What's New with Wallet & Apple Pay

Session 704

Nick Shearer iOS Apps and Frameworks
Leo Cantelmo Watch SW Engineer
Overview
Overview

- Update from Wallet
- WatchKit
- Extensions
- Testing Apple Pay
Update from Wallet

What's New with PassKit
Wallet
Not just Apple Pay
Wallet

Not just Apple Pay

There's no better time to start using passes
Wallet

Not just Apple Pay

There's no better time to start using passes
More users are engaging with Wallet than ever
Passes

Recap

Wallet - The home for Apple Pay and more

WWDC 2015
Passes

Recap

Gift cards, boarding passes, rewards cards, event tickets, membership cards, and more…
Passes

Recap

Gift cards, boarding passes, rewards cards, event tickets, membership cards, and more…

Distribute passes from app, e-mail, SMS, web link, or QR code

Wallet - The home for Apple Pay and more

WWDC 2015
Recap

Gift cards, boarding passes, rewards cards, event tickets, membership cards, and more…

Distribute passes from app, e-mail, SMS, web link, or QR code

Update your passes remotely through web services
Passes
App placement
Passes

App placement

App icons and shortcuts now have front placement
Passes
App placement

App icons and shortcuts now have front placement
Deep link directly into your app, or the App Store
Passes
App placement

App icons and shortcuts now have front placement
Deep link directly into your app, or the App Store
Great for gift card top-up
Passes
Value-added services
Passes

Value-added services

Passes can transmit secure information over NFC
Passes

Value-added services

Passes can transmit secure information over NFC

Support coming from many PoS systems
Passes

Value-added services
Passes

Value-added services

"One Tap" experience
Passes

Value-added services

"One Tap" experience

Deeply integrated into the purchase process
Passes

Value-added services

"One Tap" experience
Deeply integrated into the purchase process
Available on iPhone and Apple Watch
Passes
Value-added services
Passes

Value-added services

Distribute passes over NFC
Passes

Value-added services

Distribute passes over NFC

Sign up for loyalty programs directly from Wallet
Passes

Value-added services

Distribute passes over NFC
Sign up for loyalty programs directly from Wallet
Share and personalize passes quickly and easily
Passes
VAS and personalization
Passes

VAS and personalization

Passes must be signed using a NFC signing certificate
Passes

VAS and personalization

Passes must be signed using a NFC signing certificate.
Contact us to find out more about using VAS services.
Passes

VAS and personalization

Passes must be signed using a NFC signing certificate
Contact us to find out more about using VAS services
VAS engineers available in the labs
Card Issuers
Card Issuers

Wallet offers unique features for card issuers
Card Issuers

Wallet offers unique features for card issuers
Store credit and debit cards also supported
Card Issuers

App provisioning

Add to Apple Wallet

Add to Apple Wallet

Ajouter à Apple Wallet
Card Issuers
App provisioning

Set up cards in Apple Pay directly from your app

- Add to Apple Wallet
- 添加到 Apple Wallet
- Ajouter à Apple Wallet
Card Issuers

App provisioning

Set up cards in Apple Pay directly from your app

Available to any existing Apple Pay issuer
Card Issuers

App provisioning

Set up cards in Apple Pay directly from your app

Available to any existing Apple Pay issuer

Visit the labs or contact us for more information
Card Issuers
In-store experience
Card Issuers

In-store experience

Present your cards directly from your apps
Card Issuers

In-store experience

Present your cards directly from your apps

Redeem a coupon in your app, and immediately present a payment method
Card Issuers

In-store experience

Present your cards directly from your apps

Redeem a coupon in your app, and immediately present a payment method

```swift
let passLibrary = PKPassLibrary.init()
passLibrary.present(pass: myPaymentPass)
```
Card Issuers
In-store experience
Card Issuers

