasurementNumber:dRecord                 .historyMeasuremeNumber.intValue             UserNumber:dRecord.userNumber.intValue             MAMState:dRecord.MAMState.intValue             IsNoData:YES];     } else {         // reply Null ACK         [weakSelf             BPMBLEManagerResponseReadLastData:nil             HistoryMeasurementNumber:nil             UserNumber:nil MAMState:nil IsNoData:NO];     } } </pre>

## 6.11. Clear last 1 data of the BPM:[Attention] will not respond!

	- (void)readLastData;(3G) - (BOOL)readLastData;(4G)
Definition	Clear last 1 data of the BPM:[Attention] will not respond!
CMD	CMD3GClearLastData CMD4GClearLastData
model	MicroLifeDataModel
	<pre> case CMD3GClearLastData: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseClearLastData:data.success         .boolValue];     }     break;  case CMD4GClearLastData: {     MicroLifeDataModel *data = model;     [weakSelf         BPMBLEManagerResponseClearLastData:data.success         .boolValue];     }     break; </pre>

## 6.12. Read device SN from BPM:

	- (BOOL)readSerialNumber;
Definition	Read device SN from BPM
CMD	CMD4GReadSerialNumber
model	MicroLifeDeviceInfo
	<pre> case CMD4GReadSerialNumber: {     MicroLifeDeviceInfo *deviceInfo = model;     NSLog(@"Serial Number:%@",deviceInfo.sn);     }     break; </pre>

## 第7章 Weight interface description:

- 7.1. Set the monitoring response status and obtain the corresponding data according to CMD

```

[weakSelf weight getReadDataModelBlock:^(NSInteger CMD, id _Nonnull
model, NSError * _Nonnull error) {
    switch (CMD) {
        case CMDWeightWakeUpScale:
        case CMDWeightSleepScale:
        case CMDWeightMeasurementResult:
        case CMDWeightReadHistorys:
        case CMDWeightUpdateUserInfo:
        case CMDWeightClearAllUserInfo:
        case CMDWeightNoMoreOfflineData:
        case CMDWeightLowPower:
        case CMDWeightSyncSystemClock:
            break;
        default:
            break;
    }
    [weakSelf log:weakSelf.weight CMD:CMD Model:model Error:error];
}];

```

## 7.2. Weight WakeUp Scale:

Definition	Weight WakeUp Scale
CMD	CMDWeightWakeUpScale
	<pre> case CMDWeightWakeUpScale:     [weakSelf addLogWhitText:@"WakeUp Scale"];     [weakSelf writeUserData];     [weakSelf.weight readHistorys];     break; </pre>

## 7.3. Weight Sleep Scale:

Definition	Weight Sleep Scale
CMD	CMDWeightSleepScale
	<pre> case CMDWeightSleepScale:     [weakSelf addLogWhitText:@"Sleep Scale"];     [weakSelf.weight disconnect];     break; </pre>

## 7.4. Write User:

	- (void)writeUserData:(NSString *)ID Age:(NSInteger)age Gender:(MicroLifeUserGender)gender Height:(NSInteger)height RoleType:(MicroLifeUserRoleType)roleType Weight:(float)weight Resistance:(NSInteger)resistance;
Definition	Write User Note: The SDK also sends user information to the scale, so you may receive this callback message frequently
CMD	CMDWeightUpdateUserInfo
	<pre>case CMDWeightUpdateUserInfo:     [weakself addLogWhitText:@"Update UserInfo"];     break;</pre>

## 7.5. No More OfflineData:

Definition	No More OfflineData
CMD	CMDWeightNoMoreOfflineData
	<pre>case CMDWeightNoMoreOfflineData:     [weakself addLogWhitText:@"No More OfflineData"];     break;</pre>

## 7.6. Low Power:

Definition	Low Power
CMD	CMDWeightLowPower
	<pre>case CMDWeightLowPower:     [weakself addLogWhitText:@"Low Power"];     break;</pre>

## 7.7. Sync System Clock:

Definition	Sync System Clock
CMD	CMDWeightSyncSystemClock
model	MicroLifeDeviceInfo
	<pre>case CMDWeightSyncSystemClock:     [weakself addLogWhitText:@"Sync System Clock"];     break;</pre>

## 7.8. Clear All Users:

	- (void)clearAllUsers;
Definition	Clear All Users
CMD	CMDWeightClearAllUserInfo
	<pre>case CMDWeightClearAllUserInfo:     [weakself addLogWhitText:@"Clear All UserInfo"];     break;</pre>

## 7.9. Read all history data:

	- (void)readHistorys;
Definition	Read all history data
CMD	CMDWeightReadHistorys
model	MicroLifeBodyFat
	<pre> case CMDWeightReadHistorys:     [weakself addLogWhitText:@"Read Historys"];     weakself.currUser.resistance = ((MicroLifeBodyFat     *)model).resistance;     [weakself     saveNSUserDefaults:weakself.currUser.resistance     Key:@"resistance"];     break; </pre>

## 7.10. Read current data:

Definition	Read current data
CMD	CMDWeightMeasurementResult
model	MicroLifeBodyFat
	<pre> case CMDWeightMeasurementResult:     [weakself addLogWhitText:@"Measurement Result"];     weakself.currUser.resistance = ((MicroLifeBodyFat     *)model).resistance;     [weakself     saveNSUserDefaults:weakself.currUser.resistance     Key:@"resistance"];     break; </pre>

## 7.11. Disconnect the Bluetooth with BPM:

	- (void)disconnect;
Definition	Disconnect the Bluetooth with BPM
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)
	<pre> [self.sdk getConnectDeviceStateBlock:^(id _Nonnull device, CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull peripheral, CBPeripheralState state, NSError * _Nonnull error) {     [weakself connectDeviceInfo:device PeripheralState:state]; } CancelAllDevicesConnectionBlock:^(     [weakself addLogWhitText:@"Cancel All Devices Connection"];     [weakself.weight startScan]; }]; </pre>



## 7.12. Body fat machine human body composition analysis:

7.12.1. Just import the header file.

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

7.12.2. initialchange:

	- (void)getInstance;
Definition	Instance
Parameter	
Parameter	

7.12.3. Set user information:

	- (void)setUserInfo_height:(double)height weight:(double)weight resistance:(int)resistance sex:(int)sex age:(int)age csRoleType:(BTRoleType)typ;
Definition	Set user information
Parameter	- (double)getMoisture; - (double)getMuscle; - (double)getProtein; - (double)getBMR; - (double)getBone; - (double)getVisceralfat; - (double)getBoneMuscle; - (double)getBMI; - (int)getBodyAge; - (int)getBodyScore;
	<pre>int sex = (self.currUser.gender == MicroLifeUserGenderFemale)?0:1; BTRoleType type = (self.currUser.roleType ==     MicroLifeUserRoleTypeNormal)?BTRoleTypeNormal:BTRoleTypeSportsman; BelterScaleInfo *scaleInfo = [BelterScaleInfo getInstance]; [scaleInfo setUserInfo_height:self.currUser.height.doubleValue weight:bodyFat.weight.doubleValue     resistance:bodyFat.resistance.intValue sex:sex age:self.currUser.age.intValue csRoleType:type]; bodyFat.bmi = @(scaleInfo.getBMI); bodyFat.visceralfat = @(scaleInfo.getVisceralfat); bodyFat.bmr = @(scaleInfo.getBMR); bodyFat.muscle = @(scaleInfo.getMuscle); bodyFat.protein = @(scaleInfo.getProtein); bodyFat.bone = @(scaleInfo.getBone); bodyFat.moisture = @(scaleInfo.getMoisture); bodyFat.boneMuscle = @(scaleInfo.getBoneMuscle); bodyFat.bodyScore = @(scaleInfo.getBodyScore); bodyFat.bodyAge = @(scaleInfo.getBodyAge);</pre>

## 第8章 Oxygen Interface Description

## 8.1. Set the monitoring response status and obtain the corresponding data according to CMD

```

[self.oxygen getReadDataModelBlock:^(NSInteger CMD, id _Nonnull
model, NSError * _Nonnull error) {
[weakself log:weakself.oxygen CMD:CMD Model:model Error:error];
switch (CMD) {
case CMDOxygenVolumeTracingData:
[weakself showOxygen:model];
break;
case CMDOxygenSaturationAndPulseRateData:
break;
case CMDOxygenSaturationAndPulseRateAlarmLimits:
break;
default:
break;
}
}];

```

## 8.2. Read Oxygen Volume Tracing Data:

Definition	Oxygen Volume Tracing Data
CMD	CMDOxygenVolumeTracingData
model	MicroLifeOxygenData
	<pre> case CMDOxygenVolumeTracingData: [weakself showOxygen:model]; break; </pre>

## 8.3. Read blood oxygen saturation and pulse rate alarm limit:

	- (void)readAlarmInfo;
Definition	Read blood oxygen saturation and pulse rate alarm limit
CMD	CMDOxygenSaturationAndPulseRateData
model	MicroLifeOxygenData
	<pre> case CMDOxygenSaturationAndPulseRateData: break; </pre>

## 8.4. Write blood oxygen saturation and pulse rate alarm limit:

	-(void)alarmSPOMax:(NSInteger)maxSPO SPOmin:(NSInteger)minSPO PlusMax:(NSInteger)maxPlus Plusmin:(NSInteger)minPlus;
Definition	Write blood oxygen saturation and pulse rate alarm limit
Parameter	<p>maxSPO The lower limit of blood oxygen alarm limit, the value is 50-100.</p> <p>minSPO The upper limit of blood oxygen alarm limit, the value is 50-100.</p> <p>maxPlus The lower limit of the pulse rate alarm limit, the value is 5-250. It must be a multiple of 5 and cannot be equal to 0.</p> <p>minPlus The upper limit of the pulse rate alarm limit, the value is 5-250, must be a multiple of 5, and cannot be equal to 0.</p>
CMD	CMDOxygenSaturationAndPulseRateAlarmLimits
model	MicroLifeOxygenData
	<pre>case CMDOxygenSaturationAndPulseRateAlarmLimits:     break;</pre>

## 第9章 WatchBP O3 Interface Description

## 9.1. Set the monitoring response status and obtain the corresponding data according to CMD

```
[self.watchBP03 getReadDataModelBlock:^(NSInteger CMD, id
_Nonnull model, NSError * _Nonnull error) {
    switch (CMD) {
        case CMDWatchBP03ReadCBPData:
        case CMDWatchBP03ReadDeviceInfo:
        case CMDWatchBP03ReadDeviceTime:
        case CMDWatchBP03ReadSerialNumber:
        case CMDWatchBP03ReadBTModuleName:
        case CMDWatchBP03ClearAllHistorys:
        case CMDWatchBP03WriteUser:
        case CMDWatchBP03WriteSettingValues:
        case CMDWatchBP03ReadFunctionSettingValue:
        case CMDWatchBP03WriteDeviceTime:
        case CMDWatchBP03ReadSettingValues:
        case CMDWatchBP035SReadSettingValues:
        case CMDWatchBP03ReadHistorys:
        case CMDWatchBP03ReadUserAndVersionData:
            break;
        default:
            break;
    }
}
```

## 9.2. Read all history or current data from BPM:

	- (void)readAllHistorys;
Definition	Read all history or current data from BPM
CMD	CMDWatchBP03ReadHistorys
model	With measurement data: MicroLifeWatchBPDRecord No measurement data: MicroLifeDataModel
	<pre>case CMDWatchBP03ReadHistorys:     if ([model isKindOfClass:[MicroLifeWatchBPDRecord class]]) {         [weakSelf WB03BLEManagerResponseReadAllHistorys:model];     } else {         // reply Null ACK     }     break;</pre>

## 9.3. Read CBP data by index from BPM :

	- (void)readCBPDataWithIndex:(int)index Dformat:(Dformat)dformat;
Definition	Read diagnostic mode history data from BPM
Parameter	index: is CBP memory index dformat: Data format which BPM sent out. It is same as what APP requested. NoCBPRaw: No CBP raw data LowCBPRaw: low resolution CBP data (sampling rate =16Hz) FullCBPRaw: full CBP raw data (sampling rate=256Hz)
CMD	CMDWatchBPO3ReadCBPData
model	There is measurement data: MicroLifeWatchBPCBPdataAndCalCBP No measurement data: MicroLifeDataModel
	<pre> case CMDWatchBPO3ReadCBPData:     if ([model         isKindOfClass:[MicroLifeWatchBPCBPdataAndCalCBP             class]]) {         [weakSelf WB03BLEManagerResponseReadCBPData:model             IsNoData:NO];     } else {         [weakSelf             WB03BLEManagerResponseReadCBPData:nil             IsNoData:YES];     }     break; </pre>

## 9.4. Clear all history data of the BPM :

	- (void)clearAllHistorys;
Definition	clear all history data of the bpm
CMD	CMDWatchBPO3ClearAllHistorys
model	MicroLifeDataModel
	<pre> case CMDWatchBPO3ClearAllHistorys: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WB03BLEManagerResponseClearHistory:dataModel         .success.boolValue]; } break; </pre>

## 9.5. Disconnect the Bluetooth with BPM:

	- (void)disconnect;
Definition	Disconnect the Bluetooth with BPM
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)
	<pre>// Device connection status response [self.sdk getConnectDeviceStateBlock:^(id _Nonnull device,     CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull     peripheral, CBPeripheralState state, NSError * _Nonnull error) {     weakSelf.setting2g.hidden = weakSelf.setting5g.hidden = YES;     [weakSelf connectDeviceInfo:device PeripheralState:state]; } CancelAllDevicesConnectionBlock:^(     [weakSelf addLogWhitText:@"Cancel All Devices Connection"]; }];</pre>

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

	- (void)readUserAndVersionData;
Definition	Read user ID and version data from BPM: After the device is successfully connected, the user ID and version data will be automatically read.
CMD	CMDWatchBPO3ReadUserAndVersionData
model	Dictionary:{ UserInfo = "<MicroLifeUserInfo: User Information>"; Version = "<MicroLifeDeviceInfo: Device Information>"; }
	<pre>case CMDWatchBPO3ReadUserAndVersionData: {     MicroLifeUserInfo *userInfo = [model         valueForKey:@"UserInfo"];     MicroLifeDeviceInfo *version = [model         valueForKey:@"Version"];     [weakSelf         WB03BLEManagerResponseReadUserAndVersionData:userInfo         VersionData:version]; } break;</pre>

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

	- (void)writeUserID:(NSString *)ID;
Definition	Write a new user ID to BPM
Parameter	ID: User ID.
CMD	CMDWatchBPO3WriteUser
model	MicroLifeDataModel
	<pre>case CMDWatchBPO3WriteUser: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WB03BLEManagerResponseWriteUserID:dataModel         .success.boolValue]; } break;</pre>

## 9.8. Read ABPM setting values from BPM:

	- (void)readSettingValues;
Definition	Read ABPM setting values from BPM
CMD	CMDWatchBP03ReadSettingValues CMDWatchBP035SReadSettingValues
model	MicroLifeWatchBPSettingValues
	<pre> case CMDWatchBP03ReadSettingValues: case CMDWatchBP035SReadSettingValues:     [weakSelf         WB03BLEManagerResponseReadSettingValue:model]; break; </pre>

## 9.9. Write ABPM setting values to BPM:

	<p>-</p> <p>(BOOL)writeSettingValuesWithSelectedABPMStart:(NSInteger) ABPMStart ABPMEnd:(NSInteger)ABPMEnd ABPMInt_first:(MeasurementTime)ABPMInt_first ABPMInt_second:(MeasurementTime)ABPMInt_second HI_infPressure:(HlinfPressure)HI_infPressure SW_checkhide:(BOOL)SW_checkhide SW_SEL_silent:(BOOL)SW_SEL_silent CBP_zone1_meas_off:(BOOL)CBP_zone1_meas_off CBP_zone2_meas_off:(BOOL)CBP_zone2_meas_off CBPInt_first:(MeasurementTime)CBPInt_first CBPInt_second:(MeasurementTime)CBPInt_second;</p>
Definition	Write ABPM setting values to BPM.
Parameter	<p>ABPMStart The starting time of the first measurement time zone</p> <p>ABPMEnd The end time of the first measurement time zone</p> <p>ABPMInt_first The interval of the first measurement time zone (valid range: 5, 10, 15, 20, 30, and 60)</p> <p>ABPMInt_second The interval of the second measurement time zone (valid range: 5, 10, 15, 20, 30, and 60)</p> <p>HI_infPressure Highest inflation pressure of Auto mode</p> <p>SW_checkhide The interval of the CBP first measurement time zone Note: CBPInt_first should multiple time than ABPMInt_first.</p> <p>SW_SEL_silent Hide(true)/Show(false) readings after measurement</p> <p>CBP_zone1_meas_off the first time zone of CBP measurement true:disabled/false:enabled</p> <p>CBP_zone2_meas_off the second time zone of CBP measurement true:disabled/false:enabled</p>

	<p>CBPInt_first The interval of the CBP first measurement time zone (valid range: 5, 10, 15, 20, 30, and 60) Note: CBPInt_first should multiple time than ABPMInt_first.</p> <p>CBPInt_second The interval of the CBP second measurement time zone (valid range: 5, 10, 15, 20, 30, and 60) Note: CBPInt_second should multiple time than ABPMInt_second.</p>
CMD	CMDWatchBP03WriteSettingValues
model	MicroLifeDataModel
	<pre>[self.watchBP03 writeSettingValuesWithSelectedABPMStart:self.settingValue .ABPMStart.integerValue ABPMEnd:self.settingValue.ABPMEnd.integerValue ABPMInt_first:self.settingValue.ABPMInt_first ABPMInt_second:self.settingValue.ABPMInt_second HI_infPressure:self.settingValue.HI_infPressure SW_checkhide:self.settingValue.SW_checkhide.boolValue SW_SEL_silent:self.settingValue.SW_SEL_silent.boolValue CBP_zone1_meas_off:self.settingValue.CBP_zone1_meas_off .boolValue CBP_zone2_meas_off:self.settingValue.CBP_zone2_meas_off .boolValue CBPInt_first:self.settingValue.CBPInt_first CBPInt_second:self.settingValue.CBPInt_second];</pre>
	<pre>case CMDWatchBP03WriteSettingValues: case CMDWatchBP035SWriteSettingValues: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WB03BLEManagerResponseWriteSettingValues:dataModel         .success.boolValue]; } break;</pre>

#### 9.10. Write Setting Values from BPM(5 schedule):

	- (BOOL)writeSettingValuesWithSettingValues:(NSMutableArray *)settingValues HI_infPressure:(HIinfPressure)HI_infPressure SW_checkhide:(BOOL)SW_checkhide;
Definition	Write Setting Values from BPM(5 schedule)
Parameter	settingValues MicroLifeWatchBPSettingValues HI_infPressure HI_infPressure SW_checkhide SW_checkhide
CMD	CMDWatchBP035SWriteSettingValues
model	MicroLifeDataModel
	<pre>[self.watchBP03 writeSettingValuesWithSettingValues:self .settingValue.settingValues HI_infPressure:self.settingValue .HI_infPressure SW_checkhide:self.settingValue .SW_checkhide.boolValue];</pre>



	<pre> case CMDWatchBP03WriteSettingValues: case CMDWatchBP035SWriteSettingValues: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WB03BLEManagerResponseWriteSettingValues:dataModel         .success.boolValue];     }     break; </pre>
--	--

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

	- (void)readDeviceIDAndInfo;
Definition	Read device ID and info from BPM
CMD	CMDWatchBP03ReadDeviceInfo
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBP03ReadDeviceInfo:     [weakSelf         WB03BLEManagerResponseReadDeviceIDAndInfo:model];     break; </pre>

## 9.12. Read device Time from BPM:

	- (void)readDeviceTime;
Definition	Read device Time from BPM
CMD	CMDWatchBP03ReadDeviceTime
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBP03ReadDeviceTime:     [weakSelf WB03BLEManagerResponseReadDeviceTime:model];     break; </pre>

## 9.13. Write device Time to BPM:

	- (void)writeDeviceTime;
Definition	Write device Time to BPM: After the device is successfully connected, the time will be automatically synchronized.
CMD	CMDWatchBP03WriteDeviceTime
model	MicroLifeDataModel
	<pre> case CMDWatchBP03WriteDeviceTime: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WB03BLEManagerResponseWriteDeviceTime:dataModel         .success.boolValue];     }     break; </pre>

## 9.14. Read BPM function setting value from BPM:

	- (void)readFunctionSettingValue;
Definition	Read BPM function setting value from BPM
CMD	CMDWatchBP03ReadFunctionSettingValue
model	MicroLifeWatchBPFunctionSettingValues
	<pre> case CMDWatchBP03ReadFunctionSettingValue:     [weakSelf      WB03BLEManagerResponseReadFunctionSettingValue:model];     break; </pre>

## 9.15. Read BT module name from BPM:

	- (void)readBTModuleName;
Definition	Read BT module name from BPM
CMD	CMDWatchBP03ReadBTModuleName
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBP03ReadBTModuleName: {     MicroLifeDeviceInfo *deviceInfo = model;     [weakSelf      WB03BLEManagerResponseReadBTModuleName:deviceInfo      .BTModuleName]; } break; </pre>

## 9.16. Read device SN from BPM:

	- (void)readSerialNumber;
Definition	Read device SN from BPM
CMD	CMDWatchBP03ReadSerialNumber
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBP03ReadSerialNumber: {     MicroLifeDeviceInfo *deviceInfo = model;     NSLog(@"Serial Number:%@",deviceInfo.sn); } break; </pre>

## 第10章 WatchBP Home Interface Description

## 10.1. Set the monitoring response status and obtain the corresponding data according to CMD

```

[self.watchBPHome getReadDataModelBlock:^(NSInteger CMD, id _Nonnull model, NSError * _Nonnull
error) {
    [weakSelf log:weakSelf.watchBPHome CMD:CMD Model:model Error:error];
    switch (CMD) {
        case CMDWatchBPHomeReadUsualModeHistoryData:
        case CMDWatchBPHomeReadUsualModeHistoryDataIncludeEachMeasurement:|
        case CMDWatchBPHomeReadDiagnosticModeHistoryData:
        case CMDWatchBPHomeReadNocturnalModeSetting:
        case CMDWatchBPHomeReadDeviceInfo:
        case CMDWatchBPHomeReadDeviceTime:
        case CMDWatchBPHomeReadSerialNumber:
        case CMDWatchBPHomeReadUserAndVersionData:
        case CMDWatchBPHomeReadNocturnalModeHistoryData:
        case CMDWatchBPHomeClearHistoryDataMode:
        case CMDWatchBPHomeClearCurrentModeHistoryData:
        case CMDWatchBPHomeWriteDeviceTime:
        case CMDWatchBPHomeWriteUserID:
        case CMDWatchBPHomeChangeNocturnalMode:
        case CMDWatchBPHomeReadDeviceSettingOfMeasurement:
        case CMDWatchBPHomeWriteDeviceSettingOfMeasurement:
            break;
        default:
            break;
    }
}];

```

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

	- (void)readUsualModeHistoryData;
Definition	Read all history or current data from BPM
CMD	CMDWatchBPHomeReadUsualModeHistoryData
model	With measurement data: MicroLifeWatchBPDRecord No measurement data: MicroLifeDataModel
	<pre> case CMDWatchBPHomeReadUsualModeHistoryData: {     if ([model isKindOfClass:[MicroLifeWatchBPDRecord class]]) {         [weakSelf             WBPBLEManagerResponseReadUsualModeHistoryData:             model IsNoData:NO];     } else {         [weakSelf             WBPBLEManagerResponseReadUsualModeHistoryData:             nil IsNoData:YES];     } } break; </pre>

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

	- (void)readDiagnosticModeHistoryData;
Definition	Read diagnostic mode history data from BPM
CMD	CMDWatchBPHomeReadDiagnosticModeHistoryData
model	With measurement data: MicroLifeWatchBPDDiagnosticDRecord No measurement data: MicroLifeDataModel
	<pre> case CMDWatchBPHomeReadDiagnosticModeHistoryData: {     if ([model         isKindOfClass:[MicroLifeWatchBPDDiagnosticDRecord             class]]) {         [weakSelf             WBPBLEManagerResponseReadDiagnosticModeHistory                 Data:model IsNoData:NO];     } else {         [weakSelf             WBPBLEManagerResponseReadDiagnosticModeHistory                 Data:nil IsNoData:YES];     } } break; </pre>

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

	- (void)clearHistoryDataWithSelectedUsualMode:(BOOL)clearUsualMode DiagnosticMode:(BOOL)clearDiagnosticMode NocturnalMode:(BOOL)clearNocturnalMode;
Definition	Clear selected mode history data of the BPM
Parameter	clearUsualMode clear Usual Mode clearDiagnosticMode clear Diagnostic Mode clearNocturnalMode clear Nocturnal Mode
CMD	CMDWatchBPHomeClearHistoryDataMode
model	MicroLifeDataModel
	<pre> case CMDWatchBPHomeClearHistoryDataMode: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBPBLEManagerResponseClearCurrentModeHistoryData:dataModel.success.boolValue]; } break; </pre>

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

	- (void)clearCurrentModeHistoryData;
Definition	Clear current mode history data of the BPM
CMD	CMDWatchBPHomeClearCurrentModeHistoryData
model	MicroLifeDataModel

	<pre> case CMDWatchBPHomeClearCurrentModeHistoryData: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBPBLEManagerResponseClearCurrentModeHistoryData:d         ataModel.success.boolValue];     }     break; </pre>
--	--

## 10.6. Disconnect the Bluetooth with BPM:

	- (void)disconnect;
Definition	Disconnect the Bluetooth with BPM
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)
	<pre> [self.sdk getConnectDeviceStateBlock:^(id _Nonnull device,     CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull     peripheral, CBPeripheralState state, NSError * _Nonnull error)     {         [weakSelf connectDeviceInfo:device PeripheralState:state];     } CancelAllDevicesConnectionBlock:^(         [weakSelf addLogWhitText:@"Cancel All Devices Connection"];     }]); </pre>

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

	- (void)readUserAndVersionData;
Definition	Read user ID and version data from BPM: After the device is successfully connected, the user ID and version data will be automatically read.
CMD	CMDWatchBPHomeReadUserAndVersionData
model	Dictionary:{ UserInfo = "<MicroLifeUserInfo: User Information>"; Version = "<MicroLifeDeviceInfo: Device Information>"; }
	<pre> case CMDWatchBPHomeReadUserAndVersionData: {     MicroLifeUserInfo *userInfo = [model         valueForKey:@"UserInfo"];     MicroLifeDeviceInfo *version = [model         valueForKey:@"Version"];     [weakSelf         WBPBLEManagerResponseReadUserAndVersionData:userIn         fo VersionData:version];     }     break; </pre>

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

	- (void)writeUserID:(NSString *)ID;
Definition	Write a new user ID to BPM
Parameter	ID: User ID.
CMD	CMDWatchBPHomeWriteUserID
model	MicroLifeDataModel

	<pre> case CMDWatchBPHomeWriteUserID: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBPBLEManagerResponseWriteUserID:dataModel         .success.boolValue];     }     break; </pre>
--	--

## 10.9. Read ABPM setting values from BPM:

	- (void)readSettingValues;
Definition	Read ABPM setting values from BPM
CMD	CMDWatchBPO3ReadSettingValues CMDWatchBPO35SReadSettingValues
model	MicroLifeWatchBPSettingValues
	<pre> case CMDWatchBPO3ReadSettingValues: case CMDWatchBPO35SReadSettingValues:     [weakSelf         WBO3BLEManagerResponseReadSettingValue:model];     break; </pre>

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

	- (void)readDeviceIDAndInfo;
Definition	Read device ID and info from BPM
CMD	CMDWatchBPHomeReadDeviceInfo
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPHomeReadDeviceInfo:     [weakSelf         WBPBLEManagerResponseReadDeviceIDAndInfo:model];     break; </pre>

## 10.11. Read device Time from BPM:

	- (void)readDeviceTime;
Definition	Read device Time from BPM
CMD	CMDWatchBPHomeReadDeviceTime
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPHomeReadDeviceTime:     [weakSelf WBPBLEManagerResponseReadDeviceTime:model];     break; </pre>

## 10.12. Write device Time to BPM:

	- (void)writeDeviceTime;
Definition	Write device Time to BPM: After the device is successfully connected, the time will be automatically synchronized.
CMD	CMDWatchBPHomeWriteDeviceTime
model	MicroLifeDataModel

	<pre>case CMDWatchBPHomeWriteDeviceTime: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBPBLEManagerResponseWriteDeviceTime:dataModel         .success.boolValue];     }     break;</pre>
--	--

## 10.13. Read nocturnal mode setting :

	- (void)readNocturnalModeSetting;
Definition	Read nocturnal mode setting
CMD	CMDWatchBPHomeReadNocturnalModeSetting
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPHomeReadNocturnalModeSetting:     [weakSelf         WBPBLEManagerResponseReadNocturnalModeSetting:model];     break; </pre>

## 10.14. Change nocturnal mode setting :

	- (void)changeNocturnalModeSettingOn:(BOOL)open StartYear:(NSInteger)year StartMonth:(NSInteger)month StartDay:(NSInteger)day StartHour:(NSInteger)hour;
Definition	Change nocturnal mode setting Note : This command requires matching hardware to set. You can use "readDeviceIDAndInfo(MicroLifeDeviceInfo.openNocturnalMode)" to check if the device supports it.
Parameter	open Nocturnal ON/OFF year year month month day day hour hour
CMD	CMDWatchBPHomeChangeNocturnalMode
model	MicroLifeDataModel
	<pre> case CMDWatchBPHomeChangeNocturnalMode: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBPBLEManagerResponseChangeNocturnalModeSetting:dataModel.success.boolValue]; } break; </pre>



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

	- (void)readNocturnalModeHistoryData;
Definition	Read Nocturnal mode history data from BPM,If memory bank is blank, BPM will respond NullACK
CMD	CMDWatchBPHomeReadNocturnalModeHistoryData
model	There is measurement data: MicroLifeWatchBPNocturnalModeDRecord No measurement data: MicroLifeDataModel
	<pre> case CMDWatchBPHomeReadNocturnalModeHistoryData: {     if ([model         isKindOfClass:             [MicroLifeWatchBPNocturnalModeDRecord class]]) {         [weakSelf             WBPBLEManagerResponseReadNocturnalPatternHistoryData:model IsNoData:NO];     } else {         [weakSelf             WBPBLEManagerResponseReadNocturnalPatternHistoryData:nil IsNoData:YES];     } } break; </pre>

## 10.16. Read device SN from BPM:

	- (void)readSerialNumber;
Definition	Read device SN from BPM
CMD	CMDWatchBPHomeReadSerialNumber
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPHomeReadSerialNumber:     [weakSelf         WBPBLEManagerResponseReadSerialNumber:model]; break; </pre>

## 10.17. Read measurement setting:

	- (BOOL)readDeviceSettingOfMeasurement
Definition	Read measurement setting
CMD	CMDWatchBPHomeReadDeviceSettingOfMeasurement
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPHomeReadDeviceSettingOfMeasurement: {     MicroLifeDeviceInfo *deviceInfo = model;     NSLog(@"Read Device Setting Of Measurement:%@",deviceInfo.parseDictionary); } break; </pre>

## 10.18. Write measurement setting :

	- (BOOL)writeDeviceSettingOfMeasurement:(MeasurementTimes)measurementTimes RestTime:(NSInteger)restTime IntervalTime:(NSInteger)intervalTime ExcludeAverage:(ExcludeAverage)excludeAverage SWAfib:(BOOL)SW_Afib;
Definition	Write measurement setting
Parameter	measurementTimes Measurement times: (Default =3) It's the measurement times (1~3) for usual mode. restTime Rest time: (Default = 0) It's interval before 1st measurement. intervalTime Interval time: (Default = 15) It's interval between each measurement. excludeAverage Exclude average: (Default = 0) 0: average all measurements 1: average excludes 1st measurement SW_Afib SW_Afib enable the Afib function. (this setting is valid with the Afib option enable in "BPMSetting2".)
CMD	CMDWatchBPHomeWriteDeviceSettingOfMeasurement
model	MicroLifeDataModel
	<pre> case CMDWatchBPHomeReadDeviceSettingOfMeasurement: {     MicroLifeDeviceInfo *deviceInfo = model;     NSLog(@"Read Device Setting Of         Measurement:%@",deviceInfo.parseDictionary); }  break; case CMDWatchBPHomeWriteDeviceSettingOfMeasurement: {     MicroLifeDataModel *dataModel = model;     NSLog(@"Write Device Setting Of         Measurement:%@",dataModel.success); }  break; </pre>

## 10.19. Read usual mode history data (include each measurement) from BPM :

	- (BOOL)readUsualModeHistoryDataIncludeEachMeasurement;
Definition	Read usual mode history data (include each measurement)
CMD	CMDWatchBPHomeReadUsualModeHistoryDataIncludeEach Measurement
model	MicroLifeWatchBPDRecord
	<pre> case CMDWatchBPHomeReadUsualModeHistoryDataIncludeEachMeasurement: {     MicroLifeWatchBPDRecord *dRecord = model;     NSString *log = [NSString stringWithFormat:@"dRecord historyMeasuremeNumber:%@ MData:%ld",dRecord.historyMeasuremeNumber,dRecord.MData.count];     [self addLogWhitText:log]; }  break; </pre>

## 第11章 WatchBP Office Interface Description

## 11.1. Set the monitoring response status and obtain the corresponding data according to CMD

```

[self.watchBPOffice getReadDataModelBlock:^(NSInteger
    CMD, id _Nonnull model, NSError * _Nonnull error) {
    [weakself log:weakself.watchBPOffice CMD:CMD
        Model:model Error:error];
    switch (CMD) {
        case CMDWatchBPOfficeReadHistorys:
            [weakself
                WBOBLEManagerResponseReadAllHistorys:model
            ];
            break;
        case CMDWatchBPOfficeReadUserAndVersionData: {
            MicroLifeUserInfo *userInfo = [model
                valueForKey:@"UserInfo"];
            weakself.editUserID.text = userInfo.bpmUserID;
        }
        break;
        case CMDWatchBPOfficeReadSettingValues:
            [weakself
                WBOBLEManagerResponseReadSettingValue:model
            ];
            break;
    }
}

```

## 11.2. Read all history or current data from BPM:

	- (void)readAllHistorys;
Definition	Read all history or current data from BPM
CMD	CMDWatchBPOfficeReadHistorys
model	With measurement data: MicroLifeWatchBPRecord No measurement data: MicroLifeDataModel
	<pre> <b>case</b> CMDWatchBPOfficeReadHistorys:     <b>if</b> ([model         isKindOfClass:[MicroLifeWatchBPRecord             class]]) {         [weakself             WBOBLEManagerResponseReadAllHistorys                 :model];     } <b>else</b> {         // reply Null ACK     }     <b>break</b>; </pre>

## 11.3. Read CBP data by index from BPM:

	- (void)readCBPDataWithIndex:(int)index Dformat:(Dformat)dformat;
Definition	Read diagnostic mode history data from BPM
Parameter	index: is CBP memory index dformat: Data format which BPM sent out. It is same as what APP requested. NoCBPRaw: No CBP raw data LowCBPRaw: low resolution CBP data (sampling rate =16Hz) FullCBPRaw: full CBP raw data (sampling rate=256Hz)
CMD	CMDWatchBPOfficeReadCBPData
model	There is measurement data: MicroLifeWatchBPCBPdataAndCalCBP No measurement data: MicroLifeDataModel
	<pre> case CMDWatchBPOfficeReadCBPData: {     if ([model         isKindOfClass:         [MicroLifeWatchBPCBPdataAndCalCBP          class]]) {         [weakSelf          WBOBLEManagerResponseReadCBPData:model          IsNoData:NO];     } else {         [weakSelf          WBOBLEManagerResponseReadCBPData:nil          IsNoData:YES];     } } break; </pre>

## 11.4. Clear all history data of the BPM:

	- (void)clearAllHistorys;
Definition	clear all history data of the bpm
CMD	CMDWatchBPOfficeClearAllHistorys
model	MicroLifeDataModel
	<pre> case CMDWatchBPOfficeClearAllHistorys: {     MicroLifeDataModel *dataModel = model;     [weakSelf      WBOBLEManagerResponseClearHistory:dataMo      del.success.boolValue]; } break; </pre>

## 11.5. Disconnect the Bluetooth with BPM:

	- (void)disconnect;
Definition	Disconnect the Bluetooth with BPM
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)
	<pre> [self.sdk getConnectDeviceStateBlock:^(id _Nonnull device, CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull peripheral, CBPeripheralState state, NSError * _Nonnull error) { [weakself connectDeviceInfo:device PeripheralState:state]; } CancelAllDevicesConnectionBlock:^( [weakself addLogWhitText:@"Cancel All Devices Connection"]; }]; </pre>

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

	- (void)readUserAndVersionData;
Definition	Read user ID and version data from BPM: After the device is successfully connected, the user ID and version data will be automatically read.
CMD	CMDWatchBPOfficeReadUserAndVersionData
model	Dictionary:{ UserInfo = "<MicroLifeUserInfo: User Information>"; Version = "<MicroLifeDeviceInfo: Device Information>"; }
	<pre> case CMDWatchBPOfficeReadUserAndVersionData: { MicroLifeUserInfo *userInfo = [model valueForKey:@"UserInfo"]; MicroLifeDeviceInfo *version = [model valueForKey:@"Version"]; [weakself WBOBLEManagerResponseReadUserAndVersionD ata:userInfo VersionData:version]; } break; </pre>

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

	- (void)writeUserID:(NSString *)ID;
Definition	Write a new user ID to BPM
Parameter	ID: User ID.
CMD	CMDWatchBPOfficeWriteUser
model	MicroLifeDataModel
	<pre> case CMDWatchBPOfficeWriteUser: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBOBLEManagerResponseWriteUserID:dataModel         el.success.boolValue];     }     break; </pre>

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

	- (void)readDeviceIDAndInfo;
Definition	Read device ID and info from BPM
CMD	CMDWatchBPOfficeReadDeviceInfo
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPOfficeReadDeviceInfo:     [weakSelf         WBOBLEManagerResponseReadDeviceIDAndInfo         :model];     break; </pre>

## 11.9. Read device Time from BPM:

	- (void)readDeviceTime;
Definition	Read device Time from BPM
CMD	CMDWatchBPOfficeReadDeviceTime
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPOfficeReadDeviceTime:     [weakSelf         WBOBLEManagerResponseReadDeviceTime:model];     break; </pre>

## 11.10. Write device Time to BPM:

	- (void)writeDeviceTime;
Definition	Write device Time to BPM: After the device is successfully connected, the time will be automatically synchronized.
CMD	CMDWatchBPOfficeWriteDeviceTime
model	MicroLifeDataModel

	<pre> case CMDWatchBPOfficeWriteDeviceTime: {     MicroLifeDataModel *dataModel = model;     [weakSelf         WBOBLEManagerResponseWriteDeviceTime:dat         aModel.success.boolValue];     }     break; </pre>
--	--

#### 11.11. Read BPM function setting value from BPM:

	- (void)readFunctionSettingValue;
Definition	Read BPM function setting value from BPM
CMD	CMDWatchBPOfficeReadFunctionSettingValue
model	MicroLifeWatchBPFunctionSettingValues
	<pre> case CMDWatchBPOfficeReadFunctionSettingValue:     [weakSelf         WBOBLEManagerResponseReadFunctionSetting         Value:model];     break; </pre>

#### 11.12. Read ABPM setting values from BPM:

	- (void)readSettingValues;
Definition	Read ABPM setting values from BPM
CMD	CMDWatchBPOfficeReadSettingValues
model	MicroLifeWatchBPSettingValues
	<pre> case CMDWatchBPOfficeReadSettingValues:     [weakSelf         WBOBLEManagerResponseReadSettingValue:mo         dell];     break; </pre>

#### 11.13. Write ABPM setting values to BPM:

	<pre> - (void)writeSettingValuesWithSelectedAUS_HI_infPressure:(HlinfPressure)AUS_HI_infPressure HI_infPressure:(HlinfPressure)HI_infPressure SW_AUTO_hide:(BOOL)SW_AUTO_hide SW_SEL_silent:(BOOL)SW_SEL_silent SW_AUS_Hide:(BOOL)SW_AUS_Hide SW_AVG_no_include_first:(BOOL)SW_AVG_no_include_first SW_CBP:(BOOL)SW_CBP SW_AFib:(BOOL)SW_AFib SW_AMPM:(BOOL)SW_AMPM SW_Kpa:(BOOL)SW_Kpa RestTime:(NSInteger)RestTime IntervalTime:(NSInteger)IntervalTime AutoMeasureNumber:(NSInteger)AutoMeasureNumber; </pre>
Definition	Write ABPM setting values to BPM.
Parameter	AUS_HI_infPressure Highest inflation pressure of AUS mode.

	<p>HI_infPressure Highest inflation pressure of Auto mode.</p> <p>SW_AUTO_hide Set Show readings during rest time in auto mode. true:hide/false:show.</p> <p>SW_SEL_silent Beeper true:enabled/false:disabled</p> <p>SW_AUS_Hide Set Show cuff pressure during deflation in AUS mode. true:hide/false:show.</p> <p>SW_AVG_no_include_first Set Average is include first memory data. true:is/false:is not.</p> <p>SW_CBP Set CBP measurement true:enabled/false:disabled.</p> <p>SW_AFib Set AFib measurement true:enabled/false:disabled.</p> <p>SW_AMPM Set 12/24-hour clock true:12-hour/false:24-hour.</p> <p>SW_Kpa Set Pressure unit: true:Kpa/false:mmHg.</p> <p>RestTime Rest time of auto mode. Start countdown base on rest time before 1st measurement in auto mode.</p> <p>IntervalTime Interval time of auto mode. Start countdown base on interval time before 2nd~6th measurement in auto mode.</p> <p>AutoMeasureNumber It's number of measurements in auto mode.</p>
CMD	CMDWatchBPOfficeReadSettingValues
model	MicroLifeDataModel
	<pre>[self.watchBPOffice writeSettingValuesWithSelectedAUS_HI_infPressure:self.editAUS_HI_infPressure.text.integerValue HI_infPressure:self.editHI_infPressure.text.integerValue SW_AUTO_hide:self.SW_AUTO_hide.on SW_SEL_silent:self.SW_SEL_silent.on SW_AUS_Hide:self.SW_AUS_Hide.on SW_AVG_no_include_first:self.SW_AVG_no_include_first.on SW_CBP:self.SW_CBP.on SW_AFib:self.SW_AFib.on SW_AMPM:self.SW_AMPM.on SW_Kpa:self.SW_Kpa.on RestTime:self.editRestTime.text.integerValue IntervalTime:self.editIntervalTime.text.integerValue AutoMeasureNumber:self.editAutoMeasureNumber.text.integerValue];</pre>
	<pre>case CMDWatchBPOfficeWriteSettingValues: { MicroLifeDataModel *dataModel = model; [weakSelf WBOBLEManagerResponseWriteSettingValues: dataModel.success.boolValue]; }  break;</pre>



## 11.14. Read BT module name from BPM:

	- (void)readBTModuleName;
Definition	Read BT module name from BPM
CMD	CMDWatchBPOfficeReadBTModuleName
model	MicroLifeDeviceInfo
	<pre> case CMDWatchBPOfficeReadBTModuleName: {     MicroLifeDeviceInfo *deviceInfo = model;     [weakSelf         WBOBLEManagerResponseReadBTModuleName:deviceInfo.BTModuleName]; } break; </pre>

## 11.15. Start remote measurement:

	- (void)startRemoteMeasurementWhtihCBPFunction:(Dformat)dformat;
Definition	Start remote measurement
Parameter	dformat: Data format which BPM sent out. It is same as what APP requested. NoCBPRaw: No CBP raw data LowCBPRaw: low resolution CBP data (sampling rate =16Hz) FullCBPRaw: full CBP raw data (sampling rate=256Hz)
CMD	CMDWatchBPOfficeStartRemoteMeasurement CMDWatchBPOfficeSendRemoteMeasurementStatusEvery5seconds CMDWatchBPOfficeSendMeasurementResultsForEachMeasurement
model	Measurement period: MicroLifeWatchBPRemoteMeasurement Measurement results: MicroLifeWatchBPCurrentAndMData
	<pre> case CMDWatchBPOfficeStartRemoteMeasurement: case CMDWatchBPOfficeStopRemoteMeasurement: {     MicroLifeWatchBPRemoteMeasurement *rmModel = model;     [weakSelf         WBOBLEManagerResponseStartRemoteMeasurement:rmModel.dformat]; } break; </pre>

