Version
1.2.0
May 2015

MOL Android SDK

Development & Integration Guides

Change Log

DOCUMENT VERSION	DESCRIPTION	DATE
1.2.0	• SDK upgrade, Added MOL Global Payment Wall Method (Pay)	April 24, 2015
1..1.3	• Added PayPal Payment Method • Added MOLPay Payment Method • Revise App Package Compilation method	March 16, 2015
1.1.1	• Rename "EasyToPay" to "Easy2Pay".	September 23, 2014

Table of Contents

1 Overview

1.1 Objective

This document serves as a guideline for Android developers to develop and integrate MOL payment in Android App.

1.2 Required programming skills and experience

Readers require basic background in software development, as well as mastered in Java and Android software development.

1.3 System Requirements

OS: Windows, Linux, Max, JRE1.6 and above
Android SDK: SDK2.3 and above
IDE： Eclipse with ADT
Suggestions: Kindly get the updated version of Android SDK online

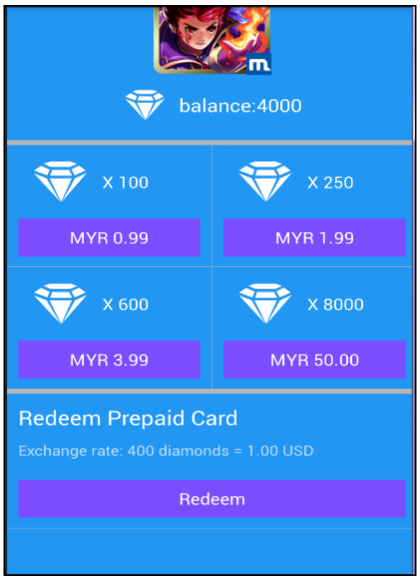
2 Introduction

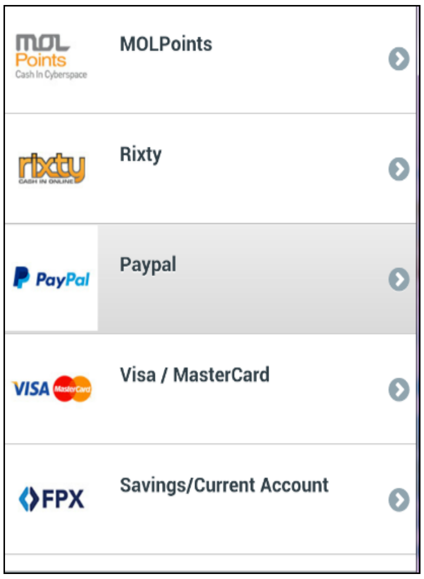
2.1 MOL Android SDK Introduction

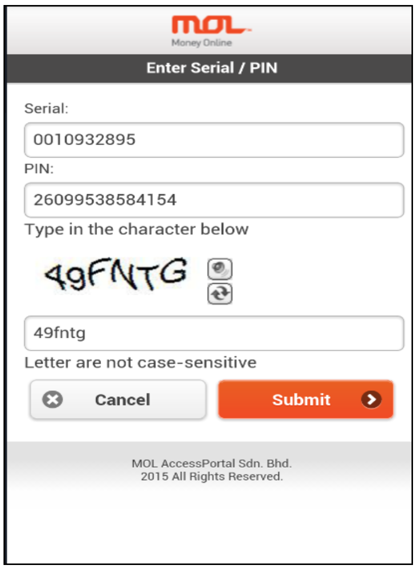
MOL Android SDK is a native SDK to facilitate your customer to make in-application payment using MOL Global payment wall. It provides easy and seamless integration for your Android application with MOL Global Payment Wall.

2.2 MOL Android SDK Purchase Flow

The SDK surface all MOL supported payment methods in payment page. Merchant only required surfacing MOL logo at their payment methods selection page in order for customer to using all supported payment methods. MOL SDK handles the displays of the payment options.