In-store experience

Use the new PKPaymentButton style for consistent branding
Card Issuers

In-store experience

Use the new PKPaymentButton style for consistent branding

let presentButton = PKPaymentButton(type: .inStore, style: .black)
Wallet

Store Credit and Debit
Wallet

Store Credit and Debit

Associate your app with your store or co-brand card
Wallet

Store Credit and Debit

Associate your app with your store or co-brand card

Default to your card when paying over NFC or within app
Wallet

Store Credit and Debit

Associate your app with your store or co-brand card

Default to your card when paying over NFC or within app

No API required—built into the card itself
Apple Pay
Apple Pay

An easy, secure, and private way to pay
Apple Pay

An easy, secure, and private way to pay
Pay in-store and within apps
Apple Pay

An easy, secure, and private way to pay
Pay in-store and within apps
Amazing customer satisfaction
Apple Pay
Within apps
Apple Pay

Within apps

Pay using Apple Pay directly from apps
Apple Pay
Within apps

Pay using Apple Pay directly from apps
Thousands of apps have already adopted
Apple Pay
Within apps

Pay using Apple Pay directly from apps
Thousands of apps have already adopted
Millions of users using Apple Pay
Apple Pay

Within apps

Pay using Apple Pay directly from apps
Thousands of apps have already adopted
Millions of users using Apple Pay
Incredible growth
“With Apple Pay, our conversion rate has tripled.”

Chairish
“Apple Pay drives twice as many new users to StubHub as any other payment method.”
China

Apple Pay 入门
China

Apple Pay 入门

Full support for China UnionPay credit and debit cards
China

Apple Pay 入门

Full support for China UnionPay credit and debit cards

Accepted by CUP, PayEase, LianlianPay, YeePay, and UMS
China

Apple Pay 入门

Full support for China UnionPay credit and debit cards
Accepted by CUP, PayEase, LianlianPay, YeePay, and UMS

What’s New
Dynamic networks and proxies
What’s New
Dynamic networks and proxies

request.supportedNetworks = ["Visa", "Mastercard", "AmEx", "Discover"]
Dynamic networks and proxies

Today, supported payment networks are hard-coded into your app

```javascript
request.supportedNetworks = ["Visa", "Mastercard", "AmEx", "Discover"]
```
What’s New
Dynamic networks and proxies

```javascript
request.supportedNetworks = ["Visa", "Mastercard", "AmEx", "Discover"]
```

Today, supported payment networks are hard-coded into your app.
Adding new networks requires you to perform SDK availability checks.
What’s New
Dynamic networks and proxies

```javascript
request.supportedNetworks = ['Visa', 'Mastercard', 'AmEx', 'Discover']
```

Today, supported payment networks are hard-coded into your app
Adding new networks requires you to perform SDK availability checks
Updating networks often means updating your app
What’s New
Dynamic networks and proxies

// Discover all supported networks available on this device
PKPaymentRequest.availableNetworks

// Set a payment processor as a supported network
// This maps to a subset of available networks and is dynamically updated
request.supportedNetworks = ["My Payment Processor"]
What’s New

Dynamic networks and proxies

// Discover all supported networks available on this device
PKPaymentRequest.availableNetworks

// Set a payment processor as a supported network
// This maps to a subset of available networks and is dynamically updated
request.supportedNetworks = ["My Payment Processor"]

Enable new networks as they're added
What’s New

Dynamic networks and proxies

// Discover all supported networks available on this device
PKPaymentRequest.availableNetworks

// Set a payment processor as a supported network
// This maps to a subset of available networks and is dynamically updated
request.supportedNetworks = ["My Payment Processor"]

Enable new networks as they're added
Use a payment processor as a supported network
What’s New
Dynamic networks and proxies

// Discover all supported networks available on this device
PKPaymentRequest.availableNetworks

// Set a payment processor as a supported network
// This maps to a subset of available networks and is dynamically updated
request.supportedNetworks = ["My Payment Processor"]

Enable new networks as they're added
Use a payment processor as a supported network
Gain support for new networks as they're added, without needing to change your code
What’s New
Swift 3 API
What’s New

Swift 3 API

Improved API for Swift users
What’s New
Swift 3 API

Improved API for Swift users
PassKit now uses stringly typed enumerations
Apple Pay
Everywhere
Apple Pay
Everywhere

