# Getting Started with Apple Pay

Version 2.1

November 2015

## Overview

Apple Pay provides an easy and secure way for users to buy physical goods and services in your app. Customers prefer using Apple Pay as they can make purchases with a single touch rather than having to fill in billing, shipping, and contact details. And customers and developers like the peace of mind that comes with the increased security of Apple Pay. Apple does not store or share customer's actual credit and debit card numbers, removing the liability for merchants and app developers to manage and secure actual credit and debit card numbers. Learn more about Apple Pay security and privacy. If you're interested in developing a rewards card pass that can be used with Apple Pay, go to the Getting Started with Wallet guide.

Due to these benefits, developers that have implemented Apple Pay in the recommended way have experienced 2x increases in checkout conversation rates, a substantial reduction in checkout time, increased customer loyalty and increased purchase frequency after integrating Apple Pay.

Within your app, users can authorize payments using Touch ID, releasing tokenized credit and debit card payment credentials that are securely stored on iPhone and iPad. Additionally, users can store their billing, shipping and contact information within the Wallet app, which can be provided along with payment credentials when the user authorizes the purchase with Touch ID within your app.

Apple Pay is currently available in Australia, Canada, the United Kingdom, and the United States.

## Apple Pay or In-App Purchase

It is important to understand the difference between Apple Pay and In-App Purchase. Use Apple Pay to sell physical goods such as groceries, clothing, and appliances. Also use Apple Pay for services such as club memberships, hotel reservations, and tickets for events. On the other hand, use In-App Purchase to sell virtual goods such as premium content for your app, and subscriptions for digital content.

The Apple Pay Programming Guide provides details on how to use the PassKit framework to integrate Apple Pay. The In-App Purchase Programming Guide provides details on how to use the StoreKit framework to integrate In-App Purchases.

## Prerequisites

In addition to implementing Apple Pay with the PassKit framework, you must:

- Set up an account with a payment processor or gateway, if you don't already have one. You can find a list on developer.apple.com/apple-pay.
- Register a Merchant Identifier via Certificates, Identifiers & Profiles
- Submit a Certificate Signing Request to obtain Public and Private keys that will be used to encrypt and decrypt Payment Tokens
- Include an Apple Pay entitlement in your app.

## App Review Guidelines

Your app must comply with the requirements specified in Section 29 of the App Review Guidelines.

## Payment Providers

You can find a list of payment providers who support Apple Pay with their SDKs on developer.apple.com/apple-pay/. Using one of these SDKs is highly recommended. Contact your payment provider for more information.

The alternative is to provide your own server-side solution to receive payments from your app, decrypt payment tokens and interface with the payment provider. Handling credit and debit card payments can be complicated and unless you already have the expertise and systems in place, an SDK from a payment provider is the quickest and most reliable way to support Apple Pay in your app.

## Presenting the Apple Pay button

PassKit provides the APIs that your app will use to determine if it is running on a device with a Secure Element and if the device has been provisioned with payment cards that you support.

If the device is Apple Pay enabled you should present the Buy with Apple Pay button using APIs supplied within PassKit.

If the device does not have Apple Pay, you can present the Set up Apple Pay button in place of the Buy with Apple Pay button, which allows the user to easily set up Apple Pay, or you can choose not to show the Apple Pay button.

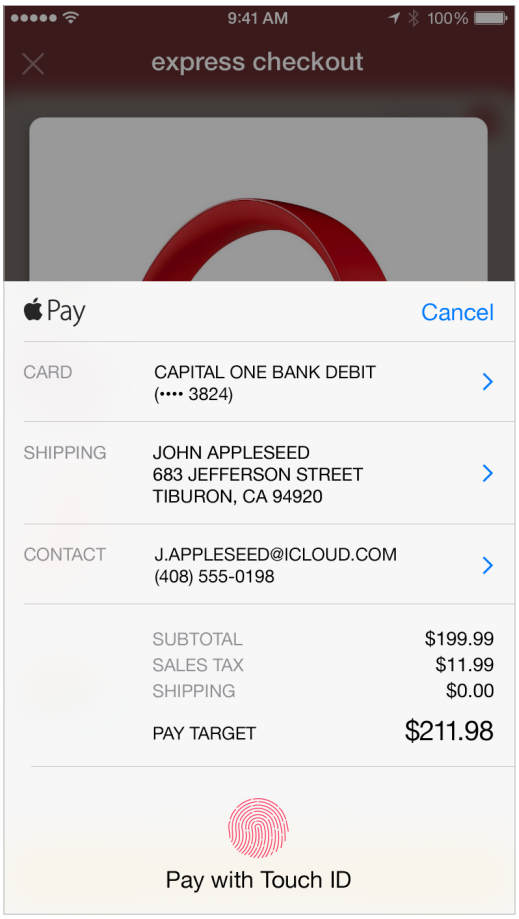
Placing the Apple Pay button in your app must be done in accordance with the Apple Pay Identity Guidelines

