

What's New in Cocoa Touch

Session 205

Olivier Gutknecht Senior Engineering Manager, UIKit

What's New in Cocoa Touch

What's New in Cocoa Touch

What's Not New in Cocoa Touch

Since Last Year

Since Last Year

3D Touch

Since Last Year

3D Touch

iPad Pro

Since Last Year

3D Touch

iPad Pro

Apple Pencil

Since Last Year

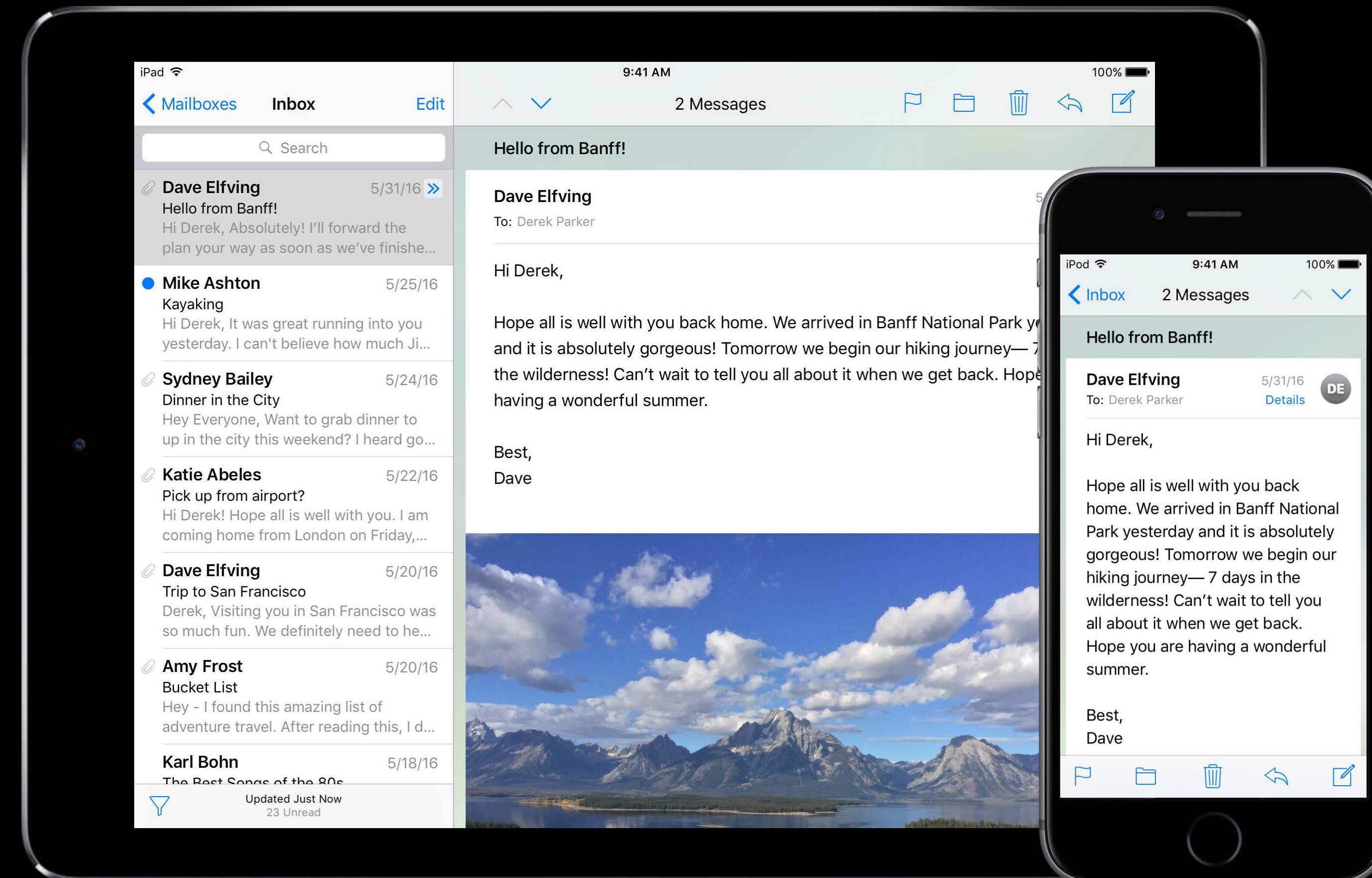
3D Touch

iPad Pro

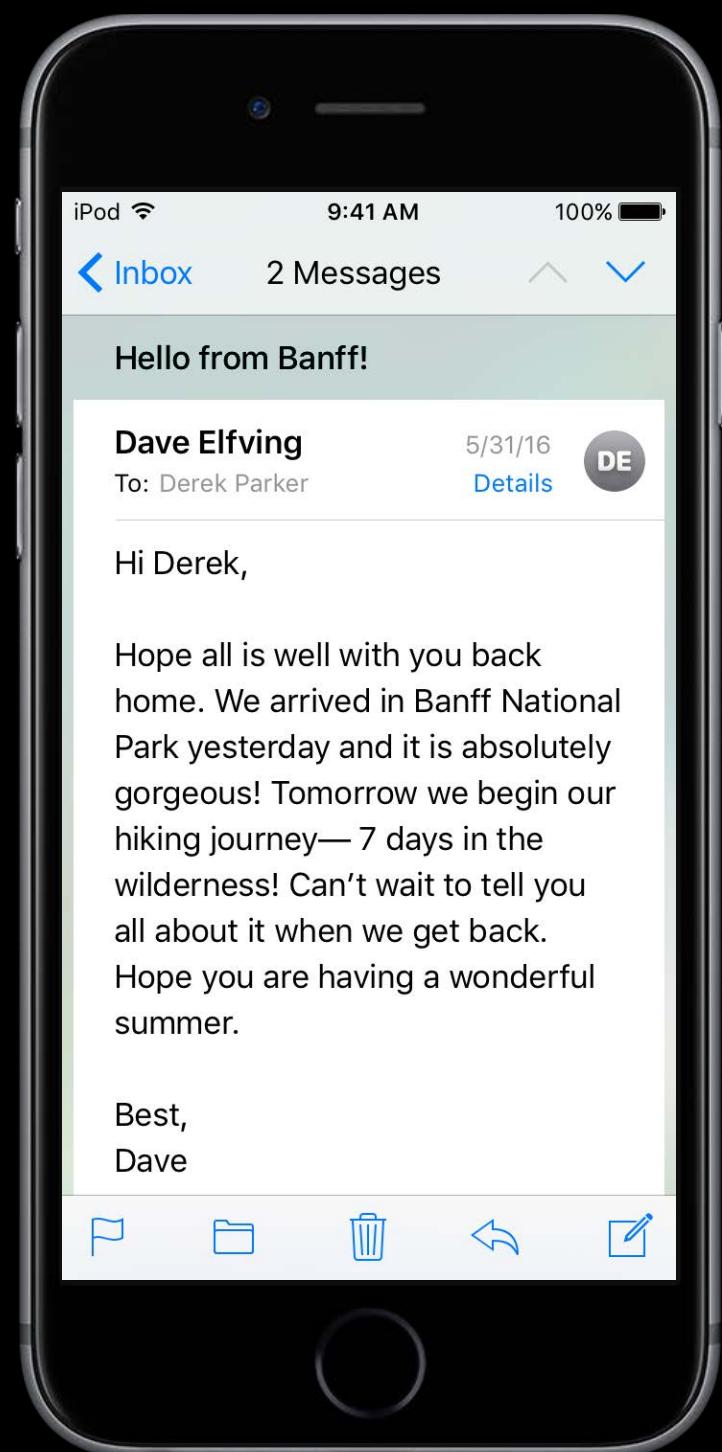
Apple Pencil

Smart Keyboard

Adaptivity

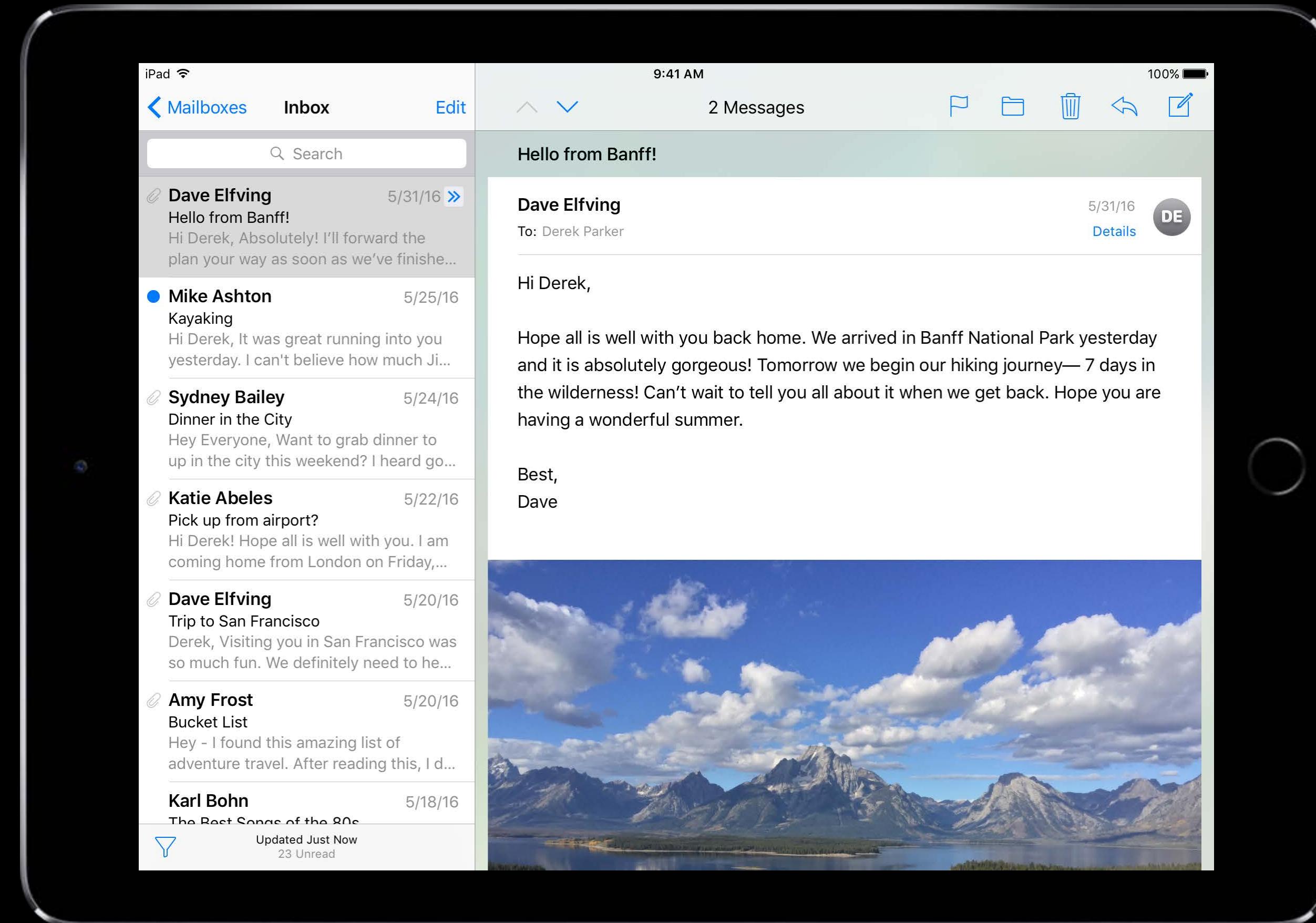


Adaptivity



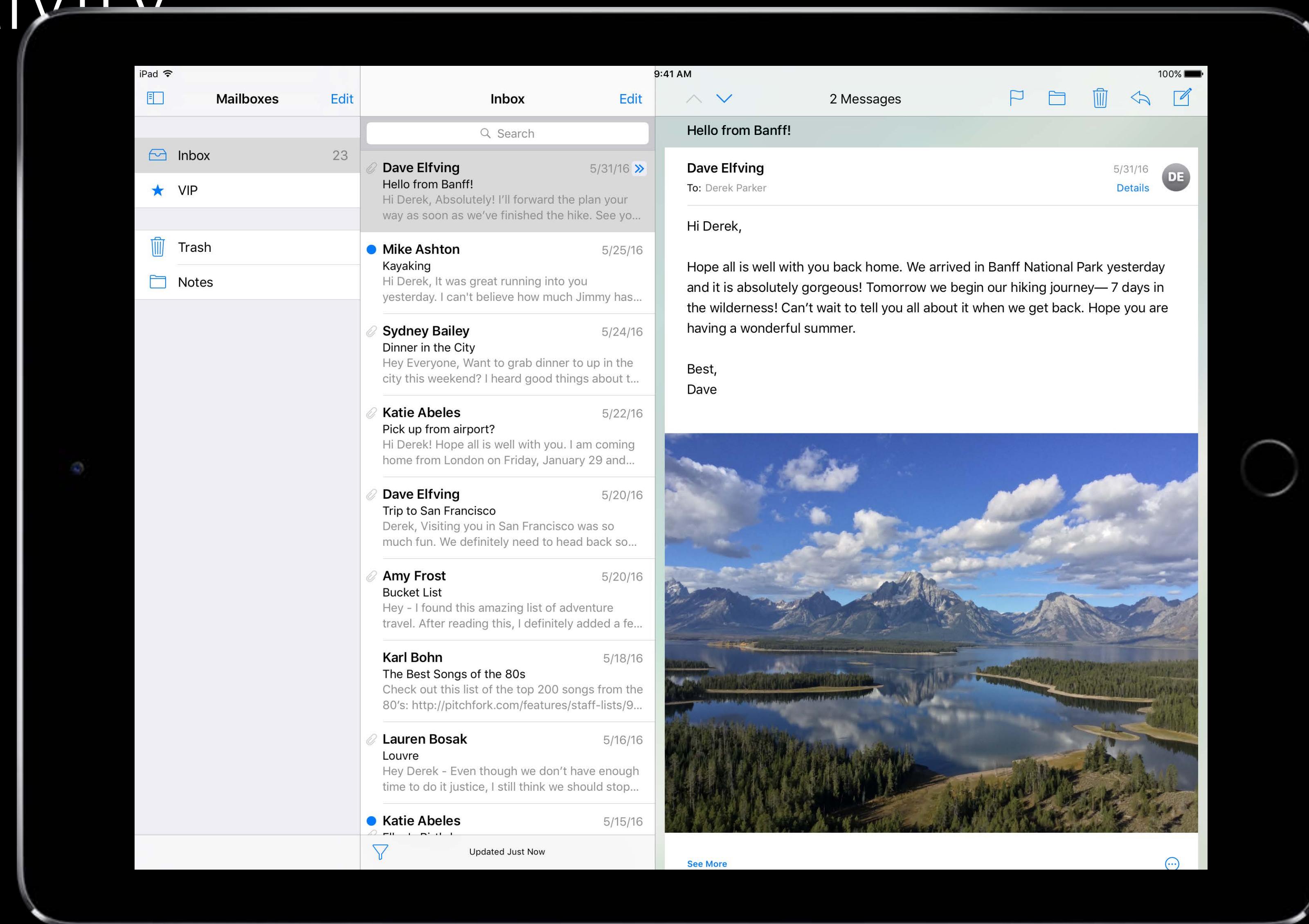
UIUserInterfaceSizeClass.compact

Adaptivity



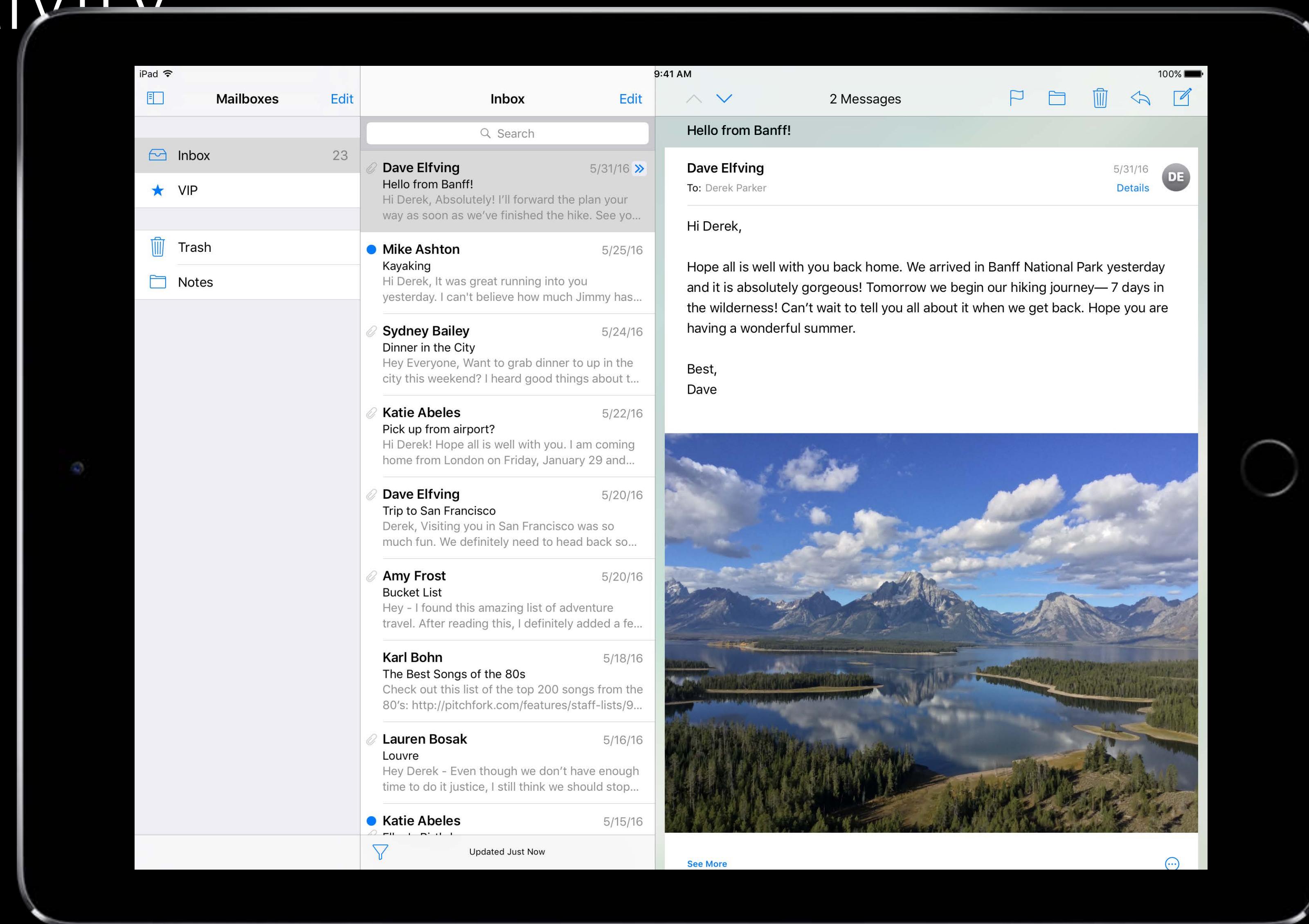
UIUserInterfaceSizeClass.regular

Adaptivity



UIUserInterfaceSizeClass.gigantic

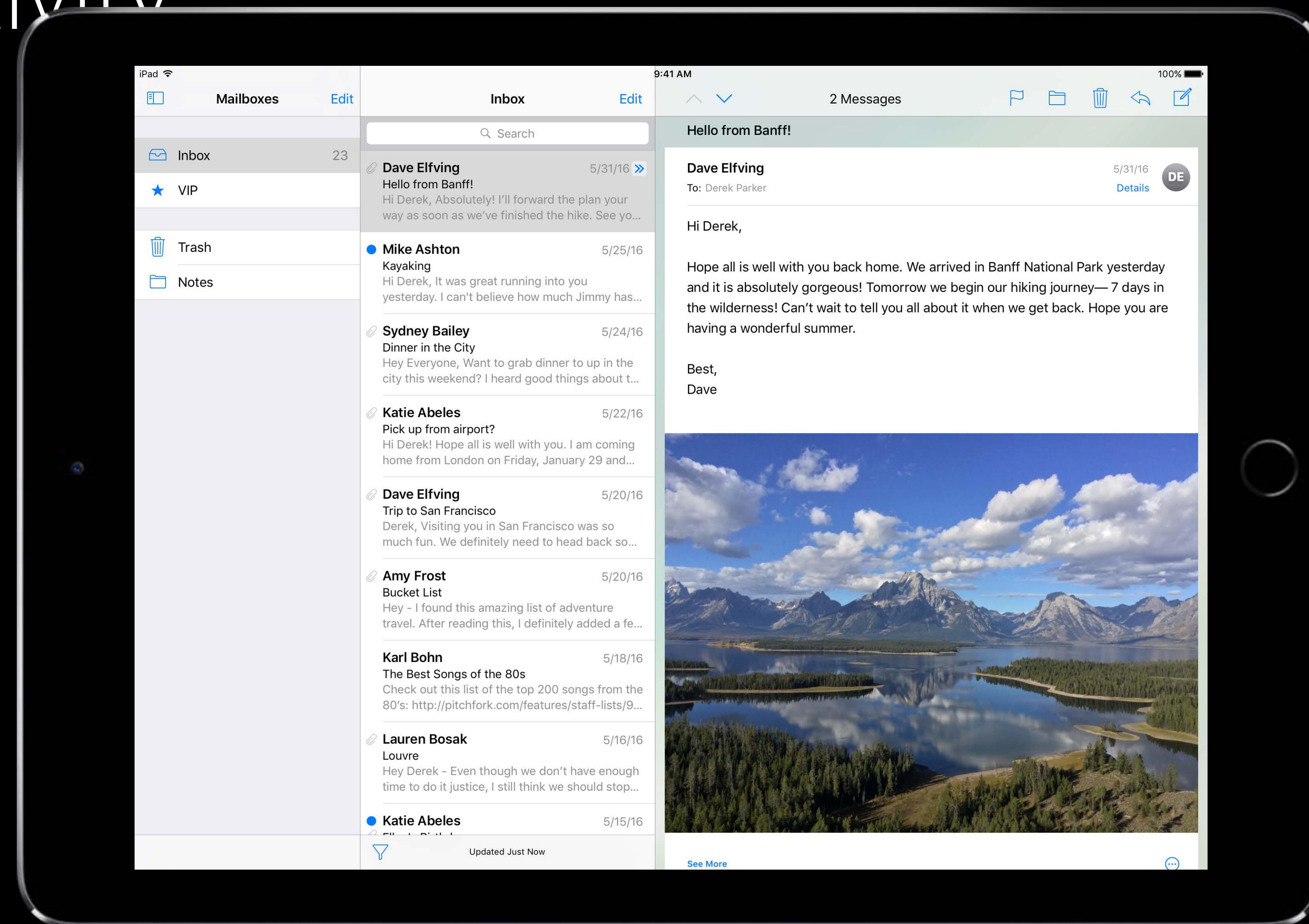
Adaptivity



UIUserInterfaceSizeClass.gigantic

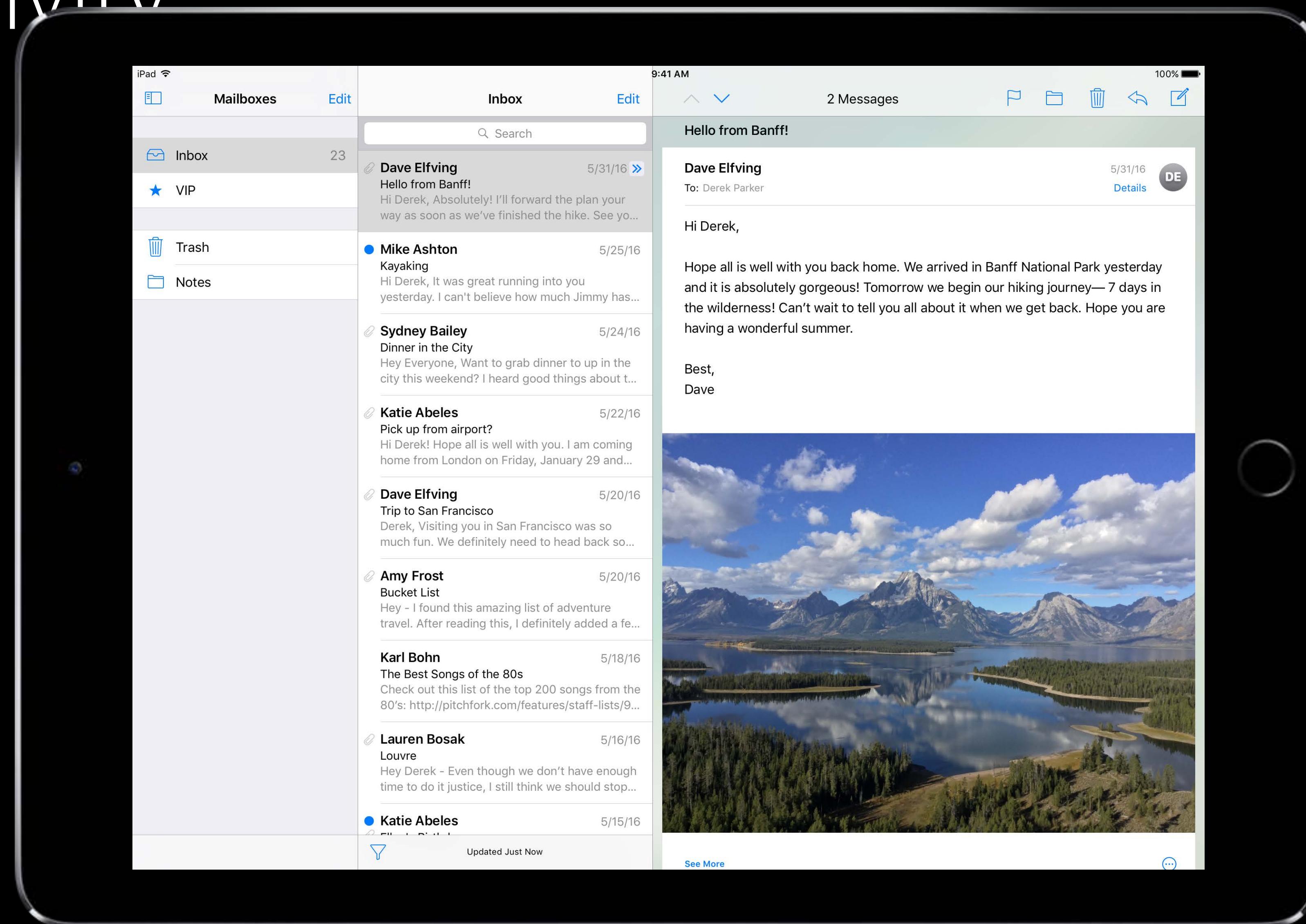


Adaptivity



UIUserInterfaceSizeClass.gigantic

Adaptivity



UIUserInterfaceSizeClass.gigantic

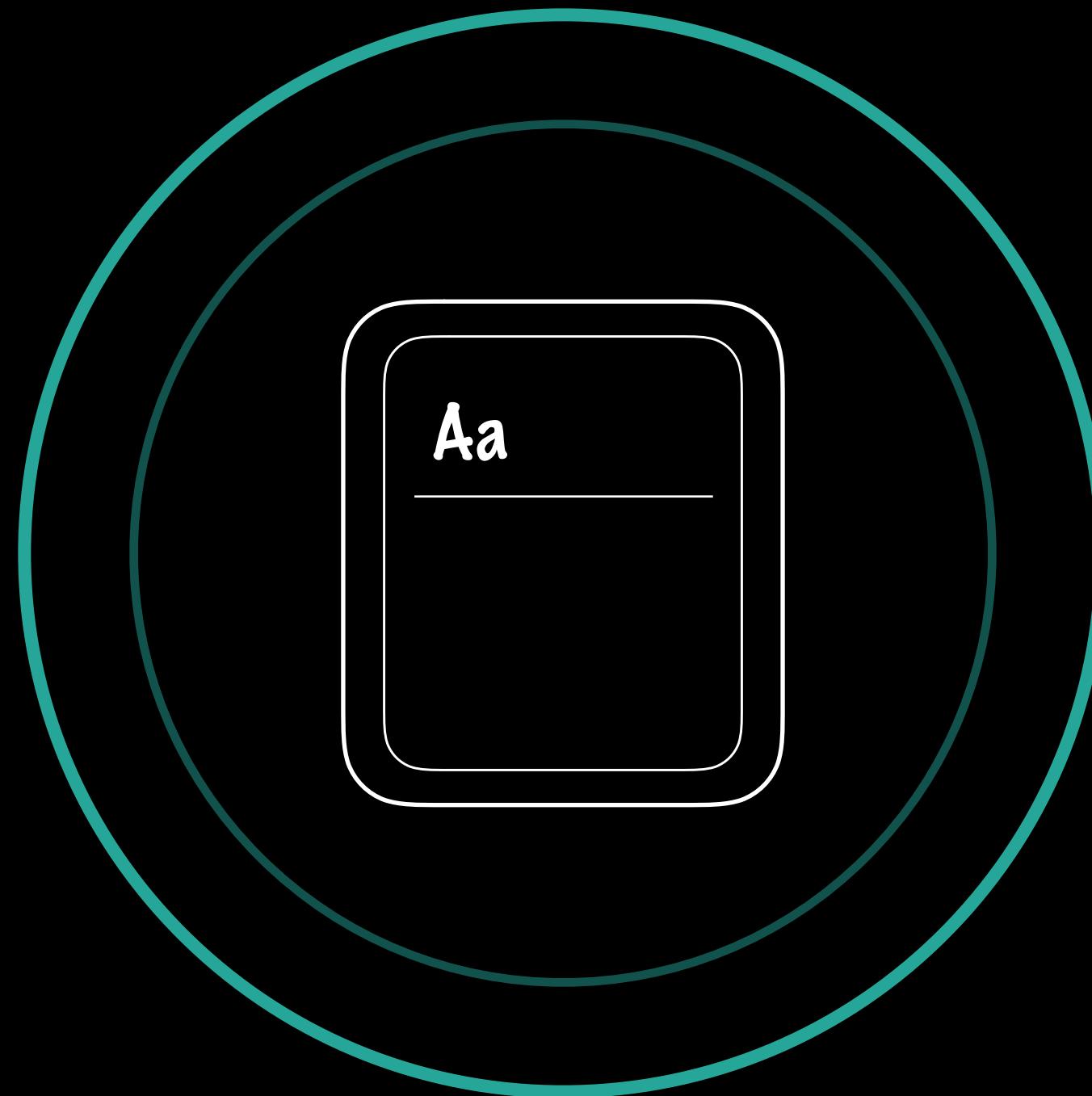
Making Apps Adaptive

The fundamentals

Traits

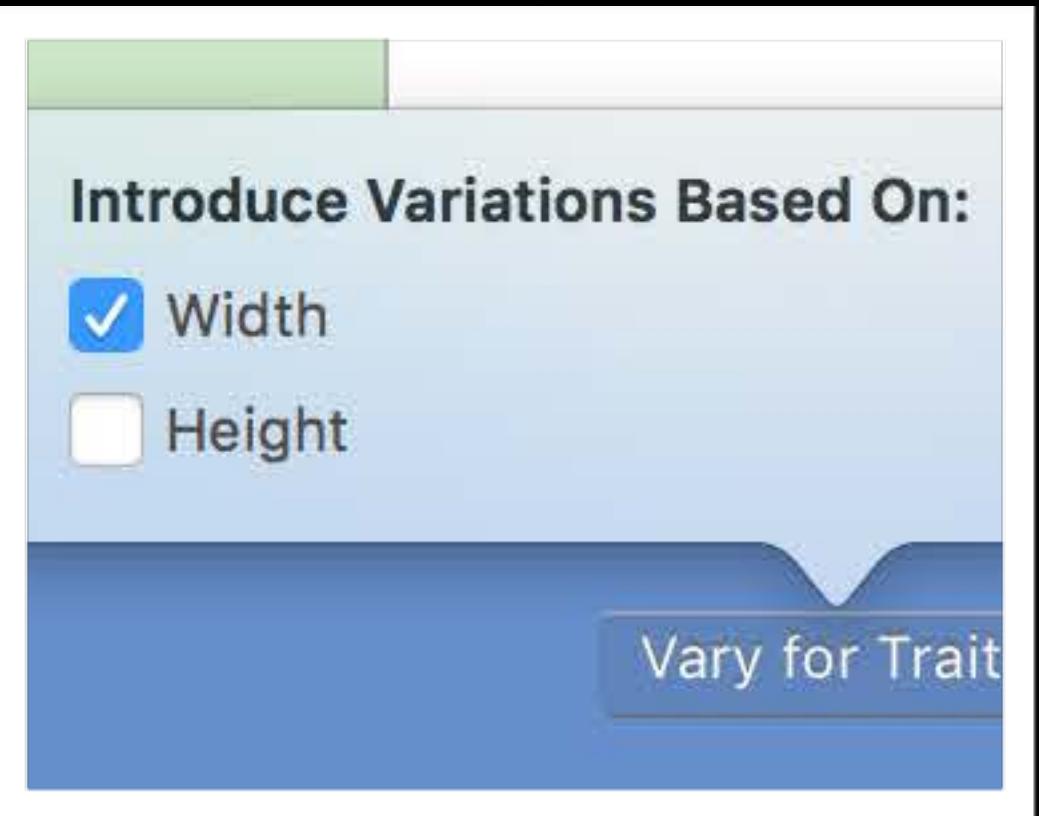
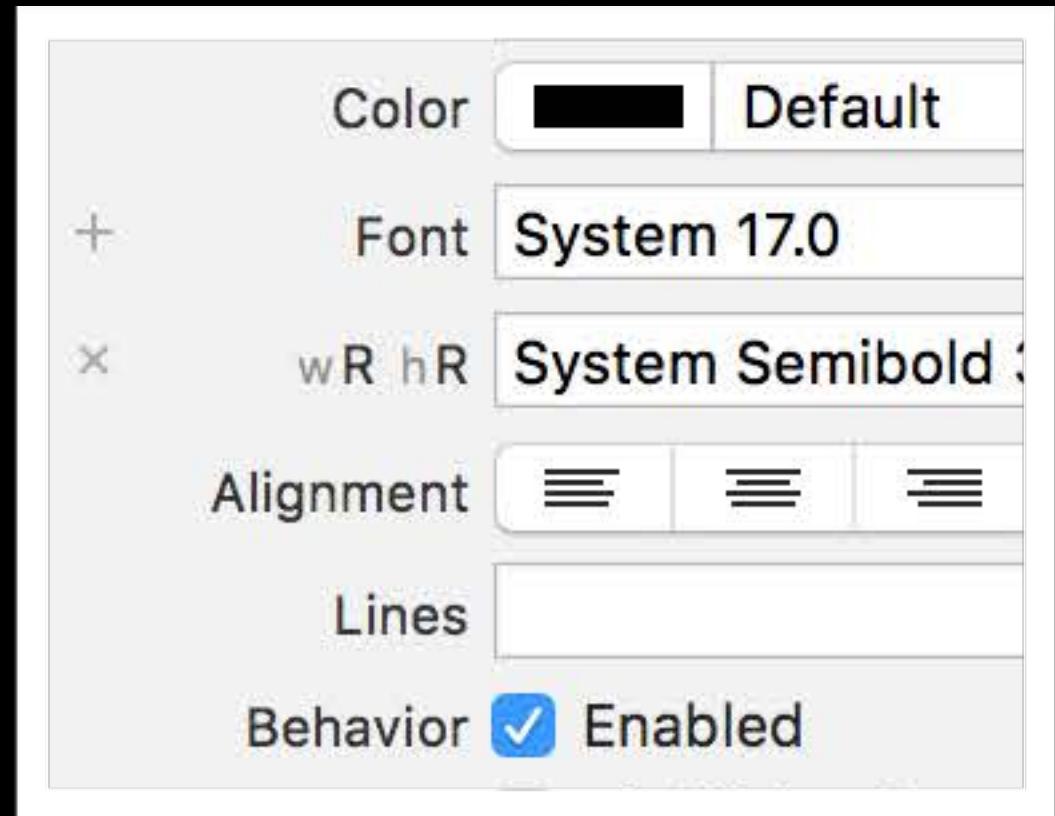
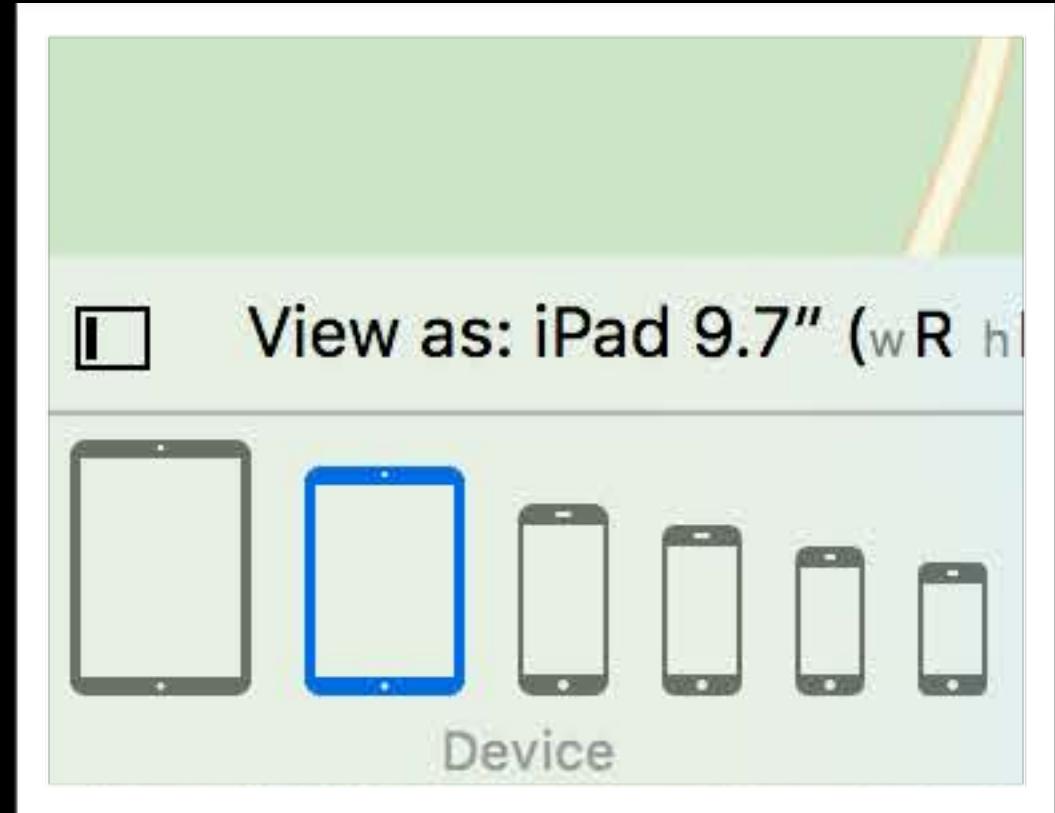
Size Classes

Size-based layouts



Making Apps Adaptive

Interface Builder



Making Apps Adaptive

Tools and Techniques

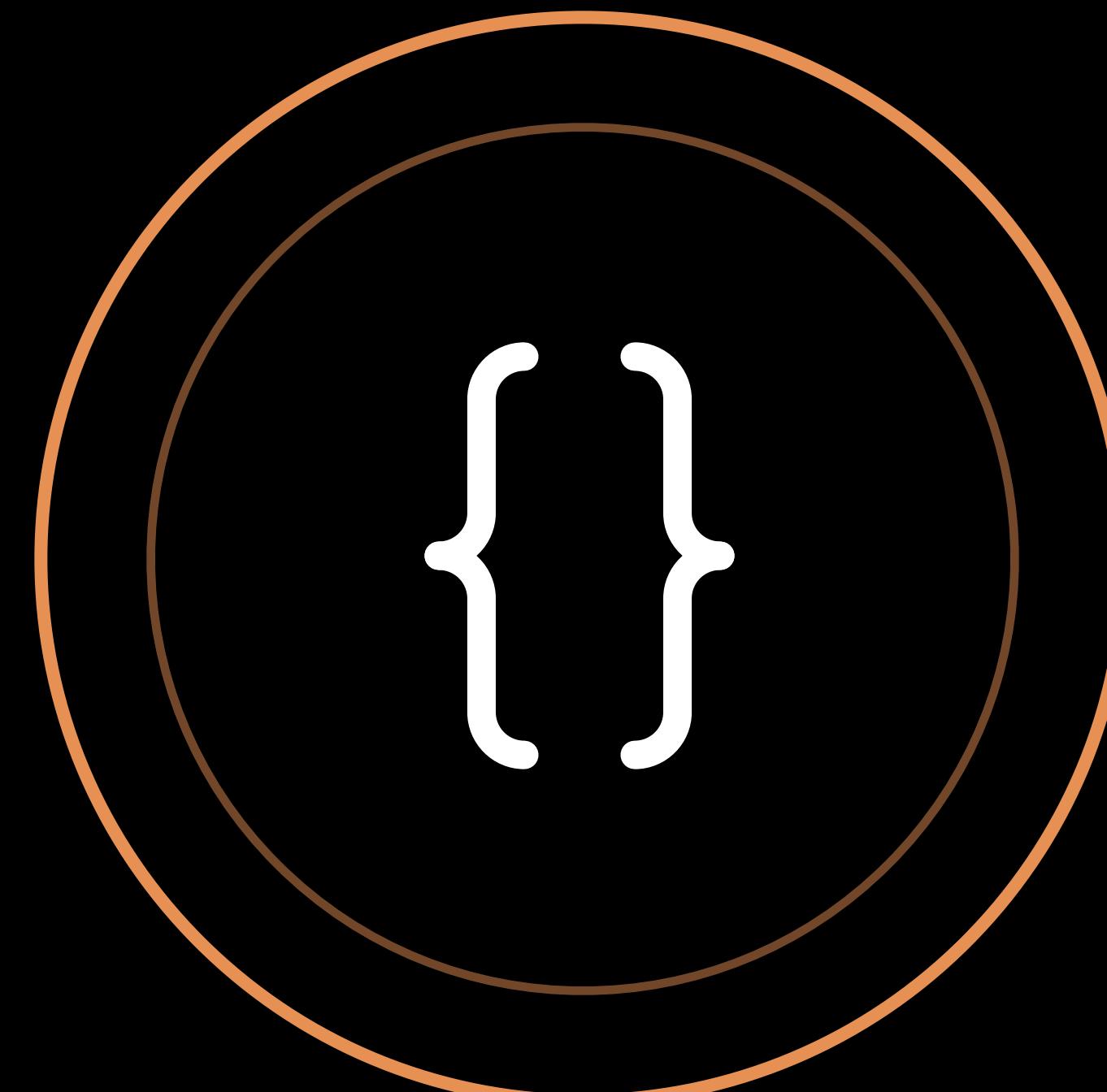
Auto Layout

Dynamic Type

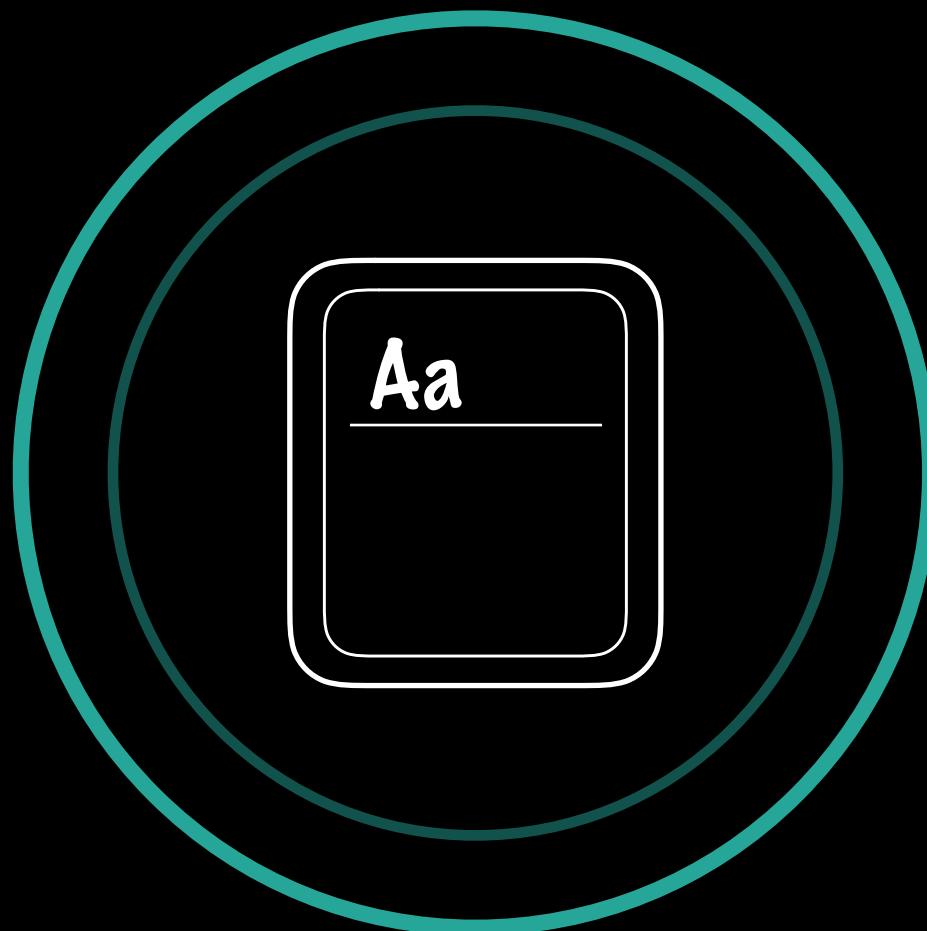
Layout Guides

UIAppearance

Asset Catalogs



Making Apps Adaptive



— PART 1 —

— PART 2 —

Making Apps Adaptive, Part I

Presidio

Thursday 11:00AM

Making Apps Adaptive, Part II

Presidio

Friday 9:00AM

Inclusive App Design

Pacific Heights

Tuesday 10:00AM

Advanced Touch Input

Advanced Touch Input

120 Hz touch scanning on
iPad Air 2 and iPad Pro



Advanced Touch Input

120 Hz touch scanning on
iPad Air 2 and iPad Pro

Orientation, Precise Location, Force, and
240 Hz scanning with Apple Pencil



Advanced Touch Input

120 Hz touch scanning on
iPad Air 2 and iPad Pro

Orientation, Precise Location, Force, and
240 Hz scanning with Apple Pencil

Force on iPhone with 3D Touch



Advanced Touch Input

120 Hz touch scanning on
iPad Air 2 and iPad Pro

Orientation, Precise Location, Force, and
240 Hz scanning with Apple Pencil

Force on iPhone with 3D Touch

New APIs in iOS 9 & iOS 9.1

Advanced Touch Input

120 Hz touch scanning on
iPad Air 2 and iPad Pro

Orientation, Precise Location, Force, and
240 Hz scanning with Apple Pencil

Force on iPhone with 3D Touch

New APIs in iOS 9 & iOS 9.1

Keyboard Support



Keyboard Support

Keyboard shortcuts



Keyboard Support

Keyboard shortcuts

Dynamic, context-sensitive



```
// Keyboard Support: UIKeyCommand

override var keyCommands: [UIKeyCommand]? {
    return [
        UIKeyCommand(input: "F",
                      modifierFlags: .Command,
                      action: #selector(ViewController.find(_:)),
                      discoverabilityTitle: "Find..."),
        UIKeyCommand(input: "N",
                      modifierFlags: [.Command, .Alternate],
                      action: #selector(ViewController.newDocument(_:)),
                      discoverabilityTitle: "New document"),
    ]
}

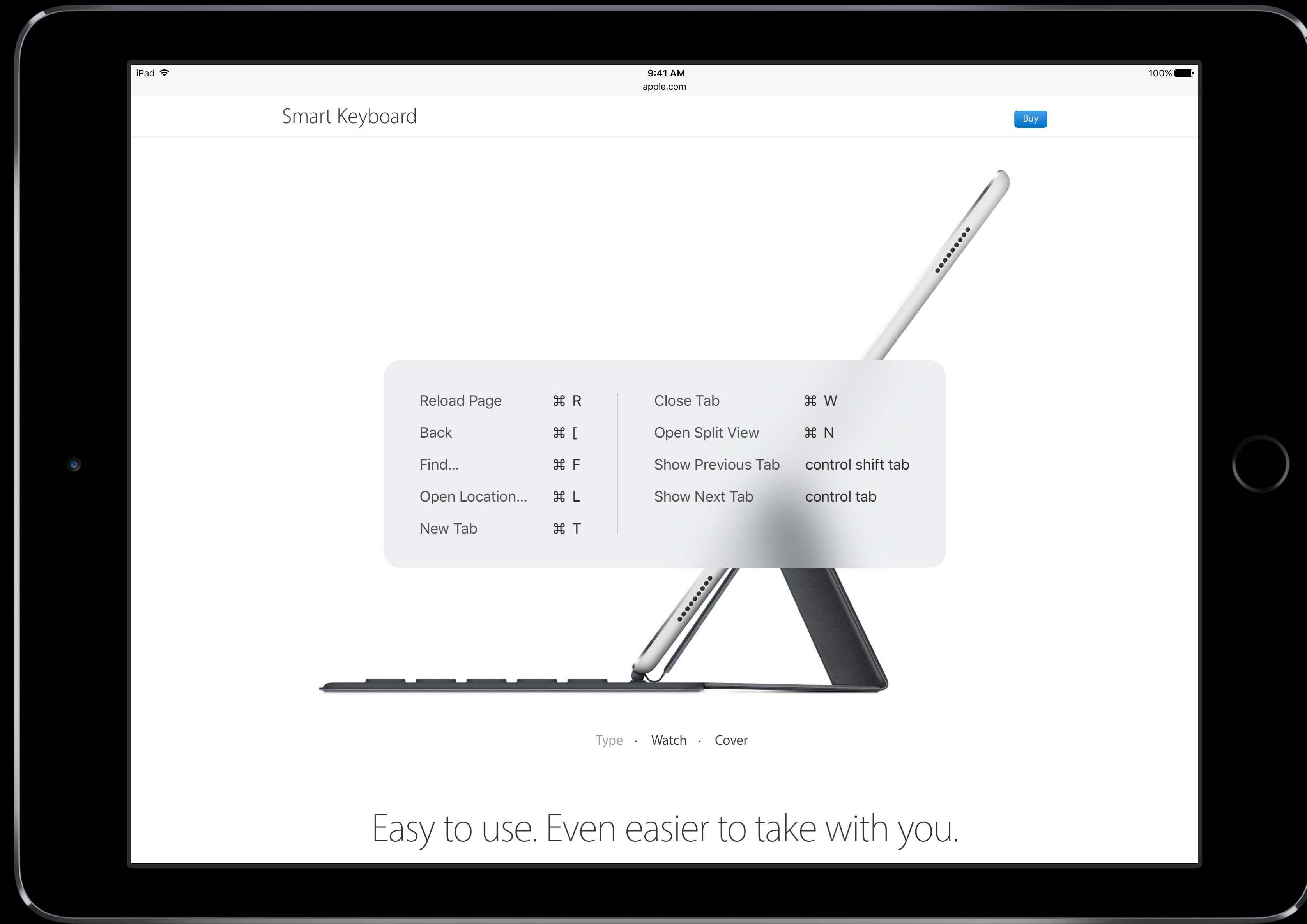
func find(sender: UIKeyCommand) {
    // ...
}
```



```
// Keyboard Support: UIKeyCommand
```

```
override var keyCommands: [UIKeyCommand]? {  
    return [  
        UIKeyCommand(input: "F",  
                     modifierFlags: .Command,  
                     action: #selector(ViewController.find(_:)),  
                     discoverabilityTitle: "Find..."),  
  
        UIKeyCommand(input: "N",  
                     modifierFlags: [.Command, .Alternate],  
                     action: #selector(ViewController.newDocument(_:)),  
                     discoverabilityTitle: "New document"),  
    ]  
}
```

```
func find(sender: UIKeyCommand) {  
    // ...  
}
```



What's Not New in Cocoa Touch

What's New in Cocoa Touch

What's New in Cocoa Touch

Agenda

Agenda

Core technologies

Agenda

Core technologies

Building better user interfaces

Agenda

Core technologies

Building better user interfaces

Adopting system features

Agenda

Core technologies

Building better user interfaces

Adopting system features

Integrating with iOS

Core Technologies



Swift 3

```
// ...  
  
let font = UIFont.preferredFontForTextStyle(UIFontTextStyleBody)  
let color = UIColor.blackColor()  
let title = content.stringByTrimmingCharactersInSet(.whitespaceAndNewlineCharacterSet())  
  
// ...  
  
let transform = CGAffineTransformRotate(baseTransform, angle)  
CGContextConcatCTM(context, transform)  
  
title.drawAtPoint(position, withAttributes: attributes)  
  
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}  
}
```

```
// ...  
  
let font = UIFont.preferredFontForTextStyle(UIFontTextStyleBody)  
let color = UIColor.blackColor()  
let title = content.stringByTrimmingCharactersInSet(.whitespaceAndNewlineCharacterSet())  
  
// ...  
  
let transform = CGAffineTransformRotate(baseTransform, angle)  
CGContextConcatCTM(context, transform)  
  
title.drawAtPoint(position, withAttributes: attributes)  
  
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.blackColor()  
let title = content.stringByTrimmingCharactersInSet(.whitespaceAndNewlineCharacterSet())  
  
// ...  
  
let transform = CGAffineTransformRotate(baseTransform, angle)  
CGContextConcatCTM(context, transform)  
  
title.draw(at: position, withAttributes: attributes)  
  
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.blackColor()  
let title = content.stringByTrimmingCharactersInSet(.whitespaceAndNewlineCharacterSet())  
  
// ...  
  
let transform = CGAffineTransformRotate(baseTransform, angle)  
CGContextConcatCTM(context, transform)  
  
title.draw(at: position, withAttributes: attributes)  
  
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.black()  
let title = content.trimmingCharacters(in: .whitespaceAndNewline())  
  
// ...  
  
let transform = CGAffineTransformRotate(baseTransform, angle)  
CGContextConcatCTM(context, transform)  
  
title.draw(at: position, withAttributes: attributes)  
  
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.black()  
let title = content.trimmingCharacters(in: .whitespaceAndNewline())
```

```
// ...  
  
let transform = CGAffineTransformRotate(baseTransform, angle)  
CGContextConcatCTM(context, transform)
```

```
title.draw(at: position, withAttributes: attributes)
```

```
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.black()  
let title = content.trimmingCharacters(in: .whitespaceAndNewline())
```

```
// ...  
  
let transform = baseTransform.rotate(angle)  
context.concatCTM(transform)
```

```
title.draw(at: position, withAttributes: attributes)
```

```
// ...  
  
let queue = dispatch_queue_create("com.example.queue", nil)  
dispatch_async(queue) {  
    // ...  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.black()  
let title = content.trimmingCharacters(in: .whitespaceAndNewline())  
  
// ...  
  
let transform = baseTransform.rotate(angle)  
context.concatCTM(transform)  
  
title.draw(at: position, withAttributes: attributes)  
  
// ...  
  
let queue = dispatch_queue_create("com.example.queue")  
dispatch_async(queue) {  
    // ...  
}  
}
```

```
// ...  
  
let font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
let color = UIColor.black()  
let title = content.trimmingCharacters(in: .whitespaceAndNewline())  
  
// ...  
  
let transform = baseTransform.rotate(angle)  
context.concatCTM(transform)  
  
title.draw(at: position, withAttributes: attributes)  
  
// ...  
  
let queue = DispatchQueue(label: "com.example.queue")  
queue.async {  
    // ...  
}
```

Grand Central Dispatch



Grand Central Dispatch

Create a private queue



Grand Central Dispatch

Create a private queue

Schedule asynchronous work items



Grand Central Dispatch

Create a private queue

Schedule asynchronous work items

GCD can automatically wrap each work item in an autorelease pool



Grand Central Dispatch

Create a private queue

Schedule asynchronous work items

GCD can automatically wrap each work item in an autorelease pool



```
let q = DispatchQueue(label: "com.example.queue", attributes: [.autoreleaseWorkItem])
```

Grand Central Dispatch

Create a private queue

Schedule asynchronous work items

GCD can automatically wrap each work item in an autorelease pool



```
let q = DispatchQueue(label: "com.example.queue", attributes: [.autoreleaseWorkItem])
```

Foundation

Foundation

Swift improvements

Foundation

Swift improvements

Units and measurements

Foundation

Swift improvements

Units and measurements

NSISO8601DateFormatter

Foundation

Swift improvements

Units and measurements

NSISO8601DateFormatter

NSDateInterval

Foundation

Swift improvements

Units and measurements

NSISO8601DateFormatter

NSDateInterval

What's New in Foundation for Swift

Mission

Tuesday 4:00PM

Measurements and Units

Presidio

Friday 4:00PM

UIPasteboard

Universal Clipboard



UIPasteboard

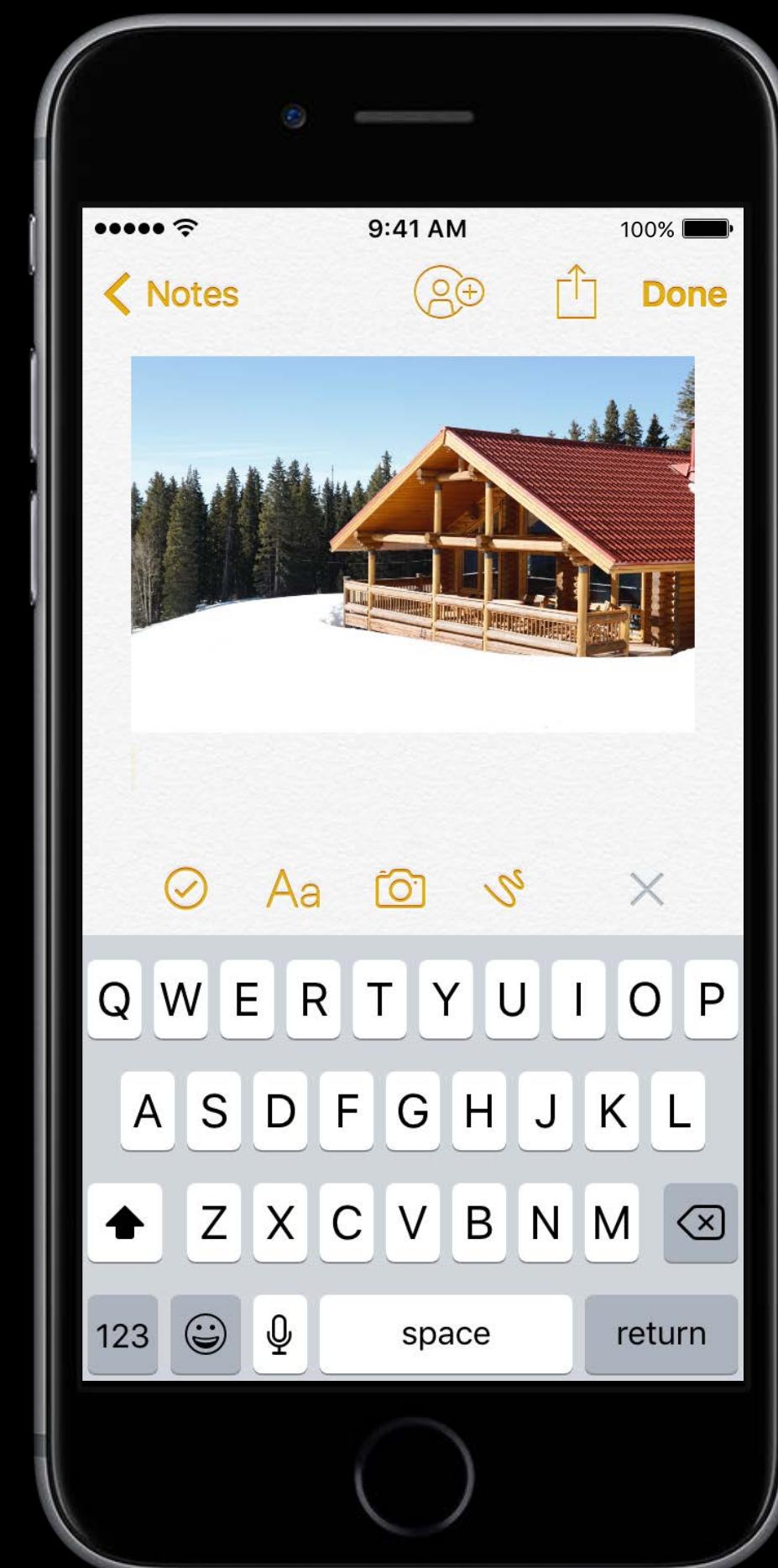
Universal Clipboard



UIPasteboard

Universal Clipboard

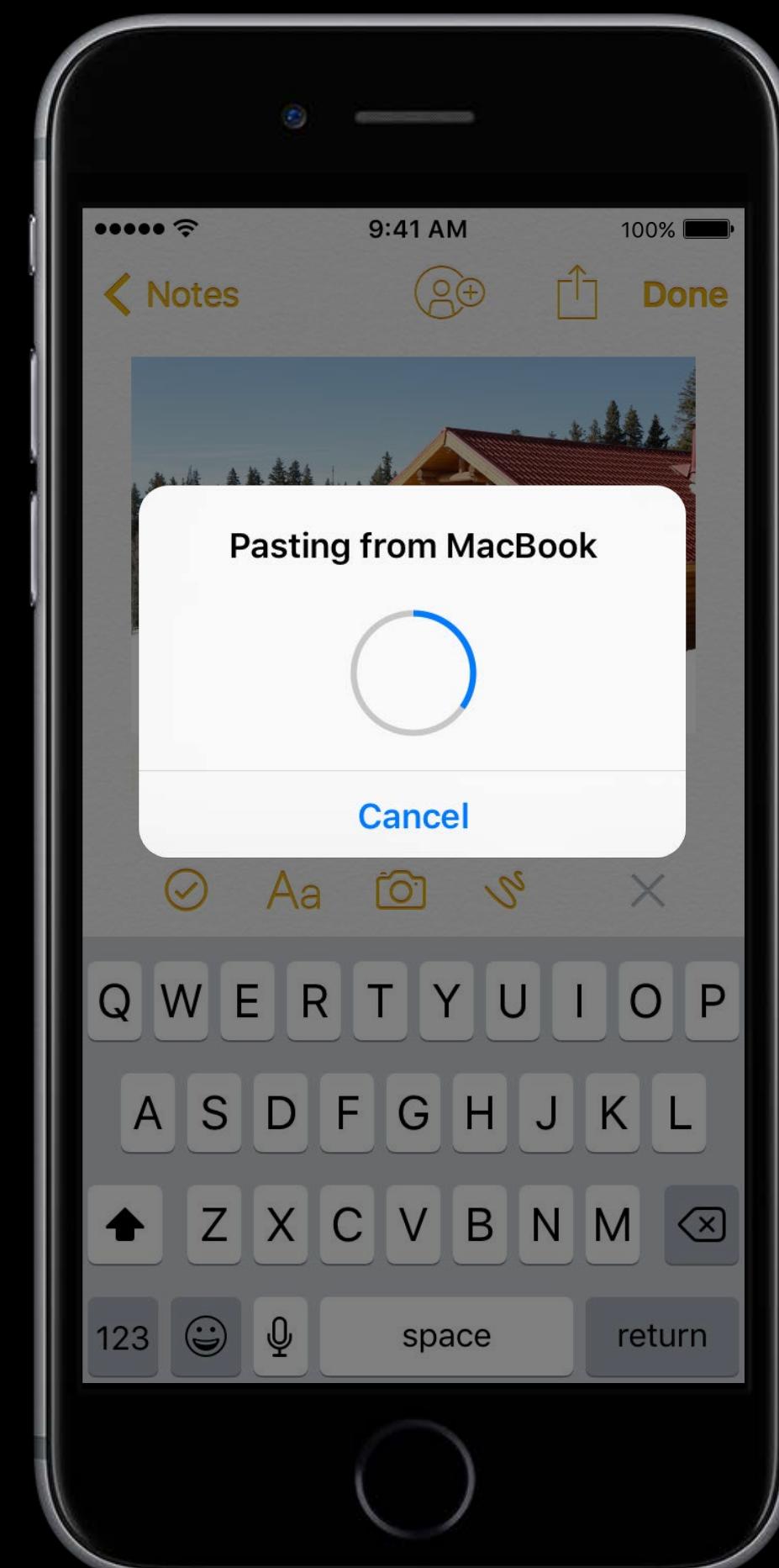
A paste operation might have to retrieve remote data



UIPasteboard

Universal Clipboard

A paste operation might have to retrieve
remote data



UIPasteboard

Universal Clipboard

A paste operation might have to retrieve
remote data

Check for Pasteboard content without fetching

UIPasteboard

Universal Clipboard

A paste operation might have to retrieve
remote data

Check for Pasteboard content without fetching

```
public class UIPasteboard : NSObject {  
  
    public var hasStrings: Bool { get }  
    public var hasURLs: Bool { get }  
    public var hasImages: Bool { get }  
    public var hasColors: Bool { get }
```

UIPasteboard

Universal Clipboard

Control what you publish

```
extension UIPasteboardOption {  
  
    public static let expirationDate: UIPasteboardOption  
    public static let localOnly: UIPasteboardOption  
  
}
```

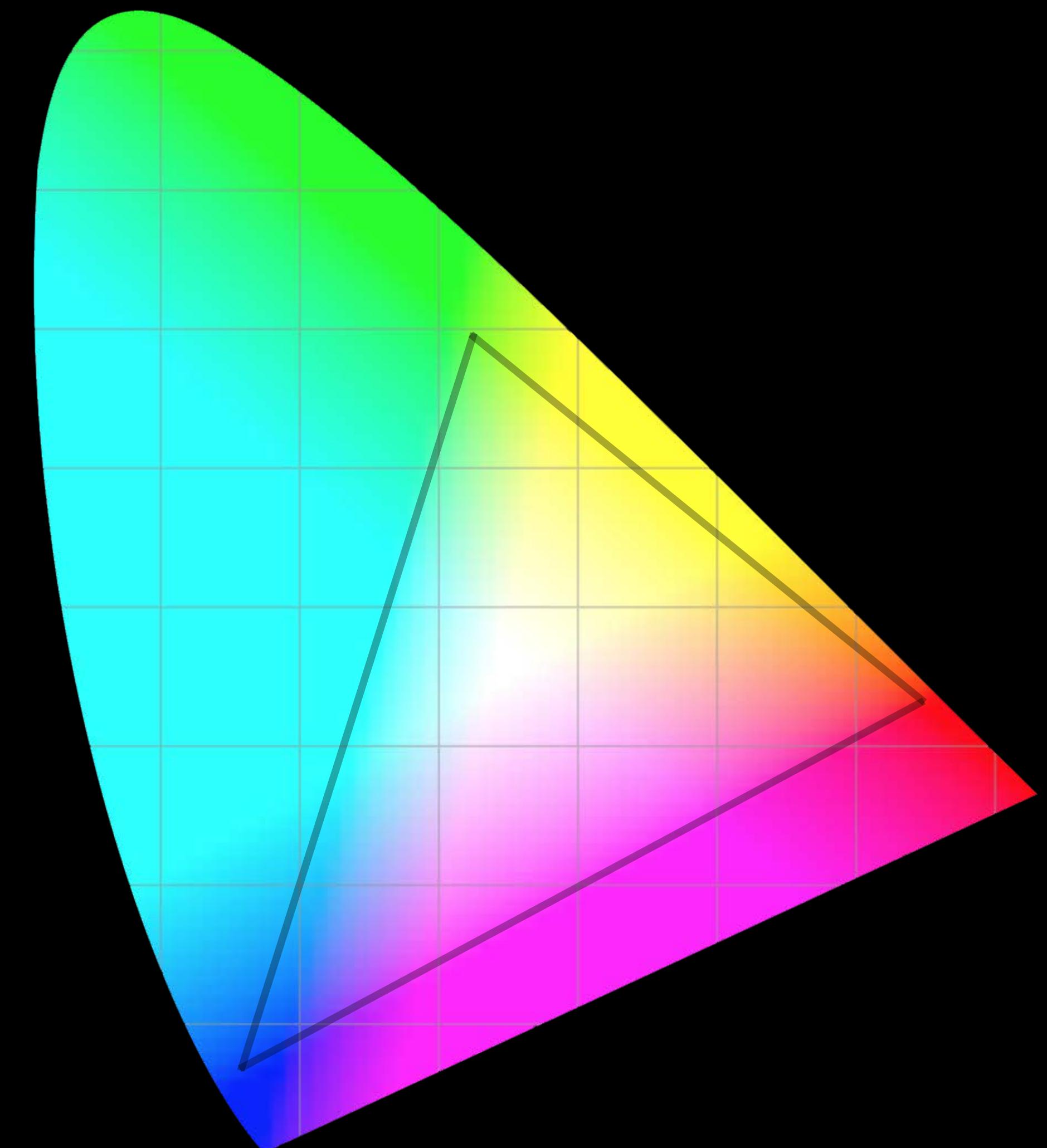
Wide Color

Technology shift



Wide Color

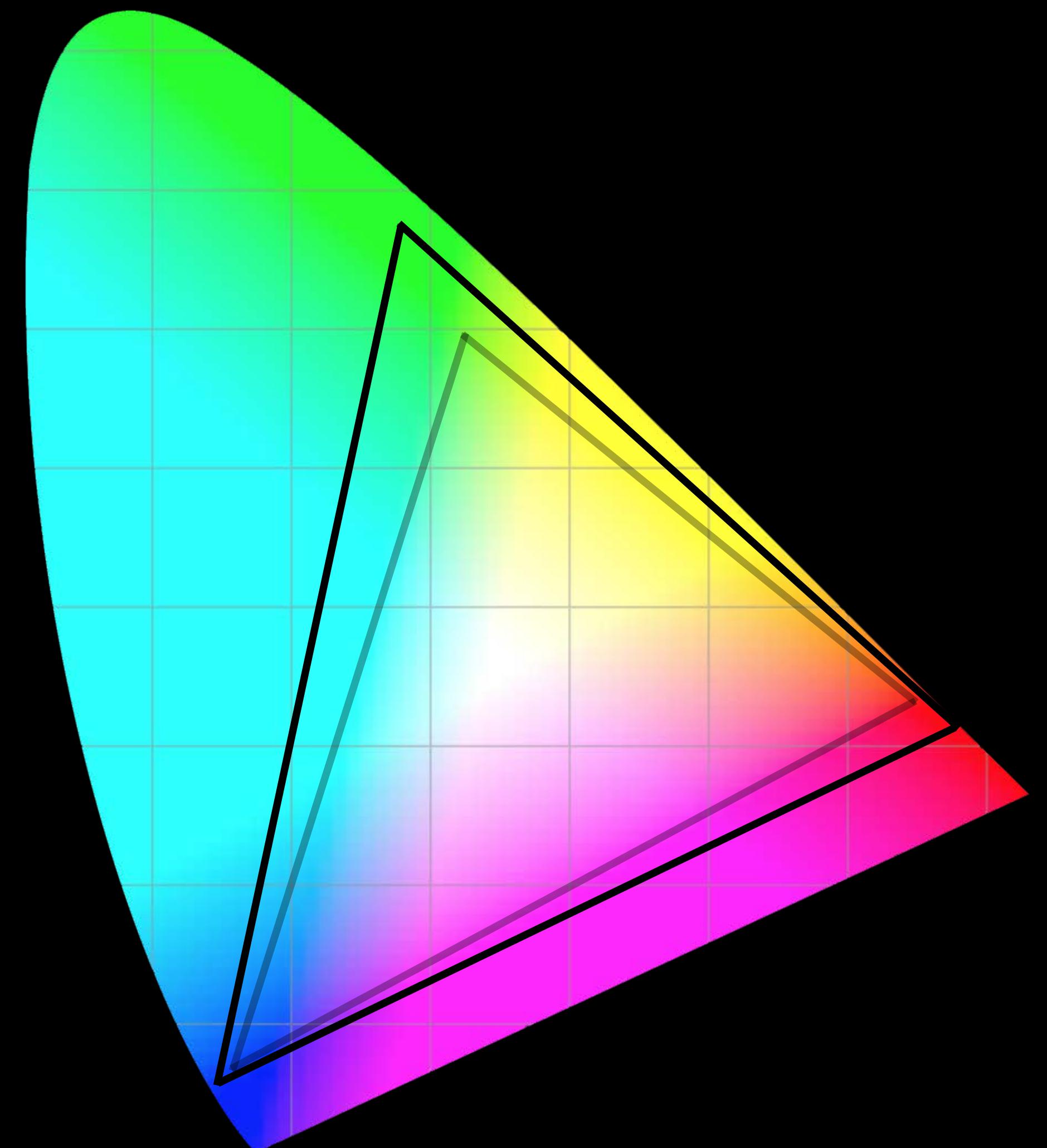
Technology shift



Wide Color

Technology shift

From sRGB to extended sRGB

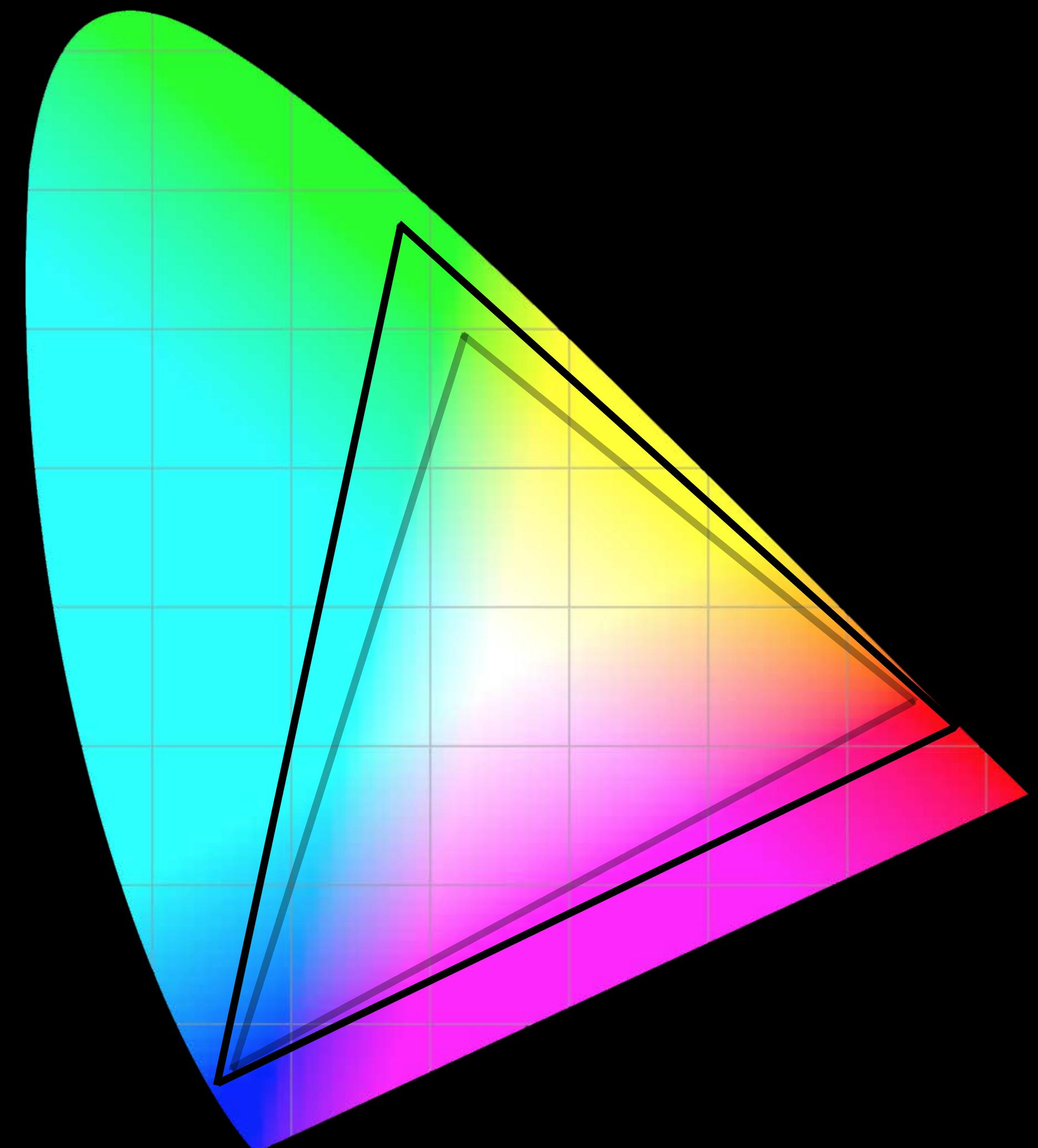


Wide Color

Technology shift

From sRGB to extended sRGB

iOS 9.3 is color managed!



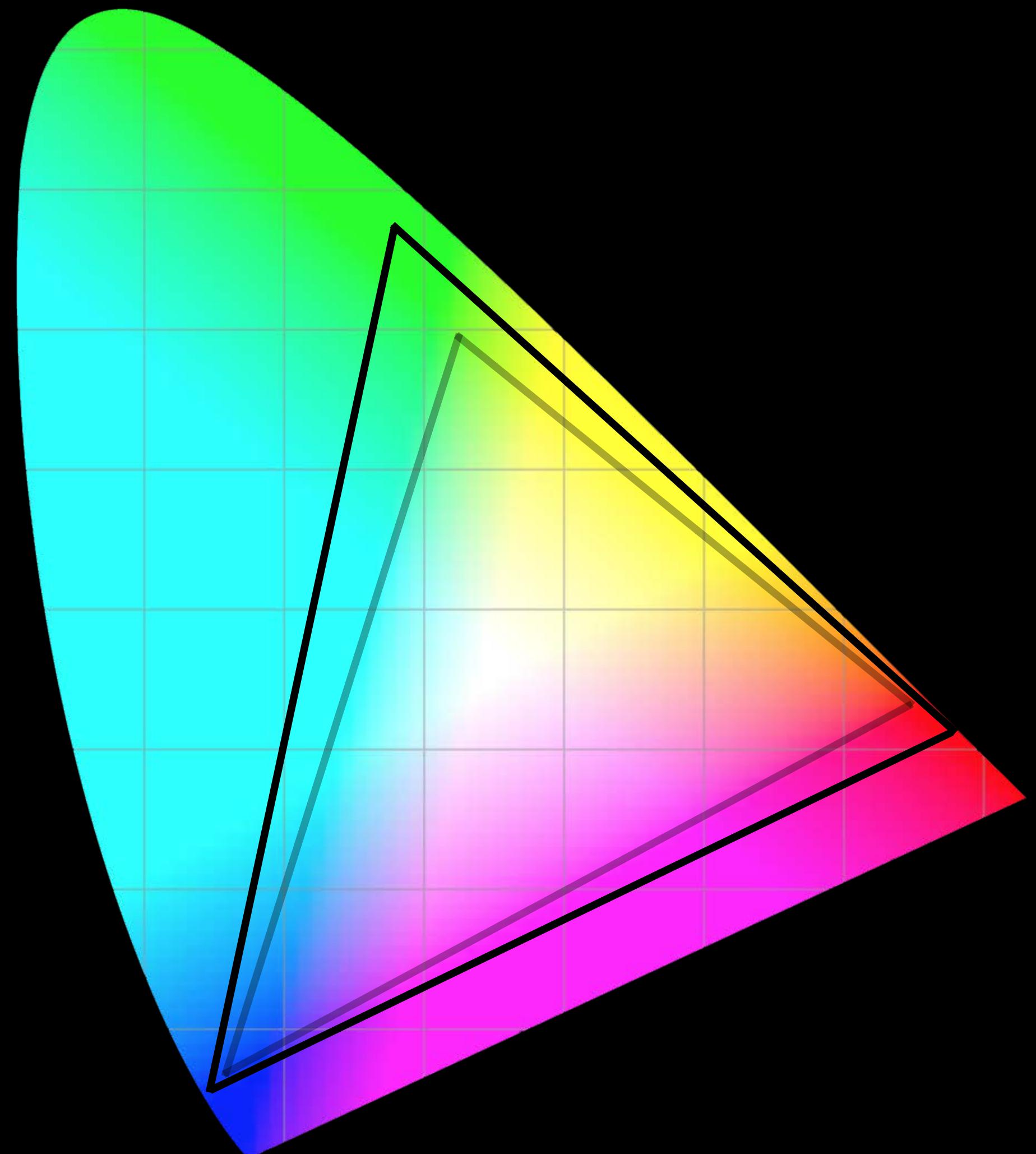
Wide Color

Technology shift

From sRGB to extended sRGB

iOS 9.3 is color managed!

Exposed as API in iOS 10.0



Wide Color

`UIImageView`, color-managed
since iOS 9.3



Wide Color UIColor Support

Wide Color

UIColor Support

```
public class UIColor : NSObject {  
  
    public init(red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)  
    public init(displayP3Red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)
```

Wide Color UIColor Support

Go beyond [0-1] for extended sRGB with the existing initializer

```
public class UIColor : NSObject {  
  
    public init(red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)  
  
    public init(displayP3Red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)
```

Wide Color

UIColor Support

Go beyond [0-1] for extended sRGB with the existing initializer

Use displayP3 for content creation and interchange

```
public class UIColor : NSObject {  
  
    public init(red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)  
  
    public init(displayP3Red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)
```

Wide Color

UIColor Support

Go beyond [0-1] for extended sRGB with the existing initializer

Use displayP3 for content creation and interchange

```
public class UIColor : NSObject {  
  
    public init(red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)  
  
    public init(displayP3Red: CGFloat, green: CGFloat, blue: CGFloat, alpha: CGFloat)
```

Image Renderer

Image Renderer

`UIGraphicsBeginImageContext` and `UIGraphicsEndImageContext`

Image Renderer

`UIGraphicsBeginImageContext` and `UIGraphicsEndImageContext`

- 32 bits and sRGB only

Image Renderer

`UIGraphicsBeginImageContext` and `UIGraphicsEndImageContext`

- 32 bits and sRGB only
- Error prone

Image Renderer

`UIGraphicsBeginImageContext` and `UIGraphicsEndImageContext`

- 32 bits and sRGB only
- Error prone
- Not extensible

Image Renderer

`UIGraphicsBeginImageContext` and `UIGraphicsEndImageContext`

- 32 bits and sRGB only
- Error prone
- Not extensible

```
func createDrawing(size: CGSize) -> UIImage {  
    let renderer = UIGraphicsBeginImageContext(size)  
    // Do your drawing here  
    let image = UIGraphicsGetImageFromCurrentImageContext()  
    UIGraphicsEndImageContext()  
    return image  
}
```



Image Renderer

NEW

Image Renderer

NEW

New `UIGraphicsRenderer` class

Image Renderer

NEW

New `UIGraphicsRenderer` class

- Fully color managed

Image Renderer

NEW

New `UIGraphicsRenderer` class

- Fully color managed
- Block-based

Image Renderer

NEW

New `UIGraphicsRenderer` class

- Fully color managed
- Block-based
- Subclasses for images and PDF

Image Renderer

NEW

New `UIGraphicsRenderer` class

- Fully color managed
- Block-based
- Subclasses for images and PDF
- Manages context lifetime

Image Renderer

NEW

New **UIGraphicsRenderer** class

- Fully color managed
- Block-based
- Subclasses for images and PDF
- Manages context lifetime

```
func createDrawing(size: CGSize) -> UIImage {  
    let renderer = UIGraphicsImageRenderer(size: size)  
    return renderer.image { rendererContext in  
        // Do your drawing here  
    }  
}
```



Asset Management

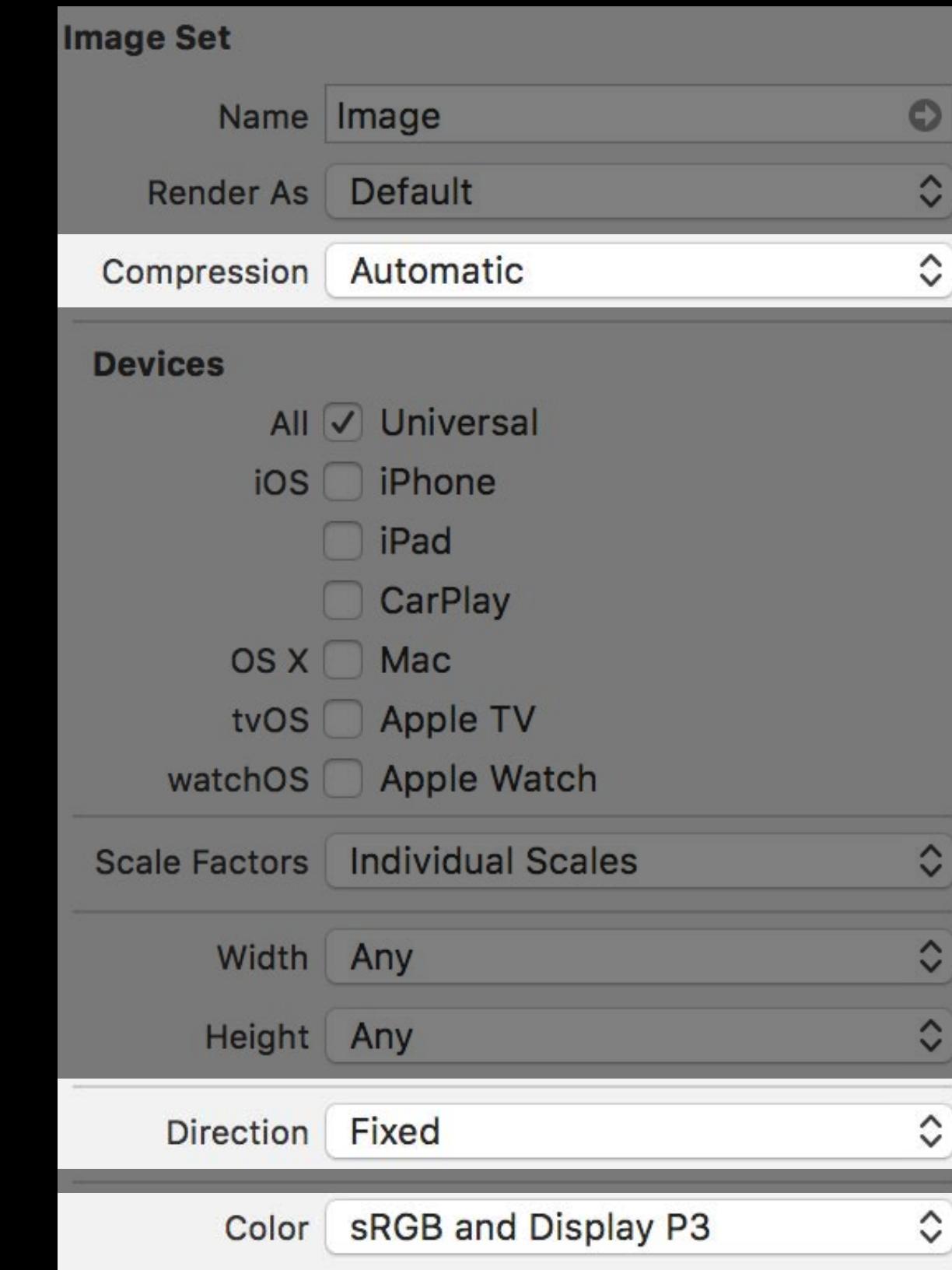
NEW

Wide color assets

Directional image assets

Compression

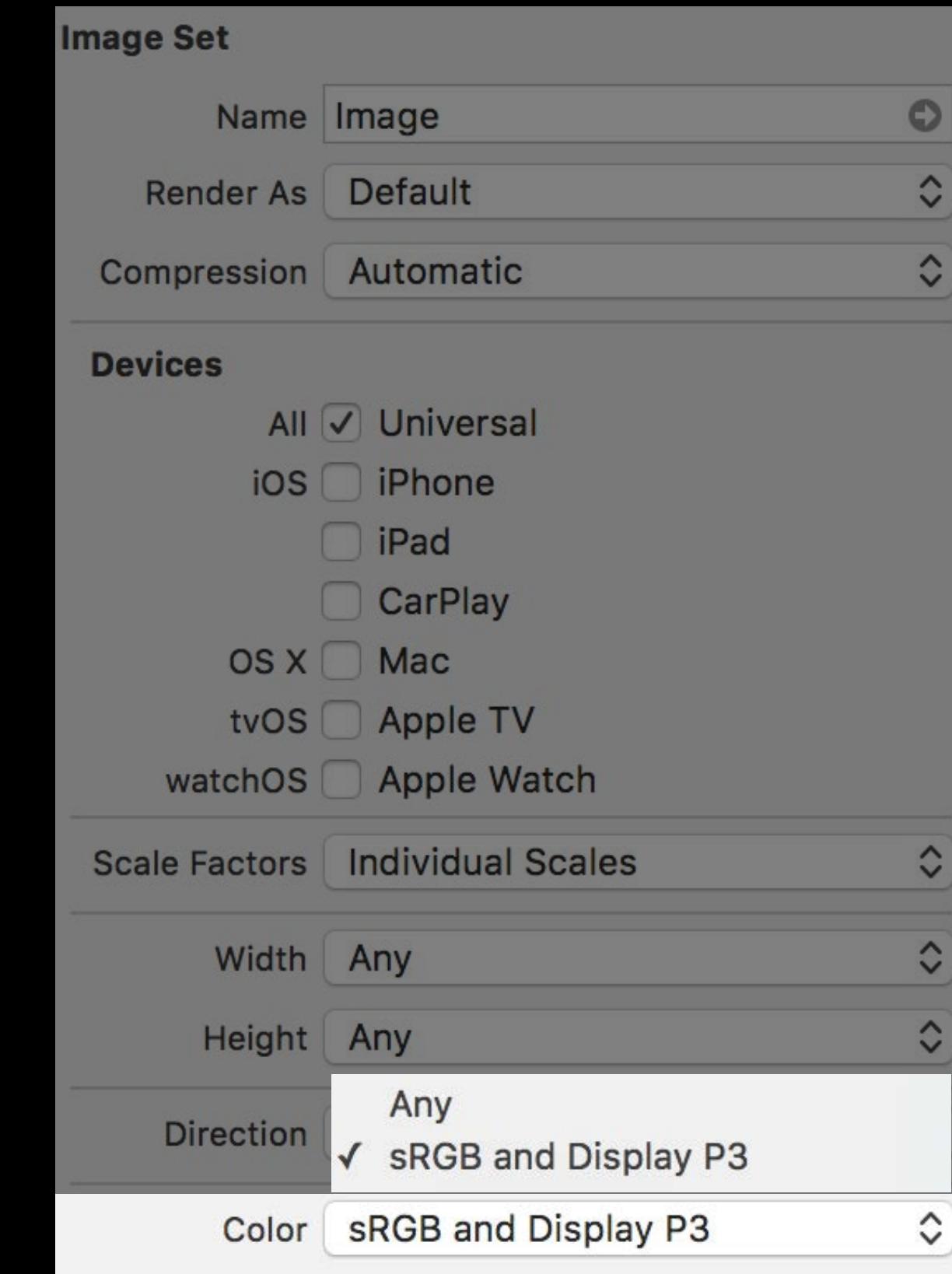
Integrated with the UIKit trait system



Asset Management

Wide color assets

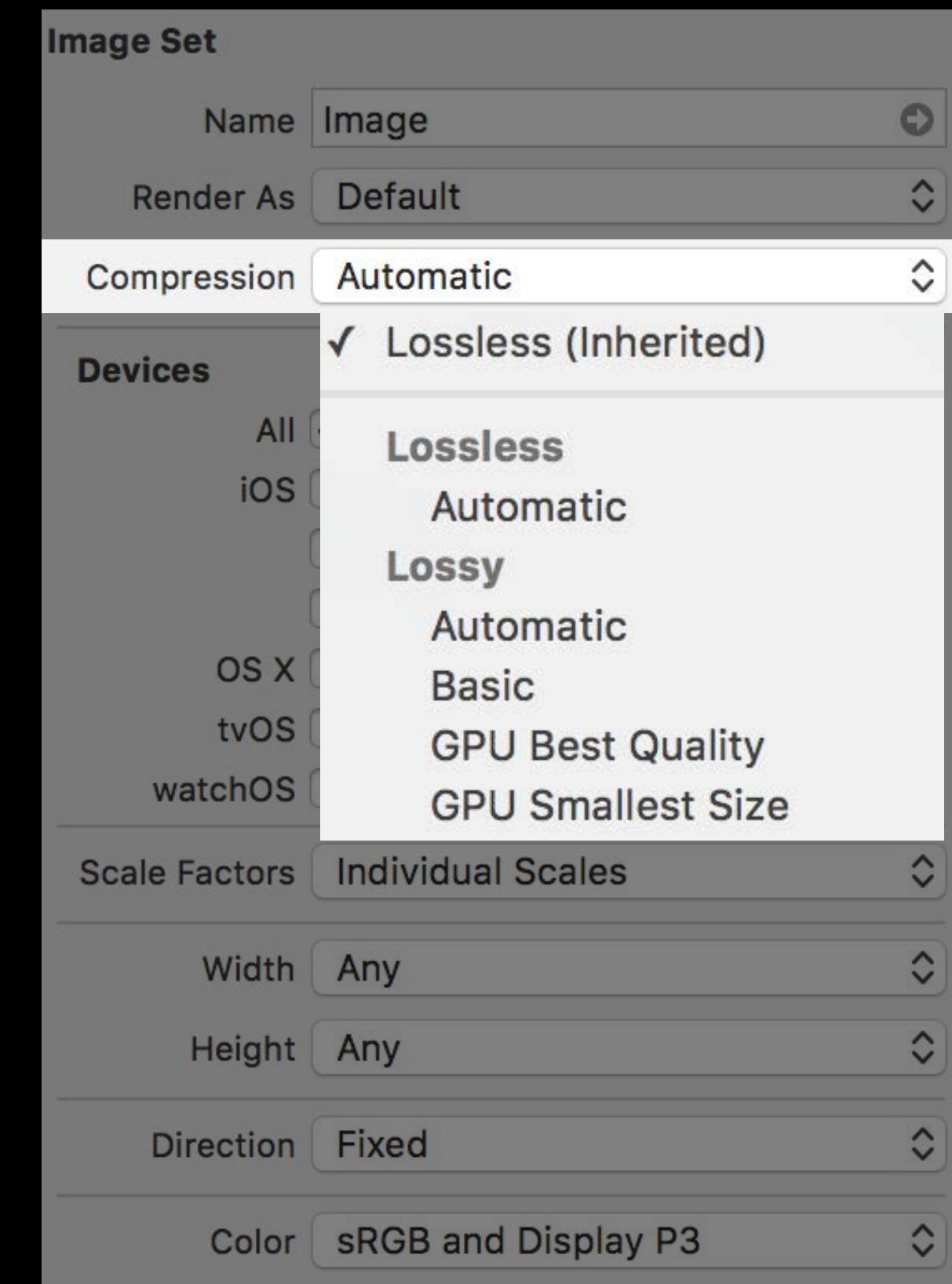
- Automatic variants generation
- Compatible with App Thinning



Asset Management

Compression

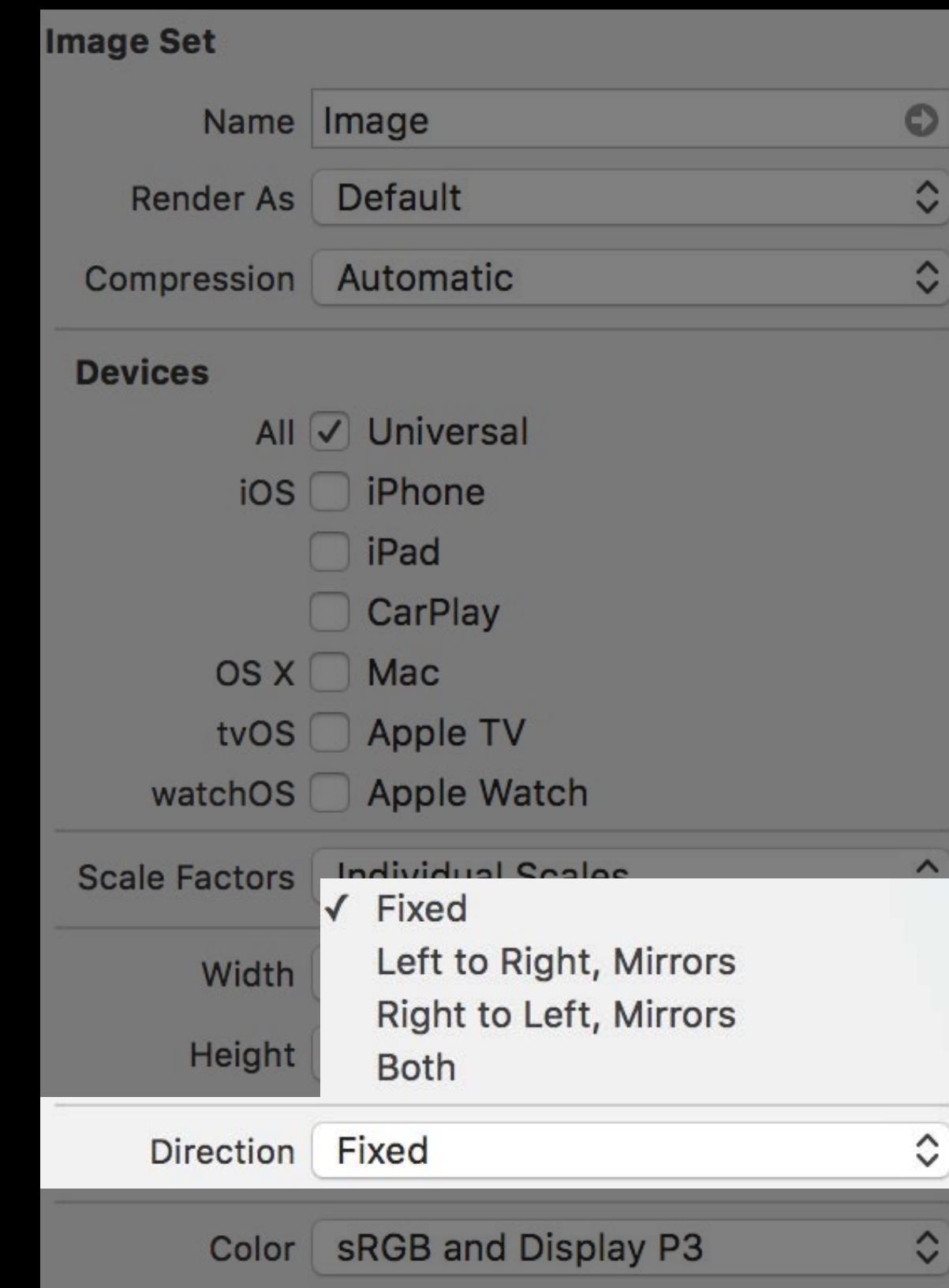
- Now supports automatic lossy compression
- Great compromise between footprint and quality
- Automatic variant will find the right tradeoff
- Compatible with App Thinning



Asset Management

Directional image assets for right-to-left and left-to-right UIs

- Specify if an image should be flipped or not
- Provide specific images

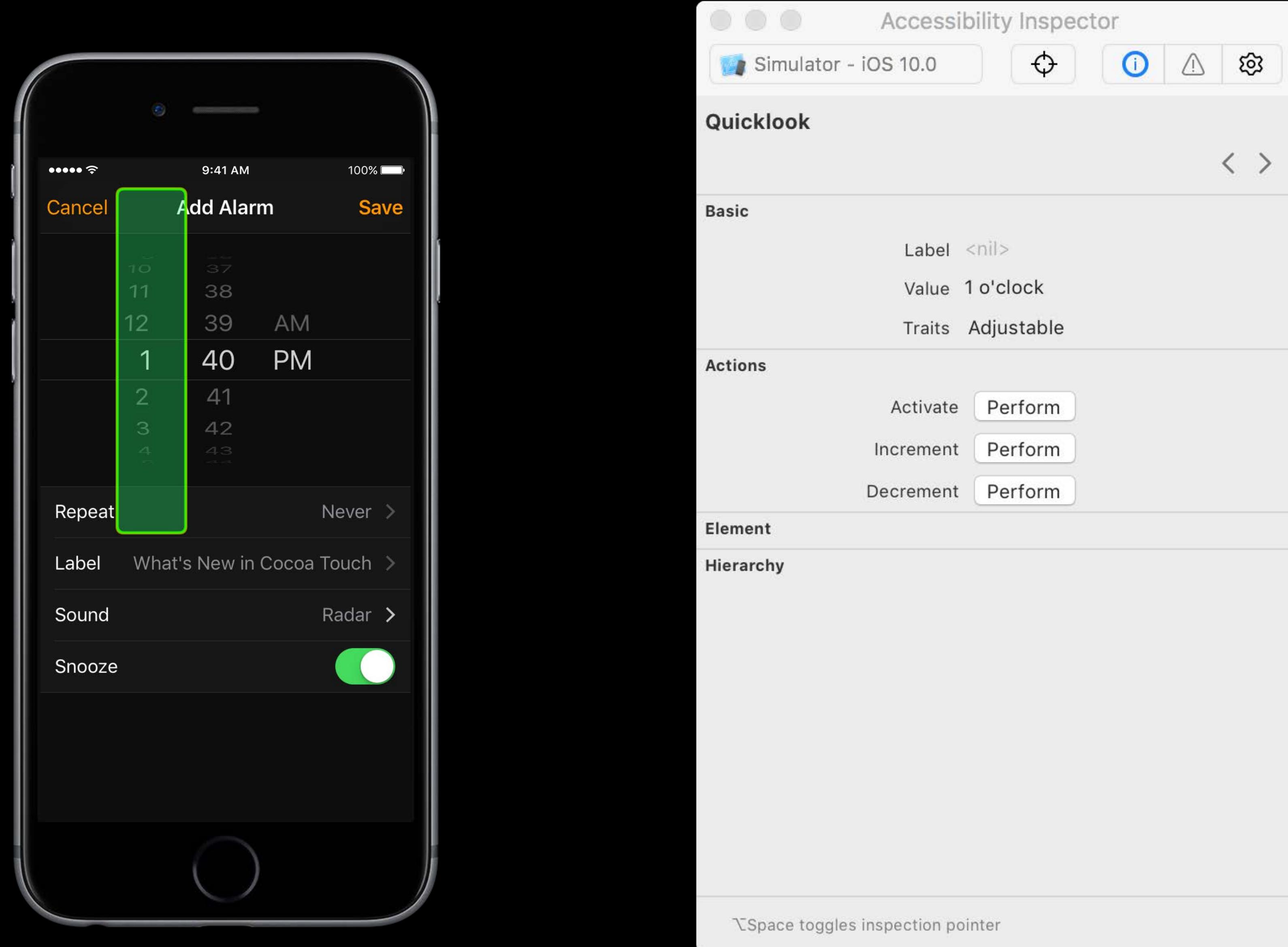


Building Better User Interfaces

Accessibility Inspector



Accessibility Inspector



The Accessibility Inspector window displays the Quicklook details for an iPhone screen showing an alarm clock interface.

Quicklook

Basic

- Label <nil>
- Value 1 o'clock
- Traits Adjustable

Actions

- Activate Perform
- Increment Perform
- Decrement Perform

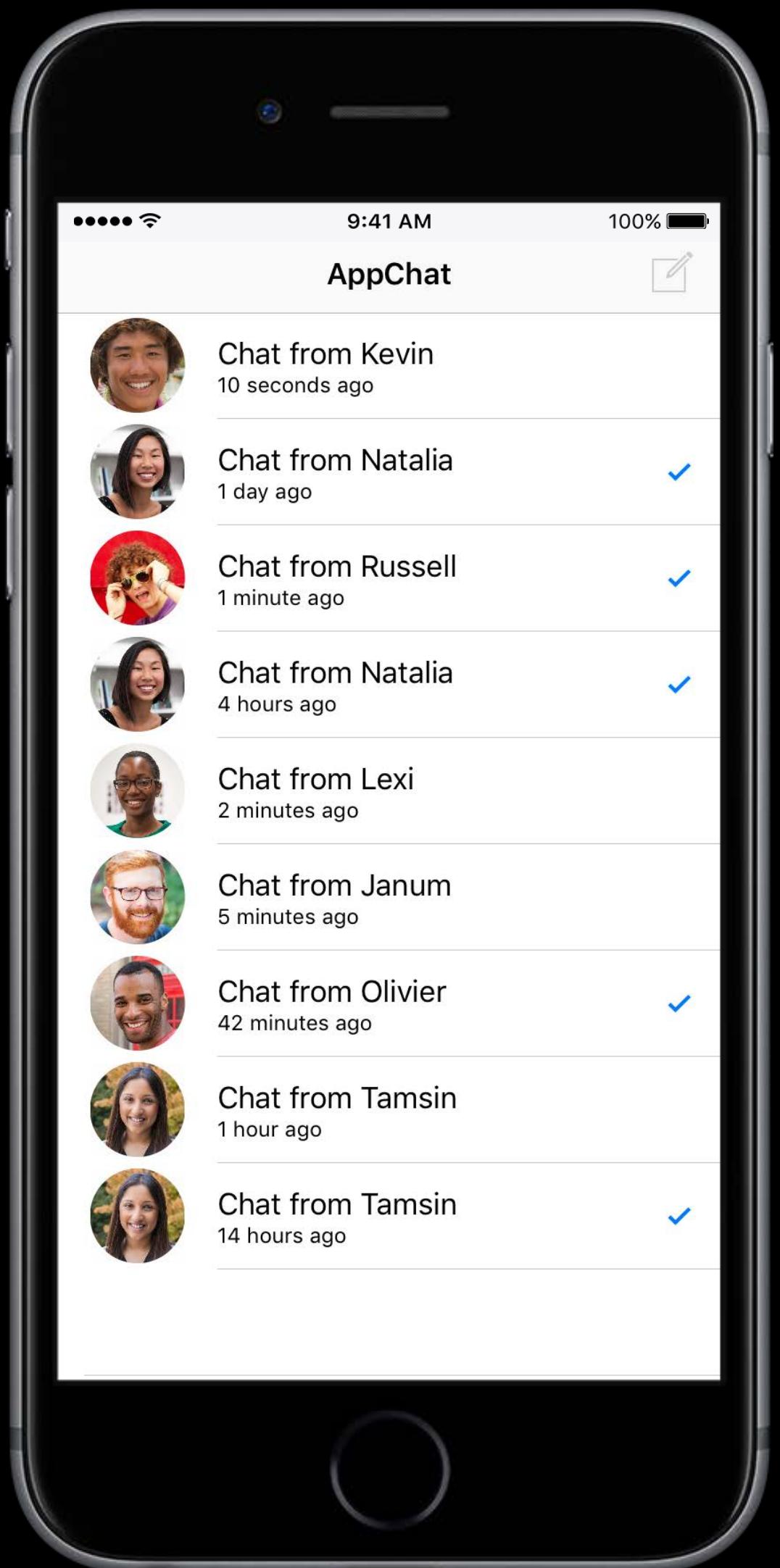
Element

Hierarchy

Space toggles inspection pointer

The iPhone screen shows the "Add Alarm" interface with a green box highlighting the hour selection dropdown menu. The menu contains the values 10, 11, 12, 1, 2, 3, and 4. The hour is set to 1 PM.

Accessibility Inspector



Accessibility Inspector

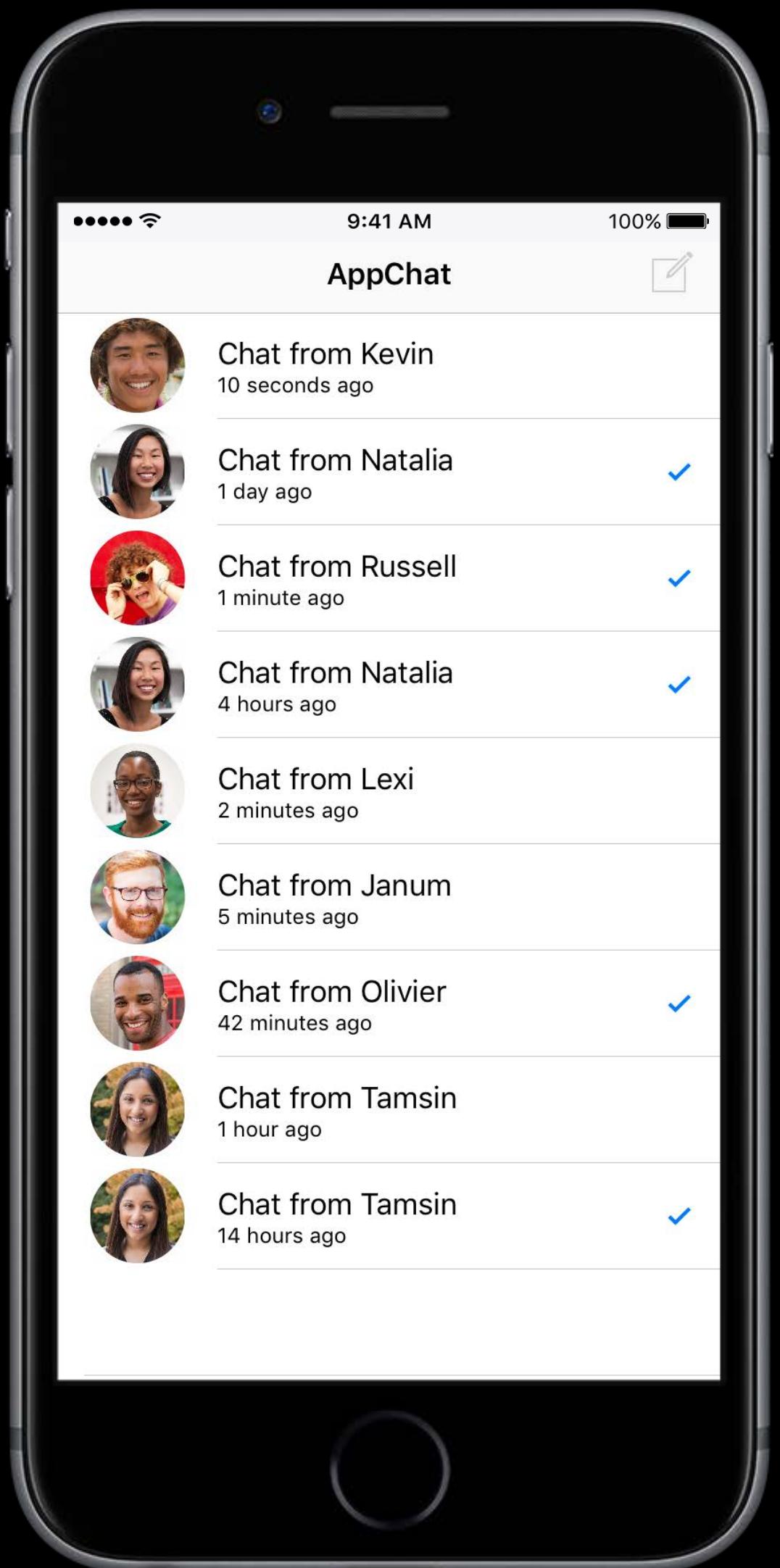
Simulator - iOS 10.0

Run audit

- ▶⚠️ Dynamic Text unsupported
This element does not support dynamic text. Consider using a dynamic text font
- ▼⚠️ Label duplicates traits
The accessibilityLabel of this element duplicates information that is available in the traits.
- ▼⚠️ Element has no description
This element is missing useful accessibility information.

3 warnings

Accessibility Inspector



Accessibility Inspector

Simulator - iOS 10.0

Run audit

- ▶⚠️ Dynamic Text unsupported
This element does not support dynamic text. Consider using a dynamic text font
- ▼⚠️ Label duplicates traits
The accessibilityLabel of this element duplicates information that is available in the traits.
- ▼⚠️ Element has no description
This element is missing useful accessibility information.

3 warnings













Speech Recognition

SFSpeechRecognizer

Speech Recognition

SFSpeechRecognizer

Continuous speech recognition

Speech Recognition

SFSpeechRecognizer

Continuous speech recognition

From audio files or audio buffers

Speech Recognition

SFSpeechRecognizer

Continuous speech recognition

From audio files or audio buffers

Optimized for free-form dictation or search-style strings

Speech Recognition

SFSpeechRecognizer

Continuous speech recognition

From audio files or audio buffers

Optimized for free-form dictation or search-style strings

```
let recognizer = SFSpeechRecognizer()  
let request = SFSpeechURLRecognitionRequest(url: audioFileURL)  
  
recognizer?.recognitionTask(with: request, resultHandler: { (result, error) in  
    print(result?.bestTranscription.formattedString)  
})
```

Speech Recognition

SFSpeechRecognizer

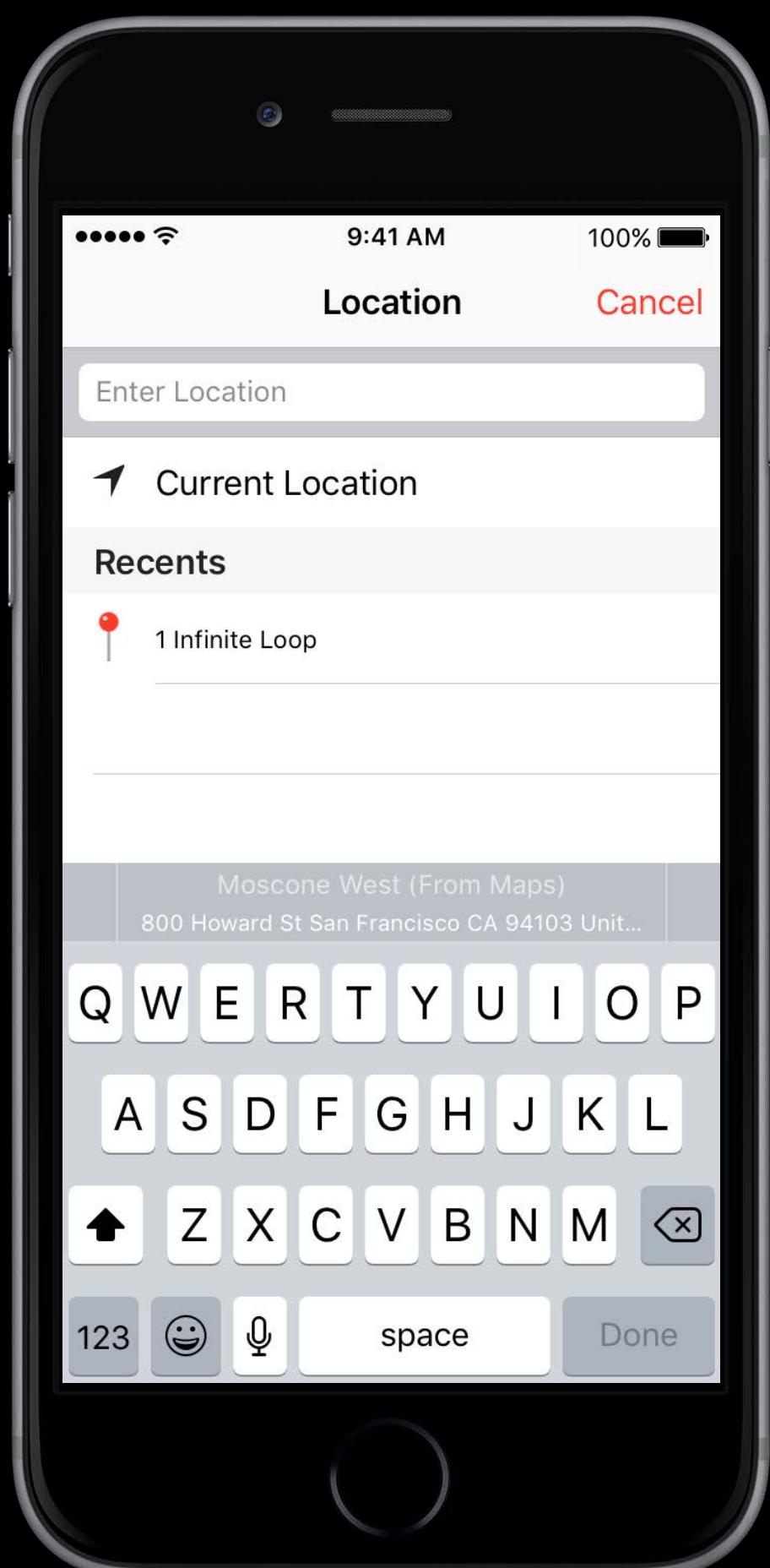
Continuous speech recognition

From audio files or audio buffers

Optimized for free-form dictation or search-style strings

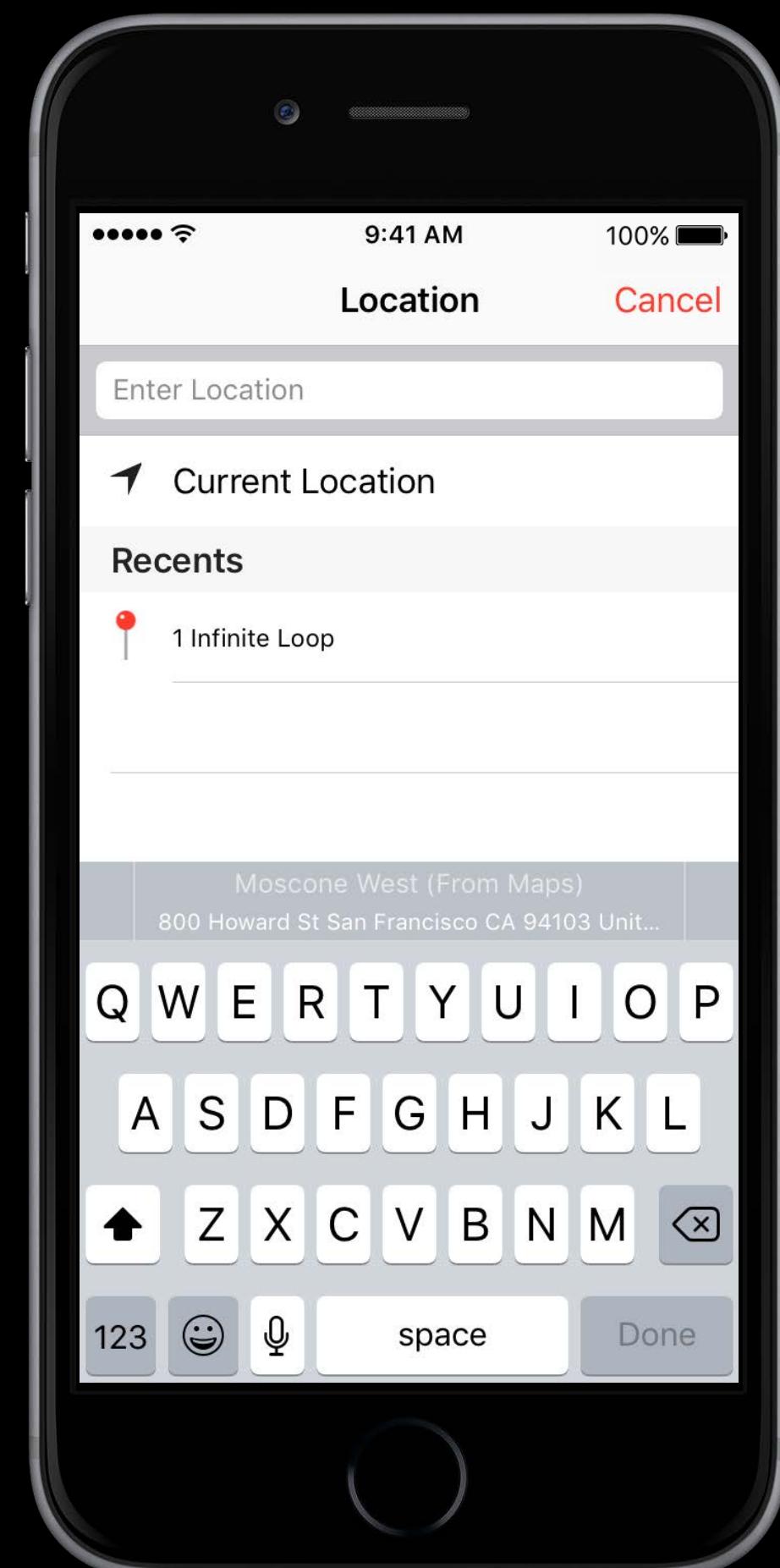
```
let recognizer = SFSpeechRecognizer()  
let request = SFSpeechURLRecognitionRequest(url: audioFileURL)  
  
recognizer?.recognitionTask(with: request, resultHandler: { (result, error) in  
    print(result?.bestTranscription.formattedString)  
})
```

Smarter Text Input



Smarter Text Input

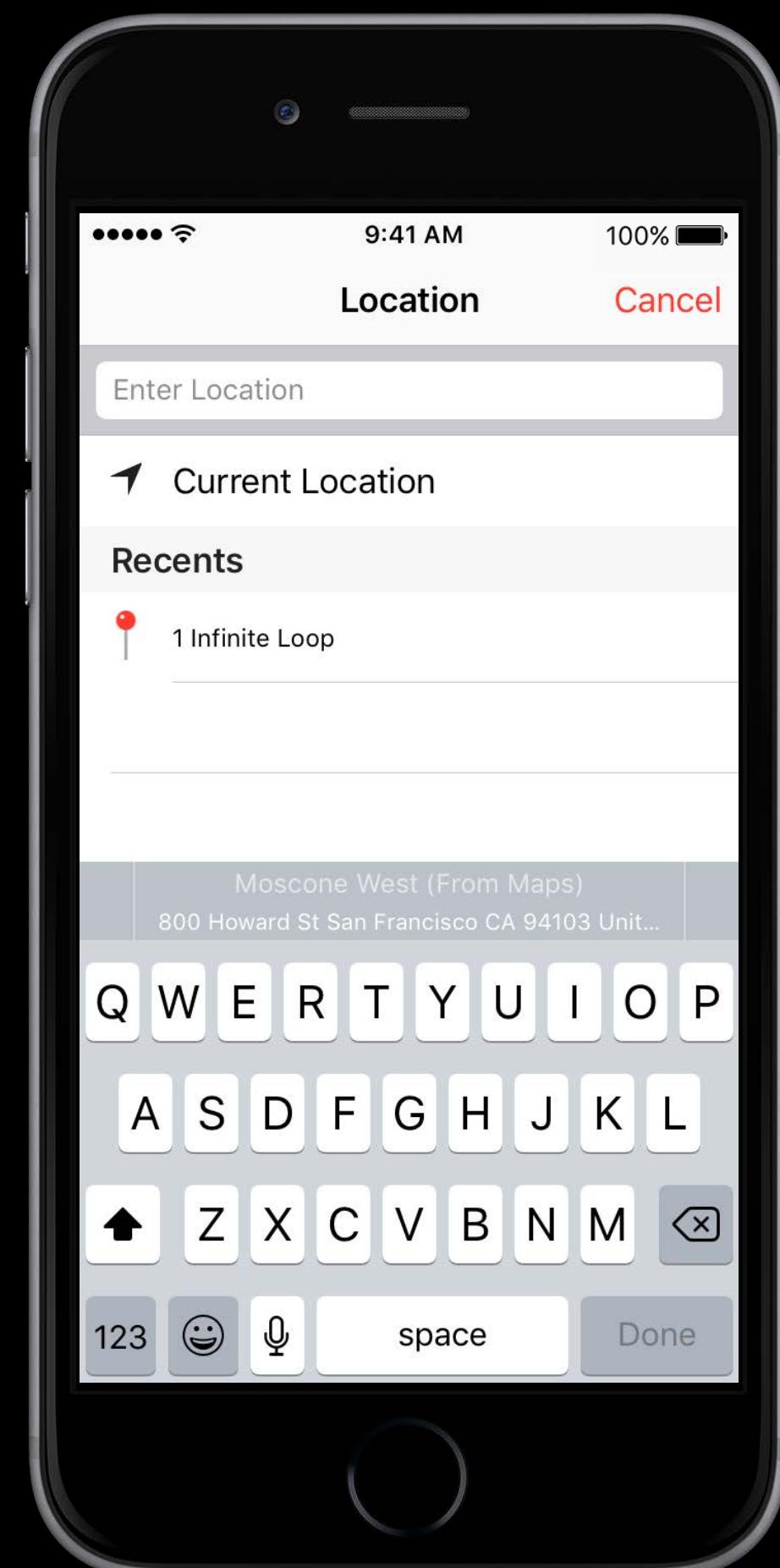
Semantic tagging of text fields, text views and web content



Smarter Text Input

Semantic tagging of text fields, text views and web content

Provides intelligent suggestions

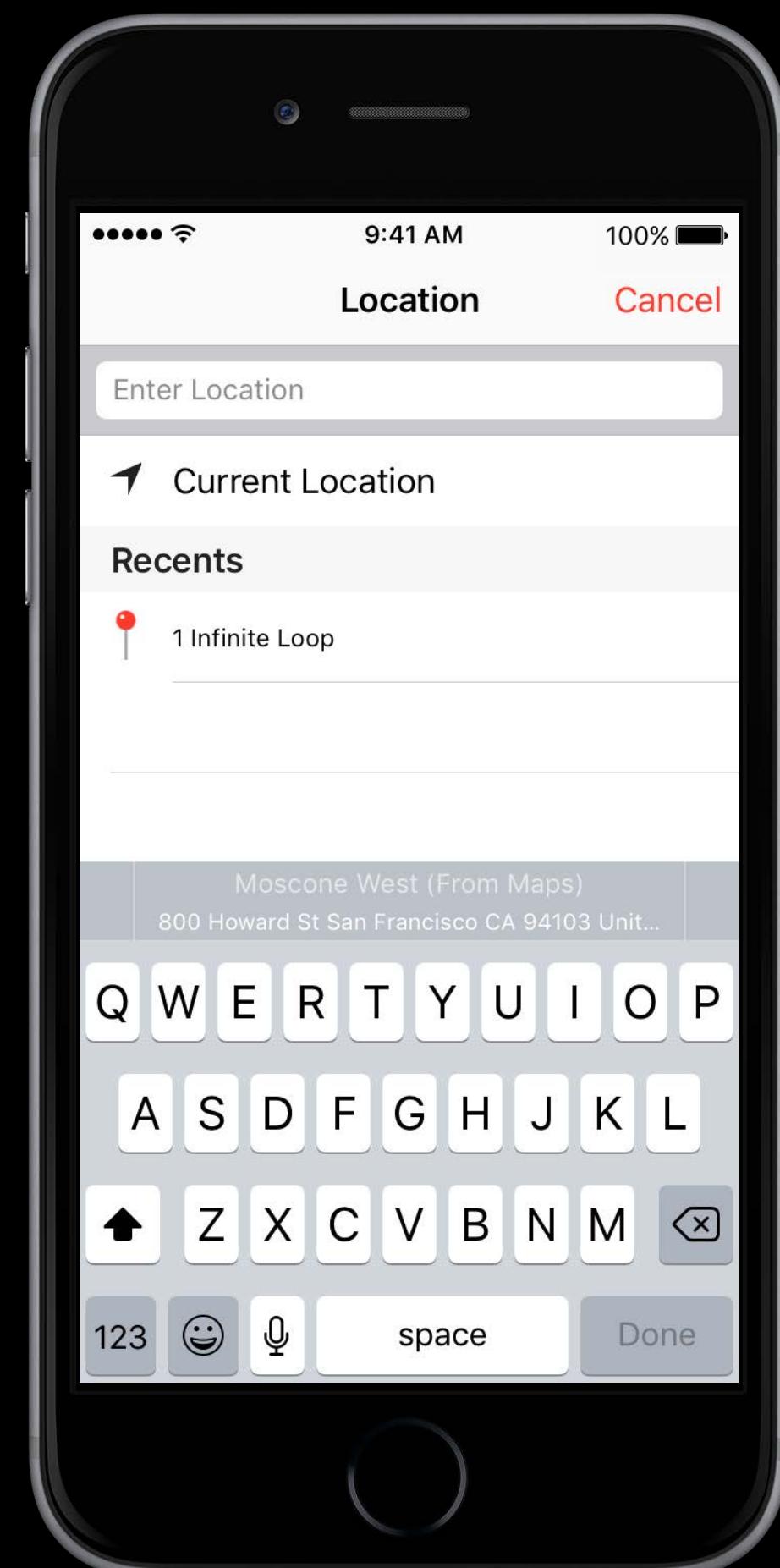


Smarter Text Input

Semantic tagging of text fields, text views and web content

Provides intelligent suggestions

Many predefined content type



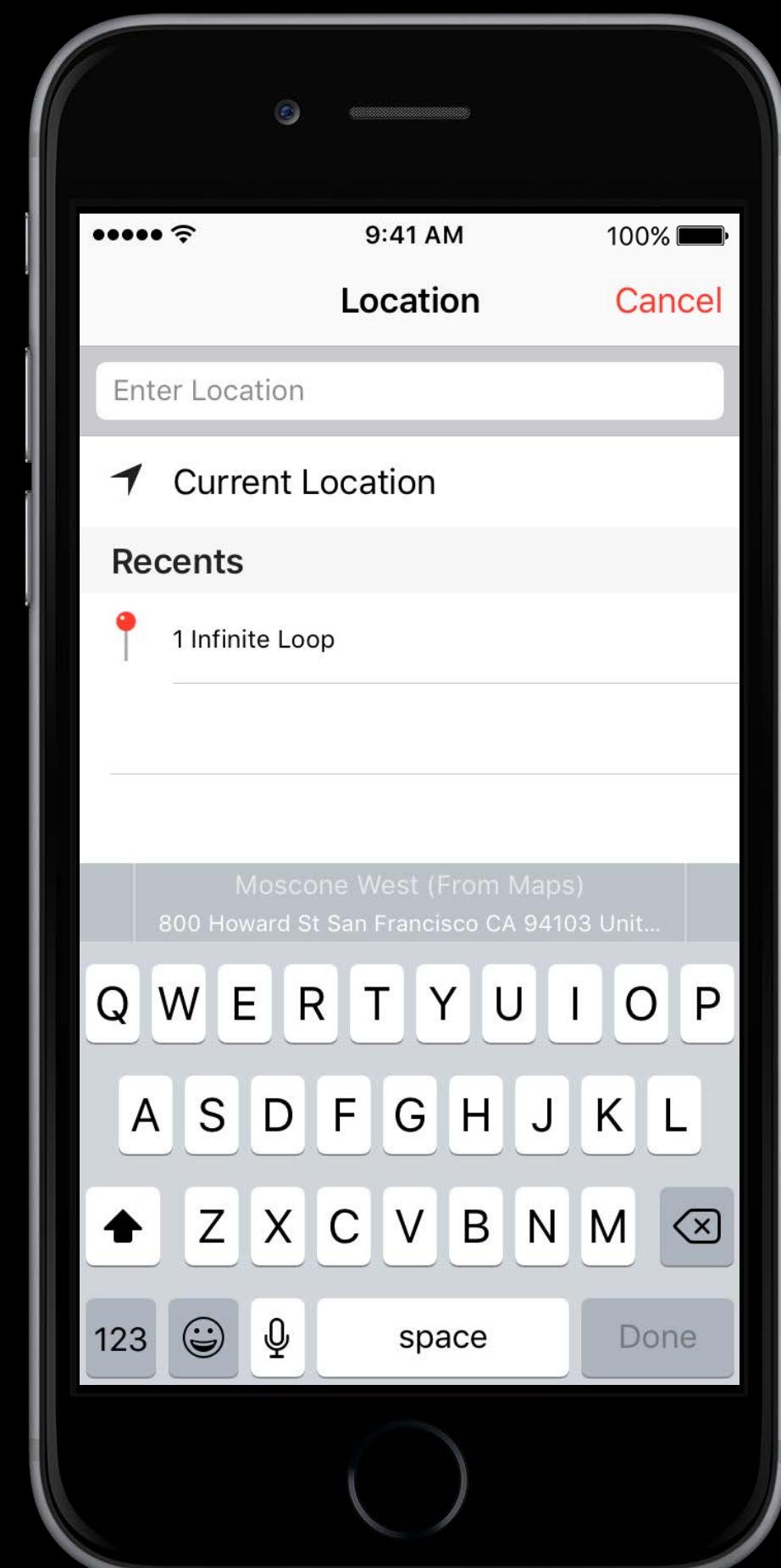
Smarter Text Input

Semantic tagging of text fields, text views and web content

Provides intelligent suggestions

Many predefined content type

People



Smarter Text Input

Semantic tagging of text fields, text views and web content

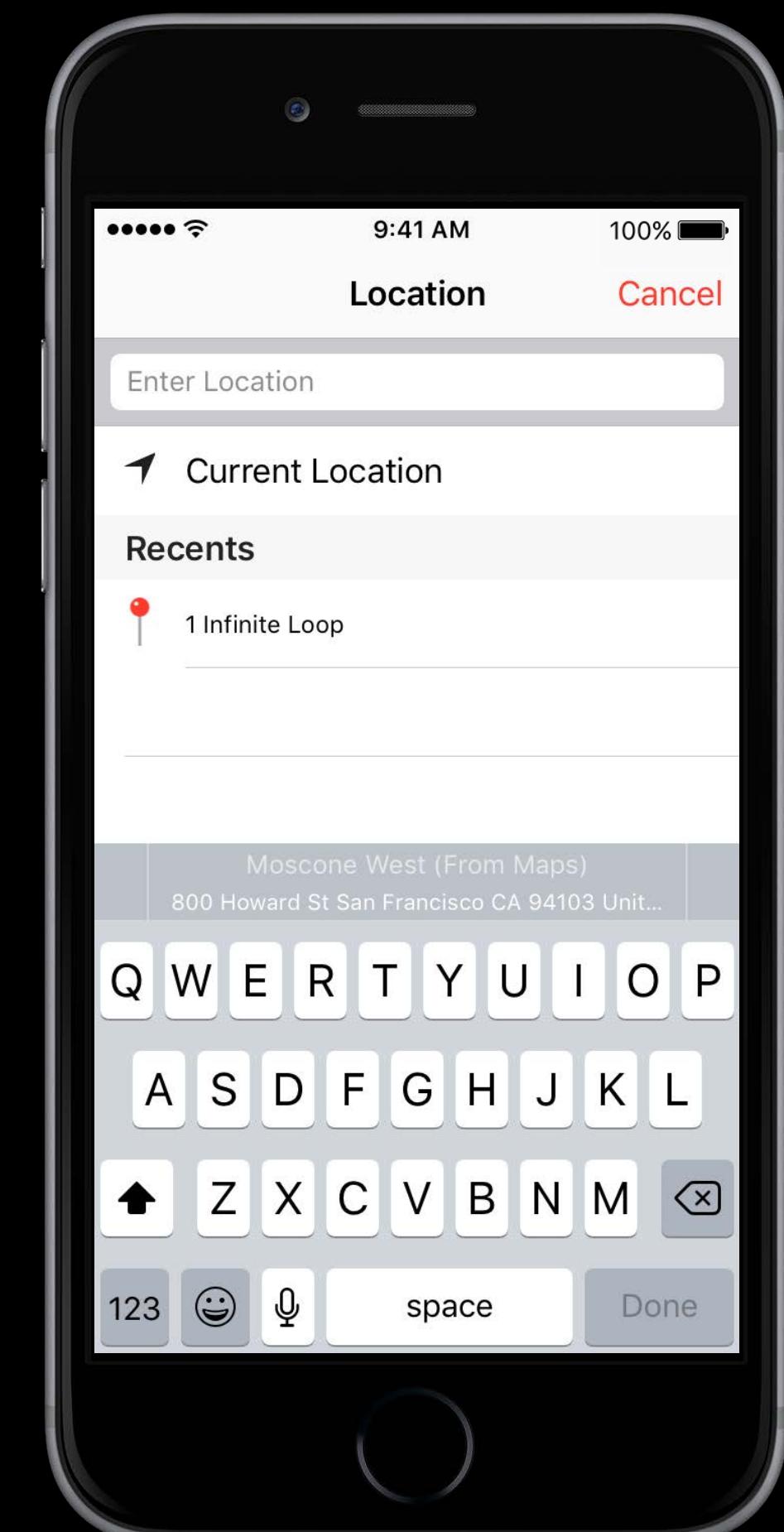
Provides intelligent suggestions

Many predefined content type

People

Locations

```
textField.textContentType = UITextContentTypeFullStreetAddress
```



Smarter Text Input

Semantic tagging of text fields, text views and web content

Provides intelligent suggestions

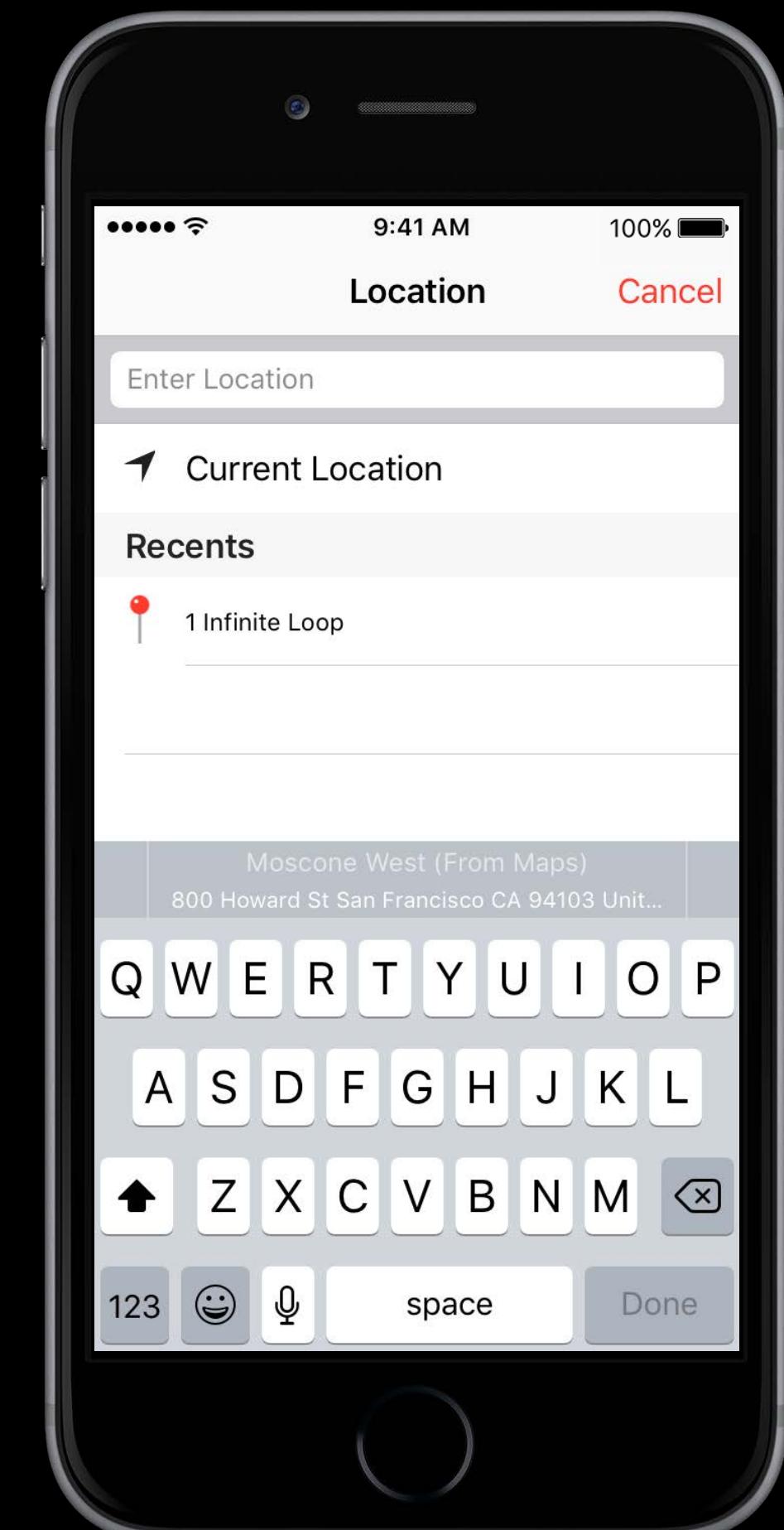
Many predefined content type

People

Locations

```
textField.textContentType = UITextContentTypeFullStreetAddress
```

Email, telephone, credit card number, ...



Dynamic Type

Content size category trait

Dynamic Type

Content size category trait

No longer a property on **UIApplication**

Dynamic Type

Content size category trait

No longer a property on **UIApplication**

No need to listen to notifications

Dynamic Type

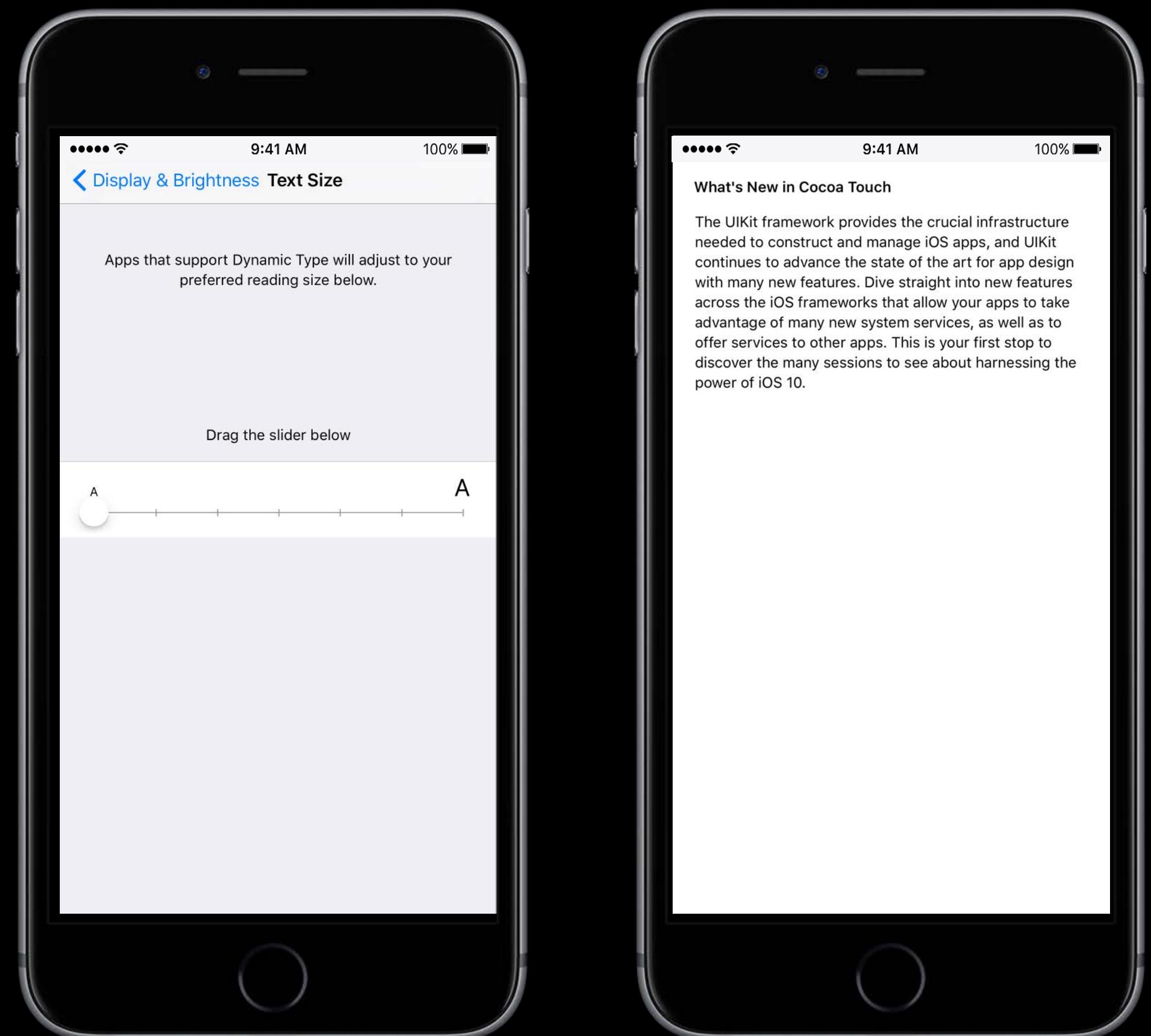
Content size category trait

No longer a property on **UIApplication**

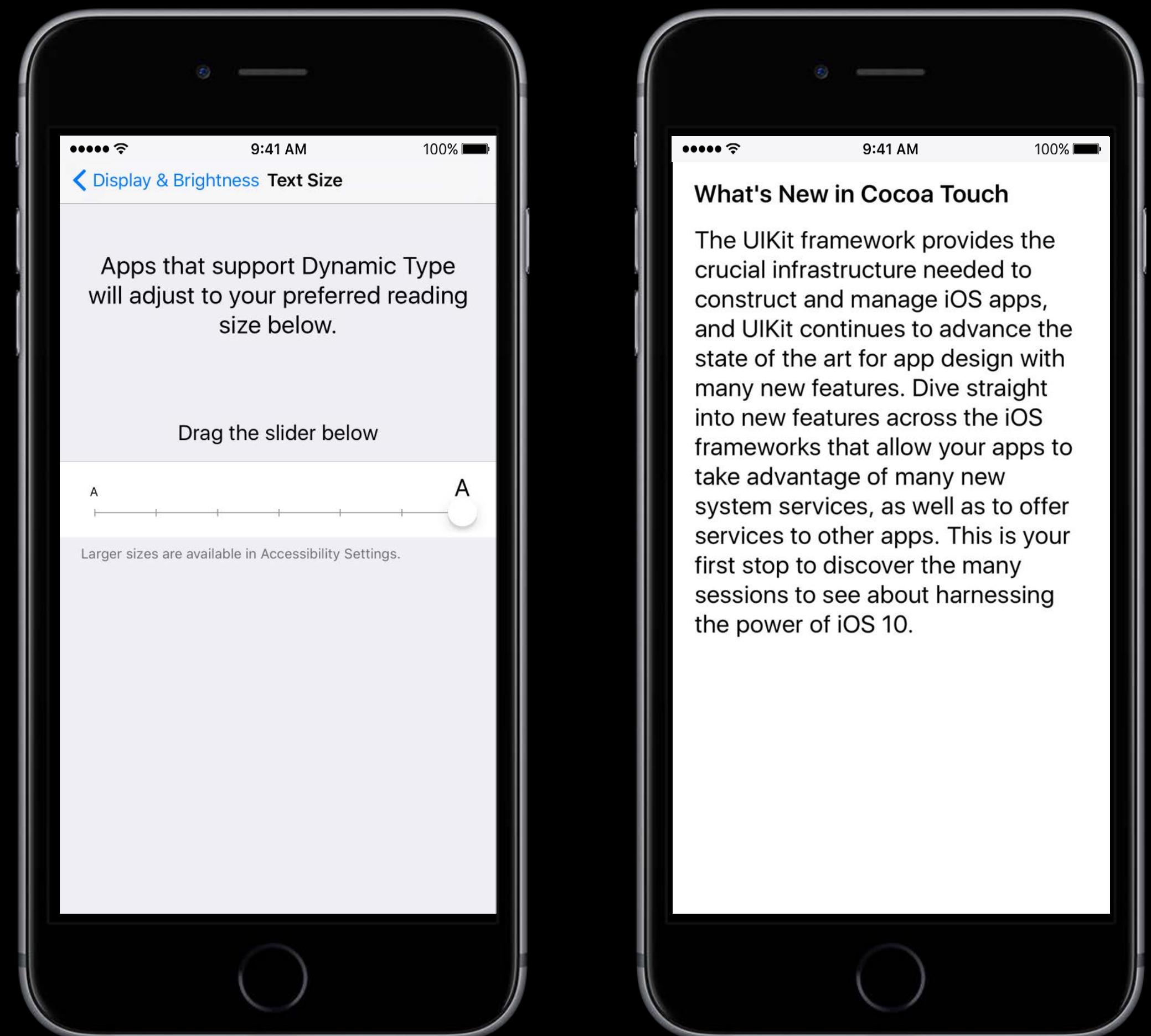
No need to listen to notifications

Now available in extensions

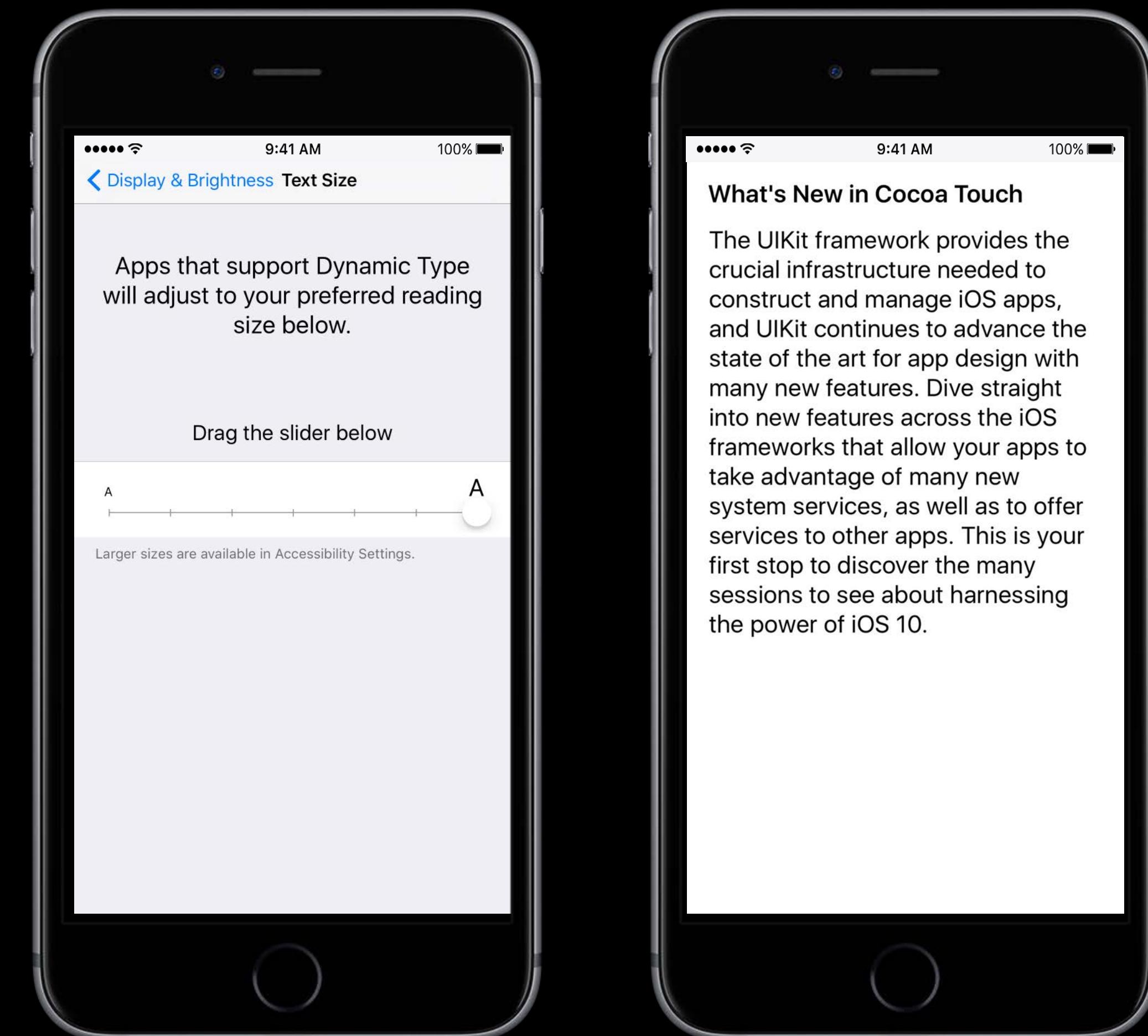
Dynamic Type



Dynamic Type



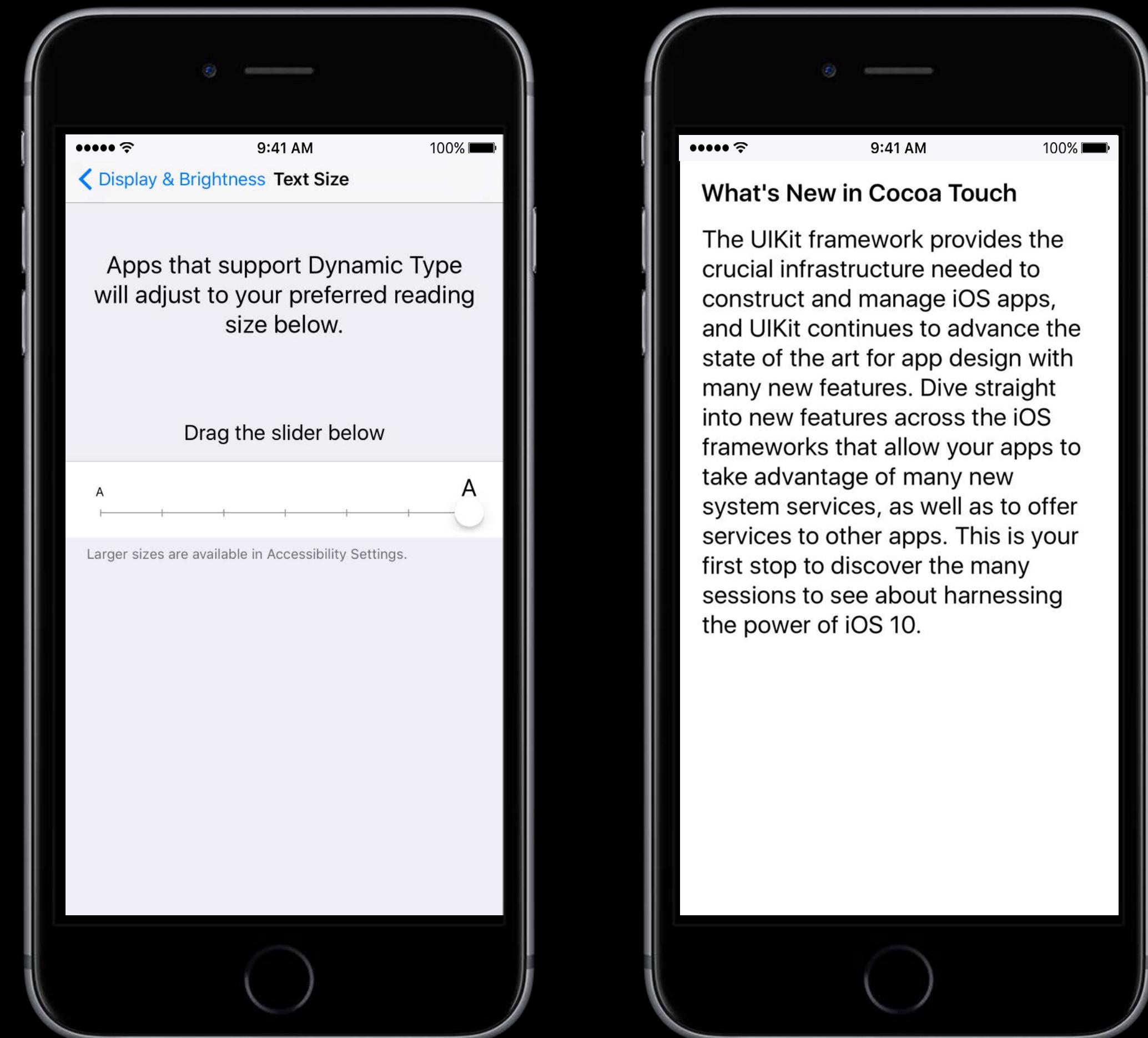
Dynamic Type



```
label.font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
label.adjustsFontForContentSizeCategory = true
```

Dynamic Type

Automatic support in label, text views and controls



```
label.font = UIFont.preferredFont(forTextStyle: UIFontTextStyleBody)  
label.adjustsFontForContentSizeCategory = true
```

Improved Customization

Tab bar items

Improved Customization

Tab bar items

Custom badge colors and text attributes

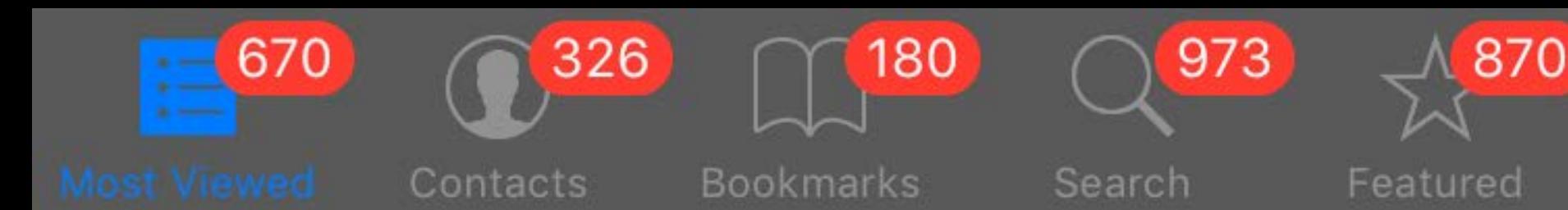
Customizable unselected Tint Color

Improved Customization

Tab bar items

Custom badge colors and text attributes

Customizable unselected Tint Color



Improved Customization

Tab bar items

Custom badge colors and text attributes

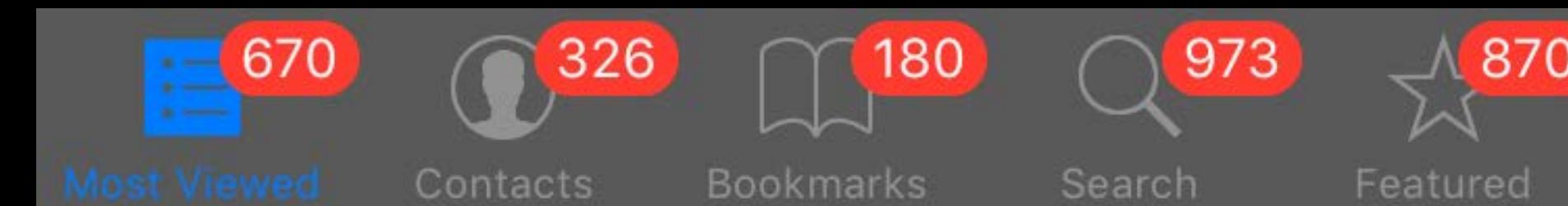
Customizable unselected Tint Color

```
tabBarItem.badgeColor = UIColor.white()

badgeTextAttributes = [ NSForegroundColorAttributeName : UIColor.blue(),
                        NSFontAttributeName : UIFont.italicSystemFont(ofSize: 12) ]

tabBarItem.setBadgeTextAttributes(textAttributes: badgeTextAttributes,
                                    forState: UIControlStateNormal)

tabBar.unselectedTintColor = UIColor.brown()
```



Improved Customization

Tab bar items

Custom badge colors and text attributes

Customizable unselected Tint Color

```
tabBarItem.badgeColor = UIColor.white()

badgeTextAttributes = [ NSForegroundColorAttributeName : UIColor.blue(),
                        NSFontAttributeName : UIFont.italicSystemFont(ofSize: 12) ]

tabBarItem.setBadgeTextAttributes(textAttributes: badgeTextAttributes,
                                    forState: UIControlStateNormal)

tabBar.unselectedTintColor = UIColor.brown()
```



Peek & Pop

Improved WKWebView Support

Peek & Pop

Improved WKWebView Support

Fine control of Peek & Pop behaviors

Peek & Pop

Improved WKWebView Support

Fine control of Peek & Pop behaviors

Custom view controllers

Peek & Pop

Improved WKWebView Support

Fine control of Peek & Pop behaviors

Custom view controllers

Preview actions

Peek & Pop

Improved WKWebView Support

Fine control of Peek & Pop behaviors

Custom view controllers

Preview actions

Pop inside your app

Peek & Pop

Improved WKWebView Support

Fine control of Peek & Pop behaviors

Custom view controllers

Preview actions

Pop inside your app

```
public func webView(_ webView: WKWebView,  
                    shouldPreviewElement elementInfo: WKPreviewElementInfo) -> Bool  
  
public func webView(_ webView: WKWebView,  
                    previewingViewControllerForElement elementInfo: WKPreviewElementInfo,  
                    defaultActions previewActions: [WKPreviewActionItem]) -> UIViewController?  
  
public func webView(_ webView: WKWebView,  
                    commitPreviewingViewController previewingViewController: UIViewController)
```

Peek & Pop

Bring your own UI!

Peek & Pop

Bring your own UI!

UIPreviewInteraction

Peek & Pop

Bring your own UI!

UIPreviewInteraction

UIKit provides the “feel” of Peek & Pop

Peek & Pop

Bring your own UI!

UIPreviewInteraction

UIKit provides the “feel” of Peek & Pop

```
func previewInteraction(_ previewInteraction: UIPreviewInteraction,  
                      didUpdatePreviewTransition transitionProgress: CGFloat, ended: Bool) {  
  
    self.updateUITo.Peek(transitionProgress)  
    if ended {  
        self.show.PeekUI()  
    }  
}
```

Peek & Pop

Bring your own UI!

UIPreviewInteraction

UIKit provides the “feel” of Peek & Pop

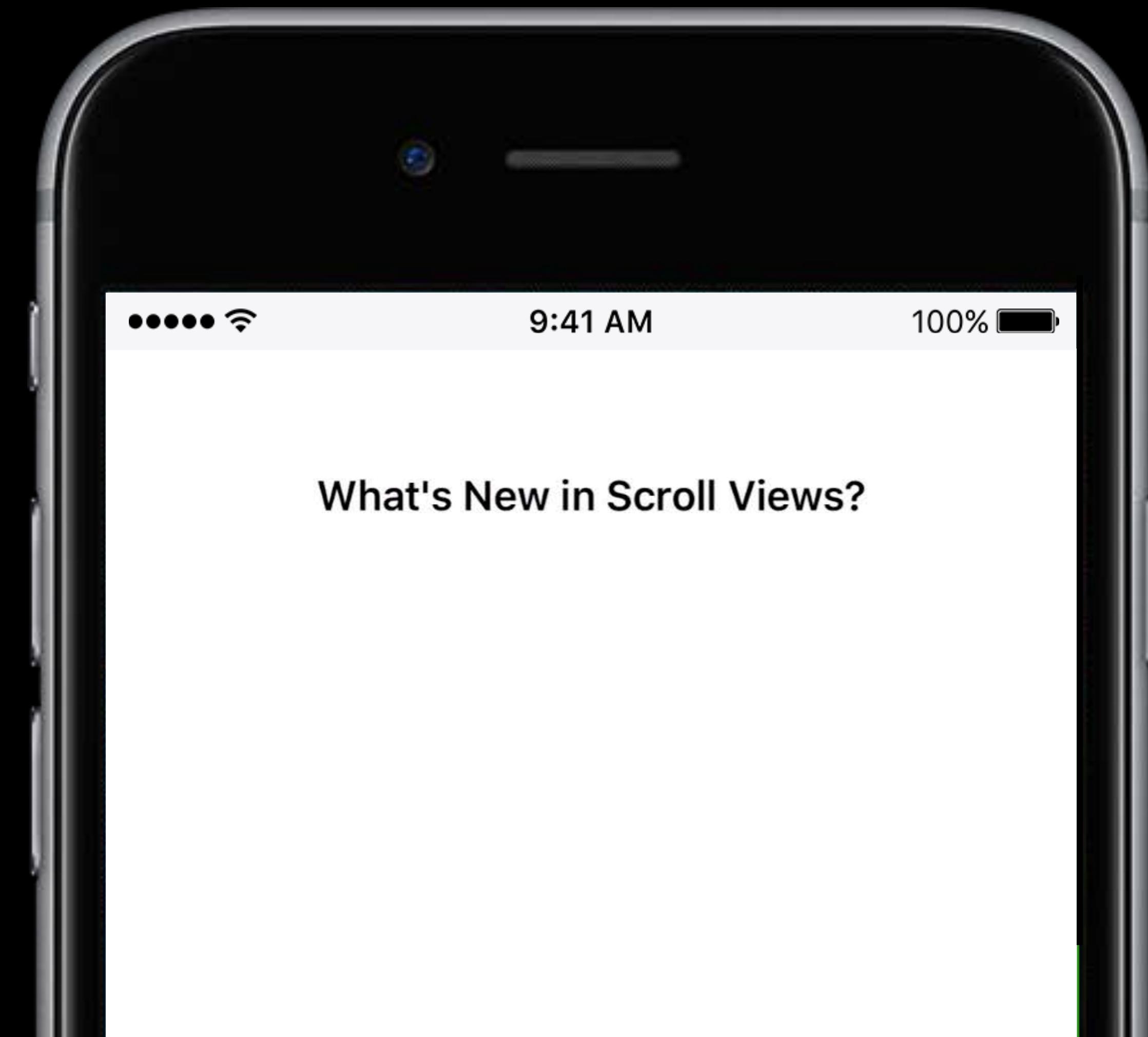
```
func previewInteraction(_ previewInteraction: UIPreviewInteraction,  
                      didUpdatePreviewTransition transitionProgress: CGFloat, ended: Bool) {  
  
    self.updateUIToPeek(transitionProgress)  
    if ended {  
        self.showPeekUI()  
    }  
}
```



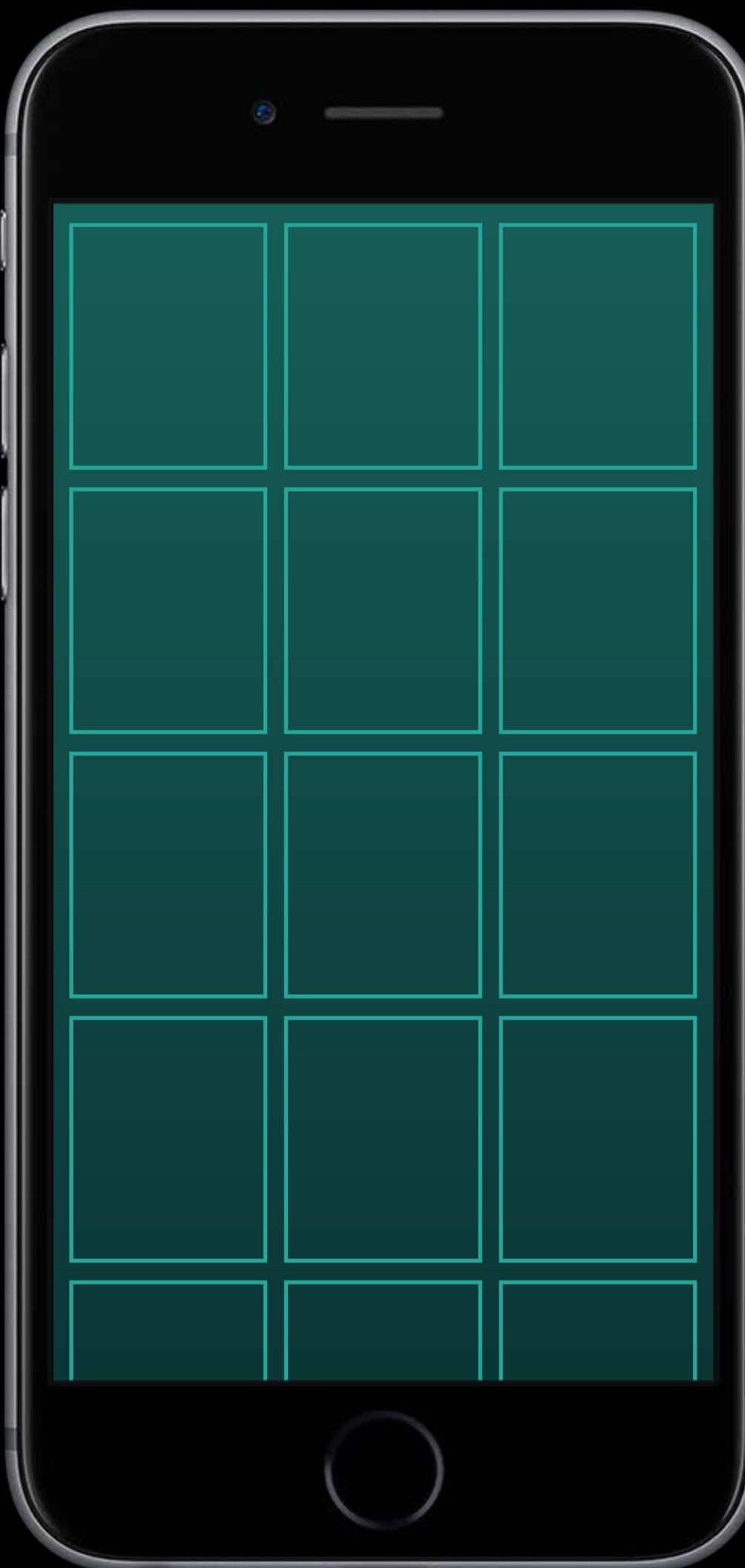


Refresh Control

Support for UIScrollView and subclasses

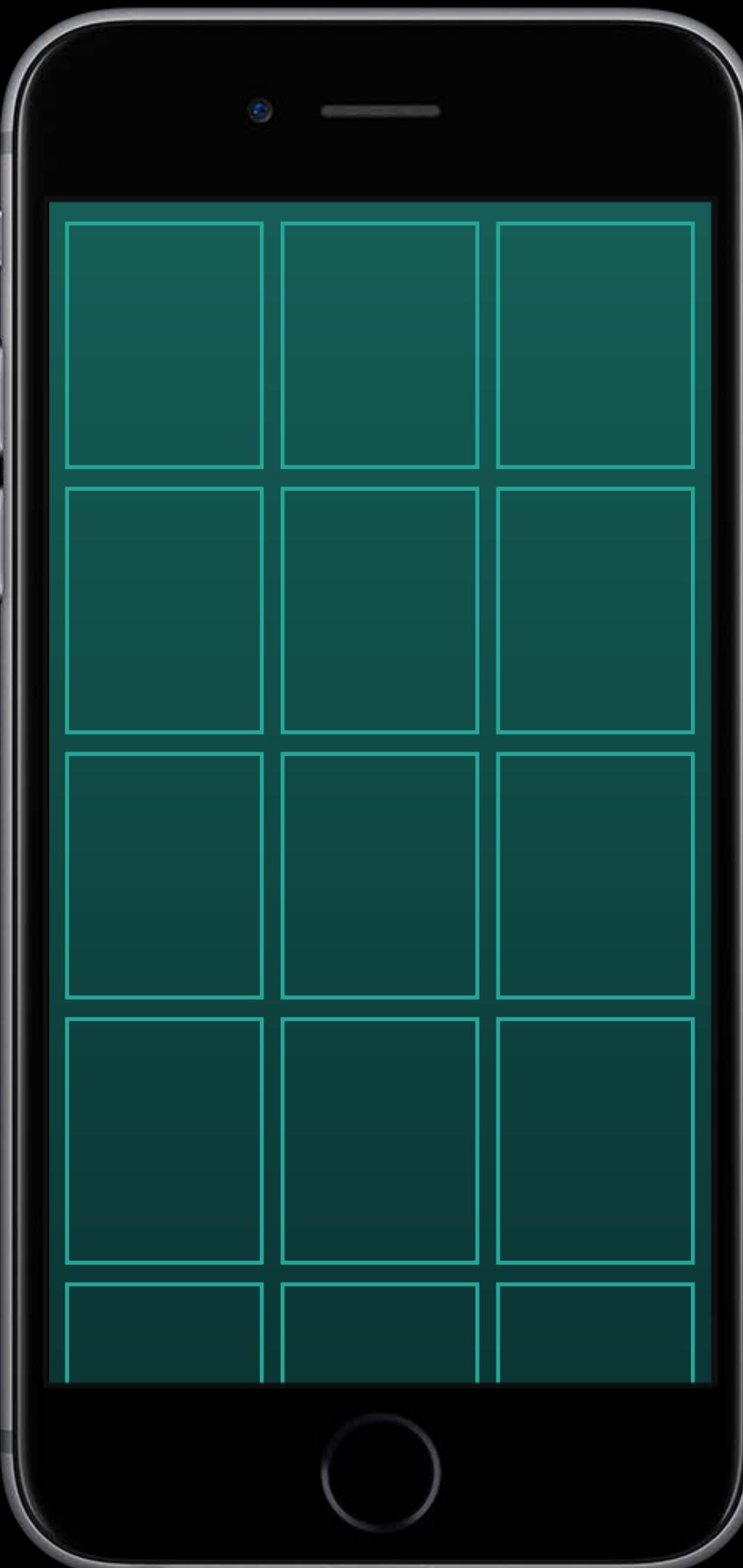


CollectionView



CollectionView

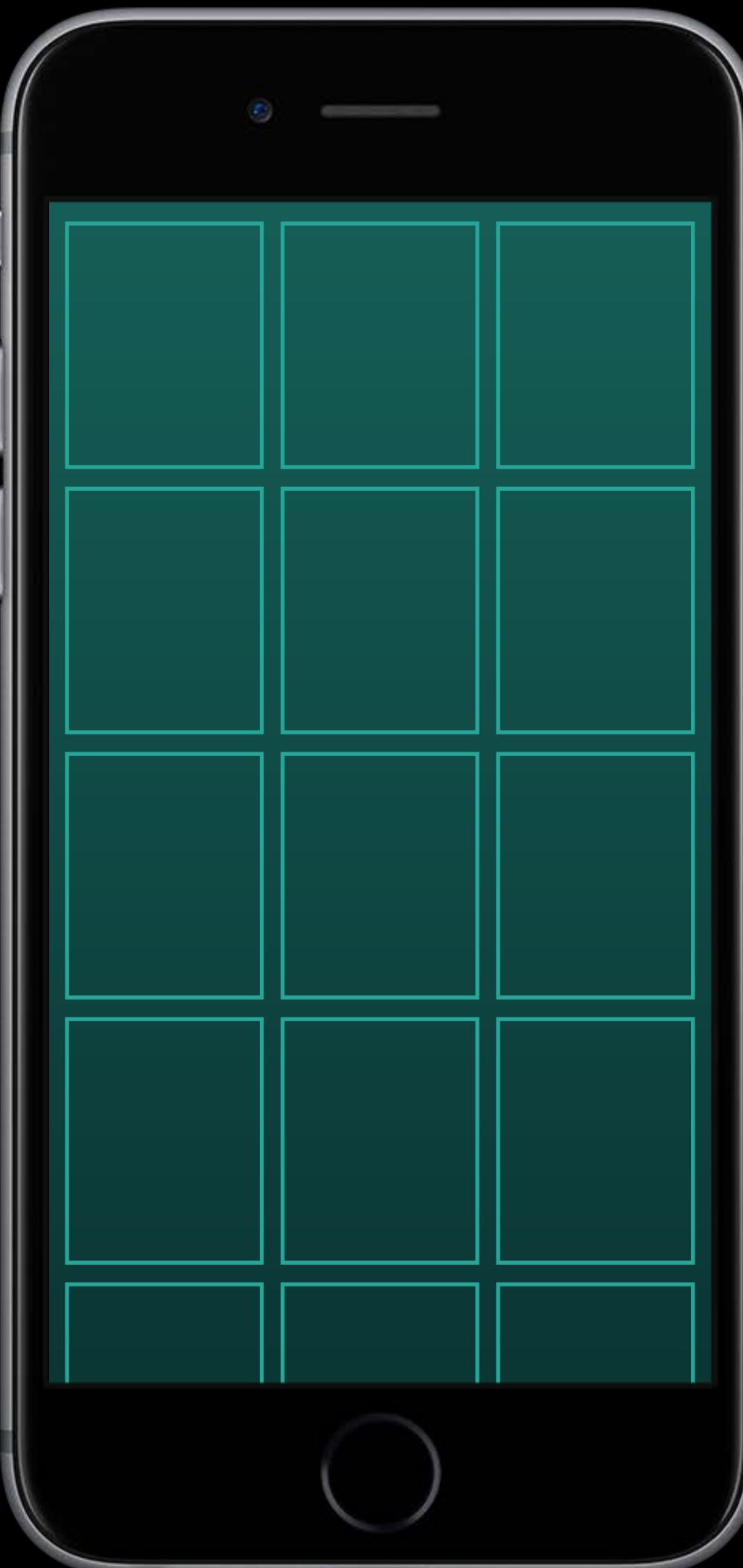
Automatic self-sizing cells
in flow layout



CollectionView

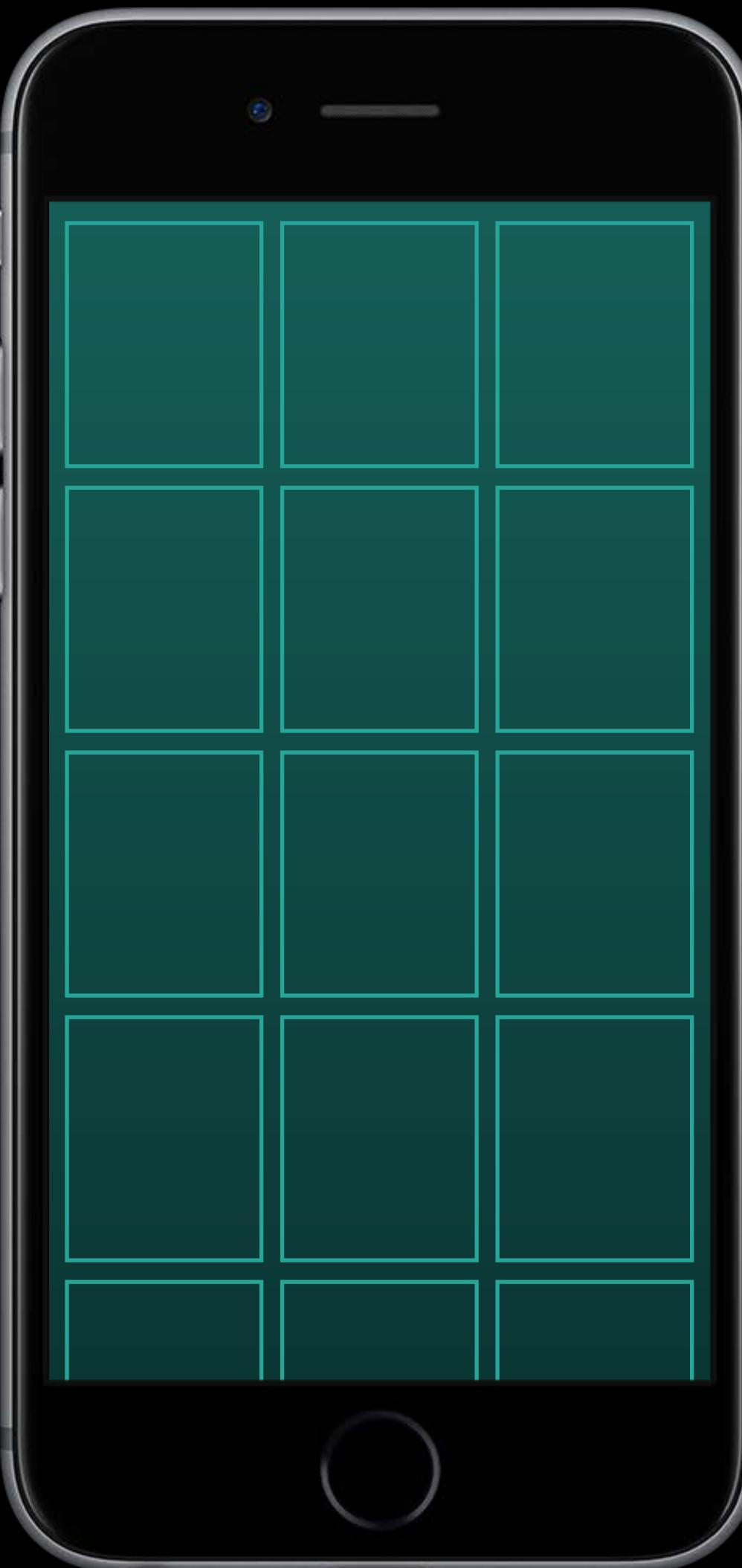
Automatic self-sizing cells
in flow layout

Paging support in collection
view reordering



CollectionView

Smooth scrolling



CollectionView

NEW

Smooth scrolling

Cell prefetching



CollectionView

NEW

Smooth scrolling

Cell prefetching

Data prefetching



CollectionView

NEW

Smooth scrolling

Cell prefetching

Data prefetching

(also available in UITableView)



CollectionView

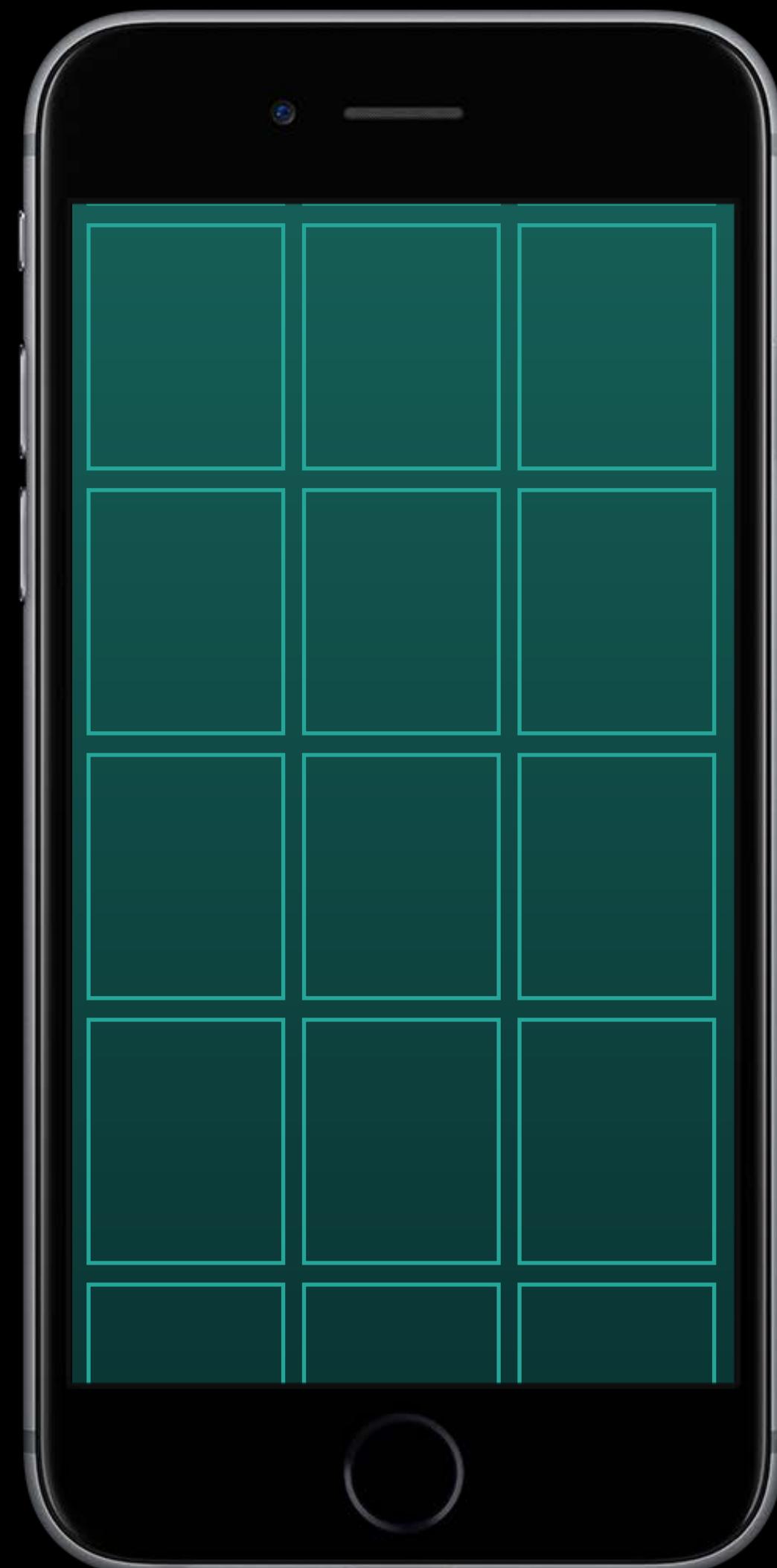
NEW

Smooth scrolling

Cell prefetching

Data prefetching

(also available in UITableView)



