**KP.006.06.006**

1

Unify Data Service System

Android SDK使用说明（健云方案）

**深圳市深新锐信息技术有限公司**



**文档变更日志**

|  |  |  |  |
| --- | --- | --- | --- |
| 起始人员 |  | | |
| 版本 | 描叙 | 撰写人员 | 日期 |
| 0.0.1 | 初稿 |  | 2018/4/23 |
| 0.0.2 | 增加ECG |  | 2019/11/8 |
| 0.0.3 | 增加体温和多天天气 |  | 2020/6/2 |
| 0.0.4 | 补充提醒id说明 |  | 2022/7/4 |
| 0.0.5 | 增加表盘相关功能 |  | 2022/10/20 |
| 0.0.6 | 增加ota固件，表盘升级和自定义壁纸更新功能 |  | 2023/01/10 |
| 0.0.7 | 增加在线表盘获取功能 |  | 2023/10/10 |
|  |  |  |  |

# 目录

[1. 目录 2](#_Toc31787)

[2. 接入须知 6](#_Toc24104)

[3. 各平台SDK接口说明 6](#_Toc657)

[3.1. Android版本说明 6](#_Toc18537)

[3.1.1. 初始化启动SDK 6](#_Toc26390)

[3.2. 数据结构说明 6](#_Toc24097)

[3.2.1. BleClientOption 6](#_Toc21692)

[3.2.2. UserProfile 7](#_Toc7904)

[3.2.3. DeviceProfile 7](#_Toc13064)

[3.2.4. AlarmInfoItem 7](#_Toc27016)

[3.2.5. Weather 8](#_Toc19728)

[4. 接口及回调 8](#_Toc11658)

[4.1. 验证SDK 8](#_Toc19035)

[4.2. 回调接口 9](#_Toc23265)

[4.3. 注册回调 9](#_Toc20303)

[4.4. 注销回调 9](#_Toc12079)

[4.5. 获取SDK授权状态 9](#_Toc2691)

[4.6. 设置参数 9](#_Toc12383)

[4.7. 设置扫描模式 9](#_Toc23857)

[4.8. 扫描蓝牙设备 9](#_Toc27065)

[4.8.1. 回调接口 10](#_Toc149)

[4.9. 连接设备 10](#_Toc26199)

[4.9.1. 设备连接状态改变 10](#_Toc22278)

[4.10. 获取连接状态 10](#_Toc31815)

[4.11. 读取连接设备蓝牙地址 10](#_Toc20230)

[4.12. 断开设备 10](#_Toc22003)

[4.13. 设置设备时间 10](#_Toc12188)

[4.13.1. 回调接口 11](#_Toc21374)

[4.14. 设置用户信息 11](#_Toc8373)

[4.14.1. 回调接口 11](#_Toc27691)

[4.15. 获取当前运动数据 11](#_Toc5987)

[4.15.1. 回调接口 11](#_Toc14367)

[4.16. 手环振动 11](#_Toc19756)

[4.16.1. 回调接口 12](#_Toc19697)

[4.17. 手环请求手机操作 12](#_Toc25295)

[4.17.1. 手环请求天气数据回调 12](#_Toc11231)

[4.18. 设置手环进入拍照模式 13](#_Toc4643)

[4.18.1. 回调接口 13](#_Toc13996)

[4.19. 设置久坐提醒时间 13](#_Toc12286)

[4.19.1. 回调接口 13](#_Toc2163)

[4.20. 设置睡眠时间段 13](#_Toc1606)

[4.20.1. 回调接口 13](#_Toc14824)

[4.21. 读取设备电量 13](#_Toc3995)

[4.21.1. 回调接口 14](#_Toc5354)

[4.22. 读取手环信息 14](#_Toc10553)

[4.22.1. 回调接口 14](#_Toc6476)

[4.23. 读取手环功能项 14](#_Toc13570)

[4.23.1. 回调接口 14](#_Toc1343)

[4.24. 设置闹钟 15](#_Toc13528)

[4.24.1. 回调接口 15](#_Toc26060)

[4.25. 特殊命令 15](#_Toc29093)

[4.26. 消息推送 15](#_Toc6191)

[4.26.1. 回调接口 16](#_Toc5059)

[4.27. 启动\关闭实时心率测量 16](#_Toc11040)

[4.27.1. 回调接口 16](#_Toc32686)

[4.27.2. 结束测试回调 16](#_Toc18248)

[4.28. 自动测试心率 16](#_Toc8474)

[4.28.1. 回调接口 17](#_Toc16500)

[4.29. 辅助功能设置 17](#_Toc24550)

[4.29.1. 回调接口 17](#_Toc15683)

[4.30. 设置12、24小时制式 17](#_Toc5509)

[4.30.1. 回调接口 17](#_Toc4449)