	<pre> <b>case</b>     CMDWatchBPOfficeSendRemoteMeasurementStatusE     very5seconds: {         MicroLifeWatchBPRemoteMeasurement *rmModel             = model;         [weakSelf             WBOBLEManagerResponderremoteMeasurementSt             atusEvery5secondsWithSTATUS:rmModel             .status             MeasurementNumber:rmModel             .measurementNumber.intValue             TotalMeasurementNumber:rmModel             .totalMeasurementNumber.intValue             Countdown:rmModel.countdown.intValue             TotalMeasuretime:rmModel             .totalMeasuretime.intValue];         }         <b>break;</b> </pre>
	<pre> <b>case</b>     CMDWatchBPOfficeSendMeasurementResultsForEac     hMeasurement: {         MicroLifeWatchBPRemoteMeasurement *rmModel             = [model valueForKey:@"RMInfo"];         MicroLifeWatchBPCurrentAndMData *mData             = [model                 valueForKey:@"Measurement"];         [weakSelf             WBOBLEManagerResponseMeasurementResultsF             orEachMeasurement:mData             HistoryMeasurementNumber:rmModel             .historyMeasuremeNumber.intValue             CurrentMeasurementTimes:rmModel             .currentMeasurementTimes.intValue             AverageCalculationWhenMeasurement:rmMode             l.isAverage.boolValue];         }         <b>break;</b> </pre>

## 11.16. Stop remote measurement:

	- (void)stopRemoteMeasurement;
Definition	Stop remote measurement
CMD	CMDWatchBPOfficeStopRemoteMeasurement
model	MicroLifeWatchBPRemoteMeasurement
	<pre>case CMDWatchBPOfficeStartRemoteMeasurement: case CMDWatchBPOfficeStopRemoteMeasurement: {     MicroLifeWatchBPRemoteMeasurement *rmModel         = model;     [weakSelf         WBOBLEManagerResponseStartRemoteMeasurem         ent:rmModel.dformat];     }     break;</pre>

## 第12章 Peak flowInterface Description

## 12.1. Set the monitoring response status and obtain the corresponding data according to CMD

```

[self.peakFlowMeter getReadDataModelBlock:^(NSInteger CMD, id _Nonnull model, NSError * _Nonnull
error) {
    [weakSelf log:weakSelf.peakFlowMeter CMD:CMD Model:model Error:error];
    switch (CMD) {
        case CMDPFMReadHistorys:
        case CMDPFMClearAllHistorys:
        case CMDPFMReadUserAndVersionData:
        case CMDPFMWriteUser:
        case CMDPFMReadLastData:
        case CMDPFMClearLastData:
        case CMDPFMReadDeviceTime:
        case CMDPFMWriteDeviceTime:
        case CMDPFMReadSerialNumber:
        case CMDPFMReadBestValue:
        case CMDPFMWriteBestValue:
        case CMDPFMCheckMode:
        case CMDPFMWaveformModeStartMeasurement:
        case CMDPFMWaveformModeReadWaveform:
            break;
        default:
            break;
    }
}];

```

## 12.2. Read all history or current data:

	- (void)readAllHistorys;
Definition	Read all history or current data
CMD	CMDPFMReadHistorys
model	Measurement data available: MicroLifePFMDRecord No measurement data: MicroLifeDataModel

## 12.3. Clear all history data:

	- (void)clearAllHistorys;
Definition	clear all history data
CMD	CMDPFMClearAllHistorys
model	MicroLifeDataModel

## 12.4. Disconnect the Bluetooth with PFM:

	- (void)disconnect;
Definition	Disconnect the Bluetooth with PFM
Parameter	void(^connectDeviceStateBlock) (id device, CBCentralManager *central, CBPeripheral *peripheral, CBPeripheralState state, NSError * error)
Parameter	void(^cancelAllDevicesConnectionBlock) (void)

	<pre>[self.sdk getConnectDeviceStateBlock:^(id _Nonnull device,     CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull     peripheral, CBPeripheralState state, NSError * _Nonnull error) {     [weakself connectDeviceInfo:device PeripheralState:state]; } CancelAllDevicesConnectionBlock:^(     [weakself addLogWhitText:@"Cancel All Devices Connection"]; }];</pre>
--	---

#### 12.5. Read user ID and version data from PFM:

	- (void)readUserAndVersionData;
Definition	Read user ID and version data from PFM: After the device is successfully connected, the user ID and version data will be automatically read.
CMD	CMDPFMReadUserAndVersionData
model	Dictionary:{ UserInfo = "<MicroLifeUserInfo: User Information>"; Version = "<MicroLifeDeviceInfo: Device Information>"; }

#### 12.6. Write a new user ID to PFM:

	- (void)writeUserID:(NSString *)ID Age:(NSInteger)age;
Definition	Write a new user ID & age to PFM
Parameter	ID: User ID。 age: age
CMD	CMDPFMWriteUser
model	MicroLifeDataModel

#### 12.7. Read device Time from PFM:

	- (void)readDeviceTime;
Definition	Read device Time from PFM
CMD	CMDPFMReadDeviceTime
model	MicroLifeDeviceInfo

#### 12.8. Write device Time to PFM:

	- (void)writeDeviceTime;
Definition	Write device Time to PFM: After the device is successfully connected, the time will be automatically synchronized.
CMD	CMDPFMWriteDeviceTime
model	MicroLifeDataModel

## 12.9. Read device SN from PFM :

	- (void)readSerialNumber;
Definition	Read device SN from PFM
CMD	CMDPFMReadSerialNumber
model	MicroLifeDeviceInfo

## 12.10. Read Best Value :

	- (void)readBestValue;
Definition	Read Best Value
CMD	CMDPFMReadBestValue
model	MicroLifeDeviceInfo
	beatValue : Peakflow Meter Best Value highValue : Peakflow Meter High Value

## 12.11. Write Best Value :

	- (void)writeBestValue:(NSInteger)bv;
Definition	Write Best Value
Parameter	bv: Best Value
CMD	CMDPFMWriteBestValue
model	MicroLifeDataModel

## 12.12. Check Mode :

	- (void)checkMode;
Definition	Check Mode
CMD	CMDPFMCheckMode
model	MicroLifeDeviceInfo
	waveformMode : 0 : Normal marriage; 1 : Waveform mode haveNewData : 0 : Have net new data; 1 : Have new data

## 12.13. Waveform Mode Start Measurement

	- (void)waveformModeStartMeasurement;
Definition	Waveform Mode Start Measurement
CMD	CMDPFMWaveformModeStartMeasurement
model	MicroLifeDataModel

## 第13章 Usage process:

## 13.1. Independent management mode (multiple devices connected at the same time):

## 13.1.1. Singleton Bailue device:

```
// Device manager
self.watchBP03 = [MicroLifeWatchBP03
    shareWhithAuthorizationkey:SDKkey_WBP];
```

- 13.1.2. Monitor the mobile phone's Bluetooth status, device scanning and connection status: When the mobile phone's Bluetooth is turned on, it will automatically scan the surrounding devices. If a matching device is found, it will automatically connect to the device by default. **After the device is connected, if no command is sent for more than 120 seconds, it will automatically disconnect.**

```
[self.watchBP03 getDidUpdateStateBlock:^(CBManagerState state) {
} ScanDeviceBlock:^(id _Nonnull device) {
} CancelScanBlock:^(
} ConnectDeviceStateBlock:^(id _Nonnull device, CBCentralManager
    * _Nonnull central, CBPeripheral * _Nonnull peripheral,
    CBPeripheralState state, NSError * _Nonnull error) {
} CancelAllDevicesConnectionBlock:^(
}];
```

- 13.1.3. Monitor command response status: Send commands based on usage needs and obtain corresponding information through CMD.

```
// Device response
[self.watchBP03 getReadDataModelBlock:^(NSInteger CMD, id
    _Nonnull model, NSError * _Nonnull error) {
}];
```

- 13.1.4. Monitor error response status: The connection will be automatically disconnected upon receiving the response.

```
[self.watchBP03 getValidationErrorBlock:^(NSString * _Nonnull
    item, NSString * _Nonnull info, NSData * _Nonnull value) {
}];
```

## 13.2. Unified management mode (single device connection):

## 13.2.1. Singleton Bluetooth Manager:

```
// Bluetooth scanner establishment
self.sdk = [BLESDK shareOne];
```

## 13.2.2. Singleton Bailue device:

```
// Device manager
self.watchBP03 = [MicroLifeWatchBP03
    shareWhithAuthorizationkey:SDKkey_WBP];
```

## 13.2.3. Monitor the Bluetooth status of mobile phone:

```
// Bluetooth status of mobile phone
[self.sdk getDidUpdateStateBlock:^(CBManagerState state) {
    NSString *log = [NSString
        stringWithFormat:@"DidUpdateState :%ld", (long)state];
    [weakSelf addLogWhitText:log];
}];
```

## 13.2.4. Monitoring device scanning status:

```
[self.sdk getScanDeviceBlock:^(id _Nonnull device) {
    NSString *log = [NSString stringWithFormat:@"ScanDevice
        Name :%@ UUID :%@ mac :%@", [device name], [device
        UUID], [device mac]];
    [weakSelf addLogWhitText:log];
} CancelScanBlock:^() {
    [weakSelf addLogWhitText:@"Cancel Scan"];
}];
```

## 13.2.5. Monitor device connection status:

```
[self.sdk getConnectDeviceStateBlock:^(id _Nonnull device,
    CBCentralManager * _Nonnull central, CBPeripheral * _Nonnull
    peripheral, CBPeripheralState state, NSError * _Nonnull error)
{
    [weakSelf connectDeviceInfo:device PeripheralState:state];
} CancelAllDevicesConnectionBlock:^(
    [weakSelf addLogWhitText:@"Cancel All Devices Connection"];
}];
```

13.2.6. Add Bailue devices: When the Bluetooth of your phone is turned on, it will automatically scan nearby devices. If a matching device is found, it will automatically connect to the device by default. **After the device is connected, if no command**



is sent for more than 120 seconds, it will automatically disconnect.

```
// Bluetooth scanner join device
[self.sdk device:@[self.watchBP03]];
```

- 13.2.7. Monitor command response status: Send commands based on usage needs and obtain corresponding information through CMD.

```
// Device response
[self.watchBP03 getReadDataModelBlock:^(NSInteger CMD, id
    _Nonnull model, NSError * _Nonnull error) {

}];
```

- 13.2.8. Monitor error response status: The connection will be automatically disconnected upon receiving the response.

```
[self.watchBP03 getValidationErrorBlock:^(NSString * _Nonnull
    item, NSString * _Nonnull info, NSData * _Nonnull value) {

}];
```

### 13.3. Device automatic scanning mechanism:

- 13.3.1. When CBManagerStatePoweredOn automatically opens scanning [Default: Open]
- 13.3.2. In unified management mode (single device connection), use 3.5 to disable the automatic scanning mechanism.

```
// stop Auto Scan
[self.sdk autoScan:NO];
```

