

Transaction Payment Solutions International Limited



ANDROID 3RD PARTY INTEGRATION API DOCUMENTATION



Change History

<i>Rev</i>	<i>Date</i>	<i>Who</i>	<i>Why</i>
1.0	24 November 2021	Pravesh Auckloo	Baseline
1.1	04 February 2022	Pravesh Auckloo	Added setEnvValues
1.2	03 October 2022	Pravesh Auckloo	Renaming of library
1.3	27 October 2023	Pravesh Auckloo	Added Response code description to the result
1.4	01 November 2023	Pravesh Auckloo	Added Reprint feature
1.5	09 October 2024	Edwin Dzamara	Added queryTransList
1.6	10 October 2024	Edwin Dzamara	Added Response Data Extra for UtiList Renamed QueryTransList to queryUtiList
1.7	11 October 2024	Edwin Dzamara	Added Mandatory permissions for PAX Android 10.x
1.8	06 March 2025	Pravesh Auckloo	Update getPaymentConfig() return value
1.9	29 May 2025	Pravesh Auckloo	Update setAirplanMode function Add generic RunTrans
1.10	03 July 2025	Pravesh Auckloo	Update query UTI limit

Associated Documents

<i>Title</i>	<i>Version</i>

Table of Contents

Contents

Table of Contents.....	1
1. INTRODUCTION	3
1.1. Purpose.....	3
1.2. Audience.....	3
1.3. Conditions.....	3
1.4. Related Documents	3
2. TestLauncher Demo Application.....	4
3. Transaction set	5
3.1. runTrans.....	6
3.2. runTrans force online.....	6
3.3. runTrans with virtual merchant data.....	7
3.4. runReversal	7
3.5. runReversal in silent mode.....	8
3.6. runNormalFleet	8
3.7. runFleetInquiry.....	9
3.8. runCardlessFleet.....	9
3.9. runEcoPay	10
3.10. queryTrans	10
3.11. runAdminFunction.....	10
3.12. runReconciliation.....	11
3.13. reprintRec	11
3.14. getPaymentConfig	11
3.15. isFlightModeEnable	11
3.16. setAirplanMode.....	12
3.17. setEnvValues	12
3.18. reprintReceipt	12
3.19. queryUtilList	13
3.20. runTrans	13
3.21. Processing response.....	13
3.22. TPSServiceTransResult.....	13
3.23. Additional extra data in the response.....	15
3.24. Additional extra data in the request.....	16

ANDROID 3RD PARTY APPLICATION INTEGRATION API

Statement of confidentiality and non-disclosure

This document contains proprietary and confidential information. All data submitted is provided in reliance upon its consent not to use or disclose any information contained herein except in the context of its business dealings with Liquid Telecom. The recipient of this document agrees to inform anyone who views or has access to the content of its confidential nature.

The recipient agrees to instruct anyone with access to the document that they must not disclose any information concerning this document to others except to the extent those matters are generally known to and are available for use by the public. The recipient also agrees not to duplicate or distribute or permit others to duplicate or distribute any material contained herein without Liquid Telecoms' express written consent.

BY ACCEPTANCE OF THIS DOCUMENT, THE RECIPIENT AGREES TO BE BOUND BY THE AFOREMENTIONED STATEMENT

1. INTRODUCTION

1.1. Purpose

Android 3rd party application integration API is used to describe the methodology of using the API functions.

1.2. Audience

This document is primarily targeted at Android developers who are expecting to create their Android application that is going to run on the PAX android terminal and integrate with Liquid Telecom payment application Apay.

1.3. Conditions

The prerequisites for this document are:

- Basic understanding of Android application development
- Basic knowledge of Liquid Telecom payment application transaction processing and flow
- A PAX Android terminal
- PAX Android SDK (NeptuneLite API)
- PAX android terminal installed with the latest EftService & Apay from Liquid Telecom
- TpsServiceLib API provided by Liquid Telecom
- TestLauncher application source code to easy integration work

1.4. Related Documents

NeptuneLite API documentation

2. TestLauncher Demo Application

The TestLauncher Demo application is a very simple application that was written to ease the work of 3rd party application developers.