[4.31. 取得某天详细运动信息 17](#_Toc20181)

[4.31.1. 回调接口 17](#_Toc1683)

[4.31.2. 回调接口 18](#_Toc17784)

[4.32. 发送手机系统使用语言 18](#_Toc29468)

[4.32.1. 回调接口 18](#_Toc5419)

[4.33. 打开关闭断连防丢提醒 18](#_Toc12350)

[4.33.1. 回调接口 18](#_Toc5301)

[4.34. 打开关闭血压，血氧，疲劳度 18](#_Toc23021)

[4.34.1. 回调接口 19](#_Toc14174)

[4.34.2. 数据回调 19](#_Toc18668)

[4.35. 取得某天多运动模式数据 19](#_Toc17170)

[4.35.1. 数据回调 19](#_Toc739)

[4.36. 设置运动目标步数 20](#_Toc18581)

[4.36.1. 回调接口 20](#_Toc30781)

[4.37. 设置心率区间 20](#_Toc30332)

[4.37.1. 回调接口 20](#_Toc26055)

[4.38. 日志功能 20](#_Toc25797)

[4.39. 固件升级功能 21](#_Toc8288)

[4.39.1. 回调接口 21](#_Toc9788)

[4.40. 设置机器串码 21](#_Toc846)

[4.40.1. 回调接口 21](#_Toc5657)

[4.41. 获取机器串码 21](#_Toc15842)

[4.41.1. 回调接口 21](#_Toc19321)

[4.42. 配置读写UUID和接收开关 21](#_Toc25490)

[4.43. 发送蓝牙数据 22](#_Toc10648)

[4.43.1. 接收数据回调接口 22](#_Toc31171)

[4.43.2. 发送数据回调接口 22](#_Toc31486)

[4.44. 获取设备信号强度 22](#_Toc22723)

[4.44.1. 回调接口 22](#_Toc14082)

[4.45. 设置提醒时间 22](#_Toc8655)

[4.45.1. 回调接口 23](#_Toc19518)

[4.46. 设置提醒内容 23](#_Toc23066)

[4.46.1. 回调接口 23](#_Toc24570)

[4.47. 修改设备名称 23](#_Toc31531)

[4.47.1. 回调接口 23](#_Toc12693)

[4.48. 开启/关闭心电测量 23](#_Toc2831)

[4.48.1. 开启/关闭回调 23](#_Toc2666)

[4.48.2. 实时心电数据回调 24](#_Toc9007)

[4.49. 获取离线心电数据 24](#_Toc9681)

[4.49.1. 离线心电次数回调 24](#_Toc17100)

[4.49.2. 离线心电状态回调 24](#_Toc6692)

[4.49.3. 离线心电数据回调 24](#_Toc29299)

[4.50. 设置血压校准参数 24](#_Toc21233)

[4.50.1. 设置血压校准参数回调 25](#_Toc10209)

[4.51. 开启/关闭体温测量 25](#_Toc28101)

[4.51.1. 开启/关闭体温回调 25](#_Toc5677)

[4.51.2. 手环体温检测开启/关闭通知 25](#_Toc13369)

[4.51.3. 实时体温数据回调 25](#_Toc8927)

[4.52. 读取设备屏幕信息 25](#_Toc21434)

[4.52.1. 读取设备屏幕信息返回 26](#_Toc1334)

[4.53. 管理表盘状态 26](#_Toc13645)

[4.53.1. 进入和退出表盘安装界面返回 26](#_Toc25943)

[4.54. 管理自定义壁纸状态 26](#_Toc10874)

[4.54.1. 打开或关闭自定义壁纸返回 26](#_Toc22718)

[4.55. 编辑表盘自定义数据 26](#_Toc7319)

[4.55.1. 编辑表盘自定义数据返回 26](#_Toc19979)

[4.56. 读取表盘自定义数据 26](#_Toc18401)

[4.56.1. 读取表盘自定义数据返回 27](#_Toc24635)

[4.57. 下发联系人校验码 27](#_Toc13386)

[4.57.1. 下发联系人信息 27](#_Toc5304)

[4.58. 下发手机当前音量配置 27](#_Toc459)

[4.59. 执行文件OTA 27](#_Toc25320)

[4.59.1. Ota升级回调 27](#_Toc3136)

[4.60. 根据图片生成壁纸更新的bin文件 28](#_Toc6428)

# 接入须知

* 使用过程中有任何问题可以发邮件反馈：
  + 收件人：[dev@keeprapid.com](mailto:dev@keeprapid.com)
  + 标题：蓝牙设备 SDK 反馈
  + 邮件正文需要提供的信息有：
    - 1. 厂商帐号，即目前的 vid，注意后面命名可能会变动。
    - 2. 设备 mac地址
    - 3. 出现场景，如果有重现的流程及场景的话请告知，越详细的信息能够帮助我 们快速定位和解决问题

