QPOS SDK
Integration Guide

Document Version：<3.2>

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	2018.03.22	Wenluo Wang	Add setShutDownTime

Copyright notification

Table

1 Introduction

1.1 Summary

QPOS is a mobile payment card reader device with pinpad 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 pinpad. 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 pinpad and touch screen, SPOS can communicate with the mobile device through many methods, such as: audio jack, Bluetooth and

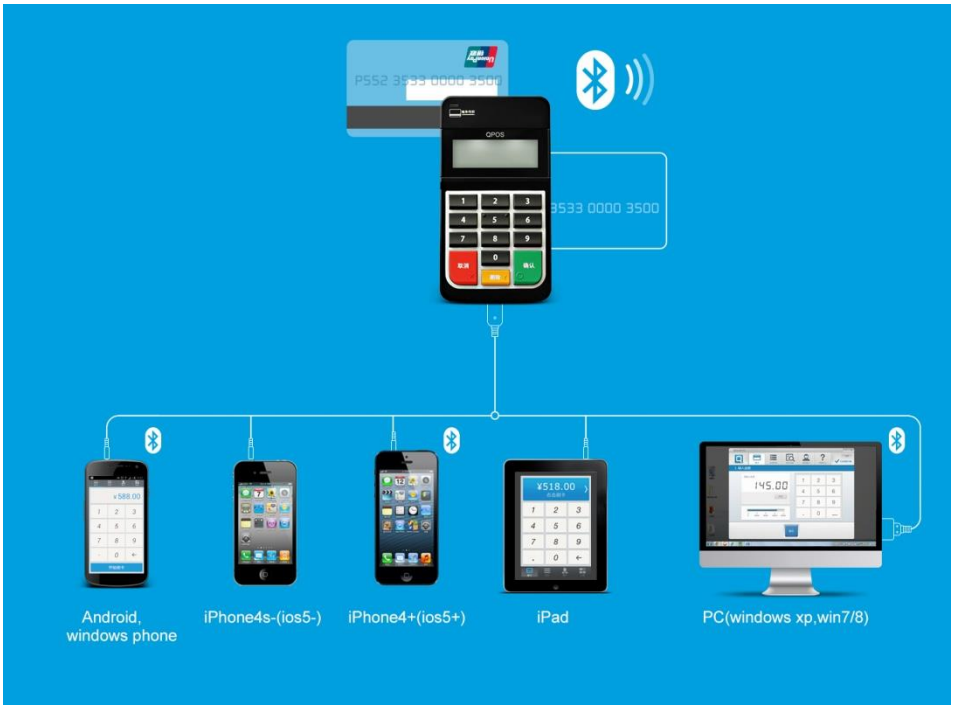
USB cable. The touch screen can be used to capture the signature of consumer in an electronic way.
QPOS share a lot in common, the SDK API is almost same for QPOS. In the following chapters.

Features:

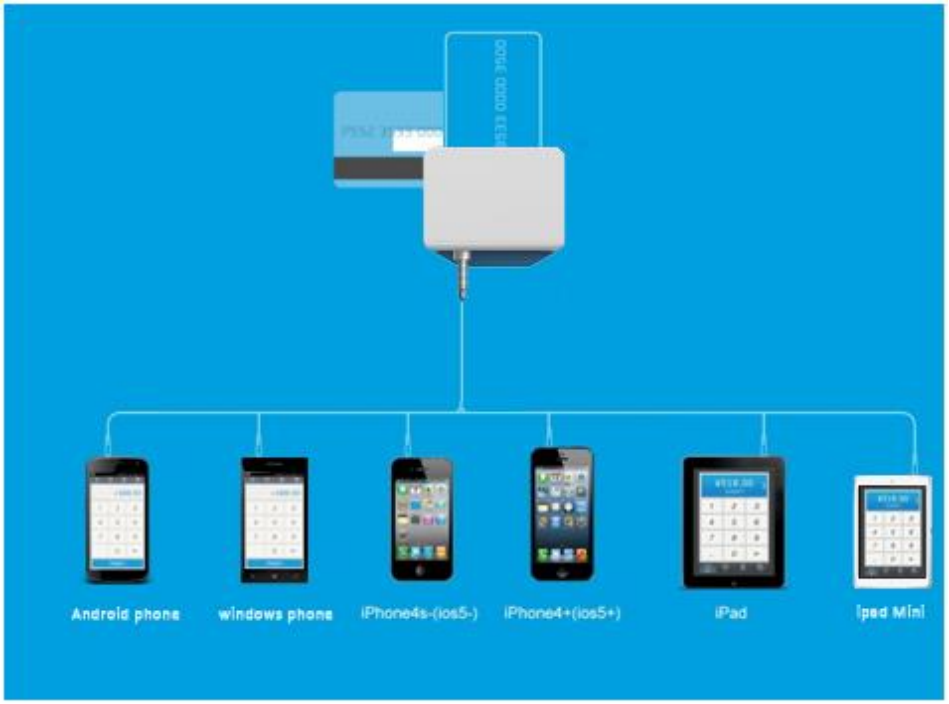
- Ensure secure transactions: integrated keyboard and multiple encryption algorithms to ensure secure transactions.
- Accept all types of bank card: supports magnetic stripe card, contact EMV IC card and contactless EMV IC card.
- Adapts to more smart devices through audio jack: Supports over 2,000 smart devices through audio jack.
- Fulfill 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

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[1]	Var	DATA-key	Triple-Des
0xC2	Online message[3]	Var	DATA-key	Triple-Des
0x70	Online EMV data message[2]	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 := 120700020000200100151
QPOS_BT_ID := QPOS0200100151

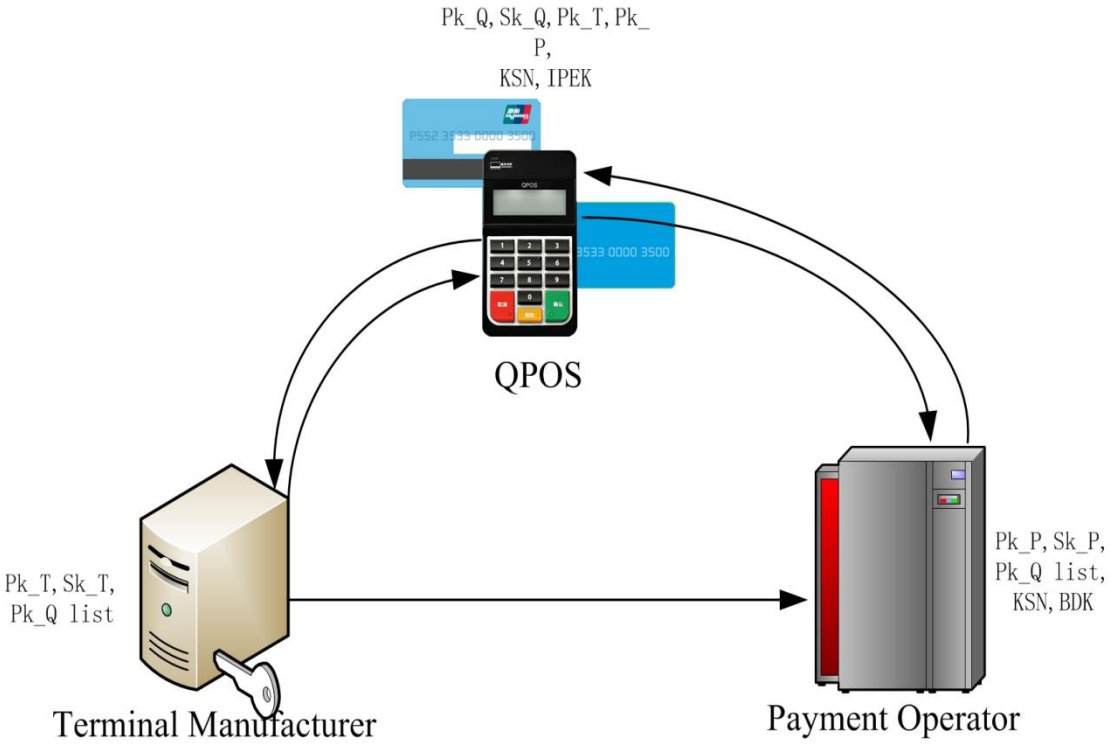
(The label is sticked on QPOS back cover)

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

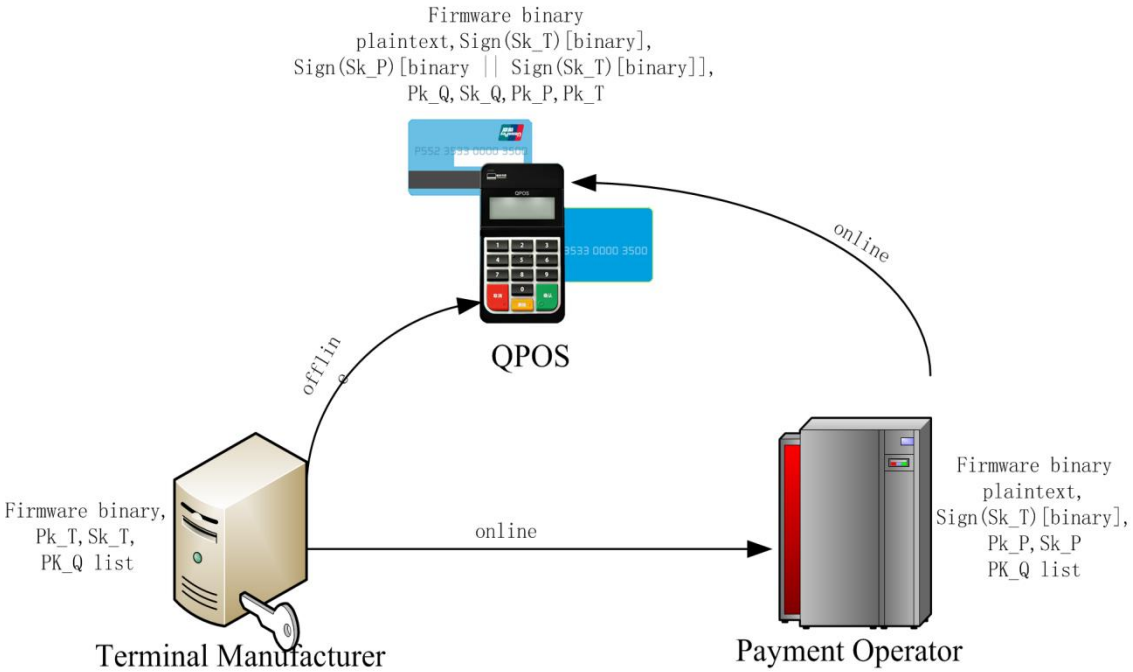
1.11.2 How to connect QPOS/SPOS through Bluetooth