## Presenting the Payment Sheet

When your user selects goods or services to buy, and selects Apple Pay as the payment method, you create a payment request and ask PassKit to present the payment sheet to the user. See Figure 1. The payment sheet must immediately follow the user tapping the Apple Pay button, without any interim screens or pop-ups.

Your app specifies the contents of the payment sheet but it does not control the user's interaction with the sheet. You must decide if it makes sense to present shipping and billing information, shipping method, and other line items to the user. You should only request the information necessary to process the transaction.

Refer to the iOS Human Interface Guidelines for detail on integrating the Apple Pay button and and customizing the Apple Pay sheet for your app.




## The Payment Token

Once authorized by the user with Touch ID, your app receives a payment token from PassKit. The payment token encapsulates the information needed to complete a payment transaction, including the device-specific account number, the amount, and a unique, one-time-use cryptogram. The encrypted payment bundle can be decrypted by the merchant with the certificate private key or by the payment processor via the SDK on behalf of the merchant.

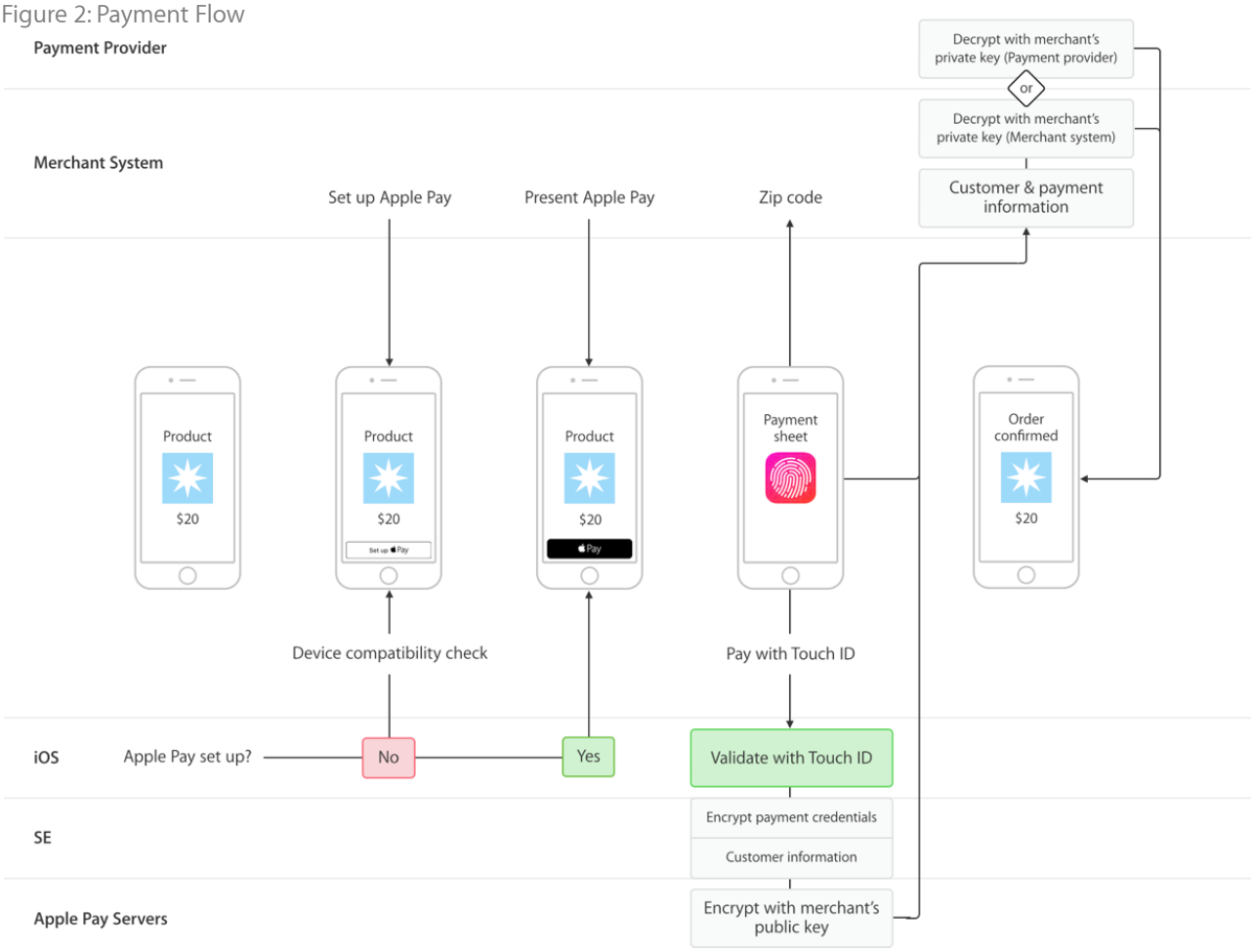
In some cases, a developer may not be set up as the merchant of record and process the payment. In this case, the developer can decrypt the payment token and pass the decrypted token downstream to the appropriate merchant for processing with its respective processor. The app must make it clear that they are an intermediary party. For more details, refer to the iOS Human Interface Guidelines.