# 各平台SDK接口说明

## Android版本说明

### 初始化启动SDK

SDK Demo程序包含的AppID，Secret和VID仅供测试使用，正式的AppID，Secret和VID请邮件联系技术支持[dev@keeprapid.com](mailto:dev@keeprapid.com)。

通过在asset目录下配置JySDK.xml资源文件，示例：

<vid>XXXXXXXXXXXX</vid>

<appid> XXXXXXXXXXXXXXX </appid>

<secret> XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX </secret>

初始化并启动验证SDK，然后实现IServiceCallback接口来获取数据和消息。

## 数据结构说明

### BleClientOption

**private** UserProfile userProfile; //个人信息

**private** DeviceProfile deviceProfile; //设备属性

**private** List<AlarmInfoItem> itemsTimer; //闹钟列表，最大支持5个闹钟

//1-4：普通闹钟

//5：久坐闹钟

**private** List<Weather> weathers; //天气参数

BleClientOption**只传入当前需要设置的对象，其他可传空**

**例如，当前只设置闹钟**

ArrayList<AlarmInfoItem> lAlarmInfo = **new** ArrayList<AlarmInfoItem>();

AlarmInfoItem item = **new** AlarmInfoItem(1, 1, 1, 11, 1, 1, 1, 1, 1, 1, 1, "",**false**);

lAlarmInfo.add(item);

BleClientOption bco = **new** BleClientOption(**null**, **null**, lAlarmInfo);

mService.setOption(bco);

mService.setAlarm();

### UserProfile

**private int stepGoal;** //步数目标

**private int height;** //身高(cm)（步长由身高和步长公式算出，具体请看附件）

**private int weight;** //体重(Kg) 卡路里由距离和体重公式算出，具体请看附件

**private int stride;** //步距(cm) 暂时没用到，保留

**private int unit;** //单位模式，0：公制，1：英制（此公英制也决定手环和app的距离，温度是否使用公英制，公制下，距离显示单位为KM，英制下距离显示单位为MILE）

**private int gender;** //性别，0：女, 1：男

**private int age;** //年龄 最大为127岁

### DeviceProfile

**private** **boolean** enableLight = **false**; //抬手亮屏设置

**private** **boolean** enableVibrate = **false**; 震动设置

**private** **boolean** enableQuite = **false**; 勿扰模式

**private** **int** quiteStartHour = 0; 勿扰模式开始小时

**private** **int** quiteEndHour = 0; 勿扰模式结束小时

**private** **int** quiteStartMin = 0; 勿扰模式开始分钟

**private** **int** quiteEndMin = 0; 勿扰模式结束分钟

20：00-8:00，则20:00-8:00消息提醒，抬手亮屏，马达震动无效，只有闹钟有效）

### AlarmInfoItem

private long alarm\_id;

闹钟编号（取值为0 to 4）可设置5个闹钟

**private** **int** enableType; 闹钟使能设置，0为Disable; 1为Enable睡觉闹钟，2为Enable其它闹钟

**private** **int** hour; 为闹钟时间的小时部分（24小时制）

**private** **int** minute; 为闹钟时间的分钟部分

**private** **int** enableMonday; 星期一闹钟使能。0为Disable; 1为Enable。

**private** **int** enableTuesday; 星期二闹钟使能。0为Disable; 1为Enable。

**private** **int** enableWednesday; 星期三闹钟使能。0为Disable; 1为Enable。

**private** **int** enableThursday; 星期四闹钟使能。0为Disable; 1为Enable。

**private** **int** enableFriday; 星期五闹钟使能。0为Disable; 1为Enable。

**private** **int** enableSaturday; 星期六闹钟使能。0为Disable; 1为Enable。

**private** **int** enableSunday; 星期天闹钟使能。0为Disable; 1为Enable。

**private** **boolean** isSingle = **false**; 单次闹钟标志。0为Disable; 1为Enable，当单次闹钟标志位Enable时，上面的使能均无效，并在闹钟有效一次后，删除闹钟

private String content; 闹钟内容

### Weather

**private** **int** timestamp = 0; 当地时间 如(System.*currentTimeMillis*()/1000)

**private** **int** DaytimeWeather = 0; 白天天气code

**private** **int** Eveningweather = 0; 晚上天气code

**private** **int** lowestT = 0; 最低温度 -127到127

**private** **int** hightestT = 0; 最高温度 -127到127

**private** **int** airQuality = 0; 空气质量 0-6 优，良，轻度污染，中度污染，重度污染，严重污染

**private** **int** PM25 = 0; PM2.52.5，当地没有PM2.5，此值为0

**private** **int** UV = 0; UV

**private** **int** AQI = 0; 空气质量指数 AQI，当地没有AQI，此值为0, PP为最低字节，QQ为最高字节

**private** **int** CurT = 0; 当前温度-127到127

# 接口及回调

## 验证SDK

异步

绑定服务后自动触发，从asset目录下读取参数

异步，结果将在[onAuthSdkResult](#_验证SDK回调)回调接口中返回

## 回调接口

void onAuthSdkResult(int errorCode);

