|  |
| --- |
|  |
| **P401M Android SDK Description** |
| **V1.0** |
|  |
| **Author:WangYong** |
| **2021/10/15** |

# Change log

|  |  |  |
| --- | --- | --- |
| **Date** | **Log** | **Author** |
| 2021-10-15 | Create | WangYong |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
|  |

# Catalog

[Change log 1](#_Toc85566933)

[Catalog 2](#_Toc85566934)

[Android SDK Intro 7](#_Toc85566935)

[1. Function and Purpose 7](#_Toc85566936)

[Integration 7](#_Toc85566937)

[1 .SDK Library 7](#_Toc85566938)

[2 .Integration 7](#_Toc85566939)

[API 9](#_Toc85566940)

[1.API initialization 9](#_Toc85566941)

[Description 9](#_Toc85566942)

[Parameters 9](#_Toc85566943)

[2. Set MTU 9](#_Toc85566944)

[Description 9](#_Toc85566945)

[Parameters 9](#_Toc85566946)

[3. Connnect Device 10](#_Toc85566947)

[Description 10](#_Toc85566948)

[Parameters 10](#_Toc85566949)

[4. Get Battery 10](#_Toc85566950)

[Description 10](#_Toc85566951)

[Parameters 11](#_Toc85566952)

[5. Get Device Version 11](#_Toc85566953)

[Description 11](#_Toc85566954)

[Parameters 11](#_Toc85566955)

[6. Query device collection status 11](#_Toc85566956)

[Description 11](#_Toc85566957)

[Parameters 11](#_Toc85566958)

[7. Stop Monitoring/Collecting 12](#_Toc85566959)

[Description 12](#_Toc85566960)

[Parameters 12](#_Toc85566961)

[8. Get Sleep Data (Real-time) 12](#_Toc85566962)

[Description 12](#_Toc85566963)

[Parameters 12](#_Toc85566964)

[9. Stop Getting Sleep Data 12](#_Toc85566965)

[Description 12](#_Toc85566966)

[Parameters 13](#_Toc85566967)

[10. Get The Signal Strength (Real-time) 13](#_Toc85566968)

[Description 13](#_Toc85566969)

[Parameters 13](#_Toc85566970)

[11. Stop Getting The Signal Strength 13](#_Toc85566971)

[Description 13](#_Toc85566972)

[Parameters 13](#_Toc85566973)

[12. Get Sleep Report 13](#_Toc85566974)

[Description 14](#_Toc85566975)

[Parameters 14](#_Toc85566976)

[13. Firmware Update 1 14](#_Toc85566977)

[Description 14](#_Toc85566978)

[Parameters 14](#_Toc85566979)

[14. Firmware Update 2 14](#_Toc85566980)

[Description 15](#_Toc85566981)

[Parameters 15](#_Toc85566982)

[15. Wait for device confirmation 15](#_Toc85566983)

[Description 15](#_Toc85566984)

[Parameters 15](#_Toc85566985)

[16. Monitor the status messages reported by the device 16](#_Toc85566986)

[Description 16](#_Toc85566987)

[Parameters 16](#_Toc85566988)

[17. Remove the status message listening reported by the device 16](#_Toc85566989)

[Description 16](#_Toc85566990)

[Parameters 16](#_Toc85566991)

[18. Bluetooth music configuration settings 16](#_Toc85566992)

[Description 17](#_Toc85566993)

[Parameters 17](#_Toc85566994)

[19. Bluetooth music configuration acquisition 17](#_Toc85566995)

[Description 17](#_Toc85566996)

[Parameters 17](#_Toc85566997)

[20. Operation related to sleep aid 17](#_Toc85566998)

[Description 17](#_Toc85566999)

[Parameters 18](#_Toc85567000)

[21. Sleep aid parameter acquisition 18](#_Toc85567001)

[Description 18](#_Toc85567002)

[Parameters 18](#_Toc85567003)

[22. Alarm clock acquisition 18](#_Toc85567004)

[Description 18](#_Toc85567005)

[Parameters 18](#_Toc85567006)

[23. Alarm clock setting 19](#_Toc85567007)

[Description 19](#_Toc85567008)

[Parameters 19](#_Toc85567009)

[24. Alarm clock Preview 20](#_Toc85567010)

[Description 20](#_Toc85567011)

[Parameters 20](#_Toc85567012)

[25. Off pillow alarm clock setting 20](#_Toc85567013)

[Description 20](#_Toc85567014)

[Parameters 20](#_Toc85567015)

