

# QPOS SDK

## Integration Guide

Document Version: <3.6>

Version	Date	Author	Description
1.0	2013.07.12	Longhui.xiao	Chinese Version
2.0	2013.08.10	William.tang	English Version
2.1	2013.09.20	William.tang	Support Audio and BT mode.
2.2	2013.09.21	Xu.Wang	Added QPOS Reader Series
2.3	2014.03.11	Longhui Xiao	Added icc apdu
2.4	2014.03.26	Longhui Xiao	Added set the sleep time interface
2.5	2014.04.08	Longhui Xiao	Added update work key interface
2.6	2015.02.11	Yanhao Lu	Perfect the documents
2.7	2015.09.07	Longhui Xiao	Added resetPosStatus interface. Perfect the documents
2.8	2016.12.27	Qianmeng Chen	Added doSetManagemgentKey and doSetBuzzerOperation interface
2.9	2017.01.04	Qianmeng Chen	Added doUpdateIPEKOperation interface and update the doc
3.0	2017.01.11	Qianmeng Chen	Added updateEmvAPP and updateEmvCAPK interface
3.1	2017.03.03	Qianmeng Chen	Added scanQpos2Mode and stopQpos2Mode interface
3.2	2022.04.25	Qianmeng Chen	Added the api operateLEDByType and playBuzzerByType
3.3	2022.09.19	Shasha Lv	Modify the description for onRequestGenerateTransportKey and onReturnUpdateIPEKResult
3.4	2024.01.04	Shasha Lv	Added the api sendCvmPin and getEncryptData()
3.5	2024.01.05	Shasha Lv	Added the api which online file included



3.6	2025.5.21	Shasha Lv	Modify the initialization method and listener methods
-----	-----------	-----------	---

### *Copyright notification*

Reproduction in whole or in part is prohibited without the prior written consent of the copyright owner. The information presented in this document does not form part of any quotation or contract, is believed to be accurate and reliable and may be changed without notice. No liability will be accepted by the publisher for any consequence of its use. Publication thereof does not convey nor imply any license under patent - or other industrial or intellectual property rights.

© Dspread technology CO.,LTD

All rights are reserved.

## Table

<b>1</b>	<b>INTRODUCTION .....</b>	<b>8</b>
<b>1.1</b>	<b>SUMMARY .....</b>	<b>8</b>
1.2	CONNECTION MAP FOR QPOS .....	9
1.3	CONNECTION MAP FOR QPOS READER.....	9
1.4	PURPOSE AND SCOPE .....	10
1.5	GLOSSARY AND DEFINITIONS .....	10
1.6	TRANSACTION KEY .....	10
1.7	EMVCo TERMINAL TYPE APPROVAL .....	10
1.8	MESSAGE FORMAT .....	11
1.9	EMV STANDARD TAGS .....	11
1.10	PROPRIETARY TAGS DESCRIPTION .....	11
1.11	BLUETOOTH MODE.....	11
1.11.1	HOW TO GET QPOS/SPOS BLUETOOTH ID .....	11
1.11.2	HOW TO CONNECT QPOS/SPOS THROUGH BLUETOOTH .....	12
1.12	MANAGEMENT KEYS FOR QPOS .....	12
1.13	FIRMWARE UPGRADE FOR QPOS.....	13
<b>2</b>	<b>ANDROID SDK API.....</b>	<b>13</b>
<b>2.1</b>	<b>SYSTEM REQUIREMENT.....</b>	<b>13</b>
2.1.1	DEVELOPMENT ENVIRONMENT .....	13
2.1.2	OS VERSION REQUIREMENT .....	13
<b>2.2</b>	<b>SDK FRAMEWORK.....</b>	<b>14</b>
<b>2.3</b>	<b>ANDROID PERMISSION .....</b>	<b>15</b>
<b>2.4</b>	<b>CLASSIFICATION PRINCIPLE IN SDK METHODS.....</b>	<b>15</b>
2.4.1	INIT METHODS .....	15
2.4.2	INTERACTIVE METHODS .....	16
2.4.3	LISTENER METHODS .....	18
<b>2.5</b>	<b>TRANSACTION FLOW .....</b>	<b>21</b>
2.5.1	EMV ICC TRANSACTION FLOW FOR QPOS.....	21
2.5.2	EMV ICC TRANSACTION FLOW FOR QPOS READER.....	22
2.5.3	EMV ICC TRANSACTION DESCRIPTION .....	23
2.5.4	MAGNETIC TRANSACTION FLOW.....	26
2.5.5	MAGNETIC TRANSACTION DESCRIPTION.....	26
<b>2.6</b>	<b>API METHODS REFERENCE.....</b>	<b>28</b>
2.6.1	GETSDKVERSION .....	28
2.6.2	RESETQPOS.....	28
2.6.3	SCANQPos2Mode.....	28
2.6.4	STOPQPos2Mode .....	29

2.6.5	OPENAUDIO .....	29
2.6.6	CLOSEAUDIO .....	29
2.6.7	CONNECTBT .....	30
2.6.8	CONNECTBLUETOOTHDEVICE .....	30
2.6.9	DISCONNECTBT .....	30
2.6.10	ISQPOSPRESENT.....	31
2.6.11	GETQPOSID.....	31
2.6.12	GETQPOSINFO.....	31
2.6.13	SETAMOUNT .....	32
2.6.14	CANCELSETAMOUNT .....	32
2.6.15	DOTRADE.....	33
2.6.16	DOCHECKCARD .....	33
2.6.17	S DOEMVAPP .....	34
2.6.18	CANCELSETAMOUNT .....	34
2.6.19	SETAMOUNT .....	34
2.6.20	SELECTEMVAPP.....	35
2.6.21	CANCELSELECTEMVAPP .....	35
2.6.22	FINALCONFIRM.....	35
2.6.23	SENDONLINEPROCESSRESULT.....	36
2.6.24	ISSERVERCONNECTED .....	36
2.6.25	SENDTIME.....	36
2.6.26	SETAMOUNTICON .....	37
2.6.27	GETPIN.....	37
2.6.28	POWERONICC .....	38
2.6.29	POWEROFFICC .....	38
2.6.30	SENDAPDU.....	38
2.6.31	SETPOSSLEEPTIME .....	38
2.6.32	UPDATEEMVCAPK .....	39
2.6.33	READEMVAPPCONFIG .....	39
2.6.34	READEMVCAPKCONFIG.....	39
2.6.35	UDPATEWORKKEY .....	40
2.6.36	MACKEYENCRYPT .....	40
2.6.37	ISQPOSPRESENT.....	41
2.6.38	SETMASTERKEY .....	41
2.6.39	CALCMACSINGLE .....	41
2.6.40	CALCMACDOUBLE .....	42
2.6.41	CALCMACSINGLENOCHECK .....	42
2.6.42	CALCMACDOUBLENOCHECK.....	42
2.6.43	DOWNLOADRSAPUBLICKEY .....	43
2.6.44	UPDATEMASTERKEYRANDOM .....	43
2.6.45	UPDATEMASTERKEY .....	44
2.6.46	PINKEY_TDES .....	45
2.6.47	PINKEY_TDESNOCHECK.....	45
2.6.48	SETSYSTEMDATETIME .....	45

2.6.49	SETMERCHANTID.....	46
2.6.50	SETTERMINALID.....	46
2.6.51	GETMAGNETICTRACKPLAINTEXT .....	46
2.6.52	GETCARDNO.....	47
2.6.53	GETICCCARDNO .....	47
2.6.54	POWEROFFNFC .....	47
2.6.55	SENDAPDUBYNFC .....	48
2.6.56	POWERONNFC .....	48
2.6.57	CBC_MAC .....	48
2.6.58	CBC_MACNOCHECK .....	49
2.6.59	INQUIRECQAMOUNT.....	49
2.6.60	ISIDLE .....	50
2.6.61	ANLYSEMVICCDATA .....	50
2.6.62	VIPOSBATCHSENDAPDU .....	50
2.6.63	SYNVIPOSBATCHSENDAPDU.....	50
2.6.64	ANLYSEMVICCDATA_QF .....	51
2.6.65	ICCCASHBACK.....	51
2.6.66	SETPOSPRESENT .....	52
2.6.67	SETCARDTRADEMODE .....	52
2.6.68	SETPINPADFLAG.....	52
2.6.69	QPOSSTATUS.....	53
2.6.70	READBUSINESSCARD.....	53
2.6.71	WRITEBUSINESSCARD .....	53
2.6.72	SYNCREADBUSINESSCARD.....	54
2.6.73	SYNCWRITEBUSINESSCARD .....	55
2.6.74	CONFIRAMOUNT .....	55
2.6.75	SETAMOUNT .....	55
2.6.76	GETPIN.....	56
2.6.77	DOSETMANAGEMENTKEY .....	56
2.6.78	DOSETBUZZEROPERATION .....	57
2.6.79	DOSETBUZZEROPERATION .....	57
2.6.80	DOUPDATEIPEKOPERATION .....	57
2.6.81	UPDATEIPEKOPERATIONBYKEYTYPE.....	58
2.6.82	UPDATEEMVAPP .....	58
2.6.83	UPDATEEMVCAPK.....	59
2.6.84	GETUPDATEKEY .....	59
2.6.85	SETSLEEPMODETIME .....	60
2.6.86	GENERATETRANSPORTKEY .....	60
2.6.87	READGASCARD .....	61
2.6.88	WRITEGASCARD .....	61
2.6.89	UPDATEIPEKBYTRANSPORTKEY.....	62
2.6.90	UPDATEWORKKEYBYTRANSPORTKEY .....	62
2.6.91	SENDCCVV .....	63
2.6.92	GETENCRYPTEDDATABLOCK .....	64

2.6.93	DISPLAYQRCODE .....	64
2.6.94	OPERATELEDBYTYPE.....	65
2.6.95	PLAYBUZZERBYTYPE.....	66
2.6.96	SENDNFCPROCESSRESULT .....	66
2.6.97	GETICCTAG.....	67
2.6.98	GETENCRYPTEDTRACK2DATA .....	67
2.6.99	LCDSHOWCUSTOMDISPLAY .....	68
2.6.100	SETCUSTOMLOGODISPLAY .....	68
2.6.101	GETENCRYPTDATA.....	69
2.6.102	SENDCVMPIN.....	69
2.6.103	INITLISTENER .....	70
2.6.104	OPENUSB .....	70
2.6.105	OPENUART .....	71
2.6.106	BYPASSPIN.....	71
2.6.107	CANCELPIN .....	71
2.6.108	PINMAPSYNC.....	72
2.6.109	UPDATEKEYBYTR_31.....	72
2.6.110	UPDATEEMVCONFIGBYXML.....	72
2.6.111	UPDATEPOSFIRMWARE.....	73
2.6.112	POLLONMIFARECARD .....	73
2.6.113	AUTHENTICATEMIFARECARD.....	74
2.6.114	OPERATEMIFARECARDATA .....	74
2.6.115	READMIFARECARD.....	75
2.6.116	WRITEMIFARECARD .....	75
2.6.117	FINISHMIFARECARD .....	75
2.6.118	FASTREADMIFARECARDATA.....	76
2.6.119	GETDEVICEECCPUBLICKEY .....	76
2.6.120	UPDATESERVERECCPUBLICKEY .....	77
2.6.121	UPDATEKEYBYTR_31.....	77
2.6.122	CALCULATEMACWITHKEY .....	78
2.6.123	GETENCRYPTEDDATAWITHTAG .....	78
2.6.124	CBC_MAC_CN .....	79
2.6.125	CBC_MAC_CNALL.....	79
<b>2.7</b>	<b>DELEGATE METHODS REFERENCE.....</b>	<b>80</b>
2.7.1	ONREQUESTWAITINGUSER .....	80
2.7.2	ONQPOSIDRESULT .....	80
2.7.3	ONQPOSINFORESULT .....	80
2.7.4	ONDOTRADERESULT .....	81
2.7.5	ONREQUESTSETAMOUNT .....	81
2.7.6	ONREQUESTSELECTEMVAPP.....	82
2.7.7	ONREQUESTISSERVERCONNECTED .....	82
2.7.8	ONREQUESTFINALCONFIRM.....	82
2.7.9	ONREQUESTONLINEPROCESS .....	83
2.7.10	ONREQUESTTIME.....	83

2.7.11	ONREQUESTTRANSACTIONRESULT .....	83
2.7.12	ONREQUESTTRANSACTIONLOG .....	84
2.7.13	ONREQUESTBATCHDATA .....	84
2.7.14	ONREQUESTQPOSCONNECTED .....	84
2.7.15	ONREQUESTQPOSDISCONNECTED .....	84
2.7.16	ONREQUESTNOQPOSDETECTED .....	85
2.7.17	ONERROR .....	85
2.7.18	ONREQUESTDISPLAY .....	85
2.7.19	ONREQUESTUPDATEWORKKEYRESULT .....	86
2.7.20	ONGETCARDNORESULT .....	86
2.7.21	ONRETURNREVERSALDATA .....	86
2.7.22	ONRETURNGETPINRESULT .....	86
2.7.23	ONRETURNPOWERONICCRESULT .....	87
2.7.24	ONRETURNPOWEROFFICCRESULT .....	87
2.7.25	ONRETURNAPDURESULT .....	88
2.7.26	ONRETURNSETSLEEPTIMERESULT .....	88
2.7.27	ONREQUESTCALCULATEMAC .....	88
2.7.28	ONRETURNCUSTOMCONFIGRESULT .....	89
2.7.29	ONRETURNSETMASTERKEYRESULT .....	89
2.7.30	ONRETURNBATCHSENDAPDURESULT .....	89
2.7.31	ONRETURNICCCASHBACK .....	90
2.7.32	ONUPDATEPOSFIRMWARERESULT .....	90
2.7.33	ONRETURNDOWNLOADRSAPUBLICKEY .....	90
2.7.34	ONPINKEY_TDES_RESULT .....	91
2.7.35	ONUPDATEMASTERKEYRESULT .....	91
2.7.36	ONEMVICCEXCEPTIONDATA .....	91
2.7.37	ONSETMANAGEMENTKEY .....	92
2.7.38	ONSETBUZZERRESULT .....	92
2.7.39	ONRETURNUPDATEIPEKRESULT .....	92
2.7.40	ONRETURNUPDATEEMVRESULT .....	92
2.7.41	ONRETURNUPDATEEMVRIDRESULT .....	93
2.7.42	ONREQUESTDEVICESCANSUCCESS .....	93
2.7.43	ONDEVICEFOUND .....	93
2.7.44	ONREQUESTUPDATEKEY .....	94
2.7.45	ONSETSLEEPMODETIME .....	94
2.7.46	ONREQUESTGENERATETRANSPORTKEY .....	94
2.7.47	ONREADGASCARDRESULT .....	95
2.7.48	ONWRITEGASCARDRESULT .....	95
2.7.49	ONRETURNOPERATELEDBYTYPERESULT .....	95
2.7.50	ONRETURNPLAYBUZZERBYTYPERESULT .....	96
2.7.51	ONRETURNDISPLAYQRCODERESULT .....	96
2.7.52	ONRETURNNFCBATCHDATA .....	96
2.7.53	ONLCDSHOWCUSTOMDISPLAY .....	97
2.7.54	ONSETCUSTOMLOGODISPLAY .....	97

2.7.55	ONSEARCHMIFARECARDRESULT .....	97
2.7.56	ONVERIFYMIFARECARDRESULT .....	98
2.7.57	ONOPERATEMIFARECARDRESULT .....	98
2.7.58	ONREADMIFARECARDRESULT .....	98
2.7.59	ONWRITEMIFARECARDRESULT .....	99
2.7.60	ONFINISHMIFARECARDRESULT .....	99
2.7.61	GETMIFAREFASTREADDATA .....	100
2.7.62	ONRETURNUPDATEKEYBYTR_31RESULT .....	100
2.7.63	ONQPOSGETDEVICEECCPUBLICKEYRESULT.....	100
2.7.64	ONQPOSUPDATESERVERECCPUBLICKEYRESULT.....	101
2.7.65	ONRETURNUPDATEKEYBYTR_31RESULT.....	101

## 1 Introduction

### 1.1 Summary

QPOS is a mobile payment card reader device with pin pad that works with mobile devices such as smart phone. It provides merchants and consumers a safe and convenient way to make mobile payments. QPOS can communicate with the mobile device through many methods, such as: audio jack, Bluetooth and USB cable.

