|  |
| --- |
|  |
| **SDC100 Android SDK Description** |
| **V1.0** |
|  |
| **Author: WangYong** |
| **2023/12/13** |

# Change log

|  |  |  |
| --- | --- | --- |
| **Date** | **Log** | **Author** |
| 2023-12-13 | Create | WangYong |
| 2024-04-10 | Add ahi related parameters to the analysis object | WangYong |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
|  |

# Catalog

[Change log 1](#_Toc153819333)

[Catalog 2](#_Toc153819334)

[Android SDK Intro 8](#_Toc153819335)

[1. Function and Purpose 8](#_Toc153819336)

[Integration 8](#_Toc153819337)

[1 .SDK framework 8](#_Toc153819338)

[2 .Integration 8](#_Toc153819339)

[Eclipse Config 8](#_Toc153819340)

[API 10](#_Toc153819341)

[1.API initialization 10](#_Toc153819342)

[Description 10](#_Toc153819343)

[Parameters 10](#_Toc153819344)

[2. Connect Device(BLE) 10](#_Toc153819345)

[Description 10](#_Toc153819346)

[Used to connect devices through Bluetooth. After connecting the device, it can be networked and controlled. 10](#_Toc153819347)

[Parameters 10](#_Toc153819348)

[3. Set the TCP server address for device connection (BLE) 11](#_Toc153819349)

[Description 11](#_Toc153819350)

[Parameters 11](#_Toc153819351)

[4. Obtain the TCP server address for device connection (BLE) 11](#_Toc153819352)

[Description 11](#_Toc153819353)

[Parameters 11](#_Toc153819354)

[5. Set WiFi information for device (BLE) 11](#_Toc153819355)

[Parameters 12](#_Toc153819356)

[6. Obtain WiFi information for device connection (BLE) 12](#_Toc153819357)

[Parameters 12](#_Toc153819358)

[7. Get device information (BLE) 12](#_Toc153819359)

[Parameters 12](#_Toc153819360)

[8. bed control (BLE) 13](#_Toc153819361)

[Parameters 13](#_Toc153819362)

[9. Intelligent Snoring Stop Switch Settings (BLE) 13](#_Toc153819363)

[Parameters 14](#_Toc153819364)

[10. Intelligent snoring stop switch get (BLE) 14](#_Toc153819365)

[Parameters 14](#_Toc153819366)

[11. Bed bottom light switch setting (BLE) 14](#_Toc153819367)

[Parameters 14](#_Toc153819368)

[12. Obtaining the working status of the electric bed (BLE) 14](#_Toc153819369)

[Parameters 15](#_Toc153819370)

[13. Login to the server(TCP) 15](#_Toc153819371)

[Description 15](#_Toc153819372)

[Parameters 15](#_Toc153819373)

[14. Query the device online status(TCP) 15](#_Toc153819374)

[Description 15](#_Toc153819375)

[Parameters 15](#_Toc153819376)

[15. Query sleep status(TCP) 16](#_Toc153819377)

[Description 16](#_Toc153819378)

[Parameters 16](#_Toc153819379)

[16. MIC module status get (TCP) 16](#_Toc153819380)

[Description 16](#_Toc153819381)

[Parameters 16](#_Toc153819382)

[17. Query collection status (TCP) 17](#_Toc153819383)

[Description 17](#_Toc153819384)

[Parameters 17](#_Toc153819385)

[18. Turn on real-time data(TCP) 17](#_Toc153819386)

[Description 17](#_Toc153819387)

[Parameters 17](#_Toc153819388)

[19. Turn off real-time data(TCP) 18](#_Toc153819389)

[Description 18](#_Toc153819390)

[Parameters 18](#_Toc153819391)

[20. End monitoring manually(TCP) 18](#_Toc153819392)

[Description 18](#_Toc153819393)

[Parameters 18](#_Toc153819394)

[21. Device firmware upgrade(TCP) 19](#_Toc153819395)

[Description 19](#_Toc153819396)

[Parameters 19](#_Toc153819397)

[22. Add bluetooth device connection status listener 19](#_Toc153819398)

