



User guide of  
EMV kernel library API  
(V1.0)

2018-02-27

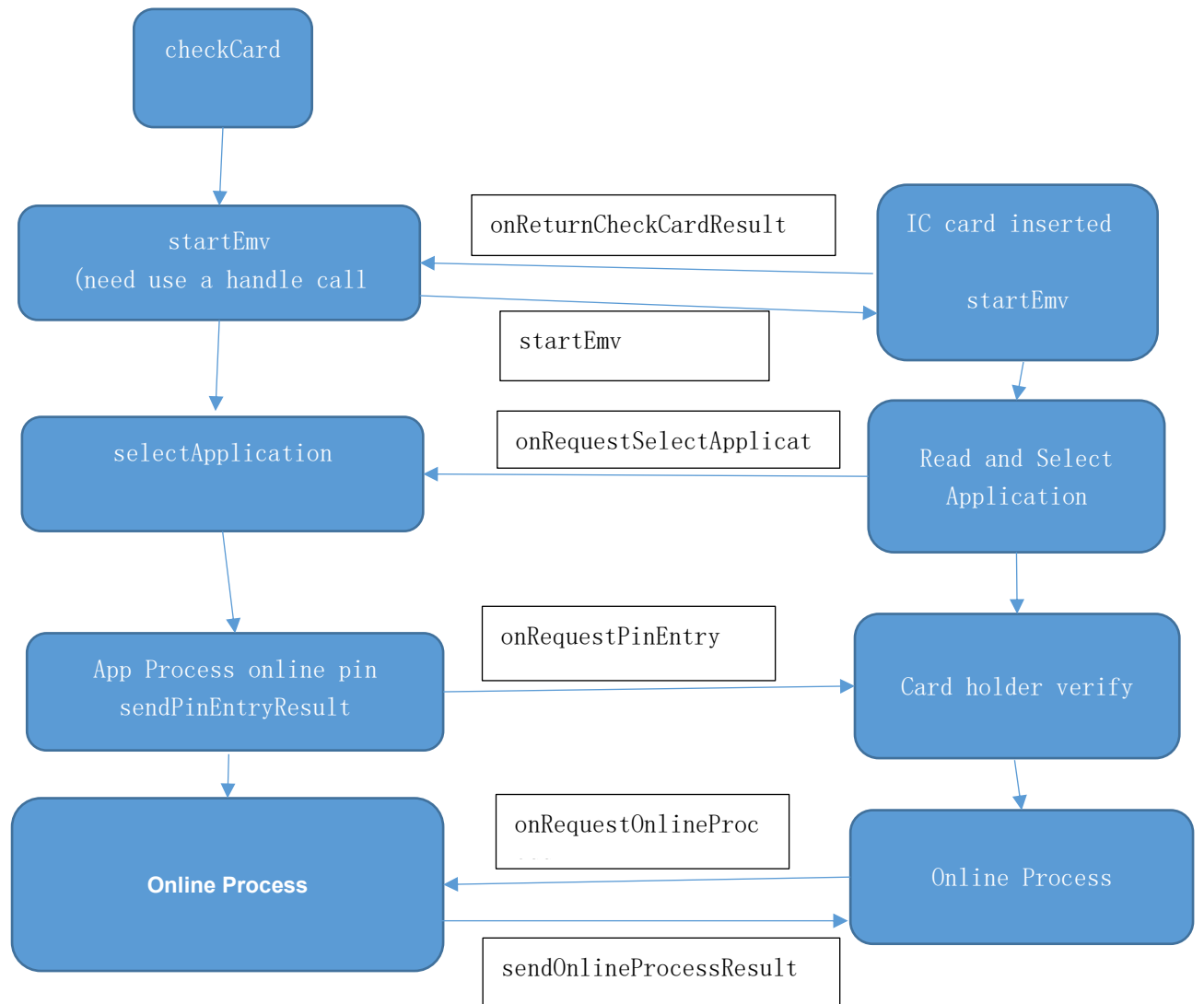
Urovo Technology Co., Ltd.