QPOS Reader is another mobile payment card reader device without pin pad. QPOS Reader can communicate with the mobile device through audio jack or USB cable.

SPOS is a yet another mobile payment card reader device with pin pad and touch screen, SPOS can communicate with the mobile device through many methods, such as: audio jack, Bluetooth and USB cable. The touch screen can be used to capture the signature of consumer in an electronic way.

QPOS share a lot in common, the SDK API is almost same for QPOS. In the following chapters.

Features:

- Ensure secure transactions: integrated keyboard and multiple encryption algorithms to ensure secure transactions.
- Accept all types of bank card: supports magnetic stripe card, contact EMV IC card and contactless EMV IC card.
- Adapts to more smart devices through audio jack: Supports over 2,000 smart devices through audio jack.
- Fulfils global standards: EMV L1&L2, PCI PTS and more.
- Supports all types of mobile systems: Such as iOS, Android, Windows phone and PC OS.

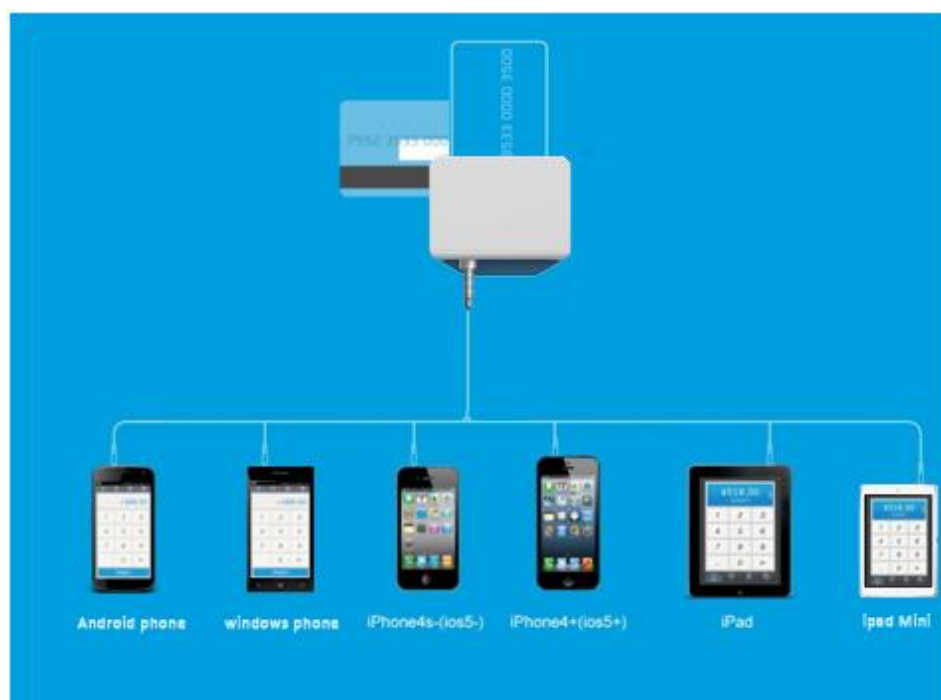


This document is to help readers to integrate QPOS android SDK into their mobile payment APPs.

## 1.2 Connection map For QPOS



## 1.3 Connection map For QPOS Reader



## 1.4 Purpose and Scope

This document is to describe the APIs of QPOS android SDK . The goal of the QPOS android SDK is to communicate between the smart device and QPOS.

The readers of the document are those who plans to use QPOS android SDK in their application.

## 1.5 Glossary and Definitions

DUKPT	DERIVED UNIQUE KEY PER TRANSACTION.
PK_Q	QPOS Public Key
SK_Q	QPOS Private Key
PK_P	Payment Operator Public Key
SK_P	Payment Operator Private Key
PK_T	Terminal Manufacturer Public Key
SK_T	Terminal Manufacturer Private Key
KSN	Key Serial Number
BDK	Base Derivation Key
IPEK	Initial PIN Encryption Key
DATA-key	The data key to be generated by KSN and IPEK or by KSN and BDK
PIN-key	The PIN key to be generated by KSN and IPEK or by KSN and BDK

## 1.6 Transaction Key

Unless otherwise specified, Triple DES encryption with EBC and DUKPT key management are assumed. DUKPT is specified in ANSI X9.24 part 1.

Refer to

[http://en.wikipedia.org/wiki/DUKPT#Key\\_Register\\_.2832\\_hexadecimal\\_digits.29](http://en.wikipedia.org/wiki/DUKPT#Key_Register_.2832_hexadecimal_digits.29)

In the Demo, the default transaction keys refer to the following table.

Key Name	Default	Length(Bytes)
KSN	00000332100300e00001	20
BDK	0123456789ABCDEFFEDCBA9876543210	32

## 1.7 EMVCo Terminal Type Approval

EMVCo has approved QPOS application EMVCo type for Terminal level 2. QPOS application is based on the requirements stated in the EMV 4.3 specification.

## 1.8 Message Format

Messages within data communication protocols between the mobile payment application and QPOS EMV kernel are encoded as a BER-TLV (Basic Encoding Rules-Tag-Length-Value) which is defined in [EMV 4.3 book3 Annex B](#).

## 1.9 EMV Standard Tags

EMV Standard Tags are defined in [EMV 4.3 book3 Annex A](#).

### 1.10 Proprietary Tags Description

Tag	Description	Length(Bytes)	Key	Algorithm
0xC0	KSN of online message	10	No	No
0xC3	KSN of Batch	10	No	No
0xC4	Masked PAN	0-10	No	No
0xC5	Batch message <sup>1</sup>	Var	DATA-key	Triple-Des
0xC2	Online message <sup>3</sup>	Var	DATA-key	Triple-Des
0x70	Online EMV data message <sup>2</sup>	Var	No	No

Note:

1. The **Batch message** is the Triple-Des encrypted result with Data-key. For using, first Triple-Des decrypted the **batch message** with **DATA-key**, the decrypted result is encoded as a BER-TLV which is defined in [EMV 4.3 book3 Annex B](#).

2. The **Online EMV data message** is encoded as a BER-TLV which is defined in [EMV 4.3 book3 Annex B](#).

3. The **Online message** is the Triple-Des encrypted result with Data-key. For using, first Triple-Des decrypted the **Online message** with **DATA-key**, the decrypted result is encoded as a BER-TLV which is defined in [EMV 4.3 book3 Annex B](#) and [Proprietary Tags](#).

## 1.11 Bluetooth Mode

### 1.11.1 How to get QPOS/SPOS Bluetooth ID

QPOS Bluetooth ID is combined by 'QPOS' strings and the last-10 numbers of the label.

For example:

QPOS\_ID := 12070002000**0200100151**

QPOS\_BT\_ID := **QPOS0200100151**



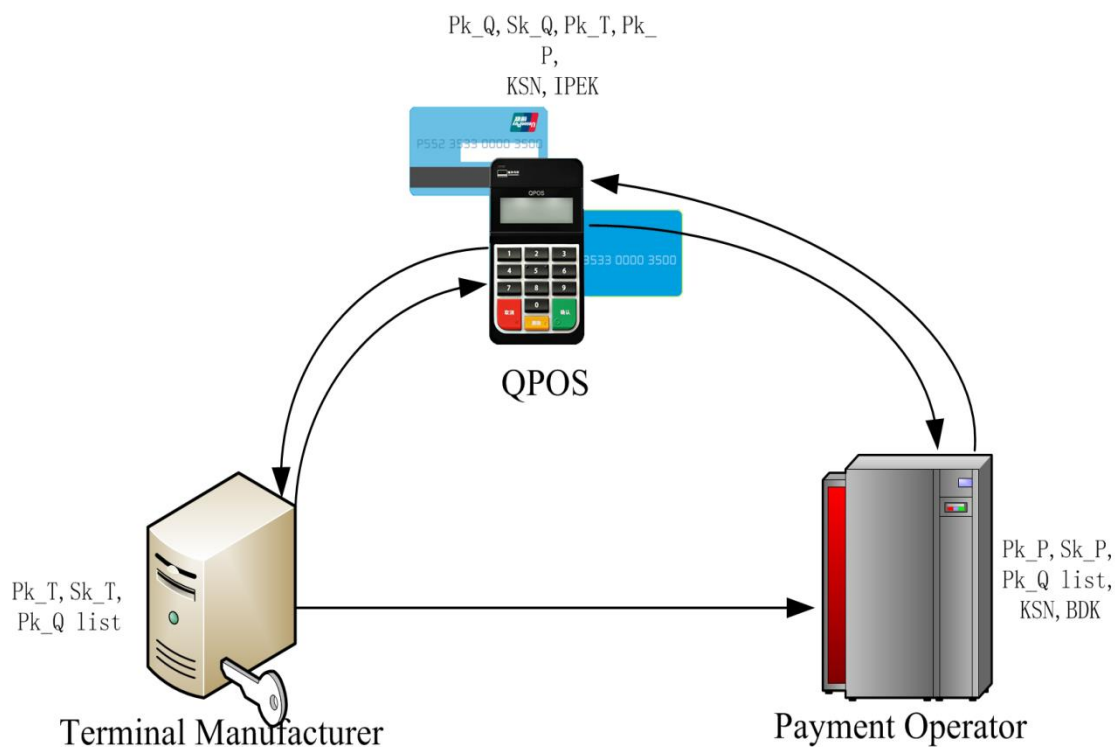
(The label is stuck on QPOS back cover)

SPOS Bluetooth ID is similar to QPOS' but starting with SPOS.

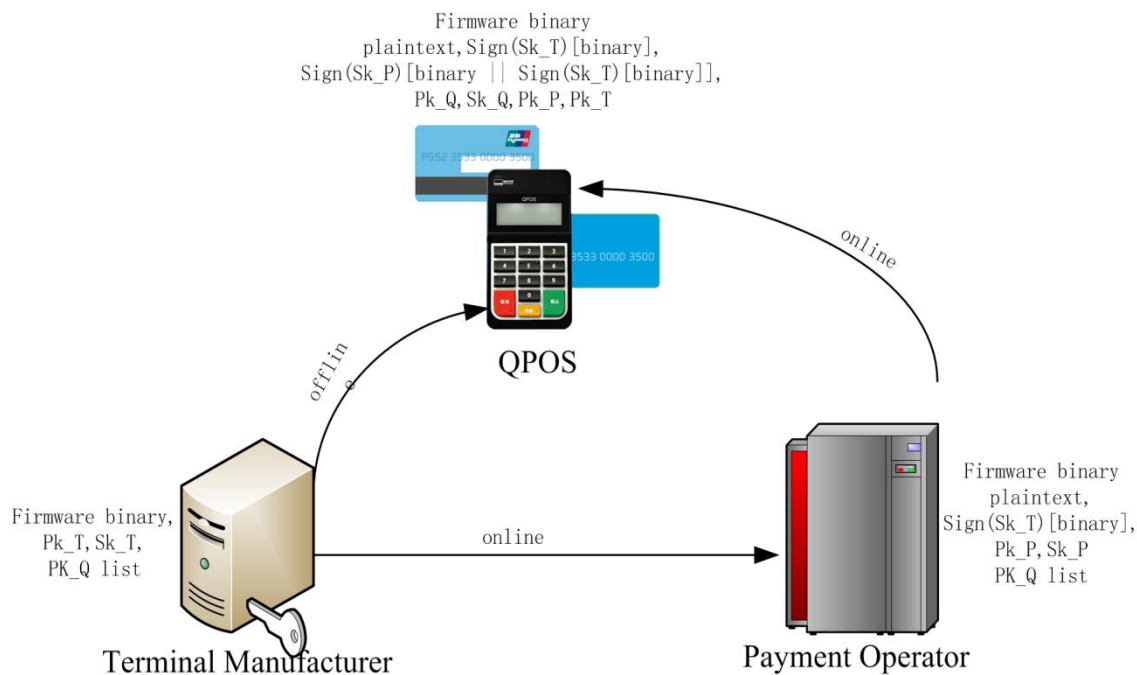
### 1.11.2 How to connect QPOS/SPOS through Bluetooth

