MultiSafepay
The Payment Professionals

Fastcheckout SDK

Manual v1.7

Document Control		
Version	**Date**	**Revision**
1.0	11/04/2018	Initial version
1.1	02/05/2018	Document reviewed and updated
1.2	03/05/2018	Add React Native
1.3	12/06/2018	Update React Native documentation
1.4	12/12/2019	Update minimum requirements
1.5	19/02/2019	Add link to Multisafepay API Documentation
1.6	03/04/2019	Add implementation interface call for Android
1.7	16/05/2019	Update iOS minimum requirements

MultiSafepay
The Payment Professionals

Table of contents

Introduction

Fastcheckout SDK

The Fastcheckout SDK is a library that you can integrate into your native Android, iOS and/or React Native application providing connection to MultiSafepay services. The Fastcheckout SDK adds a smooth, fast and native checkout experience to your mobile app, and by storing and reusing data it is the fastest checkout process available for shopping apps.

As an integrator, you only need to provide a valid MSP transaction identifier or create a transaction. The Fastcheckout SDK leverages the checkout process for you, providing notifications for all possible outcomes (success, pending, canceled, etc.) once it completes a transaction. The checkout process flow includes:

- Shipping details:
  - Preferred shipping details
  - Add shipping details
- Payment methods:
  - Preferred payment methods
  - Add payment methods
  - Coupons
- Confirmation details

Additional details

The Fastcheckout SDK provides the additional following features:

- Authentication into Fastcheckout:
  - It provides secure sign-up/login functionality
- Check orders:
  - Open seamless support tickets to related orders
- Support orders:
  - Easily open a support ticket related to an order
- Edit payment information:
  - List and edit user's stored payment details
- Manage shipping addresses:
  - List, add and edit shipping information

*If you need more information, please check the documentation that you will find inside the **SDK**, which contains all classes, methods, common troubleshooting and more details.*

Please refer to the **Multisafepay API Documentation** to perform different calls through which you can start transactions, update transactions, perform refunds and receive information about transactions: https://docs.multisafepay.com/api/#orders

Requirements

iOS

- Xcode 10.2 and iOS 12 SDK
- iOS 9+ target
- Swift 5.0 or Objective-C

Android

- Android Studio 3 and up
- Android version 4.4 and up
- Java or Kotlin

React Native

- React Native 0.54 or later is required
- For iOS, the default way to use the SDK requires ***CocoaPods***. (If you haven't installed CocoaPods, please follow the CocoaPods Getting Started to do so).

Installation

First you have to access into your Back Office and get a valid API Key (Settings ➔ Website Settings)

Once you have your valid API Key, simply download the Android and/or iOS SDK version.

iOS

Add the **FastcheckoutKit.framework** as an *embedded framework* into your project. The next steps will guide you through the process:

1. Open the folder in Finder where you have downloaded FastcheckoutKit.framework and drag it into the Project Navigator of your application's Xcode project.

2. Make sure you select **Copy items if needed** unless you have already copied the framework into your project folder.
3. Select your application project in the Project Navigator (blue project icon) to navigate to target configuration window and select the application target under the Targets heading in the sidebar.
4. In the tab bar at the top of that window, open the General panel.
5. Click on the + button under the Embedded Binaries section.
6. Select FastcheckoutKit.framework

You can now start using the FastcheckoutKit SDK in your App.

Android

Add the **msp-android-sdk-release.aar** as an *embedded framework* into your project. The next steps will guide you through the process:

1. Add the **msp-android-sdk-release.aar** into your library's /lib folder,

2. in your build.gradle (Module.app) inside of the **dependencies** script add the following

```
implementation(name: 'msp-android-sdk-release', ext: 'aar')

repositories {
   flatDir {
       dirs 'libs'
   }
 }
```

3. sync Gradle:

You can now start using the Android SDK in your App.

React Native

1. Add dependency via NPM/Yarn

```
$ yarn add react-native-fastcheckout
```

2. Link dependencies

```
$ react-native link react-native-fastcheckout
```

iOS (via Cocoapods)

To install the native dependencies, we use **Cocoapods**. So, you will need to create or update a ***Podfile*** with the following configuration inside your Podfile *target*:

```
node_modules_path = "../node_modules"
react_native_path = "#{node_modules_path}/react-native"

# React Native
pod 'React', path: "#{react_native_path}/", subspecs: [
# Comment out any unneeded subspecs to reduce bundle size.
   'Core',
   'CxxBridge',
   'DevSupport',
   'RCTActionSheet',
   'RCTAnimation',
   #'RCTBlob',
   #'RCTCameraRoll',
   #'RCTGeolocation',
   'RCTImage',
   'RCTNetwork',
   #'RCTPushNotification',
   #'RCTSettings',
   #'RCTTest',
   'RCTText',
   #'RCTVibration',
   'RCTWebSocket',
   'RCTLinkingIOS'
 ]

 # Flexbox Layout Manager Used By React Natve
 pod 'yoga', path: "#{react_native_path}/ReactCommon/yoga"

 # React Native Fastcheckout SDK
 pod 'RNFastcheckoutManager', :path => "#{node_modules_path}/react-native-fastcheckout/ios"
```

Demo

iOS

Step by step

You can easily integrate Fastcheckout iOS SDK into your app, you just need to follow the next steps:

1. Setup iOS SDK

```
import FastcheckoutKit

let manager = FastcheckoutManager(client: FastcheckoutClient(apiKey: "API_KEY"))
```

2. Start checkout with completion callback

```
manager.startCheckout(transactionId: "ID", host: self, onCompletion: { status, error in
    if let status = status {
        print(status)
    } else {
        print(error?.localizedDescription)
    }
})
```

And that's it, that's all you need to do to integrate the iOS SDK to start a checkout and retrieve the transaction status once it completes.

Complete example

```
import UIKit
import FastcheckoutKit

class ViewController: UIViewController {

   private let manager = FastcheckoutManager(client: FastcheckoutClient(apiKey: "API_KEY"))

   override func viewDidLoad() {
       super.viewDidLoad()

       // Set `self` as delegate
       manager.delegate = self
```

```
        // Add a button to the navigationBar
        let startButton = UIBarButtonItem(title: "Start",
                                          style: .done,
                                          target: self,
                                          action: #selector(self.startCheckout))
        navigationItem.rightBarButtonItem = startButton
    }

    // MARK: Actions

    @objc func startCheckout() {
        manager.startCheckout(transactionId: "id", host: self, onCompletion: { status, error in
            if let status = status {
                print(status)
            } else {
                print(error?.localizedDescription)
            }
        })
    }

}
```

Android

Step by step

You can easily integrate the Fastcheckout Android SDK into your app, you just need to follow the next steps:

1. Setup the Android SDK. Add the following to your MainActivity or Class:

```
import com.multisafepay.sdk.FastCheckoutSDK;

//declare the FastCheckoutSDK as a member variable:
private static FastCheckoutSDK SDK = FastCheckoutSDK.getInstance();

//you must implement this interface and then set the listener:
Public  class  MainActivity extends AppCompatActivity
implements ISDKCommunicator.sdkStatus {

//in your onCreate method initialize the SDK:
SDK.setApiKey(API_KEY);

//set the listener for the interface:
SDKCommunicationListener sdkCommunicationListener = new SDKCommunicationListener();
sdkCommunicationListener.setSDKListener(this);
```

```
...
...
}
```

2. Start checkout and add callback interface to your Activity

```
//your class should implement the FastCheckoutSDK.Callback interface
FastCheckoutSDK.Callback

//add registration to your onCreate method:
sdk.registerCallback(this);

//then, start your checkout:
sdk.startCheckout(transaction_id, this);

//get results in your callback method:
@Override
public void callback(FastCheckoutSDK.Result result) {

   if (result.getResult() == FastCheckoutSDK.Result.FCO_RESULT_UNCLEARED) {

   }

   if (result.getResult() == FastCheckoutSDK.Result.FCO_RESULT_OK) {

   }

   if (result.getResult() == FastCheckoutSDK.Result.FCO_RESULT_CANCELLED) {

   }
}
```

3. Setup styles in your Application class (these styles might be changed to suit your theme):

```
Styles.Builder builder = new Styles.Builder();
builder.set("mainColor", "#FF7E03");
builder.set("mainFontSize", 30);


{
Typeface typeface = Typeface.createFromAsset(this.getAssets(),
"fonts/nunito/Nunito-Medium.ttf");
   Styles.Font font = new Styles.Font("#000000", typeface, null);
   builder.setObject("labelFont", font);
}

{
 Typeface typeface = Typeface.createFromAsset(this.getAssets(),
"fonts/nunito/Nunito-Medium.ttf");
```

```
   Styles.Font font = new Styles.Font("#FFFFFF", typeface, null);
   builder.setObject("buttonFont", font);
}

{
 Typeface typeface = Typeface.createFromAsset(this.getAssets(),
"fonts/nunito/Nunito-ExtraLight.ttf");
   Styles.Font font = new Styles.Font("#000000", typeface, null);
   builder.setObject("editFont", font);
}

{
   Typeface typeface = Typeface.createFromAsset(this.getAssets(), "fonts/galada_regular.ttf");
   Styles.Font font = new Styles.Font("#000000", typeface, null);
   builder.setObject("titleFont", font);
   builder.set("title", "qwindo");
}
Styles styles = builder.build();
FastCheckoutSDK sdk = FastCheckoutSDK.getInstance();
sdk.setStyles(styles);
```

Complete example

```
import com.multisafepay.sdk.FastCheckoutSDK;

public class MainActivity extends AppCompatActivity implements FastCheckoutSDK.Callback{

private static FastCheckoutSDK SDK;
private Button btnStartNewTransaction;

@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.main_activity);

   btnStartNewTransaction = findViewById(R.id.btn_transaction);

   SDK = FastCheckoutSDK.getInstance();
   SDK.setApiKey(API_KEY);

   btnLogOut.setOnClickListener(view -> SDK.startCheckout(transaction_id, this));

   }

@Override
public void callback(FastCheckoutSDK.Result result) {
```

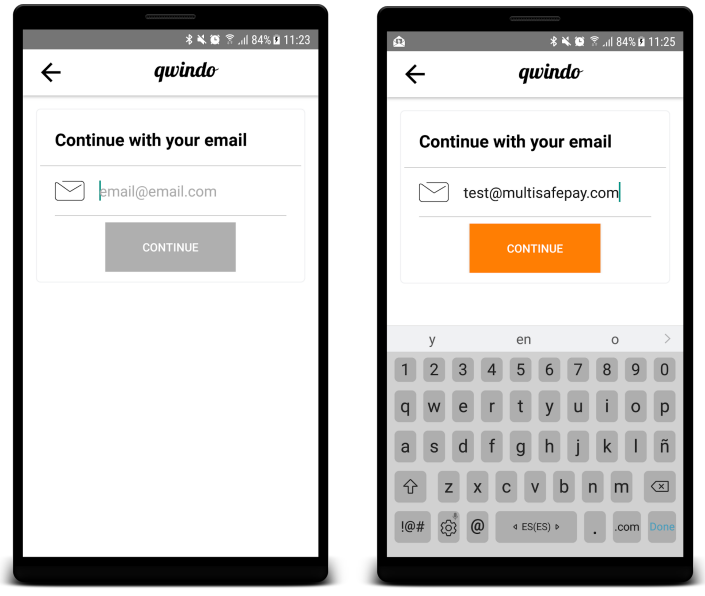
```
    if (result.getResult() == FastCheckoutSDK.Result.FCO_RESULT_UNCLEARED) {

    }

    if (result.getResult() == FastCheckoutSDK.Result.FCO_RESULT_OK) {

    }

    if (result.getResult() == FastCheckoutSDK.Result.FCO_RESULT_CANCELLED) {

    }

  }
}
```

```
import com.multisafepay.sdk.FastCheckoutSDK;

public class YourApplicationClassName extends Application {

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main_activity);

    Styles.Builder builder = new Styles.Builder();
    builder.set("mainColor", "#FF7E03");
    builder.set("mainFontSize", 30);


{
Typeface typeface = Typeface.createFromAsset(this.getAssets(),
"fonts/nunito/Nunito-Medium.ttf");
    Styles.Font font = new Styles.Font("#000000", typeface, null);
    builder.setObject("labelFont", font);
}

{
 Typeface typeface = Typeface.createFromAsset(this.getAssets(),
"fonts/nunito/Nunito-Medium.ttf");
    Styles.Font font = new Styles.Font("#FFFFFF", typeface, null);
    builder.setObject("buttonFont", font);
}
```

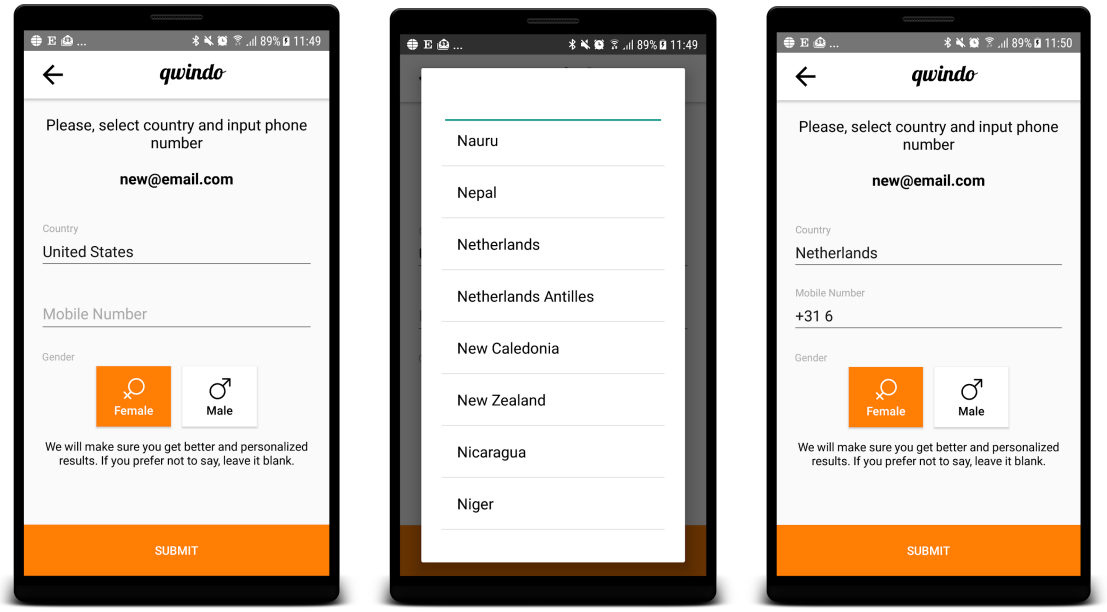
```
{
 Typeface typeface = Typeface.createFromAsset(this.getAssets(),
"fonts/nunito/Nunito-ExtraLight.ttf");
   Styles.Font font = new Styles.Font("#000000", typeface, null);
   builder.setObject("editFont", font);
}

{
   Typeface typeface = Typeface.createFromAsset(this.getAssets(), "fonts/galada_regular.ttf");
   Styles.Font font = new Styles.Font("#000000", typeface, null);
   builder.setObject("titleFont", font);
   builder.set("title", "qwindo");
}
Styles styles = builder.build();
FastCheckoutSDK sdk = FastCheckoutSDK.getInstance();
sdk.setStyles(styles);

   }
}
```

SDK Snapshots:

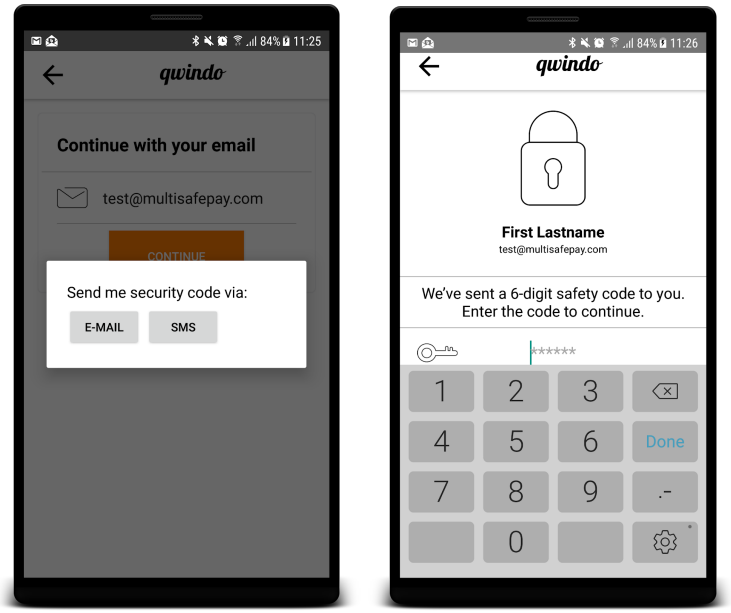
1) The following snapshots will walk you through some of features offered by the Fastcheckout Android SDK. The Fastcheckout iOS SDK provides the same functionalities.
    a) Logging in with a registered email:



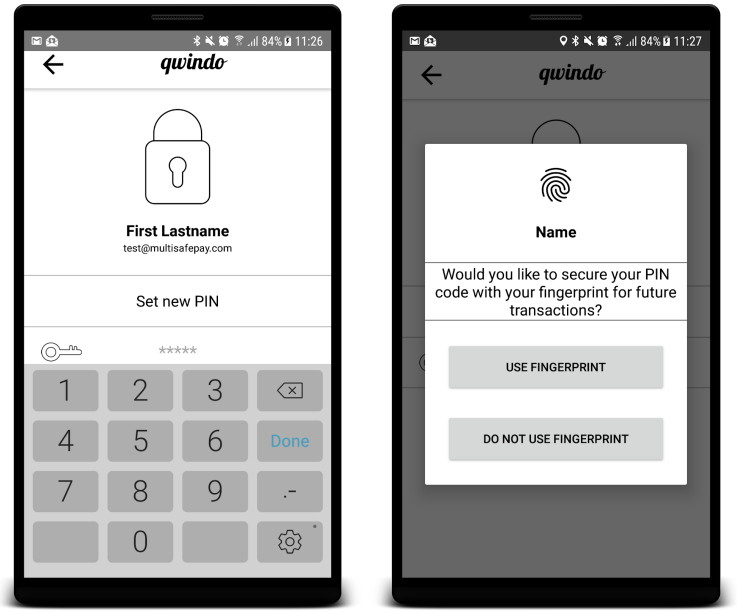
b) Logging in with a non-registered email will automatically take the user to the register SDK screen:



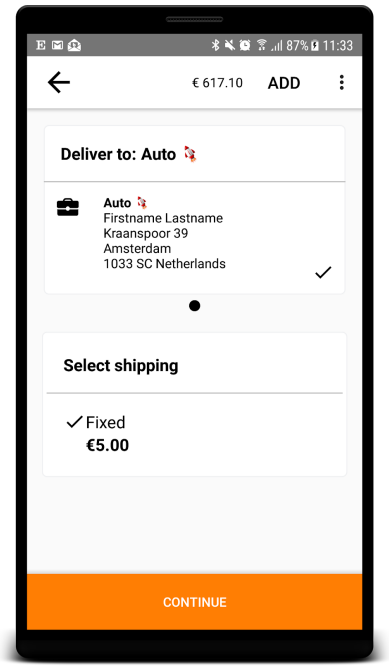
c) If the user is registered and resets its email account, the SDK will automatically send a new security code to new entered email:

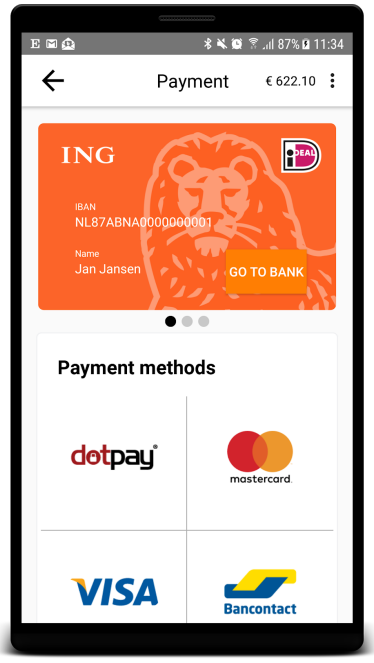


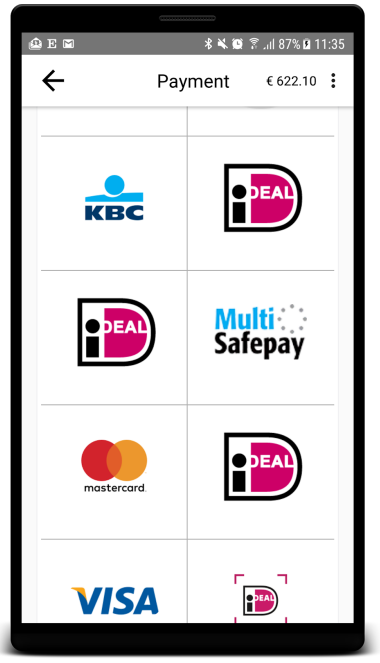
d) Once the user enters the security code received by email (if the security code is received via SMS it will automatically get added to appropriate field, and the SDK will move into the following state) the SDK will ask for a new security PIN, once entered, the SDK will provide Biometric options (in iOS face recognition is also available):

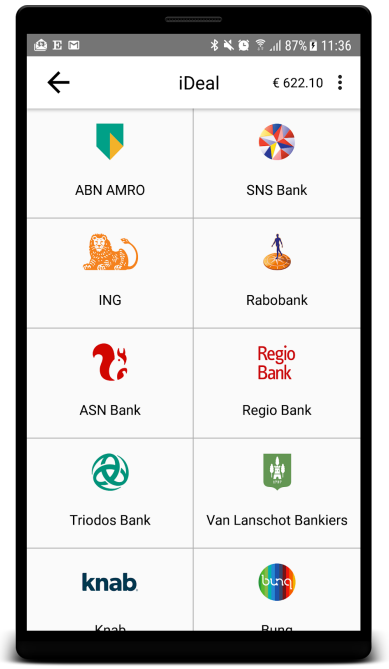


e) The checkout process follows. From a merchant's shop checkout option, the SDK enters into the checkout process. The first view will be the delivery view with shipping options if available. Once the continue button is selected, the SDK moves into the Payment screen and the payment logic follows.



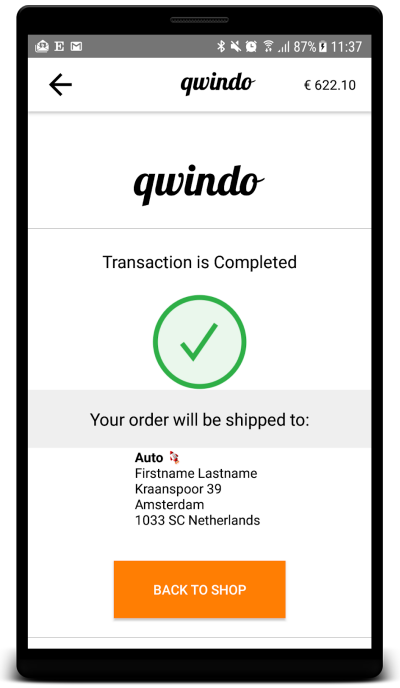






i) Once the payment process is completed the SDK will proceed to the “transaction is complete” screen. The SDK callback will notify the client App

about this or any other results (uncleared, cancel, etc.). Pressing the button “Back to Shop” will redirect the user to the Merchant’s shop.



React Native

Step by step

You can easily integrate Fastcheckout SDK into your app, you just need to follow the next steps:

1. Setup SDK

```
import { FastcheckoutManager } from "react-native-fastcheckout";

const apiKey = "API_KEY";
const environment = "ENV_KEY";
FastcheckoutManager.init(apiKey, environment);
```

2. Start checkout with completion callback

```
const status = await FastcheckoutManager.startCheckoutWithTransactionId("ID")
try {
  console.log(status);
} catch (error) {
  console.error(error);
}
```