[26. Off pillow alarm clock acquisition 21](#_Toc85567016)

[Description 21](#_Toc85567017)

[Parameters 21](#_Toc85567018)

[27. Query device working status 21](#_Toc85567019)

[Description 21](#_Toc85567020)

[Parameters 21](#_Toc85567021)

[28. Monitor the working status of registered device 21](#_Toc85567022)

[Description 22](#_Toc85567023)

[Parameters 22](#_Toc85567024)

[29. Remove device working status monitoring 22](#_Toc85567025)

[Description 22](#_Toc85567026)

[Parameters 22](#_Toc85567027)

[Object Description 22](#_Toc85567028)

[StatusCode 22](#_Toc85567029)

[Description 22](#_Toc85567030)

[Fields 23](#_Toc85567031)

[IResultCallback<T> 23](#_Toc85567032)

[Description 23](#_Toc85567033)

[Function 23](#_Toc85567034)

[CallbackData<T> 23](#_Toc85567035)

[Description 23](#_Toc85567036)

[Fields 23](#_Toc85567037)

[LoginBean 24](#_Toc85567038)

[Description 24](#_Toc85567039)

[Fields 24](#_Toc85567040)

[BatteryBean 24](#_Toc85567041)

[Description 24](#_Toc85567042)

[Fields 24](#_Toc85567043)

[CollectStatus 24](#_Toc85567044)

[Description 24](#_Toc85567045)

[Fields 24](#_Toc85567046)

[RealTimeData 25](#_Toc85567047)

[Description 25](#_Toc85567048)

[Fields 25](#_Toc85567049)

[OriginalData 25](#_Toc85567050)

[Description 25](#_Toc85567051)

[Fields 25](#_Toc85567052)

[HistoryData 26](#_Toc85567053)

[Description 26](#_Toc85567054)

[Fields 26](#_Toc85567055)

[Summary 26](#_Toc85567056)

[Description 26](#_Toc85567057)

[Fields 26](#_Toc85567058)

[Detail 27](#_Toc85567059)

[Description 27](#_Toc85567060)

[Fields 27](#_Toc85567061)

[Analysis 27](#_Toc85567062)

[Description 27](#_Toc85567063)

[Fields 27](#_Toc85567064)

[SleepStatusType 33](#_Toc85567065)

[Description 33](#_Toc85567066)

[Fields 33](#_Toc85567067)

[DeviceStateListener 33](#_Toc85567068)

[Interface introduction 33](#_Toc85567069)

[Method description 33](#_Toc85567070)

[DeviceState 33](#_Toc85567071)

[Description 33](#_Toc85567072)

[Fields 34](#_Toc85567073)

[MusicConfig 34](#_Toc85567074)

[Description 34](#_Toc85567075)

[Fields 34](#_Toc85567076)

[SleepAidConfig 35](#_Toc85567077)

[Description 35](#_Toc85567078)

[Fields 35](#_Toc85567079)

[AlarmInfo 35](#_Toc85567080)

[Description 35](#_Toc85567081)

[Fields 35](#_Toc85567082)

[OffPillowAlarmConfig 36](#_Toc85567083)

[Description 36](#_Toc85567084)

[Fields 36](#_Toc85567085)

[WorkStausListener 36](#_Toc85567086)

[Interface introduction 36](#_Toc85567087)

[Method description 36](#_Toc85567088)

[WorkStatus 37](#_Toc85567089)

[Description 37](#_Toc85567090)

[Fields 37](#_Toc85567091)

# Android SDK Intro

## Function and Purpose

The sleeppace Android SDK is a software development kit for rapid Internet of things app development on the Android platform launched by sleeppace. The SDK encapsulates the communication process between app and hardware, and provides external functions such as device configuration, device control and data query. Using the SDK, users do not need to care about complex communication protocols and underlying implementation, but only need to focus on the interaction and business level of the app.

# Integration

## 1 .SDK Library

|  |  |
| --- | --- |
| **Library** | **Description** |
| sdkcore.jar | SDK core library |
| baseautopillowsdk.jar | Automatic monitoring pillow foundation library |
| p401msdk.jar | P401M pillow Library |
| sdkalgorithm.jar | Algorithm library |
| libalgorithm.so | Algorithm library |

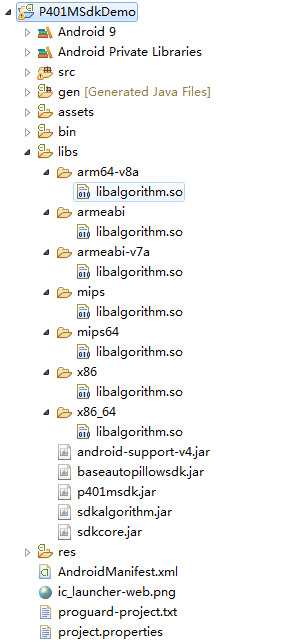
## 2 .Integration

There are many Android development tools, and here we introduce the engineering configuration method of Sleepace SDK with Eclipse.