ITEM	DEFAULT
CONNECT METHOD	Manual
PASSWORD	1234

### 1.12 Management Keys For QPOS



## 1.13 Firmware Upgrade For QPOS



## 2 Android SDK API

### 2.1 System Requirement

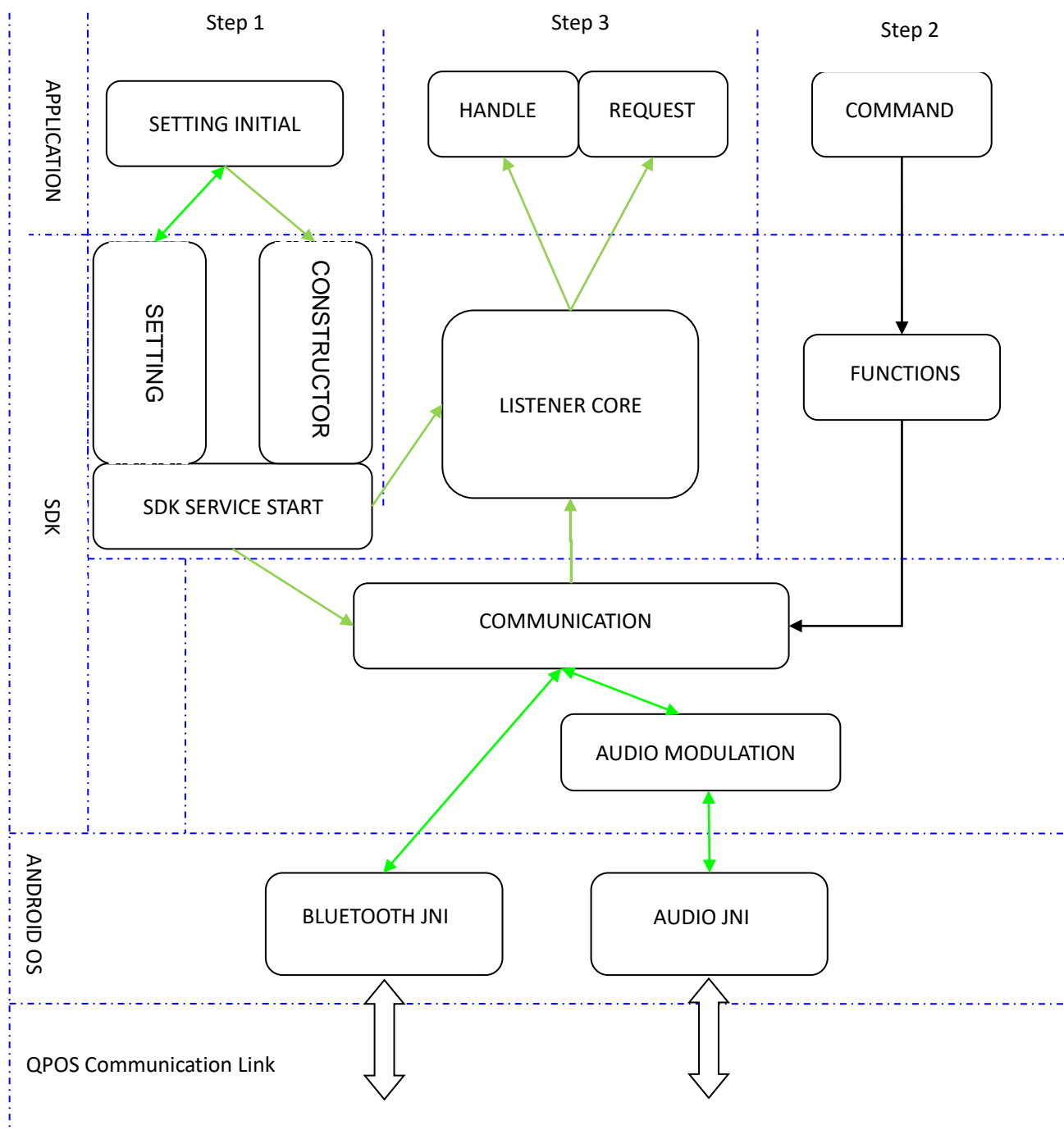
#### 2.1.1 Development Environment

JDK: Java 1.6 or above

#### 2.1.2 OS Version Requirement

OS: Android 2.2 and all above

## 2.2 SDK Framework



## 2.3 Android Permission

The library needs permission to use the audio and Bluetooth resource. The following lines must be added to the **AndroidManifest.xml** file in the APP project.

```
<manifest xmlns:android='http://schemas.android.com/apk/res/android'
package='com.android.app.myapp' >

...

<uses-permission android:name='android.permission.BLUETOOTH'/>
<uses-permission android:name='android.permission.BLUETOOTH_ADMIN'/>
<uses-permission android:name='android.permission.RECORD_AUDIO'/>
<uses-permission
android:name='android.permission.MODIFY_AUDIO_SETTINGS'/>
</manifest>
```

## 2.4 Classification Principle in SDK Methods

According to the SDK framework and the methods feature, all methods are divided into three parts:

1. Init methods;
2. Interactive methods;
3. Listener methods.

To avoid the application block and improve the speed of data interaction between the smart terminal and QPOS, the SDK framework is designed to work under asynchronous mode.

### 2.4.1 Init Methods

The Init methods group is the entry of SDK library, to create the core instance, and build the communication connection between the smart terminal and SDK through the audio jack or the Bluetooth with SPP profile, etc. The method named **'QPOSService'** is the core of SDK library. Before the APP create this core instance with the parameter of "CommunicationMode mode", the APP must register all the sub-functions in **'QPOSServiceListener'**. Then the APP call the sub-function named **'QPOSService.initListener'** of the constructor method named **'QPOSService'**, the method named **'QPOSServiceListener'** should be the input parameter of **'QPOSService.initListener'**.

The detailed description about **"'QPOSServiceListener'"** refers to the **Listener methods**.

Example code of creating the core instance, see the following code:

```
public static enum CommunicationMode{
    BLUETOOTH,// bluetooth mode
    UART// uart mode
}
```

UART Mode:

```
QPOSService pos = QPOSService.getInstance(context,CommunicationMode.UART);
pos.initListener(listener);
```

Bluetooth Mode:

```
QPOSService pos = QPOSService.getInstance(context,CommunicationMode.BLUETOOTH);
pos.initListener(listener);
```

Method Name	Description
QPOSService	Constructor Method
getSdkVersion	Get this SDK version
resetQPOS	Reset and bring the QPOS back to a known initial state.
resetPosStatus	Reset and bring the QPOS back to a known initial state. Synchronized methods
openAudio	Start the audio instance for playing, recording and modulating
closeAudio	Stop the audio instance for playing, recording and modulating
connectBT	Connect to QPOS by Bluetooth, using for exchanging data between APP and QPOS
disconnectBT	Disconnect from QPOS with bluetooth

## 2.4.2 Interactive Methods

The Interactive methods deal with the transaction from the application to QPOS, to get the device information of QPOS, to confirm the transaction information, and to set the transaction command etc. The relative handler of these methods is defined into sub functions of ‘**QPOSServiceListener**’. Some of these methods are triggered by the relative handle methods prefixed by ‘onRequest’ in ‘**QPOSServiceListener**’, and the handler result of other methods needs to be returned by the relative handler methods prefixed by ‘on’, except for ‘onRequest’.

Method Name	Description		Handle Name
getQposId	Get the serial number about Qpos	→	onQposIdResult
	Get the config		



getQposInfo	information from Qpos	→	onQposInfoResult
setAmount	Set the amount and transaction type required for EMV transaction.	←	onRequestSetAmount
cancelSetAmount	Cancel the process about setting amount	←	onRequestSetAmount
doTrade	Send the command as swiping/inserting card to QPOS.	→	onDoTradeResult
doEmvApp	Send the command as executing the EMV transaction flow to QPOS		no
selectEmvApp	Select one application of the application list returned from EMV kernel, then set the application ID to EMV kernel	←	onRequestSelectEmvApp
cancelSelectEmvApp	Cancel the process about setting one application	←	onRequestSelectEmvApp
finalConfirm	Send transaction confirmation command to Qpos	←	onRequestFinalConfirm
sendOnlineProcessResult	Send the connectivity status about network to Qpos		
isServerConnected	Send the connectivity status about network to Qpos		
sendTime	Set the date and time formatted as 'YYMMDDHHMMS S' to Qpos, based the smart terminal date and time	←	onRequestTime
<a href="#">sendPin</a>	<a href="#">Set the pin to QPOS Reader</a>	←	<a href="#">onRequestSetPin</a>

<a href="#">emptyPin</a>	<a href="#">Set the empty pin to QPOS Reader</a>	←	<a href="#">onRequestSetPin</a>
<a href="#">cancelPin</a>	<a href="#">Set cancel to QPOS Reader</a>	←	<a href="#">onRequestSetPin</a>
powerOnIcc	Turn on the EMV card.	←	onReturnPowerOnIccResult
powerOffIcc	Turn off the EMV card.	←	onReturnPowerOffIccResult
sendApu	Send data to EMV card in raw APDU formats. This is the EMV Level 1 protocol and . developers can develop their only EMV Level 2 application	←	onReturnApuResult
setPosSleepTime	Set the pos sleep time	←	onReturnSetSleepTimeResult
updateWorkKey	update the pos work key	←	onRequestUpdateWorkKeyResult
setMasterKey	update the pos master key	←	onReturnSetMasterKeyResult

### 2.4.3 Listener Methods

The Listener methods deal with the event and the handler from QPOS to the application, to handle all events from QPOS to APP, and to handle the returned result of some commands from the APP to QPOS. The method named ‘**QPOSService**’ is the constructor method, and the method named ‘**QPOSServiceListener**’ must be registered in the sub-function named “**QPOSService.initListener**” of the constructor method. Some sub-functions in ‘**QPOSServiceListener**’ are either mandatory or optional. Here we recommend the developer to define all sub-functions referring to the following table. Even if some sub-functions are optional, it is better to define them as Null functions.

**Note: Please note listener methods are all called in child thread but not in main thread**

Handle Name	Description			Method Name
onRequestWaitingUser	Qpos is ready and waiting for swiping or inserting a EMV card	o		no



onQposIdResult	Return the serial number about Qpos	o	←	getQposId
onQposInfoResult	Return the config information about Qpos	o	←	getQposInfo
onDoTradeResult	Return the action about swiping, inserting the ICC, canceling etc.	m	←	doTrade
onRequestSetAmount	Prompt to inputting the transaction amount to the application	m	→	setAmount
onRequestSelectEmvApp	Supply application list supported by EMV ICC to the application for selecting.	m	→	<u>selectEmvApp</u>
<u>onRequestIsServerConnected</u>			→	isServerConnected
		m		
onRequestFinalConfirm	Finally confirm before generating the AC by the ICC COS	m	→	finalConfirm
onRequestOnlineProcess	EMV kernel request the online handler	m	→	sendOnlineProcessResult
onRequestTime	Request setting the date and time from Qpos	o	→	sendTime



onRequestTransactionResult	After finishing this transaction, EMV kernel report this transaction result to the application	m		
onRequestTransactionLog	After finishing this transaction, EMV kernel report this transaction log to the application	o		
onRequestBatchData	EMV kernel send the batch data to the application	m		
onRequestPosConnected	SDK report to the application about the connected event between smart terminal and Qpos	m		
onRequestPosDisconnected	SDK report to the application about the disconnected event between smart terminal and Qpos	m		
onRequestNoPosDetected	SDK report to the application about the no connected event between smart terminal and Qpos	m		
onError	SDK report the error ID to the application during transaction	m		
onRequestDisplay	SDK request display to the application	o		
<a href="#">onRequestSetPin</a>	<a href="#">SDK request the application to set PIN for the EMV card. Note, this is only available for QPOS Reader since QPOS Reader doesn't has PINPAD.</a>	<a href="#">o</a>	→	<a href="#">sendPin</a> <a href="#">emptyPin</a> <a href="#">cancelPin</a>
onReturnPowerOnIccResult	Turn on the EMV card result	<a href="#">o</a>	→	powerOnIcc
onReturnPowerOffIccResult	Turn off the EMV card result	<a href="#">o</a>	→	powerOffIcc



onReturnApduResult		0	→	sendApdu
onReturnSetSleepTimeResult	Set the pos sleep time	0	→	setPosSleepTime
onRequestUpdateWorkKeyResult	update the pos work key	0	→	updateWorkKey
onReturnSetMasterKeyResult	update the pos work key	0	→	setMasterKey

## 2.5 Transaction Flow

From the return of ‘**onDoTradeResult**’, the APP can find out whether consumer use an EMV ICC card or a magnetic card.

Public void **onDoTradeResult**(DoTradeResult result, Hashtable<String, String> decodeData);

Enum **DoTradeResult** can be any of following

NONE,  
ICC,  
NOT\_ICC,  
BAD\_SWIPE,  
MCR,  
MAG\_HEAD\_FAIL,  
NO\_RESPONSE,  
TRACK2\_ONLY ,  
NFC\_TRACK2

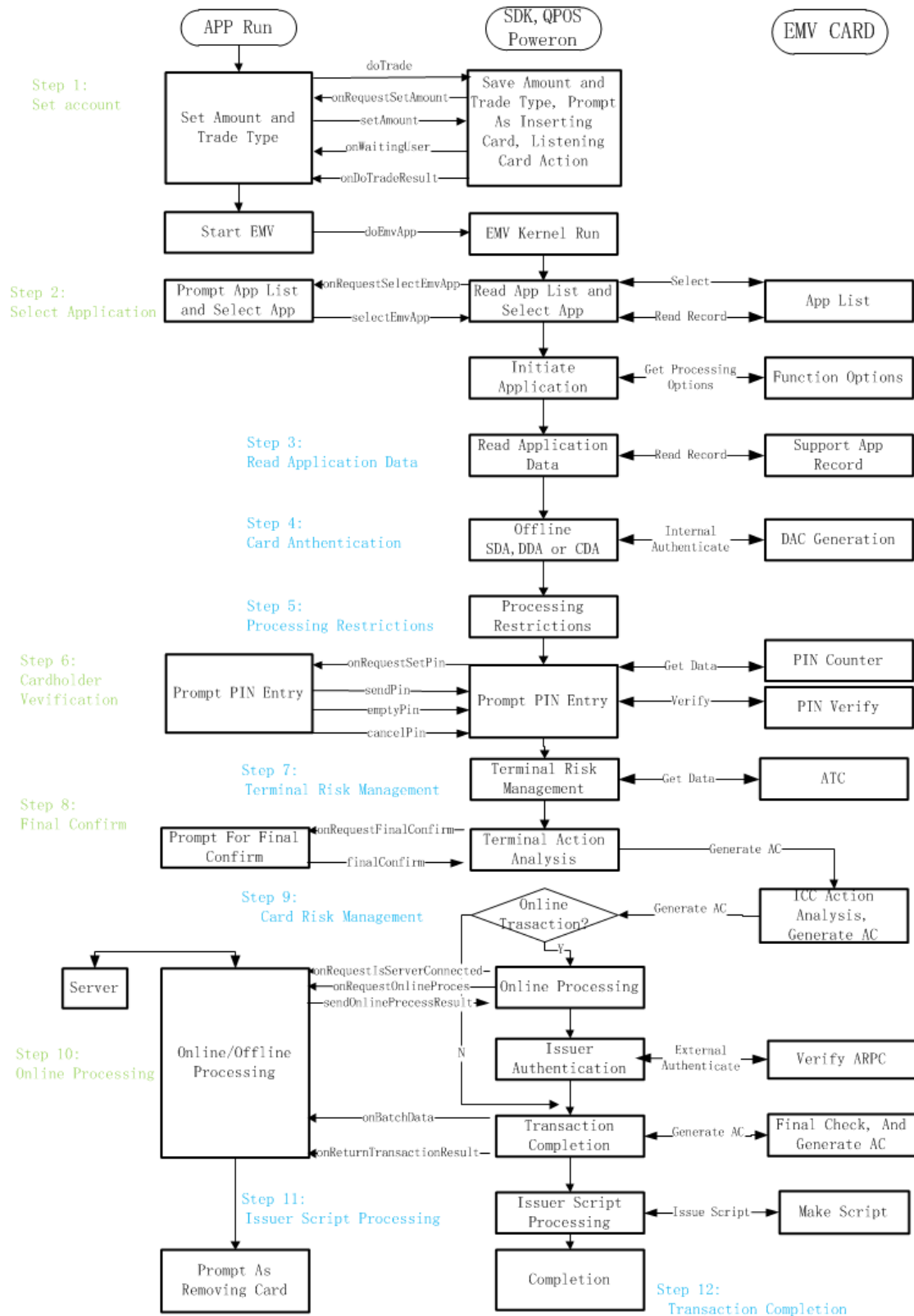
When the delegate method “**onDoTradeResult**” is triggered to return the “Enum DoTradeResult” result, if this result is **ICC**, the APP will trigger the EMV ICC transaction flow, or this result is **MCR**, the APP will trigger the magnetic transaction flow.

