安卓系列产品开发手册

版本号	说明	修改者	时间
V1.0		黎慈军	

目录

1 简介

本手册主要介绍了科曼掌机各硬件模块的使用方法，电源控制等，其目标读者为使用科曼安卓掌机并需要在上面自行开发应用程序的开发人员。

本手册最新版会在以下网址保持同步更新：https://github.com/Keymantek/AndroidSDK.git

依赖库 Hardware Git 仓库：https://github.com/Keymantek/hardware.git

如有其他方面不清楚部分，可以联系：li@keymantek.com

2 基础库及函数说明

2.1库列表

名称	说明
deviceinfo	提供机器硬件电源控制相关函数
serialport	提供串口类的数据收发
hardware	提供硬件访问相关的基础库

2.2deviceinfo 硬件控制库

2.2.1 依赖项

linq4j.jar, xmlserialize.jar,RXTXcomm.jar, xstream-1.4.7,serialport,hardware

2.2.2 初始化

2.2.2.1实例化对象

```
/**
 * 实例化对象
 *
 * @return
 */
public static DeviceInfo CreateInstance()
```

2.2.3 系统类

2.2.3.1获取序列号

```
/**
 * 获取序列号
 *
 * @return
 */
public    String getSN()
```

2.2.3.2获取资产编号

```
/**
 * 获取资产编号
 *
 * @return
 */
public    String getAssetNo()
```

2.2.3.3获取硬件版本号

```
/**
 * 获取硬件版本号
 *
 * @return
 */
public    int getHardVersion()  或
public    int HardWare()
```

2.2.3.4获取出厂日期

```
/**
 * 获取出厂日期
 *
```

```
 * @return
 */
public    String getDateTimeOfProduction()
```

2.2.3.5是否有摄像头

```
/**
 *  是否有摄像头
 *
 * @return true  表示有此功能  false 无此功能
 */
public    boolean HaveCamera()
```

2.2.3.6是否有蓝牙模块

```
/**
 *  是否有蓝牙模块
 *
 * @return  表示有此功能  false 无此功能
 */
public    boolean HaveBlueTooth()
```

2.2.3.7是否有 GPRS

```
/**
 *  是否有 GPRS
 *
 * @return  表示有此功能  false 无此功能
 */
public    boolean HaveGPRS()
```

2.2.3.8是否有 GPS 功能

```
/**
 *  是否有 GPS 功能
 *
 * @return  表示有此功能  false 无此功能
 */
```

```
public    boolean HaveGPS()
```

2.2.3.9是否有闪光灯

```
/**
 * 是否有闪光灯
 *
 * @return 表示有此功能 false 无此功能
 */
public    boolean HaveLEDFlashLight()
```

2.2.3.10 是否有 WIFI

```
/**
 * 是否有 WIFI
 *
 * @return 表示有此功能 false 无此功能
 */
public    boolean HaveWIFI()
```

2.2.4 扫描

2.2.4.1是否有扫描功能

```
/**
 * 是否有扫描功能
 *
 * @return 表示有此功能 false 无此功能
 */
public    boolean HaveScan()
```

2.2.4.2扫描电源控制

```
/**
 * 扫描电源控制
 *
 * @param bOpen
 */
public    void Scan_Power(boolean bOpen);
```

2.2.4.3扫描出光触发控制

```
/**
 * 扫描出光触发控制
 *
 * @param bOpen
 */
public void Scan_Trig(boolean bOpen);
```

2.2.4.4扫描头类型

```
/**
 * 扫描头类型
 *
 * @return 一维或二维
 */
public   ScanType ScanType();
```

```
enum ScanType {

        // 摘要:
        //     一维
        OneDimension,
        //
        // 摘要:
        //     二维 新大陆
        TwoDimension_XinDaLu,
}
```

2.2.4.5获取扫描串口名

```
/**
 * 获取扫描串口名
 *
 * @return
 */
public   int ScanPortName()
```

2.2.4.6扫描并获取结果

```
/**
 * 获取扫描到的条码 没有扫描到的时候抛出异常
 *
 * @return
 * @throws Exception
 */
public final String Scan_Code() throws Exception
```