[Description 19](#_Toc153819399)

[Parameters 19](#_Toc153819400)

[23. Remove bluetooth device connection status listener 20](#_Toc153819401)

[Description 20](#_Toc153819402)

[Parameters 20](#_Toc153819403)

[24. Register device online status change listener 20](#_Toc153819404)

[Description 20](#_Toc153819405)

[Parameters 20](#_Toc153819406)

[25. Unregister device online status change listener 20](#_Toc153819407)

[Description 20](#_Toc153819408)

[Parameters 21](#_Toc153819409)

[26. Register sleep report upload status listener 21](#_Toc153819410)

[Description 21](#_Toc153819411)

[Parameters 21](#_Toc153819412)

[27. Unregister sleep report upload status listener 21](#_Toc153819413)

[Description 21](#_Toc153819414)

[Parameters 21](#_Toc153819415)

[28. Register real-time data listener 22](#_Toc153819416)

[Description 22](#_Toc153819417)

[Parameters 22](#_Toc153819418)

[29. unregister real-time data listener 22](#_Toc153819419)

[Description 22](#_Toc153819420)

[Parameters 22](#_Toc153819421)

[30. Register for real-time sleep state monitoring 22](#_Toc153819422)

[Description 22](#_Toc153819423)

[Parameters 23](#_Toc153819424)

[31. Unregister for real-time sleep state monitoring 23](#_Toc153819425)

[Description 23](#_Toc153819426)

[Parameters 23](#_Toc153819427)

[32. register MIC state listener 23](#_Toc153819428)

[Description 23](#_Toc153819429)

[Parameters 23](#_Toc153819430)

[33. Unregister MIC state listener 24](#_Toc153819431)

[Description 24](#_Toc153819432)

[Parameters 24](#_Toc153819433)

[34. register work status listener 24](#_Toc153819434)

[Description 24](#_Toc153819435)

[Parameters 24](#_Toc153819436)

[35. Unregister work status listener 24](#_Toc153819437)

[Description 24](#_Toc153819438)

[Parameters 25](#_Toc153819439)

[Object Description 25](#_Toc153819440)

[StatusCode 25](#_Toc153819441)

[Description 25](#_Toc153819442)

[Fields 25](#_Toc153819443)

[IResultCallback<T> 25](#_Toc153819444)

[Description 25](#_Toc153819445)

[Function 25](#_Toc153819446)

[CallbackData<T> 26](#_Toc153819447)

[Description 26](#_Toc153819448)

[Fields 26](#_Toc153819449)

[DeviceInfo 26](#_Toc153819450)

[Description 26](#_Toc153819451)

[Fields 26](#_Toc153819452)

[MICState 26](#_Toc153819453)

[Description 26](#_Toc153819454)

[Fields 27](#_Toc153819455)

[ServerConfig 27](#_Toc153819456)

[Description 27](#_Toc153819457)

[Fields 27](#_Toc153819458)

[WiFiConfig 27](#_Toc153819459)

[Description 27](#_Toc153819460)

[Fields 27](#_Toc153819461)

[WorkStatus 28](#_Toc153819462)

[Description 28](#_Toc153819463)

[Fields 28](#_Toc153819464)

[SleepState 28](#_Toc153819465)

[Description 28](#_Toc153819466)

[Fields 28](#_Toc153819467)

[CollectState 29](#_Toc153819468)

[Description 29](#_Toc153819469)

[Fields 29](#_Toc153819470)

[RealTimeData 29](#_Toc153819471)

[Description 29](#_Toc153819472)

[Fields 29](#_Toc153819473)

[HistoryData 30](#_Toc153819474)

[Description 30](#_Toc153819475)

[Fields 30](#_Toc153819476)

[Summary 30](#_Toc153819477)

[Description 30](#_Toc153819478)

[Fields 30](#_Toc153819479)

[Detail 31](#_Toc153819480)

[Description 31](#_Toc153819481)

[Fields 31](#_Toc153819482)

[Analysis 31](#_Toc153819483)

[Description 31](#_Toc153819484)