### 2.5.1 EMV ICC Transaction Flow For QPOS

I



## 2.5.2 EMV ICC Transaction Flow for QPOS Reader



## 2.5.3 EMV ICC Transaction Description

**Step 1:** A transaction amount is needed for a payment transaction and it to be entered by the operator (or calculated by the inventory system) regardless of whether magnetic stripe card or EMV card. The **onRequestSetAmount** delegate method is triggered.

Public void **onRequestSetAmount** ();

The APP should prompt the operator to enter the amount and then call the **setAmount** to send the data back to EMV kernel.

Public void **setAmount**(String amount,String cashbackAmount,String currency,TransactionType transactionType);//2 decimal places.e.g.234.87

The **transactionType** can be any of the following:

GOODS,  
SERVICES,  
CASHBACK,  
INQUIRY,  
TRANSFER,  
PAYMENT

The amount has an upper limit of 1000000000.00.

The user can also select to abort the transaction.

Public void **cancelSetAmount**();

### Step 2: Select Application

An EMV card may support multiple payment applications. The EMV kernel reads the list of applications supported by the EMV card and asks the customer/operator to select the desired application.

The delegate method **onRequestSelectApplication** is triggered to return an array of application Ids.

Public void **onRequestSelectEmvApp** (ArrayList<String>applist);

The APP should prompt the user to select one application and then call the **selectEmvApp** method.

Public void **selectEmvApp** (int index)

The user can also select to abort the transaction.

Public void **cancelSelectEmvApp** ()

In most cases, there is only one default application and this step is skipped.

### Step 3: Read Application Data

In this step, EMV kernel reads the necessary data from the EMV card. The EMV kernel asks for the terminal time through the **onRequestTime** method.

This step is only done between the EMV kernel and EMV card. If this step fails, **onRequestTransactionResult** will be returned and the EMV process stops.

```
Public void onRequestTime ();
```

The terminal time in YYMMDDHHmmss formats should be sent in response:

```
Public void sendTime(String terminalTime);
```

### Step 4: Card Authentication

This step is only done between the EMV kernel and EMV ICC card. If this step fails, **onRequestTransactionResult** will be returned and the EMV process stops.

### Step 5: Processing Restrictions

This step is only done between the EMV kernel and EMV ICC card. If this step fails, **onRequestTransactionResult** will be returned and the EMV process stops.

### Step 6: Cardholder Verification

There are the different cardholder verification methods (CVMs) supported in an EMV transaction and some require the customer to enter a PIN (personal identifier number). If the EMV transaction requires PIN verification, the customer must enter his PIN by the keypad of QPOS. Where PIN is 4-12 digits. The PIN can be input via PINPAD for QPOS and SPOS, or can be input via mobile application for QPOS Reader.

Some applications (e.g. for small amount payment) does not require CVM (Cardholder Verification Method) and this step is skipped. But it is also possible that the EMV card decline a transaction without PIN. The customer/operator can press the **cancel** key on the keypad of QPOS to cancel the PIN entry and abort.

### Step 7: Terminal Risk Management

This step is only done between the EMV kernel and EMV ICC card. If this step fails, **onRequestTransactionResult** will be returned and the EMV process stops.

### Step 8: Terminal Action Analysis

This step is only done between the EMV kernel and EMV ICC card. If this step fails, **onRequestTransactionResult** will be returned and the EMV process stops. At the end of this step, a final confirmation will be needed to proceed via **onRequestFinalConfirm**.

```
Public void onRequestFinalConfirm();
```



The APP should prompt the user for a confirmation to proceed. This gives the user a chance to review the amount, the payment method, etc.

A final confirmation is sent to EMV kernel by calling this :

```
public void finalConfirm(boolean isConfirmed);
```

### Step 9: Card Risk Management

This step is only done between the EMV kernel and EMV ICC card. If this step fails, **onRequestTransactionResult** will be returned and the EMV process stops.

### Step 10: Online Processing

An EMV transaction can either be online or offline. If online processing is required, then the **onRequestOnlineProcess** delegate method is triggered.

```
public void onRequestOnlineProcess(string tlv);
```

The parameter TLV contains the tag-length-value data structure returned by the EMV kernel. After that, the client APP should send the data to the payment operator. When the processing results are returned from the payment operator, it should send the results back to EMV kernel by **sendOnlineProcessResult**.

```
public void sendOnlineProcessResult(String tlv);
```

The data elements that are required are payment operator and issuer dependent. See **chapter 1.9** and the **EVM Book 3 Annex A** for the full list of tags and the TLV structure.

The following tags are usually required but are ICC dependent:

Tags	Parameter
0089	Authorisation Code
008A	Authorisation Response Code
0091	Issuer Authentication Data
0071	Issuer Script Template 1 (needed for Step 11)
0072	Issuer Script Template 2 (needed for Step 11)

This step is skipped in offline processing.

### Step 11: Issuer Scripts Processing

This step is handled transparently between the EMV kernel and EMV ICC card if issue scripts are present in the online processing results or skipped otherwise. This step is skipped in offline processing.

### Step 12: Completion

In this step, EMV kernel sends back the final transaction result from the EMV card by the

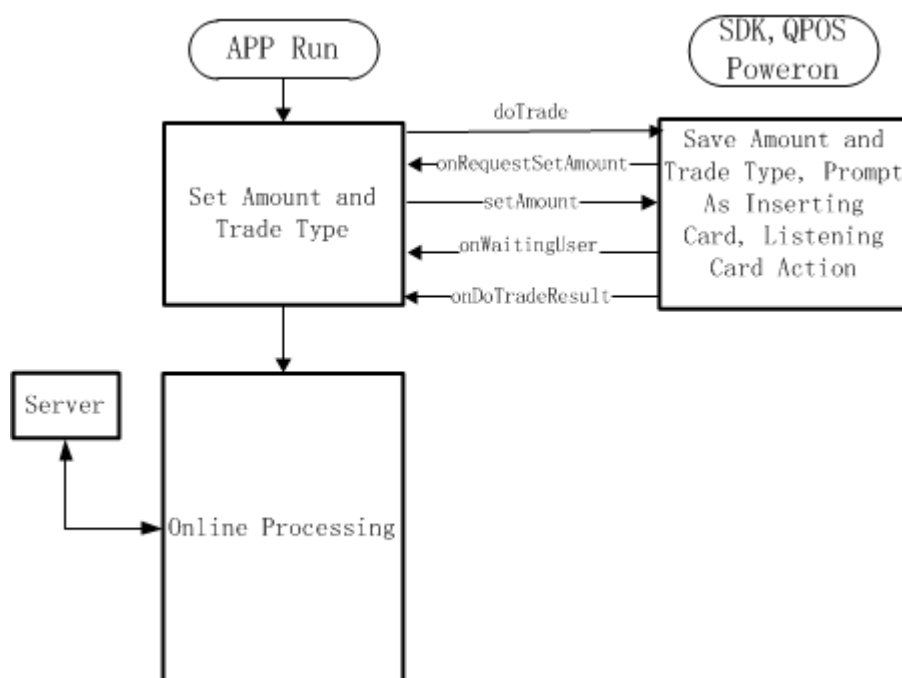
**onRequestBatchData** method.

```
public void onRequestBatchData (String tlv);
```

The data elements that are required are payment operator and issuer dependent. See the **EVM Book 3 Annex A** for the full list of tags and the TLV structure.

In this step, the client APP should store the results, display the results, print receipts, and prompt the user to remove the card from the QPOS device. Later, the batch data should be updated to the server for settlement.

## 2.5.4 Magnetic Transaction Flow



## 2.5.5 Magnetic Transaction Description

If a magnetic stripe card has been swiped, the encrypted PINBLOCK and the encrypted track data will be returned along with the **trackksn** and the **pinKsn** in the **decodeData Hashtable**.

```
Public void onDoTradeResult(DoTradeResult result, Hashtable<String, String> decodeData);
```

The **Hashtable** contain keys for the values:

Key	Description
maskedPAN	Masked card number showing at most the first 6 and last 4 digits with in-between digits masked by “X”

expiryDate	4-digit in the form of YYMM in the track data
cardHolderName	The cardholder name as seen on the card. This can be up to 26 characters.
serviceCode	3-digit service code in the track data
track1Length	Length of Track 1 data
track2Length	Length of Track 2 data
track3Length	Length of Track 3 data
encTracks	Reserved
encTrack1	Encrypted track 1 data with T-Des encryption key derived from <b>DATA-key</b> to be generated with <b>trackksn</b> and <b>IPEK</b>
encTrack2	Encrypted track 2 data with T-Des encryption key derived from <b>DATA-key</b> to be generated with <b>trackksn</b> and <b>IPEK</b>
encTrack3	Encrypted track 3 data with T-Des encryption key derived from <b>DATA-key</b> to be generated with <b>trackksn</b> and <b>IPEK</b>
partialTrack	Reserved
pinKsn	KSN of the Pin-block
trackksn	KSN of the track data
pinBlock <sup>1</sup>	Encrypted PIN data with T-Des encryption key derived from <b>PIN-key</b> to be generated with <b>pinKsn</b> and <b>IPEK</b>

Note:

1. The **PIN** format is defined as **ANSI X9.8**.

$\text{pinBlock} := (\text{PIN}^{\text{PAN}})$ ;

## 2.6 API Methods Reference

### 2.6.1 getSdkVersion

Signature	String getApiVersion()
Inputs	None
Outputs	SDK version
Description	Return the SDK version
See also	

### 2.6.2 resetQPOS

Signature	void resetQPOS()
Inputs	None
Outputs	None
Description	Reset and bring the QPOS back to a known initial state.
See also	

### 2.6.3 scanQPos2Mode

Signature	List<BluetoothDevice> scanQPos2Mode(Context context)
Inputs	context
Outputs	List<BluetoothDevice>: the list of bluetoothDevice
Description	Start to scan the devices and return the list of scanned devices
See also	The reponse method of onDeviceFound

## 2.6.4 stopQPos2Mode

Signature	void stopQPos2Mode()
Inputs	None
Outputs	None
Description	Stop to scan the device
See also	

## 2.6.5 openAudio

Signature	void openAudio ()
Inputs	None
Outputs	None
Description	Start the audio instance for playing, recording and modulating.
See also	

## 2.6.6 closeAudio

Signature	void closeAudio ()
Inputs	None
Outputs	None
Description	Stop the audio instance for playing, recording and modulating.
See also	

## 2.6.7 connectBT

Signature	void connectBT (String address)
Inputs	Bluetooth mac address.
Outputs	None
Description	Connect to QPOS by Bluetooth, using for exchanging data between APP and QPOS.
See also	

## 2.6.8 connectBluetoothDevice

Signature	boolean connectBluetoothDevice(boolean auto,int bondtime, String blueToothAddress)
Inputs	auto: Whether automatic matching.  bondtime: Connection timeout  blueToothAddress : Bluetooth mac address.
Outputs	None
Description	Connect to QPOS by Bluetooth, using for exchanging data between APP and QPOS.
See also	

## 2.6.9 disconnectBT

Signature	void disconnectBT ()/void disconnectBT (String address)
Inputs	None/Bluetooth mac address.
Outputs	None

Description	Connect to QPOS by Bluetooth, using for exchanging data between APP and QPOS.
See also	

### 2.6.10 isQposPresent

Signature	boolean isQposPresent()
Inputs	None.
Outputs	BOOL: Presence flag of the Qpos.
Description	Check if an QPOS is connected and ready.
See also	

### 2.6.11 getQposId

Signature	getQposId()
Inputs	None
Outputs	None
Description	Retrieve the POS_ID of the QPOS device. Results are returned by onQposIdResult.
See also	onQposIdResult

### 2.6.12 getQposInfo

Signature	getQposInfo()
Inputs	None
Outputs	None

Description	Retrieve parameters about the QPOS device. Results are returned by onQposInfoResult which includes: firmware version, bootloader version, USB connection and charging status, battery level, and hardware version
See also	onQposInfoResult

### 2.6.13 setAmount

Signature	void setAmount(String amount, String amountDescribe, String currencyCode, TransactionType transactionType)
Inputs	<p>amount: the amount for a transaction.<sup>[L]</sup><sub>SEP</sub></p> <p>cashbackAmount: the amount for a transaction. If this is non-zero, amount cannot be zero.</p> <p>currencyCode: three digits of the currency code, e.g. “840” for USD</p> <p>transactionType: enum of the transaction type.</p>
Outputs	None
Description	<p>Set the amount, currency and type of a transaction. This method can be called before a transaction or in response to an onRequestSetAmount call requested by the EMV engine.<sup>[L]</sup><sub>SEP</sub> The Enum TransactionType can be</p> <p>GOODS, SERVICES, CASHBACK, INQUIRY, TRANSFER, PAYMENT</p>
See also	onRequestSetAmount , cancelSetAmount

### 2.6.14 cancelSetAmount

Signature	void cancelSetAmount ()
-----------	-------------------------



Inputs	None
Outputs	None
Description	Cancel setting the amount of a transaction. This method can be called to abort a transaction in response to <code>onRequestSetAmount</code>
See also	<code>onRequestSetAmount</code>