Advances in UIKit Animations

UIViewPropertyAnimator

Advances in UIKit Animations

UIViewPropertyAnimator

Interruptible

Advances in UIKit Animations

UIViewPropertyAnimator

Interruptible

Scrabbleable

Advances in UIKit Animations

UIViewControllerAnimated

Interruptible

Scrabbleable

Reversible

Advances in UIKit Animations

UIViewPropertyAnimator

Interruptible

Scrabbleable

Reversible

Rich timing features

Advances in UIKit Animations

UIViewControllerAnimated

Interruptible

Scrabbleable

Reversible

Rich timing features

Dynamic

Advances in UIKit Animations

```
let timing = UICubicTimingParameters(animationCurve: .easeInOut)
let animator = UIViewPropertyAnimator(duration: duration, timingParameters: timing)

animator.addAnimations {
    self.squareView.center = CGPoint(x: point.x, y: point.y)
}

animator.startAnimation()
```

Advances in UIKit Animations

```
let timing = UICubicTimingParameters(animationCurve: .easeInOut)
let animator = UIViewPropertyAnimator(duration: duration, timingParameters: timing)

animator.addAnimations {
    self.squareView.center = CGPoint(x: point.x, y: point.y)
}

animator.startAnimation()
```









Adopting System Features



Opening applications



Opening applications

Improving openURL



Opening applications

Improving openURL

- Asynchronous, with a completion handler



Opening applications

Improving openURL

- Asynchronous, with a completion handler
- Let you check if a handler app is installed for universal links



Opening applications

Improving openURL

- Asynchronous, with a completion handler
- Let you check if a handler app is installed for universal links



```
UIApplication.shared().  
    open(url, options: [UIApplicationOpenURLOptionUniversalLinksOnly: true]) {  
        (didOpen: Bool) in  
            if !didOpen {  
                // No application available  
            }  
    }  
}
```

Core Data



Core Data

Query generations



Core Data

Query generations

Concurrency improvements



Core Data

Query generations

Concurrency improvements

Tooling improvements

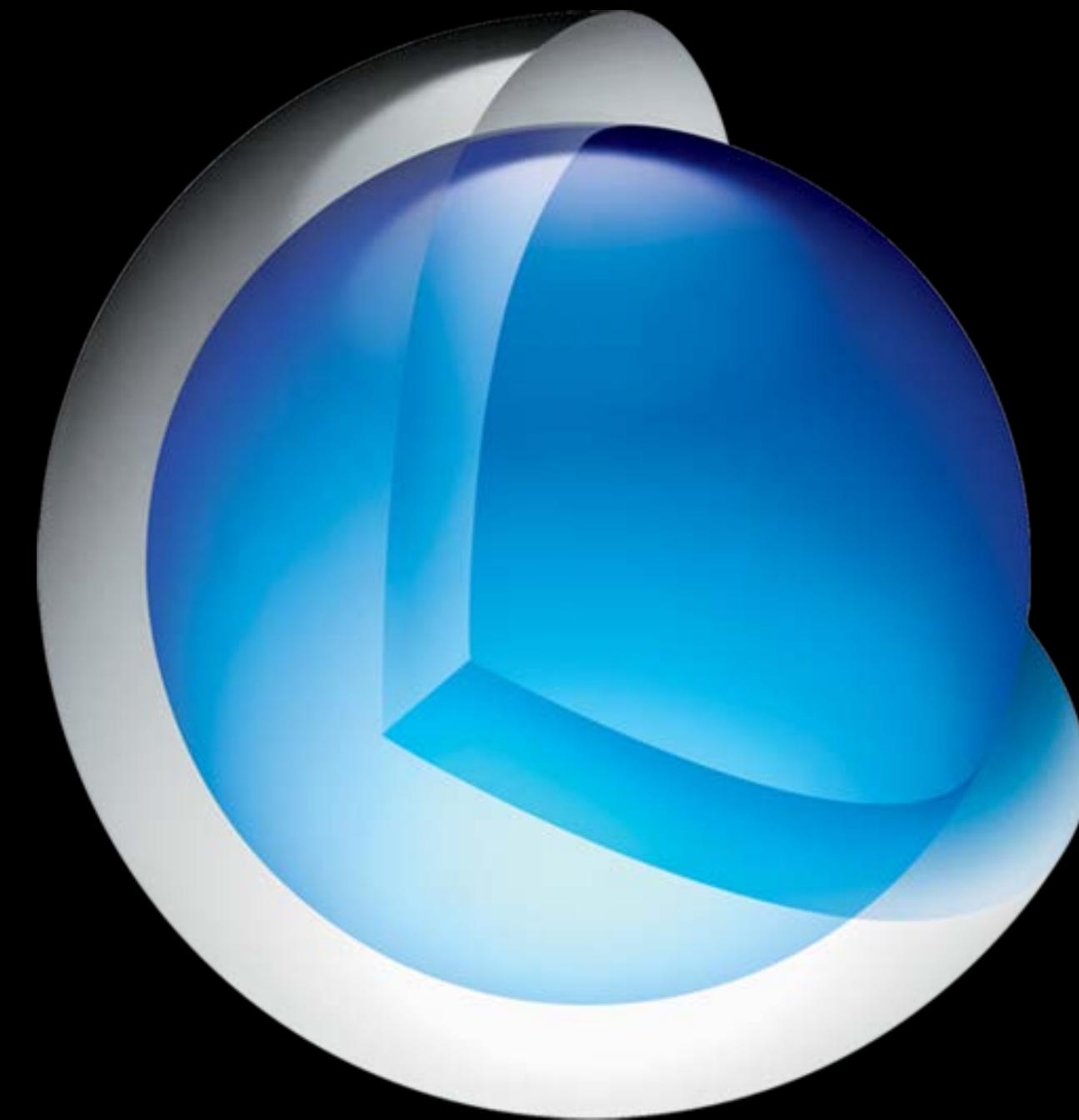


Core Data

Query generations

Concurrency improvements

Tooling improvements



CloudKit



CloudKit

Public databases

Private, per user databases



CloudKit

NEW

Public databases

Private, per user databases

Record sharing



CloudKit

UICloudSharingController

NEW



CloudKit

NEW

UICloudSharingController

Managing the invitation flow



CloudKit

NEW

UICloudSharingController

Managing the invitation flow

Private and secure



CloudKit

NEW

UICloudSharingController

Managing the invitation flow

Private and secure



```
let sharingController = UICloudSharingController(share: share, container: self.container)
```

iPad

Notes Edit

Family Grocery list
1:21 PM Dad is going to Trader Joe's on...

Summer camp details
5/25/16 Hidden Village

Island Lake Camping
5/25/16 May 15th - 19th

Soccer practice times
5/25/16 Zoe

Directions to Gavin's party
5/25/16 1 web link

New dishwasher options
5/25/16 Bosch 500 series

June trip to Santa Rosa
5/25/16 June 18 and 19.

Staff Meeting
5/25/16 June 1st.

Meeting Notes
5/25/16 Trestella project

Weekend away
5/25/16 Big Sur - July 20th

14 Notes

1:26 PM

Family Grocery list

Dad is going to Trader Joe's on T

Don't forget the bbq this Saturday. Ar

Milk - gallon

Butter

Eggs - two dozen

5 racks baby back ribs

Avocados (6)

Corn chips

Tomato sauce

Share this with others and everyone will see the latest changes.

Add People

Family Grocery list

Derek Parker (derek_wwdc16@icloud.com)

Choose how you'd like to send your invitation:

Message

Mail

Copy Link

Twitter

People you invite can make changes

iPad

Notes Edit

Family Grocery list
1:21 PM Dad is going to Trader Joe's on...

Summer camp details
5/25/16 Hidden Village

Island Lake Camping
5/25/16 May 15th - 19th

Soccer practice times
5/25/16 Zoe

Directions to Gavin's party
5/25/16 1 web link

New dishwasher options
5/25/16 Bosch 500 series

June trip to Santa Rosa
5/25/16 June 18 and 19.

Staff Meeting
5/25/16 June 1st.

Meeting Notes
5/25/16 Trestella project

Weekend away
5/25/16 Big Sur - July 20th

14 Notes

1:26 PM Not Charging

Family Grocery list

Dad is going to Trader Joe's on T

Don't forget the bbq this Saturday. Ar

Milk - gallon

Butter

Eggs - two dozen

5 racks baby back ribs

Avocados (6)

Corn chips

Tomato sauce

Add People

Share this with others and everyone will see the latest changes.

.....
.....
.....
.....

Family Grocery list
Derek Parker (derek_wwdc16@icloud.com)

Choose how you'd like to send your invitation:

Message Mail Copy Link Twitter

People you invite can make changes

iPad

Notes Edit

Family Grocery list

1:21 PM Dad is going to Trader Joe's on...

Summer camp details

5/25/16 Hidden Village

Island Lake Camping

5/25/16 May 15th - 19th

Soccer practice times

5/25/16 Zoe

Directions to Gavin's party

5/25/16 1 web link

New dishwasher options

5/25/16 Bosch 500 series

June trip to Santa Rosa

5/25/16 June 18 and 19.

Staff Meeting

5/25/16 June 1st.

Meeting Notes

5/25/16 Trestella project

Weekend away

5/25/16 Big Sur - July 20th

14 Notes

1:23 PM

Family Grocery list

Dad is going to Trader Joe's on T

Don't forget the bbq this Saturday. Ar

- Milk - gallon
- Butter
- Eggs - two dozen
- 5 racks baby back ribs
- Avocados (6)
- Corn chips
- Tomato sauce

People

People you invite can make changes

Derek Parker (Owner)

Emily Parker

Add People

Copy Link

Stop Sharing

CloudKit



What's New with CloudKit

Presidio

Thursday 3:00PM

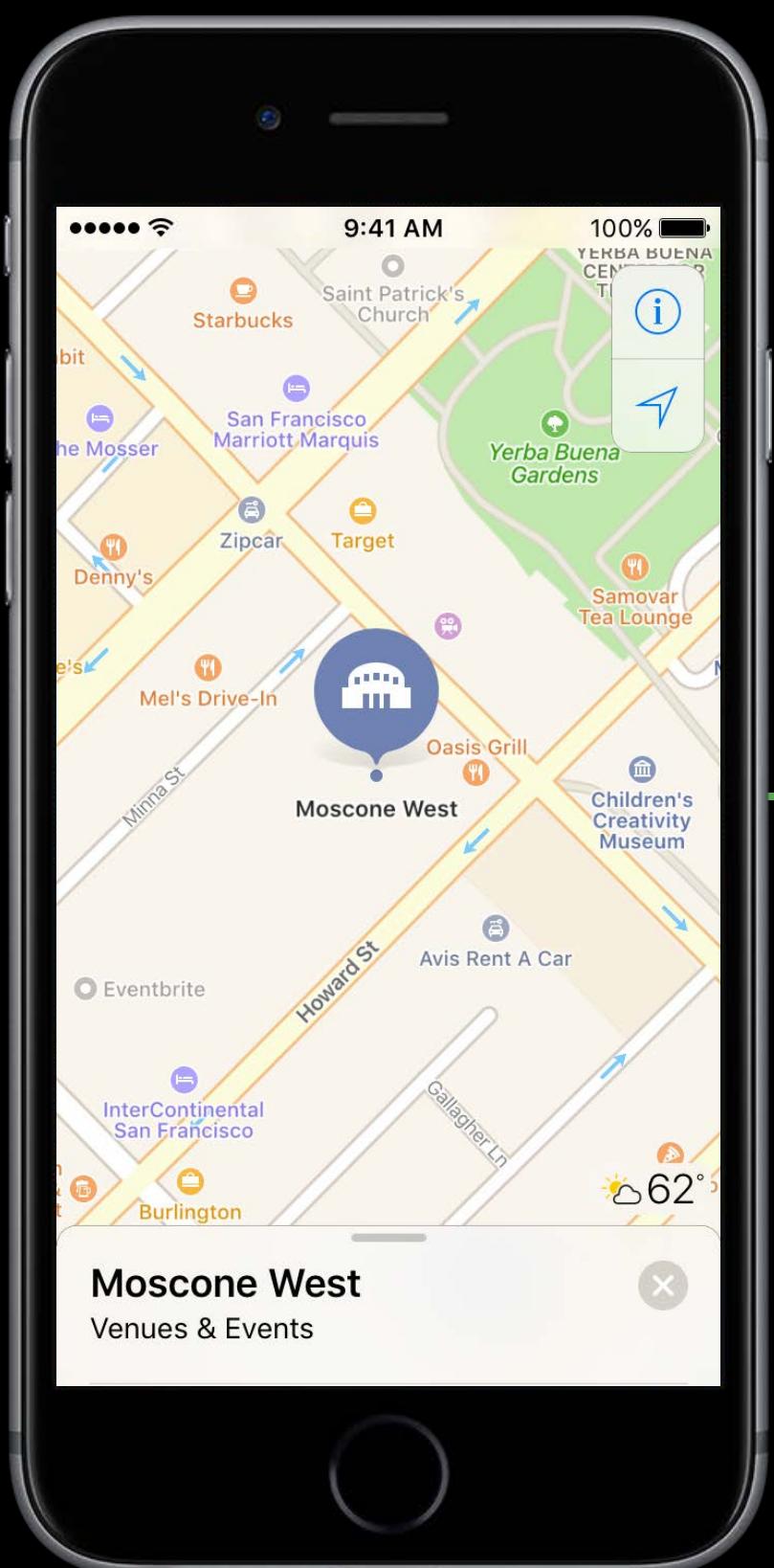
NSUserActivity

NSUserActivity

Capture the state of your application

NSUserActivity

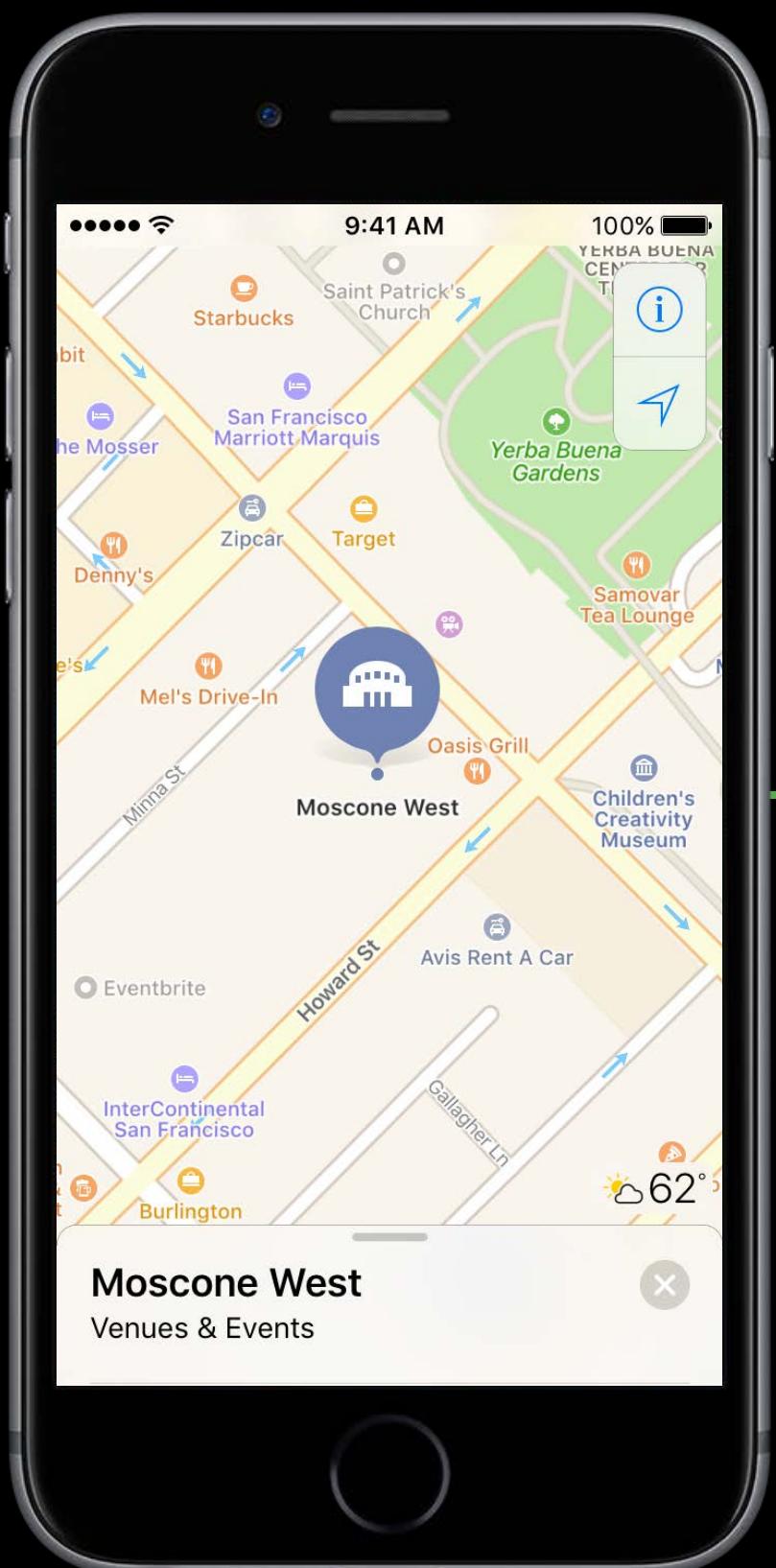
Capture the state of your application



NSUserActivity

NSUserActivity

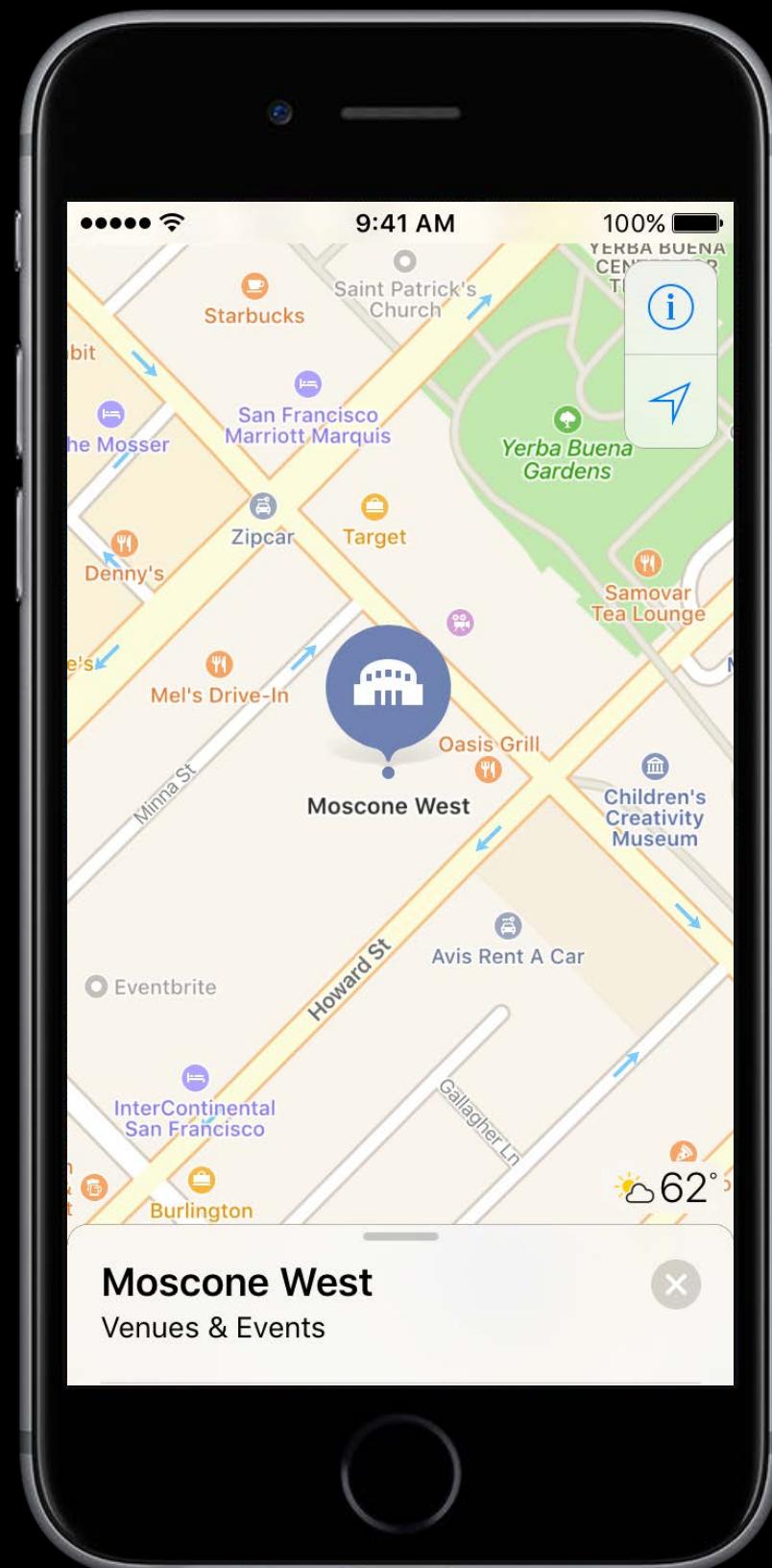
Capture the state of your application
Infrastructure for Handoff, Spotlight, ...



NSUserActivity

NSUserActivity

Capture the state of your application
Infrastructure for Handoff, Spotlight, ...
Now supports locations

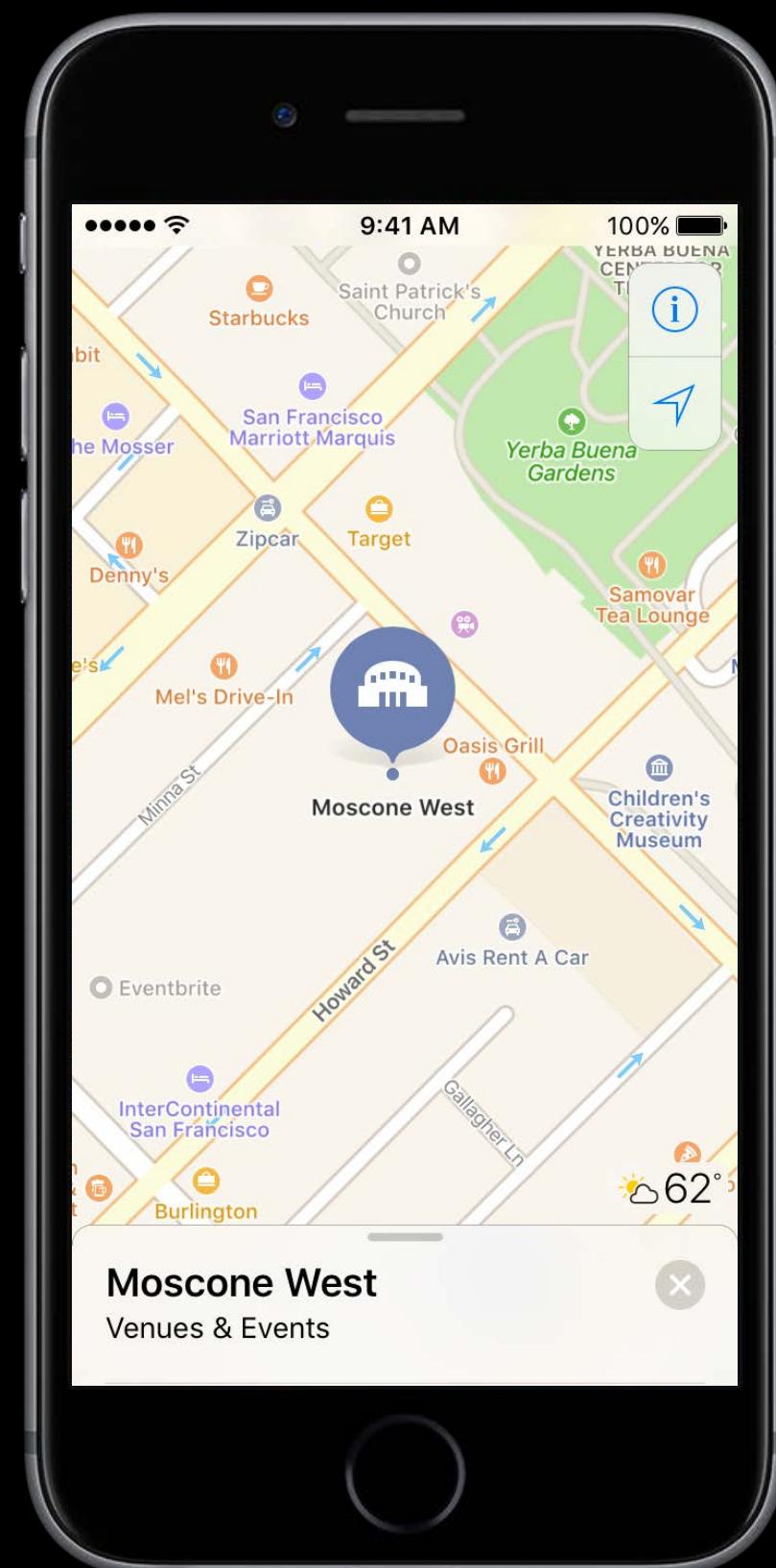


NSUserActivity

NSUserActivity

Capture the state of your application
Infrastructure for Handoff, Spotlight, ...
Now supports locations

```
activity.mapItem = myLocation
```



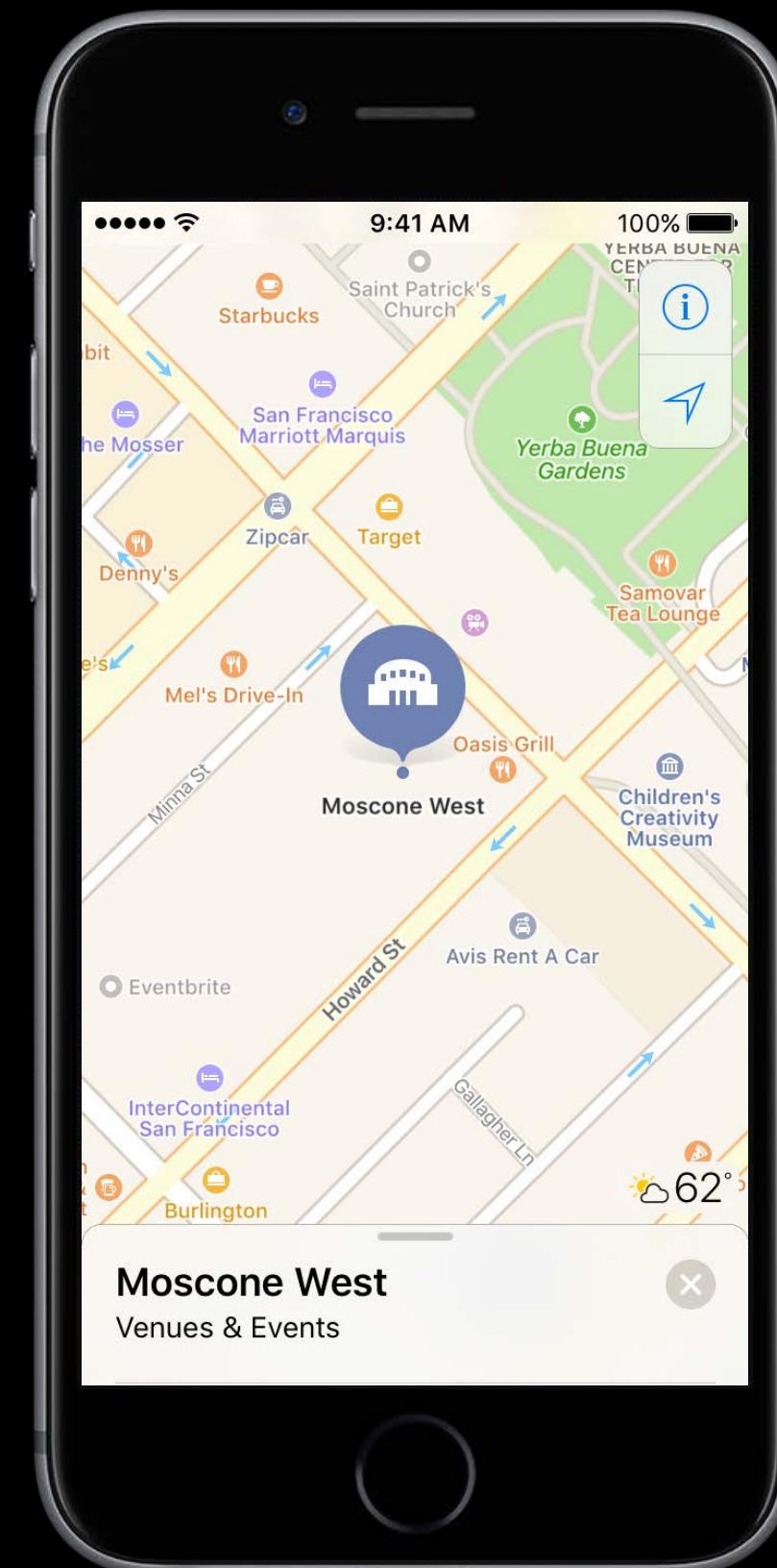
NSUserActivity

NSUserActivity

Capture the state of your application
Infrastructure for Handoff, Spotlight, ...

Now supports locations

```
activity.mapItem = myLocation
```



NSUserActivity

App Search

In iOS 9, we added support for indexed activities and indexed content

App Search

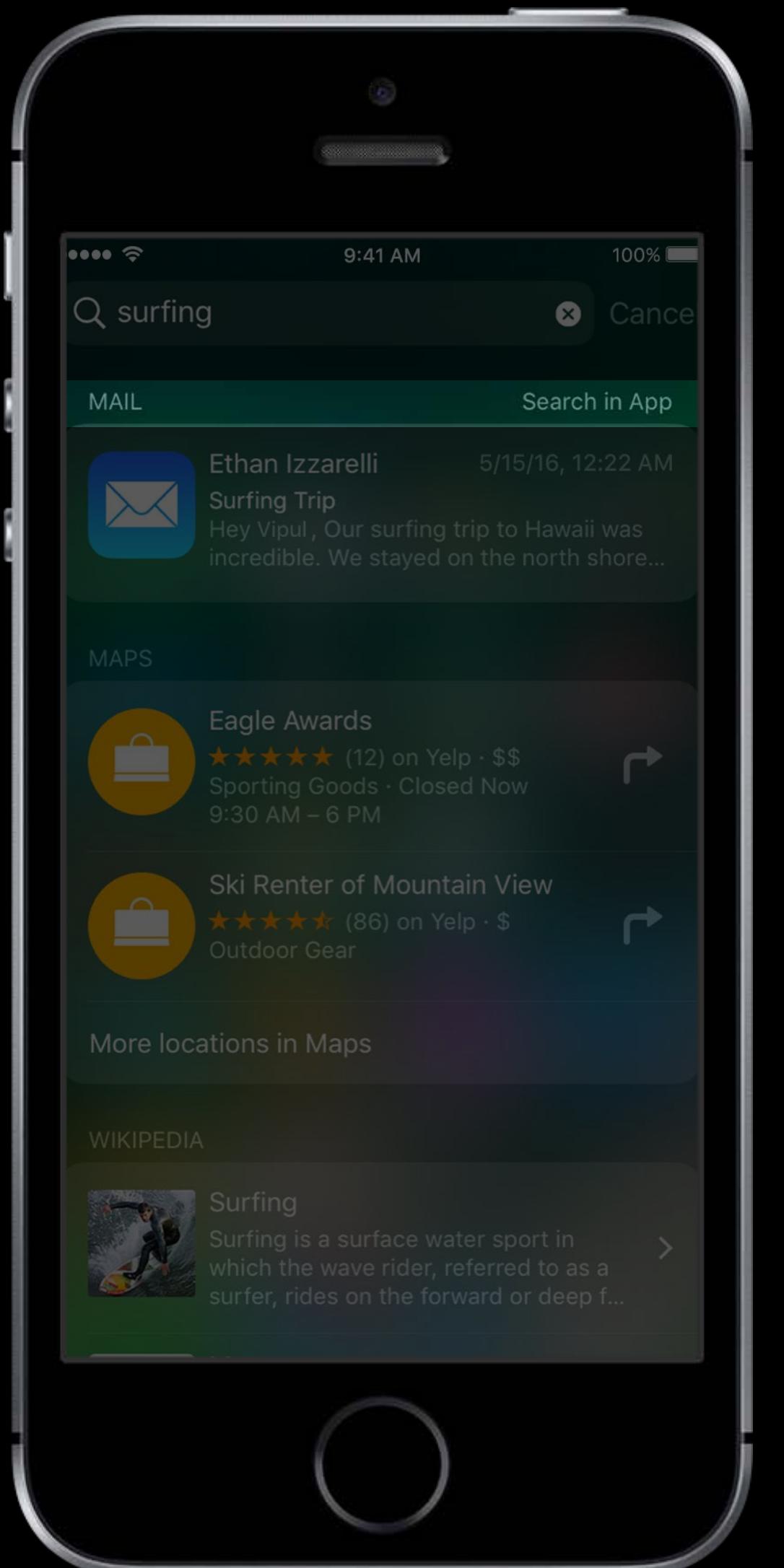
In iOS 9, we added support for indexed activities and indexed content

```
let userActivity = NSUserActivity(activityType: "myActivityType")
userActivity.eligibleForSearch = true
userActivity.eligibleForPublicIndexing = true
userActivity.title = "Presenting What's New in Cocoa Touch"
```

```
let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = ...
userActivity.contentAttributeSet = attributes
```

