iOS SDK Documentation

February 8

By Midtrans

# Table Of Contents

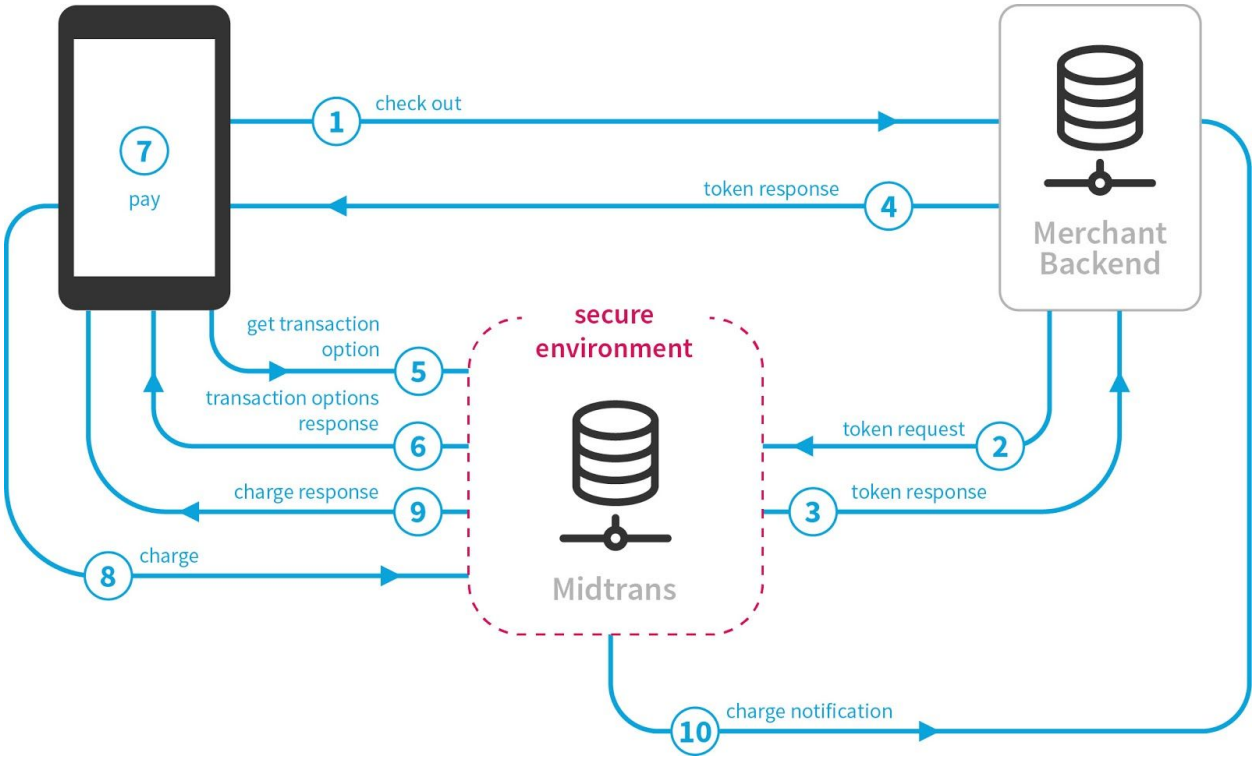
# Getting Started

Midtrans mobile SDK enable merchants to accept online payments natively in their mobile apps. We provide the drop in User interface for making transactions on all the payment types supported by Midtrans. Watch the video for the default SDK example.

There are four parties involved in the payment process for making a payment:

1. Merchant Server: The merchant backend implementation
2. Customers
3. Midtrans Backend (Payment Processor)
4. Midtrans Mobile SDK

## Transaction flow




1. Checkout: Customer clicks the Checkout button on the Host application and the app makes a request to the Merchant Server
2. Token request: Merchant Server makes a request to Veritrans server with Order Information.
3. Token response: Midtrans responds with a valid transaction token to Merchant server
4. Token response: Merchant server forwards the token to the Mobile SDK
5. Get transaction options: Mobile SDK requests payment/merchant information based on the token

6. Transaction options response: Mobile SDK renders the payment Options and payment information to make the payment
7. Pay: Customers selects the payment method and the payment details and clicks "Pay"
8. Charge: Mobile SDK sends the Charge request to the Veritrans Backend for payment Processing.
9. Charge response: Mobile SDK receives the response from the Veritrans Backend and triggers the handler on Mobile App with success/failure/pending status
10. Charge notification: Midtrans Backend sends a notification to the Merchant backend confirming the completion of transaction.

