Android Payment SDK

Integration Guide

WorldPay Payment SDK 2

How it Works 2

Payment SDK Specifications 2

Authentication Methods 2

Platform Requirements 2

Supported Hardware 2

Managed API Calls 2

Payment SDK Integration 6

Getting Started 6

Installation 6

Framework Integration 6

API Integration 7

Swiper Integration 8

References 8

# WorldPay Payment SDK

The Payment SDK makes accepting payments easy without worry of your application being in-scope for PCI-compliance.

## How it Works

Like a print driver, the WorldPay Payment SDK installs alongside your software application - adding transaction processing to your applications. The Payment SDK facilitates all transactional communication between the SecureNet payment gateway and approved hardware devices to isolate payment data and keep it separate from your software application.

Advantages of the Payment SDK

1. Easy to implement
2. Pre-configured and tested, therefore decreasing your time to market
3. Reduces PCI Compliance scope and liability for merchants
4. Provides Point to Point Encryption for all transactions
5. WorldPay maintained, so no ongoing maintenance costs

## Payment SDK Specifications

#### Authentication Methods

* Magnetic Stripe Reader (MSR)
* EMV Contact
* Manual Entry

#### Platform Requirements

* Android 4.4

#### Supported Hardware

* Miura M010

#### Managed API Calls

* Authorize
* Capture
* Capture Signature
* Refund
* Void
* Get Funding Data
* Open Batch
* Close Batch
* Credit Account
* Capture Using Secondary Account

# Payment SDK Integration

## Getting Started

Getting started with the Payment SDK is easy. Once you’ve selected your desired Platform and Hardware, the setup is similar to a direct Web Services integration.

1. Create transaction data objects in your application.
2. Payment SDK initiates commands in the terminal and gathers the card data and sends it to the SecureNet Gateway.
3. SecureNet sends a return response to the Payment SDK with details for the receipt.

## Installation

1. Download the WorldPay Payment SDK appropriate for your selected Platform.
2. Unzip the SDK and add the .AAR file to the project directory (e.g.: /<ProjectName>/).

## Framework Integration

1. Import the .AAR file into your project by clicking File->New Module->Import .AAR/.JAR file to <ProjectName>.
2. This will create a new project based on the .AAR file. Give the project a name
3. Navigate to the build.gradle file of the application that will be utilizing the project.
4. In the dependencies block, add a line: compile project(‘:<project name’) with the name of the project created off of the .AAR file. In the example below, *gatewaysdk* is the name of the project created from the .AAR file.

**Example:** apply **plugin**: **'com.android.application'***android {  
 compileSdkVersion 23  
 buildToolsVersion* ***"23.0.2"*** *defaultConfig {  
 …..**}  
 buildTypes {*

*……  
}  
}  
  
dependencies {**compile project(****':gatewaysdk'****)  
}*

1. Rebuild the project. The application now has access to the payment and swiper SDK classes.

## API Integration

1. **Create an Auth Token**. Each Task & corresponding Request object require the AuthToken, MerchantKey, MerchantId, DeveloperId, and Application Information. Before any other calls can be made an auth token must be created. **It is the responsibility of the implementer to maintain a reference of and secure the auth token.**

**Example:**

AuthTokenCreateRequest request = new AuthTokenCreateRequest();  
if (mMerchantGatewayType == MerchantGatewayType.*SECURENET*) {  
 request.setSecureNetId(BuildConfig.*GATEWAY\_ID*);  
 request.setSecureNetKey(BuildConfig.*GATEWAY\_KEY*);  
} else if (mMerchantGatewayType == MerchantGatewayType.*MERCHANT\_PARTNERS*) {  
 mMainActivity.showErrorDialog(R.string.*error\_unsupported\_gateway*);  
 return;  
} else {  
 mMainActivity.showErrorDialog(R.string.*error\_unsupported\_gateway*);  
 return;  
}  
  
request.setApplicationId(BuildConfig.*APPLICATION\_ID*);  
request.setApplicationModel(Build.*MODEL*);  
request.setApplicationOs("Android");  
request.setTerminalId("");  
request.setTerminalVendor("Miura");  
request.setDeveloperId(BuildConfig.*DEVELOPER\_ID*);

1. **Begin making calls.** All web service calls have a corresponding AsyncTask. Web service calls are made by initializing the desired Task and passing in the appropriate Request object. In order to handle the response, the implementer should override the onPostExecute method which passes the appropriate Response object back to the implementer.