[Fields 31](#_Toc153819485)

[SleepStatusType 34](#_Toc153819486)

[Description 34](#_Toc153819487)

[Fields 34](#_Toc153819488)

[OnlineStateListener 35](#_Toc153819489)

[Interface Introduction 35](#_Toc153819490)

[Method description 35](#_Toc153819491)

[SleepReportUploadStateListener 35](#_Toc153819492)

[Interface Introduction 35](#_Toc153819493)

[Method description 35](#_Toc153819494)

[RealtimeDataListener 36](#_Toc153819495)

[Interface Introduction 36](#_Toc153819496)

[Method description 36](#_Toc153819497)

[RealtimeSleepStateListener 36](#_Toc153819498)

[Interface Introduction 36](#_Toc153819499)

[Method description 36](#_Toc153819500)

[MICStateListener 36](#_Toc153819501)

[Interface Introduction 36](#_Toc153819502)

[Method description 37](#_Toc153819503)

[WorkStatusListener 37](#_Toc153819504)

[Interface Introduction 37](#_Toc153819505)

[Method description 37](#_Toc153819506)

# Android SDK Intro

## Function and Purpose

RestOn SDK,launched by Sleepace, is a software development kit for fast Internet APP development on android platform.

The SDK encapsulates the communication process between APP and hardware, and provides functions such as device configuration, device control and data query. Using the SDK, users do not need to care about complex communication protocols and the underlying implementation, only need to focus on the APP interaction and business level.

# Integration

## 1 .SDK framework

|  |  |
| --- | --- |
| **Framework** | **Description** |
| sdkcore.jar | SDK base core |
| wifidevicecommonsdk.jar | General HTTP interface package for WiFi devices |
| sdc100sdk.jar | SDC100 SDK |
| gson-2.8.0.jar | JSON parsing Toolkit (version 2.8.0 and below, there may be exceptions when compiling in the eclipse environment) |

## 2 .Integration

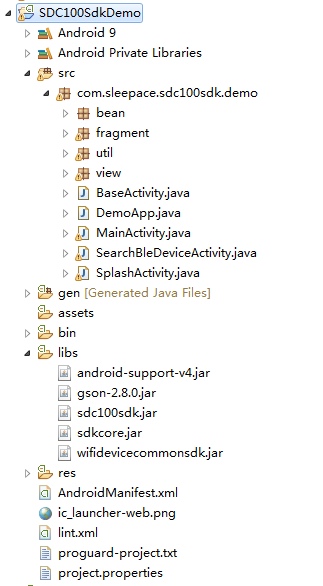
There are many Android development tools, and here we introduce the engineering configuration method of Sleepace SDK with Eclipse.

## Eclipse Config

**Step 1**：

In the project to create a "libs" folder, copy sdkcore.jar, wifidevicecommonsdk.jar, sdc100sdk.jar, gson-2.8.0.jar to "libs" folder.

Like this:



**Setp 2:**

Config the “AndroidManifest.xml”

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

<uses-permission android:name=*"android.permission.WRITE\_EXTERNAL\_STORAGE"*/>

<uses-permission android:name=*"android.permission.ACCESS\_NETWORK\_STATE"* />

# API

## 1.API initialization

SDC100Helper.getInstance(Context mContext);

### Description

SDC100Helper Initialization

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| mContext | Context | Context object. It is recommended to pass ApplicationContext |

## Connect Device(BLE)

**public** **void** connectDevice(String address, **int** timeout, **final** IResultCallback<Void> cb)

### Description

### Used to connect devices through Bluetooth. After connecting the device, it can be networked and controlled.

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| address | String | Bluetooth MAC address |
| timeout | int | Connection timeout, in milliseconds |
| cb | IResultCallback<Void> | Callback interface |

## Set the TCP server address for device connection (BLE)

**public** **void** serverConfigSet(**final** String ip, **final** **short** port, **final** IResultCallback cb)

### Description

Used to set the TCP server address for device connection. When configuring the device network, it is recommended to first set the TCP server address and then set the WiFi information