# Supported Payment Methods

1. Credit/Debit Cards - Support for making payments via credit cards and or debit cards using our two-clicks feature. We support Visa, Mastercard, American Express and JCB
2. Mandiri ClickPay
3. CIMB Clicks
4. ePay BRI
5. Indosat Dompetku
6. Mandiri e-Cash
7. Bank Transfer - Support payment using Permata Virtual Account and BCA Virtual Account.
8. Mandiri Bill Payment
9. Indomaret - Payment via convenience Stores
10. BCA Klikpay
11. KlikBCA
12. Kioson
13. Gift Card Indonesia

# Security Aspects

- There are 2 separate keys CLIENT_KEY and SERVER_KEY (available on MAP)
    - CLIENT_KEY is used for tokenizing the credit card. It can only be used from the Client(mobile device)
    - SERVER_KEY is used for acquiring the token from the Midtrans server. It is not to be used from the device, all API requests that use the SERVER_KEY need to be made from the Merchant Server.
- We use strong encryption for making connections to Merchant server, please make sure it has valid https Certificate.

Following are configurable parameters of SDK that can be used while performing transaction -

- Merchant server Endpoint / Base URL : URL of server to which transaction data will be sent. This will also be referred to as a merchant server.
- Transaction details - contains payment information like amount, order Id, payment

method etc.
- Midtrans Client Key - token that specified by merchant server to enable the transaction using credit card. Available on the MAP

# Prerequisites

1. Create a merchant account in MAP
2. In MAP, setup your merchant accounts settings, in particular Notification URL.
3. Setup your merchant server. A server side implementation is required for midtrans mobile SDK to work. You can check the server implementation reference, and walk through the API's that you may need to implement on your backend server.
4. Minimum requirements: iOS 7 above

# iOS SDK

This SDK provides an UI to take required information from user to execute transaction.

## Installation

1. Please install Cocoapods version 1.0.0 above. You can find the installation guide at this link https://guides.cocoapods.org/using/getting-started.html#installation.
2. Navigate to your project's root directory and run `pod init` to create a Podfile.
3. Open up the Podfile and add MidtransKit to your project's target

Objective C

```
def shared_pods
    pod 'MidtransCoreKit'
    pod 'MidtransKit'
end

target 'MyBeautifulApp' do
    shared_pods
end
```

Swift

```
use_frameworks!

def shared_pods
    pod 'MidtransKit'
end

target 'MyBeautifulApp' do
    shared_pods
end
```

3. Save the file and run `pod install` to install MidtransKit.
4. Cocoapods will download and install MidtransKit and also create a .xcworkspace project.
5. Once you have completed installation of MidtransKit, configure it with your `clientKey`, `merchant server URL` and `server environment` in your `AppDelegate.h`

Objective C

```
//AppDelegate.m

#import <MidtransKit/MidtransKit.h>

[CONFIG setClientKey:@"VT-CLIENT-sandbox-client-key"
         environment:MidtransServerEnvironmentSandbox
   merchantServerURL:@"https://merchant-url-sandbox.com"];
```

Swift

```
//AppDelegate.swift

import MidtransKit

MidtransConfig.shared().setClientKey("client key", environment: .sandbox,
merchantServerURL: "merchant server url")
```

# Making Trasanction

## Generate `TransactionTokenResponse` object

To create this object, you need to prepare required objects like item object etc.

Objective C

```
//ViewController.m
MidtransItemDetail *itemDetail =
[[MidtransItemDetail alloc] initWithItemID:@"item_id"
                                      name:@"item_name"
                                     price:item_price
                                  quantity:item_quantity];

MidtransCustomerDetails *customerDetail =
[[MidtransCustomerDetails alloc] initWithFirstName:@"user_firstname"
                                          lastName:@"user_lastname"
                                             email:@"user_email"
                                             phone:@"user_phone"
                                   shippingAddress:ship_address
                                    billingAddress:bill_address];

MidtransTransactionDetails *transactionDetail =
[[MidtransTransactionDetails alloc] initWithOrderID:@"order_id"
                                     andGrossAmount:items_gross_amount];

[[MidtransMerchantClient shared]
 requestTransactionTokenWithTransactionDetails:transactionDetail
 itemDetails:self.itemDetails
 customerDetails:customerDetail
 completion:^(MidtransTransactionTokenResponse *token, NSError *error)
```

```
 {
     if (token) {

     }
     else {

     }
 }];
```

Swift

```
//ViewController.swift
let itemDetail = MidtransItemDetail.init(itemID: item_id, name: item_name, price:
item_price, quantity: item_qty)

let customerDetail = MidtransCustomerDetails.init(firstName: first_name, lastName:
last_name, email: email_addr, phone: phone_number, shippingAddress: ship_addr,
billingAddress: bill_addr)

let transactionDetail = MidtransTransactionDetails.init(orderID: order_ir,
andGrossAmount: gross_amount)

MidtransMerchantClient.shared().requestTransactionToken(with: transactionDetail!,
itemDetails: [itemDetail!], customerDetails: customerDetail) { (response, error) in
    if (response != nil) {
        //handle response
    }
    else {
        //handle error
    }
}
```

## Present the `MidtransUIPaymentViewController`

We provide you a `MidtransUIPaymentViewController` to handle all the payment. Use generated `TransactionTokenResponse` as required parameter.

Objective C

```
MidtransUIPaymentViewController *vc = [[MidtransUIPaymentViewController alloc]
initWithToken:token];
[self presentViewController:vc animated:YES completion:nil];
```

Swift

```
let vc = MidtransUIPaymentViewController.init(token: response)
self.present(vc!, animated: true, completion: nil)
```

## Get Notified

SDK will give callback/ delegate transaction response to host app. To be able to do so, please follow these steps:

- Set your view controller to conform with `MidtransUIPaymentViewControllerDelegate`

Objective C

```
//ViewController.m

#import <MidtransKit/MidtransKit.h>

@interface ViewController () <MidtransUIPaymentViewControllerDelegate>

//other code
```

Swift

```
//ViewController.swift

import MidtransKit

class ViewController: UIViewController, MidtransUIPaymentViewControllerDelegate
//other code
```

- Set the delegate of `MidtransUIPaymentViewContro ller`

Objective C

```
//ViewController.m
MidtransUIPaymentViewController *vc = [[MidtransUIPaymentViewController alloc]
initWithToken:token];
//set the delegate
vc.paymentDelegate = self;
```

Objective C

```
//ViewController.swift
let vc = MidtransUIPaymentViewController.init(token: response)
//set the delegate
vc?.paymentDelegate = self
self.present(vc!, animated: true, completion: nil)
```

- Implement Implement the `MidtransUIPaymentViewControllerDelegate` functions

Objective C

```
//ViewController.m

#pragma mark - MidtransUIPaymentViewControllerDelegate

- (void)paymentViewController:(MidtransUIPaymentViewController *)viewController
paymentSuccess:(MidtransTransactionResult *)result {
    NSLog(@"success: %@", result);
}
- (void)paymentViewController:(MidtransUIPaymentViewController *)viewController
```

```
paymentFailed:(NSError *)error {
    [self showAlertError:error];
}
- (void)paymentViewController:(MidtransUIPaymentViewController *)viewController
paymentPending:(MidtransTransactionResult *)result {
    NSLog(@"pending: %@", result);
}
- (void)paymentViewController_paymentCanceled:(MidtransUIPaymentViewController
*)viewController {
    NSLog(@"canceled");
}
```

Swift

```
//ViewController.swift

#pragma mark - MidtransUIPaymentViewControllerDelegate

func paymentViewController(_ viewController: MidtransUIPaymentViewController!,
paymentFailed error: Error!) {

}
func paymentViewController(_ viewController: MidtransUIPaymentViewController!,
paymentPending result: MidtransTransactionResult!) {

}
func paymentViewController(_ viewController: MidtransUIPaymentViewController!,
paymentSuccess result: MidtransTransactionResult!) {

}
func paymentViewController_paymentCanceled(_ viewController:
MidtransUIPaymentViewController!) {

}
```

# Customisation

## Theme Customisation

We've created `MidtransUIThemeManager` to configure the theme color and font of the veritrans payment UI.

Class `MidtransUIThemeManager` needs `UIColor` object so you need to convert your HEX or RGB to UIColor, you can visit this link http://uicolor.xyz/#/hex-to-ui it has some nice tool to generate `UIColor` code.

Note: If you didn't configure this `MidtransUIThemeManager` , your theme color will follow SNAP color configuration on MAP SNAP settings preference.

Objective C

```
MidtransUIFontSource fontSource =
[[MidtransUIFontSource alloc] initWithFontNameBold:font_name
                                    fontNameRegular:font_name
                                      fontNameLight:font_name];
[MidtransUIThemeManager applyCustomThemeColor:themeColor
                                     themeFont:fontSource];
```

Swift

```
let fontSource = MidtransUIFontSource.init(fontNameBold: font_name, fontNameRegular:
font_name, fontNameLight: font_name)

MidtransUIThemeManager.applyCustomThemeColor(theme_color, themeFont: fontSource)
```

## Skipping Transaction Result Page

Sometimes you don't want to show our payment result page because you already have design for it, so we give you flexibility to use it or just hide it.

Objective C

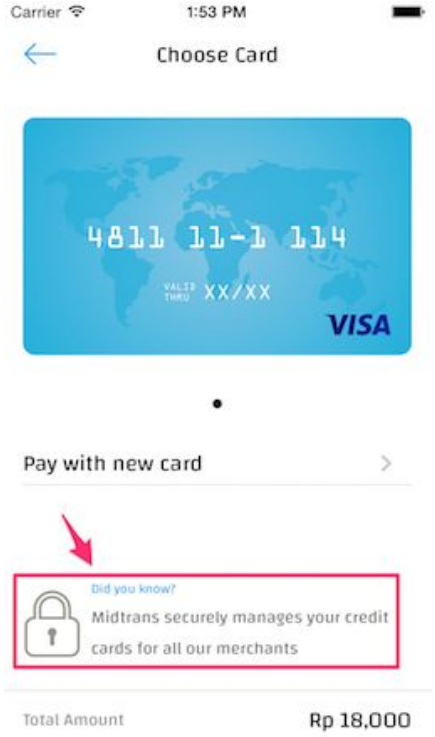
```
UICONFIG.hideStatusPage = YES;
```

Swift

```
MidtransUIConfiguration.shared().hideStatusPage = true
```

## Hide “Did You know?” Label

You can see what Did You Know label is below




If you wanto to hide it then add this code before using SDK

Objective C

```
UICONFIG.hideDidYouKnowView = YES;
```

Swift

```
MidtransUIConfiguration.shared().hideDidYouKnowView = true
```

## Set Save Card Checkbox Checked by Default

You can save card checkbox on Credit Card payment page checked by default

Objective C

```
CC_CONFIG.setDefaultCreditSaveCardEnabled = YES;
```

Swift

```
MidtransUIConfiguration.shared().setDefaultCreditSaveCardEnabled = true
```

## Enable Card Scanner (using card.io framework)

We provide a plugin to integrate card.io for allowing customers to read the credit card/debit card information using the mobile phone camera. If you want to support this external card scanner, follow these steps.

- Update your `Podfile` just like on the example code

Objective C or Swift

```
def shared_pods
    pod 'MidtransKit'
    pod 'MidtransKit/CardIO'
end

target 'MyBeautifulApp' do
    shared_pods
end
```

- Then update your pods

```
pod update --verbose
```

## Direct link to specific payment method

We have flexibility to call directly to a specific payment method. If you want to show only credit card payment page you can do it like on the example code.

Objective C

```
MidtransUIPaymentViewController *paymentVC = [[MidtransUIPaymentViewController
alloc] initWithToken:token andPaymentFeature:MidtransPaymentFeatureCreditCard];
```

Swift

```
let vc = MidtransUIPaymentViewController.init(token: response, paymentFeature:
.creditcard)
self.present(vc!, animated: true, completion: nil)
```

# Additional Payment Features

## One-Click

Please see the configuration on the example code, then implement it configuration before presenting the payment page.

Objective C

```
CC_CONFIG.paymentType = MTCreditCardPaymentTypeOneclick;
CC_CONFIG.saveCardEnabled = YES;

//1-click need token storage enabled
CC_CONFIG.tokenStorageEnabled = YES;

//1-click need 3ds enabled
CC_CONFIG.secure3DEnabled = YES;
```

Swift

```
MidtransCreditCardConfig.shared().paymentType = .oneclick
MidtransCreditCardConfig.shared().saveCardEnabled = true

//1-click need token storage enabled
MidtransCreditCardConfig.shared().tokenStorageEnabled = true

//1-click need 3ds enabled
MidtransCreditCardConfig.shared().secure3DEnabled = true
```

## Two-Clicks

Please see the configuration on the example code, then implement it configuration before presenting the payment page.

Objective C

```
CC_CONFIG.paymentType = MTCreditCardPaymentTypeTwoclick;
CC_CONFIG.saveCardEnabled = YES;
```

Swift

```
MidtransUIConfiguration.shared().paymentType = .twoclick
MidtransUIConfiguration.shared().saveCardEnabled = true
```

## Custom Acquiring Bank

We've already support these bank

1. BCA
2. BRI
3. CIMB
4. Mandiri
5. BNI
6. Maybank

Supported Bank

```
MTAcquiringBankBCA,
MTAcquiringBankBRI,
MTAcquiringBankCIMB,
MTAcquiringBankMandiri,
MTAcquiringBankBNI,
MTAcquiringBankMaybank
```

Objective C

```
CC_CONFIG.acquiringBank = MTAcquiringBankMaybank; //ex. maybank
```

Swift

```
MidtransUIConfiguration.shared().acquiringBank = MTAcquiringBankMaybank; //ex.
maybank
```