The application consists of different buttons defined in the main activity for the different transactions type (see MainActivity.java).

Comments have been added to the source code to explain how it works.

The code relies on a library that is included called tpsServiceLib-release.aar. Please look at the manifest and build.gradle files to see how it is integrated.

The IPC comms relies on a BroadcastReceiver being used to send and receive the messages. Please look at the manifest to see how this should be declared. The example implementation is in the TestLauncherReceiver.java

It is the responsibility of the calling application to return itself to the foreground once the transaction is completed. The TestLauncher application also does that in the TestReceiver.java by calling startActivity () once the result has been received.

In summary, you should be able to declare a receiver and use the included library to do the IPC comms to and from the Liquid Telecom payment application (APay).

Note: we intend to update the tpsServiceLib API with more functionality as required by the customer. Please send requests or comments to POSService@tpspay.com.

ANDROID 3RD PARTY APPLICATION INTEGRATION API

3. Transaction set

Below is the list of transaction support for the moment through the tpsServiceLib API, note that the transaction type name is case-sensitive.

More transaction types will be added based on your request.

All the transactions type may not be available for all customers since this is controlled through our LMP portal as well as what transaction set you requested Liquid Telecom to implement for you.

Transaction type	Transaction Name
Test connect	TRANSACTION_TYPE_TESTCONNECT
Sales	TRANSACTION_TYPE_SALE
Balance	TRANSACTION_TYPE_BALANCE
Refund	TRANSACTION_TYPE_REFUND
Purchase cashback with	TRANSACTION_TYPE_PURCHASEWITHCASHBACK
Cash	TRANSACTION_TYPE_CASH
Ecosales	TRANSACTION_TYPE_ECOSALE
Card fleet	TRANSACTION_TYPE_NORMALFLEET
Cardless fleet inquiry	TRANSACTION_TYPE_CARDLESSFLEETINQUIRY
Cardless fleet	TRANSACTION_TYPE_CARDLESSFLEET

Table 1: List of financial transactions

Transaction type	Transaction Name
Test connect	TRANSACTION_TYPE_TESTCONNECT
Balance	TRANSACTION_TYPE_BALANCE
Reconciliation	TRANSACTION_TYPE_RECONCILIATION

Table 2: List of administrative functions

The transaction currency cannot be changed from 3rd party application, the transaction currency depends on what is configured on LMP & Apay.

The transaction amount is in the smallest currency denomination, for example,

100 = \$1.00

1550 = \$15.50

3.1. runTrans

Prototype	Boolean runTrans(android.content.Context context, int i, String s)		
Function	Send a request to run a transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction amount	
	Parameter 3	Transaction name as per Table 1	
Return	True if transaction initiated successfully. False invalid parameters are passed		

3.2. runTrans force online

Prototype	boolean runTrans(android.content.Context context, int i, String s, String s1, String s2, String s3, boolean b)		
Function	Send a request to run a transaction with virtual merchant details and force it to go online even if the terminal is in flight mode		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction amount	
	Parameter 3	Transaction name as per Table 1	
	Parameter 4	Virtual TID	
	Parameter 5	Virtual MID	
	Parameter 6	Virtual merchant name	
	Parameter 7	Force Online	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.3. runTrans with virtual merchant data

Prototype	boolean runTrans(android.content.Context context, int i, String s, String s1, String s2, String s3)		
Function	Send a request to run a transaction with virtual merchant details		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction amount	
	Parameter 3	Transaction name as per Table 1	
	Parameter 4	Virtual TID	
	Parameter 5	Virtual MID	
	Parameter 6	Virtual merchant name	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.4. runReversal

Prototype	boolean runReversal(android.content.Context context, int i, int i1)		
Function	Send a request to reverse the last authorized transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Original transaction amount	
	Parameter 3	Original receipt number	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.5. runReversal in silent mode