**Example:**

new CreateAuthTokenTask(request) {  
 @Override  
 protected void onPostExecute(AuthTokenCreateResponse authTokenCreateResponse) {  
 super.onPostExecute(authTokenCreateResponse);  
 mMainActivity.hideProgressDialog();  
 if (authTokenCreateResponse != null) {  
 if (!authTokenCreateResponse.hasError()) {  
 if (!TextUtils.*isEmpty*(authTokenCreateResponse.getAuthToken())) {  
 WorldPaySampleApp.*getInstance*()  
 .setAuthToken(authTokenCreateResponse.getAuthToken());  
 mMainActivity.showTransactionDialogFragment();  
 } else {  
 mMainActivity.showErrorDialog(R.string.*error\_authentication\_failed*);  
 }  
 } else {  
 mMainActivity  
 .showErrorDialog(authTokenCreateResponse.getExceptionMessage());  
 }  
 } else {  
 mMainActivity.showErrorDialog(R.string.*error\_no\_network*);  
 }  
 }  
};

## Swiper Integration

1. **Import TransactionDialogFragment**. In your Activity, import the TransactionDialogFragment.

**Example:**

import com.worldpay.ui.TransactionDialogFragment;

1. **Implement TransactionDialogFragmentListener.** This interface exposes delegate methods used to communicate the transaction status to the hosting application.

Example:

public class MainActivity extends AppCompatActivity implements  
 TransactionDialogFragment.TransactionDialogFragmentListener {  
  
 @Override  
 public void onTransactionComplete(TransactionResult transactionResult,  
 PaymentResponse paymentResponse) {  
 if (transactionResult != TransactionResult.*APPROVED*) {  
 showErrorDialog("Transaction Failed:" + transactionResult.toString());  
 } else {  
 showSuccessDialog("Transaction Approved. " +  
 "\n transactionId = " + paymentResponse.getTransactionResponse().getId());  
 }  
 }  
  
 @Override  
 public void onTransactionError(  
 @NonNull TransactionDialogFragment.TransactionError transactionError,  
 @Nullable String s) {  
 showErrorDialog("Transaction Error: " + transactionError.toString() + "\n message: " + s);  
 }  
  
 @Override  
 public void onTransactionReversalFailed(ReversalRequest reversalRequest) {  
 showProgressDialog("Retrying failed reversal, please wait...");  
 PaymentRefundTask task = new PaymentRefundTask(reversalRequest) {  
 @Override  
 protected void onPostExecute(PaymentResponse paymentResponse) {  
 hideProgressDialog();  
 }  
 };  
 task.execute();  
 }  
  
}

1. **Get a new instance of the TransactionDialogFragment.**

Example:

TransactionDialogFragment dialogFragment = TransactionDialogFragment.*newInstance*();

1. **Start the swiper.** Populate the instance of TransactionDialogFragment with the required fields then show it. The dialog will show transaction status messages if necessary, and will trigger the appropriate delegate methods of TransactionDialogFragmentListener.

Example:

dialogFragment.setAuthToken(WorldPaySampleApp.*getInstance*().getAuthToken());  
dialogFragment.setCaptureMode(CaptureMode.*MANUAL*);  
dialogFragment.setDeveloperId(BuildConfig.*DEVELOPER\_ID*);  
dialogFragment.setMerchantId(BuildConfig.*GATEWAY\_ID*);  
dialogFragment.setMerchantKey(BuildConfig.*GATEWAY\_KEY*);  
dialogFragment.setTransactionType(TransactionType.*SALE*);  
dialogFragment.setSwiper(Swiper.*MIURA\_M10*);  
  
TransactionData transactionData = new TransactionData();  
transactionData.setAmount(BigDecimal.*ONE*);  
transactionData.setCashBackAmount(BigDecimal.*ZERO*);  
dialogFragment.setTransactionData(transactionData);  
  
dialogFragment.show(getSupportFragmentManager(), TransactionDialogFragment.*TAG*);

## References

Please review the Payment SDK Documentation and provided Sample Application for details and examples.