## Menu

EMV Transaction Flow .....	4
1. startEmv .....	6
2. setAmount.....	6
3. cancelSetAmount .....	7
4. cancelCheckCard .....	7
5. selectApplication.....	7
6. cancelSelectApplication .....	8
7. sendFinalConfirmResult .....	8
8. sendOnlineProcessResult .....	8
9. enableInputAmount.....	9
10. disableInputAmount .....	9
11. sendPinEntryResult .....	10
12. bypassPinEntry .....	10
13. checkCard.....	10
14. readTerminalSetting .....	11
15. updateTerminalSetting.....	11
16. updateCAPK.....	12
17. updateAID .....	12
18. onRequestSetAmount .....	13
19. onWaitingForCard .....	14
20. onReturnCancelCheckCardResult.....	14
21. onReturnCheckCardResult .....	14
22. onRequestSelectApplication .....	14
23. onRequestPinEntry.....	15
24. onRequestFinalConfirm.....	15
25. onRequestOnlineProcess .....	15
26. onReturnBatchData .....	15
27. onReturnReversalData .....	16
28. onReturnTransactionResult.....	16
29. onRequestDisplayText .....	16
30. onRequestClearDisplay .....	16
31. onReturnUpdateCAPKResult .....	16
32. onReturnUpdateAIDResult.....	17
33. getTlvByTagLists .....	17
34. clearAID.....	17
35. clearCAPK .....	18
36. getEMVLibVers .....	18
37. getEMVjarVers.....	18
38. CheckKeyStatus .....	18



## EMV Transaction Flow



## 1. startEmv

Declare:	<b>void startEmv(Hashtable&lt;String, Object&gt; data)</b>
Description:	Start the transaction with specified CheckCardMode, go through all the EMV process.
Parameters:	See below Hashtable keys table.
Return:	None
Remark:	onRequestSetAmount, onWaitingForCard

### startEmv Hashtable Keys 1/2

Transaction Settings
<ul style="list-style-type: none"><li>• <b>emvOption</b>: Indicate whether to force transaction go online, see <i>EmvOption</i> enumeration.</li><li>• <b>checkCardMode</b>: Indicate which check card mode to use, see <i>CheckCardMode</i> enumeration.</li><li>• <b>transactionType</b>: Indicate the type of financial transaction, see <i>TransactionType</i> enumeration.</li><li>• <b>amount</b>: The transaction amount.</li><li>• <b>cashbackAmount</b>: The cashback amount.</li><li>• <b>currencyCode</b>: The 3-digits transaction currency code (e.g. “840” for USD).</li><li>• <b>currencyCharacters</b>: An array of 3 symbols long currency characters to be displayed on LCD, see <i>CurrencyCharacter</i> enumeration.</li><li>• <b>disableQuickChip</b>: If Quick Chip is enabled in firmware config, set True to disable Quick Chip for this transaction. Set False or no such key, firmware will follow the config setting.</li><li>• <b>checkCardTimeout</b>: Indicate the check card timeout, in second.</li></ul>

## 2. setAmount

Declare:	<b>boolean setAmount(String amount, String cashbackAmount, String currencyCode, TransactionType transactionType, CurrencyCharacter[] currencyCharacters)</b>
Description:	Set the amount, cashback amount, currency and transaction type in response to onRequestSetAmount. This function can be called before a transaction. The API will store the values and

	<p>send to device after startEmv. If the function return False, the application should wait for onError and handle the error before proceeding further to startEmv.</p> <ul style="list-style-type: none"> <li>• <b>amount</b> : The transaction amount.</li> <li>• <b>cashbackAmount</b> : The cashback amount.</li> <li>• <b>currencyCode</b> : The 3-digits transaction currency code (e.g. “840” for USD).</li> <li>• <b>currencyCharacters</b> : An array of 3 symbols long currency characters to be displayed on LCD, see <i>CurrencyCharacter</i> enumeration.</li> </ul>
Parameters:	
Return:	<p>True - Amount values are set. False - Failed to set values, check onError for details.</p>
Remark:	onRequestSetAmount, onReturnAmountConfirmResult, onError

### 3. cancelSetAmount

Declare:	<b>void cancelSetAmount()</b>
Description:	Cancel the EMV transaction at the amount entry step. Should call at onRequestSetAmount.
Parameters:	None
Return:	None
Remark:	None

### 4. cancelCheckCard

Declare:	<b>void cancelCheckCard()</b>
Description:	Cancel the check card process. Should call at onWaitingForCard.
Parameters:	None
Return:	None
Remark:	onReturnCancelCheckCardResult

### 5. selectApplication

Declare:	<b>void selectApplication(int index)</b>
----------	--

Description:	An chip card may support multiple payment applications. The list of Applications
	IDs supported by the card and device is returned in
	onRequestSelectApplication.
	The App should prompt the customer for an application to
	continue the
Parameters:	transaction.
Return:	The index of the application selected.
Remark:	None
	onRequestSelectApplication

## 6. cancelSelectApplication

Declare:	<code>void cancelSelectApplication()</code>
Description:	Cancel the select application process. Should call at
Parameters:	onRequestSelectApplication.
Return:	None
Remark:	None

## 7. sendFinalConfirmResult

Declare:	<code>void sendFinalConfirmResult(boolean isConfirmed)</code>
Description:	Send the final confirmation to proceed or cancel the transaction
Parameters:	in response to onRequestFinalConfirm.
Return:	<ul style="list-style-type: none"> <li>• <b>isConfirmed</b>: Indicate the confirmation result. True - Confirm</li> </ul>
Remark:	to continue the transaction. False - Cancel the transaction.
	None
	onRequestFinalConfirm

## 8. sendOnlineProcessResult

Declare:	<code>void sendOnlineProcessResult(String tlv)</code>
Description:	Send back the online process result to the card in response to
	onRequestOnlineProcess. To terminate the transaction, send
	null or empty string as result.



Parameters:	<p>A set of transaction results in Tag-Length-Value format. The following tags are usually returned to the card:</p> <ul style="list-style-type: none"> <li>• Tag 8A: Authorisation Response Code</li> <li>• Tag 91: Issuer Authentication Data</li> <li>• Tag 71: Issuer Script Template 1</li> <li>• Tag 72: Issuer Script Template 2</li> </ul> <p>Example: String tlv = "8A02303091083132333435363738"</p>
Return:	None
Remark:	onRequestOnlineProcess

## 9. enableInputAmount

Declare:	<code>void enableInputAmount(Hashtable&lt;String, Object&gt; data)</code>
Description:	<p>Enable amount input on device with keypad before startEmv. To cancel the input amount, call disableInputAmount. The amount will be returned by onReturnAmount callback.</p> <p>Hashtable keys</p> <ul style="list-style-type: none"> <li>• <b>amountInputType</b>: Indicate the type of amount input, see <i>AmountInputType</i> enumeration.</li> </ul>
Parameters:	<ul style="list-style-type: none"> <li>• <b>currencyCode</b> 1: The 3-digits transaction currency code (e.g. "840" for USD).</li> <li>• <b>currencyCharacters</b> 2: An array of 3 symbols long currency characters to be displayed on LCD, see <i>CurrencyCharacter</i> enumeration.</li> </ul>
Return:	None
Remark:	onReturnEnableInputAmountResult, onReturnAmount

## 10. disableInputAmount

Declare:	<code>void disableInputAmount()</code>
Description:	Disable amount input on device with keypad before startEmv.
Parameters:	None
Return:	None
Remark:	onReturnDisableInputAmountResult

## 11. sendPinEntryResult

Declare:	<code>void sendPinEntryResult(String pin)</code>
Description:	Send the plaintext PIN in response to <code>onRequestPinEntry</code> with Phone as <code>PinEntrySource</code> , see <i>PinEntrySource</i> enumeration.
Parameters:	Plaintext PIN Example: <code>String pin = "";</code> //if use system Pinpad
Return:	None
Remark:	<code>onReturnPinEntryResult</code>

## 12. bypassPinEntry

Declare:	<code>void bypassPinEntry()</code>
Description:	Bypass the PIN entry step in response to <code>onRequestPinEntry</code> with Phone as <code>PinEntrySource</code> . If the card does not accept bypassing, the transaction will be aborted.
Parameters:	None
Return:	None
Remark:	<code>onReturnPinEntryResult</code>

## 13. checkCard

Declare:	<code>void checkCard(Hashtable&lt;String, Object&gt; data)</code>
Description:	Checks if a card has been swiped, inserted or tapped. The result is returned by the <code>onReturnCheckCardResult</code> callback. To cancel the check card process, call <code>cancelCheckCard</code> at <code>onWaitingForCard</code> .
Parameters:	Hashtable keys • <b>checkCardMode</b> : Indicate which check card mode to use, see <i>CheckCardMode</i> enumeration. • <b>checkCardTimeout</b> : Indicate the check card timeout, in second.
Return:	None
Remark:	<code>onWaitingForCard</code> , <code>onReturnCheckCardResult</code>

## 14. readTerminalSetting

Declare:	void readTerminalSetting(String tag)
Description:	Read a terminal configuration parameter, provided that the tag is supported and allow to read.
Parameters:	A tag of the terminal configuration parameter. See below table for the list of tags.
Return:	None
Remark:	onReturnReadTerminalSettingResult

## 15. updateTerminalSetting

Declare:	void updateTerminalSetting(String tlv)
Description:	Update a terminal configuration parameter, provided that the tag is supported and allow to update.
Parameters:	A terminal configuration parameter in Tag-Length-Value format. See below table for the list of tags.
Return:	None
Remark:	onReturnUpdateTerminalSettingResult

Tag	Format	Length	Length
9F01	n6-11	6 bytes	Acquirer Identifier.
9F1A	n3	2 bytes	Terminal Country Code.
5F2A	n3	2 bytes	Transaction Currency Code.
5F36	n1	1 byte	Transaction Currency Exponent.
9F4E	Ans0-40	Variable 0-40	Merchant Name and Location.
9F16	ans15	15 bytes	Merchant Identifier.
9F1C	an8	8 bytes	Terminal Identification.
9F33	b	3 bytes	Terminal Capabilities. (Read-Only)
9F35	n2	1 byte	Terminal Type. (Read-Only)

## 16. updateCAPK

Declare:	<code>void updateCAPK(CAPK capk)</code>
Description:	<p>Update a Certificate Authority Public Key object in the device. The CAPK object to be updated. The object contains the location of the CAPK in the device, the RID, index, modulus, exponent and the checksum of the CAPK, all data in hex string format.</p> <p>To remove a CAPK entry at the specified location, set with the following values:</p>
Parameters:	<ul style="list-style-type: none"><li>• Location (1 byte) = "XX"</li><li>• RID (5 bytes) = "0000000000"</li><li>• Index (1 byte) = "01"</li><li>• Exponent (1 byte or 3 bytes) = "03"</li><li>• Modulus (N bytes) = "00"</li><li>• Size (2 bytes) = not use</li><li>• Checksum (20 bytes) = SHA-1 [RID    Index    Modulus    Exponent]</li></ul>
Return:	None
Remark:	onReturnUpdateCAPKResult

## 17. updateAID

Declare:	<code>void updateAID(Hashtable&lt;String, String&gt; data)</code>
Description:	<p>Update the AID info.</p> <p><i>Note 1:</i> To add new AID, calls readAID to locate an appIndex with AID 16 bytes 0xFF, then call updateAID to update its info.</p> <p><i>Note 2:</i> To remove an App, set AID value to 16 bytes 0xFF.</p> <p><i>Note 3:</i> The online config form will not in sync with firmware config after modifying the AID info with updateAID command.</p>

	<p>Hashtable keys</p> <ul style="list-style-type: none"> <li>• <b>applIndex</b>: The index of the AID to be updated.</li> <li>• <b>aid</b>: Application Identifier (AID).</li> <li>• <b>applIndex</b>: Application Index of the AID in config.</li> <li>• <b>appVersion</b>: Application Version Number.</li> <li>• <b>contactTACDefault/contactTACDenial/contactTACOnline</b>: Contact Terminal Action Code - Default/Denial/Online.</li> <li>• <b>contactlessTACDefault/contactlessTACDenial/contactlessTACOnline</b>: Contactless Terminal Action Code - Default/Denial/Online.</li> <li>• <b>defaultTDOL</b>: Transaction Certificate Data Object List (TDOL).</li> <li>• <b>defaultDDOL</b>: Dynamic Data Authentication Data Object List (DDOL).</li> <li>• <b>contactlessTransactionLimit</b>: Contactless Transaction Limit.</li> <li>• <b>contactlessTransactionLimitODCV</b>: Contactless Transaction Limit with ODCV.</li> <li>• <b>contactlessCVMRequiredLimit</b>: Contactless CVM Required Limit.</li> <li>• <b>contactlessFloorLimit</b>: Contactless Floor Limit.</li> <li>• <b>terminalFloorLimit</b>: Contact Terminal Floor Limit.</li> </ul> <p>Hashtable Values</p> <ul style="list-style-type: none"> <li>• Data in hex string format.</li> </ul>
Parameters:	
Return:	None
Remark:	onReturnUpdateAIDResult

## Transaction

### 18. onRequestSetAmount

Declare:	<code>void onRequestSetAmount()</code>
Description:	Request input of amount in the EMV process. If amount is input at startEmv, this callback will not be triggered. To cancel the process, call cancelSetAmount.
Return:	None

## 19. onWaitingForCard

Declare:	<code>void onWaitingForCard(CheckCardMode checkCardMode)</code>
Description:	Indicate the device is ready to read a card with the supported <i>CheckCardMode</i> interface. To cancel the process, call <code>cancelCheckCard</code> . This callback will be triggered multiple times when a chip card is swiped or a fallback occurred. The App should prompt customer to use another interfaces or cards to continue with the transaction.
Return:	Return the device supported check card mode, see <i>CheckCardMode</i> enumeration.

## 20. onReturnCancelCheckCardResult

Declare:	<code>void onReturnCancelCheckCardResult(boolean isSuccess)</code>
Description:	Return whether the checkCard process has been stopped successfully.
Return:	True - Stopped successfully. False - Otherwise.

## 21. onReturnCheckCardResult

Declare:	<code>void onReturnCheckCardResult(CheckCardResult checkCardResult, Hashtable&lt;String, String&gt; decodeData)</code>
Description:	Return the results in response to a swiped, inserted or tapped card. <ul style="list-style-type: none"><li>• <b>encTrack1, encTrack2, encTrack3, encTracks</b>: Encrypted track data, in hex string format.</li></ul>
Return:	<ul style="list-style-type: none"><li>• <b>posEntryMode</b>: POS Entry Mode.</li><li>• <b>expiryDate</b>: Card expiry date, in the YYMM format.</li></ul>

## 22. onRequestSelectApplication

Declare:	<code>void onRequestSelectApplication(ArrayList&lt;String&gt; appList)</code>
Description:	Request selection of an application from the returned list. To

Return:	cancel the process, call <code>cancelSelectApplication</code> . Return a list of applications supported by the mPOS device and the EMV chip card. The applications are sorted by priority, where index 0 is the highest priority.
---------	--

## 23. onRequestPinEntry

Declare:	<code>void onRequestPinEntry(PinEntrySource pinEntrySource)</code>
Description:	Request PIN entry from the specified <code>PinEntrySource</code> . To cancel the process, call <code>cancelPinEntry</code> . This callback can be triggered by <code>startEmv</code> or <code>startPinEntry</code> .
Return:	Indicate the PIN entry source, see <i>PinEntrySource</i> enumeration.

## 24. onRequestFinalConfirm

Declare:	<code>void onRequestFinalConfirm()</code>
Description:	The EMV process request for a final confirmation before calling the first generate AC command. To proceed or cancel the transaction, call <code>sendFinalConfirmResult</code> .
Return:	None

## 25. onRequestOnlineProcess

Declare:	<code>void onRequestOnlineProcess(String tlv, String dataKsn)</code>
Description:	Request the transaction go online. To proceed or terminate the transaction, call <code>sendOnlineProcessResult</code> . A set of transaction data in Tag-Length-Value format that should be sent back to online server.
Return:	TLV : TLV data dataKsn: KSN for data encryption

## 26. onReturnBatchData

Declare:	<code>void onReturnBatchData(String tlv)</code>
Description:	Return the batch data after completion of an EMV transaction.
Return:	A set of transaction data in Tag-Length-Value format that should be sent back to server for processing.

## 27. onReturnReversalData

Declare:	<code>void onReturnReversalData(String tlv)</code>
Description:	If the transaction is stopped after the communication has begun with the processor, or the issuer has approved but card denied the transaction, a reversal will be returned.
Return:	A set of transaction data in Tag-Length-Value format that should be sent back to server for processing.

## 28. onReturnTransactionResult

Declare:	<code>void onReturnTransactionResult(TransactionResult transResult)</code>
Description:	Return the transaction result.
Return:	Indicate the transaction result, see TransactionResult enumeration.

## 29. onRequestDisplayText

Declare:	<code>void onRequestDisplayText(DisplayText displayText)</code>
Description:	Return the message to prompt customer.
Return:	Indicate the message to be displayed, see DisplayText enumeration.

## 30. onRequestClearDisplay

Declare:	<code>void onRequestClearDisplay()</code>
Description:	Clear the displayed message.
Return:	None

## 31. onReturnUpdateCAPKResult

Declare:	<code>void onReturnUpdateCAPKResult(boolean isSuccess)</code>
Description:	Return the update result of a specified Certificate Authority Public Key in response to updateCAPK.



Return:	<p><i>Note:</i> The CAPK version returned in <code>getDeviceInfo</code> will not change after successful update. It is the version injected during production or remote updated through OTA.</p> <p>True - The CAPK object is updated. False - Failed to update the CAPK object.</p>
---------	--

## 32. onReturnUpdateAIDResult

Declare:	void onReturnUpdateAIDResult(Hashtable<String, String> data)
Description:	<p>Return the AID update result. Hashtable keys</p> <ul style="list-style-type: none"> <li>Refer to <i>updateAID</i> Hashtable keys.</li> </ul>
Return:	<p>Hashtable Values</p> <ul style="list-style-type: none"> <li>Indicate the update result, see <i>TerminalSettingStatus</i> enumeration.</li> </ul>

## 33. getTlvByTagLists

Declare:	public String getTlvByTagLists(List<String> TagList)
Description:	Get the TLV string based on tag
Parameters:	TagList:All tags
Return:	TLV string
Remark:	none

## 34. clearAID

Declare:	public void clearAID()
Description:	Delete all AID paramer
Parameters:	none
Return:	none
Remark:	none

### 35. clearCAPK

Declare:	public void clearCAPK()
Description:	Delete all CAPK paramer
Parameters:	none
Return:	none
Remark:	none

### 36. getEMVLibVers

Declare:	public String getEMVLibVers()
Description:	Get emv so version
Parameters:	none
Return:	Version string
Remark:	none

### 37. getEMVjarVers

Declare:	public String getEMVjarVers()
Description:	Get emv jar version
Parameters:	none
Return:	Version string
Remark:	none