- 13.3.3. Use independent management mode (multiple devices connected simultaneously) and use 4.3 to disable the automatic scanning mechanism.

```
// stop Auto Scan
[self.watchBP03 autoScan:NO];
```

### 13.4. Device automatic connection mechanism:

- 13.4.1. When scanning for devices, if a matching device is found by default, the device will be automatically connected.
- 13.4.2. Use 4.4. Auto Connect to cancel the automatic connection mechanism.

```
// Auto Connect [Default:Open]
[self.watchBP03 setAutoConnect:NO];
```

- 13.4.3. Monitors the status of the scanned device and connects based on the response from void(^scanDeviceBlock) (id device).

```
// Scanning device response
[self.sdk getScanDeviceBlock:^(id _Nonnull device) {
    NSString *log = [NSString stringWithFormat:@"ScanDevice
        Name : %@ UUID : %@ mac : %@", [device name], [device
        UUID], [device mac]];
    [weakself addLogWhitText:log];
    [weakself.watchBP03 connectDevice];
} CancelScanBlock:^() {
    [weakself addLogWhitText:@"Cancel Scan"];
}];
```

## 第14章 V1.x migration instructions for V2.x:

- 14.1. Select Bluetooth unified management mode (single device connection) or independent management mode (multiple devices connected simultaneously) based on project requirements. Refer to Chapter 6.
- 14.2. Removed the original Bluetooth selector from V1.x version.

```
// Do any additional setup after loading the view.
self.aWBO3BLEManager = [WBO3BLEManager
    sharedInstanceWhithAuthorizationKey:SDKkey_WBP];
self.aWBO3BLEManager.dataResponseDelegate = self;

self.wbo3BindingDevice = [self.aWBO3BLEManager getBindingDevice];
```

- 14.3. -  
(void)WBO3BLEManagerCellPhoneBluetoothDidUpdateState:(MicroLifeBLEState)state , change MicroLifeBLEState to CBManagerState , and continue in getDidUpdateStateBlock:^(CBManagerState state) .
- 14.4. -  
(void)WBO3BLEManagerDidDiscoverBluetoothDeviceMacAddress:(nonnull NSData \*)macAddress Name:(nonnull NSString \*)name RSSI:(nonnull NSNumber \*)RSSI , 已被ScanDeviceBlock:^(id \_Nonnull device) 取代。
- 14.5. - (void)WBO3BLEManagerDidConnectDevice、  
- (void)WBO3BLEManagerDidDisconnectDevice、  
- (void)WBO3BLEManagerDidFailToConnectDevice 已被  
ConnectDeviceStateBlock:^(id \_Nonnull device, CBCentralManager \* \_Nonnull central, CBPeripheral \* \_Nonnull peripheral, CBPeripheralState state, NSError \* \_Nonnull error) 取代。

```

// Device
[self.watchBP03 getDidUpdateStateBlock:^(CBManagerState state) {
    [weakSelf
        WBO3BLEManagerCellPhoneBluetoothDidUpdateState:state];
} ScanDeviceBlock:^(id _Nonnull device) {
    // -
    (void)WBO3BLEManagerDidDiscoverBluetoothDeviceMacAddress:
        (nonnull NSData *)macAddress Name:(nonnull NSString
        *)name RSSI:(nonnull NSNumber *)RSSI
} CancelScanBlock:^(
} ConnectDeviceStateBlock:^(id _Nonnull device, CBCentralManager
    * _Nonnull central, CBPeripheral * _Nonnull peripheral,
    CBPeripheralState state, NSError * _Nonnull error) {
    // - (void)WBO3BLEManagerDidConnectDevice
    // - (void)WBO3BLEManagerDidDisconnectDevice
    // - (void)WBO3BLEManagerDidFailToConnectDevice
} CancelAllDevicesConnectionBlock:^(
    // - (void)WBO3BLEManagerDidDisconnectDevice
    // - (void)WBO3BLEManagerDidFailToConnectDevice
}];

```

- 14.6. - (void)WBO3BLEManagerRespondTimeOut 已被  
 getValidationErrorBlock:^(NSString \* \_Nonnull item, NSString \*  
 \_Nonnull info, NSData \* \_Nonnull value) 取代。

```

[self.watchBP03 getValidationErrorBlock:^(NSString * _Nonnull
    item, NSString * _Nonnull info, NSData * _Nonnull value) {
    // - (void)WBO3BLEManagerRespondTimeOut
}];

```

- 14.7. WBO3DataResponseDelegate has been called  
 getReadDataModelBlock:^(NSInteger CMD, id \_Nonnull model,  
 NSError \* \_Nonnull error), and the corresponding response data is  
 obtained based on CMD.

```
[self.watchBPO3 getReadDataModelBlock:^(NSInteger CMD, id
    _Nonnull model, NSError * _Nonnull error) {
    switch (CMD) {
        case CMDWatchBPO3ReadHistorys:
        case CMDWatchBPO3ReadCBPData:
        case CMDWatchBPO3ClearAllHistorys:
        case CMDWatchBPO3ReadUserAndVersionData:
        case CMDWatchBPO3WriteUser:
        case CMDWatchBPO3ReadDeviceInfo:
        case CMDWatchBPO3ReadDeviceTime:
        case CMDWatchBPO3WriteDeviceTime:
        case CMDWatchBPO3ReadSerialNumber:
        case CMDWatchBPO3ReadFunctionSettingValue:
        case CMDWatchBPO3ReadBTModuleName:
        case CMDWatchBPO3ReadSettingValues:
        case CMDWatchBPO35SReadSettingValues:
        case CMDWatchBPO3WriteSettingValues:
        case CMDWatchBPO35SWriteSettingValues:
            break;
        default:
            break;
    }
}
```

- 14.8. - (void)WBO3BLEManagerResponseReadAllHistorys:(nonnull MicroLifeDRecord \*)data, MicroLifeDRecord 以 MicroLifeWatchBPDRecord取代, MicroLifeCurrentAndMData以 MicroLifeWatchBPCurrentAndMData取代, CMD等於 CMDWatchBPO3ReadHistorys。

```
case CMDWatchBPO3ReadHistorys:
    if ([model isKindOfClass:[MicroLifeWatchBPDRecord
        class]]) {
        [weakSelf
            WBO3BLEManagerResponseReadAllHistorys:model];
    } else {
        // reply Null ACK
    }
    break;
```

- 14.9. - (void)WBO3BLEManagerResponseReadCBPData:(nonnull MicroLifeCBPdataAndCalCBP \*)data IsNoData:(BOOL)isNoData, MicroLifeCBPdataAndCalCBP以 MicroLifeWatchBPCBPdataAndCalCBP取代, CMD等於 CMDWatchBPO3ReadCBPData。

```

case CMDWatchBPO3ReadCBPData:
    if ([model
        isKindOfClass:[MicroLifeWatchBPCBPdataAndCalCBP
            class]]) {
        [weakSelf WB03BLEManagerResponseReadCBPData:model
            IsNoData:NO];
    } else {
        [weakSelf
            WB03BLEManagerResponseReadCBPData:nil
            IsNoData:YES];
    }
    break;

```

- 14.10. - (void)WB03BLEManagerResponseReadUserAndVersionData:(nonnull MicroLifeUserInfo \*)user VersionData:(nonnull MicroLifeDeviceInfo \*)verData, CMD等於CMDWatchBPO3ReadUserAndVersionData。

```

case CMDWatchBPO3ReadUserAndVersionData: {
    MicroLifeUserInfo *userInfo = [model
        valueForKey:@"UserInfo"];
    MicroLifeDeviceInfo *version = [model
        valueForKey:@"Version"];
    [weakSelf
        WB03BLEManagerResponseReadUserAndVersionData:userInfo
        VersionData:version];
}
break;

```

- 14.11. - (void)WB03BLEManagerResponseClearHistory:(BOOL)isSuccess, CMD等於CMDWatchBPO3ClearAllHistorys。

```

case CMDWatchBPO3ClearAllHistorys: {
    MicroLifeDataModel *dataModel = model;
    [weakSelf
        WB03BLEManagerResponseClearHistory:dataModel
        .success.boolValue];
}
break;

```

- 14.12. - (void)WB03BLEManagerResponseReadBTModuleName:(nonnull NSString \*)BTModuleName, CMD等於CMDWatchBPO3ReadBTModuleName。

```

case CMDWatchBP03ReadBTModuleName: {
    MicroLifeDeviceInfo *deviceInfo = model;
    [weakSelf
        WBO3BLEManagerResponseReadBTModuleName:deviceInfo
        .BTModuleName];
}
break;

```

- 14.13. -  
 (void)WBO3BLEManagerResponseReadFunctionSettingValue:(nonnull  
 | MicroLifeFunctionSettingValues \*)functionSettingValues,  
 MicroLifeFunctionSettingValues以  
 MicroLifeWatchBPFunctionSettingValues取代, CMD等於  
 CMDWatchBP03ReadFunctionSettingValue。

```

case CMDWatchBP03ReadFunctionSettingValue:
    [weakSelf
        WBO3BLEManagerResponseReadFunctionSettingValue:model];
break;

```

- 14.14. - (void)WBO3BLEManagerResponseReadDeviceTime:(nonnull  
 MicroLifeDeviceInfo \*)deviceInfo, CMD等於  
 CMDWatchBP03ReadDeviceTime。

```

case CMDWatchBP03ReadDeviceTime:
    [weakSelf WBO3BLEManagerResponseReadDeviceTime:model];
break;

```

- 14.15. - (void)WBO3BLEManagerResponseReadSettingValue:(nonnull  
 MicroLifeSettingValues \*)settingValues,  
 MicroLifeFunctionSettingValues以MicroLifeWatchBPSettingValues取  
 代, CMD等CMDWatchBP03ReadSettingValues、  
 CMDWatchBP035SReadSettingValues。

```

case CMDWatchBP03ReadSettingValues:
case CMDWatchBP035SReadSettingValues:
    [weakSelf
        WBO3BLEManagerResponseReadSettingValue:model];
break;

```

- 14.16. -  
 (void)WBO3BLEManagerResponseWriteDeviceTime:(BOOL)isSuccess,  
 CMD等於CMDWatchBP03WriteDeviceTime。

```

    case CMDWatchBP03WriteDeviceTime: {
        MicroLifeDataModel *dataModel = model;
        [weakSelf
            WBO3BLEManagerResponseWriteDeviceTime:dataModel
            .success.boolValue];
    }
    break;

```

- 14.17. - (void)WBO3BLEManagerResponseWriteSettingValues:(BOOL)isSuccess, CMD等於CMDWatchBP03WriteSettingValues、CMDWatchBP035WriteSettingValues。

```

    case CMDWatchBP03WriteSettingValues:
    case CMDWatchBP035WriteSettingValues: {
        MicroLifeDataModel *dataModel = model;
        [weakSelf
            WBO3BLEManagerResponseWriteSettingValues:dataModel
            .success.boolValue];
    }
    break;

```

- 14.18. - (void)WBO3BLEManagerResponseWriteUserID:(BOOL)isSuccess, CMD等於CMDWatchBP03WriteUser。

```

    case CMDWatchBP03WriteUser: {
        MicroLifeDataModel *dataModel = model;
        [weakSelf
            WBO3BLEManagerResponseWriteUserID:dataModel
            .success.boolValue];
    }
    break;