2.2.4.7关闭扫描端口

关闭由 Scan_Code 对串口的占用。用于串口复用时的资源释放。

```
/**
 * 关闭端口
 */
public void ScanClose()
```

2.2.5 激光红外

2.2.5.1是否有激光红外功能

```
/**
 * 是否有激光红外功能
 *
 * @return 表示有此功能 false 无此功能
 */
public   boolean HaveFixIR()
```

2.2.5.2激光红外电源控制

```
/**
 * 激光红外电源控制
 *
 * @param bOpen
 */
public    void IR_FixedPoint_Power(boolean bOpen);
```

2.2.5.3获取激光定点红外串口名

```
/**
 * 激光定点红外串口名
 *
 * @return
 */
public    int FixIRPortName()
```

2.2.6 远红外

2.2.6.1是否有远红外模块

```
/**
 * 是否有远红外模块
 *
 * @return 表示有此功能 false 无此功能
 */
public    boolean HaveIR()
```

2.2.6.2远红外电源控制

```
/**
 * 远红外电源控制
 *
 * @param bOpen
 */
public   void IR_Power(boolean bOpen);
```

2.2.6.3获取远红外串口名

```
/**
 *获取远红外串口名
 *
 * @return
 */
public    int IRPortName()
```

2.2.7 近红外

2.2.7.1是否有近红外

```
/**
 * 是否有近红外
 *
 * @return 表示有此功能 false 无此功能
 */
public    boolean HaveNearInfrared()
```

2.2.7.2近红外电源控制

```
/**
 * 近红外电源控制
 *
 * @param bOpen
 */
public    void NearInfraredPower(boolean bOpen);
```

2.2.7.3近红外串口

```
/**
 * 近红外串口
 *
 * @return
 */
public    int NearInfraredPortName()
```

2.2.8 RS232

2.2.8.1是否有 RS232 功能

```
/**
```

```
     * 是否 RS232 功能
     *
     * @return 表示有此功能 false 无此功能
     */
    public    boolean HaveRS232()
```

2.2.8.2RS232 电源控制

```
    /**
     * RS232 电源控制
     *
     * @param bOpen
     *                  True 打开 False 关闭
     */
    public    void RS232_Power(boolean bOpen);
```

2.2.8.3RS232 串口名

```
    /**
     * RS232 串口名
     *
     * @return
     */
    public    int RS232PortName()
```

2.2.9 RS485

2.2.9.1是否有 RS485 模块

```
    /**
     *是否有 RS485 模块
     *
     * @return 表示有此功能 false 无此功能
     */
    public    boolean HaveRS485()
```

2.2.9.2RS485 电源控制

```
    /**
     * RS485 电源控制
```

```
     *
     * @param bOpen
     */
    public    void RS485_Power(boolean bOpen);
```

2.2.9.3RS485 发送接收切换

```
    /**
     * RS485 发送接收切换
     *
     * @param bOpen true 表示切换到发送状态，false 表示切换到接收状态
     */
    public    void RS485_SendEnable(boolean bOpen);
```

2.2.9.4获取 RS485 串口名

```
    /**
     *  获取 RS485 串口名
     *
     * @return
     */
    public    int RS485PortName()
```

2.2.10TTL 电平串口

2.2.10.1 是否有 TTL 电平的串口功能

```
    /**
     *  是否有 TTL 串口功能
     *
     * @return  表示有此功能  false 无此功能
     */
    public    boolean HaveTTLComPort()
```

2.2.10.2 TTL 串口电源控制

```
    /**
     * TTL 电平串口电源控制
     *
```

```
 * @param bOpen
 */
public    void TTLComPort_Power(boolean bOpen);
```

2.2.10.3 TTL 电平串口串口号

```
/**
 * TTL 电平串口串口号
 *
 * @return
 */
public    int TTLComPortPortName()
```

2.2.11 智能卡 7816 接口

2.2.11.1 是否有智能卡 7816 接口

```
/**
 *  是否有智能卡 7816 接口
 *
 * @return  表示有此功能，false 无此功能
 */
public final boolean HaveISO7816()
```

