JioMoney Android SDK Integration Steps

***JioMoney wallet(PPI)can be integrated into your android application using 4 simple steps.***

***Step 1: Import library into your project***

***Step 2: Initialize the SDK***

***Step 3: Set necessary parameters***

***Step 4: Call payment method to process the payment.***

**Import library into your project**

1. SDK supports minimum version of android API LEVEL 10 that is version 2.3.3

2. Import .aar library into android project

You can do this in following two ways:

1. Add project dependency inside build.gradle file of app

Add .arr file into libs directory of project

Add below code to project level build.gradle

allprojects {

repositories {

jcenter()

flatDir {

dirs 'libs'

}

}

}

Add dependency to app level of build.gradle

compile(name:'jm\_sdk', ext:'aar')

Then synchronise the project .

1. Go to project structure of android studio and add file dependency in dependencies tab.

Below is the link for instructions for importing .aar file into Android Studio

<https://stackoverflow.com/questions/16682847/how-to-manually-include-external-aar-package-using-new-gradle-android-build-syst>

**Initialize the SDK**

1. Initialize the SDK with below code.

JMPaymentConfig.getInstance()  
 .setJMEnvironment(JMEnvironment.PRE\_PROD)  
 .setClientId(“**client\_id”**)  
 .setMerchantId("**merchant\_id**")  
 .setReturnUrl("**return url**")  
 .setVersion(**JMPaymentConfig.JMVersion.VERSION2\_0**)  
 .setCurrency(**JMPaymentConfig.JMCurrency.INR**)  
 .setEnableLog(**true**)  
 .setAutoSubmitOTP(**true**)  
 .init(**this**);

**Set necessary parameters**

1. Create the instance of JMPayment model with basic values(mandatory)

JMPayment payment = new JMPayment(“**extRefNo**",  
 “**timestamp**”,  
 **amount**,  
 **mobile**,  
 “**checksumHash**”);

**Call payment method to process the payment.**

1. Call payment method to process the transaction

JMPaymentService.getInstance().makePayment(context, JMPayment, new JMPaymentTransactionCallback() {  
 @Override  
 public void onResponse(final JMPaymentResponse JMPaymentResponse, String rawResponse) {  
 Log.d(TAG, "response: " + JMPaymentResponse.toString());  
   
 }  
  
 @Override  
 public void onError(int errorCode, final String error) {  
   
 }  
});

**Parameter Details:**

**client\_id**:- This is provided by Integration team for integration and testing in Sandbox environment.

**merchant\_id**:- This is provided by Integration team for integration and testing in Sandbox environment.

**return\_url**:- This is the parameter where you will get the response of the transaction after completion. This URL must be hosted at merchant’s end.

**version**: - 2.0 needs to be passed as this SDK supports version 2.0

**currency**: - INR should be passed, as the supported currency is INR(as of now)

**setEnableLog**: - **T**his method is used to see the logs, in the prod environment.The value must be set to false.

**extRefNo**: - Thisis a unique ref number generated by merchant which will be used as reference in-case of customer queries.

**timestamp**: - This is used to maintain merchant transaction time at JioMoney side

**amount**: - This is actual parameter of type double for payment

**mobile**: - Thisparameter is the valid mobile number which will be displayed in the payment screen before login

**checksum\_hash**:- This is the value generated at server side to make transaction temper free.

The checksum logic is shared inside integration document.

**makePayment**: - Method of JMpaymentService class is used to initiate transaction which include below parameters

**context**: - as first parameter of type android context

**JMPayment**: - This must be initialized with basic parameters before calling make payment

**JMPaymentTransactionCallback**: - contains callback method for success and failure case.

Note: the parameters mentioned in the request must be same while generating checksum hash

1. Checksum generation logic for version 2.0
2. If you are setting product description and UDF’s in JMPayment instance use below checksum generation logic

Clientid|Amount|Extref|Channel|MerchantId|Token|ReturnUrl|TxnTimeStamp|TxnType|subscriber.mobilenumber|productdescription|UDF1|UDF2|UDF3|UDF4|UDF5

1. If you are not setting product description and UDF’s in JMPayment instance use below checksum generation logic

Clientid|Amount|Extref|Channel|MerchantId|Token|ReturnUrl|TxnTimeStamp|TxnType|subscriber.mobilenumber

1. If you want to add more parameters mentioned in Integration doc, there is a provision to set those values using setter method of JMPayment class.