**Step 1**：

In the project to create a "libs" folder, copy sdkcore.jar, baseautopillowsdk.jar, p401msdk.jar, sdkalgorithm.jar to "libs" folder, copy libalgorithm.so to "libs \ armeabi" folder.

Like this:



**Setp 2:**

Config the “AndroidManifest.xml”

<uses-permission android:name=*"android.permission.BLUETOOTH"*/>

<uses-permission android:name=*"android.permission.BLUETOOTH\_ADMIN"*/>

<uses-feature android:name=*"android.hardware.bluetooth\_le"* android:required=*"true"*/>

<uses-permission android:name=*"android.permission.ACCESS\_FINE\_LOCATION"* />

After the above operations, you can normally use all the functions provided by the sleep SDK.

# API

## 1.API initialization

P401MHelper.getInstance(Context mContext);

### Description

P401MHelper Initialization

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| mContext | Context | Conetxt |

## Set MTU

**public** **void** setMTU(**int** mtu, IResultCallback callback)

### Description

It is used to set the MTU size. It is recommended to set it before connecting the device, which can significantly improve the firmware upgrade speed. For specific usage, please refer to the demo

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| mtu | int | MTU size, interval: [20, 509], the size supported by the actual device will be returned in the callback |
| callback | IResultCallback | Callback the interface. When the interface call is successful, the actual supported MTU size of the device is returned. |

## Connnect Device

**public** **void** login(String deviceName, String address, DeviceType deviceType, **int** userId, **int** timeout, IResultCallback<[LoginBean](#_LoginBean_1)> cb)

### Description

Connect Pillow and setting userId

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceName | String | deviceName |
| address | String | Bluetooth address |
| deviceType | enum | The device type depends on the access device. Refer to the DeviceType class. The current device type is:  ***DEVICE\_TYPE\_P401M*** | |
| userId | int | userId does not belong to Sleepace.  userId belong to partner  **Why need it：**  Pillow separates the data according to userId.  It mean user A connects to device, generates and gets the data which only belong to user A. User A can’t get the data of user B |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<[LoginBean](#_LoginBean)> | Callback function, if success,return [LoginBean](#_LoginBean_1) Obj |

## Get Battery

**public** **void** getBattery(**int** timeout, IResultCallback<[BatteryBean](#_BatteryBean_1)> cb)

### Description

Get battery

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<[BatteryBean](#_BatteryBean)> | Callback function, if success,return [BatteryBean](#_BatteryBean) Object |

## Get Device Version

**public** **void** getDeviceVersion(**int** timeout, IResultCallback<String> cb)

### Description

Get current version of device

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<String> | Callback function, if success,return the version of device |

## Query device collection status

**public** **void** queryCollectStatus(**int** timeout, IResultCallback<[CollectStatus](#_CollectStatus)> cb)

### Description

It is used to query whether the equipment is currently being monitored and how long it has been monitored

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout in milliseconds |
| cb | IResultCallback<[CollectStatus](#_CollectStatus)> | Call back the interface to return the current status information of the equipment. For details, refer to:[CollectStatus](#_CollectStatus) |

## Stop Monitoring/Collecting

**public** **void** stopCollection(**int** timeout, IResultCallback<Void> cb)

### Description

Stop Monitoring/Collecting

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | IResultCallback<Void> | Callback function |

## Get Sleep Data (Real-time)

**public** **void** startRealTimeData(**int** timeout, IResultCallback<RealTimeData> cb)

### Description

Get Real-time Data

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<[RealTimeData](#_RealTimeData_1)> | Callback function |

## Stop Getting Sleep Data

**public** **void** stopRealTimeData(**int** timeout, IResultCallback<Void> cb)

### Description

Stop getting real-time data

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<Void> | Callback function |

## Get The Signal Strength (Real-time)

**public** **void** startOriginalData(**int** timeout, IResultCallback<OriginalData> cb)

### Description

Get the signal strength

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<[OriginalData](#_OriginalData_1)> | Callback function |