Enter Serial / PIN

Serial: 0010932895

PIN: 26099538584154

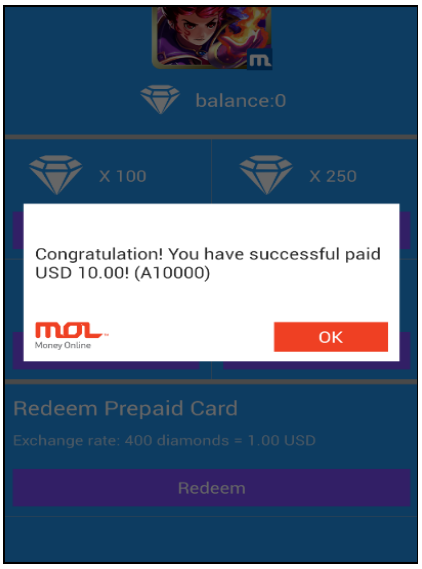
Type in the character below

49fntg

Letter are not case-sensitive

Cancel Submit

MOL AccessPortal Sdn. Bhd.
2015 All Rights Reserved.


balance:0

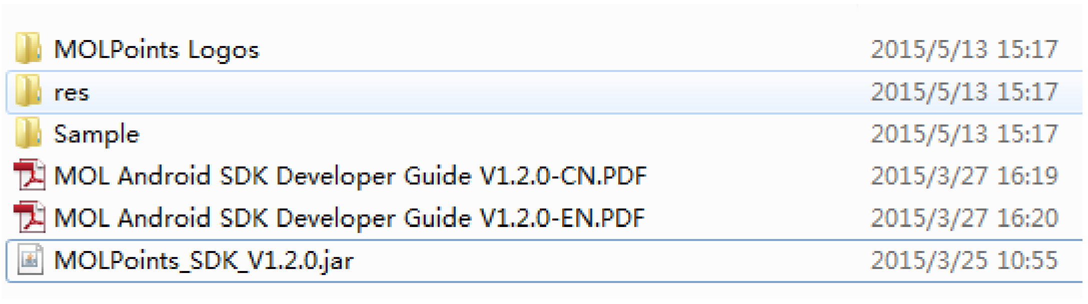
Congratulation! You have successful paid USD 10.00! (A10000)

OK

Payment Flows Diagram via MOL SDK

3 Get Started

3.1 Download SDK to local, unzip the files and directories:



MOLPoints Logos	2015/5/13 15:17
res	2015/5/13 15:17
Sample	2015/5/13 15:17
MOL Android SDK Developer Guide V1.2.0-CN.PDF	2015/3/27 16:19
MOL Android SDK Developer Guide V1.2.0-EN.PDF	2015/3/27 16:20
MOLPoints_SDK_V1.2.0.jar	2015/3/25 10:55

- **MOL_SDK_x.x.x.jar**: MOL Android SDK jar package
- **res**: resource files that are used by MOL Android SDK, it is required to copy the content of 'res' folder to the corresponding MOL Android SDK integration project folder under the same name.
- **MOLSample** : a simple integration example
- **MOL Android SDK Developer Guide**: MOL Android SDK integration guidelines and details are provided

3.2 Developer Account Application

MOL will provide a set of application account which detail as below.

	Description
Application Code	A uniquely identifying merchant application which integrating with MOL Android SDK.
Secret Key	Secret Key is a server-side security key.

4 Integration Development

4.1 Import jar files

“MOL_SDK_x.x.x.jar” need to copy into the “libs” folder. If there is no “libs” folder, please use the following way to import:

- Select your project in eclipse
- Right click to pop up menu
- Select “Built Path” -> “Add External Archives”
- Select “MOL_SDK_x.x.x.jar” in the prompted file selection window
- Click for confirmation and import is done.

4.2 Copy the Required Resource Files

Copy all the files in “res” folder (from 3.1) to the “res” folder in your project, all the MOL SDK resource files names are started with mol, which will not replace the existing resource files.

4.3 Add the Required Permissions

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="com.android.launcher.permission.READ_SETTINGS" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.SEND_SMS" />
```

*"<uses-permission android:name="android.permission.SEND_SMS" />" permission is required for Easy2Pay channel which to allow device to send SMS.

4.4 Add the Required Activity

```
<activity android:name="com.mol.payment.MOLPointsActivity"
          android:configChanges="orientation|keyboardHidden|screenSize">
</activity>
```

4.5 Call Endpoint

- MOL SDK environment setting:
  - How to use

```
MOLPayment.setTestMode(true);
```

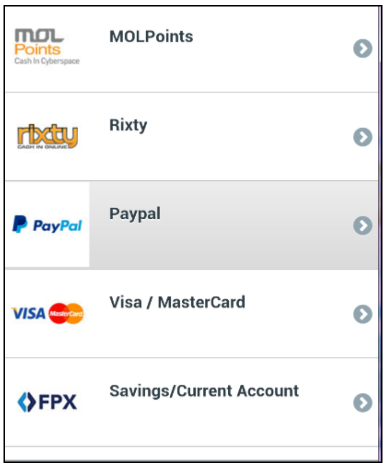
  - Description: Firstly, MOL SDK run mode is required to be set during integration testing, there are two MOL SDK run modes: test mode (true) and live mode (false). Live mode (false) will be set by default.

- MOLPayment Payment Object Creation:
  - Example