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

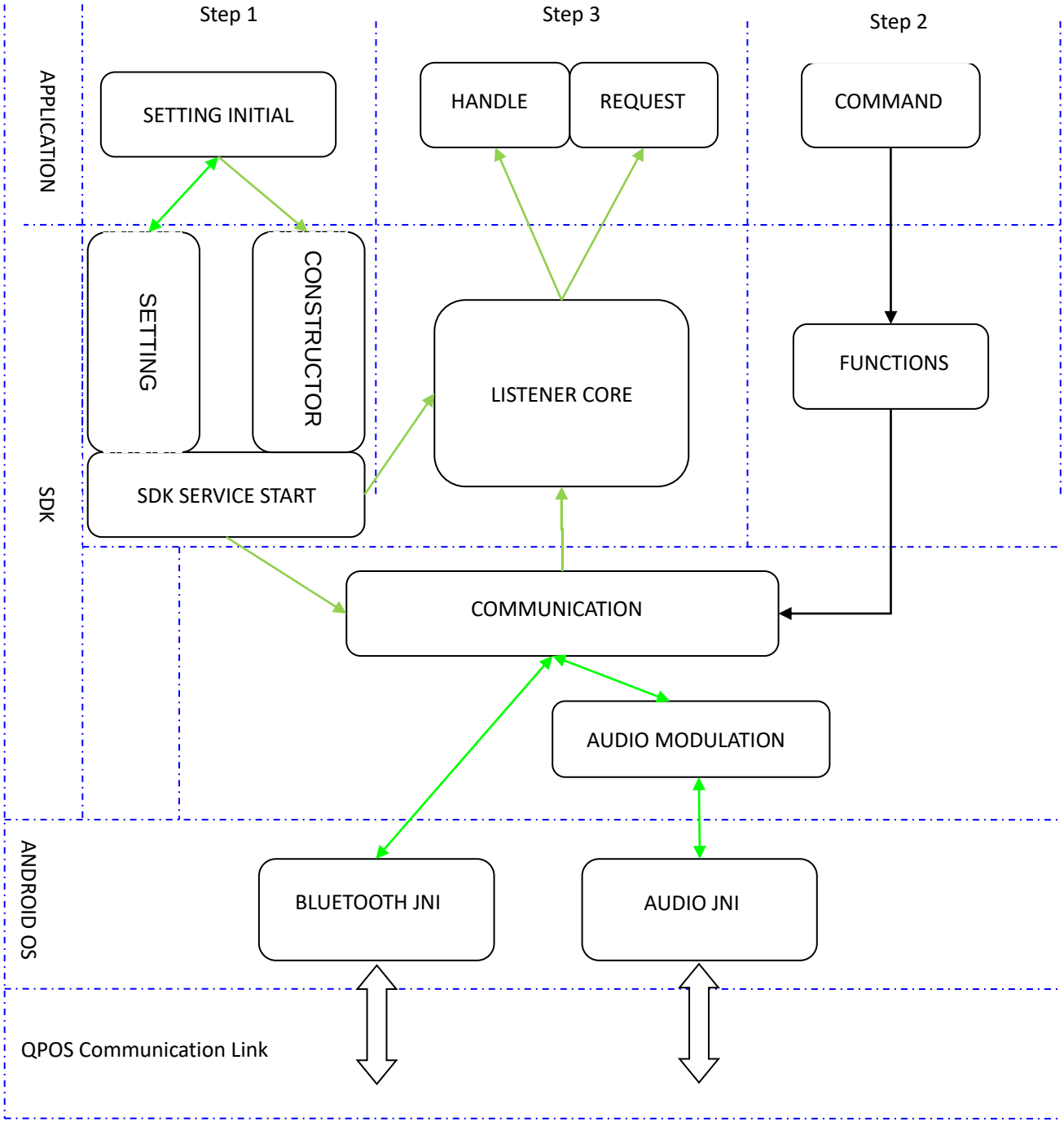
2.1.1 Development Environment

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 devided 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
    AUDIO// audio mode
}
```

Audio Mode:
QPOSService pos = QPOSService.getInstance(CommunicationMode.AUDIO);
pos.initListener(MainActivity.this, listener);

Bluetooth Mode:
QPOSService pos = QPOSService.getInstance(CommunicationMode.BLUETOOTH);
pos.initListener(MainActivity.this, 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 'YYMMDDHHMMSS' to Qpos,based the smart terminal date and time	←	onRequestTime
sendPin	Set the pin to QPOS Reader	←	onRequestSetPin

emptyPin	Set the empty pin to QPOS Reader	←	onRequestSetPin
cancelPin	Set cancel to QPOS Reader	←	onRequestSetPin
powerOnIcc	Turn on the EMV card.	←	onReturnPowerOnIccResult
powerOffIcc	Turn off the EMV card.	←	onReturnPowerOffIccResult
sendApdu	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	←	onReturnApduResult
setPosSleepTime	Set the pos sleep time	←	onReturnSetSleepTimeResult
udpateWorkKey	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.

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	→	selectEmvApp
onRequestIsServerConnected		m	→	isServerConnected
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		
onRequestSetPin	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.	o	→	sendPin emptyPin cancelPin
onReturnPowerOnIccResult	Turn on the EMV card result	o	→	powerOnIcc
onReturnPowerOffIccResult	Turn off the EMV card result	o	→	powerOffIcc
onReturnApduResult		o	→	sendApdu

onReturnSetSleepTimeResult	Set the pos sleep time	o	→	setPosSleepTime
onRequestUpdateWorkKeyResult	update the pos work key	o	→	udpateWorkKey
onReturnSetMasterKeyResult	update the pos work key	o	→	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.

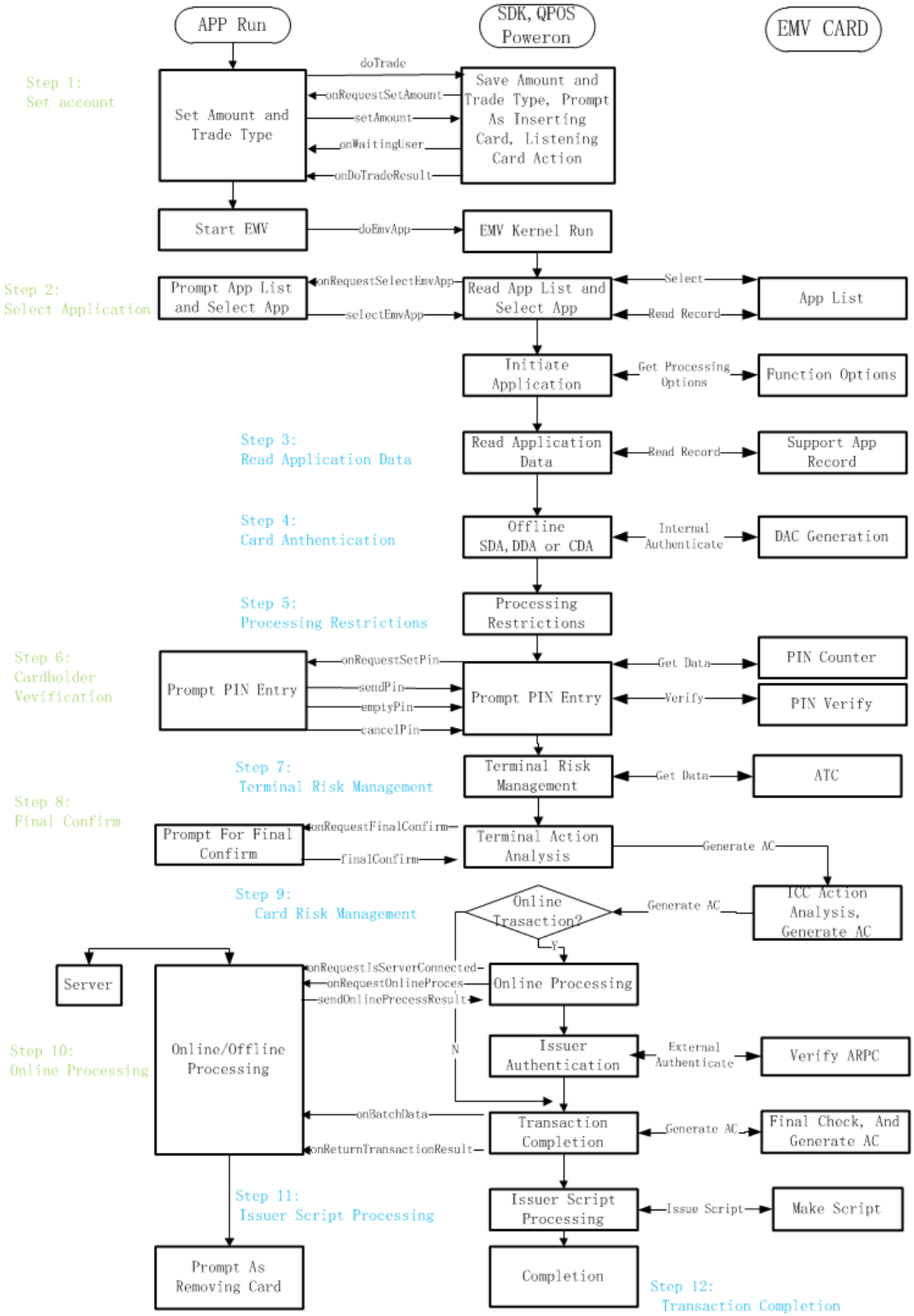
Public void **onDoTradeResult**(DoTradeResult result, Hashtable<String, String> decodeData);
Enum **DoTradeResult** can be any of following

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

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