Prototype	boolean runReversal(android.content.Context context, int i, String s, boolean b, boolean b1)		
Function	Send a request to reverse a specific transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Original transaction amount	
	Parameter 3	Original UTI to be reversed	
	Parameter 4	Disable receipt printing	
	Parameter 5	Run in silent mode	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.6. runNormalFleet

Prototype	boolean runNormalFleet(android.content.Context context, int i, String s, String s1)		
Function	Send a request to run a fleet card transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction amount	
	Parameter 3	Transaction name as per Table 1	
	Parameter 4	Fleet data	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.7. runFleetInquiry

Prototype	boolean runFleetInquiry(android.content.Context context, String s, String s1, String s2, String s3)		
Function	Send a request to get the available funds in an account to process a fleet transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction name as per Table 1	
	Parameter 3	token	
	Parameter 4	Expiry date	
	Parameter 5	Fleet data	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.8. runCardlessFleet

Prototype	boolean runCardlessFleet(android.content.Context context, int i, String s, String s1, String s2, String s3)		
Function	Send a request to run a cardless fleet transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction amount	
	Parameter 3	Transaction name as per Table 1	
	Parameter 4	token	
	Parameter 5	Expiry Date	
	Parameter 6	Fleet data	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.9. runEcoPay

Prototype	boolean runEcoPay(android.content.Context context, String s, int i, String s1)		
Function	Send a request to run an Ecopay transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Ecopay phone number	
	Parameter 3	Transaction amount	
	Parameter 4	Transaction name as per Table 1	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.10. queryTrans

Prototype	boolean queryTrans(android.content.Context context, String s)		
Function	Send a request to query a transaction to get all the details of the transaction		
Parameters	Parameter 1	Context	
	Parameter 2	UTI	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.11. runAdminFunction

Prototype	boolean runAdminFunction(android.content.Context context, String s)		
Function	Send a request to run a test connect or balance transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction name as per Table 2	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.12. runReconciliation

Prototype	boolean runReconciliation(android.content.Context context, String s)		
Function	Send a request to run a reconciliation		
Parameters	Parameter 1	Context	
	Parameter 2	Transaction name as per Table 2	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.13. reprintRec

Prototype	void reprintRec(android.content.Context context)		
Function	Send a request to reprint the last reconciliation		
Parameters	Parameter 1	Context	
Return	none		

3.14. getPaymentConfig

Prototype	void getPaymentConfig(android.content.Context context)		
Function	Send a request to get the payment configuration details		
Parameters	Parameter 1	Context	
Return	None – Check eft.com.CONFIGURATION_RESULT for return values		

3.15. isFlightModeEnable

Prototype	boolean isFlightModeEnabled(android.content.Context context)		
Function	Check if the terminal is in flight mode		
Parameters	Parameter 1	Context	
Return	True: - flight mode is enabled False: - flight mode is not enabled		

3.16. setAirplaneMode

Prototype	void setAirplaneMode(android.content.Context context, boolean enable)		
Function	Toggle flight mode		
Parameters	Parameter 1	Context	
	Parameter 2	True – to enable flight mode False – to disable flight mode	
Return	None		

3.17. setEnvValues

Prototype	void setEnvValues(android.content.Context context, int i1, int i2, int i3)		
Function	Set the power-off protected variables batch number, stan, andreceipt number		
Parameters	Parameter 1	Context	
	Parameter 2	Batch number maximum value 999	
	Parameter 3	Stan maximum value 9999999	
	Parameter 4	Receipt number maximum value 999999	
Return	None		

3.18. reprintReceipt

Prototype	boolean reprintReceipt(android.content.Context context, int receiptNumber)		
Function	Send a request to Apay to reprint a transaction receipt		
Parameters	Parameter 1	Context	
	Parameter 2	Receipt Number	
Return	True: - transaction initiated successfully False: - invalid parameters are passed		