2.2.11.2 智能卡 7816 接口卡检测

```
/**
 *  智能卡 7816 接口卡检测
 *
 * @return
 */
public    boolean ISO7816_CardDetect();
```

2.2.11.3 智能卡 7816 接口电源控制

```
/**
 *  智能卡 7816 接口电源控制
 *
```

```
 * @param bOpen
 */
public    void ISO7816_Power(boolean bOpen);
```

2.2.11.4 智能卡 7816 接口电源复位

```
/**
 *  智能卡 7816 接口电源复位
 */
public    void ISO7816_Reset();
```

2.2.11.5 智能卡 7816 接口串口名

```
/**
 *  智能卡 7816 接口串口名
 *
 * @return
 */
public    int ISO7816PortName()
```

2.2.12安全单元

2.2.12.1 是否有安全单元

```
/**
 *  是否有安全单元
 *
 * @return  表示有此功能  false 无此功能
 */
public    boolean HaveSecUnit()
```

2.2.12.2 安全单元电源控制

```
/**
 *  安全单元电源控制
 *
 * @param bOpen
 */
public    void SecUnit_Power(boolean bOpen);
```

2.2.12.3 安全单元复位

```
/**
 * 安全单元复位
 */
public    void SecUnit_Reset();
```

2.2.12.4 获取安全单元串口名

```
/**
 * 获取安全单元串口名
 *
 * @return
 */
public    int SecUnitPortName()
```

2.2.13TESAM

2.2.13.1 是否有 TESAM 功能

```
/**
 * 是否有 TESAM 功能
 *
 * @return  表示有此功能  false 无此功能
 */
public    boolean HaveTESAM()
```

2.2.13.2 TESAM 电源控制

```
/**
 * TESAM 电源控制
 *
 * @param bOpen
 */
public    void TESAM_Power(boolean bOpen);
```

2.2.13.3 TESAM SPI CS

```
/**
 * TESAM SPI CSs
 *
 * @param bOpen
 */
public    void TESAM_CS(boolean bOpen);
```

2.2.14 ESAM 模块

2.2.14.1 是否有 ESAM 模块

```
/**
 *  是否有 ESAM 模块
 *
 * @return  表示有此功能  false 无此功能
 */
public    boolean HaveESAM()
```

2.2.14.2 ESAM 模块电源控制

```
/**
 * ESAM 模块电源控制
 *
 * @param bOpen
 */
public void ESAM_Power(boolean bOpen);
```

2.2.14.3 ESAM 电源复位

```
/**
 * ESAM 电源复位
 */
public void ESAM_Reset();
```

2.2.14.4 ESAM 模块串口名称

```
/**
 * ESAM 模块串口名称
 *
 * @return
 */
public    int ESAMPortName()
```

2.2.15超高频 915M

2.2.15.1 是否有 RFID915M

```
/**
 *  是否有 RFID915M
 *
 * @return  表示有此功能  false 无此功能
 */
public    boolean HaveRFID915M()
```

2.2.15.2 RFID915M 超高频电源控制

```
/**
 * RFID915M 超高频电源控制
 *
 * @param bOpen
 */
public   void RFID915M_Power(boolean bOpen);
```

2.2.15.3 RFID915M 超高频串口

```
/**
 * RFID915M  超高频串口
 *
 * @return
 */
public    int RFID915MPortName()
```

2.2.16高频 13.56M

2.2.16.1 是否有 RFID13.56M

```
/**
 * 是否有 RFID13.56M
 *
 * @return 表示有此功能 false 无此功能
 */
public    boolean HaveRFID1356()
```

2.2.16.2 RFID13.56M 电源控制

```
/**
 * RFID13.56M 电源控制
 *
 * @param bOpen
 */
public    void RFID1356_Power(boolean bOpen);
```

2.2.16.3 RFID13.56M 串口

```
/**
 * RFID13.56M 串口
 *
 * @return
 */
public    int RFID1356PortName()
```

2.2.17RF433

2.2.17.1 是否有 RF433

```
/**
 * 是否有 RF433
 *
 * @return 表示有此功能 false 无此功能
```

```
 */
public    boolean HaveRF433()
```