## Stop Getting The Signal Strength

**public** **void** stopOriginalData(**int** timeout, IResultCallback<Void> cb)

### Description

Stop Getting The Signal Strength

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback<Void>](#_IDataCallback<T>) | Callback function |

## Get Sleep Report

**public** **void** historyDownload(**int** startTime, **int** endTime, **int** sex, IResultCallback<List<[HistoryData](#_HistoryData_1)>> cb)

### Description

It is used to download historical data. Before downloading data, query how long the equipment has collected by querying the equipment collection status interface. When it is more than 10 minutes, Download historical data. Before downloading data, if the original data is opened, close the original data first. If the real-time data is opened, close the real-time data first. For specific usage, refer to demo.

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| starTime | int | Start time(timestamp), Unit(second) |
| endTime | int | End time(timestamp), Unit(second) |
| sex | int | Gender,1:male 0:female |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<List<[HistoryData](#_HistoryData_1)>> | Callback function |

## Firmware Update 1

**public** **void** upgradeDevice(**final** String pKey, **final** String hashCode, **final** File file, **final** IResultCallback<Integer> cb)

### Description

Firmware Update

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| pKey | String | Get it from Sleepace |
| hashCode | String | Get it from Sleepace |
| file | File | Firmware object |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<Integer> | Callback function, Return upgrade progress |

## Firmware Update 2

**public** **void** upgradeDevice(**final** String pKey, **final** String hashCode, **final** File file, **final** IResultCallback<Integer> cb)

### Description

Firmware Update

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| pKey | String | Get it from Sleepace |
| hashCode | String | Get it from Sleepace |
| is | InputStream | File data flow |
| timeout | int | Timeout, Unit(Millisecond) |
| cb | [IResultCallback](#_IDataCallback<T>)<Integer> | Callback function, Return upgrade progress |

## Wait for device confirmation

**public** **void** waitConfirm(IResultCallback cb)

### Description

It is used to open the device confirmation notification. After calling the interface successfully, press the power button on the device twice continuously, and the device will report the status message 0xf0. For specific usage, refer to demo

Application scenario: when the user binds the device, in order to prevent the wrong device from being bound, call the interface, then press the power button on the device twice, and bind after the app receives the confirmation notice. (this step is not necessary, and the customer determines whether it is necessary according to his own needs)

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback | Callback function |

## Monitor the status messages reported by the device

**public** **void** addDeviceStateListener([DeviceStateListener](#_DeviceStateListener) deviceStateListener)

### Description

Used to add device status message listening

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceStateListener | [DeviceStateListener](#_DeviceStateListener) | Device status listening callback |

## Remove the status message listening reported by the device

**public** **void** removeDeviceStateListener([DeviceStateListener](#_DeviceStateListener) deviceStateListener)

### Description

Used to remove device status message listening

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceStateListener | [DeviceStateListener](#_DeviceStateListener) | Device status listening callback |

## Bluetooth music configuration settings

**public** **void** musicConfigSet(**byte** onoff, **short** duration, IResultCallback<?> cb)

### Description

Used to set Bluetooth music parameters

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| onoff | byte | Intelligent stop switch, 1: on 0: off |
| duration | short | Timing duration, in minutes |
| cb | IResultCallback | Callback function |

## Bluetooth music configuration acquisition

**public** **void** musicConfigGet(IResultCallback<[MusicConfig](#_MusicConfig)> cb)

### Description

Used to get Bluetooth music configuration

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | |
| cb | IResultCallback | | Callback function |

## Operation related to sleep aid

**public** **void** sleepAidConfigSet(**byte** musicFlag, **short** musicId, **byte** volume, **byte** circleMode, **byte** smartFlag, **short** sleepAidDuration, IResultCallback<?> cb)

### Description

Used to start, stop, configure sleep aid parameters, etc

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| musicFlag | byte | Music switch, 0: off, 1: on |
| musicId | short | Music ID |
| volume | byte | Volume [1,8] |
| circleMode | byte | Loop mode 0: playlist 1: single loop |
| smartFlag | byte | Intelligent stop playback 0: off 1: on |
| sleepAidDuration | short | Sleep aid duration, unit: minute |
| cb | IResultCallback | Callback function |

## Sleep aid parameter acquisition

**public** **void** sleepAidConfigGet(IResultCallback< [SleepAidConfig](#_SleepAidConfig)> cb)

### Description

Used to obtain sleep aid parameter information

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | |
| cb | IResultCallback | | Callback function |

