

PassKit (Apple Pay and Wallet)

Documentation / PassKit (Apple Pay and Wallet) / Apple Pay / Setting up Apple Pay

Language: Swift API Changes: None

Apple pay support

Apple Pay

Apple Pay setup

Setting up Apple Pay

Offering Apple Pay in Your App

Complying with regional regulations

Apple Pay availability

PKPaymentAuthorizationController

PKPaymentAuthorizationViewController

Apple Pay buttons

PKPaymentButton

PayWithApplePayButton

PayWithApplePayButtonLabel

PayWithApplePayButtonStyle

Payment requests

PKPaymentRequest

PKRecurringPaymentRequest

PKAutomaticReloadPaymentRequest

PKDeferredPaymentRequest

PKPaymentTokenContext

Payment sheet interactions and authorization

PKPaymentAuthorizationResult

PKPaymentOrderDetails

PKPaymentAuthorizationController

PKPaymentAuthorizationViewController

PKPayment

Payment sheet updates

PKPaymentRequestMerchantSessionUpdate

PKPaymentRequestPaymentMethodUpdate

PKPaymentRequestShippingContactUpdate

PKPaymentRequestShippingMethodUpdate

PKPaymentRequestCouponCodeUpdate

PKPaymentRequestUpdate

QR transaction information

PKPaymentInformationEventExtension

PKPaymentInformationRequestHandling

Entitlements

Filter

Article

Setting up Apple Pay

Fulfill the requirements to provide Apple Pay as a payment option on your website or in your app.

Overview

To set up your Apple developer account and Xcode to implement Apple Pay in your apps, you complete three steps:

Create a merchant identifier.

Create a Payment Processing certificate.

Enable Apple Pay in Xcode.

Create a merchant identifier

To enable your app to use Apple Pay, register an identifier with Apple that uniquely identifies your business as a merchant able to accept payments. This ID never expires, and you can use it in multiple websites and apps. See [Create a merchant identifier](#) for the setup steps.

Create a payment processing certificate

Using your registered merchant identifier, create a certificate to secure transaction data. Apple Pay servers use the certificate's public key to encrypt payment data. You (or your payment service provider) use the private key to decrypt the data to process payments. See [Create a payment processing certificate](#) for the setup steps.

Note

If you use an e-commerce provider or a payment platform, contact them for information about how to use their service with Apple Pay. See [Payment Platforms](#) for a list of service providers.

Enable Apple Pay capability in Xcode

After creating a merchant identifier, enable the Apple Pay capability in your Xcode project.

Open your project with Xcode. In the Project navigator, select the project.

Choose the target for the app from either the Project/Targets pop-up menu or in the Targets section of the outline view.

Click the Signing & Capabilities tab in the project editor.

In the toolbar, click the Library button (+) to open the Capabilities library and select the Apple Pay capability.

Within the Apple Pay capability, click the refresh button to synchronize your merchant identifiers from the Apple Developer site.

Select the merchant identifier to use with this app.

The screenshot below shows the Apple Pay capability without any merchant identifiers:



For more information, see [Adding capabilities to your app](#).

Configure Apple Pay on the web

If you're also developing websites using [Apple Pay on the Web](#), you can use the same merchant ID and Payment Processing Certificate for your website. However, Apple Pay on the web requires additional setup; see [Configuring Your Environment](#) for more information.

See Also

Apple Pay setup

Offering Apple Pay in Your App

Collect payments with iPhone and Apple Watch using Apple Pay.

Complying with regional regulations

Check regional regulations for possible requirements for your Apple Pay-based implementation.