```
MOLPayment molPayment = new MOLPayment(this, Secret_Key, Application_Code);
```

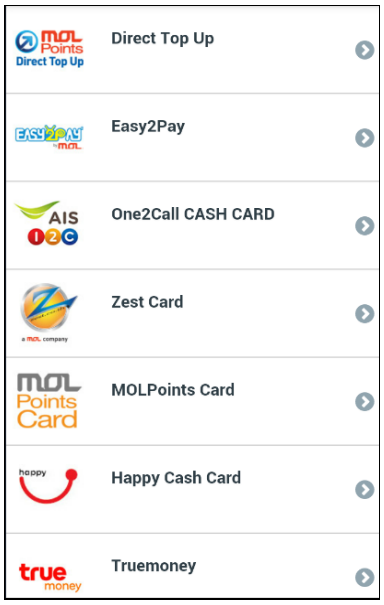
  - Description: Context target, SecretKey and ApplicationCode in Developer Account Application are required to pass in

- **Pay Payment**
  - Method Name： pay
  - Description : There is 2 type of payment channels provided which is Flexi denomination channels and Fixed denomination channels.
    - Flexi denomination channels (E-Wallet or online banking)
      Example #1 : your product is 5000 Diamonds = MYR 39.99
      You need to request payment with parameters **currencyCode=MYR** and **amount=3999**.
      Then users can choose channels like MOLPoints, PayPal, Rixty and others to proceed with payments. Please see below screenshot for details.



- Fixed denomination channels (Prepaid Card)
  Example #1 :
  {**virtualCurrencyName**} passed from merchant is “Diamonds”,
  {**virtualCurrencyRate**} is 5.5 Diamonds = 1 USD,
  MOL Forex Rate for SGD to USD is 1 SGD = 0.7416 USD
  {amount} = 5 Singapore Dollars or 500 in fractional units
  {amount} converted to USD (Based on MOL Forex Rate)
  = SGD 5 X 0.7416USD = 3.7080 USD
  Then, the value of {**virtualCurrencyAmount**} returned from MOL during Payout Payment Result is 3.7080 USD X 5.5 Diamonds = 20.40

  * This parameter is NOT for reconciliation purposes, but for ease of merchant calculation references only.



- Example:
  - Flexi denomination channels, Bundle must include **currencyCode & amount** parameters

```
Bundle inputBundle = new Bundle();
inputBundle.putString(MOLConst.B_Key_ReferenceId, "TRX1708901");      //    Must
inputBundle.putLong(MOLConst.B_Key_Amount, 3999);
inputBundle.putString(MOLConst.B_Key_CurrencyCode, "MYR");
inputBundle.putString(MOLConst.B_Key_Description, "5000 diamonds");    //Optional
inputBundle.putString(MOLConst.B_Key_CustomerId, "12321144221");      //Must
try {
        molPayment.pay(this, inputBundle, new PaymentListener() {
                @Override
                public void onBack(int action, Bundle outputBundle) {
                        // TODO Auto-generated method stub
                        showInfo(outputBundle.toString());
                }
```

```
    });
} catch (Exception e) {
    showInfo(e.getMessage());
}
```

- Fixed denomination channels, Bundle must include **virtualCurrencyName & virtualCurrencyRate** parameters

```
Bundle inputBundle = new Bundle();
inputBundle.putString(MOLConst.B_Key_ReferenceId, "TRX1708901");      //    Must
inputBundle.putString(MOLConst. B_Key_VirtualCurrencyName, "Diamond");
inputBundle.putFloat(MOLConst.B_Key_VirtualCurrencyRate,300f);
inputBundle.putString(MOLConst.B_Key_Description, "Product A");            //Optional
inputBundle.putString(MOLConst.B_Key_CustomerId, "12321144221");      //Must
try {
    molPayment.Pay(this, inputBundle, new PaymentListener() {
        @Override
        public void onBack(int action, Bundle outputBundle) {
            // TODO Auto-generated method stub
            showInfo(outputBundle.toString());
        }
    });
} catch (Exception e) {
    showInfo(e.getMessage());
}
```

- Required parameters: pay method
  - Context target
  - Bundle target: pass in related payment info
  - PaymentListener callback target: to receive payment result purpose
- When the parameter(s) passed in is invalid (for example blank Context, missing ReferenceId in Bundle, etc.) abnormal exception will be thrown, error message can be obtained via e.getMessage().
- More: Specific content in Bundle, Exception list and introduction of PaymentListener

- **Query inquiry:**
  - Method Name : paymentQuery
  - Example:

```
Bundle inputBundle = new Bundle();
```

```
//ReferenceId or PaymentId is required
inputBundle.putString(MOLConst.B_Key_ReferenceId, "TRX1708901");
// inputBundle.putString(MOLConst.B_Key_PaymentId, "MPO101913");
try {
        molPayment.paymentQuery(this, inputBundle, new PaymentListener() {
                @Override
                public void onBack(int action, Bundle outputBundle) {
                        // TODO Auto-generated method stub
                        showInfo(outputBundle.toString());
                }
        },false);
} catch (Exception e) {
        showInfo(e.getMessage());
}
```

- Required parameters
  - Context target
  - Bundle target: pass in inquiry with related info (ReferenceId or PaymentId)
  - PaymentListener callback target: to receive payment result purpose
  - Background inquiry Boolean value: true for background inquiry, false for LoadingDialog
- When the parameter(s) passed in is invalid (for example blank Context, missing ReferenceId in Bundle, etc.) abnormal exception will be thrown, error message can be obtained via e.getMessage().
- More: Specific content in Bundle, Exception list and introduction of PaymentListener

4.6 PaymentListener Callback Method Introduction

- Description: use onBack() function which in PaymentListener module to perform callback result after SDK payment or query process is complete.
- Parameters
  - action: indicate for types of operation (currently there are 2 types: MOLConst.Action_Pay, MOLConst.Action_Query, meaning for pay and query operation respectively), mainly to support one PaymentListener target in multiple endpoints.
  - resultdata: for operation result, please refer to Content List in Bundle Target and Introduction of Key and Value in Bundle

4.7 Specified Screen Orientation

To ensure the payment process will not be interrupted by changes of screen orientation, it is required to implement one of the following in calling the payment endpoint Activity:

- **<Recommended> Specify the Activity screen orientation in AndroidManifest folder, for example:**

```
<activity android:name=".PinPayActivity" android:screenOrientation="landscape"></activity>
```

Or

```
<activity android:name=".PinPayActivity" android:screenOrientation="portrait"></activity>
```

- Handle the change of screen orientation in Activity, for example:
  - Specify android:configChanges in AndroidManifest folder

```
<activity android:name=".PinPayActivity" android:configChanges="orientation|screenSize"></activity>
```

  - Overload method for onConfigurationChanged in Activity