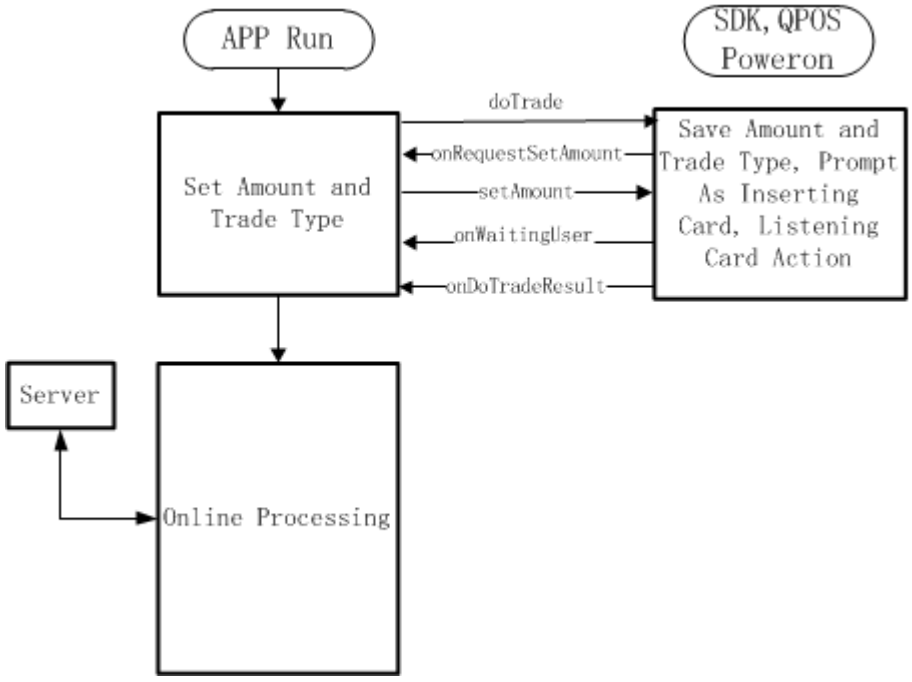
onRequestBatchData method.

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 **DATA-key** to be generated with **trackksn** and **IPEK**
encTrack2	Encrypted track 2 data with T-Des encryption key derived from **DATA-key** to be generated with **trackksn** and **IPEK**
encTrack3	Encrypted track 3 data with T-Des encryption key derived from **DATA-key** to be generated with **trackksn** and **IPEK**
partialTrack	Reserved
pinKsn	KSN of the Pin-block
trackksn	KSN of the track data
pinBlock[1]	Encrypted PIN data with T-Des encryption key derived from **PIN-key** to be generated with **pinKsn** and **IPEK**

Note:

1. The **PIN** format is defined as **ANSI X9.8**.
   pinBlock := (**PIN^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	amount: the amount for a transaction. cashbackAmount: the amount for a transaction. If this is non-zero, amount cannot be zero. currencyCode: three digits of the currency code, e.g. “840” for USD transactionType: enum of the transaction type.
Outputs	None
Description	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.The Enum TransactionType can be GOODS, SERVICES, CASHBACK, INQUIRY, TRANSFER, PAYMENT
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 onRequestSetAmount
See also	onRequestSetAmount

2.6.15 doTrade

Signature	void doTrade(int timeout)
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 onDoTradeResult delegate method.
See also	onDoTradeResult

2.6.16 doCheckCard

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

2.6.17s 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 aterminalTime)
Inputs	aterminalTime：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 udpateWorkKey

Signature	void udpateWorkKey (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 randemkey 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 dataTime,int timeout)
Inputs	dataTime 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 protocal（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 cach 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	cardType：card type address：the address of the read-in data data ：read-in the data cardPin：the 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（synchronize mode）
See also	

2.6.74 confirmAmount

Signature	void confirmAmount (String amount, int timeout)
Inputs	amount：payment amount timeout：overtime
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

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

2.6.79 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

2.6.80 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

2.6.81 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

2.6.82 getUpdateKey

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

2.6.83 setShutDownTime

Signature	void setShutDownTime(int time)
Inputs	time:it’s value in 10s and 1000s.
Outputs	None
Description	Set the device shut down time.
See also	onSetShutDownTime

2.6.84 setSleepModeTime

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

2.7 Delegate Methods Reference

2.7.1 onRequestWaitingUser

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

2.7.2 onQposIdResult

Signature	void onQposIdResult (Hashtable<String, String> posIdTable)
Inputs	Hashtable<String, String> posIdTable
Outputs	None
Description	Callback of pos id response
See also	getQposId()

2.7.3 onQposInfoResult

Signature	void onQposInfoResult (Hashtable<String, String> 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	udpateWorkKey()

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 onReturnApduResult

Signature	void onReturnApduResult (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 sendApdu. If the apdu data are encrypted, the KSN returned after the powerOnIcc command is used for encryption.
See also	sendApduByNFC() sendApdu()

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 SetBuzzerr
See also	

2.7.39 onReturnUpdateIPEKResult

Signature	void onReturnUpdateIPEKResult(boolean b)
Inputs	b:the flag of updating IPEK if is successful
Outputs	None
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 onSetShutDownTime

Signature	void onSetShutDownTime(boolean isSuccess)
Inputs	isSuccess:the flag of setting shut down time if is successful
Outputs	None
Description	The response of setShutDownTime
See also	

2.7.46 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	