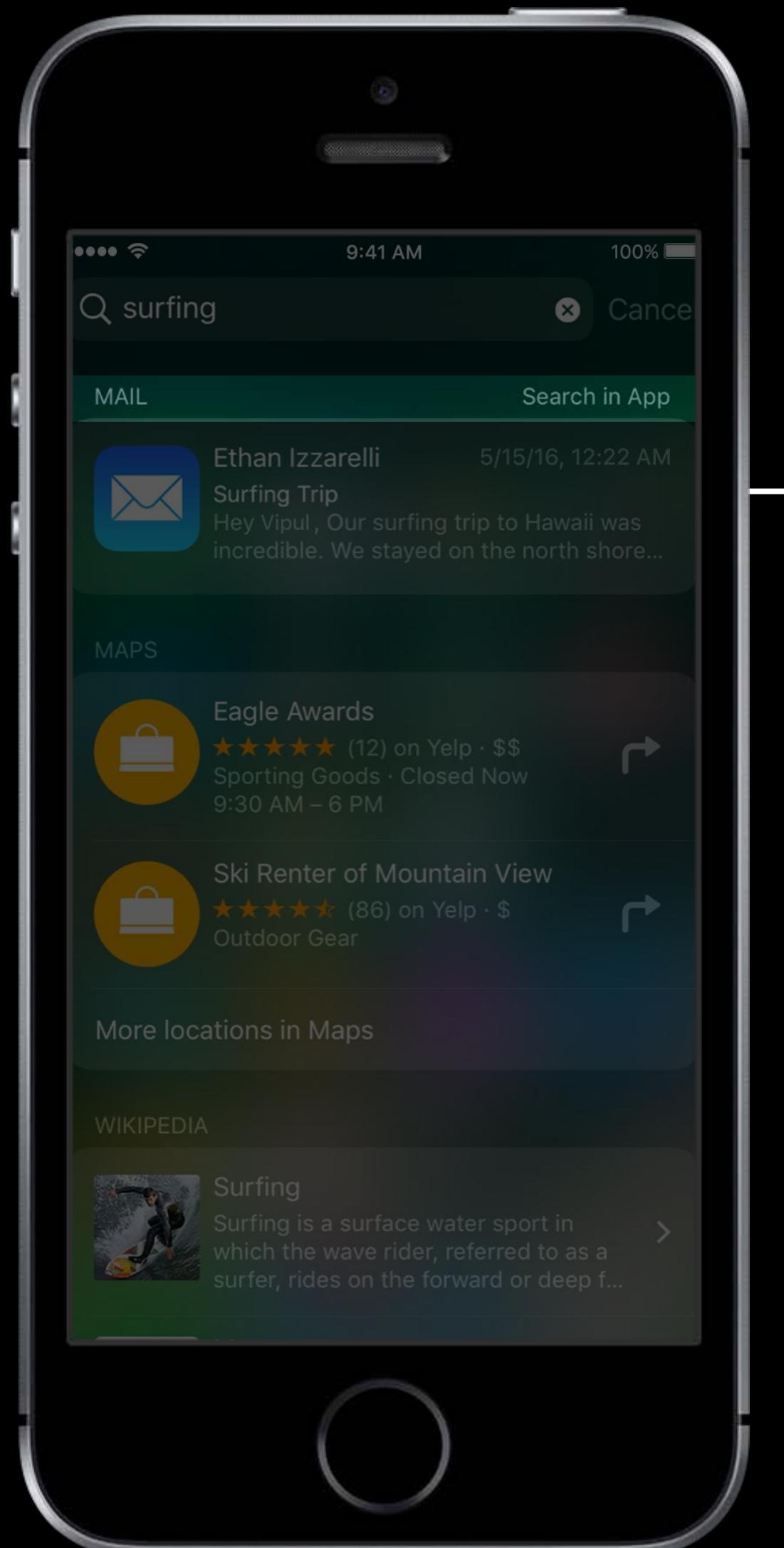
App Search

NEW



App Search

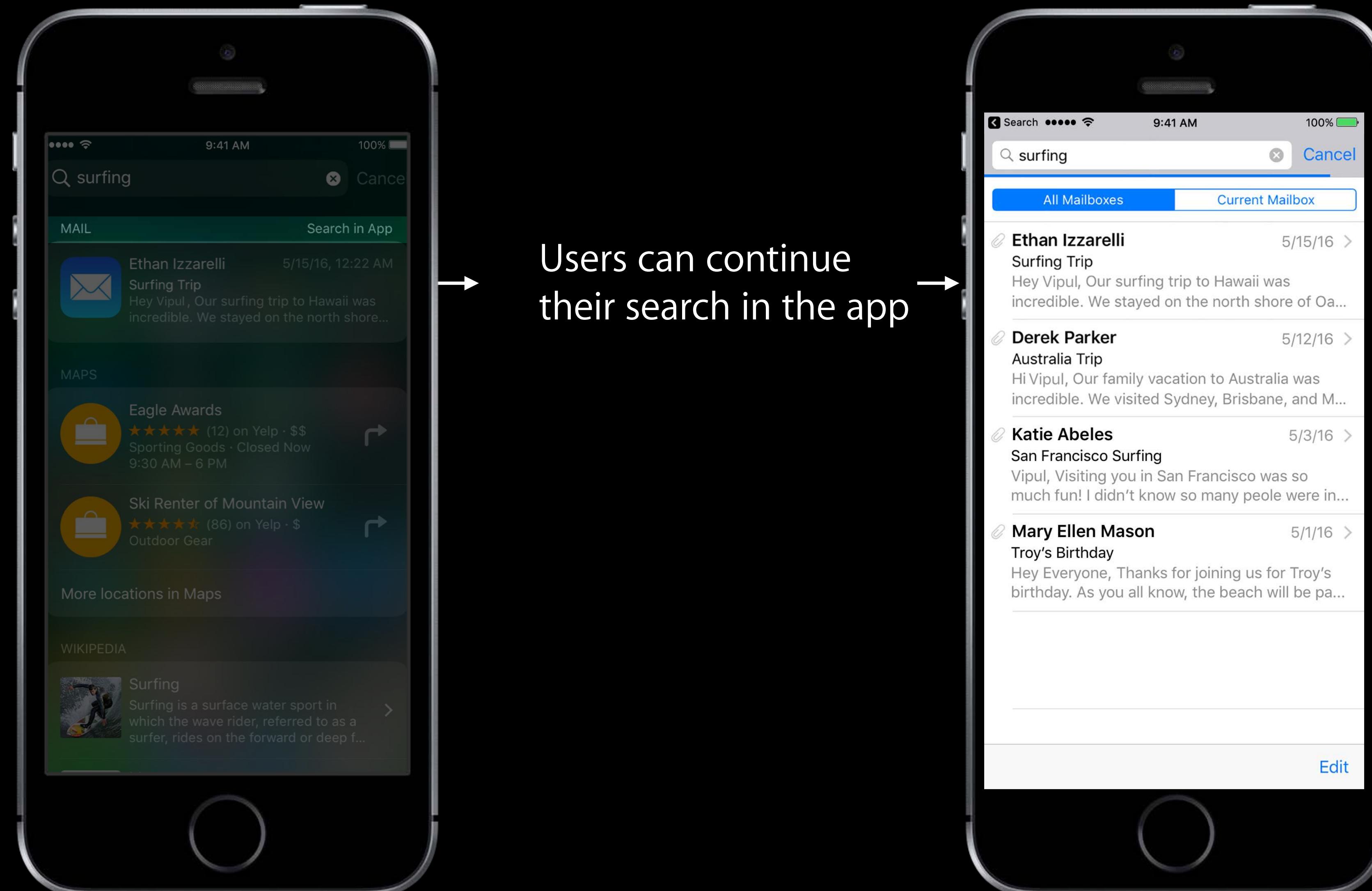
NEW



Users can continue
their search in the app

App Search

NEW



App Search

App Search

Add a `CoreSpotlightContinuation` key in your Info plist

App Search

Add a `CoreSpotlightContinuation` key in your Info plist

Implement a new UIApplicationDelegate method

App Search

Add a `CoreSpotlightContinuation` key in your Info.plist

Implement a new UIApplicationDelegate method

```
func application(application: UIApplication,
                 continueUserActivity userActivity: NSUserActivity,
                 restorationHandler: ([AnyObject]?) -> Void) -> Bool {
    if userActivity.activityType == CSQueryContinuationActionType {
        if let searchQuery = userActivity.userInfo?[CSSearchQueryString] as? String {
            // Search
        }
        return true
    }
    return false
}
```

CoreSpotlight Search API

CSSearchQuery

CoreSpotlight Search API

CSSearchQuery

Search the data you've already indexed with Spotlight

CoreSpotlight Search API

CSSearchQuery

Search the data you've already indexed with Spotlight

Great power and performance, full content search

CoreSpotlight Search API

CSSearchQuery

Search the data you've already indexed with Spotlight

Great power and performance, full content search

Powerful query syntax

CoreSpotlight Search API

CSSearchBar

Search the data you've already indexed with Spotlight

Great power and performance, full content search

Powerful query syntax

```
let query = CSSearchBar(queryString: queryString, attributes: ["displayName"])

query.foundItemsHandler = {
    (items: [CSSearchableItem]) in
        /* process received items */
}

query.start()
```

CoreSpotlight Search API

CSSearchBar

Search the data you've already indexed with Spotlight

Great power and performance, full content search

Powerful query syntax

```
let query = CSSearchBar(queryString: queryString, attributes: ["displayName"])

query.foundItemHandler = {
    (items: [CSSearchableItem]) in
        /* process received items */
}

query.start()
```

ReplayKit

RPBroadcastActivityViewController



ReplayKit

RPBroadcastActivityViewController

Now supports live broadcasting



ReplayKit

RPBroadcastActivityViewController

Now supports live broadcasting

Third-party services support



ReplayKit

RPBroadcastActivityViewController

Now supports live broadcasting

Third-party services support



SceneKit

Looping All the Sides

In this puzzle, Byte must collect four gems that are located in the same relative locations around a square. You'll create a **loop** that repeats the code below for each of the sides to solve the entire puzzle.

- ① Drag out a `for` loop from the code library, then drop it above the existing code.
- ② Tap the bottom curly brace to select the loop.
- ③ Tap and hold on that curly brace, then drag it downwards to pull the existing code into the loop.

```
for i in 1 ... number {
    moveForward()
    collectGem()
    moveForward()
    moveForward()

}

moveForward()
turnRight()
```

Run My Code Hint

for collectGem() moveForward() turnRight()

SceneKit

New realistic rendering

- Physically-based rendering
- High dynamic range
- Linear color space





NEW



Advances in SceneKit Rendering

Presidio

Tuesday 2:00PM

Apple Pay

Apple Pay

Currently

	In Apps	In Stores
iPhone		
Apple Watch		
iPad		

Apple Pay

NEW

Fall 2016

	In Apps	In Stores	Web	Extensions
iPhone				
Apple Watch				
iPad				
Mac				

Apple Pay

Apple Pay

Apple Pay in UI code

Apple Pay

NEW

Apple Pay in UI code

Apple Pay in Safari

Apple Pay

NEW

Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay

NEW

Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Apple Pay

NEW

Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps

Apple Pay

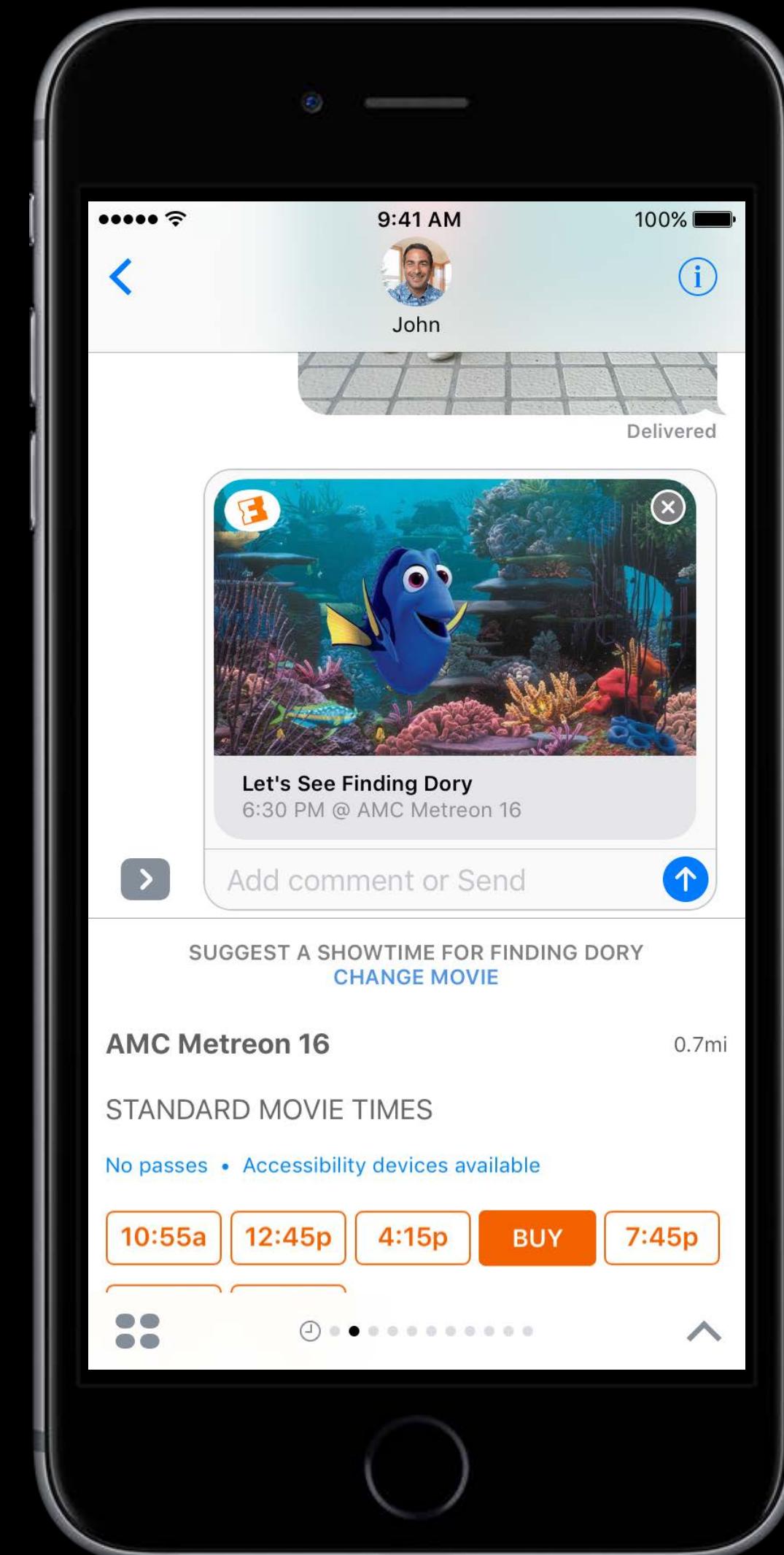
Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps



Apple Pay

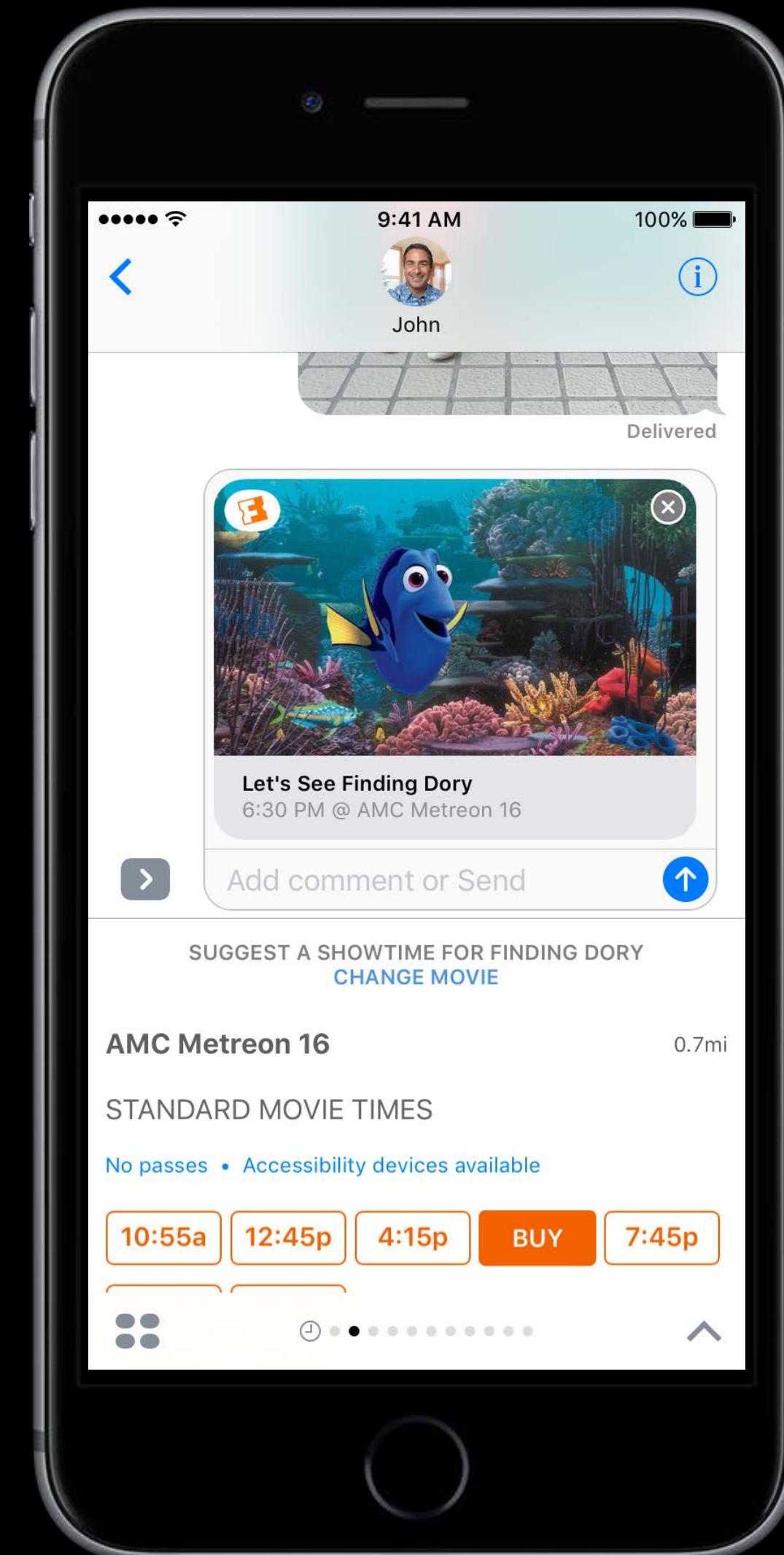
Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps



Apple Pay

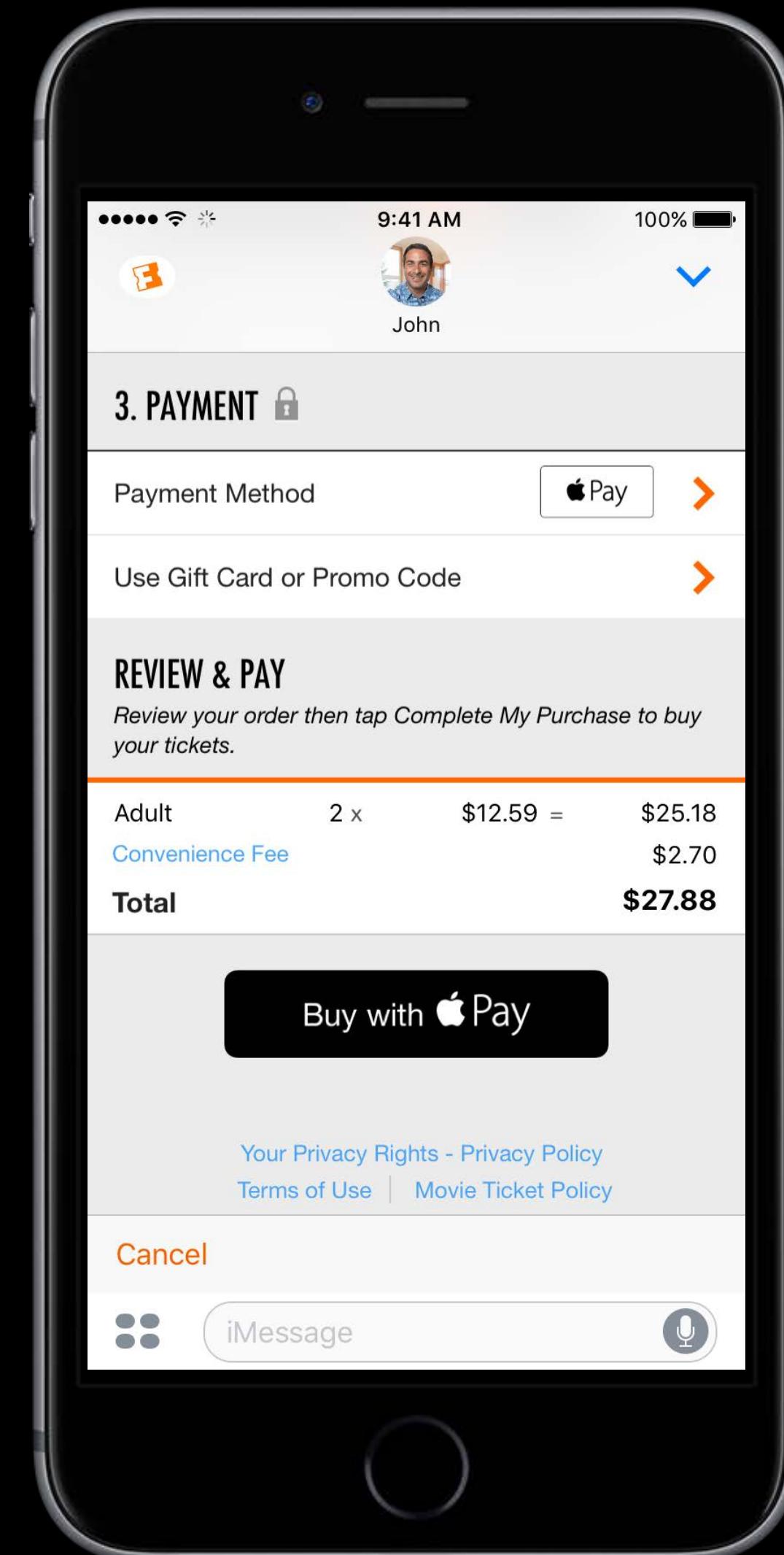
Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps



Apple Pay

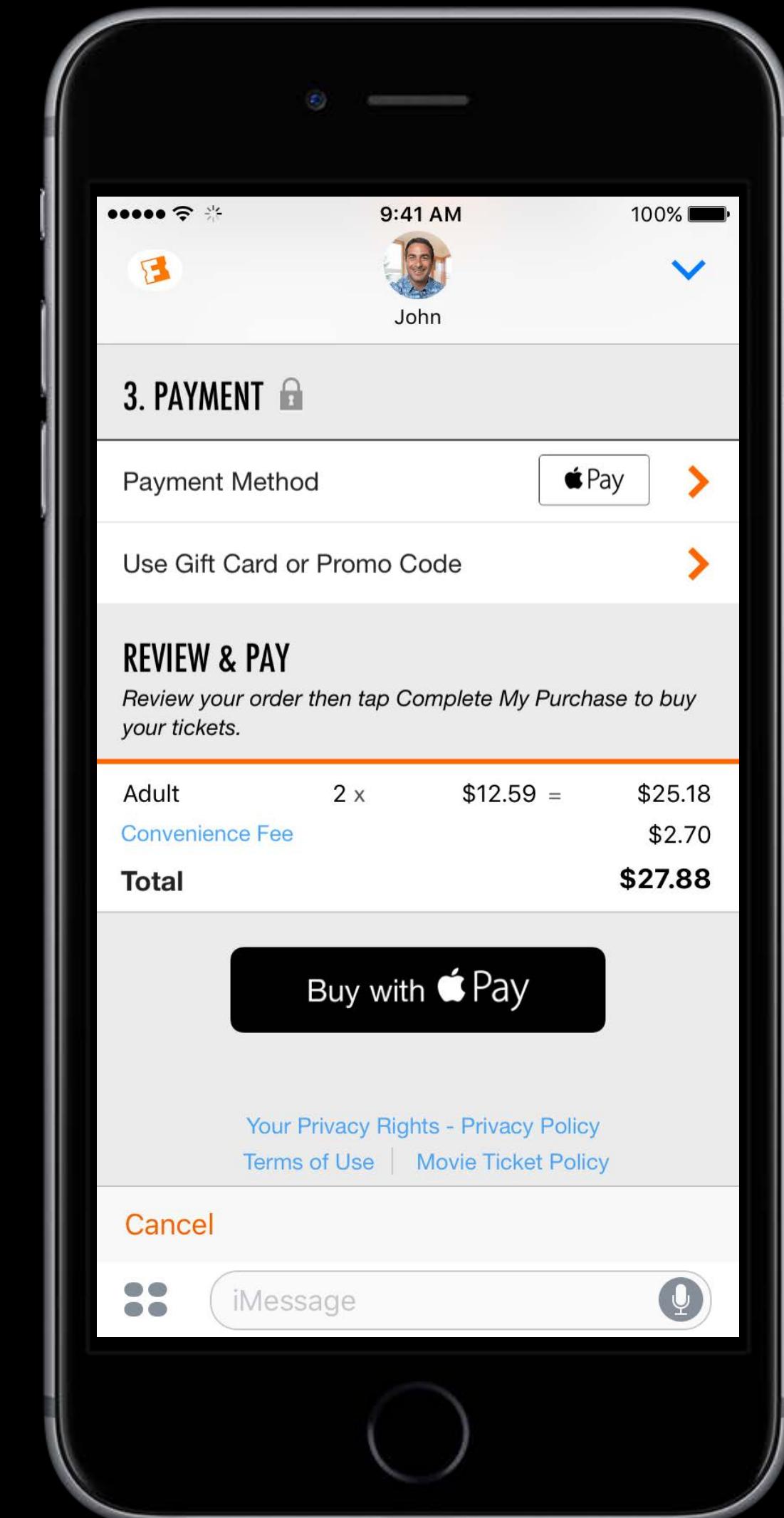
Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps



Apple Pay

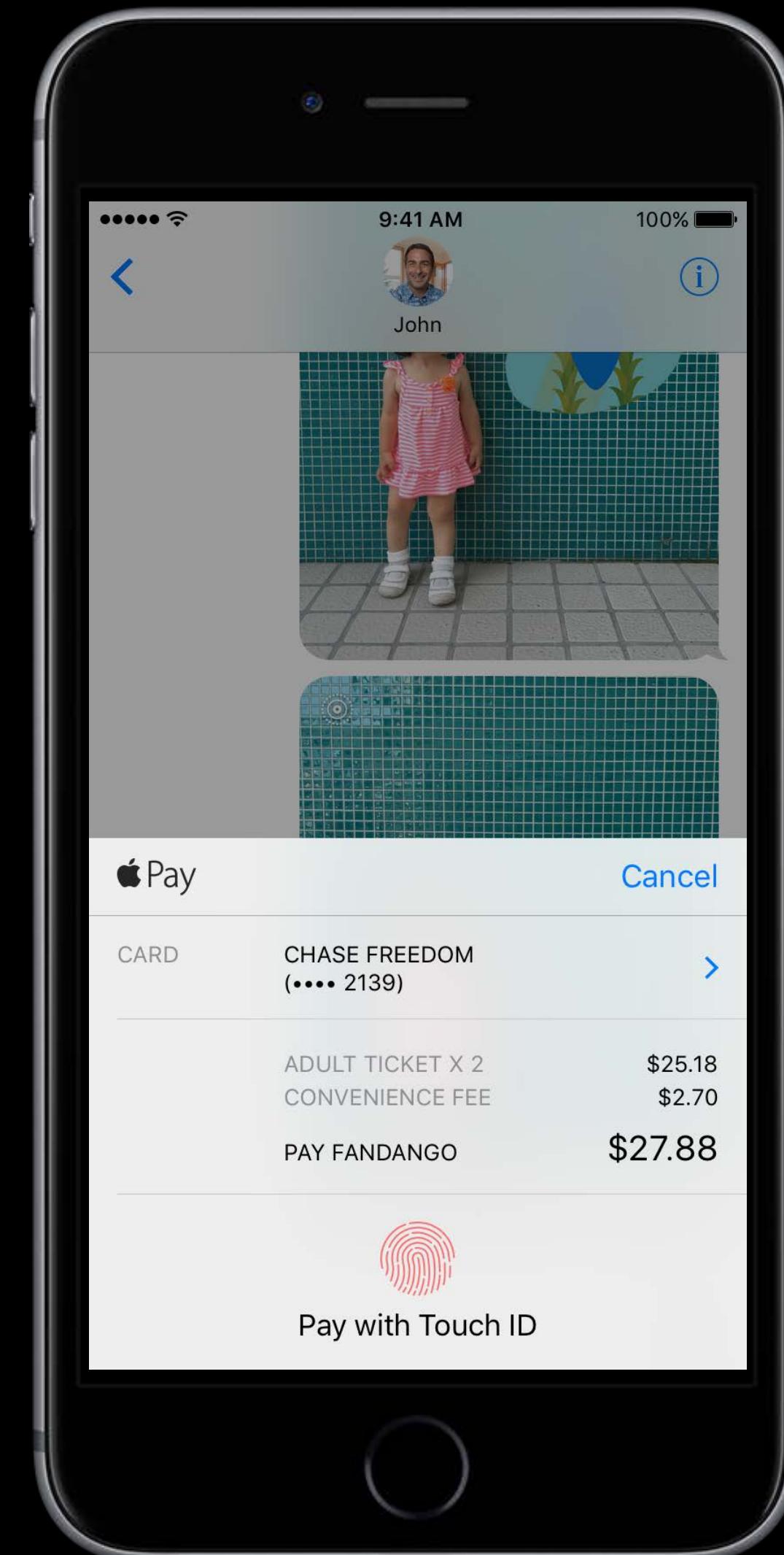
Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps



Apple Pay

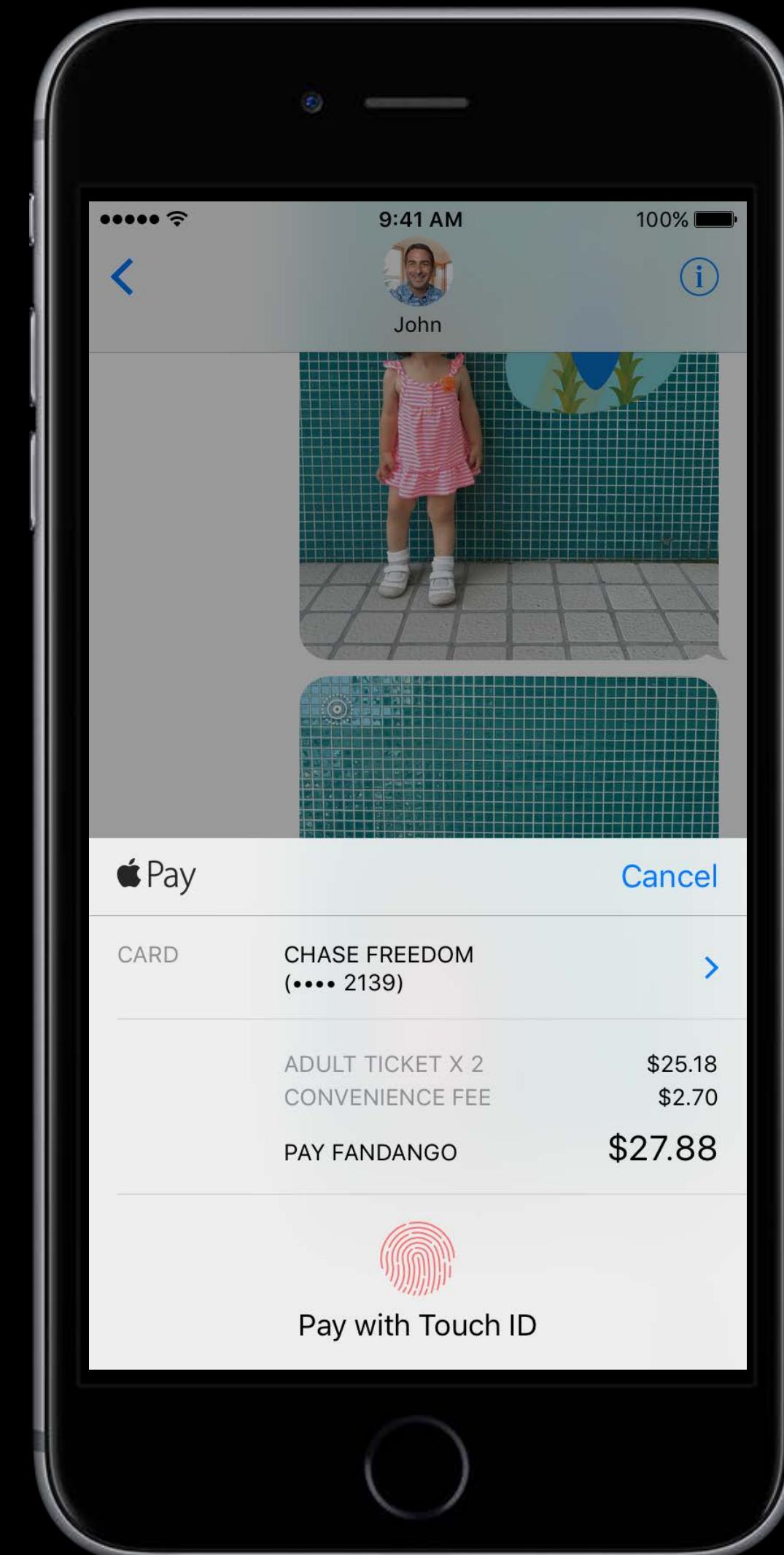
Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps



Apple Pay is available in the following contexts:

Apple Pay

Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps

Apple Pay on the Web

Mission

Tuesday 1:40PM

What's New with Wallet and Apple Pay

Mission

Tuesday 3:00PM

Apple Pay

Apple Pay in UI code

Apple Pay in Safari

- Also available in **SFSafariViewController**

Apple Pay in non-UI extensions

Great feature for your iMessage apps

Apple Pay on the Web

Mission

Tuesday 1:40PM

What's New with Wallet and Apple Pay

Mission

Tuesday 3:00PM

Integrating with iOS





Keyboards Extensions

Automatically switch your multi-language keyboard extension based on text content

Add system Globe Key functionality in your own keyboard extension



Widgets

Display modes

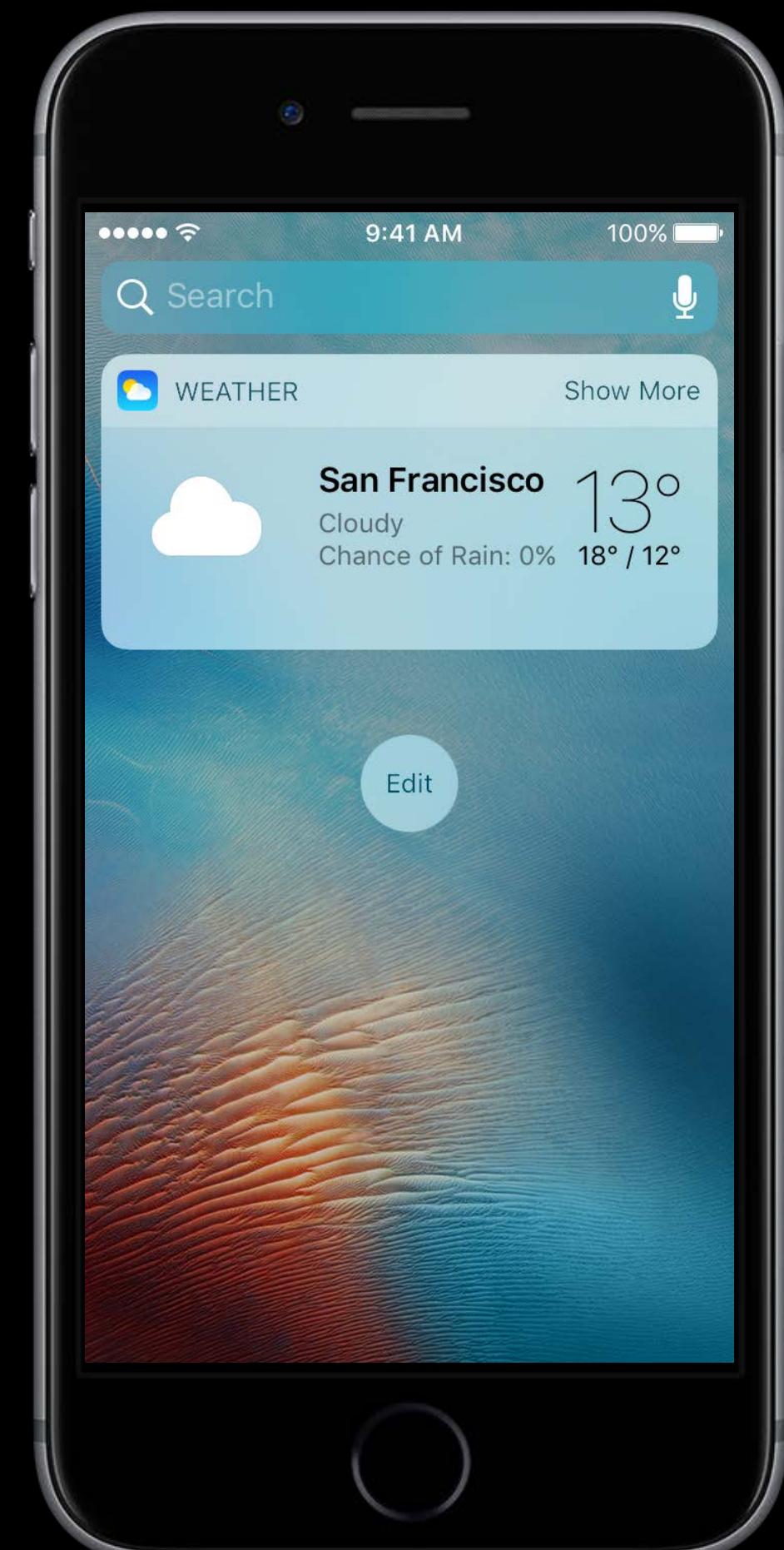
Widgets now have the concept
of “display modes”

Widgets

Display modes

Widgets now have the concept of “display modes”

- User-controlled
- Compact is fixed height



Widgets

Display modes

Widgets now have the concept of “display modes”

