EMVSwipe API

Integration Guide

Android Version

Aug 2014

V2.3.0

Revision History

Rev	Date	Description
1.0.0	4 Feb 2013	Initial Draft
1.1.0	9 Apr 2013	Update **onReturnDeviceInfo** method to return hardwareVersion and supportedTrack Add **resetEmvSwipeController** method Modify the **setAmount** method to add parameter cashbackAmount
1.2.0	19 Apr 2013	Add FID 22
1.2.1	2 May 2013	Retry Interval is changed from 2 seconds to 1 second Modify signature of setAmount method, max amount is changed to 1000000000
1.3.0	10 May 2013	Add EMV Level 1 command sendApdu for ADPU exchange Add powerOnIcc and powerOffIcc to turn ICC on and off Add callback functions onReturnApduResult , onReturnPowerOnIccResult, onReturnPowerOffIccResult
1.4.0	6 June 2013	Remove releaseAudioResource. Change currency in setAmount from int to string.
1.4.1	14 June 2013	Add getKsn and onReturnKsn Modify onReturnDeviceInfo to return Hashtable. onReturnEmvKsn is renamed to onReturnKsn and the signature is changed to return a table.
1.5.0	21 June 2013	Add NFC functions - powerOnNfc, powerOffNfc, nfcDataExchange - onReturnPowerOnNfcResult, onReturnPowerOffNfcResult, onReturnNfcDataResult Add 4 new TransactionResult enums CARD_NOT_SUPPORTED, MISSING_MANDATORY_DATA, CARD_BLOCKED_OR_NO_EMV_APPS, INVALID_ICC_DATA
1.5.1		Add enum USE_ICC_CARD to CheckCardResult
1.5.2		Fix some phone volume compatibility issues
1.5.3		APDU command bug fix (Firmware version 4.3 update required)
1.6.0	July 2013	Add cardEmulation and onCardEmulationResult
1.6.1	12 July 2013	Rename cardEmulation and onCardEmulationResult to getEmvCardData and onReturnEmvCardDataResult Add batteryPercentage to deviceInfoData Fix currency bug and adjust decimal places according to currency code. Allow "," as decimal point besides ".".

Continue on next page…

Rev	Date	Description
1.6.2	31 July 2013	Fix some phone audio issues Add check of currency Add currency exponent table Fix bug when stopAudio is called before ACK is sent.
1.6.3	2 Oct 2013	Fix some phone audio issues Add FID 60 Add product Type to decodeData Add enum CONDITION_NOT_SATISFIED to TransactionResult Add isDeviceHere, onDeviceHere, sendApduWithPkcs7Padding and onReturnApduResultWithPkcs7Padding
1.6.4	7 Nov 2013	Update return value of setAmount function Add encryptPin method Add onReturnEncryptPinResult callback Add key PAN to onReturnCheckCardResult hashtable Several functions are marked deprecated Updated EMV Level 2 flow diagram Add tags to getEmvCardData output parameters
1.6.5	11 Nov 2013	Rename TransactionResult enum from SELECT_APP_FAIL to CARD_BLOCKED Rename TransactionResult enum from CARD_BLOCKED_OR_NO_EMV_APPS to NO_EMV_APPS Add TransactionResult enum APPLICATION_BLOCKED
1.6.6	25 Nov 2013	Fix issues in Sony LT22i, HTC ADR6410LVW, LG P925, LG VS920 4G, LG MS840, Motorola MB853, Motorola Droid X2, Motorola DROID Pro, Samsung SPH-M950, Samsung SPH-M830
1.7.0	9 Dec 2013	Add encryptData and onReturnEncryptDataResult methods Add return parameter trackEncoding to onReturnCheckCardResult
1.8.0-beta1	6 Feb 2014	Change void onReturnNfcDataResult(boolean isSuccess, String data, int dataLength) to void onReturnNfcDataResult(NfcDataExchangeStatus, nfcDataExchangeStatus, String data, int dataLength) Add NfcDataExchangeStatus enums SUCCESS, NOT_YET_POWER_ON, NO_RESPONSE Add isSupportedNfc to getDeviceInfo output Add cancelCheckCard method
1.8.0	10 Mar 2014	Add onReturnCancelCheckCardResult callback

Continue on next page…

Rev	Date	Description
1.8.1	19 Mar 2014	Fix some phone audio issues
1.8.2	24 Mar 2014	Add pinKsn, trackKsn, emvKsn and uid to getDeviceInfo output
1.8.3	29 Apr 2014	Fix some phone audio issues
1.8.4	12 May 2014	Fix KSN bug
2.0.0-beta1	23 Apr 2014	Add readTerminalSetting and updateTerminalSetting methods Add onReturnReadTerminalSettingResult and onReturnUpdateTerminalSettingResult callbacks Add TerminalSettingStatus enums SUCCESS, TAG_NOT_FOUND, LENGTH_INCORRECT, TLV_INCORRECT, TAG_INCORRECT Add startEmv method with terminalTime as parameter. Speed optimization
2.0.0	29 May 2014	Fix some phone audio issues
2.0.1	11 June 2014	Fix some phone audio issues
2.0.2	30 June 2014	Fix some phone audio issues
2.1.0-beta1	7 July 2014	Add FID 61 Add overloaded function checkCard Add overloaded function startEmv Deprecate enum StartEmvResult and onReturnStartEmvResult callback. Add finalMessage key to onReturnCheckCardResult
2.1.0	11 July 2014	Bug fix for FID 61
2.2.0	21 July 2014	Add FID 46 Add viposExchangeApdu and onReturnViposExchangeApduResult
2.3.0-beta2	25 July 2014	Add function signature for onReturnTransactionResult and deprecate the old version Add function viposBatchExchangeApdu, sendVerifyIDResult, startAutoConfig, cancelAutoConfig, setConfig Add callback function onReturnViposBatchExchangeApduResult, onRequestVerifyID, onReturnAutoConfigResult, onReturnSetConfigResult
2.3.0-beta3	28 July 2014	Fix some phone audio issues
2.3.0	19 Aug 2014	Add csn to getDeviceInfo and getKsn output Add function: decodeTlv, viposGetIccData, getEmvCardNumber Add callback function: onReturnEmvCardNumber

Overview

EMVSwipe is a payment card reading device that works with mobile devices such as mobile phones, tablet computers and notebook computers. It has a magnetic card reader to read magnetic stripe cards and also an EMV card reader to read EMV cards. Some models are capable of reading NFC cards.
It communicates with the mobile device through the audio channel and USB.

This document provides the guideline on how to integrate EMVSwipe into their mobile payment application (or App) to accept either magnetic or EMV cards.