WatchKit
Apple Pay
Everywhere

WatchKit

Extensions
Apple Pay Everywhere

WatchKit

Extensions

Safari
Apple Pay in WatchKit

Leo Cantelmo Watch SW Engineer
Apple Pay + Apple Watch
Apple Pay—WatchKit

Overview
Apple Pay—WatchKit

Overview

Quick recap
Overview

Quick recap
Creating a payment request
Apple Pay—WatchKit

Overview

Quick recap
Creating a payment request
Presenting the payment sheet
Apple Pay—WatchKit

Overview

Quick recap
Creating a payment request
Presenting the payment sheet
Demo
Apple Pay—WatchKit

Overview

Quick recap
Creating a payment request
Presenting the payment sheet
Demo
Design considerations
<table>
<thead>
<tr>
<th>CARD</th>
<th>CHASE FREEDOM (**** 5555)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHIPPING</td>
<td>JOHN APPLESEED</td>
</tr>
<tr>
<td></td>
<td>1035 STEVENS CREEK BLVD</td>
</tr>
<tr>
<td></td>
<td>CUPERTINO CA 95104</td>
</tr>
<tr>
<td></td>
<td>UNITED STATES</td>
</tr>
<tr>
<td>METHOD</td>
<td>FREE SHIPPING</td>
</tr>
<tr>
<td></td>
<td>DELIVERS: JUNE 15</td>
</tr>
<tr>
<td>CONTACT</td>
<td><a href="mailto:J.APPLESEED@ICLOUD.COM">J.APPLESEED@ICLOUD.COM</a></td>
</tr>
<tr>
<td>SUBTOTAL</td>
<td>$20.00</td>
</tr>
<tr>
<td>FREE SHIPPING</td>
<td>$0.00</td>
</tr>
<tr>
<td>ESTIMATED TAX</td>
<td>$1.75</td>
</tr>
<tr>
<td>PAY EMPORIUM</td>
<td>$21.75</td>
</tr>
</tbody>
</table>

Pay with Touch ID
Emporium

Apple Pay

Card: CHASE FREEDOM (**** 5555)
Shipping: JOHN APPLESEED
1035 STEVENS CREEK BLVD
CUPERTINO CA 95104
UNITED STATES
Method: FREE SHIPPING
Delivers: JUNE 15
Contact: J.APPLESEED@ICLOUD.COM

Subtotal: $20.00
Free Shipping: $0.00
Estimated Tax: $1.75
Pay Emporium: $21.75

Pay with Touch ID
Quick Recap

How payments work
Quick Recap
How payments work

Payment Request

Payment Authorization Controller
Quick Recap
How payments work

Payment Request

Delegate

Payment Authorization Controller
Quick Recap
How payments work

Payment Authorization Controller

Delegate
Secure Element
Apple Servers
Quick Recap

How payments work

Flowchart:
- Payment Request
  - Payment Authorization Controller
    - Secure Element
    - Apple Servers
    - Payment Token
  - Delegate
Before You Start

Configuring your environment
Before You Start

Configuring your environment

Register a merchant identifier and set up cryptographic keys in the Developer Portal
Before You Start

Configuring your environment

Register a merchant identifier and set up cryptographic keys in the Developer Portal.
Enable Apple Pay in the Capabilities pane inside Xcode.
// Creating a Payment Request

import PassKit

let paymentRequest = PKPaymentRequest()
paymentRequest.countryCode = "US"
paymentRequest.currencyCode = "USD"
paymentRequest.merchantIdentifier = "merchant.com.example.emporium"
paymentRequest.merchantCapabilities = .capability3DS
paymentRequest.supportedNetworks = PKPaymentRequest.availableNetworks()
paymentRequest.paymentSummaryItems = [
    PKPaymentSummaryItem(label:"Emporium", amount:NSDecimalNumber(string:"10.99"))
]
// Creating a Payment Request

import PassKit

let paymentRequest = PKPaymentRequest()
paymentRequest.countryCode = "US"
paymentRequest.currencyCode = "USD"
paymentRequest.merchantIdentifier = "merchant.com.example.emporium"
paymentRequest.merchantCapabilities = .capability3DS
paymentRequest.supportedNetworks = PKPaymentRequest.availableNetworks()
paymentRequest.paymentSummaryItems = [
    PKPaymentSummaryItem(label:"Emporium", amount: NSDecimalNumber(string:"10.99"))
]
// Creating a Payment Request

```swift
import PassKit

let paymentRequest = PKPaymentRequest()

paymentRequest.countryCode = "US"
paymentRequest.currencyCode = "USD"
paymentRequest.merchantIdentifier = "merchant.com.example.emporium"
paymentRequest.merchantCapabilities = .capability3DS
paymentRequest.supportedNetworks = PKPaymentRequest.availableNetworks()
paymentRequest.paymentSummaryItems = [
    PKPaymentSummaryItem(label:"Emporium", amount:NSDecimalNumber(string:"10.99"))
]
```
// Creating a Payment Request

import PassKit

let paymentRequest = PKPaymentRequest()
paymentRequest.countryCode = "US"
paymentRequest.currencyCode = "USD"
paymentRequest.merchantIdentifier = "merchant.com.example.emporium"
paymentRequest.merchantCapabilities = .capability3DS
paymentRequest.supportedNetworks = PKPaymentRequest.availableNetworks()

paymentRequest.paymentSummaryItems = [
    PKPaymentSummaryItem(label:"Emporium", amount:NSDecimalNumber(string:"10.99"))
]
// Presenting the Payment Sheet

import PassKit

let paymentController = PKPaymentAuthorizationController(paymentRequest: paymentRequest)
paymentController.delegate = self
paymentController.present { (success) in
    // ...
}

// MARK: - PKPaymentAuthorizationControllerDelegate
func paymentAuthorizationControllerDidFinish(controller: PKPaymentAuthorizationController) {
    controller.dismiss {
        // ...
    }
}
// Presenting the Payment Sheet

import PassKit

let paymentController = PKPaymentAuthorizationController(paymentRequest: paymentRequest)
paymentController.delegate = self
paymentController.present { (success) in
    // ...
}

// MARK: - PKPaymentAuthorizationControllerDelegate

func paymentAuthorizationControllerDidFinish(controller: PKPaymentAuthorizationController) {
    controller.dismiss {
        // ...
    }
}
import PassKit

let paymentController = PKPaymentAuthorizationController(paymentRequest: paymentRequest)
paymentController.delegate = self
paymentController.present { (success) in
    // ...
}

// MARK: - PKPaymentAuthorizationControllerDelegate
func paymentAuthorizationControllerDidFinish(controller: PKPaymentAuthorizationController) {
    controller.dismiss {
        // ...
    }
}
// Presenting the Payment Sheet

import PassKit

let paymentController = PKPaymentAuthorizationController(paymentRequest: paymentRequest)
paymentController.delegate = self

paymentController.present { (success) in
    // ...
}

// MARK: - PKPaymentAuthorizationControllerDelegate
func paymentAuthorizationControllerDidFinish(controller: PKPaymentAuthorizationController) {
    controller.dismiss {
        // ...
    }
}
import PassKit

let paymentController = PKPaymentAuthorizationController(paymentRequest: paymentRequest)
paymentController.delegate = self
paymentController.present { (success) in
   // ...
}

// MARK: - PKPaymentAuthorizationControllerDelegate

func paymentAuthorizationControllerDidFinish(controller: PKPaymentAuthorizationController) {
   controller.dismiss {
      //...
   }
}
Presenting the Payment Sheet
PKPaymentAuthorizationController
Presenting the Payment Sheet

PKPaymentAuthorizationController

New controller class available in PassKit.framework
Presenting the Payment Sheet
PKPaymentAuthorizationController

New controller class available in PassKit.framework
Responsible for controlling the payment authorization flow
Presenting the Payment Sheet
PKPaymentAuthorizationController

New controller class available in PassKit.framework
Responsible for controlling the payment authorization flow
Same API semantics of PKPaymentAuthorizationViewController
Presenting the Payment Sheet
PKPaymentAuthorizationController

New controller class available in PassKit.framework
Responsible for controlling the payment authorization flow
Same API semantics of PKPaymentAuthorizationViewController
Allows for presentation of the payment sheet from WatchKit extensions
Presenting the Payment Sheet
PKPaymentAuthorizationController

New controller class available in PassKit.framework
Responsible for controlling the payment authorization flow
Same API semantics of PKPaymentAuthorizationViewController
Allows for presentation of the payment sheet from WatchKit extensions
Supported across watchOS and iOS allowing for code reuse
Demo
Making payments on watchOS
Design considerations
Make shopping easy
Design considerations
Make shopping easy

Design short interactions for small screens
Design considerations
Make shopping easy

- Design short interactions for small screens
- Don’t require unnecessary information
Design considerations
Make shopping easy

Design short interactions for small screens
Don’t require unnecessary information
Billing and shipping options from iPhone
Design considerations
Make shopping easy

Design short interactions for small screens
Don’t require unnecessary information
Billing and shipping options from iPhone
Use provided WKInterfacePaymentButton
Design considerations

Make shopping easy

- Design short interactions for small screens
- Don’t require unnecessary information
- Billing and shipping options from iPhone
- Use provided WKInterfacePaymentButton
- Storyboard support in Interface Builder

Cappuccino $3.49
Buy with Apple Pay
Design considerations

Make shopping easy

Design short interactions for small screens
Don’t require unnecessary information
Billing and shipping options from iPhone
Use provided WKInterfacePaymentButton
Storyboard support in Interface Builder
Follow our Apple Pay Identity Guidelines
Apple Pay—WatchKit

Summary
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes
• Use PKPaymentAuthorizationController in your WatchKit extension
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes

• Use PKPaymentAuthorizationController in your WatchKit extension
• Use provided methods to present and dismiss the payment sheet
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes

• Use PKPaymentAuthorizationController in your WatchKit extension
• Use provided methods to present and dismiss the payment sheet

Create a great Apple Watch experience for your users
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes
• Use PKPaymentAuthorizationController in your WatchKit extension
• Use provided methods to present and dismiss the payment sheet
Create a great Apple Watch experience for your users
• Interactions are short and screens small
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes

• Use PKPaymentAuthorizationController in your WatchKit extension
• Use provided methods to present and dismiss the payment sheet

Create a great Apple Watch experience for your users

• Interactions are short and screens small
• Use WKInterfacePaymentButton following our guidelines
Apple Pay—WatchKit

Summary

Same code from iOS will work with minimal changes

• Use PKPaymentAuthorizationController in your WatchKit extension
• Use provided methods to present and dismiss the payment sheet

Create a great Apple Watch experience for your users

• Interactions are short and screens small
• Use WKInterfacePaymentButton following our guidelines
Extensions

Messages, Siri, Maps, and more…
Extension Support

Previous releases
Extension Support

Previous releases

Apple Pay has been difficult to support in extensions
Extension Support

Previous releases

Apple Pay has been difficult to support in extensions
Not many interesting places to use it
Extension Support

New opportunities
Extension Support

New opportunities

New extensions in iOS 10 offer many new Apple Pay opportunities
Extension Support

New opportunities

New extensions in iOS 10 offer many new Apple Pay opportunities.

New PKPaymentAuthorizationController API allows presentation in non-UI contexts.
Extension Support

Messages

Split items and purchases
Extension Support

Messages

- Split items and purchases
- Send a gift to a friend
Messages

- Split items and purchases
- Send a gift to a friend
- Organize outings
Extension Support

Intents—Siri and Maps
Extension Support

Intents—Siri and Maps

Get a ride or cab from Maps and Siri
Extension Support

Intents—Siri and Maps

Get a ride or cab from Maps and Siri
Pay directly from the extension
Extension Support

Intents—Siri and Maps

Get a ride or cab from Maps and Siri
Pay directly from the extension
Extension Support

Using Apple Pay
Extension Support

Using Apple Pay

Requesting and presenting payment is identical to WatchKit
Extension Support

Using Apple Pay

Requesting and presenting payment is identical to WatchKit

Use the new PKPaymentAuthorizationController in both UI and non-UI extensions
Extension Support

Using Apple Pay

Requesting and presenting payment is identical to WatchKit
Use the new PKPaymentAuthorizationController in both UI and non-UI extensions
Share your payment code between app and extension
Apple Pay
Sample code
Apple Pay
Sample code

New sample code with WatchKit and Intents samples available
Apple Pay
Sample code

New sample code with WatchKit and Intents samples available
Simplified to show a shared Apple Pay model
Testing Apple Pay
Testing Apple Pay
The simulator
Testing Apple Pay
The simulator

Test your iOS, WatchKit, Web, and Intents in the Simulator
Testing Apple Pay
The simulator

Test your iOS, WatchKit, Web, and Intents in the Simulator
Returns dummy payment data
Testing Apple Pay

The simulator

Test your iOS, WatchKit, Web, and Intents in the Simulator
Returns dummy payment data
Useful for UI development and testing
Testing Apple Pay
The simulator
Testing Apple Pay

The simulator

Not real card data
Testing Apple Pay
The simulator

Not real card data
Not representative of real device behavior
Testing Apple Pay

The simulator

Not real card data
Not representative of real device behavior
Testing with real devices and cards not always feasible
The Apple Pay Sandbox
The Apple Pay Sandbox

A brand new testing environment for Apple Pay
The Apple Pay Sandbox

A brand new testing environment for Apple Pay
Provision test cards directly onto devices
The Apple Pay Sandbox

A brand new testing environment for Apple Pay
Provision test cards directly onto devices
Returns test encrypted payment data
The Apple Pay Sandbox
Getting set up
The Apple Pay Sandbox

Getting set up

Create a testing iCloud Account at iTunes Connect
The Apple Pay Sandbox

Getting set up

Create a testing iCloud Account at iTunes Connect
Log in to that account on your device
The Apple Pay Sandbox

Getting set up

Create a testing iCloud Account at iTunes Connect
Log in to that account on your device
Set your region
The Apple Pay Sandbox

Getting set up

Create a testing iCloud Account at iTunes Connect
Log in to that account on your device
Set your region
Use test cards at developer.apple.com
The Apple Pay Sandbox

Tips
The Apple Pay Sandbox

Tips

Environments are switched automatically when you sign in/out of iCloud.
The Apple Pay Sandbox

Tips

Environments are switched automatically when you sign in/out of iCloud
Validate your apps and sites with production cards before launching
The Apple Pay Sandbox
Support
The Apple Pay Sandbox
Support

Supports American Express, MasterCard, and Visa in Seed 1
The Apple Pay Sandbox

Support

Supports American Express, MasterCard, and Visa in Seed 1

Additional networks coming soon
Summary

New Wallet and Apple Pay API and features
Summary

New Wallet and Apple Pay API and features
Apple Pay in WatchKit
Summary

New Wallet and Apple Pay API and features
Apple Pay in WatchKit
Apple Pay in Extensions
Summary

New Wallet and Apple Pay API and features
Apple Pay in WatchKit
Apple Pay in Extensions
Testing in the Sandbox and Simulator
Apple Pay
On the web
Apple Pay
On the web

Use Apple Pay on mobile websites
Apple Pay
On the web

Use Apple Pay on mobile websites
Authorize payment on using your Apple Pay device
Apple Pay

On the web

Use Apple Pay on mobile websites
Authorize payment on using your Apple Pay device
Apple Pay
On the web

Use Apple Pay on mobile websites
Authorize payment on using your Apple Pay device
More Information

https://developer.apple.com/wwdc16/704
Apple Pay Partners

Bank and private label inquiries
• In-app provisioning and co-brand/private label features
  - apple-pay-provisioning@apple.com

Value-added service inquiries
• NFC passes and loyalty
  - https://developer.apple.com/contact/passkit/
## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple Pay on the web</td>
<td>Mission</td>
<td>Tuesday 1:40PM</td>
</tr>
<tr>
<td>Designing Great Apple Watch Experiences</td>
<td>Presidio</td>
<td>Wednesday 1:40PM</td>
</tr>
<tr>
<td>Labs</td>
<td>Frameworks Lab B</td>
<td>Tuesday 4:00PM</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Apple Pay Lab 1</td>
<td>Frameworks Lab D</td>
<td>Wednesday 9:00AM</td>
</tr>
<tr>
<td>Apple Pay Lab 2</td>
<td>Frameworks Lab D</td>
<td>Wednesday 9:00AM</td>
</tr>
</tbody>
</table>