```
@Override
public void onConfigurationChanged(Configuration newConfig) {
   super.onConfigurationChanged(newConfig);
}
```

4.8 Content List in Bundle Target for Payment

Method	Bundle	Value	Required
Pay	*Input Bundle*	*referenceId*	*Must*
		amount	*Optional(must when currencyCode exist)*
		currencyCode	*Optional(must when amount exist)*
		customerId	*Must*
		description	*Optional*
		virtualCurrencyName	*Optional(must when virtualCurrencyRate exist)*
		virtualCurrencyRate	*Optional(must when virtualCurrencyName exist)*
	Output Bundle	*result*	*Must*
		resultInfo	*Must*
		referenceId	*Must*
		paymentId	*Must(result==MOLConst.Result_Success)*
		currencyCode	*Optional (result==MOLConst.Result_Success)*
		amount	*Optional (result==MOLConst.Result_Success)*
		paymentStatusDate	*Must(result==MOLConst.Result_Success)*
		virtualCurrencyAmount	*Optional*
		customerId	*Optional*
paymentQuery	Input Bundle	referenceId	Must(referenceId or paymentId)
		paymentId	
	Output Bundle	*result*	*Must*
		resultInfo	*Must*
		referenceId	*Must*
		paymentId	*Must(result==MOLConst.Result_Success)*
		currencyCode	*Must(result==MOLConst.Result_Success)*
		amount	*Must(result==MOLConst.Result_Success)*
		paymentStatusDate	*Must(result==MOLConst.Result_Success)*
		virtualCurrencyAmount	*Optional*
		customerId	*Optional*

4.9 Introduction of Key and Value in Bundle

referenceId

Reference Id is a unique identifier generated by merchant for each distinct transaction.

Attribute	Description
Data Type	String
Max Length	50 Characters
Bundle Key	*MOLConst.B_Key_ReferenceId*

currencyCode

Currency Code refers to three characters global currencies code as refer to ISO 4217.

Attribute	Description
Data Type	String
Max Length	3 Characters
Bundle Key	*MOLConst.B_Key_CurrencyCode*

customerId

Customer Id is a unique identifier of customer generated by the merchant.

Attribute	Description
Data Type	String
Max Length	50 Characters
Bundle Key	*MOLConst.B_Key_CustomerId*

amount

Amount refers as payment/redemption amount of the transaction in fractional unit (lowest common denominator) of the respective currency code. Thousand comma separator should be removed before assign value to this parameter.

Attribute	Description
Data Type	Integer
Max Length	20 Numbers
Bundle Key	*MOLConst.B_Key_Amount*

Currency Code	Decimal Places	Example	Amount in Fractional Unit
USD	2	USD 1.00	100
MYR	2	MYR 1.00	100
AUD	2	AUD 1.00	100
BRL	2	BRL 1.00	100
IDR	2	IDR 1.00	100
INR	2	INR 1.00	100
NZD	2	NZD 1.00	100
PHP	2	PHP 1.00	100
SGD	2	SGD 1.00	100
THB	2	THB 1.00	100
TWD	2	TWD 1.00	100
VND	2	VND 1.00	100
TRY	2	TRY 1.00	100
EUR	2	EUR 1.00	100

description

Payment description refers to statement that describes the payment. The statement will be displayed in UI of the several payment providers.

Attribute	Description
Data Type	String
Max Length	50 Characters
Bundle Key	*MOLConst.B_Key_Description*

result

Payment or query result status, please refer to 2.5 payment result status list.

Attribute	Description
Data Type	String
Bundle Key	*MOLConst.B_Key_Result*

resultInfo

Payment or query result message, please refer to 2.5 payment result status list.

Attribute	Description
Data Type	String
Bundle Key	*MOLConst.B_Key_Result_Info*

paymentId

Payment Id is a unique identifier given by MOL Payout for transaction references purpose.

Attribute	Description
Data Type	String
Max Length	50 Characters
Bundle Key	*MOLConst.B_Key_PaymentId*

paymentStatusDate

Payment Status Date indicates the last updated date of the payment's status. The date will be in UTC (Coordinated Universal Time) format.

Attribute	Description
Data Type	String
Format	yyyy-MM-ddTHH:mm:ssZ
Bundle Key	*MOLConst.B_Key_PaymentStatusDate*

VirtualCurrencyName

VirtualCurrencyName indicates naming of virtual currency in game. Example 400 Diamonds = USD 1, then VirtualCurrencyName = Diamonds

Attribute	Description
Data Type	String
Bundle Key	*MOLConst.B_Key_VirtualCurrencyName*

VirtualCurrencyRate