Figure 2 illustrates a typical payment flow. First the app checks that it can offer Apple Pay as a payment method. In this example, the app needs the postal code from the selected shipping address to calculate shipping cost and update the total amount due. When the user authorizes payment, your app receives a payment token from the Secure Element, via PassKit.

Finally the app calls appropriate APIs in the payment processor SDK to pass the payment information to the payment processor, they process the transaction.

## Supported Transaction Types

Figure 2: Payment Flow




Payment Processor SDKs that support Apple Pay are required to handle the following kinds of eCommerce transactions.

| | | |
|---|---|---|
| Authorizations | Payment processor | To reserve funds on a customer's account |
| Capture | Payment processor | To transfer money to your bank account once an order is successfully completed |
| Partial shipment | Payment processor | To divide a purchase into multiple payments for goods that are not shipped together |
| Recurring | Payment processor - but should be clearly labeled in app | To handle repeating payments for services like a monthly gym membership |
| Refunds | Payment processor | To return money to a customer's account |

| | | |
|---|---|---|
| Chargeback | Payment processor | To handle fraudulent or disputed transactions |

**Best Practices**
Review and follow guidance from the iOS Human Interface Guidelines as well as from the WWDC session Apple Pay Within Apps. In addition, follow these best practices to see the best results within your app.

**Do not require registration.**
Unless necessary, do not require registration for purchases to be made with Apple Pay. You can get necessary contact information when the purchase is made, and customers are less likely to abandon the purchase experience when there is as little friction as possible.

**Add the Apple Pay button to the product detail page as well as cart checkout.**
Place an Apple Pay button on your product detail pages in addition to your 'Add to Cart' button so your customers can find what they want to buy and check out immediately. Make sure to also add Apple Pay as a payment option with your cart checkout.

**Display the Apple Pay button prominently, or set the default payment method to Apple Pay for customers that have it set up.**
Identify if the customer has Apple Pay set up, and if they do then provide that as the primary payment option.

**Do not ask for any information outside of the Apple Pay sheet that is available on the Apple Pay sheet.**
The Apple Pay sheet provides shipping address, billing address, phone number, email, and name. The customer can also choose their shipping method from right within the sheet. Requesting these fields outside of the sheet adds friction which will cause purchase abandonment.

## Common Questions and Answers

**Which Payment Providers support this service?**
For a list of payment provider please visit https://developer.apple.com/apple-pay/

**Which payment networks are supported?**
Visa, MasterCard, and American Express. Discover support for Apple Pay in apps will be coming soon.

**Which card types are supported?**
Both credit and debit cards from the major issuing banks are supported. Store credit and debit cards are also supported.

**Are there additional fees to accept Apple Pay?**

Apple does not charge users, merchants or developers to use Apple Pay for payments.

**Can I integrate Apple Pay into my app that has a web checkout experience?**
Apple Pay works with both native and hybrid apps. For those merchants leveraging a web-based checkout experience, please use a WebKit bridge to allow data to be passed from web content to native APIs and vice versa.

**Can I make an Apple Pay purchase without knowing the final amount?**
For customers with iOS 9 and later you can set the amount to "Pending". For iOS versions prior to 9, you should charge a base amount with and note within the PAY line of the Apple Pay sheet that the final amount is pending.

Copyright © 2015 Apple Inc. All rights reserved.