System Requirement

Development Platform:
JDK: Java 1.6 or above

Target System:
OS: Android 2.2 or above
CPU: 600MHz
RAM: 512MB

Android Permission

The library needs permission to use the audio 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.RE
CORD_AUDIO"/>
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
</manifest>
```

The EmvSwipeController Class

The **EmvSwipeController** class is the core of this API library. It has a number of general and utility methods that manage the class itself and a number of methods that communicate with the EMVSwipe device through the audio channel. Communications between the phone and EMVSwipe are bi-directional and a communication can be initiated by both sides. For commands sent to the App from the EMVSwipe, listener functions will be triggered.
Even when a command is initiated by the App, because of the nature of the communication channel and the operations, some methods will require a longer time to finish. To avoid blocking the App, all the methods are handled asynchronously. Handler methods are used to obtain the results. The listener and handler functions all are prefixed by "on", e.g. **onPowerDown()**.

General Methods

Method Name	Description
EmvSwipeController	Constructor Method
resetEmvSwipeController	Reset and bring the device back to a known initial state.
getApiVersion	Return the API version
isDevicePresent	Check if the EMVSwipe is plugged. Please note that a normal Headset will also return true.
startAudio	Start the audio resource for playing and recording
stopAudio	Stop the audio resource for playing and recording

Methods to communicate with EMVSwipe

Method Name	Description
getKsn	Retrieve the KSNs of the EMVSwipe device used for encryption
isDeviceHere	Check if an EMVSwipe is plugged in
getDeviceInfo	Retrieve parameters about the EmvSwipe device. Results are returned by **onReturnDeviceInfo**.
readTerminalSetting	Get the EMV terminal setting
updateTerminalSetting	Update the EMV terminal setting
setAmount	Set the amount and transaction type required for EMV transaction. This can be done before the startEmv command or in response to **onRequestSetAmount**
cancelSetAmount	Cancel the process in response to **onRequestSetAmount**

powerOnIcc	Turn on the EMV card. Results are returned by **onReturnPowerOnIccResult.**
powerOffIcc	Turn off the EMV card. Results are returned by **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. Results are returned by **onReturnApduResult.**
sendApduWithPkcs7Padding	Send APDU to ICC. Response from ICC is returned by **onReturnAdpuResultWithPkcs7Padding**
exchangeApdu	Send APDU to device. Response is returned by **onReturnExchangeApduResult**
batchExchangeApdu	Send batch APDU commands to device. Response is returned by **onReturnBatchExchangeApduResult**
powerOnNfc	Turn on the NFC transceiver. Results are returned by **onReturnPowerOnNfcResult.**
powerOffNfc	Turn off the NFC transceiver. Results are returned by **onReturnPowerOffNfcResult.**
nfcDataExchange	Exchange data with NFC card
checkCard (2 overloads)	Check if a card has been inserted or swiped. Results are returned by **onReturnCheckCardResult**
cancelCheckCard	Cancel the **checkCard** process.
encryptPin	Encrypt PIN for magnetic swipe operation with PIN Key
encryptData	Encrypt Data for other purposes with Data Key
getEmvCardData	Get card data in the EMV card. This is not part of the EMV payment process
getEmvCardNumber	Get card number from EMV card. This is not part of the EMV payment process
startEmv (3 overloads)	Start an EMV transaction. After receiving this command, EMVSwipe will take control and execute the EMV operation flow. This will start the EMV Level2 application using our approval EMV Level 2 kernel.
selectApplication	Select an application from a list of applications acceptable by the terminal and the EMV card in response to **onRequestSelectApplication**
cancelSelectApplication	Cancel the process in response to **onRequestSelectApplication**
sendPinEntryResult(deprecated)	Send PIN result to EMVSwipe in response to **onRequestPinEntry**
bypassPinEntry(deprecated)	No pin to send in response to **onRequestPinEntry**
cancelPinEntry(deprecated)	Cancel the PIN entry step in response to **onRequestPinEntry**
sendVerifyIDResult	Send verify cardholder ID result in response to **onRequestVerifyID** for PBOC
sendFinalConfirmResult	Send confirmation to EMVSwipe in response to **onRequestFinalConfirm**.
sendReferProcessResult(deprecated)	Send referral results from the processor back to EMVSwipe in response to **onRequestReferProcess**.
cancelReferProcess(deprecated)	Cancel referral process and abort transaction
sendAdviceProcessResult(deprecated)	Send advice results from the processor back to EMVSwipe in response to **onRequestAdviceProcess**.
sendOnlineProcessResult	Send transaction results from the processor back to EMVSwipe in response to

	onRequestOnlineProcess.
sendServerConnectivity	Send the connectivity status to EMVSwipe in response to **onRequestServerConnectivity**
sendTerminalTime	Send the terminal time in YYMMDDHHmmss format to EMVSwipe in response to **onRequestTerminalTime**
startAutoConfig	Starts auto configuration to get a set of suitable device configurations. Results are returned by **onReturnAutoConfigResult**
cancelAutoConfig	Cancels the auto configuration process. **onReturnAutoConfigResult** will be triggered after the process can be canceled
setConfig	Sets device configurations settings by auto configuration. Results are returned by **onReturnSetConfigResult**
decodeTlv	Utility function for converting TLV to a hashtable
viposGetIccData	Utility function for converting a normal TLV to a VIPOS relevant TLV data

The constructor method of the **EmvSwipeController** class is

```
public EmvSwipeController(Context context, EmvSwipeControllerListener
EmvSwipeControllerListener)
```

The input parameter passed must implement the interface **EmvSwipeControllerListener** which has methods to handle different asynchronous events that occur during the operation of the EMVSwipe. The developer must design a class that implements the interface and pass an instance of the class to the constructor method.

EmvControllerListener Methods

Method Name	Description
onWaitingForCard	EMVSwipe is ready and is waiting for a card swipe or a EMV card insert.
onReturnKsn	EMVSwipe has sent back the **getKsn** result.
onDeviceHere	EMVSwipe has responded to **isDeviceHere**
onReturnDeviceInfo	EMVSwipe has sent back the **getDeviceInfo** result.
onReturnReadTerminalSettingResult	EMVSwipe has sent back the **readTerminalSetting** result.
onReturnUpdateTerminalSettingResult	EMVSwipe has sent back the **updateTerminalSetting** result.
onReturnPowerOnIccResult	EMVSwipe has responded to **powerOnIcc**.
onReturnPowerOffIccResult	EMVSwipe has responded to **powerOffIcc**.
onReturnApduResult	EMVSwipe has returned APDU in response to **sendApdu**
onReturnApduResultWithPkcs7Padding	EMVSwipe has returned APDU in response to **sendApduWithPkcs7Padding**

onReturnExchangeApduResult	EMVSwipe has returned APDU In response to **exchangeApdu**
onReturnPowerOnNfcResult	EMVSwipe has responded to **powerOnNfc**
onReturnPowerOffNfcResult	EMVSwipe has responded to **powerOffNfc**
onReturnNfcDataResult	EMVSwipe has returned APDU in response to **nfcDataExchange**
onReturnCheckCardResult	EMVSwipe has sent back the **checkCard** result.
onReturnCancelCheckCardResult	EMVSwipe has returned in response to **cancelCheckCard**
onReturnEncryptPinResult	EMVSwipe has returned in response to **encryptPin**
onReturnEncryptDataResult	EMVSwipe has returned in response to **encryptData**
onReturnEmvCardDataResult	EMVSwipe has sent back **getEmvCardData** result.
onReturnEmvCardNumber	EMVSwipe has sent back **getEmvCardNumber** result.
onReturnStartEmvResult	EMVSwipe has responded to **startEmv**
onRequestSetAmount	EMVSwipe has requested to set amount.
onRequestSelectApplication	EMVSwipe has requested to select application.
onRequestPinEntry(deprecated)	EMVSwipe has requested for a PIN entry
onRequestVerifyID	EMVSwipe has requested for checking cardholder ID for PBOC
onRequestCheckServerConnectivity	EMVSwipe has requested for a check of server connectivity
onRequestFinalConfirm	EMVSwipe has requested for a final confirm before calling the first generate AC command
onRequestOnlineProcess	EMVSwipe has requested for online processing
onRequestReferProcess(deprecated)	EMVSwipe has requested for referral processing
onRequestAdviceProcess(deprecated)	EMVSwipe has requested for Advice processing
onRequestTerminalTime	EMVSwipe has requested for the terminal time.
onRequestDisplayText	EMVSwipe has requested to display some text.
onReturnClearDisplay	EMVSwipe has requested to clear the display
onReturnTransactionResult	EMVSwipe has sent back the batch data result.
onReturnTransactionLog(deprecated)	EMVSwipe has sent back the transaction log.
onReturnBatchData	EMVSwipe has sent back the batch data
onReturnReversalData	EMVSwipe has sent back the reversal data
onReturnAutoConfigResult	EMVSwipe has sent back the auto configuration settings
onReturnSetConfigResult	EMVSwipe has sent back the set configuration result
onDevicePlugged	EMVSwipe has been plugged in
onDeviceUnplugged	EMVSwipe has been unplugged
onNoDeviceDetected	No EMVSwipe is detected
onBatteryLow	The battery level EMVSwipe is low or critically low. It can occur after checkCard, startEmv or getDeviceInfo. When the level is low, the normal response will still happen. If the level is critically low, the normal response will not occur.
onError	An error has occurred.

Instantiation

An instance of an **EmvSwipeController** must be created first. There should not be more than one instance at any moment.
When the **EmvSwipeController** instance is no longer used, it should be disposed properly.
At any time, the **EMVSwipeController** can be reset and brought back to a known initial state by using the method **resetEmvSwipeController**.

Power On/Off

The EMVSwipe is normally powered down to minimize power consumption and it will be turned on upon any commands received from the host device.

Get Information about the Device

The **isDeviceHere** can be called to check if an EMVSwipe device is connected. This command can be used to differentiate from an audio interface device.

Information of the device can be obtained by **getKsn** and **getDeviceInfo**.

void getKsn()
void onReturnKsn (Hashtable<String, String> ksnDict)

A unique identification of the crypto-processor and three sets of KSN used for data encryption are returned.

Key	**Description**
uid	Unique identifier of the crypto-processor
trackKsn	KSN used for encryption of track data
emvKsn	KSN used for encryption of EMV data, including ATR response, APDU data and other online messages.
pinKsn	KSN used for encryption of online PIN
csn	CSN for VIPOS

public void getDeviceInfo()
public void onReturnDeviceInfo(Hashtable<String, String> deviceInfoData)

Key	Description
bootLoaderVersion	Bootloader Version
firmwareVersion	Firmware Version
hardwareVersion	HardwareVersion
batteryLevel	4-digit voltage level reading
batteryPercentage	Integer of battery percentage e.g. “100”, “99”
isCharging	Indicator of the charging status
isUsbConnected	Indicator of the USB connection status
isSupportedTrack1	Indicator of whether Track 1 is supported
isSupportedTrack2	Indicator of whether Track 2 is supported
isSupportedTrack3	Indicator of whether Track 3 is supported
isSupportedNfc	Indicator of whether NFC is supported
pinKsn	KSN used for encryption of online PIN
trackKsn	KSN used for encryption of track data
emvKsn	KSN used for encryption of EMV data, including ATR response, APDU data and other online messages.
csn	CSN for VIPOS
uid	Unique identifier of the crypto-processor

EMV Level 1 APDU exchange

EMV Level 1 APDU exchange can be achieved by using the **sendApdu** or **sendApduWithPkcs7Padding** method and response from the EMV card will be returned by the **onReturnApduResult** or **onReturnApduResultWithPkcs7Padding** method.

Before the data exchange the EMV card has to be turned on first by the **powerOnIcc method**. When the data exchange is finished, the EMV card can be turned off by **powerOffIcc**. After **powerOnIcc**, if there are no data communications within 15 minutes, the EMVSwipe will timeout and power off.

The **powerOnIcc**/**powerOffIcc** methods are only required for this EMV Level1 data exchange. There is no need to call them in the **checkCard** and **startEmv** methods below.

NFC Data Exchange

Similarly, NFC data exchange can be achieved by using the **nfcDataExchange** method and response from the EMV card will be returned by the **onReturnNfcDataResult** method.

public void nfcDataExchange(String data, int dataLength)
public void onReturnNfcDataResult(NfcDataExchangeStatus nfcDataExchangeStatus, String data, int dataLength)

Before the data exchange the NFC transceiver has to be turned on first by the **powerOnNfc** method. When the data exchange is finished, the NFC transceiver can be turned off by **powerOffNfc** method.

Check Card

EMVSwipe is capable of reading both magnetic stripe cards and EMV cards.
The payment application should first use the **checkCard** method to determine whether the cardholder is using magnetic stripe cards or EMV cards. There is no need to call **powerOnIcc** before this step and the power on/off is done transparently.
If the EMVSwipe supports NFC, then **checkCard** can return encTrack2 that contains the Track 2 equivalent stored on an EMV contactless card.
Note: There is another version of **checkCard** with a Hashtable as an input parameter for device using FID61.

The **onReturnCheckCardResult** delegate method returns the relevant information.

public void checkCard()
public void onReturnCheckCardResult(CheckCardResult checkCardResult, Hashtable<String, String> decodeData);

checkCardResult can be one of the following values:

```
public enum CheckCardResult {
    NONE,
    ICC,
    NOT_ICC,
    BAD_SWIPE,
```

```
        MCR,
        MAG_HEAD_FAIL,
        NO_RESPONSE,
        TRACK2_ONLY,
        NFC_TRACK2
    }
```

If a magnetic stripe card has been swiped, the encrypted track data will be returned along with the KSN (key serial number) in the decodeData Hashtable.

The Hashtable contain keys for the values:

Key	**Description**
formatID	The format of the output track data. It supports 22, 36 and 54
ksn	KSN of the device
PAN	Full PAN (optional)
maskedPAN	Masked card number showing at most the first 6 and last 4 digits with in-between digits masked by “X”
cardholderName	The cardholder name as seen on the card. This can be up to 26 characters.
expiryDate	4-digit in the form of YYMM in the track data
serviceCode	3-digit service code in the track data
encTracks	Encrypted track1 and track2
encTrack1	Encrypted track 1 data with encryption key derived from KSN
encTrack2	Encrypted track 2 data with encryption key derived from KSN
encTrack3	Encrypted track 3 data with encryption key derived from KSN
track1length	Length of Track 1 data
track2length	Length of Track 2 data
track3length	Length of Track 3 data
partialTrack	The 26 characters found in track1 after the first ^ symbol. This part is necessary for the reconstruction of track1.
trackEncoding	Tell if track2 and Track3 data are packed in “ASCII” or “BCD”
finalMessage	(Optional) FID61 device only

The checkCard process can be stopped by the **cancelCheckCard** method.

Start EMV

If an EMV card has been inserted, the **startEmv** command can be called to start the EMV Level2 payment process. Again, there is no need to call **powerOnIcc** explicitly. The EMVSwipe will take control and go through the EMV steps.

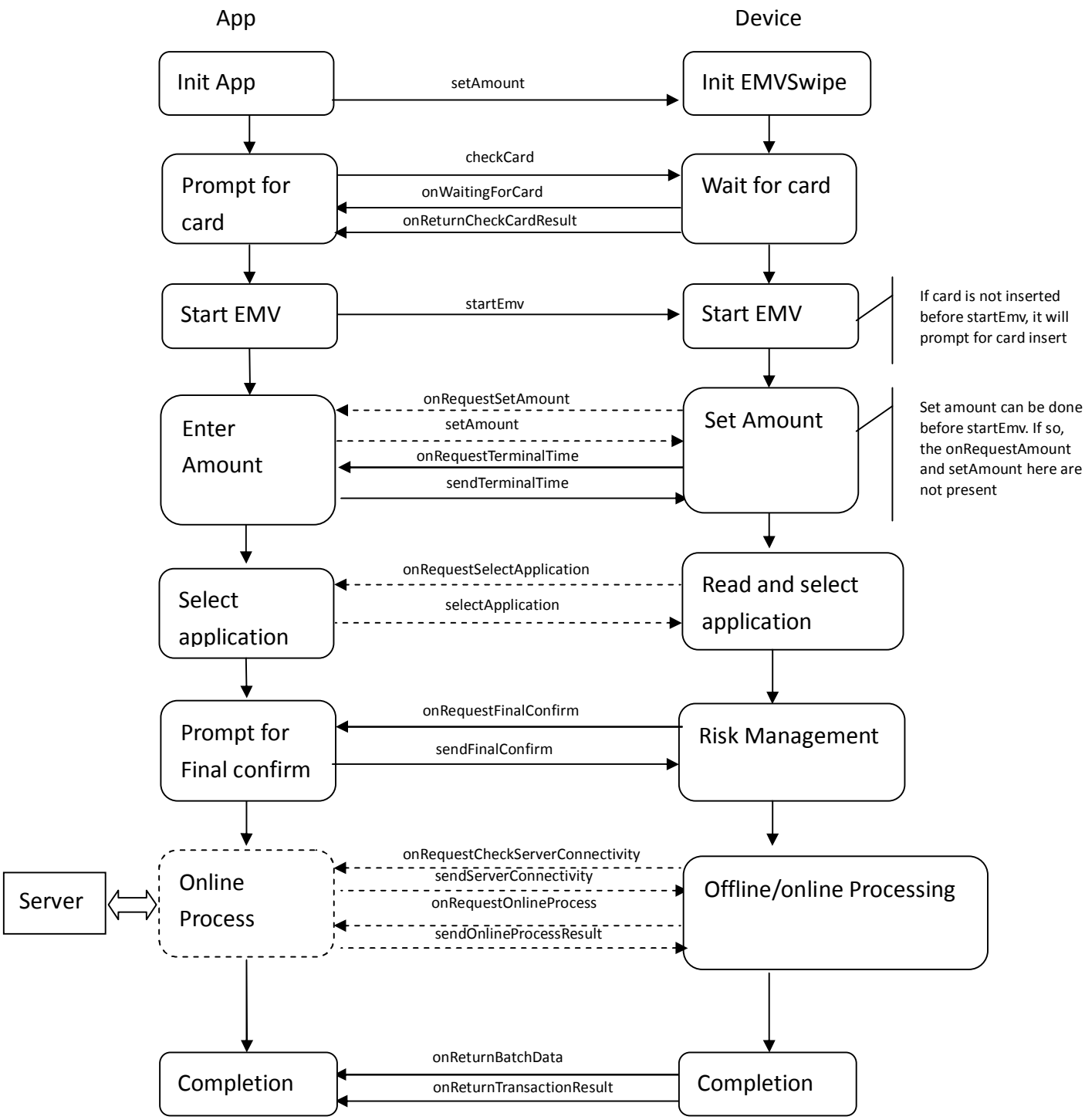
public void startEmv(EmvOption emvOption)
where EmvOption can be used to force an online transaction