Web Utilities

Technology required PHP and Web Server (apache)

1. Utility to generate checksum

<?php

/\*\*

\* Class : Checksum\_Generator

\* Description : To be hosted by Merchant for checksum generation.

\*

\*/

class Checksum\_Generator{

var $data;

//Constructor

function \_\_construct($request){

$this->data = $request;

$this->validateRequest();

}

/\*\*

\* Name : validateRequest

\* Description : Validate request

\* return : JSON if error occurred.

\*/

private function validateRequest(){

//validate json and convert into array

$arrRequest = json\_decode($this->data, TRUE);

//Check for valid JSON

if(!$arrRequest == NULL){

//Check for mandatory fields in JSON request

$mendatory\_fields = array('clientid','transaction.amount','transaction.extref','channel','merchantid','token','returl','transaction.timestamp','subscriber.mobilenumber');

//Data keys

$post\_fields = array\_keys($arrRequest);

//Check For Mandatory fields

$result = array\_diff($mendatory\_fields, $post\_fields);

if(!empty($result)){

echo '{"responseCode": 402, "checksumHash": "", "message" : "Missing mandatory fields:"'.implode($result,',').'}';

exit;

}else{

$udf\_flag = FALSE;

//Check for UDF fields

foreach($arrRequest as $k=>$v){

if("udf" == substr(strtolower($k),0,3)){

$udf\_flag = TRUE;

}

}

//To generate checksum

$this->generateChecksum($arrRequest,$udf\_flag);

}

}else{

echo '{"responseCode": 502, "checksumHash": "", "message" : "Invalid JSON"}';

exit;

}

}

/\*\*

\* Name : genrateChecksum

\* Description : purchase checksum generation logic

\* return : JSON

\*/

private function generateChecksum($data,$udf){

$clientId = 'you client id';

$merchantId = 'your merchant key';

$checksumKey = 'your checksum key';

//Check for UDF fields

if($udf){

/\* CHECKSUM GENRATION FORMAT WITH DESCRIPTION & UDFs:

\* Clientid|Amount|Extref|Channel|MerchantId|Token|ReturnUrl|TxnTimeStamp|TxnType|subscriber.mobilenumber|productdescription |UDF1|UDF2|UDF3|UDF4|UDF5

\*/

$checksum\_str = $clientId."|".$data['transaction.amount']."|".$data['transaction.extref']."|".$data['channel']."|".$merchantId."||".$data['returl']."|".$data['transaction.timestamp']."|".$data['transaction.txntype']."|".$data['subscriber.mobilenumber']."|".$data['productdescription']."|".$data['udf1']."|".$data['udf2']."|".$data['udf3']."|".$data['udf4']."|".$data['udf5'];

}else{

/\* CHECKSUM GENRATION FORMAT:

\* Clientid|Amount|Extref|Channel|MerchantId|Token|ReturnUrl|TxnTimeStamp|TxnType|subscriber.mobilenumber

\*/

$checksum\_str = $clientId."|".$data['transaction.amount']."|".$data['transaction.extref']."|".$data['channel']."|".$merchantId."||".$data['returl']."|".$data['transaction.timestamp']."|PURCHASE|".$data['subscriber.mobilenumber'];

}

$checksum = hash\_hmac('SHA256',$checksum\_str, $checksumKey);

echo '{"responseCode": 200, "checksumHash": "'.$checksum.'", "message" : "Success"}';

exit;

}

}

//To accept only POST request.

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

//$response\_data contains response String

$request\_data = file\_get\_contents('php://input');

//Creating object of class and pass data to constructor

$JioMoney\_Obj = new Checksum\_Generator($request\_data);

}

?>

1. Utility to handle response through return url

<?php

/\*\*

\* It will accept JioMoney B2B response

\* You have handle your business logic on basis of response

\* same utility can be used for different clients like web, mobile

\* Without validation

\*/

//To accept only POST request.

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

if(isset($\_POST['response'])){

//Response String

$response = $\_POST['response'];

print\_r("<script language='javascript'>

window.load = sendResponse();

function sendResponse(){

try{

console.log('".$response."');

MobileClient.onResponse('".$response."');

}catch(e){

console.log('unable to find the client');

}

}

</script>");

}

}

?>