CyberSource Mobile Point of Sale SDK for Android

The CyberSource Mobile Point of Sale SDK is a semi-integrated solution that enables you to add mobile point-of-sale functionality to your payment application, including card-present and EMV capabilities. The merchant's application invokes this SDK to complete an EMV transaction. The SDK handles the EMV workflow as well as securely submitting the EMV transaction for processing. The merchant's application never touches any EMV data at any point.

Supported Types

Accepted card types:

- Visa
- MasterCard
- American Express
- Discover

Accepted currency—USD

Supported payment types:

- Authorization
- Capture
- Authorization reversal
- Void
- Refund

Supported transaction types:

- Contact chip & signature
- Magstripe read of a chip card (fallback)
- Magstripe read of regular cards

Point of Sale Flow

Step 1 Insert approved card reader into the device.

Step 2 Insert card with EMV chip into the reader.

Step 3 Select the application, if prompted. If there is only a compatible application on the card, the application is selected automatically.

Step 4 Confirm the amount.

Step 5 Do not remove the card until the transaction is complete. If at any time the user cancels the transaction, the EMV transaction is cancelled.

Step 6 If at any time the user cancels the transaction, the EMV transaction is cancelled.

Prerequisites

Before integrating the SDK into your app, you must first obtain *test* and *live* card readers, generate client credentials, register and activate the device, and generate a session token as explained below.

Card Readers

Separate card readers with separate security keys must be used for *testing* and for *live* transaction processing. Card readers can be obtained at the TASQ portal. Talk to your CyberSource account manager for more information.

Generating Client Credentials

Before integrating the SDK, you must first create a client ID and client secret that you can use to generate tokens to authenticate your application.

To generate the client credentials:

Step 1 Log in to the CyberSource Business Center and navigate to **Account Management > Client Integration Management**. The Transaction Processing page appears.

Step 2 Click the **Transaction Processing** drop-down menu and select **OAuth 2.0**.

Step 3 Click the plus **+** sign in the lower-right corner of the screen. The Create Credentials page appears. Select from the following options:

Table 1 Generate Credentials Options

Option	Description
mPOS Permissions	mPOS Device Management—permission to manage devices in the merchant interface.
	mPOS Device Access—permission to access a particular device.
	mPOS Device Terminal ID Management—permission to manage the terminal ID (TID) for a particular device.
Configuration	Client Description—Description of what this client is used for. This is helpful for auditing a merchant's various clients.
	Client Version—the client's version number. This is useful if a merchant wants different clients with different software products.
	Token Inactivity Duration—indicates how long a token can remain unused before it is invalidated.
	Access Token Validity Duration—the maximum time for which every access token generated by this client remains valid.
Mobile Permission	When set to true, the mobile clients are expected to provide a device ID in requests for an access token.
Access Grant Type	Client Credentials—enables the user to authenticate using the client secret and ID. See "Registering the Device," page 4 for more information.
	Password—enables the user to authenticate using a the username and password that they use to log into the CyberSource Business Center.

Step 4 Store the key and secret in your server and not in your application or device.

Registering the Device

Each device that is used for transactions must be registered. You should set up your application to make a device registration call the first time any user logs in to your application. Use the following fields:

Table 2 Device Registration Fields

Field	Description
client_id	Obtain the client ID in the Business Center as shown in the preceding section "Generating Client Credentials," page 2. You can submit this field if you chose the Client Credentials grant type during registration. **Important** For security reasons, you should store this data in your server and not in the application.
client_secret	Obtain the client secret in the Business Center as shown in the preceding section "Generating Client Credentials," page 2. You can submit this field if you chose the Client Credentials grant type during registration. **Important** For security reasons, you should store this data in your server and not in the application.
username	The username that is used to log in to the CyberSource Business Center. You can submit this field if you chose the Password grant type during registration.
password	The password that is used to log in to the CyberSource Business Center. You can submit this field if you chose the Password grant type during registration.
merchant_id	The merchant ID that is used to log in to the CyberSource Business Center.
terminal_id	The ID of the merchant's terminal.
device_id	The device ID must be a unique value.
description	Merchant-defined description.
device_platform	Use this field to identify the OS version. For example, Android 4.2.1.
platform	Use the value 1.

Registration Request

```
POST /apiauth/v1/oauth/device

Content-Type: application/x-www-form-urlencoded

client_id=DigkpILFw7&client_
secret=a8d24186173d1dfc6a37c9b7f9a1daa8&merchant_id=your_merchant_
id&device_id=123456&description=Bob&terminal_id=abc&device_
platform=Android%204.1&platform=1
```

Registration Response

```
201 Created
Server:  Apache-Coyote/1.1
Content-Type:  application/json
Content-Length:  48
Date:  Tue, 15 Mar 2016 01:22:57 GMT

{"device_id":"123456","status":"pending"}
```

This device is registered in a "pending" state, and you must log in to the Business Center to activate it before any transactions can be processed.

Activating the Device

Before it can be used, the device must be activated.

To activate the device:

Step 1 Log into the CyberSource Business Center.