VirtualCurrencyRate indicates exchange rate of virtualcurrency to USD 1. Example 400 Diamonds = USD 1, then VirtualCurrencyRate = 400

Attribute	Description
Data Type	Float
Bundle Key	*MOLConst.B_Key_VirtualCurrencyRate*

VirtualCurrencyAmount

VirtualCurrencyAmount is parameters return from MOL server when users complete the payment. This values indicates total Virtual Currency Amount which users entitle.
***For merchant reference only**

Attribute	Description

Data Type	Double
Bundle Key	*MOLConst.B_Key_VirtualCurrencyAmount*

4.10 Payment Result Status List

Result	MOLConst Value	resultInfo	Remarks
A10000	**MOLConst.Result_Success**	{success info}	Payment success
A10001	**MOLConst.Result_TimeOut**	Network timeout.	Please check transaction status by invoke PaymentQuery service
A10002	**MOLConst. Result_User_CancelPayment**	User cancel the payment.	User cancel the payment
A10004	**MOLConst.Result_NetWork_Fail**	network failed	Please check transaction status by invoke PaymentQuery service
A10005	**MOLConst. Result_InComplete**	Payment has not complete or in middle of processing	Please check transaction status by invoke PaymentQuery service
A10006	**MOLConst.Result_Payment_Expired**	Payment has been failed as expired.	Payment has been failed
A10007	**MOLConst. Result_Proceed_Fail**	Payment for the given transaction failed.	Payment for the given transaction failed
A10019	**MOLConst.Result_UnSupport**	The device unSupport.	Users device is not supported, might cause by device not supported for UTF-8 encoding
A10020	**MOLConst.Result_SDK_Error**	MOL SDK error.	MOL SDK error, please retry again, If error still occur, please update MOL SDK to latest version.
40003	**MOLConst. Result_CurrencyCode_Invalid**	Invalid CurrencyCode	Invalid currency code or currency code is not supported
40004	**MOLConst. Result_Duplicate_ReferenceId**	Duplicate Reference Id.	ReferenceId is already exist, please request with new unique referenceId.
40008	**MOLConst. Result_InsufficientBalance_Invalid**	Insufficient Balance	Insufficient Balance
40101	**MOLConst. Result_ApplicationCode_Invalid**	Invalid Application Code.	Invalid Application Code.
40103	**MOLConst. Result_Signature_Invalid**	Invalid secret key.	Invalid secret key.
40106	**MOLConst. Result_NotTransact_Wallet_Invalid**	User does not has matched wallet to transact.	User does not has matched wallet to transact.
40107	**MOLConst. Result_Transaction_expired**	User does not has matched wallet to transact.	Payment has been failed

40010	**Result_Invalid_CarrierCode**	Invalid Carrier Code or not supported.	Carrier Code is not supported
40011	**Result_Invalid_AmountOrCurrencyCode**	Invalid Amount or Currency Code not supported.	Currency Code and amount is not supported. Please get the latest supported denomination list from MOL Business Team
40109	**Result_BlackList_TelNo**	MSISDN is blacklisted.	Mobile Number is blacklisted
40012	**Result_WrongFormat_TelNo**	MSISDN is giving in a wrong format	Invalid Mobile Number format
40013	**Result_Exceed_Accept_Amount**	Exceed channel accepted amount	Payment amount is exceed channel maximum amount limit.
40014	**Result_Below_Accept_Amount**	Below channel accepted amount.	Payment amount is below channel maximum amount limit.

4.11 Payment Exception Referring Table

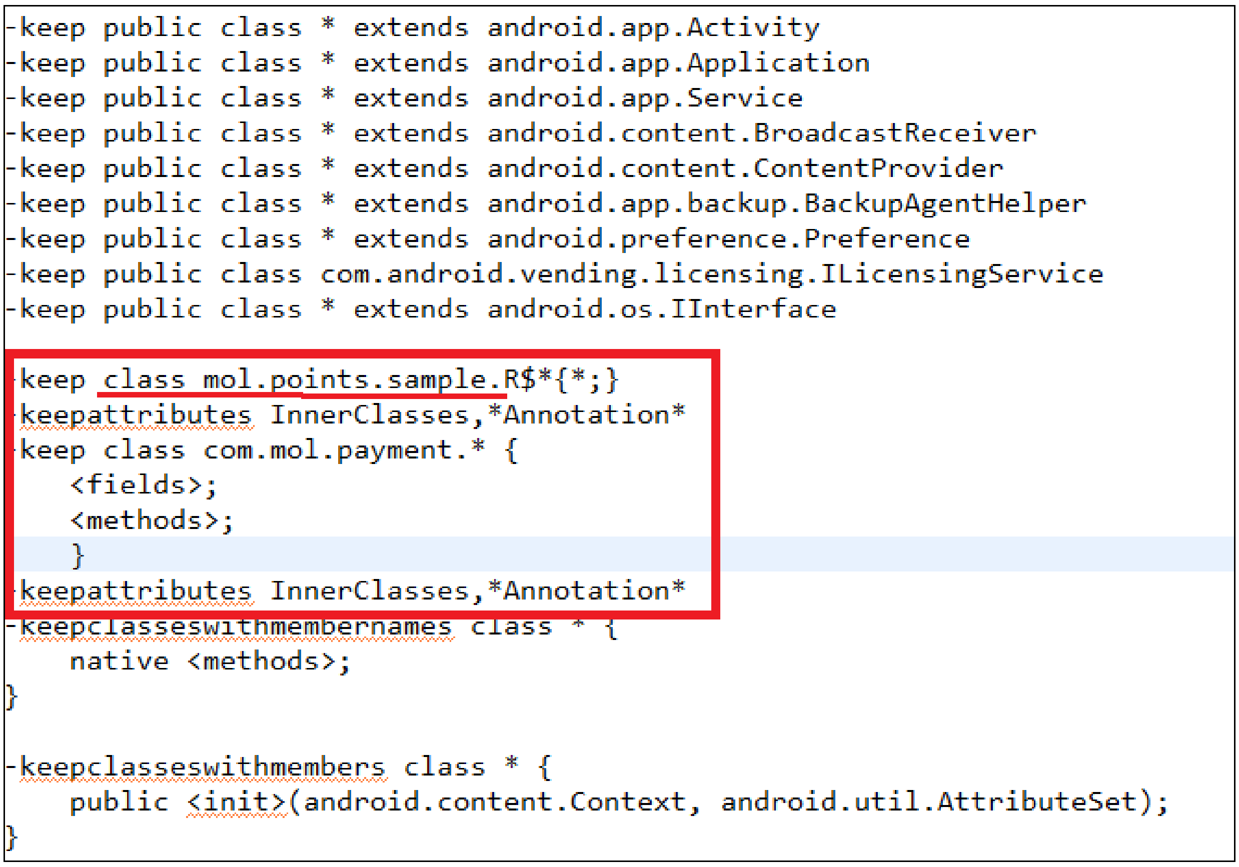
Error Type	Exception message
callback listerner is null	The paymentListener is null.
applicationCode invalid(null or len>50)	Invalid Application Code.
secretkey invalid(null or len>50)	Invalid SecretKey.
Android Context is null	The context is null.
referenceID invalid(null or len>50)	Invalid Reference Id.
customerID invalid(len>50)	The customerId's length is exceeds the maximum length(50)!
currencyCode invalid(len>3)	Invalid CurrencyCode
No SIM Card available.	No SIM Card available.

4.12 App Package Compilation

If your App release requires compile package apk, please include below code in your compile setting file proguard.cfg.