## Alarm clock acquisition

**public** **void** alarmGet(**byte** alarmId, IResultCallback<List<[AlarmInfo](#_AlarmInfo)>> cb)

### Description

Used to obtain alarm clock information

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | |
| cb | IResultCallback | | Callback function |

## Alarm clock setting

**public** **void** alarmSet(**byte** alarmId, **byte** enable, **byte** smartAwaken, **byte** smartOffset, **byte** hour, **byte** minute,

**byte** repeat, **byte** snoozeCount, **byte** snoozeDuration, **byte** volume, **short** musicId, **int** timestamp, **byte** useFul, IResultCallback cb)

### Description

Used to set alarm parameters

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| alarmId | byte | 0: indicates the local alarm clock. 1: indicates the alarm clock added by app, increasing in turn |
| enable | byte | Alarm switch, 0: off, 1: on |
| smartAwaken | byte | Whether to wake up intelligently, 0: No, 1: Yes |
| smartOffset | byte | Intelligent wake-up offset, unit: minutes, for example, 30 minutes, the alarm clock will look for the light sleep state 30 minutes in advance to ring the alarm clock. |
| hour | byte | 24-hour hour system (0-23) |
| minute | byte | Minutes (0-59) |
| repeat | byte | Week 1-7 switch (position 0-6): 0: off, 1: on  All 0 indicates a single alarm clock, and the time stamp is set as the expiration standard |
| snoozeCount | byte | Number of sleepiness |
| snoozeDuration | byte | Sleep duration, in minutes |
| volume | byte | Alarm volume [1,8] |
| brightness | byte | Maximum light brightness [0100] 0: not on, this parameter of the device is invalid, pass it to 0 |
| musicId | short | Music ID of alarm clock |
| timestamp | int | Time stamp of single alarm clock, in seconds |
| useFul | byte | 0 indicates that the alarm clock has been deleted. 1 the alarm clock is valid. To delete the alarm clock, use this interface and set this field to 0 |
| cb | IResultCallback | Callback function |

## Alarm clock Preview

**public** **void** alarmPreview(**byte** action, **short** musicId, **byte** volume, IResultCallback cb)

### Description

Used to preview the alarm clock and listen to the alarm music

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| action | byte | 1: Play  0: stop  2: Adjust volume |
| musicId | short | Music ID |
| volume | byte | volume |
| cb | IResultCallback | Callback function |

## Off pillow alarm clock setting

**public** **void** offPillowAlarmSet(**byte** enable, **byte** action, IResultCallback cb)

### Description

Used for parameter setting of off pillow alarm clock

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| enable | byte | 1: on  0: off |
| action | byte | 0：enter sleepiness. This field takes effect only when the headrest release switch is turned on  1：Turn off the alarm clock |
| cb | IResultCallback | Callback function |

## Off pillow alarm clock acquisition

**public** **void** offPillowAlarmGet(IResultCallback<[OffPillowAlarmConfig](#_OffPillowAlarmConfig)> cb)

### Description

Used to obtain parameter information of off pillow alarm clock

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback | Callback function |

## Query device working status

**public** **void** workStatusGet(IResultCallback<[WorkStatus](#_WorkStatus_1)> cb)

### Description

Used to query device working status

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | |
| cb | IResultCallback | | Callback function |

## Monitor the working status of registered device

**public** **void** addWorkStatusListener([WorkStatusListener](#_WorkStatus) workStatusListener)

### Description

Used to register device working status monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| workStatusListener | [WorkStatusListener](#_WorkStausListener) | Device working status monitoring interface |

## Remove device working status monitoring

**public** **void** removeWorkStatusListener([WorkStatusListener](#_WorkStatus) workStatusListener)

### Description

Used to remove device working status monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| workStatusListener | [WorkStatusListener](#_WorkStausListener) | Device working status monitoring interface |

# Object Description

## StatusCode

### Description

Status of execution

### Fields

|  |  |
| --- | --- |
| **Field** | **Description** |
| SUCCESS | success |
| DISCONNECT | failed |
| TIMEOUT | timeout |
| FAIL | Bluetooth is disconnected |
| NOT\_ENABLE | Bluetooth is not open |
| PARAMETER\_ERROR | Parameter error |

## IResultCallback<T>

### Description

Callback interface

### Function

**void** onDataCallback(CallbackData<T> cd)

callback function

## CallbackData<T>

### Description

Callback object

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| status | Int | [Status](#_StatusCode) of execution |
| type | int | Interface Type, used to distinguish between operating interface |
| result | T | The result of execution |

## LoginBean

### Description

The result of Connnect Device.

