# Instamojo SDK Integration Document

## Getting SDK:

Add the following maven repository to the build.gradle. Please make sure that you add it in the project dependency section and NOT the buildScript section.

`Maven repository HERE`

## Manifest Permissions:

The following are the minimum set of permissions required by the SDK. The SDK currently supports Android >= 14. Add the following set of permissions in the application Manifest file above the application tag.

<uses-permission android:name="android.permission.INTERNET" />  
<uses-permission android:name="android.permission.READ\_SMS" />  
<uses-permission android:name="android.permission.RECEIVE\_SMS" />  
<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  
<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

## Creating a Transaction object:

Every transaction requires the set of mandatory fields mentioned below

1. Name of the buyer
2. Email of the buyer
3. Purpose of the transaction
4. Phone number of the buyer
5. Transaction amount
6. Access Token

Access token should be generated using your Auth token on your server. These token should not be used on the application at anytime. Once the required fields are available, transaction object should be created using these fields as shown below.

Transaction transaction = **new** Transaction(name, email, phone, amount, purpose, **accessToken**);

Replace appropriate fields with their values.

## Making the Request:

Using the Transaction object, an order can be created using the following code snippet.

Request request = **new** Request(transaction, **new** MojoRequestCallBack() {  
 @Override  
 **public void** onFinish(Transaction transaction, Exception error) {

//Stop the progress dialog here  
 **if** (error != **null**) {  
 **if** (error **instanceof** Errors.ConnectionException) {  
 *//handle no internet connection error* Log.*d*(**"App"**, **"No internet connection"**);  
 } **else if** (error **instanceof** Errors.ServerException) {  
 *//handle field errors* Log.*d*(**"App"**, **"Sever error - "** + error.getMessage());  
 } **else** {  
 *//handle other errors* Log.*d*(**"App"**, error.getMessage());  
 }  
 **return**;  
 }  
  
 Intent intent = **new** Intent(getBaseContext(), FormActivity.**class**);  
 intent.putExtra(FormActivity.***TRANSACTION***, transaction);  
 startActivityForResult(intent, 9);  
 }  
});

//Start the progress dialog here  
request.execute();

## Handling errors:

There are a total of two Exceptions that have to handled while making the request to Instamojo Server.

1. ConnectionException: ConnectionException is the superset of all the Network level errors. If the mobile is not connected to internet, this exception is returned from the callback.
2. ServerException: ServerException is the super of all the errors related to fields in the Transaction Object. The message of the exception will be in the JSON format.
   1. If the access token is invalid or not provided, the following JSON response is returned.

{

"success": false,

"message": "Authentication credentials were not provided."

}

* 1. If one or more of the three fields namely buyer’s name, email, or phone in the Transaction object are invalid, the following response will be returned.

{

"buyer\_phone": [

"This is not a valid phone number."

],

"buyer\_email": [

"Enter a valid email address."

],

"buyer\_name": [

"This field is required."

]

}

NOTE: Each object in the response pertains to one invalid field. Hence, the response needs to be parsed to pinpoint which field/s are invalid.

## Receiving the Payment result:

Once the request to create the order is executed without any error, User will be shown the Debit and Netbanking forms to make the Payment. Once the User either makes or declines the payment, the order ID and payment status can be retrieved using the below mentioned code snippet.

@Override  
**protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **super**.onActivityResult(requestCode, resultCode, data);  
 **if** (requestCode == 9) {  
 **if** (resultCode == ***RESULT\_OK***) {  
 *//handle successful transaction here* String status = data.getStringExtra(PaymentActivity.***TRANSACTION\_STATUS***);  
 String id = data.getStringExtra(PaymentActivity.***ORDER\_ID***);  
 Toast.*makeText*(**this**, status + **" - "** + id, Toast.***LENGTH\_LONG***).show();  
 } **else** {  
 *//handle failed transaction here* Toast.*makeText*(**this**, **"Cancelled"**, Toast.***LENGTH\_LONG***).show();  
 }  
 }  
}

## Proguard Rules:

If your application has Proguard enabled. Please add the following lines to application’s proguard configuration.

*# Parcelable rules*

**-keep** class \* implements android.os.Parcelable {  
 **public static final android.os.Parcelable$Creator \*;**}

*# Javascript interface rules*

**-keepclassmembers** class \* {  
 **@android.webkit.JavascriptInterface <methods>;**}  
**-keepattributes** JavascriptInterface  
**-keep** public class com.instamojo.mojosdk.network.JavaScriptInterface  
**-keep** public class \* implements com.instamojo.mojosdk.network.JavaScriptInterface  
**-keepclassmembers** class com.instamojo.mojosdk.network.JavaScriptInterface{  
 **<methods>;**}

*# to make it easier for debugging*

**-keepattributes** SourceFile,LineNumberTable  
**-keepattributes** Signature

*# OhHttp rules*  
**-dontwarn** com.squareup.\*\*  
**-dontwarn** okio.\*\*

**-keep** interface com.squareup.okhttp.\*\* { **\*;** }  
**-dontwarn** com.squareup.okhttp.\*\*

*# apache http rules***-dontwarn** org.apache.http.\*\*  
**-dontwarn** android.net.http.AndroidHttpClient  
  
*# Juspay rules***-keep** class in.juspay.\*\* {**\*;**}  
**-dontwarn** in.juspay.\*\*

*# android support rules***-keep** class android.support.v4.\*\* { **\*;** }  
**-keep** class android.support.v7.\*\* { **\*;** }

**-dontwarn** android.support.\*\*