200成功

0失败

## 注册回调

void registerCallback(IServiceCallback cb);

## 注销回调

void unregisterCallback(IServiceCallback cb);

## 获取SDK授权状态

int isAuthrize();

## 设置参数

int setOption(in BleClientOption opt);

根据需求，配合特定方法使用

## 设置扫描模式

**int** setScanMode(**int** scanMode);

scanMode: 在scanDevice之前设置

ScanSettings.SCAM\_MODE\_LOW\_POWER // 低功耗

ScanSettings.SCAM\_MODE\_BALANCED // 中功耗

ScanSettings.SCAM\_MODE\_LOW\_LATENCY // 高功耗

## 扫描蓝牙设备

int scanDevice(boolean enable);

当App的targetSdkVersion>=23时, 请务必打开 手机的定位服务 和 App的定位权限, 否则某些手机会搜索不到蓝牙设备

### 回调接口

void onScanCallback(String deviceName, String deviceMacAddress, int rssi);

deviceName : 设备名称

deviceMacAddress：设备地址

rssi:设备信号

## 连接设备

int connectBt(String name, String addr);

### 设备连接状态改变

void onConnectStateChanged(int state);

0:未连接

1：连接中

2：已连接

## 获取连接状态

boolean isConnectBt();

true:已连接

false:未连接

## 读取连接设备蓝牙地址

String getConnectedDevice();

## 断开设备

void disconnectBt(boolean enable);

enable: true不自动重连

## 设置设备时间

int setDeviceTime();设置设备时间 刚连上时SDK会下发

### 回调接口

在void onSetDeviceTime(int result);返回 1成功 0失败

## 设置用户信息

int setUserInfo(); 设置用户信息 要先设置UserProfile和setOption

### 回调接口

在void onSetUserInfo(int result);返回1成功 0失败

## 获取当前运动数据

getCurSportData();

### 回调接口

在void onGetCurSportData(int type,long timestamp,int step,int distance,int cal,int cursleeptime,int totalrunningtime,int steptime);

参数分别为当前时间戳,以秒为单位 ,当前步数,当前距离(米),当前卡路里(大卡)，当前睡眠时间（秒），当前运动总时间，当前记步时间

有效的返回值根据类型而定

type = 0 当前运动信息

返回有效值为：手环当前时间戳,以秒为单位 ,当前步数,当前距离(米),当前卡路里(大卡)，当前睡眠时间（秒）

Type = 1 当前跑步信息

返回有效值为：当前时间戳,以秒为单位 当前步数 当前运动总时间 当前记步时间

## 手环振动

sendVibrationSignal(int times); 用于手机找手环等

Times ：手环震动次数

### 回调接口

在void onSendVibrationSignal(int result);中返回

异步，当result =1 设置成功，result = 0 ，设置失败

## 手环请求手机操作

onGetDeviceAction (int type)

当type = 1 ,手机震动和发出铃声

当type = 2 手机拍照

当type = 4 ，挂断电话

当type = 5 ，手环向APP请求天气数据

当type = 8 ，接通电话

当type = 16 ，播放或者暂停音乐播放

当type = 32 ，播放下一首

当type = 64 ，播放前一首

当type = 65 ，打开摄像头

当type = 66 ，关闭摄像头

当type = 67 ，请求同步手机时间

当type = 68 ，加大手机音量

当type = 69 ，减少手机音量

### 手环请求天气数据回调

onGetDeviceAction (5) //新增类型type = 5, 手环向APP请求天气数据

APP回复手环请求天气：

先构造天气对象 ，示例：

Weather weather = new Weather((int) (System.currentTimeMillis()/1000), 300, 400, 7, 28, 2, 0, 0, 0, 20);

//参数依次为：时间 白天（300）、晚上天气（400） 、最低最高温（7、28） 空气质量(2)、PM2.5 UV AQI 当前温度（20）

BleClientOption opt3 = new BleClientOption(null, null, null,weather);

再调用int sendWeather();

天气code参考附件

## 设置手环进入拍照模式

setPhontMode(boolean enable);

### 回调接口

在onSetPhontMode回调

## 设置久坐提醒时间

6、setIdleTime(int time,int startHour,int startMinute,int endHour,int endMinute); time 提醒时间长度 （秒） 开始小时、分钟、结束小时、分钟

### 回调接口

回调：void onSetIdleTime(int result);

异步，当result =1 设置成功，result = 0 ，设置失败

## 设置睡眠时间段

中午睡眠时间开始和结束点、晚上睡眠开始和结束点

setSleepTime(int startHourNoon,int startMinuteNoon,int endHourNoon,int endMinuteNoon,int startHourNight,int startMinuteNight,int endHourNight,int endMinuteNight);

116版固件

1.删除中午睡眠时间设定

### 回调接口

onSetSleepTime(int result)回调

## 读取设备电量

int getDeviceBatery();

### 回调接口

void onGetDeviceBatery(int batery,int statu);回调，返回电量和状态

statu ：0没有充电，1正在充电

batery：电量

## 读取手环信息

int getDeviceInfo();

### 回调接口

void onGetDeviceInfo(int version,String deviceMacAddress,String vendorCode,String productCode,int crcResult);//返回手环固件版本（如138），手环mac地址，厂商代码，产品型号，CRC校验结果

crcResult 校验是否合法手环1：成功 0:失败

## 读取手环功能项

int getBandFunction();

### 回调接口

void onGetBandFunction(int result,in boolean[] list);

当result =1 获取成功，result = 0 ，获取失败

List:功能开关列表

返回状态 true为打开 false为未打开

参数依次为 ：

1.是否打开微信运动功能，非0，打开

2.是否打开公英制，非0，打开

3.是否打开天气预报

4.是否打开12小时，24小时开关

5.是否打开防丢选项

6.是否打开血压选项，打开血压即血氧和疲劳度都打开

7.是否打开心率选项

8.是否打开睡眠时间设置选项

9.是否打开日出日落时间显示功能

10.是否有心电功能

11.是否有体温检测功能

12.是否有来电拒接功能

13.手环是否带有GPS功能

14.是否带有步行运动模式

15.是否带有跑步运动模式

16.是否带有骑单车运动模式

17.是否带有登山运动模式

18.是否带有游泳运动模式

19.是否带有心率区间设置功能

20.是否新增10-18类型的信息提醒开关

21.是否带有喝水吃药设置提醒功能

25.是否带有多运动模式

## 设置闹钟

int setAlarm();

需先设置AlarmInfoItem （）闹钟id:0-4,闹钟使能类型0为Disable; 1为Enable睡觉闹钟，2为Enable其它闹钟，小时和分钟，星期日到星期六提醒周期 提醒内容最大支持（18个字母，6个汉字）

### 回调接口

void onSetAlarm(int result);返回结果

异步，当result =1 设置成功，result = 0 ，设置失败

## 特殊命令

int setDeviceMode(int type); 特殊命令

type //1、OTA 固件升级命令：APP向手环发起固件升级的命令，此时，手环会断开连接并进入升级模式，此时手机需要等待25秒，重新连接手环，并开始升级

2、低功耗 3、 重启手环 4、重置数据，删除数据：

由于可能出现设备首先进入了这个模式，有可能出现命令回复不了的，此命令不需等待命令返回

**//重启命令, 目前没发现有什么用，暂时删除：**

## 消息推送

12、setNotify(String id,int type,String title,String content);

id: 编号id，使用者自定义，SDK记录最后一次id,当前id与最后一次id相等时会直接return false，不推送。

type: 推送类型 phone = 0，短信 = 1，微信 = 2，qq = 3，facebook message = 4，skype = 5，twitter = 6，whatsapp = 7, facebook=8, LINE=9

(当4.23.1中20支持) 自定义app=10，钉钉=11，阿里旺旺=12，Instagram=13，Linkedln=14，Snapchat=15，微博=16，Tumblr=17

title: 推送标题

content: 推送内容

### 回调接口

void onSetNotify(int result);

异步，当result =1 设置成功，result = 0 ，设置失败

## 启动\关闭实时心率测量

setHeartRateMode(boolean enable,int time);

启动\关闭实时心率测量模式 ，心率打开的最长时间，秒为单位

### 回调接口

在 void onGetSenserData(int result,long timestamp,int heartrate,int sleepstatu); 回调，返回启动\关闭结果，当前手环时间，心率值，当前睡眠状态0x00-0x03， 0 没有睡眠 3为最好的睡眠质量，1为最差的睡眠质量

注意，成功进入心率模式后，手环会以一定的间隔发送本报文给手机App。

### 结束测试回调

void onSensorStateChange(int type , int state); //type 1:心率 2：血压血氧 , state 1:打开 0：关闭

## 自动测试心率

int setAutoHeartMode(boolean enable, int startHour, int startMinute, int endHour, int endMinute, int interval, int duration);

打开之后, 手环会在后台静默监测, 不会上报. 可通过同步历史心率获取

心率开始时分结束时分，打开心率的时间间隔 以分为单位，持续打开心率的时间 以分为单位， 目前不用，默认间隔15分钟，打开2分钟

### 回调接口

在setAutoHeartMode（result）返回结果

异步，当result =1 设置成功，result = 0 ，设置失败

## 辅助功能设置

setDeviceInfo

需先设置DeviceProfile 允许抬手亮屏、震动、勿扰模式、勿扰模式开始时分、结束时分

### 回调接口

在void onSetDeviceInfo(int result);返回

异步，当result =1 设置成功，result = 0 ，设置失败

## 设置12、24小时制式

setHourFormat() 设置设备时间成功后SDK会下发一次

### 回调接口

在 void onSetHourFormat(int result);返回

异步，当result =1 设置成功，result = 0 ，设置失败

## 取得某天详细运动信息

int getDataByDay(int type,int day);

type 1： 取得当天详细运动信息, 返回步数与睡眠

2：取得当天详细心率数据,

12: 取得当天详细体温数据

day: 时间范围 0-7 （0：当天，1：前一天）

### 回调接口

void onGetDataByDay(int type,long timestamp,int step,int heartrate);

返回数据 type : 1 运动信息 2 睡眠信息 3 心率信息 12 体温数据

当type = 1 时 step为步数 心率为0（无效）

当type = 2 时 step为睡眠质量数据

如果为0表明没有记录或没有睡眠，有记录时的取值范围为（1-100）。100为最好的睡眠质量，1为最差的睡眠质量） 心率为0（无效）

当type = 3 时 step为0（无效） heartrate为心率值

当type = 12 时 step为体表温度（>300, 无效） heartrate为体温温度（>300, 无效）

timestamp: 时间戳（秒）

step: 步数

heartrate: 心率

### 回调接口

onGetDataByDayEnd(long timestamp);

手环数据上传结束标记

## 发送手机系统使用语言

int setLanguage();

调用后会下发一次当前语言字符给手环

### 回调接口

在 void onSetLanguage(int result)返回

## 打开关闭断连防丢提醒

int setAntiLost(boolean enable);

true: 打开防丢提醒，打开后，断开手机响铃震动，手环震动

false: 关闭后，断开手机不响铃，不震动，手环也不震动

手环默认关闭断连提醒

### 回调接口

void onSetAntiLost(int result);

## 打开关闭血压，血氧，疲劳度

int setBloodPressureMode(boolean enable);

true 打开 false 关闭

### 回调接口

void onSetBloodPressureMode(int result);

### 数据回调

//当手环采集到有效心率血压血氧值后，会通过此命令返回数据给app

void onReceiveSensorData(int heartrate, int Systolicpressure, int Diastolicpressure, int Oxygen, int Fatiguevalue);

heartrate：心率，

Systolicpressure:血压，收缩压

Diastolicpressure：血压，舒张压

Oxygen：血氧

Fatiguevalue：疲劳值

## 取得某天多运动模式数据

int getMultipleSportData(int day);

day: 表示具体哪一天的数据，0表示当天，1表示一天前的数据，2表示两天前的数据，依次类推，.....6表示第六天前的数据，总共支持七天；

### 数据回调

void onGetMultipleSportData(int flag,String recorddate,int mode,int value);

flag: 0:无数据或返回失败 1:传输中 2:传输完成

recorddate:时间，格式为yyyy-MM-dd HH:mm:ss

mode：模式，

value:值，在步行，跑步和登山模式模式下，OO，PP高4位，为这一分钟步数，其它模式下，为卡路里，卡路里例子，86，则为0.86千卡

运动模式值和相对应的模式：

1. 不在运动模式下
2. 步行模式
3. 跑步模式
4. 骑行模式
5. 登山模式
6. 游泳模式

步行 卡路里和每分钟步数的换算公式：

Calorie = step \* 36 / 1000 千卡，保留两位小数

跑步

Calorie = step \* 49 / 1000 千卡，保留两位小数

登山

Calorie = step \* 45 / 1000 千卡，保留两位小数

## 设置运动目标步数

int setGoalStep(int step);

step:目标步数

### 回调接口

void onSetGoalStep(int result);

获取手环功能项

## 设置心率区间

int setDeviceHeartRateArea(boolean enable,int max ,int min);

enable，心率区间开关

max:最高心率Min：最低心率

心率区间开关打开后，当前心率低于最低心率，马达震动

当前心率高于最高心率，马达震动

心率区间开关关闭后，心率提醒关闭

### 回调接口

void onSetDeviceHeartRateArea(int result);

## 日志功能

测试用 默认关闭

int openSDKLog(boolean enable, String filePath, String fileName);

ture : 打开SDK日志

false : 关闭SDK日志

打开后，运行时会写入日志到filePath+fileName目录下，用于追踪问题

示例: filePath = “/jy/logs/”, fileName = “ble.log”