3.19. queryUtilist

Prototype	boolean queryUtilist(android.content.Context context, int count)		
Function	Send a request to query a transaction to get all the details of the transaction		
Parameters	Parameter 1	Context	
	Parameter 2	Total Transactions to query, max 20	
Return	<p>True: - transaction initiated successfully – The list is sorted in descending order, that is the latest transaction comes first in the list</p> <p>False: - list is null/count is greater than 20</p>		

3.20. runTrans

Prototype	boolean runTrans(android.content.Context context, TPSServiceTransEvent transEvent)		
Function	Send a transaction with multiple variables inside transEvent		
Parameters	Parameter 1	Context	
	Parameter 2	transEvent – See table 4	
Return	<p>True: - transaction initiated successfully</p> <p>False: - invalid parameters are passed</p>		

3.21. Processing response

Apay will broadcast the result response using intent and TPSServiceTransResult.

3.22. TPSServiceTransResult

Prototype	boolean isApproved()
Function	Check if a transaction is approved or declined
Parameters	
Return	<p>True if the transaction is approved.</p> <p>False if the transaction is declined</p>
Remarks	Use intent.getStringExtra("ResponseCode") to get the response code if the transaction is declined.

ANDROID 3RD PARTY APPLICATION INTEGRATION API

Prototype	int getReceiptNumber()
Function	Retrieve the receipt number
Parameters	
Return	Return the receipt number

Prototype	String getRrn()
Function	Retrieve the rrn
Parameters	
Return	Return the RRN

Prototype	long getAmount()
Function	Retrieve the transaction amount
Parameters	
Return	Return the amount of the transaction in the smallest currency denomination

Prototype	String getTransType()
Function	Retrieve the transaction type
Parameters	
Return	Return the transaction type as the per Table 1 naming format

3.23. Additional extra data in the response

You can add additional response data in the intent, refer to TestLauncherReceiver.java onReceive() class.

Note that extra data are case-sensitive. Find below the full list of extra response data currently supported. We intend to update the intent extra with more variables as required by the customer. Please send requests or comments to POSService@tpspay.com.

Date Type	Name
String	UTI
String	MsgStatus
Boolean	Approved
Boolean	Cancelled
Boolean	SigRequired
Boolean	PINVerified
String	AuthMode
String	Currency
String	Tid
String	Mid
String	Version
String	Header1
String	Header2
String	Header3
String	Header4
String	Header5
String	ResponseCode
String	ResponseCodeDescription
Int	Stan
String	AuthCode
String	AvailableBalance
String	BatchNumber
String	AID
String	TSI
String	TVR
String	CardHolder

ANDROID 3RD PARTY APPLICATION INTEGRATION API

String	Cryptogram
String	CryptogramType
String	CardName
String	PAN
String	ExpiryDate
String	StartDate
String	Scheme
Int	PSN
ArrayList	UtiList
String	Version

Table 3: List of supported data extra

3.24. Additional extra data in the request

Date Type	Function Name
Long	setAmount()
Int	setBatchNumber()
Double	setCbAmount()
Int	setCount()
String	setCurrencyCode()
String	setDateTimyyyMMddhhmmss()
Boolean	setDisableChecks()
Boolean	setDisablePrinting()
Boolean	setDisableRestarts()
Boolean	setEcr()
Double	setExtraAmount()
String	setFlightReference()
Boolean	setForceOnline()
Boolean	setIncorrectSignature()
Boolean	setOnslip()
String	setOriginatingAppName()
Int	setReceiptNumber()
String	setReferenceNumber()

ANDROID 3RD PARTY APPLICATION INTEGRATION API

Byte	setSendReceipt()
Int	setStan()
Boolean	setTimeout()
String	setTransType()
Double	setTxAmount()
String	setUti()
Double	setVatAmount()
String	CardHolder
String	Cryptogram

Table 4: List of supported data input

3.25. Extra Mandatory permissions for Pax Android 10.x

Android 10.x Mandatory permissions

```
<uses-permission
android:name="com.pax.permission.PAX_START_ACTIVITIES_FROM_BACKGROUND"
```

ANDROID 3RD PARTY APPLICATION INTEGRATION API

```
/>
```

```
<uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
```