2.2.17.2 RF433 电源控制

```
/**
 * RF433 电源控制
 *
 * @param bOpen
 */
public    void RF433_Power(boolean bOpen);
```

2.2.17.3 获取 RF433 串口名

```
/**
 *  获取 RF433 串口名
 *
 * @return
 */
public    int RF433PortName()
```

2.2.18RFID

2.2.18.1 RFID 电源控制

```
/**
 * RFID 电源控制
 *
 * @param bOpen
 */
public    void RFID_Power(boolean bOpen);
```

2.3serialport 串口库

2.3.1 说明

首先实例化串口对象，然后配置串口对应的参数，订阅串口接收到的数据通知，打开串口，在订阅的方法中读取串口接收到的数据，需要发送数据的时候直接调用发送数据方法。

2.3.2 依赖项

convert.jar,event.jar,RXTXcomm.jar ,hardware

2.3.3 控制操作

2.3.3.1实例化

```
    /**
     * 实例化
     *
     * @return 串口实例
     * @throws IOException
     */
    public static SerialPort getInstance() throws IOException
    /**
     * 实例化
     *
     * @param portName
     *            串口名
     * @return
     * @throws IOException
     */
    public static SerialPort getInstance(int portName) throws
IOException
    /**
     * 实例化
     *
     * @param portName
     *            串口名
     * @param baudRate
     *            波特率
     * @return
     * @throws IOException
     */
    public static SerialPort getInstance(int portName, int baudRate)
throws IOException

    /**
     * 实例化
     *
```

```
     * @param portName
     *            串口名
     * @param baudRate
     *            波特率
     * @param parity
     *            校验位
     * @return
     * @throws IOException
     */
    public static SerialPort getInstance(int portName, int baudRate,
Parity parity) throws IOException

    /**
     * 实例化
     *
     * @param portName
     *            串口名
     * @param baudRate
     *            波特率
     * @param parity
     *            校验位
     * @param dataBits
     *            数据位
     * @return
     * @throws IOException
     */
    public static SerialPort getInstance(int portName, int baudRate,
Parity parity, int dataBits) throws IOException

    /**
     * 实例化
     *
     * @param portName
     *            串口名
     * @param baudRate
     *            波特率
     * @param parity
     *            校验位
     * @param dataBits
     *            数据位
     * @param stopBits
     *            停止位
     * @return
     * @throws IOException
```

```
     */
    public static SerialPort getInstance(int portName, int baudRate,
Parity parity, int dataBits, StopBits stopBits)
            throws IOException
```

2.3.3.2打开串口

```
    /**
     * 摘要：打开一个新的串行端口连接。
     *
     *
     * @throws IOException
     *             IOException
     */
    public void Open() throws IOException
```

2.3.3.3关闭串口

```
    /**
     * 摘要：关闭端口连接，将 System.IO.Ports.SerialPort.IsOpen 属性设置
为 false，并释放内部
     * System.IO.Stream 对象。
     *
     * 异常：System.InvalidOperationException：指定的端口未打开。
     */
    public void Close()
```

2.3.3.4串口是否打开

```
    /**
     * @return 是否打开
     */
    public final boolean getIsOpen()
```

2.3.4 串口配置

2.3.4.1获取端口号

```
/**
 * @return the portName 端口号
 */
public final int getPortName()
```

2.3.4.2设置端口号

```
/**
 * 设置端口号
 *
 * @param portName
 *            the portName to set 端口号
 * @throws IOException
 *             IOException
 */
public void setPortName(int portName) throws IOException
```

2.3.4.3获取波特率

```
/**
 * @return the baudRate 波特率
 */
public final int getBaudRate()
```

2.3.4.4设置波特率

```
/**
 * 设置波特率
 *
 * @param baudRate
 *            波特率
 * @throws IOException
 *             the baudRate to set
 */
```

```
public void setBaudRate(int baudRate) throws IOException
```

2.3.4.5获取校验位

```
/**
 * @return the parity 校验位
 */
public final Parity getParity()
```