## 固件升级功能

**int** getOtaInfo(**boolean** auto);

auto: 是否自动升级

### 回调接口

**void** onGetOtaInfo(**boolean** isUpdate, String info, String path);

isUpdate: 是否需要升级

info: 服务器固件信息, Json格式的字符串, 详细说明见固件配置说明

path: 固件本地下载路径

## 设置机器串码

**int** setDeviceCode(**byte**[] bytes);

bytes: 字节数组, 只取前18位

### 回调接口

**void** onSetDeviceCode(**int** result);

## 获取机器串码

**int** getDeviceCode();

### 回调接口

**void** onGetDeviceCode(**byte**[] bytes);

bytes: 18位字节数组

## 配置读写UUID和接收开关

**int** setUuid(**String**[] uuidRead, **String**[] uuidWrite, **boolean** bOpen);

uuidRead: 接收蓝牙数据的uuid列表, 请在连接手环前配置

uuidWrite: 发送蓝牙数据的uuid列表, 请在连接手环前配置

bOpen: 是否接收原始数据, 打开后回调只有onCharacteristicChanged

## 发送蓝牙数据

**int** writeCharacteristic(**String** uuidWrite, **byte**[] bytes);

uuidWrite: 发送蓝牙数据的uuid, 存在配置uuidWrite列表中有效

bytes: 发送的蓝牙数据

### 接收数据回调接口

**void** onCharacteristicChanged(**String** uuid, **in byte**[] bytes);

uuid: 接收蓝牙数据的uuid, 存在配置uuidRead列表中有效

bytes: 接收的蓝牙数据

### 发送数据回调接口

**void** onCharacteristicWrite(**String** uuid, **in byte**[] bytes, **int** status);

uuid: 发送蓝牙数据的uuid

bytes: 发送的蓝牙数据

status: 发送状态

## 获取设备信号强度

int getDeviceRssi();

### 回调接口

void onGetDeviceRssi(int rssi);

## 设置提醒时间

int setReminder(int time,int startHour,int startMinute,int endHour,int endMinute, int id, int type);

time: 间隔时间(秒), 0表示关闭, 其他任意值表示打开

startHour: 开始时间的小时

startMinute: 开始时间的分钟

endHour: 结束时间的小时

endMinute: 结束时间的分钟

id: 提醒id, 取值为0 - 5，目前总共支持6个提醒

type: 提醒类型 (当4.23.1中21支持)1: 饮水提醒, 2: 吃药提醒

### 回调接口

void onSetReminder(int result);

## 设置提醒内容

int setReminderText(int id, String text);

id: 提醒id

text: 提醒内容(18个英文字符, 约6个汉字)

### 回调接口

void onSetReminderText(int result);

## 修改设备名称

int setDeviceName(String name);

name: 设备名称(11个英文字符)

重启设备生效, 可使用setDeviceMode(3)重启手环

### 回调接口

void onSetDeviceName(int result);

## 开启/关闭心电测量

int setEcgMode(boolean enable, int mode);

enable: true打开, false关闭

mode: 0手环心电, 1胸贴心电

### 开启/关闭回调

void onSetEcgMode(int result, int state);

result: 1操作成功, 0操作失败

state: 0关闭心电, 1打开心电, 2心电功能忙

### 实时心电数据回调

void onGetEcgValue(int state, in int[] values);

state: 0等待手指按下, 1手指按下后离开, 2正常测试

values: 长度12的心电数据数组

## 获取离线心电数据

int getEcgHistory(int timestamp);

timestamp: 时间戳(秒), 获取此时间戳之后的所有数据

### 离线心电次数回调

void onGetEcgHistory(long timestamp, int number);

timestamp: 暂时无用

number: 时间戳之后有几次离线心电测试(0-253)

254心电功能忙稍后查询, 255所有数据传输完成

### 离线心电状态回调

void onGetEcgStartEnd(int id, int state, long timestamp);

id: 第几次心电数据

state: 1开始传输, 255完成传输

timestamp: 该次测量时间戳

### 离线心电数据回调

void onGetEcgHistoryData(int id, in int[] values);

id: 第几次心电数据

values: 长度12的心电数据数组

## 设置血压校准参数

int setBPAdjust(int sbp, int dbp);

sbp: 参考的sbp

dbp: 参考的dbp

### 设置血压校准参数回调

void onSetBPAdjust(int result);

result: 1操作成功, 0操作失败

## 开启/关闭体温测量

int setTemperatureMode(boolean enable);

enable: true打开, false关闭

### 开启/关闭体温回调

void onSetTemperatureMode(int result);

result: 1操作成功, 0操作失败

### 手环体温检测开启/关闭通知

void onTemperatureModeChange(int enable);

enable: 1打开, 0关闭

### 实时体温数据回调

void onGetTemperatureData(int surfaceTemp,int bodyTemp);

surfaceTemp: 体表温度

bodyTemp: 体温温度

## 读取设备屏幕信息

int getDeviceDial();

### 读取设备屏幕信息返回

void onGetDeviceDial(String productType, String productId, int watchWidth, int watchHeight, int unitWidth, int colorMode, int isCustom, int dialId, int reviewWatchWidth, int reviewWatchHeight, int shapeType);

## 管理表盘状态

int setDeviceDialState(int state);

### 进入和退出表盘安装界面返回

void onSetDeviceDialState();

## 管理自定义壁纸状态

int setDeviceWallpaperState(int state);

### 打开或关闭自定义壁纸返回

void onSetDeviceWallpaperState();

## 编辑表盘自定义数据

int editDeviceDialCustom(int p1, int p2, int p3, int p4);

### 编辑表盘自定义数据返回

void onEditDeviceDialCustom();

## 读取表盘自定义数据

int getDeviceDialCustom();

### 读取表盘自定义数据返回

void onGetDeviceDialCustom(int timePos, int timeAboveContent, int timeBelowContent, int fontColorType);

## 下发联系人校验码

int setContactCrc(String contactCrc);

### 下发联系人信息

int setContactInfo(in ContactInfo contactInfo);

## 下发手机当前音量配置

int sendPhoneVolume(int curMusicVolume, int maxMusicVolume, int curVoiceCallVolume, int maxVoiceCallVolume);

## 执行文件OTA

int startFileOta(int mode, String filePath)

mode: 1 固件升级

2 表盘更新

3 壁纸更新

filePath: 进行以上操作需要的bin文件全路径

### Ota升级回调

**void** onGetOtaUpdate(**int** step, **int** progress);

getOtaInfo的参数auto为true时, 会有此回调

startFileOta 会有下面部分数据

step:

**public static int** *STEP\_DOWNLOAD\_FILE* = 1; // 下载文件  
**public static int** *STEP\_ENTRY\_OTA* = 2; // 进入升级模式  
**public static int** *STEP\_CONNECT\_DEVICE* = 3; // 连接升级设备  
**public static int** *STEP\_OTA* = 4; // 正在升级  
**public static int** *STEP\_FINISH* = 5; // 升级完成

progress:

**public static int** *PROGRESS\_START* = 0; // 开始  
**public static int** *PROGRESS\_WAIT* = 50; // 进行中  
**public static int** *PROGRESS\_END* = 100; // 结束  
**public static int** *PROGRESS\_MD5\_ERR* = 101; // 升级固件md5错误

PROGRESS\_MD5\_ERR只在STEP\_DOWNLOAD\_FILE时出现

## 根据图片生成壁纸更新的bin文件

**String translateBmpToBin(  
 String imgFilename,  
 String wallpaperFilePath,  
 String wallpaperFileName,  
 int watchColorType,  
 int watchShapeType,  
 int watchWidth,  
 int watchHeight,  
 int watchUnitWidth,  
 int watchReviewWidth,  
 int watchReviewHeight,  
 int timePos,  
 int timeAboveContent,  
 int timeBelowContent,  
 int fontRed,  
 int fontGreen,  
 int fontBlue);**

返回值：bin文件的完整路径

参数：

imgFilename:壁纸图片文件，支持Jpeg, jpg, png图片格式，图片大小与watchWidth和watchHeight一致

wallpaperFilePath:输出bin文件的子目录

wallpaperFileName:输出斌文件的文件名

watchColorType:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

watchShapeType:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

watchWidth:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

watchHeight:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

watchUnitWidth:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

watchReviewWidth:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

watchReviewHeight:设备屏幕属性，调用getDeviceDial()获取，在onGetDeviceDial回调中返回

timePos:time label display position. 0: right bottom, 1: right top, 2: left bottom, 3: left top

timeAboveContent:above time label content. 0: empty, 1: date, 2: sleep, 3:heart rate, 4: steps

timeBelowContent:under time label content. 0: empty, 1: date, 2: sleep, 3:heart rate, 4: steps

fontRed:font color rgb(red)

fontGreen:font color rgb(green)

fontBlue:font color rgb(blue)

## 读取在线表盘信息

int getDialServerInfo(String fw);

fw: productType + productId

### 读取在线表盘信息回调

**void onNotifyDialJsonContent(String content);**

**Content**：返回的表盘信息，json格式

## 下发AppId信息

int setAppId(String appId);

appId: Unique Id

### 下发AppId信息回调

**void onNotifyClassicBtInfo(int btState, int pareState, String deviceMac, String phoneMac);**

btState：bt state of the watch.

0: Disable

1: Enable

pareState: bt pared state of the watch

0: Disable

1: Enable

deviceMac: bt mac address of the watch

phoneMac: bt mac address of the phone