```

- 14.19. - (void)WBO3BLEManagerResponseReadDeviceIDAndInfo:(nonnull MicroLifeDeviceInfo \*)deviceInfo, CMD等於CMDWatchBP03ReadDeviceInfo。

```

    case CMDWatchBP03ReadDeviceInfo:
        [weakSelf
            WBO3BLEManagerResponseReadDeviceIDAndInfo:model];
    break;

```

- 14.20. Read Serial Number, CMD等於CMDWatchBP03ReadSerialNumber。

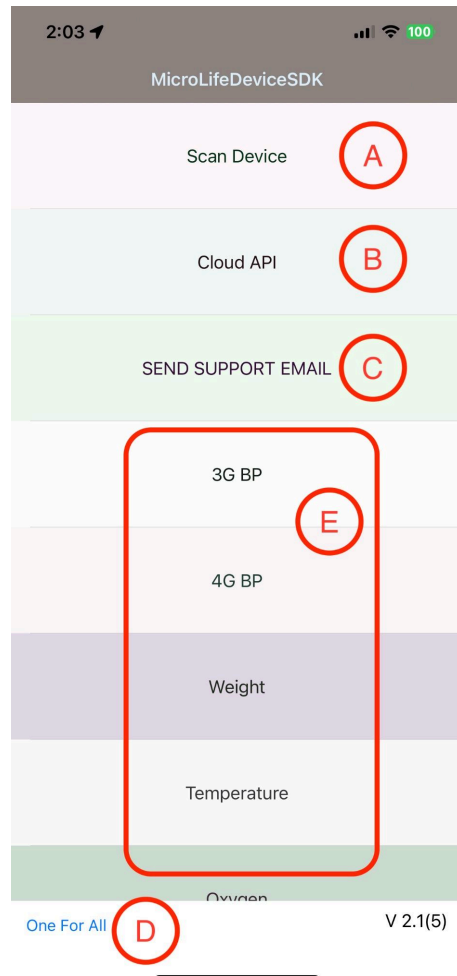


```
case CMDWatchBP03ReadSerialNumber: {  
    MicroLifeDeviceInfo *deviceInfo = model;  
    NSLog(@"Serial Number:%@",deviceInfo.sn);  
}  
break;
```

## 第15章 Dome Code Description:

### 15.1. Dome Code Tour Page:

- 15.1.1. A: Scan Device, Bluetooth SDK function display.
- 15.1.2. B: Cloud API, cloud API SDK function display.
- 15.1.3. C: Send local log to MicroLife.
- 15.1.4. D: One For All, multiple devices are automatically connected and displayed, using independent management mode (multiple devices are connected at the same time).
- 15.1.5. E: Demonstration of SDK functions of each Bailue product.



## 15.2. Temperature SDK interface description:

15.2.1. A: Log display area.

15.2.2. B: Function instruction area.

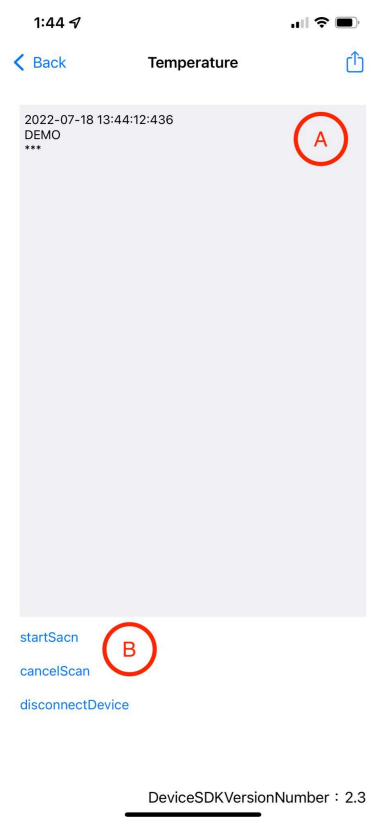
15.2.3. For related code, see TEMPViewController.

15.2.4. Instructions for use:

15.2.4.1. When you enter the display page, the device search will be automatically performed.

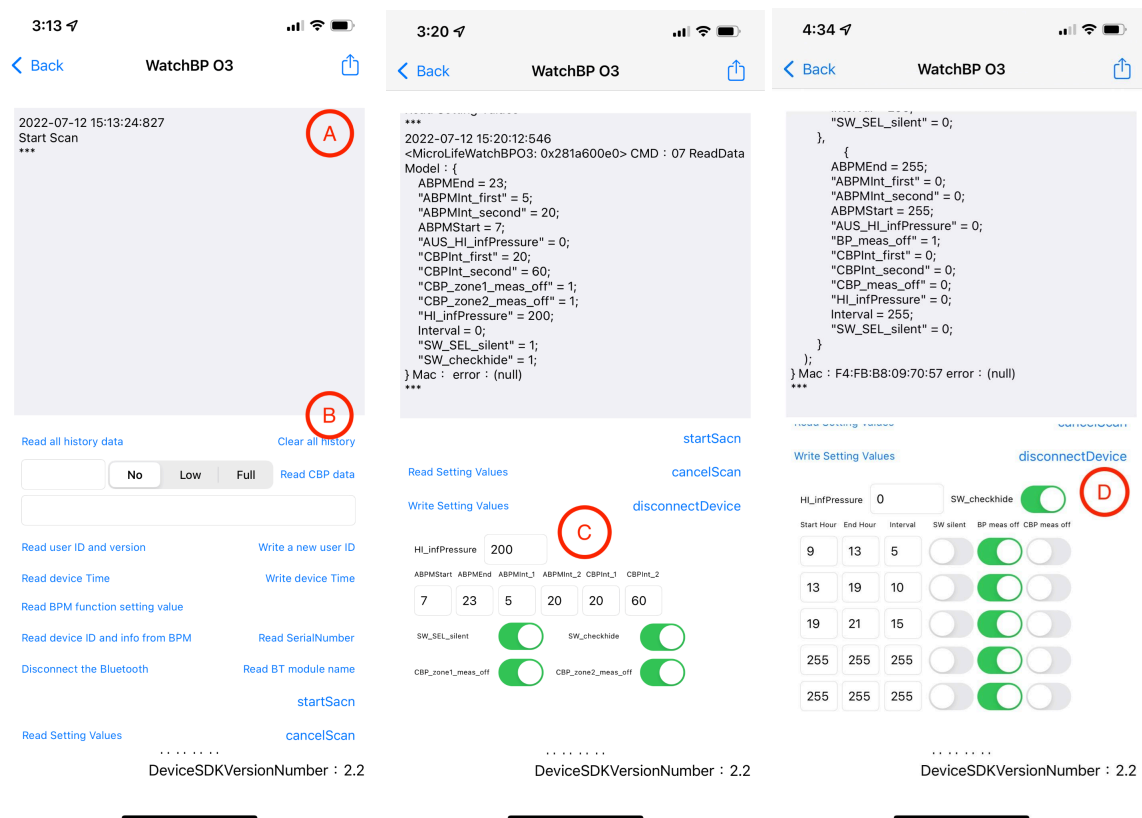
15.2.4.2. If a matching device is found, it will automatically connect to the device.

15.2.4.3. After the device is connected, if no response is received for more than 120 seconds, it will automatically disconnect.



## 15.3. WatchBP O3 II SDK interface description:

- 15.3.1. A: Log display area.
- 15.3.2. B: Function command area, can be slid up and down.
- 15.3.3. C: Read Setting Values (2 schedule) parameter setting interface.
- 15.3.4. D: Read Setting Values (5 schedule) parameter setting interface.
- 15.3.5. For related code, refer to WBO3ViewController.
- 15.3.6. Instructions for use:
  - 15.3.6.1. When you enter the display page, the device search will be automatically performed.
  - 15.3.6.2. If a matching device is found, it will automatically connect to the device.
  - 15.3.6.3. After the device is connected, the SDK will automatically send "Write device Time to BPM" and "Read user ID and version data from BPM".
  - 15.3.6.4. After the device is connected, select a function operation in the device function command area.
  - 15.3.6.5. After the device is connected, if no command is sent for more than 120 seconds, it will automatically disconnect.
  - 15.3.6.6. After using "Read Setting Values", the parameter setting interface will be displayed according to WatchBP O3 II (2 schedule / 5 schedule).





## 第16章    Operating Instructions

## 16.1.    Search &amp; Connect:

3:21

< Back    WatchBP O3    >

2022-07-12 15:21:40:662  
Start Scan  
\*\*\*  
2022-07-12 15:21:48:705  
ScanDevice Name : Watch BP O3 UUID : 15E26CD2-FBFB-0EE9-2C03-3779104A68AF mac : {length = 6, bytes = 0xf4fbb8097057}  
\*\*\*  
2022-07-12 15:21:49:059  
connect Device : <MicroLifeWatchBPO3: 0x281a380e0>  
PeripheralState : Connected  
\*\*\*  
2022-07-12 15:21:51:307  
<MicroLifeWatchBPO3: 0x281a380e0> CMD : 0D  
ReadData Model : {  
  CMD = 13;  
  Mac = "F4:FB:B8:09:70:57";  
  dataValue = "";  
  deviceType = 7;  
  indexYear = 0;

Read all history data    Clear all history

No

Low

Full

Read CBP data

new ID test 01Ã

Read user ID and version    Write a new user ID

Read device Time    Write device Time

Read BPM function setting value

Read device ID and info from BPM    Read SerialNumber

Disconnect the Bluetooth    Read BT module name

startSacn


Read Setting Values    cancelScan

.....

DeviceSDKVersionNumber : 2.2

1. After entering the display page, the SDK will automatically start searching and connecting to a compatible device.
2. ScanDevice Name : Watch BP O3 UUID : 15E26CD2-FBFB-0EE9-2C03-3779104A68AF mac : {length = 6, bytes = 0xf4fbb8097057}
3. connect Device : <MicroLifeWatchBPO3: 0x2825c10a0> PeripheralState : Connected

## 16.2. pair:



The screenshot shows the WatchBP O3 app interface. At the top, the status bar displays the time 3:50 and battery level. The app title is 'WatchBP O3'. Below the title, there is a log of events:

```
2022-07-13 15:50:32:595
Start Scan
***
2022-07-13 15:50:38:188
ScanDevice Name : Watch BP O3 UUID : 15E26CD2-
FBFB-0EE9-2C03-3779104A68AF mac : (null)
***
2022-07-13 15:50:38:462
connect Device : <MicroLifeWatchBPO3: 0x281a98540>
PeripheralState : Connected
***
```

A Bluetooth pairing dialog is displayed in the center, titled '藍牙配對要求' (Bluetooth Pairing Requirement). The dialog text reads: '「Watch BP O3」想要與您的 iPhone 配對。請確認「Watch BP O3」上已顯示代碼「198620」。請勿在任何配件上輸入此密碼。' (The 'Watch BP O3' wants to pair with your iPhone. Please confirm that the code '198620' is displayed on the 'Watch BP O3'. Do not enter this password on any accessory.). The dialog has two buttons: '取消' (Cancel) and '配對' (Pair).