### 38. CheckKeyStatus

Declare:	public int CheckKeyStatus()
Description:	DUKPT exist or not
Parameters:	none
Return:	0: normal -1: no key or key lost
Remark:	none

```

public static enum Error
{
    INPUT_INVALID, INPUT_OUT_OF_RANGE, INPUT_INVALID_FORMAT,
    CASHBACK_NOT_SUPPORTED,
    COMM_ERROR, CRC_ERROR, TIMEOUT, CMD_NOT_AVAILABLE, DEVICE_BUSY,
    COMM_LINK_UNINITIALIZED,
    INVALID_FUNCTION_IN_CURRENT_CONNECTION_MODE,
    FAIL_TO_START_AUDIO, VOLUME_WARNING_NOT_ACCEPTED,
    BTV4_NOT_SUPPORTED, FAIL_TO_START_BT, AUDIO_PERMISSION_DENIED,
    BLUETOOTH_PERMISSION_DENIED,
    BLUETOOTH_LOCATION_PERMISSION_DENIED, SERIAL_PERMISSION_DENIED,
    USB_DEVICE_PERMISSION_DENIED,
    USB_DEVICE_NOT_FOUND, USB_NOT_SUPPORTED, FAIL_TO_START_SERIAL,
    HARDWARE_NOT_SUPPORTED, PCI_ERROR,
    UNKNOWN

}

public static enum EmvOption
{
    START, START_WITH_FORCE_ONLINE
}

public static enum CheckCardMode{
    SWIPE, INSERT, TAP, SWIPE_OR_INSERT, SWIPE_OR_TAP, INSERT_OR_TAP,
    SWIPE_OR_INSERT_OR_TAP, MANUAL_PAN_ENTRY
}

public static enum TransactionType
{
    GOODS, SERVICES, CASHBACK, INQUIRY, TRANSFER, PAYMENT, REFUND,
    VOID, REVERSAL

}

public static enum AmountInputType{
    AMOUNT_ONLY, CASHBACK_ONLY, AMOUNT_AND_CASHBACK
}

public static enum CardScheme{
    VISA, MASTER, UNIONPAY
}

public static enum CheckCardResult{
    NO_CARD, INSERTED_CARD, NOT_ICC, BAD_SWIPE, MSR, MAG_HEAD_FAIL,
    USE_ICC_CARD,
    TAP_CARD_DETECTED, MANUAL_PAN_ENTRY
}

```

```

    }

    public static enum TransactionResult
    {
        APPROVED, TERMINATED, DECLINED, CANCELED, TIMEOUT,
        CANCELED_OR_TIMEOUT, CAPK_FAIL,
        NOT_ICC, CARD_BLOCKED, DEVICE_ERROR, NO_EMV_APPS,
        ICC_CARD_REMOVED, CARD_SCHEME_NOT_MATCHED,
        CARD_NOT_SUPPORTED, MISSING_MANDATORY_DATA, SELECT_APP_FAIL,
        INVALID_ICC_DATA, CONDITION_NOT_SATISFIED,
        APPLICATION_BLOCKED
    }

    public static enum TerminalSettingStatus{
        SUCCESS, INVALID_TLV_FORMAT, TAG_NOT_FOUND, INVALID_LENGTH,
        BOOTLOADER_NOT_SUPPORTED, TAG_NOT_ALLOWED_TO_ACCESS,
        TAG_NOT_WRITTEN_CORRECTLY, INVALID_VALUE
    }

    public static enum DisplayText
    {
        APPROVED, CALL_YOUR_BANK, DECLINED, ENTER_AMOUNT, ENTER_PIN,
        INCORRECT_PIN, INSERT_CARD,
        NOT_ACCEPTED, PIN_OK, PLEASE_WAIT, REMOVE_CARD, USE_MAG_STRIPE,
        TRY_AGAIN,
        PRESENT_CARD, PROCESSING, INSERT_OR_SWIPE_CARD,
        MULTIPLE_CARDS_DETECTED,
        APPROVED_PLEASE_SIGN, AUTHORISING,
        INSERT_SWIPE_OR_TRY_ANOTHER_CARD, REFER_TO_YOUR_PAYMENT_DEVICE,
        PRESENT_CARD_AGAIN, LAST_PIN_TRY, TRANSACTION_TERMINATED,
        SELECT_ACCOUNT, TIMEOUT,
        APPLICATION_EXPIRED, FINAL_CONFIRM, SHOW_THANK_YOU,
        PIN_TRY_LIMIT_EXCEEDED, NOT_ICC_CARD,
        CARD_INSERTED, CARD_REMOVED, NO_EMV_APPS

    }

    public static enum AccountSelectionResult{
        SUCCESS, CANCEL, TIMEOUT, INVALID_SELECTION
    }

    public static enum PinEntryResult{
        ENTERED, CANCEL, TIMEOUT, BYPASS, WRONG_PIN_LENGTH, INCORRECT_PIN
    }

    public static enum PinEntrySource{
        PHONE, KEYPAD
    }

```