```
public enum EmvOption {
    START,
    START_WITH_FORCE_ONLINE,
    START_WITH_BATCH_DATA_CAPTURE,
    START_WITH_ONLINE_DATA_CAPTURE
}
```

Another **startEmv** method has the terminalTime parameter in the input. When this overloaded version of **startEmv** is used, the onRequestTerminalTime and onRequestCheckServerConnectivity callbacks will not be triggered and this helps to reduce the transaction time.

Another overloaded version off **startEmv** can be used to set various timeouts in the EMV transaction.

EMV Level 2 Transaction Flow



EMV Level 2 Operation

EMV Step 0. Start EMV

The **setAmount** command should be called to capture the amount in the transaction. After that, the EMV payment process is started by calling **startEmv**.
The EMVSwipe asks for the terminal time through **onRequestTerminalTime**.
The terminal time in YYMMDDHHmmss format should be sent in response by **sendTerminalTime**.

EMV Step 1. Select Application

An EMV card may support multiple payment applications. The EMVSwipe 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. The app should prompt the user to select one application and then call the **selectApplication** method. The user can also select to abort the transaction by **cancelSelectApplication**.

EMV Step 2. Read Application Data

In this step, EMVSwipe will read the necessary data from the EMV card.

EMV Step 3. Card Authentication

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

EMV Step 4. Processing Restrictions

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

EMV Step 5. Cardholder Verification

Depending on the requirement of the card, either signature or no verification is required.

EMV Step 6. Terminal Risk Management

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

EMV Step 7. Terminal Action Analysis

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

At the end of this step, a final confirmation is required and **onRequestFinalConfirm** is triggered. 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 EMVSwipe by calling **sendFinalConfirmResult**.

EMV Step 8. Card Risk Management

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

EMV Step 9. Online Processing

An EMV transaction can either be online or offline.

If online processing is required, then the **onRequestOnlineProcess** delegate method is triggered. The parameter tlv contains the tag-length-value data structure returned by the EMV kernel.

After reformatting, the client app should send the data to the processor. When the processing results are returned from the processor, it should send the results back to EMVSwipe by **sendOnlineProcessResult.**

The tags and data elements that are required are processor and issuer dependent. Check the processor integration guide for details. See also the EVM Book 3 Annex A for the full list of tags and the TLV structure.

The following tags are usually returned to the ICC.

Tag	Parameter
008A	Authorisation Response Code
0091	Issuer Authentication Data
0071	Issuer Script Template 1 (needed for Step 10)
0072	Issuer Script Template 2 (needed for Step 10)

This step is skipped in offline processing.

EMV Step 10. Issuer Scripts Processing

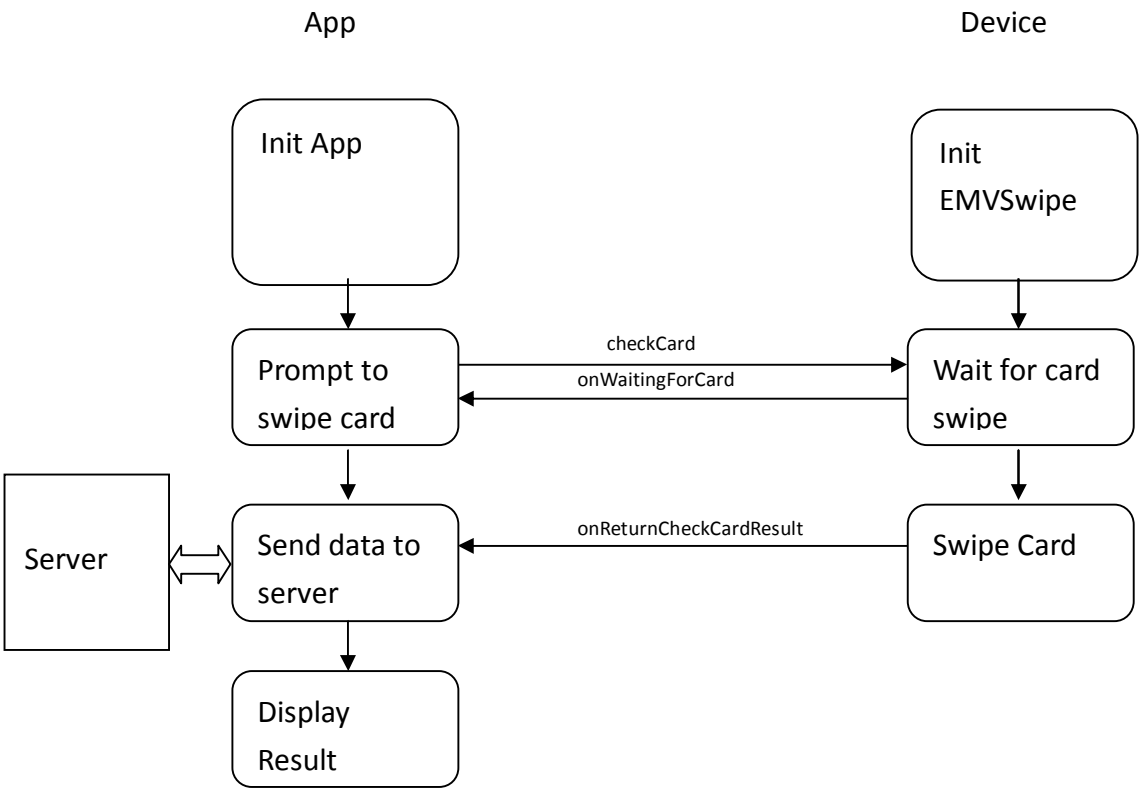
This step is handled transparently between the EMVSwipe and EMV card if issue

scripts are present in the online processing results or skipped otherwise.
This step is skipped in offline processing.

EMV Step 11. Completion
In this step, EMVSwipe sends back the final transaction result from the EMV card by **onReturnBatchData**.

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 EMVSwipe device. Later, the batch data should be updated to the server for settlement.

Magnetic Card Flow



The operation for magnetic stripe operation is much simpler.
After calling the **checkCard** method, if a card has been swiped successfully, the data are returned in a table containing values encTrack1, encTrack2, encTrack3 and ksn, among others. With these parameters, the track data can be decrypted back in the server.

public void checkCard();

public void onReturnCheckCardResult(CheckCardResult checkCardResult, Hashtable<String, String> decodeData);

API Methods Reference

getApiVersion

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

resetEmvSwipeController

Signature	void resetEmvSwipeController()
Inputs	None
Outputs	None
Description	Reset the state of the EmvSwipe controller
See also	

startAudio

Signature	void startAudio()
Inputs	None
Outputs	None
Description	Start the audio resource for playing and recording
See also	stopAudio, onAudioStarted

stopAudio

Signature	void stopAudio()
Inputs	None
Outputs	None
Description	Stop and release the audio resource for playing and recording
See also	startAudio

isDevicePresent

Signature	boolean isDevicePresent()
Inputs	None
Outputs	true – presence of an audio device. false – no audio device is plugged
Description	Check if the audio jack is plugged in.
See also	isDeviceHere

isDeviceHere

Signature	void isDeviceHere()
Inputs	None
Outputs	None
Description	Check if an EmvSwipe is connected.
See also	isDevicePresent, onDeviceHere

getKsn

Signature	void getKsn()
Inputs	None
Outputs	None
Description	Retrieve the KSN of the EmvSwipe device. Results are returned by onReturnKsn.
See also	onReturnKsn

getDeviceInfo

Signature	void getDeviceInfo()
Inputs	None
Outputs	None
Description	Retrieve parameters about the EmvSwipe device. Results are returned by onReturnDeviceInfo which includes: firmware version, bootloader version, USB connection and charging status, battery level, and hardware version
See also	onReturnDeviceInfo

readTerminalSetting

Signature	void readTerminalSetting(String tag)
Inputs	tag: EMV tag of the terminal configuration to be read, e.g. “9F1A”
Outputs	None
Description	Retrieve an EMV terminal configuration parameter. Common configurable tags include: 9F01: Acquirer Identifier (format n6..11) 9F16: Merchant Identifier (format ans15) 9F1A: Terminal Country Code (format n3) 9F4E: Merchant Name and Location (variable length)
See also	onReturnReadTerminalSettingResult

updateTerminalSetting

Signature	void updateTerminalSetting(String tlv)
Inputs	tlv: EMV tag of the terminal configuration to be updated
Outputs	None
Description	Update an EMV terminal configuration parameter, provided that it is there. Common configurable tags include: 9F01: Acquirer Identifier (format n6..11) 9F16: Merchant Identifier (format ans15) 9F1A: Terminal Country Code (format n3) 9F4E: Merchant Name and Location (variable length) 9F1B: Terminal floor limit. (format: binary 32) e.g. tlv=“9F1A020840” sets the terminal country code to USA. **Note**: for tag “9F1B”, it will change the terminal floor limit of all applications (AIDs) in the terminal setting. It is not possible to update the floor limit of only one AID at this moment. WARNING: This method should not be called frequently because there is a limited write-life-cycle of the flash memory used to store the parameters. It should only be called once for each deployment/redeployment.
See also	onReturnUpdateTerminalSettingResult

Notes

The next three methods **powerOnIcc**, **powerOffIcc**, **sendApdu**, **sendApduWithPkcs7Padding** are EMV Level 1 Commands. For EMV Level 2 operation, calling the methods powerOnIcc and powerOffIcc are not required and it is handled by the EMV level 2 operation flow automatically.

powerOnIcc

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

powerOffIcc

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

sendApdu

Signature	void sendApdu(String apdu, int apduLength)
Inputs	apdu: the apdu data to be sent apduLength: length of apdu to be sent
Outputs	None
Description	Send APDU exchange to ICC. Response data are returned by onReturnApduResult.
See also	onReturnApduResult

sendApduWithPkcs7Padding

Signature	void sendApduWithPkcs7Padding(String apdu)
Inputs	apdu: the apdu data to be sent
Outputs	None
Description	Send APDU exchange to ICC. The data are padded by PKCS7 padding and then encrypted. Response data are returned by onReturnApduResultWithPkcs7Padding.
See also	onReturnApduResultWithPkcs7Padding

exchangeApdu

Signature	void exchangeApdu (String apdu)
Inputs	apdu: the apdu data to be sent
Outputs	None
Description	APDU exchange for VI-POS. Response data are returned by onReturnExchangeApduResult.
See also	onReturnExchangeApduResult

batchExchangeApdu

Signature	void batchExchangeApdu (Hashtable<Integer, String[]> apduCommands)
Inputs	apduCommands: **Hashtable Keys** *Integer* Tag number. Refer to Q/CUP 037.2.4-2011 Section 11.8.3 **HashMap Values** *String[]* in the order of 0: encryption key number for INIT_FOR_DESCRYPT 1: submitTime for INIT_FOR_DESCRYPT 2: DES_CRYPT APDU command
Outputs	None
Description	APDU exchange in batch for VI-POS. Response data are returned by onReturnBatchExchangeApduResult.
See also	onReturnBatchExchangeApduResult

The next three methods **powerOnNfc**, **powerOffNcc**, **nfcDataExchange** are NFC Commands.

powerOnNfc

Signature	void powerOnNfc(String data)
Inputs	data: a protocol dependent string to initiate the NFC reader.
Outputs	None
Description	Provide power to NFC transceiver for data exchange
See also	onReturnPowerOnNfcResult

powerOffIcc

Signature	void powerOffNfc()
Inputs	None
Outputs	None
Description	Cut off power to NFC transceiver after data exchange
See also	onReturnPowerOffNfcResult

nfcDataExchange

Signature	void nfcDataExchange(String data, int dataLength)
Inputs	data: the data to be sent dataLength: length of data to be sent
Outputs	None
Description	Send data to NFC card. Response data are returned by onReturnNfcDataResult.
See also	onReturnNfcDataResult

setAmount

Signature	boolean setAmount(String amount, String cashbackAmount, 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	true: setAmount is successful, false: setAmount fails
Description	Set the amount, cashback amount, currency and type of a transaction. The amount can have at most one decimal point “.” or “,” and the EMV data field will be adjusted according to the currencyCode. For example “100.00” USD will be represented as 10000 while “100” JPY will be represented as 100. The total of amount and cashback must be at most 12 digits. This method can be called at the beginning of a transaction or in response to an onRequestSetAmount call requested by the EMV engine. If the return result is false, the application should wait for onError to handle the error before proceeding further and calling startEmv. The Enum **TransactionType** can be GOODS, SERVICES, CASHBACK, INQUIRY, TRANSFER, PAYMENT
See also	onRequestSetAmount, cancelSetAmount

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

checkCard

Signature	void checkCard()
Inputs	None
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 onReturnCheckCardResult delegate method.
See also	onReturnCheckCardResult

checkCard

Signature	checkCard(Hashtable<String, String> data)
Inputs	data: parameter tables
Outputs	None
Description	Check the status of the Magnetic Card Reader or the EMV Card reader. It checks if a card has been swiped or an EMV card is inserted. The result is returned by the onReturnCheckCardResult delegate method. The keys include: checkCardTimeout: String of the timeout in second orderID: (Optional) 32 hexadecimal digit String used in FID61 devices randomNumber: (Optional) 6 hexadecimal digit String used in FID61 devices
See also	cancelCheckCheck, onReturnCheckCardResult

cancelCheckCard

Signature	void cancelCheckCard()
Inputs	None
Outputs	None
Description	Stop the checkCard process
See also	checkCard, onReturnCancelCheckCardResult

encryptPin

Signature	void encryptPin(String pin, String pan)
Inputs	pin: 4-12 digits clearText PIN pan: 13-19 digits the full PAN of the card
Outputs	None
Description	Encrypt PIN with PIN key for magnetic stripe operation . The result is returned by the onReturnEncryptPinResult delegate method.
See also	onReturnEncryptPinResult

encryptData

Signature	void encryptData(String data)
Inputs	data: the data to be encrypt in Hexstring, the data must be packed to multiple of 16 hex characters.
Outputs	None
Description	A general encrypt data operation with Data key. DUKPT key, TDES encryption, CBC mode with zero IV and no padding is used. The result is returned by the onReturnEncryptDataResult delegate method.
See also	onReturnEncryptDataResult

getEmvCardData

Signature	void getEmvCardData()
Inputs	None
Outputs	None
Description	Get the card data from an EMV card. This is not part of the EMV payment process.
See also	onReturnEmvCardDataResult

getEmvCardNumber

Signature	void getEmvCardNumber()
Inputs	None
Outputs	None
Description	Get the card number from an EMV card. This is not part of the EMV payment process.
See also	onReturnEmvCardNumber

startEmv

Signature	void startEmv(EmvOption emvOption)
Inputs	emvOption: enum to indicate select online or offline emv.
Outputs	None
Description	Start the EMV process. The EmvSwipe will take control and go through the EMV steps. When started normally, whether the transaction will go online will be a joint decision of EmvSwipe and the EMV card. The Enum **EmvOption** can be START, START_WITH_FORCE_ONLINE, START_WITH_BATCH_DATA_CAPTURE, START_WITH_ONLINE_DATA_CAPTURE
See also	

startEmv

Signature	void startEmv(EmvOption emvOption, String terminalTime)
Inputs	emvOption: enum to indicate select online or offline emv. terminalTime: Current local time in the format YYMMDDHHmmss
Outputs	None
Description	Start the EMV process. The EmvSwipe will take control and go through the EMV steps. When started normally, whether the transaction will go online will be a joint decision of EmvSwipe and the EMV card. The Enum **EmvOption** can be START, START_WITH_FORCE_ONLINE, START_WITH_BATCH_DATA_CAPTURE, START_WITH_ONLINE_DATA_CAPTURE When this overloaded version is called, the onRequestTerminalTime and onRequestCheckServerConnectivity will be skipped.
See also	

startEmv

Signature	void startEmv(Hashtable<String, String> data)
Inputs	data: table of input parameters
Outputs	None
Description	Start the EMV process. The EmvSwipe will take control and go through the EMV steps. When started normally, whether the transaction will go online will be a joint decision of EmvSwipe and the EMV card. The keys include: emvOption: can be START, START_WITH_FORCE_ONLINE, START_WITH_BATCH_DATA_CAPTURE, START_WITH_ONLINE_DATA_CAPTURE (in String format) terminalTime: 12-digit String in YYMMDDHHmmss format checkCardTimeout: String of the timeout in second setAmountTimeout: String of the timeout in second selectApplicationTimeout: String of the timeout in second onlineProcessTimeout: String of the timeout in second finalConfirmTimeout: String of the timeout in second orderID: (Optional) 32 hexadecimal digit String used in FID61 devices randomNumber: (Optional) 6 hexadecimal digit String used in FID61 devices
See also	

selectApplication

Signature	void selectApplication(int index)
Inputs	index: index to the selected application.
Outputs	None
Description	An EMV card may support multiple payment applications. The EmvSwipe reads the list of applications IDs supported by the EMV card and triggers the onRequestSelectApplication delegate method. The app should prompt the user to select one application and then call this method.
See also	onRequestSelectApplication, cancelSelectApplication

cancelSelectApplication

Signature	void cancelSelectApplication()
Inputs	None
Outputs	None
Description	An EMV card may support multiple payment applications. The EmvSwipe reads the list of applications IDs supported by the EMV card and triggers the onRequestSelectApplication delegate method and expect the operator to select a payment application. The EMV transaction can be aborted at this step by calling this method.
See also	onRequestSelectApplication, selectApplication

sendPinEntryResult (deprecated)

Signature	void sendPinEntryResult(String pin)
Inputs	pin: clear text PIN
Outputs	None
Description	This method can be called to send the plaintext Pin in response to onRequestPinEntry.
See also	onRequestPinEntry, cancelPinEntry, bypassPinEntry

bypassPinEntry (deprecated)

Signature	void bypassPinEntry()
Inputs	None
Outputs	None
Description	This method can be called to try to bypass the Pin Entry step in response to onRequestPinEntry. If the card does not accept bypassing, the transaction will be aborted.
See also	onRequestPinEntry, cancelPinEntry

cancelPinEntry (deprecated)

Signature	void cancelPinEntry()
Inputs	None
Outputs	None
Description	This method can be called to try to abort the transaction in response to onRequestPinEntry.
See also	onRequestPinEntry, bypassPinEntry

sendVerifyIDResult

Signature	void sendVerifyIDResult (boolean isSuccess)
Inputs	isSuccess: true – cardholder ID verified false – cannot verify cardholder ID
Outputs	None
Description	An ID checking process for PBOC may be initiated by the card or by the terminal and the onRequestVerifyID method is triggered. The operator should check for the cardholder ID.
See also	onRequestVerifyID

sendFinalConfirmResult

Signature	void sendFinalConfirmResult(boolean isConfirmed)
Inputs	isConfirmed: true – proceed, false - cancel transaction
Outputs	None
Description	The EMV process requested for a final confirm before calling the first generate AC command and the onRequestFinalConfirm method is triggered. The operator can choose to proceed or cancel with this method.
See also	onRequestFinalConfirm

sendTerminalTime

Signature	void sendTerminalTime(String terminalTime)
Inputs	terminalTime: Current local time in the format YYMMDDHHmmss
Outputs	None
Description	Send terminal time in response to onRequestTerminalTime where the EMV process has requested to get the current time.
See also	onRequestCheckServerConnectivity

sendServerConnectivity

Signature	void sendServerConnectivity:(boolean isConnected)
Inputs	isConnected: true – connected, false – no connection
Outputs	None
Description	Send server connectivity result in response to onRequestCheckServerConnectivity where the EMV process has requested to get the connectivity status before online operation.
See also	onRequestCheckServerConnectivity

sendReferProcessResult (deprecated)

Signature	void sendReferProcessResult(ReferralResult referralResult)
Inputs	ReferralResult: –APPROVED: the transaction is approved DECLINED: the transaction is declined
Outputs	None
Description	A manual voice referral process may be initiated by the card or by the issuer and the onRequestReferProcess method is triggered. The operator should call the bank to ask for the referral approval. The attendant may manually override the referral process and may accept or decline the transaction without performing a referral.
See also	onRequestReferProcess

cancelReferProcess (deprecated)

Signature	void cancelReferProcess()
Inputs	None
Outputs	None
Description	Abort the EMV transaction
See also	onRequestReferProcess

sendOnlineProcessResult

Signature	void sendOnlineProcessResult(String tlv)
Inputs	tlv: The TLV message data of online processing result.
Outputs	None
Description	Send back the online process result to the ICC.
See also	onRequestOnlineProcess

startAutoConfig

Signature	void startAutoConfig()
Inputs	None
Outputs	None
Description	Starts auto configuration to get a set of suitable device configurations. This process may take a few minutes to complete. The process can be canceled using cancelAutoConfig function. Results will be returned by onReturnAutoConfigResult.
See also	cancalAutoConfig, onReturnAutoConfigResult

cancelAutoConfig

Signature	void cancalAutoConfig()
Inputs	None
Outputs	None
Description	Cancels the auto configuration process. onReturnAutoConfigResult will be triggered after the process can be canceled.
See also	onReturnAutoConfigResult

setConfig

Signature	void setConfig(String settings)
Inputs	settings: The auto configuration settings returned by onReturnAutoConfigResult
Outputs	None
Description	Sets device configuration. Result will be returned by onReturnSetConfigResult
See also	onReturnAutoConfigResult, onReturnSetConfigResult

decodeTlv

Signature	Hashtable<String, String> decodeTlv (String tlv)
Inputs	tlv: tag value results to be decoded into the hashtable
Outputs	None
Description	Utility function for decode the TLV and output the hashtable where the keys are the TLV tags and the values are the TLV values
See also	

viposGetIccData

Signature	String viposGetIccData(String tlv)
Inputs	tlv: tag value results to be converted
Outputs	None
Description	Utility function for converting a normal TLV to a VIPOS relevant TLV data
See also	

Delegate Methods Reference

onBatteryLow

Signature	void onBatteryLow(BatteryStatus batteryStatus)
Inputs	batteryStatus: enum of the battery status
Outputs	None
Description	EMVSwipe is woken up by an audio command and is powered up and is ready for operations. Enum **BatteryStatus** can be LOW, CRITICALLY_LOW When battery is low, the operation can still be completed. When battery is critically low, the operation cannot be completed.
See also	

onPowerDown

Signature	void onPowerDown()
Inputs	None
Outputs	None
Description	EmvSwipe is powered down because of inactivity
See also	

onDeviceHere

Signature	void onDeviceHere(boolean isHere)
Inputs	isHere: true – Presence of the EMVSwipe false – There is no device plugged in or another audio device is plugged in
Outputs	None
Description	Response to isDeviceHere query
See also	

onDevicePlugged

Signature	void onDevicePlugged()
Inputs	None
Outputs	None
Description	A device is plugged in.
See also	onDeviceUnplugged

onDeviceUnplugged

Signature	void onDeviceUnplugged()
Inputs	None
Outputs	None
Description	A device is unplugged.
See also	onDevicePlugged

onNoDeviceDetected

Signature	void onNoDeviceDetected()
Inputs	None
Outputs	None
Description	No EmvSwipe is detected.
See also	

onReturnKsn

Signature	void onReturnKsn (Hashtable<String, String> ksnData)
Inputs	ksnDict: IDs and KSNs of the device in hexadecimal format
Outputs	None
Description	Return KSN in response to getKsn **Hashtable Keys** uid: Unique identifier of the crypto-processor trackKsn: KSN used for encryption of track data emvKsn: KSN used for encryption of EMV data, including ATR response, APDU data and other online messages. pinKsn: KSN used for encryption of online PIN csn: CSN for VIPOS
See also	getKsn

onReturnDeviceInfo

Signature	void onReturnDeviceInfo(Hashtable<String, String> deviceInfoData)
Inputs	deviceInfoData: device data table
Outputs	None
Description	Return device info in response to getDeviceInfo **Hashtable Keys** firmwareVersion: version of the firmware bootLoaderVersion: the version of the bootloader hardwareVersion: version of the hardware batteryLevel: 4 digits value of the voltage of the battery batteryPercentage: Integer of battery percentage e.g. “100”, “99” isCharging: “true” – is charging, “false” – no charging isUsbConnected: “true” – Connected, “false” – Not connected. isSupportedTrack1: “true” – Supported, “false” – not supported isSupportedTrack2: “true” – Supported, “false” – not supported isSupportedTrack3: “true” – Supported, “false” – not supported isSupportedNfc: “true” – Supported, “false” – not supported pinKsn: KSN used for encryption of online PIN trackKsn: KSN used for encryption of track data emvKsn: KSN used for encryption of EMV data, including ATR response, APDU data and other online messages. csn: CSN for VIPOS uid: Unique identifier of the crypto-processor
See also	getDeviceInfo

onWaitingForCard

Signature	void onWaitingForCard()
Inputs	None
Outputs	None
Description	Triggered in response to checkCard
See also	

onReturnReadTerminalSettingResult

Signature	void onReturnReadTerminalSettingResult(TerminalSettingStatus terminalSettingStatus, String value)
Inputs	terminalSettingStatus: Indicates status of operation value: the value of the corresponding tag
Outputs	None

Description	Return EMV terminal setting in response to readTerminalSetting
	The enum TerminalSettingStatus is defined as below:
	public enum TerminalSettingStatus {
	SUCCESS,
	TAG_NOT_FOUND,
	LENGTH_INCORRECT,
	TLV_INCORRECT,
	TAG_INCORRECT
	}

See also	readTerminalSetting

onReturnUpdateTerminalSettingResult

Signature	void onReturnUpdateTerminalSettingResult(TerminalSettingStatus terminalSettingStatus)
Inputs	terminalSettingStatus: Indicates status of operation
Outputs	None

Description	Return result in response to updateTerminalSetting
	The enum TerminalSettingStatus is defined as below:
	public enum TerminalSettingStatus {
	SUCCESS,
	TAG_NOT_FOUND,
	LENGTH_INCORRECT,
	TLV_INCORRECT,
	TAG_INCORRECT
	}

See also	updateTerminalSetting

onReturnCheckCardResult

Signature	void onReturnCheckCardResult(CheckCardResult checkCardResult, Hashtable<String, String> decodeData)
Inputs	checkCardResult: Enum to show the card status decodeData: card data read by magnetic card reader, if a card swipe is captured successfully **Hashtable Keys:** *formatID* – Format ID *PAN* – Full PAN (optional) *maskedPAN* – Masked PAN *cardholderName* – cardholder name *expiryDate* – Expiry Date of the card *serviceCode* – 3-digit service code *ksn* – Key Serial Number for track data encryption. *encTrack1* – Encrypted Track 1 in HEX string. *encTrack2* – Encrypted Track 2in HEX string. *encTrack3* – Encrypted Track 3in HEX string. *encTracks* – Encrypted tracks in HEX string. *track1Length* – length of track 1 *track2Length* – length of track 2 *track3Length* – length of track 3 *partialTrack* – part of track 1 *productType* – product type *trackEncoding* – Indicates the encoding of track2 and track3 data. Can be “ASCII” or “BCD”. *finalMessage – proprietary data in FID61 devices*
Outputs	None
Description	Return the status of the checkCard command and also the card swipe result, if present. Enum **CheckCardResult** can be NONE, ICC, NOT_ICC, BAD_SWIPE, MCR, MAG_HEAD_FAIL, NO_RESPONSE,

	TRACK2_ONLY, NFC_TRACK2
See also	checkCard

onReturnCancelCheckCardResult

Signature	void onReturnCancelCheckCardResult(boolean isSuccess)
Inputs	isSuccess: result of the cancelCheckCard command
Outputs	None
Description	Respond to cancelCheckCard. If isSuccess is true, the checkCard process has been stopped successfully
See also	cancelCheckCard

onReturnEncryptPinResult

Signature	void onReturnEncryptPinResult(String epb, String ksn)
Inputs	epb: encrypted PIN block ksn: KSN used for encryption of PIN
Outputs	None
Description	Respond to encryptPin.
See also	encryptPin

onReturnEncryptDataResult

Signature	void onReturnEncryptDataResult(String encryptedData, String ksn)
Inputs	encryptedData: encrypted data block ksn: KSN used for encryption of Data
Outputs	None
Description	Respond to encryptData
See also	encryptData

onReturnPowerOnIccResult

Signature	void onReturnPowerOnIccResult(boolean isSuccess, String ksn, String atr, int atrLength)
Inputs	isSuccess: 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 apduLength: 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

onReturnPowerOffIccResult

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

onReturnApduResult

Signature	void onReturnApduResult(boolean isSuccess, String apdu, int apduLength)
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	sendApdu

onReturnApduResultWithPkcs7Padding

Signature	void onReturnApduResultWithPkcs7Padding(boolean isSuccess, String apdu)
Inputs	isSuccess: true – success, false – failure apdu: data returned
Outputs	None
Description	Return data in response to the level 1 EMV method sendApduWithPkcs7Padding. If the apdu data are encrypted, the KSN returned after the powerOnIcc command is used for encryption.
See also	sendApduWithPkcs7Padding

onReturnExchangeApduResult

Signature	void onReturnExchangeApduResult (String apdu)
Inputs	apdu: apdu returned
Outputs	None
Description	Return data in response to the method exchangeApdu for VI-POS.
See also	exchangeApdu

onReturnBatchExchangeApduResult

Signature	void onReturnExchangeApduResult (Hashtable<Integer, String> data)
Inputs	data: Hashtable Key: Tag number. Refer to Q/CUP 037.2.4-2011 Section 11.8.3 Hashtable Value: APDU Response of DES_CRYPT APDU command.
Outputs	None
Description	Return data in response to the method batchExchangeApdu for VI-POS.
See also	batchExchangeApdu

onReturnPowerOnNfcResult

Signature	void onReturnPowerOnNfcResult(boolean isSuccess, String response, int responseLength)
Inputs	isSuccess: true – success, false – failure response: protocol dependent data returned upon power on. responseLength: length of the response data
Outputs	None
Description	Respond to powerOnNfc.
See also	powerOnNfc

onReturnPowerOffNfcResult

Signature	void onReturnPowerOffNfcResult(boolean isSuccess)
Inputs	isSuccess: true – success, false – failure
Outputs	None
Description	Respond to powerOffNfc
See also	powerOffNfc

onReturnNfcDataResult

Signature	void onReturnNfcDataResult(NfcDataExchangeStatus nfcDataExchangeStatus, String data, int dataLength)
Inputs	nfcDataExchangeStatus: enum SUCCESS, NOT_YET_POWER_ON, NO RESPONSE data: data returned dataLength: length of the APDU data
Outputs	None
Description	Return data in response to the NFC data exchange.
See also	nfcDataExchange

onReturnEmvCardDataResult

Signature	void onReturnEmvCardDataResult (String tlv)
Inputs	tlv: a list of card data in TLV format.
Outputs	None
Description	Respond to getEmvCardData. Information about the card is returned. This includes cardholder name (5F20), expiration date (5F24), masked PAN (custom tag C4), encrypted tags 5A, 57, 9F1F (custom tag C5), KSN for encryption key used in C5 (custom tag C3)
See also	getEmvCardData.

onReturnEmvCardNumber

Signature	void onReturnEmvCardNumber(String cardNumber)
Inputs	cardNumber: the EMV card number
Outputs	None
Description	Respond to getEmvCardNumber.
See also	getEmvCardNumber

onReturnStartEmvResult (deprecated)

Signature	void onReturnStartEmvResult(StartEmvResult result, String ksn)
Inputs	result: enum of the startEmv result. ksn: KSN of EMVSwipe
Outputs	None
Description	Return data in response to startEmv
See also	selectApplication

onRequestSelectApplication

Signature	void onRequestSelectApplication:(ArrayList<String> appList)
Inputs	appList: Array of applications found in EMV card
Outputs	None
Description	EmvSwipe is requesting user to select an application in the EMV process and a list of applications supported is returned.
See also	selectApplication

onRequestSetAmount

Signature	void onRequestSetAmount()
Inputs	None
Outputs	None
Description	EmvSwipe is requesting the amount. The app should prompt the user of the transaction amount.
See also	setAmount, cancelSetAmount

onRequestPinEntry(deprecated)

Signature	void onRequestPinEntry()
Inputs	None
Outputs	None
Description	EmvSwipe is requesting PIN entry in the EMV process.
See also	sendPinEntryResult, cancelPinEntry, bypassPinEntry

onRequestVerifyID

Signature	void onRequestVerifyID(String tlv)
Inputs	tlv: information about cardholder name in TLV format
Outputs	None
Description	An ID checking process for PBOC may be initiated by the card or by the terminal and the onRequestVerifyID method is triggered. The operator should check for the cardholder ID and respond by using sendVerifyIDResult function.
See also	sendVerifyIDResult

onRequestCheckServerConnectivity

Signature	void onRequestCheckServerConnectivity()
Inputs	None
Outputs	None
Description	EmvSwipe is requesting checking of the server connectivity in the EMV process.
See also	sendServerConnectivity

onRequestTerminalTime

Signature	void onRequestTerminalTime()
Inputs	None
Outputs	None
Description	EmvSwipe is requesting the current local time in the EMV process.
See also	sendTerminalTime

onRequestFinalConfirm

Signature	void onRequestFinalConfirm()
Inputs	None
Outputs	None
Description	A final confirm request for the operator to check the transaction information such as amount before processing the EMV transaction and Generate AC.
See also	sendFinalConfirmResult

onReturnTransactionResult(deprecated)

Signature	void onReturnTransactionResult(TransactionResult result)
Inputs	result: enum of transaction result
Outputs	None
Description	Return the transaction result on an EMV transaction Enum **TransactionResult** can be APPROVED, TERMINATED, DECLINED, CANCEL, CAPK_FAIL, NOT_ICC, CARD_BLOCKED, DEVICE_ERROR, CARD_NOT_SUPPORTED, MISSING_MANDATORY_DATA, NO_EMV_APPS, INVALID_ICC_DATA CONDITION_NOT_SATISFIED, APPLICATION_BLOCKED
See also	

onReturnTransactionResult

Signature	void onReturnTransactionResult(TransactionResult result, Hashtable<String, String> data)
Inputs	result: enum of transaction result data: data for the transaction **Hashtable Keys:** *receiptData* – receipt data in TLV format
Outputs	None
Description	Return the transaction result on an EMV transaction Enum **TransactionResult** can be APPROVED, TERMINATED, DECLINED, CANCEL, CAPK_FAIL, NOT_ICC, CARD_BLOCKED, DEVICE_ERROR, CARD_NOT_SUPPORTED, MISSING_MANDATORY_DATA, NO_EMV_APPS, INVALID_ICC_DATA CONDITION_NOT_SATISFIED, APPLICATION_BLOCKED
See also	

onReturnTransactionLog(deprecated)

Signature	void onReturnTransationLog(String tlv)
Inputs	tlv: tag value results to be sent back to server for processing
Outputs	None
Description	Return the transaction data after completion of an EMV transaction.
See also	

onReturnReversalData

Signature	void onReturnReversalData(String tlv)
Inputs	tlv: tag value results to be sent back to server for processing
Outputs	None
Description	Return the reversal data after completion of an EMV transaction. ***Note***: the data in tlv is in proprietary format for FID61 devices.
See also	

onReturnBatchData

Signature	void onReturnBatchData(String tlv)
Inputs	tlv: tag value results to be sent back to server for processing
Outputs	None
Description	Return the batch data after completion of an EMV transaction. ***Note***: the data in tlv is in proprietary format for FID61 devices.
See also	

onRequestOnlineProcess

Signature	void onRequestOnlineProcess(String tlv)
Inputs	tlv: tag value results to be sent back to server for processing
Outputs	None
Description	Return data for online processing. ***Note***: the data in tlv is in proprietary format for FID61 devices.
See also	sendOnlineProcessResult

onRequestAdviceProcess(deprecated)

Signature	void onRequestAdviceProcess(String tlv)
Inputs	tlv: tag value results to be sent back to server for processing
Outputs	None
Description	Return data for advice processing.
See also	sendOnlineProcessResult

onRequestReferProcess(deprecated)

Signature	void onRequestReferProcess(String pan)
Inputs	pan: the card number to be communicated to the voice authentication operator
Outputs	None
Description	A manual voice referral process may be initiated by the card or by the issuer and the onRequestReferProcess method is triggered. The operator should call the bank to ask for the referral approval. The attendant may manually override the referral process and may accept or decline the transaction without performing a referral.
See also	sendOnlineProcessResult

onRequestDisplayText

Signature	void onRequestDisplayText(DisplayText displayMessage)
Inputs	displayMessage: enum of a display message
Outputs	None
Description	EmvSwipe has requested to display a message.
See also	onRequestClearDisplay

onRequestClearDisplay

Signature	void onRequestClearDisplay()
Inputs	None
Outputs	None
Description	EmvSwipe has requested to clear the display
See also	onRequestDisplayText

onError

Signature	void onError(Error errorState)
Inputs	errorState: Enum of the error that occurs
Outputs	None
Description	A generic method to report error
See also	See Error Enumerations section

onReturnAutoConfigResult

Signature	void onReturnAutoConfigResult(boolean isSuccess, String settings)
Inputs	isSuccess:true – auto configuration settings can be found false – auto configuration failed settings: Encrypted auto configuration settings
Outputs	None
Description	Encrypted auto configuration settings are returned. The app may save it into an external storage and load it to EmvSwipeController everytime it initializes by setConfig function
See also	startAutoConfig, setConfig

onReturnSetConfigResult

Signature	void onReturnSetConfigResult(boolean isSuccess)
Inputs	isSuccess:true – set configuration success false – set configuration failed
Outputs	None
Description	Return the result of setConfig function
See also	setConfig

Enumeration

Error

```
public enum Error {
    UNKNOWN,
    CMD_NOT_AVAILABLE,
    TIMEOUT,
    DEVICE_RESET,
    DEVICE_BUSY,
    INPUT_OUT_OF_RANGE,
    INPUT_INVALID_FORMAT,
    INPUT_ZERO_VALUES,
    INPUT_INVALID,
    CASHBACK_NOT_SUPPORTED,
    CRC_ERROR,
    COMM_ERROR
}
```

DisplayText

```
public enum DisplayText {
        AMOUNT,
        AMOUNT_OK_OR_NOT,
        APPROVED,
        CALL_YOUR_BANK,
        CANCEL_OR_ENTER,
        CARD_ERROR,
        DECLINED,
        ENTER_AMOUNT,
        ENTER_PIN,
        INCORRECT_PIN,
        INSERT_CARD,
        NOT_ACCEPTED,
        PIN_OK,
        PLEASE_WAIT,
        PROCESSING_ERROR,
        REMOVE_CARD,
        USE_CHIP_READER,
        USE_MAG_STRIPE,
        TRY_AGAIN,
        REFER_TO_YOUR_PAYMENT_DEVICE,
        TRANSACTION_TERMINATED,
        TRY_ANOTHER_INTERFACE,
        ONLINE_REQUIRED,
        PROCESSING,
        WELCOME,
        PRESENT_ONLY_ONE_CARD,
        CAPK_LOADING_FAILED,
        LAST_PIN_TRY
    }
```