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| ip | String | IP address or domain name |
| port | short | server port |
| cb | IResultCallback<Void> | Callback interface |

## Obtain the TCP server address for device connection (BLE)

**public** **void** serverConfigGet(**final** IresultCallback<[ServerConfig](#_ServerConfig)> cb)

### Description

Used to obtain TCP server address information for device connections

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback<[ServerConfig](#_ServerConfig)> | Callback interface |

## Set WiFi information for device (BLE)

**public** **void** wifiConfigSet(**final** **byte**[] ssidRaw, **final** String pwd, **final** IResultCallback cb)

**Description**

WiFi information used to set device connections

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| ssidRaw | byte[] | Byte array of WiFi names, refer to demo |
| pwd | String | WiFi password |
| cb | IResultCallback<Void> | Callback interface |

## Obtain WiFi information for device connection (BLE)

**public** **void** wifiConfigGet(**final** IresultCallback<[WiFiConfig](#_WiFiConfig)> cb)

**Description**

Used to obtain WiFi information for device connections

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback<[WiFiConfig](#_WiFiConfig)> | Callback interface |

## Get device information (BLE)

**public** **void** getDeviceInfo(**final** IResultCallback<[DeviceInfo](#_DeviceInfo)> cb)

**Description**

Used to obtain device information

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback<[DeviceInfo](#_SleepState)> | Callback interface |

## bed control (BLE)

**public** **void** bedControl(**final** **byte** action, **final** **byte** value, **final** **byte** position, **final** IResultCallback cb)

**Description**

Used to control the electric bed

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| action | byte | Control code  0: Stop  1: Descending  2: Rising  3: Viewing mode  4: Reading mode  5: Snoring cessation mode  6: 0 gravity mode  7: FLAT mode  8: M1 Custom Mode  9: M2 Custom Mode  10: Child lock |
| value | byte | Control parameters  ① When the control code is child lock, 0- close child lock 1- open child lock  ② When the control code is up/down, 0- short press 1- long press  ③ When the control code is M1/M2, 0- Operation 1- Save  ④ Other control codes, this field is invalid |
| position | byte | Control position  0: Back 1: Legs 2: Head 3: Waist |
| cb | IResultCallback | Callback interface |

## Intelligent Snoring Stop Switch Settings (BLE)

**public** **void** smartStopSnoringSet(**final** **byte** onoff, **final** IResultCallback cb)

**Description**

Used to set the intelligent snoring stop switch

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| onoff | byte | 0: Close 1: Open |
| cb | IResultCallback | Callback interface |

## Intelligent snoring stop switch get (BLE)

**public** **void** smartStopSnoringGet(**final** IresultCallback<Byte> cb)

**Description**

Used to obtain the status of the intelligent snoring stop switch

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback<Byte> | Callback interface |

## Bed bottom light switch setting (BLE)

**public** **void** bedBottomLightSet(**final** **byte** onoff, **final** IResultCallback cb)

**Description**

Used for setting the bed bottom light switch

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| onoff | byte | 0: Close 1: Open |
| cb | IResultCallback | Callback interface |

## Obtaining the working status of the electric bed (BLE)

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

**Description**

Used for obtaining the working status of an electric bed, including the status of the child lock switch and the bed bottom light switch

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback<[WorkStatus](#_WorkStatus)> | Callback interface |

## Login to the server(TCP)

**public** **void** login(**final** String ip, **final** **int** port, **final** String sid, **final** String deviceId, **final** IResultCallback<Void> cb)

### Description

It is used to connect and log in to the server. IP, port and Sid are obtained through the authentication interface. Please refer to demo for specific usage

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| ip | String | Server IP |
| port | int | Server port |
| sid | String | Sid information |
| deviceId | String | Device ciphertext ID |
| cb | IResultCallback<Void> | Callback interface |

## Query the device online status(TCP)

**public** **void** queryDeviceOnlineState(**final** **short** deviceType, **final** String deviceId, **final** IResultCallback<Byte> cb)

### Description

It is used to query the online status of Device

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| cb | IResultCallback<Byte> | Callback interface, 0: not online 1: online 0xff: unknown state |

## Query sleep status(TCP)

**public** **void** querySleepState(**final** **short** deviceType, **final** String deviceId, **final** **int** num, **final** IResultCallback<[SleepState](#_SleepState)> cb)

### Description

Used to query sleep status

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| num | int | Left and right side numbering, 0 left side (single person version), 1 right side |
| cb | IResultCallback< [SleepState](#_SleepState)> | Callback interface to return sleep state information. Please refer to SleepState for details |

## MIC module status get (TCP)

**public** **void** micStateGet(**final** IResultCallback<[MICState](#_MICState)> cb)

### Description

Used to obtain the status of the MIC module

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| cb | IResultCallback<[MICState](#_MICState)> | Callback interface |

## Query collection status (TCP)

**public** **void** queryCollectState(**final** **short** deviceType, **final** String deviceId, **final** **int** num, **final** IResultCallback<[CollectState](#_RealTimeData_1)> cb)

### Description

Used to query device collection status

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| num | int | Left and right side numbering, 0 left side (single person version), 1 right side |
| cb | IResultCallback<[CollectState](#_RealTimeData_1)> | Callback interface |

## Turn on real-time data(TCP)

**public** **void** startRealTimeData(**final** **short** deviceType, **final** String deviceId, **final** **int** num, **final** IResultCallback<[RealTimeData](#_RealTimeData_1)> cb)

### Description

Used to view real-time heart rate breathing

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| num | int | Left and right side numbering, 0 left side (single person version), 1 right side |
|  |  |  |
| cb | IResultCallback< [RealTimeData](#_RealTimeData_1)> | Callback interface. After successful operation, the callback interface will report heart rate and respiratory data in real time. For details, please refer to: RealtimeData |

## Turn off real-time data(TCP)

**public** **void** stopRealTimeData(**final** **short** deviceType, **final** String deviceId, **final** **int** num, **final** IResultCallback<Void> cb)

### Description

It is used to turn off the real-time data. After shutdown, the device will not report the heart rate and respiratory data in real time, but it is still under monitoring.

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| num | int | Left and right side numbering, 0 left side (single person version), 1 right side |
| cb | IResultCallback<Void> | Callback interface |

## End monitoring manually(TCP)

**public** **void** stopCollection(**final** **int** userId, **final** **short** deviceType, **final** String deviceId, **final** **int** num, **final** IResultCallback<Void> cb)

### Description

Used to end monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| userId | int | User ID |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| num | int | Left and right side numbering, 0 left side (single person version), 1 right side |
| cb | IResultCallback<Void> | Callback interface |

## Device firmware upgrade(TCP)

**public** **void** upgradeDevice(**final** **float** deviceVersion, **final** **short** deviceType, **final** String deviceId, **final** IResultCallback<Integer> cb)

### Description

For device firmware upgrade

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceVersion | float | Firmware target version number |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| cb | IResultCallback<Integer> | Callback the interface. After the upgrade is successful, the interface returns the upgrade progress |

## Add bluetooth device connection status listener

**public** **void** addBleStateCallback(IConnectionStateCallback stateCallback)

### Description

Used to monitor the connection status of Bluetooth devices

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| IConnectionStateCallback | Custom interface | Bluetooth device connection status listener |

## Remove bluetooth device connection status listener

**public** **void** removeBleStateCallback(IConnectionStateCallback stateCallback)

### Description

Used to remove bluetooth device connection status listener

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| IConnectionStateCallback | Custom interface | Bluetooth device connection status listener |

## Register device online status change listener

**public** **void** registOnlineStateListener([OnlineStateListener](#_OnlineStateListener) onlineStateListener)

### Description

It is used to monitor the online status of Device

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [OnlineStateListener](#_OnlineStateListener) | Custom interface | Device online status monitoring, for details, refer to: OnlinestateListener |

## Unregister device online status change listener

**public** **void** unregistOnlineStateListener([OnlineStateListener](#_OnlineStateListener) onlineStateListener)

### Description

Used to log off device online status monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [OnlineStateListener](#_OnlineStateListener) | Custom interface | Device online status monitoring, for details, refer to: OnlinestateListener |

## Register sleep report upload status listener

**public** **void** registSleepReportUploadStateListener([SleepReportUploadStateListener](#_SleepReportUploadStateListener) sleepReportUploadState)

### Description

Used to monitor sleep report upload status

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [SleepReportUploadStateListener](#_SleepReportUploadState) | Custom interface | Sleep report upload state listener, for details, refer to: [SleepReportUploadStateListener](#_SleepReportUploadState) |

## Unregister sleep report upload status listener

**public** **void** unregistSleepReportUploadStateListener(SleepReportUploadStateListener onlineStateListener)

### Description

Used to log off sleep report upload status monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [SleepReportUploadStateListener](#_SleepReportUploadState) | Custom interface | Sleep report upload state listener, for details, refer to: [SleepReportUploadStateListener](#_SleepReportUploadState) |

## Register real-time data listener

**public** **void** registRealtimeDataListener([RealtimeDataListener](#_RealtimeDataListener) realtimeDataListener)

### Description

Used to monitor real-time data

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [RealtimeDataListener](#_RealtimeDataListener) | Custom interface | Real-time data monitoring, refer to: [RealtimeDataListener](#_RealtimeDataListener) |

## unregister real-time data listener

**public** **void** unregistRealtimeDataListener([RealtimeDataListener](#_RealtimeDataListener) realtimeDataListener)

### Description

Used to log off real-time data monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [RealtimeDataListener](#_RealtimeDataListener) | Custom interface | Real-time data monitoring, refer to: [RealtimeDataListener](#_RealtimeDataListener) |

## Register for real-time sleep state monitoring

**public** **void** registRealtimeSleepStateListener([RealtimeSleepStateListener](#_RealtimeSleepStateListener) realtimeSleepStateListener)

### Description

Used to monitor real-time sleep status

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [RealtimeSleepStateListener](#_RealtimeSleepStateListener) | Custom interface | Real-time data monitoring, refer to: [RealtimeSleepStateListener](#_RealtimeSleepStateListener) |

## Unregister for real-time sleep state monitoring

**public** **void** unregistRealtimeSleepStateListener([RealtimeSleepStateListener](#_RealtimeSleepStateListener) realtimeSleepStateListener)

### Description

Used to log off real-time sleep status monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| [RealtimeSleepStateListener](#_RealtimeSleepStateListener) | Custom interface | Real-time data monitoring, refer to: [RealtimeSleepStateListener](#_RealtimeSleepStateListener) |

## register MIC state listener

**public** **void** registMICStateListener(MICStateListener stateListener)

### Description

Used to monitor the status of the MIC module

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| MICStateListener | Custom interface | MIC module status listener |

## Unregister MIC state listener

**public** **void** unregistMICStateListener(MICStateListener stateListener)

### Description

Used for logging out of MIC module status monitoring

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| MICStateListener | Custom interface | MIC module status listener |

## register work status listener

**public** **void** registWorkStatusListener(WorkStatusListener listener)

### Description

Used to monitor the working status of the smart bed

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| WorkStatusListener | Custom interface | Work status listener |

## Unregister work status listener

**public** **void** unregistWorkStatusListener(WorkStatusListener listener)

### Description

Used to cancel the monitoring of the working status of the smart bed

### Parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| WorkStatusListener | Custom interface | Work status listener |

# Object Description

## StatusCode

### Description

Status of execution

### Fields

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Value** | **Description** |
| STATUS\_SUCCESS | int | 0x00 | success |
| STATUS\_FAILED | int | -1 | failed |
| STATUS\_TIMEOUT | int | -2 | timeout |
| STATUS\_DISCONNECT | int | -3 | Bluetooth is disconnected |
| STATUS\_BLUETOOTH\_NOT\_OPEN | int | -4 | Bluetooth is not open |
| STATUS\_PARAMETER\_ERROR | int | -5 | Parameter error |

## IResultCallback<T>

### Description

Callback interface

### Function

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

callback function

## CallbackData<T>

### Description

Callback object

### Fields

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

## DeviceInfo

### Description

Device Information

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceType | short | Device type |
| deviceId | String | Device ciphertext ID |
| firmwareVersion | String | Firmware version |

## MICState

### Description

MIC module status class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| leftMICState | byte | Left MIC module status 0: Normal 1: Not inserted 2: Poor contact |
| rightMICState | byte | Right side MIC module status 0: Normal 1: Not inserted 2: Poor contact |
| leftSensorState | byte | Left sensor module status 0: Normal 1: Not inserted 2: Poor contact |
| rightSensorState | byte | Right sensor module status 0: Normal 1: Not inserted 2: Poor contact |
| leftThreshold | byte | Left threshold, left snoring algorithm determines noise filtering threshold |
| rightThreshold | byte | Right threshold, right snoring algorithm determines noise filtering threshold |

## ServerConfig

### Description

Server address information class for device connection

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| ip | String | Server IP address or domain name |
| port | short | server port |

## WiFiConfig

### Description

WiFi information class for device connection

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| ssidRaw | byte[] | Byte array of WiFi names |
| password | String | wifi password |

## WorkStatus

### Description

Intelligent bed working status class

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| mode | byte | Working mode  0: Unknown state  1: Reserved  2: Reserved  3: Viewing mode  4: Reading mode  5: Snoring cessation mode  6: 0 gravity mode  7: FLAT mode  8: M1 Custom Mode  9: M2 Custom Mode |
| pushRodState | byte | Push rod operation status 0: Stop 1: Running |
| lockState | byte | Child lock status 0: closed 1 opened |
| bedBottomLightState | byte | Bedlight status 0: Off 1: On |
| backAngle | short | Back angle 0-90 ° |
| legAngle | short | Leg angle 0-90 ° |
| headAngle | short | Head angle 0-90 ° |
| waistAngle | short | Lumbar angle 0-90 ° |

## SleepState

### Description

Sleep state

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| asleepFlag | byte | Falling asleep sign 0x00: Not falling asleep 0x01: Falling asleep Other: Invalid |
| wakeupFlag | byte | Awake sign 0x00: No wake-up 0x01: Wake-up Others: Invalid |
| outOfBedFlag | byte | Out of bed sign 0x00: Not out of bed 0x01: Out of bed Other: Invalid |
| sleepState | byte | Sleep depth 0x00: invalid (indicating that the function is not supported yet) 0x01: awake 0x02: light sleep 0x03: mid sleep 0x04: deep sleep |
| situpFlag | byte | Sitting up 0x00: lying down 0x01: sitting up |

## CollectState

### Description

Device collection status

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| state | byte | Device collection status 0x00: Not collected 0x01: Collecting |
| startTime | int | Time stamp at the beginning of acquisition, in seconds |

## RealTimeData

### Description

Real-time data

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| deviceId | String | Device ciphertext ID |
| deviceType | short | Device type |
| leftRight | byte | Left and right side numbering, 0 left (single person), 1 right side |
| status | byte | Status identification, see SleepStatusType for detailed description |
| statusValue | short | The value of status. Unit(second) |
| sleepFlag | byte | Asleep or not  1: asleep  0: not |
| wakeFlag | byte | Awake critical logo :  1: yes  0: no |

## HistoryData

### Description

Historical data

### 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、reston shutdown c、upgrade)  3: restart |
| timeStep | int | Record interval (default 60s, ie: 1 minute a time a point) |
| timezone | int | Timezone |
| arithmeticVer | String | Algorithm version |
| firmwareVers | String | Firmware version |
| deviceType | short | Device Type |

## Detail

### Description

Detail of sleep report

### Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| 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** |
| averageBreathRate | int | Average breath rate(n counts per min) |
| averageHeartBeatRate | 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(Abandoned)~~ |
| 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) |
| wakeAllTime | int | Duration of Awake)(Unit:min) |
| breathPauseAllTime | int | Duration of Apnea(Unit:seconds) |
| ~~heartBeatPauseAllTime~~ | ~~int~~ | ~~Duration of heart beat pause)(Unit:seconds)(~~ ~~Abandoned)~~ |
| leaveBedAllTime | 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,....]  0.5>&>-0.1: awake 1.5>&>=0.5: light sleep 2.5>&>=1.5: moderate sleep 3>&>=2.5: 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 |
| sleepState | 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(Abandoned)~~ |
| 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 | Score Deduction:Score Deduction due to good sleeping (ratio of middle sleep/deep sleep) |
| bmCnt | int | Number of body movements and turns after falling asleep |
| ahIndex | int | Normal: ahIndex<5; Low risk: 5<=ahIndex<15; Medium risk: 15<=ahIndex<30; High risk: 30<=ahIndex |
| ahiMaxDur | int | The duration of the longest respiratory event (central, obstructive, or hypoventilation) during sleep, in seconds |
| csaDur | int | Duration of central apnea during sleep, in seconds |
| csaCnt | int | The number of occurrences of central apnea during sleep |
| csaMaxDur | int | The longest duration of central apnea during sleep, in seconds |
| osaDur | int | Duration of obstructive apnea/hypopnea during sleep, in seconds |
| osaCnt | int | The number of occurrences of obstructive apnea/hypopnea during sleep |
| osaMaxDur | int | The longest duration of obstructive apnea/hypopnea during sleep, in seconds |
| ahiArrayStr | String | Hourly related statistical array string for apneaArr [0]: The start time of storing detailed information of each hour of apnea (actual sleep point: HHMM),Arr [1]: The end time of storing detailed information of each hour of apnea (time of awakening point: HHMM) Arr [2]: Duration (how many whole points are spanned), subsequent position storage, hourly AHI index. The specific storage protocol is as follows: |

ahiArrayStr Explanation：

|  |  |  |
| --- | --- | --- |
| index | type | Description |
| 0 | short | Start time, integer point of high byte storage time, minute of low byte storage time |
| 1 | short | End time, integer point of high byte storage time, minute of low byte storage time |
| 2 | short | Duration N hours |
| 3 | short | AHI index for the first hour |
| ... | short | ... |
| N | short | AHI index for the nth hour |

## 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(Abandoned)~~ |
| 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 |

## OnlineStateListener

### Interface Introduction

Device online status monitoring

### Method description

**void** onlineStateChanged(**byte** onlineState);

onlineState：0: offline 1: online 0xFF: unknown state

## SleepReportUploadStateListener

### Interface Introduction

Sleep report upload status monitoring

### Method description

**void** onStateChanged(**byte** state);

state：

0: No data (no need to wait any longer)

1: ready

2: syncing

3: uploading

4: Upload failed (no need to wait any longer)

5: Successful upload

6: End of upload

note:

0, subsequent content is invalid

1, indicating that the device has started the upload process, the subsequent content is invalid

2. It indicates that the device starts to synchronously monitor the device data process, and the subsequent content is invalid

Process description: The entire upload task starts in the preparation state (1) and ends in the end state (6)

## RealtimeDataListener

### Interface Introduction

Real-time data monitoring

### Method description

**void** onReceive([RealTimeData](#_RealTimeData_1) realTimeData);

realTimeData：Real-time data reported by the device, including heart rate and breath rate, etc. refer to：[RealTimeData](#_RealTimeData_1)

## RealtimeSleepStateListener

### Interface Introduction

Real-time sleep state monitoring

### Method description

**void** onReceive([SleepState](#_SleepState_1) sleepState);

sleepState：The real-time sleep status reported by the device refer to：[SleepState](#_SleepState_1)

## MICStateListener

### Interface Introduction

MIC module status monitoring

### Method description

**void** onReceive([MICState](#_MICState) state);

state：Real time MIC module status reported by the device, detailed reference:[MICState](#_MICState)

## WorkStatusListener

### Interface Introduction

Monitoring of device working status

### Method description

**void** onStatusChanged([WorkStatus](#_WorkStatus) status);

status：Real time working status reported by the device, detailed reference:[WorkStatus](#_WorkStatus)