### 2.6.15 doTrade

Signature	<code>void doTrade(int timeout)</code>
Inputs	Timeout
Outputs	None
Description	Check the status of the Magnetic Card Reader, the EMV Card reader, or NFC transceiver. It checks if a card has been swiped, a NFC card has been tapped or an EMV card is inserted. The result is returned by the <code>onDoTradeResult</code> delegate method.
See also	<code>onDoTradeResult</code>

### 2.6.16 doCheckCard

Signature	<code>void doCheckCard (int timeout)</code>
Inputs	Timeout
Outputs	None
Description	Start transaction, no pin input
See also	<code>onDoTradeResult</code>

## 2.6.17 s doEmvApp

Signature	void doEmvApp (EmvOption emvOption)
Inputs	EmvOption: enum {  START START_WITH_FORCE_ONLINE }
Outputs	None
Description	Start Emv app, Emv card
See also	onDoTradeResult

## 2.6.18 cancelSetAmount

Signature	void cancelSetAmount ()
Inputs	None
Outputs	None
Description	Cancel Set Amount
See also	

## 2.6.19 setAmount

Signature	void setAmount (String amount, String cashbackAmount, String currencyCode, TransactionType transactionType) OR void setAmount (String amount, String cashbackAmount, String currencyCode, TransactionType transactionType, boolean isPosDisplayAmount)
Inputs	Amount:  CashbackAmount



	CurrencyCode  TransactionType  IsPosDisplayAmount
Outputs	None
Description	Set amount
See also	

### 2.6.20 selectEmvApp

Signature	void selectEmvApp (int index)
Inputs	Index: select emv app index
Outputs	None
Description	Select Emv App
See also	

### 2.6.21 cancelSelectEmvApp

Signature	void cancelSelectEmvApp ()
Inputs	None
Outputs	None
Description	Cancel Select Emv App
See also	

### 2.6.22 finalConfirm

Signature	void finalConfirm (bool isConfirmed)
Inputs	isConfirmed

Outputs	None
Description	Not support
See also	

### 2.6.23 sendOnlineProcessResult

Signature	void sendOnlineProcessResult (String tlv)
Inputs	Tlv: type + length + value
Outputs	None
Description	Send transaction results from the processor back to EMVSwipe
See also	onRequestOnlineProcess

### 2.6.24 isServerConnected

Signature	void isServerConnected (bool isConnected)
Inputs	isConnected: whether the signature connect the successfully or not
Outputs	None
Description	Send the connectivity status to EMVSwipe
See also	onRequestIsServerConnected

### 2.6.25 sendTime

Signature	void sendTime (String atterminalTime)
Inputs	atterminalTime: terminalTime
Outputs	None

Description	Send the terminal time in yyyyMMddHHmmss format to EMVSwipe
See also	onRequestTime

### 2.6.26 setAmountIcon

Signature	void setAmountIcon (String aAmountIcon)/ void setAmountIcon (AmountType amtType String aAmountIcon)
Inputs	amtType: amount type  aAmountIcon: amountIcon
Outputs	None
Description	Set Amount symbol.  String Data is a string with length below 4 characters, and can only be ASCII string. below are some valid examples :  “RUP”  “USD”  “CNY”
See also	

### 2.6.27 getPin

Signature	void getPin(String aTransactionData)
Inputs	aTransactionData: transactionData
Outputs	None
Description	Get Pin
See also	onReturnGetPinResult

## 2.6.28 powerOnIcc

Signature	void powerOnIcc ()
Inputs	None
Outputs	None
Description	Provide power to ICC for level 1 APDU exchange
See also	onReturnPowerOnIccResult

## 2.6.29 powerOffIcc

Signature	void powerOffIcc ()
Inputs	None
Outputs	None
Description	Cut off power to ICC after level 1 APDU exchange
See also	onReturnPowerOnIccResult

## 2.6.30 sendApdu

Signature	void sendApdu (String apduStr)
Inputs	apduStr: apdu character string
Outputs	None
Description	Send APDU exchange to ICC. Response data are returned by onReturnApduResult
See also	onReturnApduResult

## 2.6.31 setPosSleepTime

Signature	void setPosSleepTime (int sleepTime)
-----------	--------------------------------------

Inputs	sleepTime
Outputs	None
Description	Set Sleep Time
See also	onReturnSetSleepTimeResult

### 2.6.32 UpdateEmvCAPK

Signature	void updateEmvCAPK (String emvAppCfg,String emvCapkCfg)
Inputs	emvAppCfg emvCapkCfg
Outputs	None
Description	update emv config files(CAPK and AID)
See also	onReturnCustomConfigResult

### 2.6.33 readEmvAppConfig

Signature	void readEmvAppConfig ()
Inputs	None
Outputs	None
Description	read emv app config
See also	onReturnCustomConfigResult

### 2.6.34 readEmvCapkConfig

Signature	void readEmvCapkConfig ()
Inputs	None

Outputs	None
Description	read emv capk config
See also	onReturnCustomConfigResult

### 2.6.35 udapteWorkKey

Signature	void udapteWorkKey (String pik,String pikCheck,String trk,String trkCheck,String mak,String makCheck,String tnsk,String tnskCheck,int keyIndex,int timeout)
Inputs	pik:  pikCheck  trk  trkCheck  mak  makCheck  tnsk  tnskCheck  keyIndex  timeout
Outputs	None
Description	Update Work Key
See also	onRequestUpdateWorkKeyResult

### 2.6.36 MacKeyEncrypt

Signature	void MacKeyEncrypt (String macStr)
Inputs	macStr: mac character string



Outputs	None
Description	macKey Encrypt
See also	onRequestCalculateMac

### 2.6.37 isQposPresent

Signature	void isQposPresent ()
Inputs	None
Outputs	None
Description	Get Qpos available status
See also	

### 2.6.38 setMasterKey

Signature	void setMasterKey (String key,String ckValue,int keyIndex,int timeout)
Inputs	key: master key  ckValue: check value  keyIndex  timeout
Outputs	None
Description	Set Master Key
See also	onReturnSetMasterKeyResult

### 2.6.39 calcMacSingle

Signature	void setMasterKey (String macStr)
Inputs	macStr: mac character string

Outputs	None
Description	Calculate mac (single )
See also	onRequestCalculateMac

### 2.6.40 calcMacDouble

Signature	void calcMacDouble (String macStr,int keyIndex,int timeout)
Inputs	macStr: mac character string  keyIndex  timeout
Outputs	None
Description	Calculate mac (double )
See also	onRequestCalculateMac

### 2.6.41 calcMacSingleNoCheck

Signature	void calcMacSingleNoCheck (String macStr,int timeout)
Inputs	macStr: mac character string  timeout
Outputs	None
Description	Calculate mac (double)
See also	onRequestCalculateMac

### 2.6.42 calcMacDoubleNoCheck

Signature	void calcMacDoubleNoCheck (String macStr,int keyIndex,int timeout)
-----------	--

Inputs	macStr: mac character string  keyIndex  timeout
Outputs	None
Description	Calculate mac (double)
See also	onRequestCalculateMac

### 2.6.43 downloadRsaPublicKey

Signature	void downloadRsaPublicKey (int useType,String rid,String index,String module,String exponent,int timeout)
Inputs	useType  rid: public Key RID  index: Public Key Index  module: Public Key Module  exponent: Public Key Exponent  timeout
Outputs	None
Description	Acquire server RSA public key 详细操作步骤: 第一步: 调用服务器接口, 从服务器获取公钥 第二步: 调用此接口获取用这公钥加密的 randomkey 第三步: 将 randomkey 上送服务器, 服务器解密 randomkey 后返回用 randomkey 明文加密的终端主密钥 第四步: 调用 setMasterKey 设置主密钥
See also	onReturnDownloadRsaPublicKey

### 2.6.44 updateMasterKeyRandom

Signature	void updateMasterKeyRandom (int step,String keyIndex,String masterKey,String masterKeyCheck,int
-----------	---

	timeout)
Inputs	step: step index  keyIndex  masterKey  masterKeyCheck  timeout
Outputs	None
Description	Update Master Key in Random method 详细操作步骤: 第一步: 调用此接口获取随机数并上送服务器, 服务器返回用这个随机数加密的主密钥 第二步: 拿到主密钥密文后再调用下面接口设置主密钥
See also	onUpdateMasterKeyResult

### 2.6.45 updateMasterKey

Signature	void updateMasterKey (int step,String RN1,String RN2,String masterKey,String masterKeyCheck,int timeout)
Inputs	step: step index  RN1: random key 1  RN2: random key 2  masterKey  masterKeyCheck  timeout
Outputs	None
Description	Set Master Key
See also	onUpdateMasterKeyResult

## 2.6.46 pinKey\_TDES

Signature	void pinKey_TDES (int keyIndex,String pin,int timeout)
Inputs	keyIndex  pin  timeout
Outputs	None
Description	Use pinKey to encrypt data
See also	onPinKey_TDES_Result

## 2.6.47 pinKey\_TDESNoCheck

Signature	void pinKey_TDESNoCheck (int keyIndex,String pin,int timeout)
Inputs	keyIndex  pin  timeout
Outputs	None
Description	Use pinKey to encrypt data (NoCheck)
See also	onPinKey_TDES_Result

## 2.6.48 setSystemDateTime

Signature	void setSystemDateTime (String dateTime,int timeout)
Inputs	dateTime  timeout
Outputs	None

Description	Set SystemDate Time
See also	onSetParamsResult

### 2.6.49 setMerchantID

Signature	void setMerchantID (String merchantID,int timeout)
Inputs	merchantID timeout
Outputs	None
Description	Set Merchant ID
See also	onSetParamsResult

### 2.6.50 setTerminalID

Signature	void setTerminalID (String terminalID,int timeout)
Inputs	terminalID timeout
Outputs	None
Description	Set Terminal ID
See also	onSetParamsResult

### 2.6.51 getMagneticTrackPlaintext

Signature	void getMagneticTrackPlaintext (int timeout)
Inputs	timeout
Outputs	None
Description	Get Magnetic Track Plaintext

See also	onDoTradeResult
----------	-----------------

### 2.6.52 getCardNo

Signature	void getCardNo ()
Inputs	None
Outputs	None
Description	Acquire cardPan (Magnetic stripe card)
See also	onGetCardNoResult

### 2.6.53 getIccCardNo

Signature	void getIccCardNo (String aterminalTime)
Inputs	aterminalTime: terminalTime
Outputs	None
Description	Acquire cardPan (IC、Magnetic stripe card)
See also	onGetCardNoResult onReturniccCashBack onRequestBatchData

### 2.6.54 powerOffNFC

Signature	void powerOffNFC (int timeout)
Inputs	timeout
Outputs	None
Description	Turn off the NFC transceiver
See also	onReturnPowerOffNFCResult

### 2.6.55 sendApduByNFC

Signature	void sendApduByNFC (String apduString,int timeout)
Inputs	apduString  timeout
Outputs	None
Description	Send data to EMV card in raw APDU formats by nfc
See also	onReturnNFCApduResult

### 2.6.56 powerOnNFC

Signature	void powerOnNFC (int isEncrypt,int timeout)
Inputs	isEncrypt  timeout
Outputs	None
Description	Turn on the NFC transceiver
See also	onReturnPowerOnNFCResult

### 2.6.57 cbc\_mac

Signature	void cbc_mac (int keyLen, int algorithmType, int operatorType, String data, int timeout)
Inputs	keyLen  algorithmType  operatorType  data  timeout



Outputs	None
Description	With 3DES CBC mode calculate Mac
See also	onCbcMacResult

### 2.6.58 cbc\_macNoCheck

Signature	void cbc_mac (int keyLen, int algorithmType, int operatorType, String data, int timeout)
Inputs	keyLen  algorithmType  operatorType  data  timeout
Outputs	None
Description	Calculate Mac (No Check)with 3DES CBC mode
See also	onCbcMacResult

### 2.6.59 inquireECQAmount

Signature	void inquireECQAmount (String transactionTime)
Inputs	transactionTime
Outputs	None
Description	Inquire IC card Electronic pocket balance
See also	onGetCardNoResult onReturniccCashBack onRequestBatchData

## 2.6.60 isIdle

Signature	Boolean isIdle ()
Inputs	None
Outputs	The status of pos is trading
Description	Flag indicate pos is working
See also	

## 2.6.61 anlysEmvIccData

Signature	void anlysEmvIccData (String tlv)
Inputs	tlv: type + length + value
Outputs	None
Description	Parsing ICC data
See also	

## 2.6.62 VIPOSBatchSendAPDU

Signature	void VIPOSBatchSendAPDU (LinkedHashMap<Integer, String[]> batchAPDU)
Inputs	batchAPDU
Outputs	None
Description	send Batch APDU command VIPOS protocol
See also	onReturnBatchSendAPDUResult

## 2.6.63 synVIPOSBatchSendAPDU

Signature	void synVIPOSBatchSendAPDU (Boolean isOpen,
-----------	---

	LinkedHashMap<Integer, String[]> batchAPDU)
Inputs	isOpen: whether the signature is open or not  batchAPDU: batchAPDU command
Outputs	None
Description	send Batch APDU command VIPOS protocol (synchronize mode)
See also	onReturnBatchSendAPDUResult

### 2.6.64 anlysEmvIccData\_qf

Signature	void anlysEmvIccData_qf (String tlv)
Inputs	Tlv : type + length + value
Outputs	None
Description	Parsing ICC data qf
See also	

### 2.6.65 iccCashBack

Signature	void iccCashBack (String transactionTime, String random)
Inputs	transactionTime: transactionTime  random: random character string
Outputs	None
Description	Specific customer IC card cash back trade
See also	onGetCardNoResult onReturnicccashBack onRequestBatchData

## 2.6.66 setPosPresent

Signature	void setPosPresent (boolean flag)
Inputs	flag: whether the POS device has exist
Outputs	None
Description	Flag for POS existence
See also	

## 2.6.67 setCardTradeMode

Signature	void setCardTradeMode (CardTradeMode cardTradeMode)
Inputs	cardTradeMode:  enum{ ONLY_INSERT_CARD//only IC  ONLY_SWIPE_CARD//Only Magnetic stripe card  SWIPE_INSERT_CARD//Magnetic stripe card and IC  UNALLOWED_LOW_TRADE//the devices will remind to insert the card if the card are both chip and track }
Outputs	None
Description	Set Card Trade Mode, ( Only Magnetic stripe card, only IC, Magnetic stripe card and IC )
See also	

## 2.6.68 setPinPadFlag

Signature	void setPinPadFlag (boolean flag)
Inputs	flag: whether the key signature exists or not
Outputs	None

Description	Device keypad flag
See also	

### 2.6.69 qposStatus

Signature	void qposStatus ()
Inputs	None
Outputs	None
Description	Check APP connect status
See also	