Below the dialog, there is a list of test commands:

- Read all history
- Write a new user ID
- Read device Time
- Write device Time
- Read BPM function setting value
- Read device ID and info from BPM
- Read SerialNumber
- Disconnect the Bluetooth
- Read BT module name
- startSacn
- Read Setting Values
- cancelScan

At the bottom, the text 'DeviceSDKVersionNumber : 2.2' is displayed.

1. If the device requires Bluetooth pairing, a Bluetooth pairing dialog window will pop up asking for pairing.
2. After pairing is complete, select the test command.

## 16.3. Test command (e.g. Write a new user ID):

1. Select the test command:  
Write a new user ID.
2. Write a new user ID:  
AB12345678. The user ID is an ASCII code.
3. <MicroLifeWatchBPO3: 0x281c782a0> CMD: 06 ReadData  
ReadData Model: {  
    CMD = 6;  
    Mac = "F4:FB:B8:09:70:57";  
    dataValue = "";  
    deviceType = 7;  
    indexYear = 0;  
    null = 0;  
    protocolVersion = 0;  
    success = 1;  
} Mac: F4:FB:B8:09:70:57 error: (null)  
: Write status response.
4. You can use "Read user ID and version data" to check the newly written user ID.



## 16.4. Test command (e.g. Read user ID and version data):

4:08

WatchBP 03

2022-07-13 16:07:43:919

<MicroLifeWatchBPO3: 0x2825c10a0> CMD: 05 ReadData Dictionary: {

UserInfo = "<MicroLifeUserInfo: 0x282ddd400>";

Version = "<MicroLifeDeviceInfo: 0x121620310>";

} error: (null)

\*\*\*

2022-07-13 16:07:43:927

<MicroLifeWatchBPO3: 0x2825c10a0> CMD: 05 ReadData Model: {

FWName = RE1;

batteryVoltage = 4;

day = 23;

maxMemory = 330;

month = 7;

optionAfib = 1;

optionCBP = 1;

optionIHB = 0;

optionPAD = 0;

protocolVersion = "-5117805771225310652";

unit = 0;

year = 2021;

} Mac: F4:FB:B8:09:70:57 error: (null)

\*\*\*

2022-07-13 16:07:43:934

<MicroLifeWatchBPO3: 0x2825c10a0> CMD: 05 ReadData Model: {

bpmUserID = "new ID test 01\U00c3";

gender = 0;

roleType = 0;

} Mac: F4:FB:B8:09:70:57 error: (null)

\*\*\*

Read all history data

Clear all history

No

Low

Full

Read CBP data

new ID test 01Ã

Read user ID and version

Write a new user ID

Read device Time

Write device Time

Read BPM function setting value

Read device ID and info from BPM

Read SerialNumber

Disconnect the Bluetooth

Read BT module name

startSacn

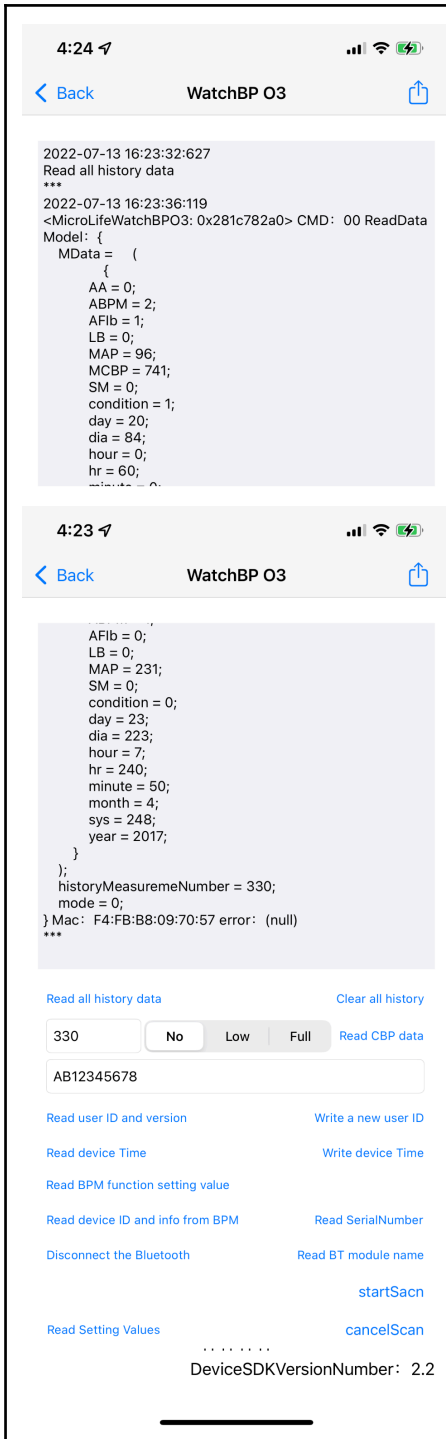
Read Setting Values

cancelScan

DeviceSDKVersionNumber: 2.2

1. Select the test command:  
Read user ID and version data.
2. Response:  
<MicroLifeWatchBPO3: 0x2825c10a0> CMD: 05 ReadData Dictionary: {  
    UserInfo =  
    "<MicroLifeUserInfo: 0x282ddd400>";  
    Version =  
    "<MicroLifeDeviceInfo: 0x121620310>";  
} error: (null)
3. MicroLifeUserInfo: User information
4. MicroLifeDeviceInfo: Device Information

## 16.5. Test command (e.g. Read all history data):



4:24 WatchBP O3

2022-07-13 16:23:32:627  
Read all history data  
\*\*\*

2022-07-13 16:23:36:119  
<MicroLifeWatchBPO3: 0x281c782a0> CMD: 00 ReadData  
Model: {  
  MData = (  
    {  
      AA = 0;  
      ABPM = 2;  
      AFib = 1;  
      LB = 0;  
      MAP = 96;  
      MCBP = 741;  
      SM = 0;  
      condition = 1;  
      day = 20;  
      dia = 84;  
      hour = 0;  
      hr = 60;  
      minute = 0;  
      month = 4;  
      sys = 120;  
      year = 2017;  
    }  
  );  
  historyMeasuremeNumber = 330;  
  mode = 0;  
} Mac: F4:FB:B8:09:70:57 error: (null)  
\*\*\*

Read all history data Clear all history

330 No Low Full Read CBP data

AB12345678

Read user ID and version Write a new user ID

Read device Time Write device Time

Read BPM function setting value

Read device ID and info from BPM Read SerialNumber

Disconnect the Bluetooth Read BT module name

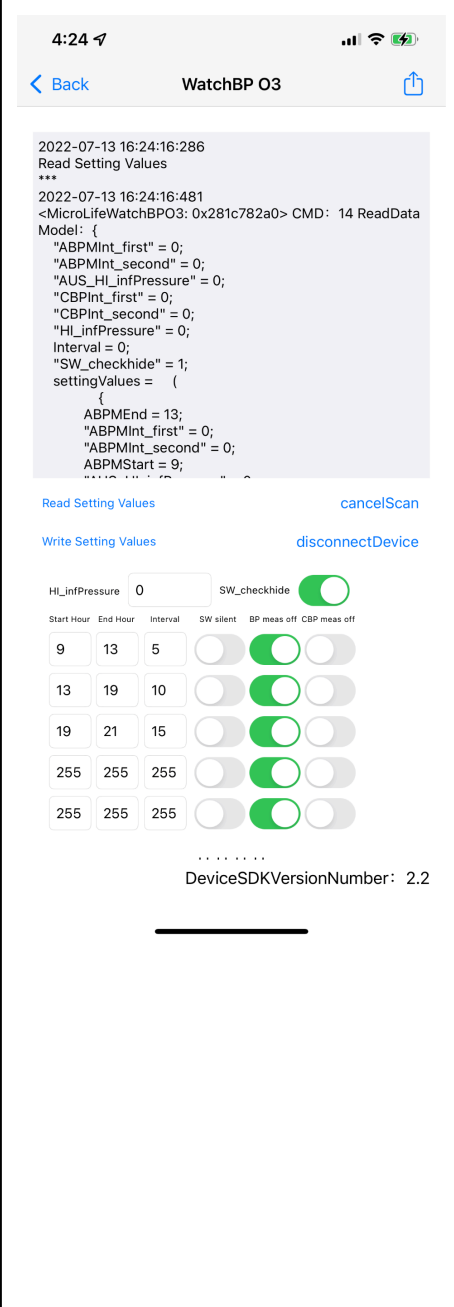
startSacn

Read Setting Values cancelScan

DeviceSDKVersionNumber: 2.2

1. Select the test command:  
Read all history data.
2. <MicroLifeWatchBPO3: 0x281c782a0> CMD: 00  
ReadData Model: {  
  MData = (  
    {  
      AA = 0;  
      ABPM = 2;  
      AFib = 1;  
      LB = 0;  
      MAP = 96;  
      MCBP = 741;  
      SM = 0;  
      condition = 1;  
      day = 20;  
      is = 84;  
      hour = 0;  
      hr = 60;  
      minute = 0;  
      month = 4;  
      sys = 120;  
      year = 2017;  
    },  
    {...},  
    ...,  
    {...}  
  );  
  historyMeasuremeNumber = 330;  
  mode = 0;  
}

## 16.6. Test command (e.g. Read Setting values):



The screenshot shows the WatchBP O3 app interface. At the top, the time is 4:24 and the device name is WatchBP O3. Below the title bar, there is a log of commands and their results. The log shows a 'Read Setting Values' command being executed, followed by a 'ReadData Model' response. The response is a JSON object containing various settings like 'ABPMInt\_first', 'ABPMInt\_second', 'AUS\_HI\_infPressure', 'CBPInt\_first', 'CBPInt\_second', 'HI\_infPressure', 'Interval', 'SW\_checkhide', 'ABPMEnd', 'ABPMInt\_first', 'ABPMInt\_second', and 'ABPMStart'. Below the log, there are buttons for 'Read Setting Values', 'cancelScan', 'Write Setting Values', and 'disconnectDevice'. At the bottom, there is a table for configuring settings, including 'HI\_infPressure', 'SW\_checkhide', 'Start Hour', 'End Hour', 'Interval', 'SW silent', 'BP meas off', and 'CBP meas off'. The 'DeviceSDKVersionNumber' is displayed as 2.2.

```

1. Select the test command:
Read Setting values.
2.<MicroLifeWatchBPO3:
0x281c782a0> CMD: 14
ReadData Model: {
    "ABPMInt_first" = 0;
    "ABPMInt_second" = 0;
"AUS_HI_infPressure" = 0;
    "CBPInt_first" = 0;
    "CBPInt_second" = 0;
    "HI_infPressure" = 0;
    Interval = 0;
    "SW_checkhide" = 1;
    settingValues = (
        {
            ABPMEnd = 13;
            "ABPMInt_first" = 0;
            "ABPMInt_second" = 0;
            ABPMStart = 9;
            "AUS_HI_infPressure" = 0;
            "BP_meas_off" = 1;
            "CBPInt_first" = 0;
            "CBPInt_second" = 0;
            "CBP_meas_off" = 0;
            "HI_infPressure" = 0;
            Interval = 5;
            "SW_SEL_silent" = 0;
        },
        {...},
        {...},
        {...},
        {...}
    );
} Mac: F4:FB:B8:09:70:57 error:
(null)

```

第17章    Update History:  
[iOS SDK Update History](#)