# Smartpos flutter插件接口使用文档说明

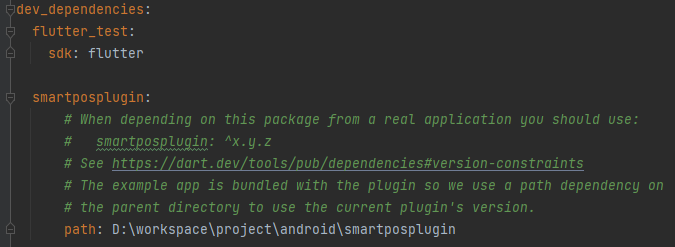
|  |  |  |  |
| --- | --- | --- | --- |
| 项目编号 |  | | |
| 文件状态 | [√]草稿 [] 正式发布 [ ]正在修改 | | |
| 当前版本 | V1.0 | | |
| 拟 制 | 杨瑞贤 | 日期 | 2023/08/07 |
| 审 核 |  | 日期 |  |
| 批 准 |  | 日期 |  |
|  |  |  |  |

**修订历史记录**

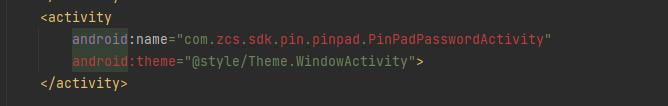
**A** - 增 加 **M** - 修 订 **D** - 删 除

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 变更版本号 | 日期 | 变更类型  （**A\*M\*D**） | 修改人 | 摘 要 | 备注 |
| V1.0 |  |  | 杨瑞贤 |  |  |

使用步骤，将插件导入工程，在pubspec.yaml文件配置插件的路径，如下图所示：



安卓工程配置SD卡读写权限，注册密码键盘组件



插件接口的使用,参考\smartposplugin\example\lib\main.dart的调用实现

**插件接口实例化**

Smartposplugin smartpos =Smartposplugin.getInstance();

**插件相关接口**

|  |  |
| --- | --- |
| 接口名字 | 功能说明 |
| Future<String> sysInit() | 初始化接口，在调用其他接口之前必现先调用此接口，调用成功返回‘0’，否则失败。 |
| Future<String> getPid() | 获取设备的序列号 |
| Future<String>getFirmwareVer() | 获取设备的固件版本号 |
| Future<String>iccRest(int slot) | IC卡复位接口，slot=0 |
| Future<String> iccExchangeAPDU(int slot,Uint8List bytes) | IC卡APDU交换指令接口 slot=1， bytes =apdu |
| Future<String>iccPowerDown(int slot) | IC卡下电指令 |
| Future<String>iccGetStatus(int slot) | 获取IC卡槽的状态 slot=0 |
| Future<String> rfRest() | 非接IC卡复位接口 |
| Future<String> rfSearchCard(int cardType) | 寻非接卡 ，cardType =0x01 A卡，cardType =0x02 B卡，可组合 cardType=0x01|0x02 寻 AB卡 |
| Future<String> rfExchangeAPDU(Uint8List sendApdu) | 非接APDU交换指令 |
| Future<String> rfPowerDown()  Future<String> magClose() | 非接卡下电接口  打开读磁卡接口 |
| Future<String> magIfBrush() | 检测是否刷磁卡 |
| Future<String> magClearData() | 清除磁卡缓存数据 |
| Future<String> magReadData() | 读取磁卡数据，检测到刷卡动作之后调用此接口 |
| Future<String> magClose() | 磁卡关闭 |
| Future<String>prnBitmap(Uint8List pucBitmap) | 打印图片接口，pucBitmap 传入的是bmp字节数组 |
| Future<String>prnStr(Uint8List pucStr) | 打印字符串接口，传入ASCII码字符串 |
| Future<String>getPrnStatus() | 获取打印机的状态，缺纸返回“-1403” |
| Future<String>prntEpson(Uint8List pucStr) | 传入ESC/POS指令打印 |
| Future<String>addCapk(EmvCapk capk) | 添加EMV参数-添加公钥 |
| Future<String> addApp(EmvApp aid) | 添加EMV参数-添加AID |
| Future<String> delAllCapk() | 删除所有的公钥 |
| Future<String> delAllAid() | 删除所有的AID |
| Future<String> separateOnlineResp(String pucStr) | 传入银行后台返回的联机响应数据 |
| Future<String> pinPadUpMastKey(int keynum,String ckeyStr) | 下载主密钥 |
| Future<String> pinPadUpWorkKey(int keynum,String pin\_keystr, String mac\_keystr,String tdk\_keystr) | 下载用于加密pin mac data相关的工作密钥 |
| Future<String>emvTrans(int cardType) | EMV的执行流程，cardType =0执行接触式IC卡，cardType=1执行非接触IC卡 |
| Future<String> getEmvData() | 获取Emv 55域相关数据，需要在执行EmvTans 函数完成之后调用 |

注意 如果需要用到EMV内核，需要实现以下接口用于接收内核回调的函数，具体参考demo

static const platform = MethodChannel('smartposplugin');

@override

void initState() {

// TODO: implement initState

super.initState();

platform.setMethodCallHandler((call) async {

if (call.method == 'onSelApp') {

platform.invokeMapMethod('messageCall','onSelApp');

String result = call.arguments;

print(result);

}else if(call.method == 'onConfirmCardNo'){

platform.invokeMapMethod('messageCall','onConfirmCardNo');

String result = call.arguments;

print(result);

}else if(call.method == 'onInputPIN'){

platform.invokeMapMethod('messageCall','onInputPIN');

String result = call.arguments;

print(result);

}else if(call.method == 'onCertVerify'){

platform.invokeMapMethod('messageCall','onCertVerify');

String result = call.arguments;

print(result);

}

else if(call.method == 'onlineProc'){

String result = call.arguments;

print(result);

String respcode='Z1';

String issuerResp='8D238A029F02069F03069F1A0295055F2A029A039C019F3704';

String OnlineResp='$respcode;$issuerResp';

platform.invokeMapMethod('messageCall','onlineProc:$OnlineResp');

}

else{

platform.invokeMapMethod('messageCall','');

}

});