### 2.6.70 readBusinessCard

Signature	void readBusinessCard (String cardType, String address, int readLen, String cardPin, int vender_id , int timeout)
Inputs	cardTyp: card type  address: The address of reading data  readLen: The length of reading data  cardPin: password of the card  vender_id: The ID of the Vender  timeout: overtime
Outputs	None
Description	Read Business Card
See also	onReadBusinessCardResult

### 2.6.71 writeBusinessCard

Signature	void writeBusinessCard (String cardType, String address, String data, String cardPin, boolean isUpdatePinFlag, int
-----------	--

	vender_id, int timeout)
Inputs	cardType: card type  address: read-in address  data : read-in data  cardPin: password of the card  isUpdatePinFlag: whether update the password or not  vender_id: : The ID of the Vender  timeout: overtime
Outputs	None
Description	write Business Card
See also	onWriteBusinessCardResult

### 2.6.72 syncReadBusinessCard

Signature	void syncReadBusinessCard (String cardType, String address, int readLen, String cardPin, int vender_id , int timeout)
Inputs	cardType: card type  address: the address of reading data  readLen: the length of reading data  cardPin: the password of the card  vender_id: the ID of the vender  timeout: overtime
Outputs	None
Description	Read Business Card (synchronize mode)
See also	

### 2.6.73 syncWriteBusinessCard

Signature	void syncWriteBusinessCard (String cardType, String address, String data, String cardPin, boolean isUpdatePinFlag, int vender_id, int timeout)
Inputs	<p>cardType: card type</p> <p>address: the address of the read-in data</p> <p>data : read-in the data</p> <p>cardPin: the password of the card</p> <p>isUpdatePinFlag: whether update the password or not</p> <p>vender_id: the ID of the vender</p> <p>timeout: overtime</p>
Outputs	None
Description	Write Business Card (synchronize mode)
See also	

### 2.6.74 confirmAmount

Signature	void confirmAmount (String amount, int timeout)
Inputs	<p>amount: payment amount</p> <p>timeout: overtime</p>
Outputs	None
Description	Confirm Amount
See also	onConfirmAmountResult

### 2.6.75 setAmount

Signature	void setAmount (String amount, String cashbackAmount, String currencyCode, TransactionType
-----------	--



	transactionType,boolean isPosDisplayAmount)
Inputs	amount: transaction amount  cashbackAmount: enchashment amount  currencyCode: currency code  transactionType: transaction type  isPosDisplayAmount: whether display on POS device or not
Outputs	None
Description	Set Amount
See also	

### 2.6.76 getPin

Signature	void getPin (int encryptType, int keyIndex, int maxLen, String typeFace, String cardNo, String data, int timeout)
Inputs	encryptType: default:0  keyIndex: default:0  maxLen: max length of pin  typeface: display the font  cardNo:the cardPan  data: attached data  timeout
Outputs	None
Description	Get Pin
See also	onReturnGetPinResult

### 2.6.77 doSetManagementKey

Signature	void doSetManagementKey(String publicKey)
-----------	---



Inputs	publicKey
Outputs	None
Description	Set the management publicKey
See also	onSetManagementKey

## 2.6.78 doSetBuzzerOperation

## 2.6.79 doSetBuzzerOperation

Signature	void doSetBuzzerOperation(int time)
Inputs	The buzzer ringing's time
Outputs	None
Description	Set the buzzer
See also	onSetBuzzerResult

## 2.6.80 doUpdateIPEKOperation

Signature	void doUpdateIPEKOperation(String ipekgroup,String trackksn,String trackipek,String trackipekCheckvalue,String emvksn,,String emvipek,String emvipekCheckvalue,String pinksn,String pinipek,String pinipekCheckvalue)
Inputs	ipekgroup,1字节trackksn10字节,trackipek16字节,trackipekCheckvalue8字节,emvksn10字节,emvipek16字节,emvipekCheckvalue8字节,pinksn10字节,pinipek16字节,pinipekCheckvalue8字节
Outputs	None
Description	Update the IPEK
See also	onReturnUpdateIPEKResult



## 2.6.81 updateIPEKOperationByKeyType

Signature	void updateIPEKOperationByKeyType(String ipekgroup,String trackksn,String trackipek,String trackipekCheckvalue,String emvksn,,String emvipek,String emvipekCheckvalue,String pinksn,String pinipek,String pinipekCheckvalue)
Inputs	ipekgroup,1字节trackksn10字节,trackipek16字节,trackipekCheckvalue8字节,emvksn10字节,emvipek16字节,emvipekCheckvalue8字节,pinksn10字节,pinipek16字节,pinipekCheckvalue8字节
Outputs	None
Description	Update single or multiple IPEK KEY
See also	onReturnUpdateIPEKResult

## 2.6.82 updateEmvAPP

Signature	void updateEmvAPP(EMVDataOperation operationType,String data)
Inputs	operationType:the type of operation。  It's has six type.  Clear:clear all AID  Add:add one AID  Delete:delete one AID  AttainList:get the list of AID  Update:update the AID  getEmv:get the AID data.  data:have different data according to the operationType
Outputs	None
Description	Update the EMV that id is AID

See also	onReturnUpdateEMVResult
----------	-------------------------

### 2.6.83 updateEmvCAPK

Signature	void updateEmvCAPK(EMVDataOperation operationType,String data)
Inputs	operationType:the type of operation  It's has five type.  Clear:clear all RID  Add:add one RID  Delete:delete one RID  AttainList:get the list of RID  getEmv:get the RID data.  data:have different data according to the operationType
Outputs	None
Description	Update the EMV that id is RID
See also	onReturnUpdateEMVRIDResult

### 2.6.84 getUpdateKey

Signature	void getUpdateKey()
Inputs	None
Outputs	None
Description	Get the update key
See also	onRequestUpdateKey

## 2.6.85 setSleepModeTime

Signature	void setSleepModeTime(int time)
Inputs	time:it's value in 10s and 1000s.
Outputs	None
Description	Set the sleepTime
See also	onSetSleepModeTime

## 2.6.86 generateTransportKey

Signature	void generateTransportKey(int timeout)
Inputs	Timeout
Outputs	None
Description	Generate the transport key encrypted by the server's public key
See also	onRequestGenerateTransportKey
Exmple data	<p>onRequestGenerateTransportKey  result:390CE532205DE88E7F8B713527B4EA333ECC122  133D39725B5D72D012E36EA736A8865259713C2DEEE2  33E5BF7F940B4C4583C570C28C82ED3CB03D53919D0  BAA41BA90A9429B81AA1A9DFCB56DF4D350CCCF8  DDBA4D75DBA928CFFC33A4B1B9DFC580CBA34F1B  6E8752565622C606CE7D442248FE39DA551CC5481D53  F4539A</p> <p>/com.dspread.demoui D/POS_SDK: buffer:  229279554278282164552639152755323120960999101795  343691155748798250587365222587650371312677802138  126191059275870574890923331610493022385892345779  459702394270879814244748352069111116056878737185  532071846202252684560390146332477338246621878771  054158830191791270694079345427843541353397122076  07202837129624038033</p>

	2021-03-16 19:26:12.909 9563-9563/com.dspread.demoui D/POS_SDK: decrypted result: F692C5AF86F8D4DB88B87BBDDF1BB5C6
--	--

### 2.6.87 readGasCard

Signature	void readGasCard(String cardType, String address, int readLen, int timeout)
Inputs	cardType: 0x00- FDW4442 0x01- FDW4428  address: The address of reading data  readLen: The length of reading data  timeout: overtime
Outputs	None
Description	Read Gas Card
See also	onReadGasCardResult
Exmple data	GasCardData==06400400400200000040000000400500

### 2.6.88 writeGasCard

Signature	void writeGasCard(String cardType, String password, String address, int dataLen, String data, int timeout)
Inputs	cardType: 0x00- FDW4442 0x01- FDW4428  password: FDW4428-2 bytes, FDW4442-3 bytes  address: The address of writing data  dataLen: The length of writing data  data: The data of writing  timeout: overtime
Outputs	None

Description	Write Gas Card
See also	onWriteGasCardResult
Exmple data	com.dspreload.demoui.debug I/POS_SDK: WriteCard Success

### 2.6.89 updateIPEKByTransportKey

Signature	void updateIPEKByTransportKey(String ipekgroup,String trackksn,String trackipek,String trackipekCheckvalue,String emvksn,,String emvipek,String emvipekCheckvalue,String pinksn,String pinipek,String pinipekCheckvalue)
Inputs	Ipekgroup 1 byte, trackksn 10 byte, trackipek 16byte, trackipekCheckvalue 8byte, emvksn 10byte, emvipek 16byte, emvipekCheckvalue 8byte, pinksn 10byte, pinipek 16byte, pinipekCheckvalue 8byte
Outputs	None
Description	Use transport key to 3des ecb ipek plaintext and update ipek
See also	onReturnUpdateIPEKResult
Example data	ipekgroup: 00ipekgroup: 09118012400705E00000trackksn: 09118012400705E00000trackipek: 4B7F6768C67C85AE4B7F6768C67C85AEtrackipekcheck value: 82E13665B4624DF5emvksn: 09118012400705E00000emvipek: 4B7F6768C67C85AE4B7F6768C67C85Aeemvipekcheckv alue: 82E13665B4624DF5pinksn: 09118012400705E00000pinipek: 4B7F6768C67C85AE4B7F6768C67C85Aepinipekcheckval ue: 82E13665B4624DF5  11959-11959/com.dspreload.demoui D/POS_SDK: onReturnUpdateIPEKResult(Boolean arg0):true

### 2.6.90 updateWorkKeyByTransportKey

Signature	void updateWorkKeyByTransportKey(String pik,String pikCheck,String trk,String trkCheck,String mak,String
-----------	--

	makCheck ,int keyIndex,int timeout)
Inputs	Pinkey 16byte, pinkeyCheckValue 8byte, trackKey 16byte, trackKeyCheckValue 8byte, macKey 16byte, macKeyCheckValue 8byte, keyIndex, timeout
Outputs	None
Description	Use transport key to 3des ecb work key plaintext and update work key
See also	onRequestUpdateWorkKeyResult
Example data	<p>com.dspread.demoui D/POS_SDK: pinKey: 42F53F7DE22B029842F53F7DE22B0298pinKeyCheckValue: 82E13665B4624DF5trackKey: 42F53F7DE22B029842F53F7DE22B0298trackKeyCheckValue: 82E13665B4624DF5macey: 42F53F7DE22B029842F53F7DE22B0298macKeyCheckValue: 82E13665B4624DF5</p> <p>2021-03-31 16:43:15.386 8510-8510/com.dspread.demoui D/POS_SDK: onRequestUpdateWorkKeyResult(UpdateInformationResult result):UPDATE_SUCCESS</p>

### 2.6.91 sendCVV

Signature	Public boolean sendCVV(String cvvString)
Inputs	CVV String
Outputs	Result of call the sendCVV function
Description	Send CVV to reader
See also	None
Example data	<p>10212-10212/com.dspread.demoui D/POS_SDK: onRequestOnlineProcess()</p> <p>10212-10212/com.dspread.demoui D/POS_SDK: send cvv to pos</p>

## 2.6.92 getEncryptedDataBlock

Signature	Hashtable<String, String> getEncryptedDataBlock(int keyIndex)
Inputs	keyIndex 0 ~ 9
Outputs	Hashtable of encrypted data block
Description	Get the corresponding data according to DUKPT, MK/SK
See also	None
Example data	2021-03-31 16:47:18.807 10212-10212/com.dspreload.demoui D/POS_SDK: hashtable: { KCV=C80059, CRC32=AD1B58EB, KSN=09118012400705E00001, CardSensitiveData=47EC31C89D84234D32AE3F070DC62F295BFF6BF6F58B6164B5F0E93EA67ED900D47B94FD31C035FF08781A78F61E3D894DE7FE55EEBF3A5F0BAE66CD612391FEF3ABDA12666459B02E8EE6A7A69A03D1914E4677C38C4644CF4A552000D7D5B6A3AA10AC61B88EE5 }

## 2.6.93 displayQRCode

Signature	Public void displayQRCode(String customDisplayStr, String qrLink)
Inputs	customDisplayStr  qrLink
Outputs	None
Description	Display QR code on the pos
See also	Public void onReturnDisplayQRCodeResult(Boolean result)
Example data	com.dspreload.demoui D/POS_SDK: customDisplayStr: 999999999999 qrLink: https://www.dspreload.com  com.dspreload.demoui D/POS_SDK: onReturnDisplayQRCode result: success



## 2.6.94 operateLEDByType

Signature	Public void operateLEDByType (LEDType ledType,String colorValue,LEDDirection ledDirection, LEDStatus status,int lightTime, int lightOffTime, int blinksTimes)
Inputs	<p>ledType: FIXED_COLOR/RGB_COLOR</p> <p>colorValue: if the ledType is FIXED_COLOR, you can fill in the below color value</p> <p>"0000": blue led</p> <p>"0001": yellow led</p> <p>"0002": green led</p> <p>"0003": red led</p> <p>if the ledType is RGB_COLOR, you can fill in the rgb565 format value, like 3666</p> <p>ledDirection: ALL/LEFT/RIGHT/UP/DOWN</p> <p>status: ON/OFF/BLINKS</p> <p>lightTime: the led light on time when the status param is "BLINKS"</p> <p>lightOffTime: the led light off time when the status param is "BLINKS"</p> <p>blinksTimes: the led blinks times when the status param is "BLINKS"</p>
Outputs	None
Description	To operate the different color LED according type
See also	Public void onReturnOperateLEDByTypeResult(boolean result)

## 2.6.95 playBuzzerByType

Signature	playBuzzerByType(BuzzerType type,int buzzerOnTime, int buzzerOffTime, int buzzerTimes)
Inputs	<p>type: COMMON/ PAYMENT_SUCCESS/ PAYMENT_ERROR/ PAIRING_SUCCESS</p> <p>COMMON: play the common type buzzer</p> <p>PAYMENT_SUCCESS: play the "PAYMENT_SUCCESS" buzzer that MP shared</p> <p>PAYMENT_ERROR: play the "PAYMENT_ERROR" buzzer that MP shared</p> <p>PAIRING_SUCCESS: play the "PAIRING_SUCCESS" buzzer that MP shared</p> <p>buzzerOnTime: the time of the buzzer on when type is "COMMON"</p> <p>buzzerOffTime: the time of the buzzer off when type is "COMMON"</p> <p>buzzerTimes: the times of buzzer play on when type is "COMMON"</p>
Outputs	None
Description	To play different buzzer according type
See also	Public void onReturnPlayBuzzerByTypeResult(boolean result)

