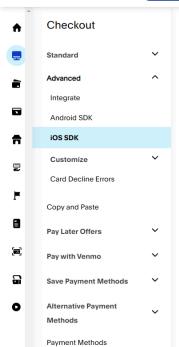




APIs & SDKs

Community Support PayPal.com



Checkout / Advanced / iOS SDK

# Integrate card payments in iOS apps

SDK CURRENT ADVANCED Last updated: December 7th 2023, @ 10:43:54 am

Accept PayPal, credit, and debit card payments in a web or native experience using the PayPal Mobile iOS SDK. Use customizable PayPal buttons with your custom checkout UI to align with your business branding. For more implementation details, see the PayPal GitHub repository 2.

# Know before you code

You need a developer account to get sandbox credentials:

- PayPal uses REST API credentials which you can get from the developer dashboard.
- Client ID: Authenticates your account with PayPal and identifies an app in your sandbox.
- Client secret: Authorizes an app in your sandbox. Keep this secret safe and don't share it.

Read Get started with PayPal APIs for more information.

You need a combination of PayPal and third-party tools:

- iOS SDK ☑: Adds PayPal-supported payment methods for iOS.
- Orders REST API: Create, update, retrieve, authorize, and capture orders.

Use Postman to explore and test PayPal APIs.



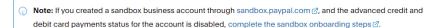
# 1. Before you begin your integration

### Check your account setup for advanced card payments

This integration requires a sandbox business account with the Advanced Credit and Debit Card Payments capability. Your account should automatically have this capability.

To confirm that Advanced Credit and Debit Card Payments are enabled for you, check your sandbox business account as follows:

- 1. Log into the PayPal Developer Dashboard, toggle Sandbox, and go to Apps & Credentials.
- 2. In REST API apps, select the name of your app.
- 3. Go to Features > Accept payments.
- 4. Select the Advanced Credit and Debit Card Payments checkbox and select Save Changes.



## Check 3D Secure requirements

Add 3D Secure to reduce the chance of fraud and improve the payment experience by authenticating a cardholder through their card issuer.

Visit our 3D Secure page to see if 3D Secure is required in your region and learn more about implementing 3D Secure in your app.

# 2. Integrate the SDK into your app

Integrate 3 different types of payments using the PayPal Mobile SDK:

- · Card payments: Add card fields that align with your branding.
- PayPal native payments: Launch a checkout page within your app, instead of a popup.
- PayPal web payments: A lighter integration that launches a checkout page in a browser within your app.

Card Native payments Web payments

## Integrate with card payments

Build and customize the card fields to align with your branding.

### 1. Add card payments module to your app

Add the CardPayments package dependency for your app using Swift Package Manager or CocoaPods:

### On this page

Know before you code

- 1. Before you begin your integration
- 2. Integrate the SDK into

Payment buttons and fraud

Go live

Swift Package CocoaPods Manager 1. Open Xcode. 2. Follow the guide  $\ensuremath{ \mathbb{Z} }$  to add package dependencies to your app. 3. Enter https://github.com/paypal/paypal-ios/♂ ♂ as the repository URL. 4. Select the checkbox for the CardPayments framework. 2. Create CardClient

A CardClient helps you attach a card to a payment.

In your iOS app:

- 1. Use the CLIENT\_ID to construct a CoreConfig .
- 2. Construct a CardClient using your CoreConfig object.

```
1 let coreConfig = CoreConfig(clientID: "CLIENT_ID", environment: .sandbox)
2 let cardClient = CardClient(config: coreConfig)
```

### 3. Get Order ID

On your server:

- 1. Create an ORDER\_ID by using the Orders v2 API.
- 2. Pass your ACCESS\_TOKEN in the Authorization header. To get an ACCESS\_TOKEN, use the Authentication API.
  - Note: This access token is only for the sandbox environment. When you're ready to go live, request a live access token by changing the request sandbox endpoint to https://api-m.paypal.com/v1/oauth2/token 2.
- 3. Pass the intent . You'll need to pass either AUTHORIZE or CAPTURE as the intent type. This type must match the /authorize or /capture endpoint you use to process your order.

```
Sample request Sample response
                                                                                                                                               -H 'Content-Type: application/json' \
-H 'Authorization: Bearer ACCESS_TOKEN' \
      --data-raw '{
  "intent": "CAPTURE|AUTHORIZE",
                "currency_code": "USD",
"value": "5.00"
```

When a buyer starts a payment, send the ORDER\_ID from your server to your client app.

### 4. Create card request

A CardRequest object:

- · Attaches a card to an ORDER ID .
- Launches 3D Secure when a payment requires additional authentication.

# 1. Collect card payment details

Build a card object with the buyer's card details:

```
expirationMonth: "01",
expirationYear: "2025",
securityCode: "123",
cardholderName: "Jane Smith",
    addressLine1: "123 Main St.",
addressLine2: "Apt. 1A",
      locality: "City",
region: "IL",
postalCode: "12345",
```

```
14 )
15 )
```

Collecting a billing address can reduce the number of authentication challenges to customers.

## 2. Build CardRequest

Build a  $\mbox{ CardRequest }$  with the  $\mbox{ card }$  object and your  $\mbox{ ORDER\_ID }$  :

```
1 let cardRequest = CardRequest(
2 orderID: "ORDER_ID",
3 card: card,
4 sca: .scaAlways // default value is .scaWhenRequired
5 )
```

3D Secure is supported for all card payments to comply with the Second Payment Services Directive (PSD2) 2. PSD2 is a European Union regulation that introduces Strong Customer Authentication (SCA) 2 and other security requirements.

Select your SCA launch option type using the sca parameter in the CardRequest initializer:

- SCA.scaWhenRequired launches an SCA challenge when applicable. This is enabled by default.
- SCA.scaAlways requires an SCA challenge for all card transactions.

### 5. Approve order

After your CardRequest has the card details, call cardClient.approveOrder() to process the payment.

```
1 class MyViewController: UIViewController {
2   func cardCheckoutTapped(cardRequest: CardRequest) {
3         cardClient.approveOrder(request: cardRequest)
4   }
5 }
```

### 6. Handle payment result scenarios

Set up your CardDelegate to handle successful payments, errors, cancellations, and 3D Secure transaction flows.

```
textension MyViewController: CardDelegate {
  func setupCardClient() {
    cardClient.delegate = self
  }
  // MARK: - CardDelegate
  func card(_ cardClient: CardClient, didFinishWithResult result: CardResult) {
    // order was approved and is ready to be captured/authorized (see step 8)
  }
  func card(_ cardClient: CardClient, didFinishWithError error: CoreSDKError) {
    // handle the error by accessing `error.localizedDescription`
    }
  func cardDidCancel(_ cardClient: CardClient) {
        // 3D Secure auth was canceled by the user
    }
  func cardThreeDSecureWillLaunch(_ cardClient: CardClient) {
        // 3D Secure auth did finish successfully
    }
}

func cardThreeDSecureDidFinish(_ cardClient: CardClient) {
        // 3D Secure auth did finish successfully
}
```

## 7. Authorize and capture order

Submit your ORDER\_ID for authorization or capture when the PayPal iOS SDK calls the didFinishWithResult method.

Call the authorize endpoint of the Orders V2 API to place the money on hold:

Sample request: Authorize order

```
1 curl --location --request POST 'https://api-m.sandbox.paypal.com/v2/checkout/orders/ORDER_ID/authorize' \
2 -H 'Content-Type: application/json' \
3 -H 'Authorization: Bearer ACCESS_TOKEN' \
4 --data-raw ''
```

Call the capture endpoint of the Orders V2 API to capture the money immediately: