Android SDK

Integration Guide

Version 1.3.130

last update: **2017-06-12**

Document history

Date	Name	Change
10.01.2017	Paul Sprotte	First Version
05.05.2017	Armando Shkurti	Version 1.2.120
12.06.2017	Armando Shkurti	Version 1.3.130

Introduction

1. Overview

This is the documentation for the Android SDK of Computop to integrate payments in your Android app.
Here you could see an overview of steps to make your app runnable with computop's SDK (more details below):

A. configure your merchant account on the computop paygate system
B. make your android project runnable with gradle and add the SDK to your project
C. configure the SDK with the data retrieved from Computop
D. configure which payment methods you want to use
E. starts the authentication against your merchant backend
F. hand over the payment data
G. start the check out
H. check for errors

1.1 List of supported payment methods

The Android SDK currently supports the following payment methods:

- Credit Cards
- SEPA Direct Debit
- PayPal

2. Requirements

To use the SDK you need to ensure to fulfill the following things:

Preparation
- existing merchant account at Computop
- merchant ID which you will receive from from computop after creation of the merchant account

Development
- using gradle as dependency management in your android project

- Android min SDK Version is 15 (4.0.3)

3. Installation

Add jCenter in the main build.gradle:

```
allprojects {
    repositories {
        jcenter()
    }
}
```

Then add the following line to your app build.gradle:

```
compile 'com.computop.sdk:lib:<current_version>'
```

and sync gradle:

4. How to implement

You could find an demo on https://github.com/ComputopPayment/Computop-Android-Example

Use this to see how it works or use the following step by step introduction.

4.1 Configuration

Configure the SDK by importing the Computop class and inserting the configuration parameters you received from Computop, like *merchantID, authURL*.

The provide merchantID in a strings.xml

```
<resources>
    <string name="computop_merchant_id">XXX</string>
    <string name="computop_auth_url">XXX</string>
```

```
</resources>
```

4.2 Authentification

The only requirement for the Mobile SDK is to insert the respective Merchant's URL in order to be able to receive the auth token. The SDK is responsible to retrieve then the token under the hood and use it appropriately when executing payment requests.

```
<resources>
    <string name="computop_merchant_id">XXX</string>
    <string name="computop_auth_url">XXX</string>
</resources>
```

4.3 Making a payment

To make a payment of one to many products you want to sell in your app you first you need to get the list of the available payment methods and then you need to configure the payment data.

First initialize an instance of the SDK, activity should be instance of FragmentActivity.

```
Computop computop = Computop.with(getActivity())
```

4.3.1 Available Payment Methods

To get all the available payment methods you need to call this:

```
computop.requestPaymentMethods()
      .subscribeOn(Schedulers.io())
      .observeOn(AndroidSchedulers.mainThread())
      .subscribe((paymentMethods, throwable) -> {
         if (throwable != null) {
            //manage error
             Log.e(TAG, "PaymentMethods error: ", throwable);
             return;
```

```
        }

        //manage the paymentMethods list (List<PaymentMethod>)
    });
```

For a payment method you get the icon and the localized description.

4.3.2 Payment Data

The next step is to select a payment method and then fill payment data with the proper key and values based on the documentation for your payment method (more details on `Payment` in Paygate documentation).

```
Payment getPayment(@NonNull PaymentMethod paymentMethod) {

   Payment payment  = paymentMethod.getPayment();

   payment.setParamWithKey("TransID", "********");
   payment.setParamWithKey("Amount", "100");
   payment.setParamWithKey("Currency", "EUR");
   payment.setParamWithKey("URLSuccess", "http://*****/api/success");
   payment.setParamWithKey("URLNotify", "https://*****/computop/index.php");
   payment.setParamWithKey("URLFailure", "http://*****/api/failure");
   payment.setParamWithKey("RefNr", "******");
   payment.setParamWithKey("OrderDesc", "Tests");
   payment.setParamWithKey("AddrCity", "Berlin");
   payment.setParamWithKey("FirstName", "Lorem");
   payment.setParamWithKey("LastName", "Ipsum");
   payment.setParamWithKey("AddrZip", "10000");
   payment.setParamWithKey("AddrStreet", "Berlin");
   payment.setParamWithKey("AddrState", "AL");
   payment.setParamWithKey("AddrCountryCode", "DE");
   payment.setParamWithKey("Phone", "01777777124");
   payment.setParamWithKey("LandingPage", "Login");
   payment.setParamWithKey("eMail", "******-accounts@****.com");
   payment.setParamWithKey("ShopID", "1");
   payment.setParamWithKey("Subject", "******-accounts@****.com");

   return payment;
}
```

The method *setParamWithKey()* will return **true** if the value will be set successfully for the specified key.

The values of `Amount` and `Currency` will be validated during the checkout process. So you have to ensure that you are use valid data.
The currency is the currency you want to use you for the payment. You have to give up to three character (DIN/ISO 4217), e.g. “EUR”.
The amount is the lowest unit of the currency you are using. That means if you are use EUR as currency the amount needs to be cents, e.g. an amount of 100 is 1 EUR.

For the rest of the params you should take a look into the paygate documentation. Some of them must be defined by you and are mandatory.

4.3.3 Make the checkout

Now that you have configured the SDK, selected the payment method and filled the values for your payment data, you are ready to make a payment.

For the Checkout we are using Retrofit underneath.
Start the checkout by instantiating subscription object (Observable). The `Computop` class is a top-level class that facilitates the payment procedure. It is responsible for validating payment data and instantiating a `WebView` when a new payment is triggered by passing the respective `payment` and `paymentMethod`.

```
Disposable disposable = computop
      .withPaymentMethod(method)
      .setWebViewListener(() -> {
         Log.i(TAG, "[WebViewClient] onPageFinishedLoading");
      })
      .checkout()
      .subscribeOn(Schedulers.io())
      .observeOn(AndroidSchedulers.mainThread())
      .subscribe(o ->
      {
         Log.v(TAG, "Payment received");
      }, throwable -> {

         if (throwable instanceof ComputopError) {
            Log.e(TAG, "Computop Error " + (throwable));
            showErrorAlert((ComputopError) throwable);
         } else if (throwable instanceof InternalError) {
            Log.e(TAG, "Internal Error boolean isPaymentCanceled(): " + ((InternalError)
throwable).isPaymentCanceled());
         } else {
```

```
            Log.e(TAG, "Payment error: ", throwable);
        }
    });
```

`Observable<T> subscribeOn(Schedulers.io())` will execute the operation on the background thread. When you subscribe to this observable you can show a loading indicator to the user. To get notified when the Webview is loaded you can set the listener `OnWebViewLoadedListener` and hide the loading indicator.

In case of a successful transaction you will get notified in onNext Consumer.

In case of an error you will receive the response in onError Consumer. You have to handle all types of errors including network errors during the checkout process and give feedback to the user.

4.4 Support screen rotation

If the activity where you are doing the payment using Computop SDK (only for payment methods that use WebView) needs to support screen rotation than you have to handle the configuration change.
To do this you have to add this flag in your AndroidManifest.xml file:

```
<activity android:name=".MyActivity"
          android:configChanges="orientation|screenSize"
          android:label="@string/app_name">
```

In this case you will have to handle the configuration changes yourself for your activity. For more info please check:

- https://developer.android.com/guide/topics/resources/runtime-changes.html#HandlingTheChange
- https://developer.android.com/reference/android/app/Activity.html#ConfigurationChanges