## 2.6.96 sendNfcProcessResult

Signature	Public void sendNfcProcessResult(String tlv)
Inputs	Tlv(8A+91+71/72 tag)
Outputs	None
Description	Perform double tap

See also	Public void onRequestNFCBatchData(TransactionResult result, String tlv)
Example data	pos.sendNfcProcessResult("8A02303091100102030405060708000000000000000072189F180400000001860F8C2400000A8E081122334455667788");

### 2.6.97 getICCTag

Signature	public Hashtable<String, String> getICCTag(EncryptType encryType, int cardType, int tagCount, String tagArrStr)
Inputs	EncryptType:  ENCRYPTED means the data is encrypted; PLAINTEXT means the data is plaintext;  cardType:  <b>0</b> means the card type is contact; <b>1</b> means the card type is contactless;  tagCount: number of tags;  tagArrStr: tags string;
Outputs	The tlv format data corresponding to the incoming tags
Description	This method is used in getting the tlv format data of the corresponding tags
See also	None
Example data	pos.getICCTag(QPOSService.EncryptType.ENCRYPTED, 1, 2, "5A57");

### 2.6.98 getEncryptedTrack2Data

Signature	Public Hashtable<String, String> getEncryptedTrack2Data()
Inputs	None

Outputs	Ksn and EncryptedTrack2data
Description	This method is used in getting ksn and EncryptedTrack2data
See also	None

### 2.6.99 lcdShowCustomDisplay

Signature	Public void lcdShowCustomDisplay(LcdModeAlign lcdModeAlign, String lcdFont, int timeout)
Inputs	LcdModeAlign: The alignment of the displayed content  LCD_MODE_ALIGNCENTER; LCD_MODE_ALIGNLEFT; LCD_MODE_ALIGNRIGHT;  lcdFont;  timeout;
Outputs	None
Description	This method is used in setting the lcd display on the device screen
See also	protected void onLcdShowCustomDisplay(final boolean isSuccess)
Example data	pos.lcdShowCustomDisplay(LcdModeAlign.LCD_MODE_ALIGNCENTER, customDisplayString, 60);

### 2.6.100 setCustomLogoDisplay

Signature	Public void setCustomLogoDisplay(LcdModeAlign lcdModeAlign, String customLogoStr, int timeout)
Inputs	LcdModeAlign: The alignment of the displayed logo.  LCD_MODE_ALIGNCENTER; LCD_MODE_ALIGNLEFT;

	LCD_MODE_ALIGNRIGHT;  customLogoStr: The logo you want to display on the main interface; If an empty string is passed in, it will shows the default logo;  timeout;
Outputs	None
Description	This method is used in dynamic modify logo
See also	Public void onSetCustomLogoDisplay(boolean result)
Example data	pos.setCustomLogoDisplay(LcdModeAlign.LCD_MODE_ALIGNRIGHT, "WELCOME", 60)

### 2.6.101 getEncryptData

Signature	public Hashtable<String, String> getEncryptData()
Inputs	None
Outputs	RandomData, PAN, AESKey, isOnlinePin, pinTryLimit
Description	This method is used to get the data which used to encrypt PIN
See also	None
Example data	pos.getEncryptData()

### 2.6.102 sendCvmPin

Signature	public void sendCvmPin(String pinblock, boolean isEncrypted)
Inputs	pinblock : ISO format 4 pinblock ; isEncrypted: True/False
Outputs	None

Description	This method is used to transfer pinblock to device
See also	None
Example data	pos.sendCvmPin();

### 2.6.103 initListener

Signature	public void initListener(QPOSServiceListener listener)
Inputs	listener
Outputs	None
Description	This method is used to init a listener
See also	None
Example data	pos.initListener(listener);

### 2.6.104 openUsb

Signature	public void openUsb(UsbDevice usbDevice)
Inputs	usbDevice
Outputs	None
Description	This method is used to open USB
See also	None
Example data	pos.openUsb(usbDevice);

## 2.6.105 openUart

Signature	public void openUart()
Inputs	None
Outputs	None
Description	This method is used to open serial port
See also	None
Example data	pos.openUart();

## 2.6.106 bypassPin

Signature	public void bypassPin()
Inputs	None
Outputs	None
Description	This method is used to bypass PIN
See also	None
Example data	pos.bypassPin();

## 2.6.107 cancelPin

Signature	public void cancelPin()
Inputs	None
Outputs	None
Description	This method is used to cancel PIN
See also	None



Example data	pos.cancelPin();
--------------	------------------

### 2.6.108 pinMapSync

Signature	public void pinMapSync(String datas, int timeout)
Inputs	PIN coordinates; time
Outputs	None
Description	This method is used to synchronize PIN coordinates to firmware for Smart POS
See also	None
Example data	pos.pinMapSync(value, 20);

### 2.6.109 updateKeyByTR\_31

Signature	public void updateKeyByTR_31(String keyBlock, int timeout)
Inputs	TR31 keyBlock; time
Outputs	None
Description	This method is used to update IPEK
See also	None
Example data	pos.updateKeyByTR_31(data, 30);

### 2.6.110 updateEMVConfigByXml

Signature	public void updateEMVConfigByXml(String xmlContent)
-----------	---





Inputs	xmlContent
Outputs	None
Description	This method is used to update emv
See also	None
Example data	pos.updateEMVConfigByXml(new String(FileUtils.readAssetsLine("mini_emv_app_config.xml"))

### 2.6.111 updatePosFirmware

Signature	public int updatePosFirmware(byte[] data, String deviceAddress)
Inputs	Firmware data; blueTootchAddress
Outputs	None
Description	This method is used to update firmware
See also	None
Example data	pos.updatePosFirmware(data, blueTootchAddress);

### 2.6.112 pollOnMifareCard

Signature	public void pollOnMifareCard(int timeout)
Inputs	time
Outputs	None
Description	This method is used to power on and detect Mifare card
See also	None
Example data	pos.pollOnMifareCard(20);

## 2.6.113 authenticateMifareCard

Signature	public void authenticateMifareCard(MifareCardType cardType, String keyType, String block, String keyValue, int timeout)
Inputs	MifareCardType cardType: CLASSIC、UITRALIGHT keyType: Key A、Key B block keyValue time
Outputs	None
Description	This method is used to verify Key A/B
See also	None
Example data	pos.authenticateMifareCard(QPOSService.MifareCardType. CLASSIC, keyclass, blockaddr, keyValue, 20);

## 2.6.114 operateMifareCardData

Signature	public void operateMifareCardData(MifareCardOperationType type, String block, String data, int timeout)
Inputs	MifareCardType cardType: CLASSIC、UITRALIGHT block data time
Outputs	None
Description	This method is used to add/reduce/restore Mifare card
See also	None
Example data	pos.operateMifareCardData(QPOSService.MifareCardOperat ionType.ADD, blockaddr, cardData, 20);

## 2.6.115 readMifareCard

Signature	pos.readMifareCard(MifareCardType cardType,String block,int timeout)
Inputs	MifareCardType cardType: CLASSIC、UITRALIGHT block time
Outputs	None
Description	This method is used to read Mifare card
See also	None
Example data	pos.readMifareCard(QPOSService.MifareCardType.CLASSIC, blockaddr, 20);

## 2.6.116 writeMifareCard

Signature	public void writeMifareCard(MifareCardType cardType, String block, String data, int timeout)
Inputs	MifareCardType cardType: CLASSIC、UITRALIGHT block data time
Outputs	None
Description	This method is used to write Mifare card
See also	None
Example data	pos.writeMifareCard(QPOSService.MifareCardType.CLASSIC, blockaddr, cardData, 20);

## 2.6.117 finishMifareCard

Signature	public void finishMifareCard(int timeout)
-----------	---

Inputs	time
Outputs	None
Description	This method is used to finish the operation for Mifare card
See also	None
Example data	pos.finishMifareCard(20);

### 2.6.118 fastReadMifareCardData

Signature	public void fastReadMifareCardData(String startBlock, String endBlock, int timeout)
Inputs	startBlock; endBlock; time
Outputs	None
Description	This method is used to read Mifare Ultralight card
See also	None
Example data	pos.fastReadMifareCardData(startAddr, endAddr, 20);

### 2.6.119 getDeviceECCPublicKey

Signature	public void getDeviceECCPublicKey(int timeout)
Inputs	time
Outputs	None
Description	get the device ECC pulic key value
See also	None
example	pos.getDeviceECCPublicKey(10);

## 2.6.120 updateServerECCPublicKey

Signature	public void updateServerECCPublicKey(String serverPublicKeyStr, int timeout)
Inputs	serverPublicKeyStr: "30819b...eed29" timeout :time
Outputs	None
Description	update the server ECC pulic key value
See also	None
example	pos.updateServerECCPublicKey("30819b...eed29",10);

## 2.6.121 updateKeyByTR\_31

Signature	public void updateKeyByTR_31(int keyIndex, String ksn, String keyBlock)
Inputs	keyIndex: 0 ksn:"FFFFFFFFF1045200000001E" keyBlock:"D0112B1AX00E000092D825CC86CDB6167B01FAD5E038575C0D73AC7E485FC39543B5520D54A4BEFBEE7E7889531091855303EFB1CD12CFD8"
Outputs	None
Description	update the TR31 key block and ksn
See also	None
example	pos.updateKeyByTR_31(0,"FFFFFFFFF1045200000001E", "D0112B1AX00E000092D825CC86CDB6167B01FAD5E038575C0D73AC7E485FC39543B5520D54A4BEFBEE7E7889531091855303EFB1CD12CFD8")

## 2.6.122 calculateMacWithKey

Signature	public Hashtable<String,String> calculateMacWithKey(KeyPart keyPart, CryptMode cryptMode, KeyManager keyManager, KeyType keyType, String data, boolean isAddKSN)
Inputs	keyPart: QPOSService.KeyPart.KEY_ALL cryptMode: QPOSService.CryptMode.AES_CBC_ENCRYPT keyManager:QPOSService.KeyManager.DUKPT_KEY keyType: QPOSService.KeyType.TRACK_KEY data:"22222222222222222222" isAddKSN:false
Outputs	None
Description	Use different key to calculate mac
See also	None
example	Hashtable<String,String> result= pos.calculateMacWithKey(QPOSService.KeyPart.KEY_ALL, QPOSService.CryptMode.AES_CBC_ENCRYPT, QPOSService.KeyManager.DUKPT_KEY, QPOSService.KeyType.TRACK_KEY,"22222222222222222222",false) String encryptedData = result.get("result") String ksn = result.get("ksn")

## 2.6.123 getEncryptedDataWithTag

Signature	public Hashtable<String,String> getEncryptedDataWithTag(String tagName, int cardType, CryptMode cryptMode)
Inputs	tagName: tag name, ex: 57 cardType: 0 is IC, 1 is NFC cryptMode: QPOSService.CryptMode.AES_CBC_ENCRYPT

Outputs	None
Description	get the certain encrypted tag value
See also	None
example	<pre>Hashtable&lt;String,String&gt; data = pos.getEncryptedDataWith Tag("57",0, QPOSService.CryptMode.AES_CBC_ENCRYP T);</pre>

### 2.6.124 cbc\_mac\_cn

Signature	cbc_mac_cn(int keyLen, int algorithmType, int operatorType, string data, int timeout)
Inputs	keyLen: the length of key and unit is byte algorithmType: 0 operatorType: 0 data: eg: 782bbabd8d2337fcfe1d0b9a1d71c94184ae16cbc4bd682f950d4918fa06fcda timeout: time that unit is second
Outputs	None
Description	Calculate the mac value
See also	None
example	<pre>pos.cbc_mac_cn(32,0,0,"782bbabd8d2337fcfe1d0b9a1d71c 94184ae16cbc4bd682f950d4918fa06fcda",10);</pre>

### 2.6.125 cbc\_mac\_cnAll

Signature	cbc_mac_cnAll(int keyLen, int algorithmType, int operatorType, string data, int timeout)
Inputs	keyLen: the length of key and unit is bytes algorithmType: 0 operatorType: 0 data: 0123456789ABCDEF



	timeout: time that unit is second
Outputs	None
Description	Calculate the mac value
See also	None
example	<code>pos.cbc_mac_cnAll(32,0,0,"0123456789ABCDEF",5);</code>

## 2.7 Delegate Methods Reference

### 2.7.1 onRequestWaitingUser

Signature	<code>void onRequestWaitingUser()</code>
Inputs	None
Outputs	None
Description	In the callback to waiting user operating
See also	

### 2.7.2 onQposIdResult

Signature	<code>void onQposIdResult (Hashtable&lt;String, String&gt; posIdTable)</code>
Inputs	<code>Hashtable&lt;String, String&gt; posIdTable</code>
Outputs	None
Description	Callback of pos id response
See also	<code>getQposId()</code>

### 2.7.3 onQposInfoResult

Signature	<code>void onQposInfoResult (Hashtable&lt;String, String&gt;</code>
-----------	---



	posIdTable)
Inputs	Hashtable<String, String> posIdTable
Outputs	None
Description	Callback of pos info response
See also	getQposInfo()

## 2.7.4 onDoTradeResult

Signature	void onDoTradeResult (DoTradeResult result, Hashtable<String, String> decodeData)
Inputs	DoTradeResult result Hashtable<String, String> decodeData
Outputs	None
Description	Callback of track data response
See also	doTrade()  doEmvApp(EmvOption emvOption)

## 2.7.5 onRequestSetAmount

Signature	void onRequestSetAmount ()
Inputs	None
Outputs	None
Description	The callback of request user set amount
See also	doTrade()  setAmount(String amount, String cashbackAmount, String currencyCode, TransactionType transactionType)

## 2.7.6 onRequestSelectEmvApp

Signature	void onRequestSelectEmvApp (ArrayList<String> appList)
Inputs	ArrayList<String> appList
Outputs	None
Description	Callback method of request user to select emv app
See also	doTrade()  selectEmvApp(int index)  cancelSelectEmvApp()

## 2.7.7 onRequestIsServerConnected

Signature	void onRequestIsServerConnected ()
Inputs	None
Outputs	None
Description	In the method judge the network state
See also	doTrade()  isServerConnected(boolean isConnected)

## 2.7.8 onRequestFinalConfirm

Signature	void onRequestFinalConfirm ()
Inputs	None
Outputs	None
Description	Final confirm amount response
See also	doTrade()  finalConfirm (boolean isConfirmed)

## 2.7.9 onRequestOnlineProcess

Signature	void onRequestOnlineProcess (String tlv)
Inputs	Tlv type+length+value
Outputs	None
Description	Response of request data to server
See also	doTrade()  anlysEmvIccData ()  sendOnlineProcessResult()

## 2.7.10 onRequestTime

Signature	void onRequestTime ()
Inputs	None
Outputs	None
Description	Set terminal time response
See also	doTrade()  sendTime()

## 2.7.11 onRequestTransactionResult

Signature	void onRequestTransactionResult (TransactionResult transactionResult)
Inputs	None
Outputs	None
Description	Transaction response
See also	doTrade()

### 2.7.12 onRequestTransactionLog

Signature	void onRequestTransactionResult (String tlv)
Inputs	Tlv type+length+value
Outputs	None
Description	reserve
See also	

### 2.7.13 onRequestBatchData

Signature	void onRequestBatchData (String tlv)
Inputs	Tlv type+length+value
Outputs	None
Description	Response of ICC transactions
See also	doTrade()

### 2.7.14 onRequestQposConnected

Signature	void onRequestQposConnected ()
Inputs	None
Outputs	None
Description	Callback of connected success
See also	connectBluetoothDevice()

### 2.7.15 onRequestQposDisconnected

Signature	void onRequestQposDisconnected ()
Inputs	None

Outputs	None
Description	Callback of disconnect
See also	disconnectBT()

### 2.7.16 onRequestNoQposDetected

Signature	void onRequestNoQposDetected ()
Inputs	None
Outputs	None
Description	Callback of connected fail
See also	connectBluetoothDevice()

### 2.7.17 onError

Signature	void onError (Error errorState)
Inputs	enum Error errorState error type information
Outputs	None
Description	SDK error information response
See also	

### 2.7.18 onRequestDisplay

Signature	void onRequestDisplay (Display displayMsg)
Inputs	enum Display displayMsg: display content
Outputs	None
Description	Notify the terminal displays the current related content
See also	

### 2.7.19 onRequestUpdateWorkKeyResult

Signature	void onRequestUpdateWorkKeyResult (UpdateInformationResult result)
Inputs	enum UpdateInformationResult result
Outputs	None
Description	Update work key response
See also	updateWorkKey()

### 2.7.20 onGetCardNoResult

Signature	void onGetCardNoResult (String result)
Inputs	String result of the cardPan.
Outputs	None
Description	Get Card pan response
See also	getCardNo()

### 2.7.21 onReturnReversalData

Signature	void onReturnReversalData (String tlv)
Inputs	tlv : type+length+value
Outputs	None
Description	ic card reversal data response
See also	doEmvApp()

### 2.7.22 onReturnGetPinResult

Signature	void onReturnGetPinResult (Hashtable<String, String> result)
-----------	--

Inputs	Hashtable<String, String> result 包含pinKsn 和 pinBlock
Outputs	None
Description	Get pin response
See also	getPin()

### 2.7.23 onReturnPowerOnIccResult

Signature	void onReturnPowerOnIccResult (boolean result, String ksn, String atr, int atrLen)
Inputs	result: true – success, false – failure ksn: EMV KSN used for encryption of ATR data and APDU data. If ksn is all FF, then ATR and APDU data are not encrypted. atr: data returned in ATR atrLen: length of the ATR data
Outputs	None
Description	Respond to powerOnIcc. If ksn is all FF, then ATR and APDU are not encrypted. Otherwise, ATR and APDU are encrypted by the key derived from EMV KSN.
See also	powerOnIcc()

### 2.7.24 onReturnPowerOffIccResult

Signature	void onReturnPowerOffIccResult (boolean result)
Inputs	result: true – success, false – failure
Outputs	None
Description	Respond to powerOffIcc.
See also	powerOffIcc()

## 2.7.25 onReturnApuResult

Signature	void onReturnApuResult (boolean isSuccess, String apdu, int apduLen)
Inputs	isSuccess: true– success, false – failure apdu: data returned apduLength: length of the apdu data
Outputs	None
Description	Return data in response to the level 1 EMV method sendApu. If the apdu data are encrypted, the KSN returned after the powerOnIcc command is used for encryption.
See also	sendApuByNFC() sendApu()

## 2.7.26 onReturnSetSleepTimeResult

Signature	void onReturnSetSleepTimeResult (boolean isSuccess)
Inputs	isSuccess: true– success, false – failure
Outputs	None
Description	The response of set device sleep time
See also	setPosSleepTime()

## 2.7.27 onRequestCalculateMac

Signature	void onRequestCalculateMac (String calMac)
Inputs	calMac: calculate mac result data
Outputs	None
Description	Calculate mac response method
See also	doCalculateMac() doDoubleMac()



	doSingleMac()
--	---------------

### 2.7.28 onReturnCustomConfigResult

Signature	void onReturnCustomConfigResult (boolean isSuccess, String result)
Inputs	isSuccess: true– success, false – failure result : data of result
Outputs	None
Description	Save Emv Config response
See also	updateEmvConfig()

### 2.7.29 onReturnSetMasterKeyResult

Signature	void onReturnSetMasterKeyResult (boolean isSuccess)
Inputs	isSuccess: true– success, false – failure
Outputs	None
Description	Set masterKey response
See also	setMasterKey()

### 2.7.30 onReturnBatchSendAPDUResult

Signature	void onReturnBatchSendAPDUResult (LinkedHashMap<Integer, String> batchAPDUResult)
Inputs	LinkedHashMap<Integer, String> batchAPDUResult
Outputs	None
Description	Response of batch APDU instruction execution
See also	DoVIPOSBatchSendApdu()

### 2.7.31 onReturniccCashBack

Signature	void onReturniccCashBack (Hashtable<String, String> result)
Inputs	Hashtable<String, String> result
Outputs	None
Description	Cash back response (Special needs)
See also	iccCashBack()  getIccCardNo()  inquireECQAmount()

### 2.7.32 onUpdatePosFirmwareResult

Signature	void onUpdatePosFirmwareResult(UpdateInformationResult result)
Inputs	enum UpdateInformationResult result
Outputs	None
Description	Response of upgrade the firmware
See also	updatePosFirmware()

### 2.7.33 onReturnDownloadRsaPublicKey

Signature	void onReturnDownloadRsaPublicKey (HashMap<String, String> map)
Inputs	HashMap<String, String> map
Outputs	None
Description	Callback of encrypt random number using RSA public key
See also	downloadRsaPublicKey()

### 2.7.34 onPinKey\_TDES\_Result

Signature	void onPinKey_TDES_Result (String result)
Inputs	result: data of response
Outputs	None
Description	Response of 3DES encrypt pin
See also	pinKey_TDES()

### 2.7.35 onUpdateMasterKeyResult

Signature	void onUpdateMasterKeyResult (boolean result, Hashtable<String, String> resultTable)
Inputs	boolean result Hashtable<String, String> resultTable
Outputs	None
Description	Update masterKey response
See also	updateMasterKeyRandom() updateMasterKey()

### 2.7.36 onEmvICCExceptionData

Signature	void onEmvICCExceptionData (String tlv)
Inputs	Tlv : type+length+value
Outputs	None
Description	Emv ICC Exception response
See also	

### 2.7.37 onSetManagementKey

Signature	void onSetManagementKey (boolean b)
Inputs	b:the flag of the key if is setted successful
Outputs	None
Description	The response of SetManagementKey
See also	

### 2.7.38 onSetBuzzerResult

Signature	void onSetBuzzerResult(boolean b)
Inputs	b:the flag of the buzzer if is setted successful
Outputs	None
Description	The response of SetBuzzer
See also	

### 2.7.39 onReturnUpdateIPEKResult

Signature	void onReturnUpdateIPEKResult(boolean b)
Inputs	None
Outputs	b:the flag to checking updating IPEK successful or not
Description	The response of updateIPEK
See also	

### 2.7.40 onReturnUpdateEMVResult

Signature	void onReturnUpdateEMVResult(boolean isSuccess)
Inputs	isSuccess:the flag of updating EMV that id is AID if is

	successful
Outputs	None
Description	The response of updateEMV that id is AID
See also	

### 2.7.41 onReturnUpdateEMVRIDResult

Signature	void onReturnUpdateEMVRIDResult(boolean isSuccess)
Inputs	isSuccess:the flag of updating EMV that id is RID if is successful
Outputs	None
Description	The response of updateEMV that id is RID
See also	

### 2.7.42 onRequestDeviceScanFinished

Signature	void onRequestDeviceScanFinished()
Inputs	None
Outputs	None
Description	The response of scan device finished
See also	

### 2.7.43 onDeviceFound

Signature	void onDeviceFound(BluetoothDevice arg0)
Inputs	Arg0: is the bluetoothDevice that searched
Outputs	None
Description	The response of scanQpos2mode

See also	The method of scanQpos2mode
----------	-----------------------------

## 2.7.44 onRequestUpdateKey

Signature	void onRequestUpdateKey(String result)
Inputs	The updated key
Outputs	None
Description	The response of getUpdateKey
See also	getUpdateKey

## 2.7.45 onSetSleepModeTime

Signature	void onSetSleepModeTime(boolean isSuccess)
Inputs	isSuccess:the flag of setting sleepTime if is successful
Outputs	None
Description	The response of setSleepModeTime
See also	

## 2.7.46 onRequestGenerateTransportKey

Signature	void onRequestGenerateTransportKey (final Hashtable result)
Inputs	None
Outputs	Result include two parameter for transportKey and checkValue,details as below:  transportKey: transport key encrypted by the server's public key

	checkValue: transport key encrypted 8 byte 0
Description	The response of generateTransportKey
See also	

### 2.7.47 onReadGasCardResult

Signature	void onReadGasCardResult (boolean b, String result)
Inputs	None
Outputs	Return the data read by the gas card
Description	The response of readGasCard
See also	

### 2.7.48 onWriteGasCardResult

Signature	void onWriteGasCardResult (final boolean b)
Inputs	None
Outputs	Returns the status of write gas card
Description	The response of writeGasCard
See also	

### 2.7.49 onReturnOperateLEDByTypeResult

Signature	void onReturnOperateLEDByTypeResult (final boolean result)
Inputs	result: <b>true</b> means operate led success <b>false</b> means operate led failed
Outputs	None

Description	Returns the result of operate LED
See also	

### 2.7.50 onReturnPlayBuzzerByTypeResult

Signature	void onReturnPlayBuzzerByTypeResult (final boolean result)
Inputs	result: <b>true</b> means play buzzer success <b>false</b> means play buzzer failed
Outputs	None
Description	Returns the result of play buzzer
See also	

### 2.7.51 onReturnDisplayQRCodeResult

Signature	void onReturnDisplayQRCodeResult (final boolean b)
Inputs	None
Outputs	Returns the result of display QR code
Description	The response of displayQRCode
See also	

### 2.7.52 onReturnNFCBatchData

Signature	void onRequestNFCBatchData(TransactionResult result, String tlv)
Inputs	None
Outputs	Returns the result of double tap and tlv data
Description	The callback of sendNfcProcessResult
See also	



### 2.7.53 onLcdShowCustomDisplay

Signature	void onLcdShowCustomDisplay(final boolean result)
Inputs	None
Outputs	Returns the result of set custom lcd show display
Description	The callback of lcdShowCustomDisplay
See also	

### 2.7.54 onSetCustomLogoDisplay

Signature	void onSetCustomLogoDisplay(boolean result)
Inputs	None
Outputs	Returns the result of custom logo setting:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onSetCustomLogoDisplay
See also	

### 2.7.55 onSearchMifareCardResult

Signature	void onSearchMifareCardResult( Hashtable<String, String> cardData);
Inputs	None
Outputs	Returns the result of searching Mifare Card:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onSearchMifareCardResult

See also	pos.pollOnMifareCard()
----------	------------------------

## 2.7.56 onVerifyMifareCardResult

Signature	void onVerifyMifareCardResult(boolean flag);
Inputs	None
Outputs	Returns the result of verifying Mifare Card:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onVerifyMifareCardResult
See also	pos.authenticateMifareCard()

## 2.7.57 onOperateMifareCardResult

Signature	void onOperateMifareCardResult(Hashtable<String, String> flag);
Inputs	None
Outputs	Returns the result of operating Mifare Card:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onOperateMifareCardResult
See also	pos.operateMifareCardData()

## 2.7.58 onReadMifareCardResult

Signature	void onReadMifareCardResult(Hashtable<String, String> flag);
Inputs	None

Outputs	Returns the result of reading Mifare Card:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onReadMifareCardResult
See also	pos.readMifareCard()

### 2.7.59 onWriteMifareCardResult

Signature	void onWriteMifareCardResult(boolean flag);
Inputs	None
Outputs	Returns the result of writing Mifare Card:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onWriteMifareCardResult
See also	pos.writeMifareCard()

### 2.7.60 onFinishMifareCardResult

Signature	void onFinishMifareCardResult(boolean flag);
Inputs	None
Outputs	Returns the result of finishing Mifare Card:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onFinishMifareCardResult
See also	pos.finishMifareCard()

### 2.7.61 getMifareFastReadData

Signature	void getMifareFastReadData(Hashtable<String, String> flag);
Inputs	None
Outputs	Returns the result of getting MifareFastReadData:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of getMifareFastReadData
See also	pos.fastReadMifareCardData()

### 2.7.62 onReturnupdateKeyByTR\_31Result

Signature	void onReturnupdateKeyByTR_31Result(boolean result, String keyType);
Inputs	None
Outputs	Returns the result of updating key by TR_31:  <b>true</b> means success; <b>false</b> means failure;
Description	The callback of onReturnupdateKeyByTR_31Result
See also	pos.updateKeyByTR_31()

### 2.7.63 onQposGetDeviceECCPublicKeyResult

Signature	public void onQposGetDeviceECCPublicKeyResult(String devicePublicKey)
Inputs	None
Outputs	devicePublicKey: device PublicKey

Description	The callback of onQposGetDeviceECCPublicKeyResult
See also	pos.getDeviceECCPublicKey(10);

## 2.7.64 onQposUpdateServerECCPublicKeyResult

Signature	public void onQposUpdateServerECCPublicKeyResult(boolean result )
Inputs	None
Outputs	result: true or false
Description	The callback of onQposUpdateServerECCPublicKeyResult
See also	pos.updateServerECCPublicKey("30819b...eed29",10);

## 2.7.65 onReturnUpdateKeyByTR\_31Result

Signature	public void onReturnUpdateKeyByTR_31Result( boolean result)
Inputs	None
Outputs	result: true or false
Description	The callback of onReturnUpdateKeyByTR_31Result
Call back	public void onReturnUpdateKeyByTR_31Result( boolean result)
See also	pos.updateKeyByTR_31(0,"FFFFFFFFF1045200000001E","D0112B1AX00E000092D825CC86CDB6167B01FAD5E038575C0D73AC7E485FC39543B5520D54A4BEFBEE7E7889531091855303EFB1CD12CFD8")