```
-keep class mol.payment.test.R$*{*;}
-keepattributes InnerClasses,*Annotation*
-keep class com.mol.payment.* {
        <fields>;
        <methods>;
        }
```

Specific method: add "-keep class mol.points.sample.R$*{*;}" in compile setting files proguard.cfg, where "mol.points.sample" is the package name of your apk. For example:



```
-keep public class * extends android.app.Activity
-keep public class * extends android.app.Application
-keep public class * extends android.app.Service
-keep public class * extends android.content.BroadcastReceiver
-keep public class * extends android.content.ContentProvider
-keep public class * extends android.app.backup.BackupAgentHelper
-keep public class * extends android.preference.Preference
-keep public class com.android.vending.licensing.ILicensingService
-keep public class * extends android.os.IInterface

keep class mol.points.sample.R$*{*;}
keepattributes InnerClasses,*Annotation*
keep class com.mol.payment.* {
   <fields>;
   <methods>;
   }
keepattributes InnerClasses,*Annotation*
-keepclasseswithmembernames class * {
   native <methods>;
}

-keepclasseswithmembers class * {
   public <init>(android.content.Context, android.util.AttributeSet);
}
```

5 Payment Host Callback (Optional)

5.1 Introduction

This service is for MOL server to notify merchant server of the payment result that has been completed by their customer. Merchant will require setup a callback URL as per application. The merchant's callback page must exist and actively listen to this service for payment status update.
NOTE: **Merchant shall able approve their customer order based on the payment status code returned from this service.**

5.2 Host Callback URL Registration

Please contact MOL to register your Payment Host Callback URL.

5.3 Payment Result

Name	Parameters
HTTP Method	**POST /**{ callback URL }
Request Parameters in HTTP Body (*x-www-form-urlencoded format*)	***Format:*** **applicationCode=**{ applicationCode }**&referenceId=**{ referenceId }**&paymentId=**{ paymentId } **&version={** version **}&amount=**{ amount }**&currencyCode=**{ currencyCode }**&paymentStatusCode=**{ paymentStatusCode }**&paymentStatusDate**={ paymentStatusDate }**&customerId**={ customerId }**&signature=**{ signature } ***Example:*** **applicationCode=**3f2504e04f8911d39a0c0305e82c3301**&referenceId=**TRX1708901**&paymentId=**MPO000000000001**&version=**v1**&amount=**1000**&currencyCode=**MYR**&paymentStatusCode=**00**&paymentStatusDate=**2012-12-31T14%3A59%3A59Z**&customerId=**12321144221**&signature=**67626c0bde4e0cf66658fa403b91bf57
Response	***Merchant server just need to response HTTP 200 to MOL server when receive callback notification.***

5.4 Generate Signature

- A Signature is a MD5 hash string combination of a sequence of parameters and a ***Secret Key***.
- Secret Key is a server-side shared secret, this key is assigned to merchant by MOL.
- All parameters use in the message exchange will form a part of the signature hash ***Except*** :
  - Empty parameter value (NOT zero)
  - Signature parameter itself.
- All parameter values that form a part of the signature hash must **sort alphabetically** based on parameter name.
- All parameters that form a part of the signature hash must in their original form (**not URL encoded**).
- All parameters that form a part of the signature hash ***ARE*** case sensitive.
- All strings will have leading and trailing whitespace stripped off.

Example

The following example explains how to generate signature for parameters with ***non-empty*** values:

Secret Key: **Ziu61T9xY227aazS530Pk8C5424y663r**

Parameter Name	Value
applicationCode	3f2504e04f8911d39a0c0305e82c3301
referenceId	TRX1708901
paymentId	MPO000000000001
version	v1
paymentStatusCode	00
paymentStatusDate	2012-12-31T14:59:59Z
amount	1000
currencyCode	MYR
customerId	12321144221

1. Sort parameter name alphabetically.