2.3.4.6设置校验位

```
/**
 * 设置校验位
 *
 * @param parity
 *            parity 校验位
 * @throws IOException
 *             IOException
 */
public void setParity(Parity parity) throws IOException
```

2.3.4.7获取数据位

```
/**
 * @return the dataBits 数据位
 */
public final int getDataBits()
```

2.3.4.8设置数据位

```
/**
 * @param dataBits
 *            the dataBits to set
 * @throws IOException
 *             IOException
 *
 */
public void setDataBits(int dataBits) throws IOException
```

2.3.4.9获取停止位

```
/**
 * @return the stopBits 停止位
 */
public final StopBits getStopBits()
```

2.3.4.10 设置停止位

```
/**
 * 设置停止位
 *
 * @param stopBits
 *            the stopBits to set 停止位
 * @throws IOException
 *             IOException
 */
public void setStopBits(StopBits stopBits) throws IOException
```

2.3.4.11 获取日志消息前缀

```
/**
 * @return the messageTitle 日志消息前缀
 */
public final String getMessageTitle()
```

2.3.4.12 设置日志消息前缀

```
/**
 * 设置日志消息前缀
 *
 * @param messageTitle
 *            日志消息前缀 the messageTitle to set
 */
public final void setMessageTitle(String messageTitle)
```

2.3.5 数据操作

2.3.5.1串口接收到数据通知

```
/**
 * 串口接收到数据通知
 */
public OnDataReceivedListener OnDataReceived;
```

2.3.5.2获取接收到的数据总数

```
/**
 * @return the bytesToRead 接收到的数据总数
 * @throws IOException
 *             ,Exception
 */

public final int getBytesToRead() throws IOException
```

2.3.5.3获取输入流

```
/**
 * @return 获取输入流
 */
public final InputStream getInputStream()
```

2.3.5.4获取输出流

```
/**
 * @return 获取输出流
 */
public final OutputStream getOutputStream()
```

2.3.5.5读取数据

```
/**
 * 读取数据
 *
 * @param buffer
 *            读取数据存放区
 * @return 读取数据长度
 * @throws IOException
 *             IOException
 */
public final int Read(byte[] buffer) throws IOException


/**
 * 摘要: 从 System.IO.Ports.SerialPort 输入缓冲区读取一些字节并将那些字节写入字节数组中指定的偏移量处。
 *
 * @param buffer
 *            将输入写入到其中的字节数组。
 * @param offset
 *            缓冲区数组中开始写入的偏移量。
 * @param count
 *            要读取的字节数。
 * @return 读取的字节数。
 * @throws IOException
 *             IOException
 */
public int Read(byte[] buffer, int offset, int count) throws IOException
```

2.3.5.6写数据

```
/**
 * 使用缓冲区的数据将指定数量的字节写入串行端口。
 *
 * @param oneByte
 *            包含要写入端口的数据的字节。
 * @throws IOException
 */
public final void Write(byte oneByte) throws IOException
```

```
    /**
     * 使用缓冲区的数据将指定数量的字节写入串行端口。
     *
     * @param buffer
     *            包含要写入端口的数据的字节数组。
     * @throws IOException
     */
    public final void Write(byte[] buffer) throws IOException

    /**
     * 使用缓冲区的数据将指定数量的字节写入串行端口。
     *
     * @param buffer
     *            包含要写入端口的数据的字节数组。
     * @param offset
     *            buffer 参数中从零开始的字节偏移量，从此处开始将字节复制
到端口。
     * @throws IOException
     */
    public final void Write(byte[] buffer, int offset) throws
IOException

    /**
     * 使用缓冲区的数据将指定数量的字节写入串行端口。
     *
     * @param buffer
     *            包含要写入端口的数据的字节数组。
     * @param offset
     *            buffer 参数中从零开始的字节偏移量，从此处开始将字节复制
到端口。
     * @param count
     *            要写入的字节数。
     * @throws IOException
     */

    public final void Write(byte[] buffer, int offset, int count)
throws IOException
```

2.3.6 日志消息订阅

```
/**
 * @return the logMessage 日志消息订阅
 */
public final MessageEvent getLogMessage()
```