Step 2 Navigate to **Account Management > Device Management**.

Step 3 Find the device ID of the device you would like to activate in the Device ID column.

Step 4 In the Status column, click the status for this device and select **Active**.

Generating a Token

Authentication is performed by obtaining a token.

To obtain a token:

Step 1 Log in to the CyberSource Business Center and navigate to **Account Management > Transaction Security Keys**.

Step 2 Click **Generate Client ID**.

Step 3 Click **Generate Secret**.

Step 4 Pass the client ID, device ID, and the merchant credentials that you use to access the Business Center in a token request.

Below is an example of a token request and response.

Token Request

```
POST /apiauthservice/oauth/token

Content-Type: application/x-www-form-urlencoded

_type=client_credentials&client_id=3ck9WMz0Zh&client_
secret=6450bc7f1b4000dc88c6b6c8c2546018&merchant_id=mpos_
sdk&=1&device_id=mpos_sdk_device_id10
```

Token Response

```
{"access_token":"0e4f71a1-a67d-41c0-b2f1-373cb1168735","token_
type":"bearer","expires_in":28799,"scope":"PaymentCreditPermission
PaymentDebitPermission PaymentVerificationPermission","client_
status":"active"}
```

Integrating the SDK

The merchant's Android application is assumed to have the following credentials from CyberSource before the SDK can be integrated.

- Merchant ID
- Device ID
- Session Token

```
String  merchantId = "REPLACE_WITH_YOUR_MERCHANT_ID"

String  deviceId = "REPLACE_WITH_YOUR_DEVICE_ID”

String  sessionToken = "REPLACE_WITH_YOUR_SESSION_TOKEN"
```

To integrate the SDK:

Step 1 Create a new Android project with one module for the reference app and a second module for the SDK by importing ***cybersource-mpos-android-sdk-debug.aar***. Add this SDK module as a dependency to the merchant's Android application.

Step 2 Using the Manager class, initialize the SDK with either the PROD or TEST environment, device ID, terminal ID, terminal ID alternate and merchant ID (MID) value.

```
Settings settings = new Settings(Settings.Environment.ENV_
TEST,”deviceID”,”terminalID”,”terminalIDAlternate”,”mid”);
```

```
Manger manager = new Manager(settings);
```

Method Definition:

```
public Settings(Environment environment, String deviceID, String
terminalID, String terminalIDAlternate, String mid)

public Manager(Settings settings)
```

Environment – enum with values ENV_TEST for CAS and ENV_LIVE for production environment.

Step 3 Prepare the `PaymentRequest` object:

```
paymentReqObject = new PaymentRequest();

paymentReqObject.setMerchantId(merchantID);

paymentReqObject.setAccessToken(oAuthToken);

PurchaseTotal purchaseTotal = new PurchaseTotal();

purchaseTotal.setCurrency("USD");

purchaseTotal.setGrandTotalAmount(amount);

paymentReqObject.setPurchaseTotal(purchaseTotal);

paymentReqObject.setMerchantReferenceCode(Long.toString(System.currentTim
eMillis()));
```

Step 4 Create a manager delegate to receive callbacks after the transaction is complete.

```
private final ManagerDelegate managerDelegate = new ManagerDelegate() {

 @Override

  public void performPaymentDidFinish(PaymentResponse paymentResponse,
PaymentError paymentError) {

 // Perform actions

 }

};
```

Step 5 Start the payment transaction.

```
manager.performPayment(paymentReqObject, this, managerDelegate);
```

Method definition: `public void performPayment(PaymentRequest paymentRequest, Context context, final ManagerDelegate managerDelegate)`

Permissions

Android 6.0 (API level 23) and later versions enable users to grant permissions to apps while the app is running. The SDK requires the permissions shown below from the merchant's app:

- RECORD_AUDIO
- MODIFY_AUDIO_SETTINGS
- WRITE_EXTERNAL_STORAGE
- READ_EXTERNAL_STORAGE

Example Setting Permissions

```
if (ContextCompat.checkSelfPermission(BaseActivity.this,
        Manifest.permission.RECORD_AUDIO)
        != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(BaseActivity.this,
                new String[]{Manifest.permission.RECORD_AUDIO,
Manifest.permission.MODIFY_AUDIO_SETTINGS},
                MY_PERMISSIONS_REQUEST_AUDIO);
}
```

UI Settings

The merchant's app has enough access to change few properties of SDK pages like Company Logo, Font Color, Font Type and others by setting values to **UISettings** object.

Example UI Settings

```
UISettings uiSettings = UISettings.getInstance();

uiSettings.setBackgroundColor("#FF0000");

uiSettings.setSpinnerColor("#800000");

uiSettings.setTextLabelColor("#FFFF00");

uiSettings.setTitleImageURL("image url");

uiSettings.setFontFamily(Typeface.SERIF);
```

Android Compatibility:

The SDK requires minimum Android SDK version 15 and its targeted version is 23.

Example Android Compatibility

```
android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"
    defaultConfig {
        minSdkVersion 15
        targetSdkVersion 23
    }

}
```