```
{ amount } + { applicationCode } + { currencyCode } + { customerId } + { paymentId } +
{ paymentStatusCode } +  { paymentStatusDate } +  { referenceId } + { version }
```

2. Concatenate/combine the actual parameter's value.

```
10003f2504e04f8911d39a0c0305e82c3301MYR12321144221TRX1708901002012-1
2-31T14:59:59ZMPO000000000001v1
```

3. Append *Secret Key* at the end of the concatenated string.

```
10003f2504e04f8911d39a0c0305e82c3301MYR12321144221TRX1708901002012-1
2-31T14:59:59ZMPO000000000001v1Ziu61T9xY227aazS530Pk8C5424y663r
```

4. Hash concatenated string using MD5 algorithm.

```
MD5(10003f2504e04f8911d39a0c0305e82c3301MYR12321144221TRX17089010020
12-12-31T14:59:59ZMPO000000000001v1Ziu61T9xY227aazS530Pk8C5424y663r) =
5e2a170eabcb54db0b2937874c39549b
```

5. Use hashed value generated from above step as Signature parameter.

```
applicationCode=3f2504e04f8911d39a0c0305e82c3301&referenceId=TRX1708901
&paymentId=MPO000000000001&version=v1&amount=1000&currencyCode=MYR
&paymentStatusCode=00&paymentStatusDate=2012-12-31T14%3A59%3A59Z&cust
omerId=12321144221&signature=c578878a380d4313d67d29bfa7632877
```

5.5 Validate Signature

All service request and response message must have a Signature parameter and will be validated by MOL to prevent data tampering. If the signature is invalid then MOL will returns Http Status 401.

It’s highly **RECOMMENDED** for merchant to perform similar validation to ensure data validity against the origin source. Repeat the same steps from 1 - 4 described in *5.2* to generate signature and compare with the signature received from MOL.

5.6 Introduction of Key and Value in Payment Host Callback

paymentId

Payment Id is a unique identifier given by MOL Payout for transaction references purpose.

Attribute	Description
Data Type	String
Max Length	50 Characters

referenceId

Reference Id is a unique identifier generated by merchant for each distinct transaction.

Attribute	Description
Data Type	String
Max Length	50 Characters

currencyCode

Currency Code refers to three characters global currencies code as refer to ISO 4217.

Attribute	Description
Data Type	String
Max Length	3 Characters

amount

Amount refers as payment/pin amount of the transaction in fractional unit (lowest common denominator) of the respective currency code. Thousand comma separators should be removed before assign value to this parameter.

Attribute	Description
Data Type	Integer
Max Length	20 Numbers

Currency Code	Decimal Places	Example	Amount in Fractional Unit
USD	2	USD 1.00	100
MYR	2	MYR 1.00	100
AUD	2	AUD 1.00	100
BRL	2	BRL 1.00	100
IDR	2	IDR 1.00	100
INR	2	INR 1.00	100
NZD	2	NZD 1.00	100
PHP	2	PHP 1.00	100
SGD	2	SGD 1.00	100
THB	2	THB 1.00	100
TWD	2	TWD 1.00	100
VND	2	VND 1.00	100

customerId

Customer Id is a unique identifier of customer generated by the merchant.

Attribute	Description
Data Type	String
Max Length	50 Characters

paymentStatusDate

Payment Status Date indicates the last updated date of the payment's status. The date will be in UTC (Coordinated Universal Time) format.

Attribute	Description
Data Type	String
Format	yyyy-MM-ddTHH:mm:ssZ

paymentStatusCode

Payment Status Code refers to two characters status indicator for Success, Failure, or Pending of a payout payment transaction.

Attribute	Description
Data Type	String
Max Length	2 Characters

List of payment status codes:

Code	Message	Description
00	Success	Payment completed and paid.
01	Incomplete	Payment has not complete or in middle of processing.
02	Expired	Payment has been failed as expired.
99	Failure	Payment for the given transaction failed.

VirtualCurrencyAmount

VirtualCurrencyAmount is parameters return from MOL server when users complete the payment. This values indicates total Virtual Currency Amount which users entitle.

***For merchant reference only**

Attribute	Description
Data Type	Decimal

6 Payment Method Logo & Display Text

Logo Files for supported payment are included in this package. Please see example below.

Payment Method	Logo	Text
Pay	MOL® Money Online	