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| hardwareVersion | String | Device version |
| deviceId | String | Device id |

## BatteryBean

### Description

The result of getting battery

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| chargingState | int | Charging or not，  0：not  1: charging |
| quantity | int | Percentage of battery, It’s disable when charging |

## CollectStatus

### Description

Device collection data status information class

### Fields

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | |
| collectState | byte | Acquisition status  0: not collected 1: collecting |
| timestamp | int | Time stamp of starting collection, in seconds, current time stamp – start time stamp, which shows how long the equipment has collected |

## RealTimeData

### Description

The result of getting sleep data (Real-time)

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| heartRate | short | Heart rate |
| breathRate | short | Breath rate |
| status | byte | [SleepStatusType](#_SleepStatusType) |
| statusValue | int | The value of status. Unit(Millisecond) |
| sleepFlag | int | Asleep or not  1: asleep  0: not |
| wakeFlag | int | Awake or not  1: awake  0: not |

## OriginalData

### Description

The result of getting signal strength

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| heartRate | float[] | Heart rate |
| breathRate | Float[] | Breath rate |

## HistoryData

### Description

The result of getting sleep report

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| summary | Summary | History data [Summary](#_Summary) |
| detail | Detail | History data [Detail](#_Detail) |
| analy | Analysis | [Analysis](#_Analysis) |

## Summary

### Description

Summary of sleep report

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| recordCount | int | The length of Collecting.  Unit(minute)  Eg:  300. It means you collect for 300 minutes |
| startTime | int | Start time(timestamp). Unit(second) |
| stopMode | int | How to stop collecting：  0: Call the method “Stop Collecting”  1: stop automatically if you leave the bed for an hour  2: Error(a、Collect more than 24 hours，b、Pillow shutdown c、upgrade)  3: restart |
| timeStep | int | Record interval (default 60s, ie: 1 minute a time a point) |
| timezone | int | Timezone |

## Detail

### Description

Detail of sleep report

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| data1 | int[] | raw data1 |
| data2 | int[] | raw data2 |
| data3 | int[] | raw data3 |
| data4 | int[] | raw data4 |
| breathRate | int[] | Breath rate |
| heartRate | int[] | Heart rate |
| status | int[] | Status |
| statusValue | int[] | The value of status |

## Analysis

### Description

Analysis of sleep report

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| avgBreathRate | int | Average breath rate(n counts per min) |
| avgHeartRate | int | Average heart rate(n counts per min) |
| fallAlseepAllTime | int | Fall asleep time(Unit:min) |
| wakeAndLeaveBedBeforeAllTime | int | Duration of awake before getting up(Unit:min) |
| leaveBedTimes | int | Counts of leaving bed |
| trunOverTimes | int | Counts of turning over |
| bodyMovementTimes | int | Counts of body movement |
| heartBeatPauseTimes | int | Counts of Heat beat Pause |
| breathPauseTimes | int | Counts of apnea |
| deepSleepPerc | int | Deep sleep percentage |
| inSleepPerc | int | Mid sleep percentage |
| lightSleepPerc | int | Light sleep percentage |
| wakeSleepPerc | int | Awake percentage |
| duration | int | Sleep duration(Unit:min) |
| wakeTimes | int | Awake times |
| lightSleepAllTime | int | Duration of Light sleep(Unit:min) |
| inSleepAllTime | int | Duration of Mid sleep(Unit:min) |
| deepSleepAllTime | int | Duration of Deep sleep)(Unit:min) |
| wake | int | Duration of Awake)(Unit:min) |
| breathPauseAllTime | int | Duration of Apnea(Unit:seconds) |
| heartBeatPauseAllTime | int | Duration of heart beat pause)(Unit:seconds) |
| outOfBedDuration | int | Duration of leaving bed(Unit:min) |
| maxHeartBeatRate | int | Maximum heart rate(n counts per min) |
| maxBreathRate | int | Maximum breath rate(n counts per min) |
| minHeartBeatRate | int | Minimum heart rate)(n counts per min) |
| minBreathRate | int | Minimum breath rate(n counts per min) |
| heartBeatRateFastAllTime | int | Duration of tachycardia(Unit:seconds) |
| heartBeatRateSlowAllTime | int | Duration of bradycardia(Unit:seconds) |
| breathRateFastAllTime | int | Duration of tachypnea(Unit:seconds |
| breathRateSlowAllTime | int | Duration of bradypnea(Unit:seconds) |
| sleepScore | int | Score  90>=score<=100 Bravo!  80>=score<90 Good!  60>=score<80, average!  score <60 Bad |
| sleepCurveArray | float[] | Example:  [0.212,1.231,2.111,0.212,1.231,2.111,....]  0: awake 0 ~ 1: light sleep 1 ~ 2: moderate sleep 2 ~ 3: deep sleep  Drawing sleep curve (Unit:min) |
| sleepCurveStatusArray | ~~short[]~~ | ~~Sleep Event Flag (Unit:min)~~ |
| breathRateStatusAry | int[] | Apnea, It used to draw the graph  Example:  [0,0,1,0,2]  0: nothing  Other: Duration of Apnea in this minute(Unit:seconds) |
| heartRateStatusAry | int[] | Heart beat pause, It used to draw the graph  Example:  [0,0,1,0,2]  0: nothing  Other: Duration of Heat beat Pause in this minute(Unit:seconds) |
| leftBedStatusAry | int[] | Leave bed, It used to draw the graph  Example:  [0,0,1,0,2]  0: nothing  Other: Duration of leaving bed in this minute(Unit:seconds) |
| turnOverStatusAry | int[] | Turn over, It used to draw the graph  Example:  [0,0,1,0,2]  0: nothing  Other: the times of turning over |
| scaArray | byte[] | Analytical Sleep Status  0:awake, 1:light Sleep, 2:middle sleep, 3:deep sleep |
| algorithmVer | String | Algorithm version |
| fallsleepTimeStamp | int | The time you fall asleep(timestamp) |
| wakeupTimeStamp | int | The time you wake up(timestamp) |
| reportFlag | int | 1. Long report(>3h) 2. Short report(>10m && <3h) |
| md\_body\_move\_decrease\_scale | short | Score Deduction:Score Deduction due to body movement |
| md\_leave\_bed\_decrease\_scale |  | Score Deduction:Score Deduction due to the times of leaving bed |
| md\_wake\_cnt\_decrease\_scale | short | Score Deduction:Score Deduction due to the wake count |
| md\_start\_time\_decrease\_scale | short | Score Deduction:Score Deduction due to sleeping time (too late) |
| md\_fall\_asleep\_time\_decrease\_scale | short | Score Deduction:Score Deduction due to long falling sleep time |
| md\_perc\_deep\_decrease\_scale | short | Score Deduction:Score Deduction due to the deep sleep |
| md\_sleep\_time\_decrease\_scale | short | Score Deduction due to sleeping time too short |
| md\_sleep\_time\_increase\_scale | short | Score Deduction due to sleeping time too long |
| md\_breath\_stop\_decrease\_scale | short | Score Deduction:Score Deduction due to breathing stop |
| md\_heart\_stop\_decrease\_scale | short | Score Deduction:Score Deduction due to Heart beat stop |
| md\_heart\_low\_decrease\_scale | short | Score Deduction:Score Deduction due to slow heart beat |
| md\_heart\_high\_decrease\_scale | short | Score Deduction:Score Deduction due to Rapid heart beat |
| md\_breath\_low\_decrease\_scale | short | Score Deduction:Score Deduction due to slow breathing |
| md\_breath\_high\_decrease\_scale | short | Score Deduction:Score Deduction due to rapid breathing |
| md\_perc\_effective\_sleep\_decrease\_scale | short | Deduction item: score deduction for benign sleep |
| exceptioncode | short | Error code  0: report normal  50: short report less than 3 hours  51: short report of more than 3 hours, but the opposite direction of awake point and sleep point is greater than or equal to 3 hours  52: more than 3 hours, sleeping point and waking point less than or equal to 20 minutes  53: prompt report for more than 3 hours and higher off bed density  100: wrong version number  101: clipping function allocation failed  102: the data pointer is null  103: the internal allocation char array is empty  104: one of the input arrays is all 0  105: failed to allocate the collation status array  106: sleep point calculation failed  107: sober point calculation failed  110: Universal allocation failed  111: heart rate respiratory structure dynamic allocation failure  112: the average value of heart rate and respiratory rate is 0, and the data is invalid |

## SleepStatusType

### Description

Status value of monitoring

### Fields

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Value** | **Description** |
| SLEEP\_OK | byte | 0x00 | normal |
| SLEEP\_INIT | byte | 0x01 | init |
| SLEEP\_B\_STOP | byte | 0x02 | apnea |
| SLEEP\_H\_STOP | byte | 0x03 | Heartbeat pause |
| SLEEP\_BODYMOVE | byte | 0x04 | Body movement |
| SLEEP\_LEAVE | byte | 0x05 | Leaving bed |
| SLEEP\_TURN\_OVER | byte | 0x06 | Turning over |
| SLEEP\_BODYMOVE\_TEMP | byte | 0x07 | Amplitude of body motion |
| SLEEP\_INVALID | byte | -1 | invalid |

## DeviceStateListener

### Interface introduction

Device status listening callback

### Method description

**public** **void** onDeviceStateChanged([DeviceState](#_DeviceState) deviceState)

This method is called back when the state of the device side changes

## DeviceState

### Description

Device status information class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| type | byte | Device status message type:  0x01: low power  0x02: Collection start (there will be no such notification when normal start)  0x03: end of collection (there will be no such notification after normal end)  0x04: in bed  0x05: out of bed  0x06: sleep  0x07: wake up  0x08: awake  0xf0: Add Device confirmation (Note: byte type 0xf0 is negative) |
| value | byte | Status value. When the status is 0x01, it indicates the percentage of power (0-100)  In other states, the value is 0, which is meaningless |

## MusicConfig

### Description

Bluetooth music configuration information class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| onoff | byte | Intelligent stop switch, 1: on 0: off |
| duration | short | Timing duration, in minutes |

## SleepAidConfig

### Description

Sleep aid configuration information class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| musicFlag | byte | Music switch, 0: off, 1: on |
| musicId | short | Music ID |
| volume | byte | Volume [1,8] |
| circleMode | byte | Loop mode 0: playlist 1: single loop |
| smartFlag | byte | Intelligent stop playback 0: off 1: on |
| sleepAidDuration | short | Sleep aid duration, unit: minute |

## AlarmInfo

### Description

Alarm information class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| alarmId | byte | 0: indicates the local alarm clock. 1: indicates the alarm clock added by app, increasing in turn |
| enable | byte | Alarm switch, 0: off, 1: on |
| smartAwaken | byte | Whether to wake up intelligently, 0: No, 1: Yes |
| smartOffset | byte | Intelligent wake-up offset, unit: minutes, for example, 30 minutes, the alarm clock will look for the light sleep state 30 minutes in advance to ring the alarm clock. |
| hour | byte | 24-hour hour system (0-23) |
| minute | byte | Minutes (0-59) |
| repeat | byte | Week 1-7 switch (position 0-6): 0: off, 1: on  All 0 indicates a single alarm clock, and the time stamp is set as the expiration standard |
| snoozeCount | byte | Number of sleepiness |
| snoozeDuration | byte | Sleep duration, in minutes |
| volume | byte | Alarm volume [1,8] |
| brightness | byte | Maximum light brightness [0100] 0: not on, this parameter of the device is invalid, pass it to 0 |
| musicId | short | Music ID of alarm clock |
| timestamp | int | Time stamp of single alarm clock, in seconds |
| useFul | byte | 0 indicates that the alarm clock has been deleted. 1 the alarm clock is valid. To delete the alarm clock, use this interface and set this field to 0 |

## OffPillowAlarmConfig

### Description

Off pillow alarm clock configuration

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| enable | byte | 1: on  0: off |
| action | byte | 0: enter sleepiness. This field takes effect only when the headrest release switch is turned on  1: Turn off the alarm clock |

## WorkStausListener

### Interface introduction

Device working status monitoring callback

### Method description

**public** **void** onWorkStatusChanged([WorkStatus](#_WorkStatus_1) workStatus);

This method is called back when the working state of the device side changes

## WorkStatus

### Description

device working status information class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| alarmStatus | byte | Alarm status, 0: stop 1: alarm operation 2: sleepy 3 alarm preview mode |
| sleepAidStatus | byte | Sleep aid status, 0x00: not running, 0x01: running |
| musicStatus | byte | Music status, 0: stop, 1: play |
| volume | byte | Sleep music volume |
| musicId | short | Sleep music ID |
| sleepAidLeftTime | byte | Remaining duration of sleep aid, unit: minute |
| circleMode | byte | Loop mode, 0 list play 1 single loop |
| pinCodeSwitch | byte | Pin code switch, 0: off, 1: on |
| btState | byte | Bluetooth status, 0: disconnected 1: connected 2: play |
| btVolume | byte | Bluetooth music volume |
| musicDuration | int | Total length of currently playing music |
| currentPosition | int | Current music playing position |