- User-controlled
- Compact is fixed height
- Expanded is variable



Widgets

Privacy best practices

Widgets

Privacy best practices

Your widget will be on the lock screen

Widgets

Privacy best practices

Your widget will be on the lock screen

Don't surprise your users

User Notifications



User Notifications

New Framework in iOS 10

User Notifications

New Framework in iOS 10



User Notifications

New Framework in iOS 10

Feature Parity



User Notifications

New Framework in iOS 10

Feature Parity

Unifies local and remote notification



User Notifications

New Framework in iOS 10

Feature Parity

Unifies local and remote notification

Better delivery management



User Notifications

New Framework in iOS 10

Feature Parity

Unifies local and remote notification

Better delivery management

In-app presentation option



User Notifications

New Framework in iOS 10

Feature Parity

Unifies local and remote notification

Better delivery management

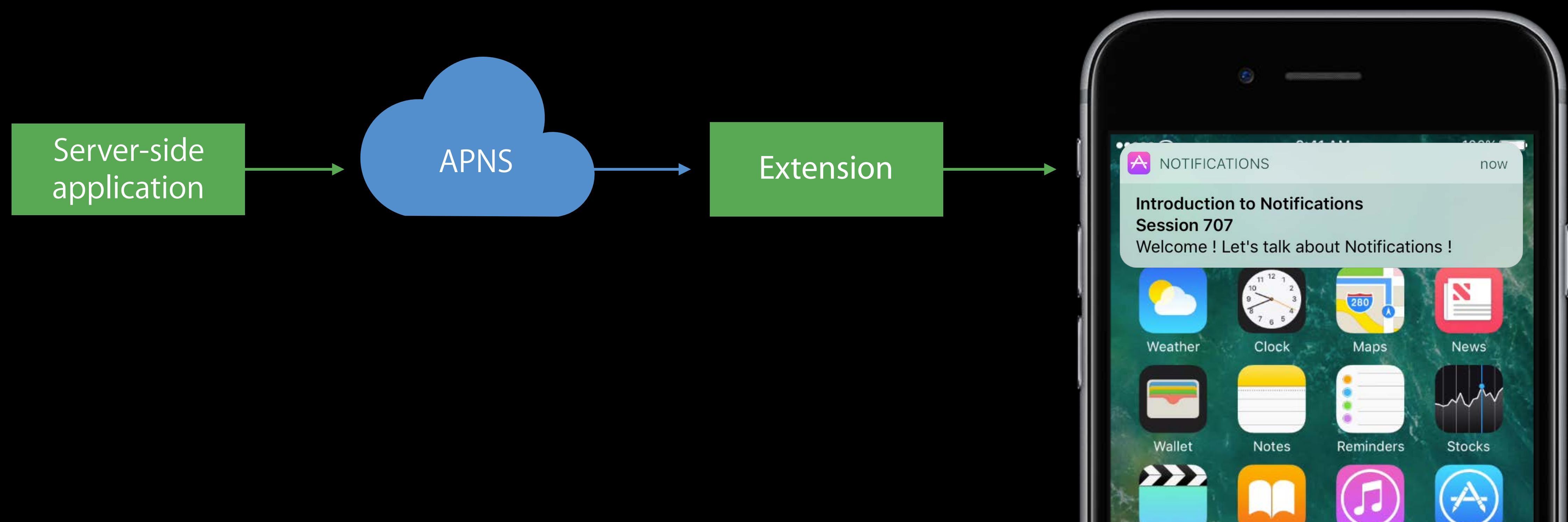
In-app presentation option

Multi-Platform



User Notifications

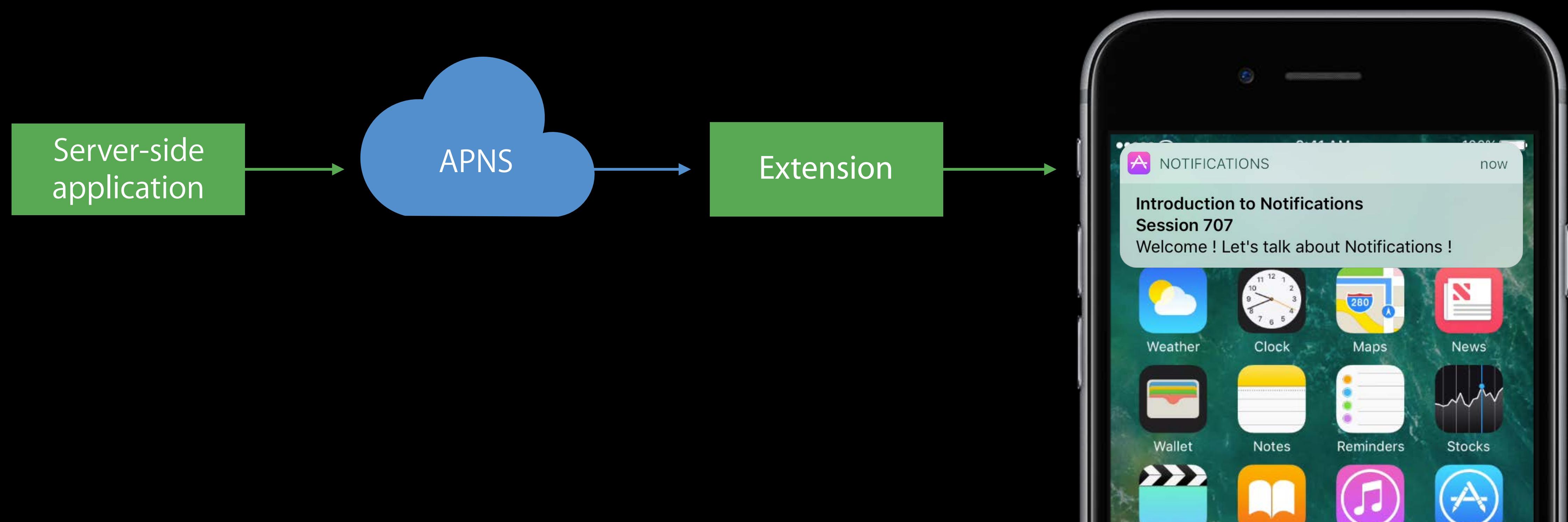
Service extension



User Notifications

Service extension

Non-UI extension point

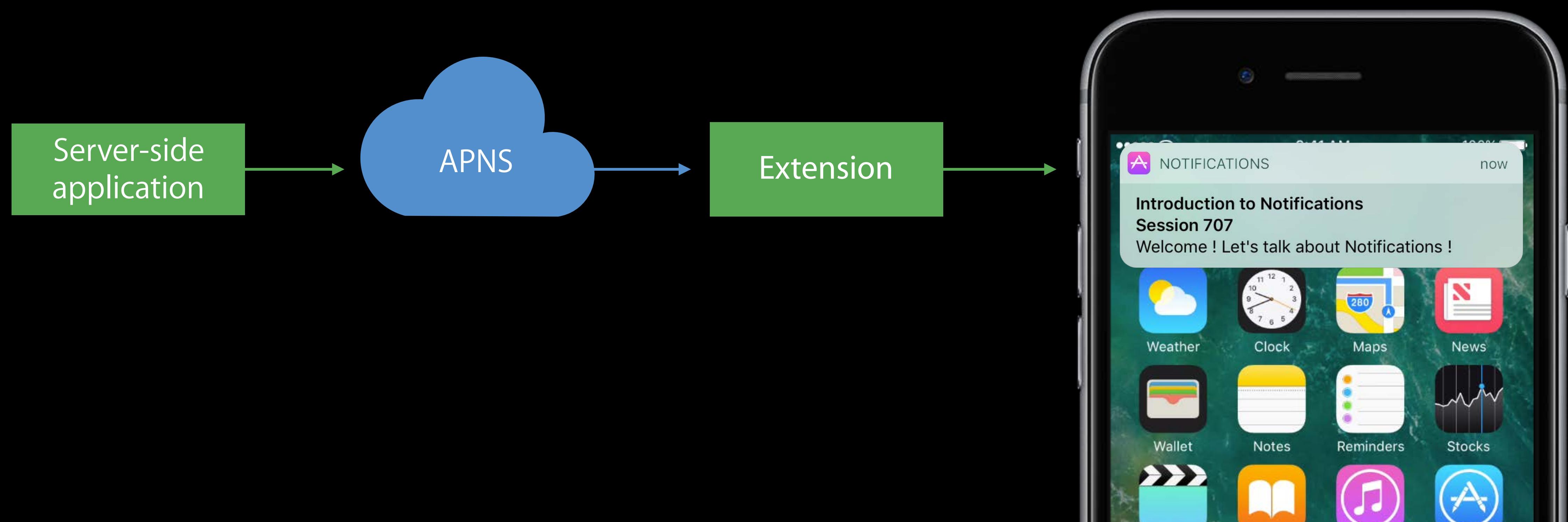


User Notifications

Service extension

Non-UI extension point

Use cases



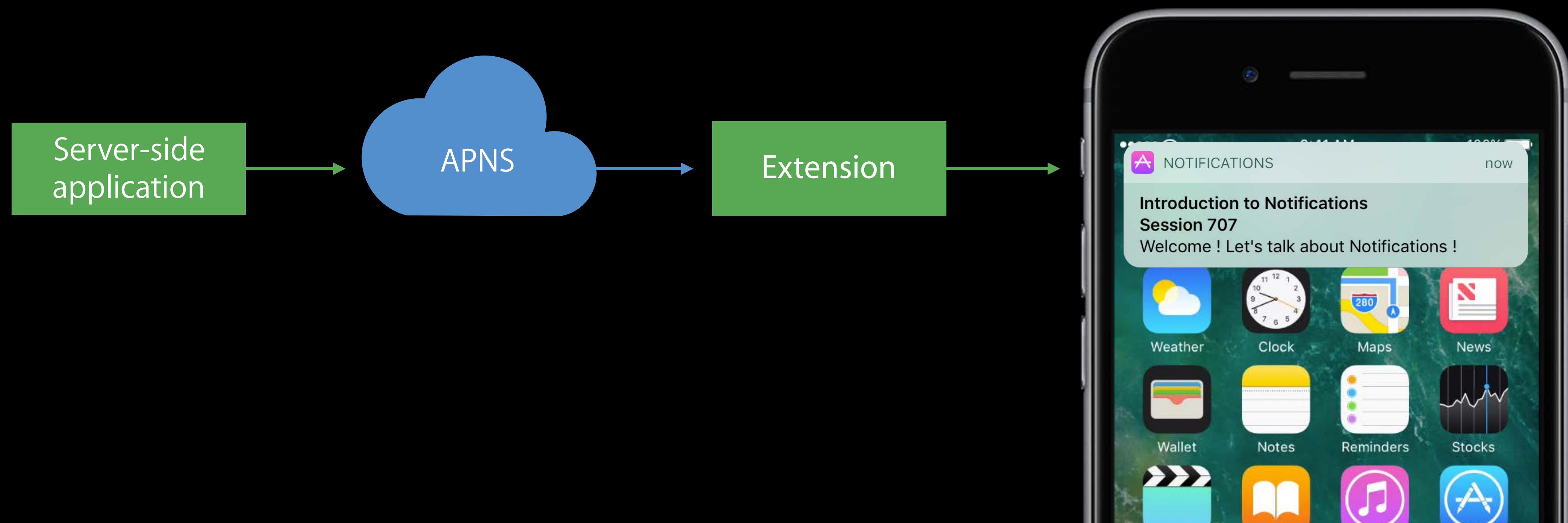
User Notifications

Service extension

Non-UI extension point

Use cases

Media attachments



User Notifications

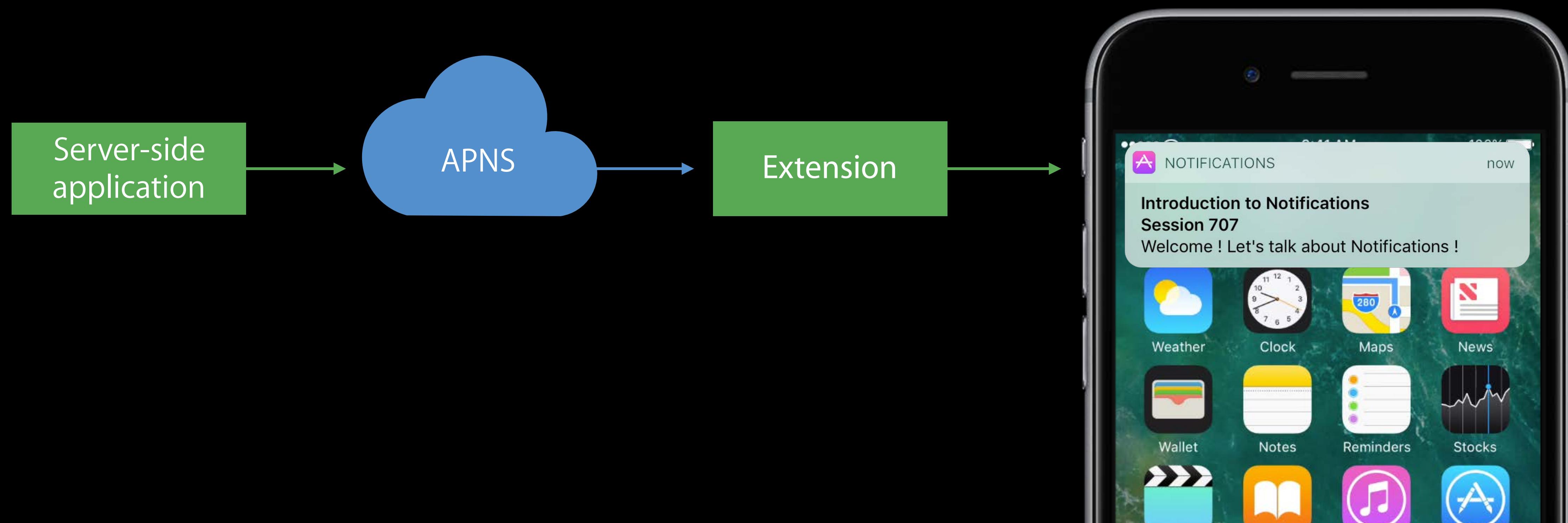
Service extension

Non-UI extension point

Use cases

Media attachments

End-to-end encryption



User Notifications

Content extension



User Notifications

Content extension



User Notifications

Content extension

UI extension point



User Notifications

Content extension

UI extension point

Custom views



User Notifications

Content extension

UI extension point

Custom views

No direct interaction



User Notifications

Introduction to Notifications

Pacific Heights

Wednesday 9:00AM

Advanced Notifications

Pacific Heights

Wednesday 10:00AM

CallKit

Directory Extension

CallKit

Directory Extension

Blocking

CallKit

Directory Extension

Blocking

Identification

CallKit

Directory Extension

Blocking Identification

CallKit

Call Provider API



CallKit

Call Provider API

A 1st party experience for your VoIP application



CallKit

Call Provider API

A 1st party experience for your VoIP application

Full screen incoming call UI



CallKit

Call Provider API

A 1st party experience for your VoIP application

Full screen incoming call UI

Integrated with other types of calls



CallKit

Call Provider API

A 1st party experience for your VoIP application

Full screen incoming call UI

Integrated with other types of calls

VoIP calls appears in Favorites and Recents



CallKit

Call Provider API

A 1st party experience for your VoIP application

Full screen incoming call UI

Integrated with other types of calls

VoIP calls appears in Favorites and Recents

Supports Siri, CarPlay, Do Not Disturb, Bluetooth



CallKit

Call Provider API

A 1st party experience for your VoIP application

Full screen incoming call UI

Integrated with other types of calls

VoIP calls appears in Favorites and Recents

Supports Siri, CarPlay, Do Not Disturb, Bluetooth



Enhancing VoIP Apps with CallKit

Mission

Thursday 5:00PM

SiriKit



SiriKit



SiriKit



SiriKit



SiriKit



Recognition

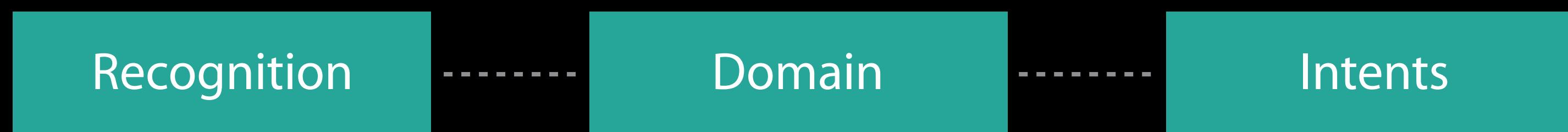
SiriKit



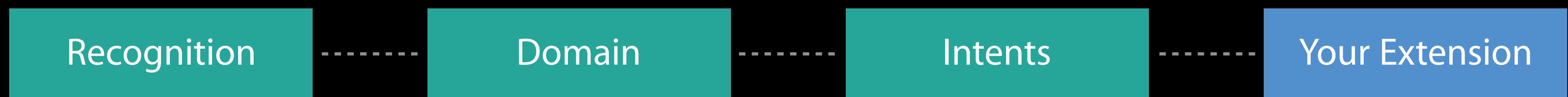
Recognition

Domain

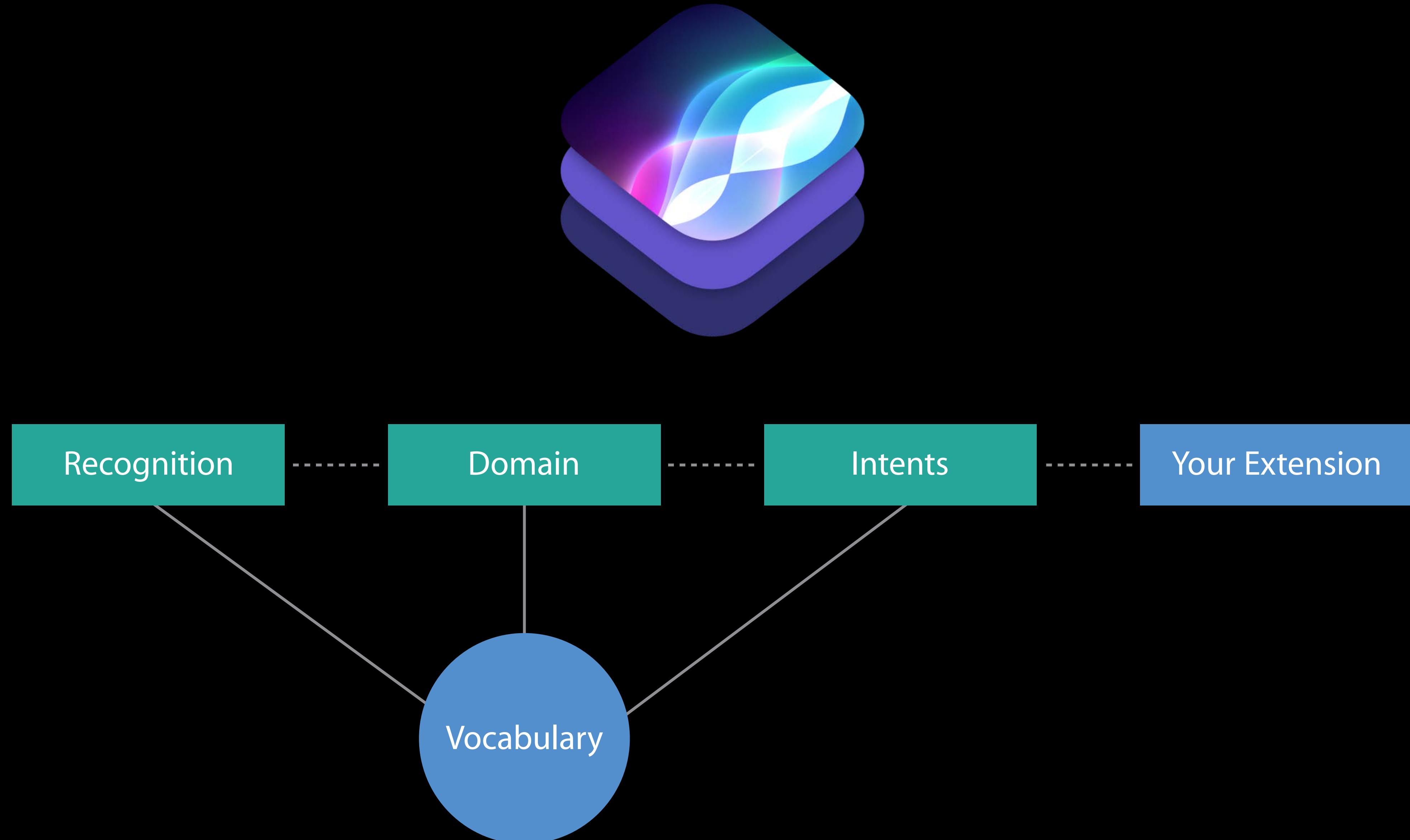
SiriKit



SiriKit



SiriKit



Intents Extension

Handle the interaction between
Siri and your application

- Intents and responses

Intents Extension

Handle the interaction between
Siri and your application

- Intents and responses

Intents are domain specific

Intents Extension

Handle the interaction between
Siri and your application

- Intents and responses

Intents are domain specific

- Make sure Siri and your app
agree on the request before
performing it

Intents Extension

Handle the interaction between
Siri and your application

- Intents and responses

Intents are domain specific

- Make sure Siri and your app
agree on the request before
performing it

*“Tell Miko on WWDCChat
we need to meet after this session”*

Intents Extension

Handle the interaction between
Siri and your application

- Intents and responses

Intents are domain specific

- Make sure Siri and your app
agree on the request before
performing it

Send message
intent

Recipient

App Name

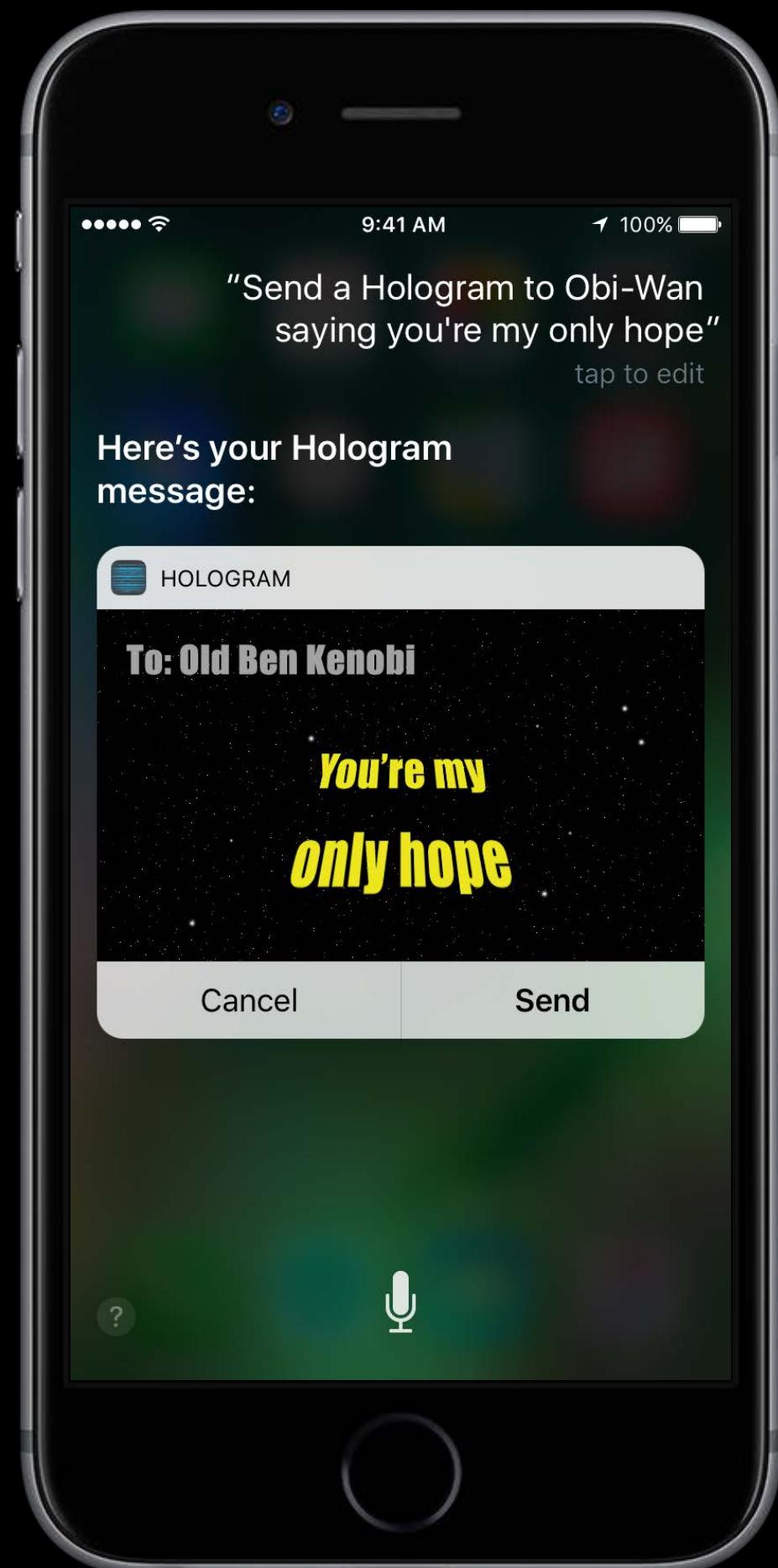
*“Tell Miko on WWDCChat
we need to meet after this session”*

Content

IntentsUI Extension

Embed your own UI in the Siri Transcript

- Optional



Intents are Shared

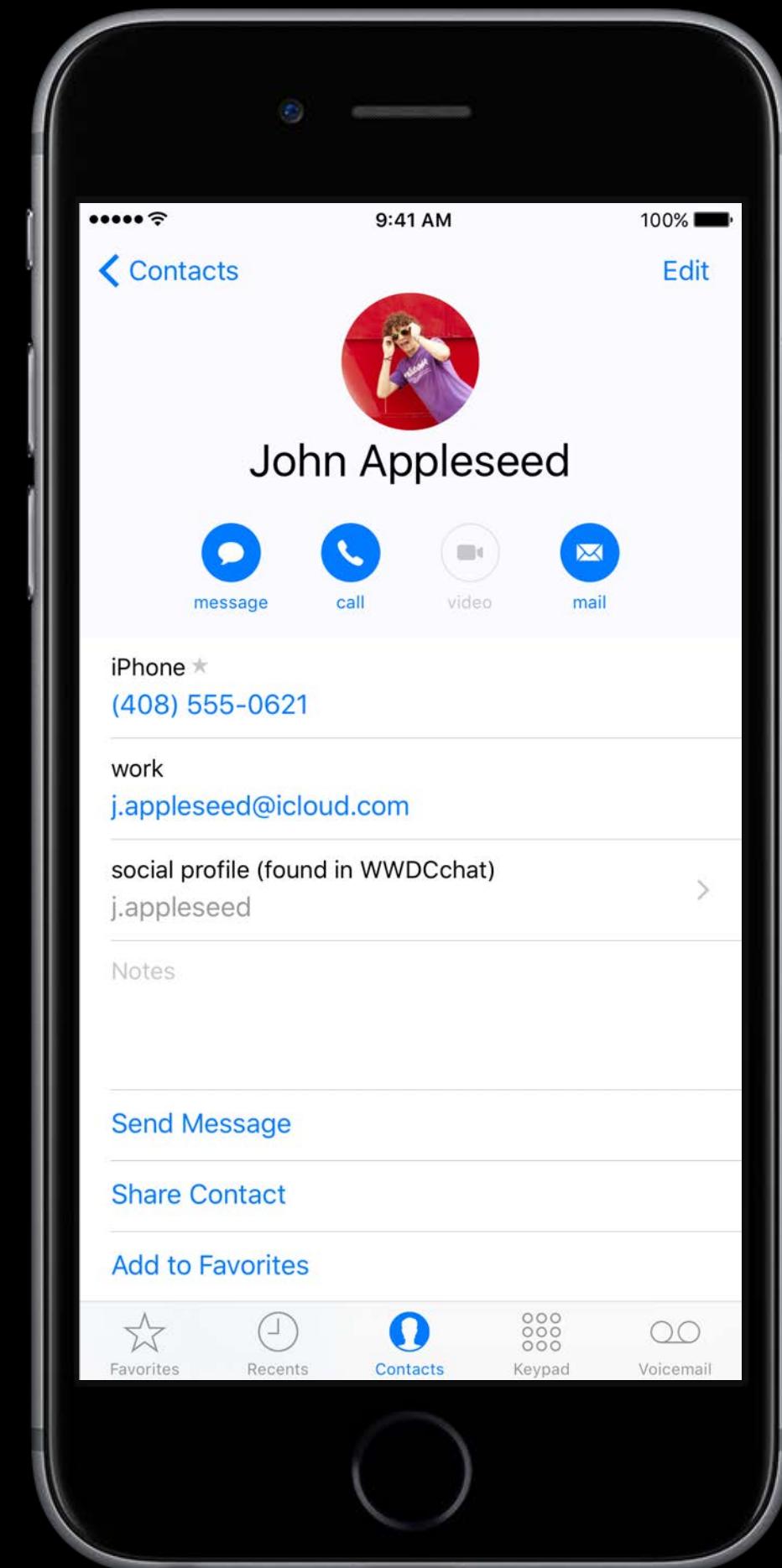
Intents describe requests

- For Siri to communicate with your app
- To integrate with CallKit
- For Ride Sharing in Maps
- To donate information to the system about a contact

Intents are Shared

Intents describe requests

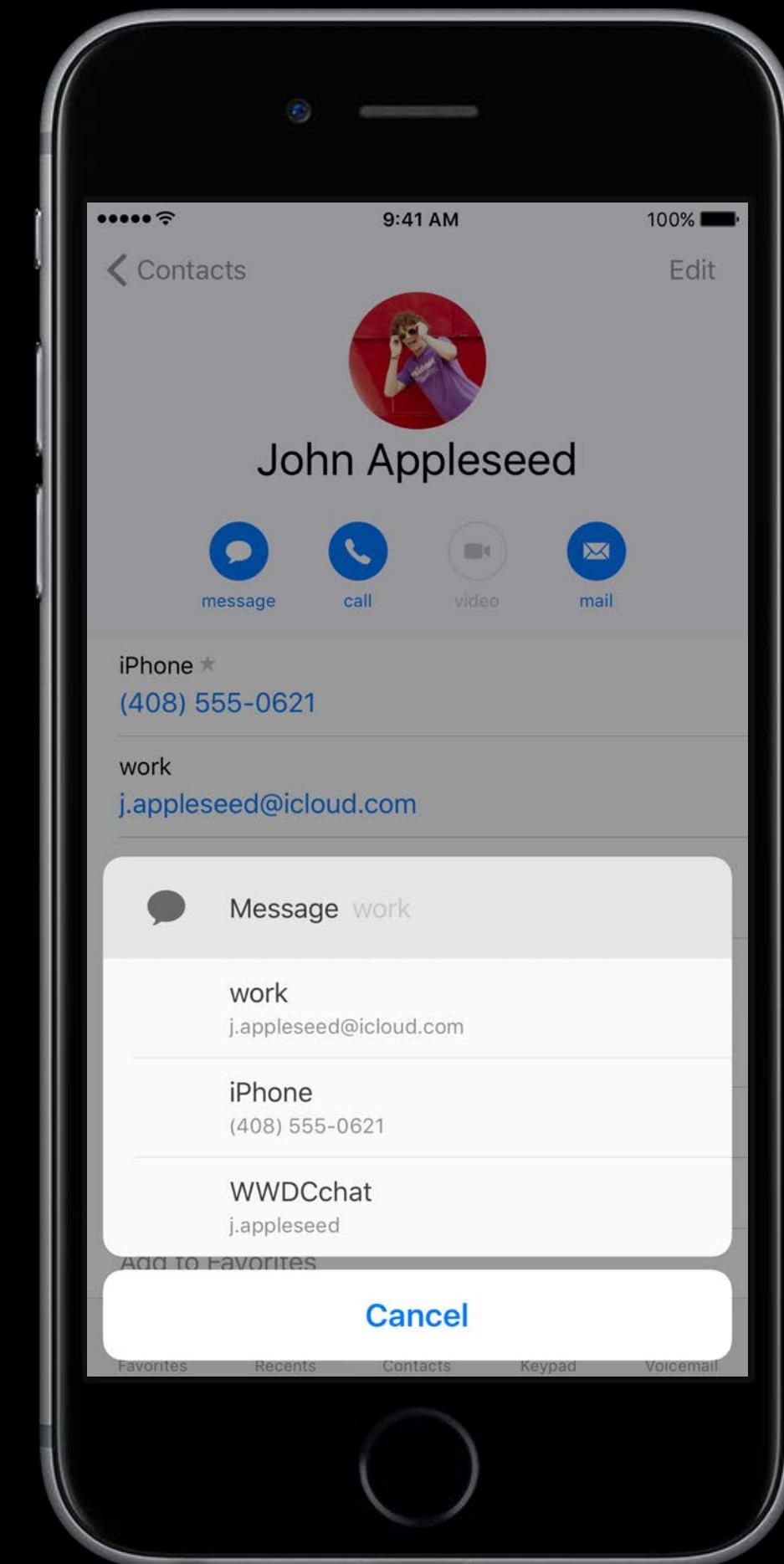
- For Siri to communicate with your app
- To integrate with CallKit
- For Ride Sharing in Maps
- To donate information to the system about a contact



Intents are Shared

Intents describe requests

- For Siri to communicate with your app
- To integrate with CallKit
- For Ride Sharing in Maps
- To donate information to the system about a contact



Intents are Shared

Intents describe requests

- For Siri to communicate with your app
- To integrate with CallKit
- For Ride Sharing in Maps
- To donate information to the system about a contact



iMessage Apps

iMessage Apps

Write apps for Messages

iMessage Apps

Write apps for Messages

Sticker packs

iMessage Apps

Write apps for Messages

Sticker packs

Messages extension

iMessage Apps

Sticker Packs



iMessage Apps

Sticker Packs

No code required



iMessage Apps

Sticker Packs

No code required

Package and distribute your images



iMessage Apps

Message Extensions



iMessage Apps

Message Extensions

Dynamic stickers content



iMessage Apps

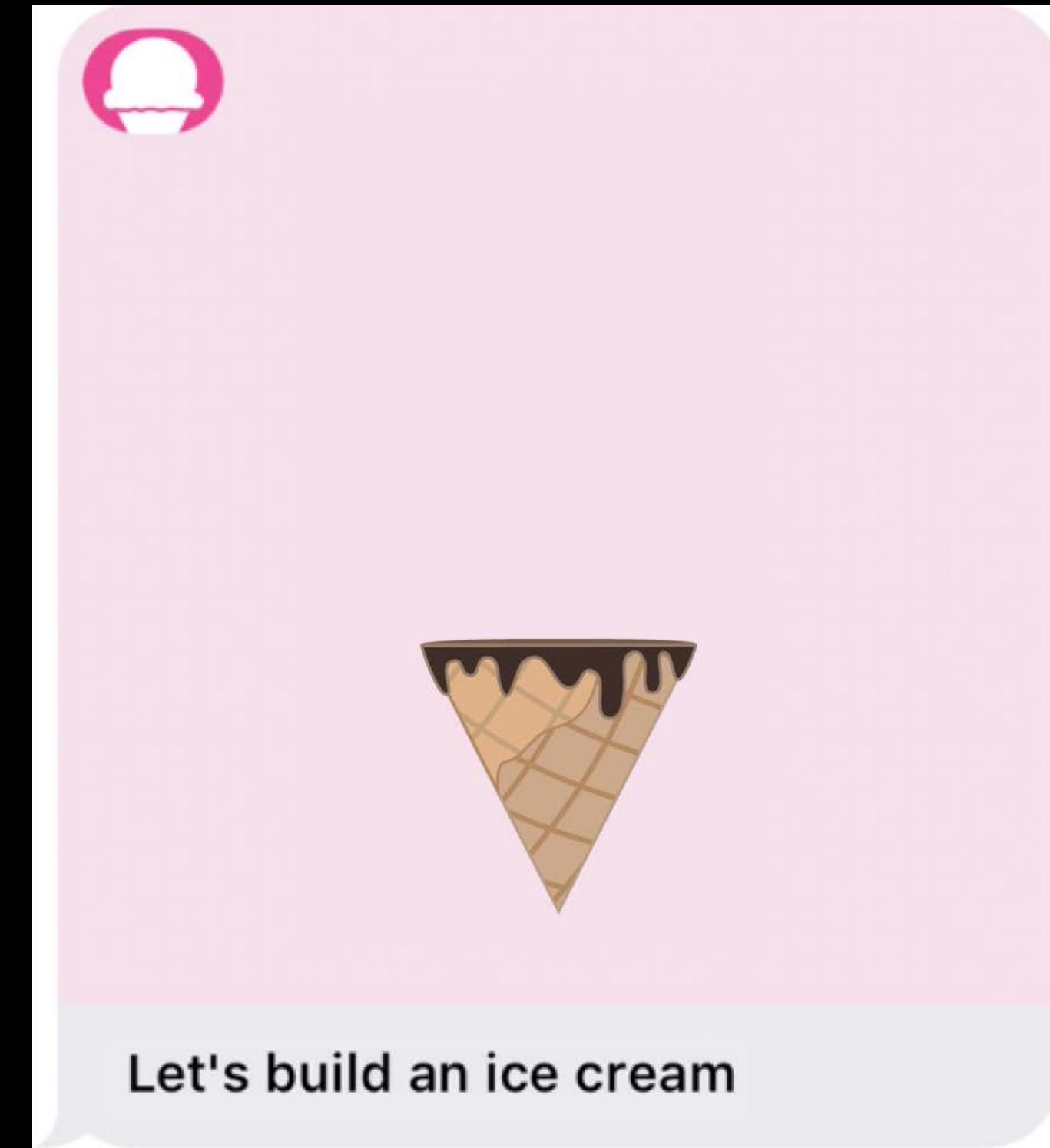
Message Extensions

Dynamic stickers content
Customize your UI



iMessage Apps

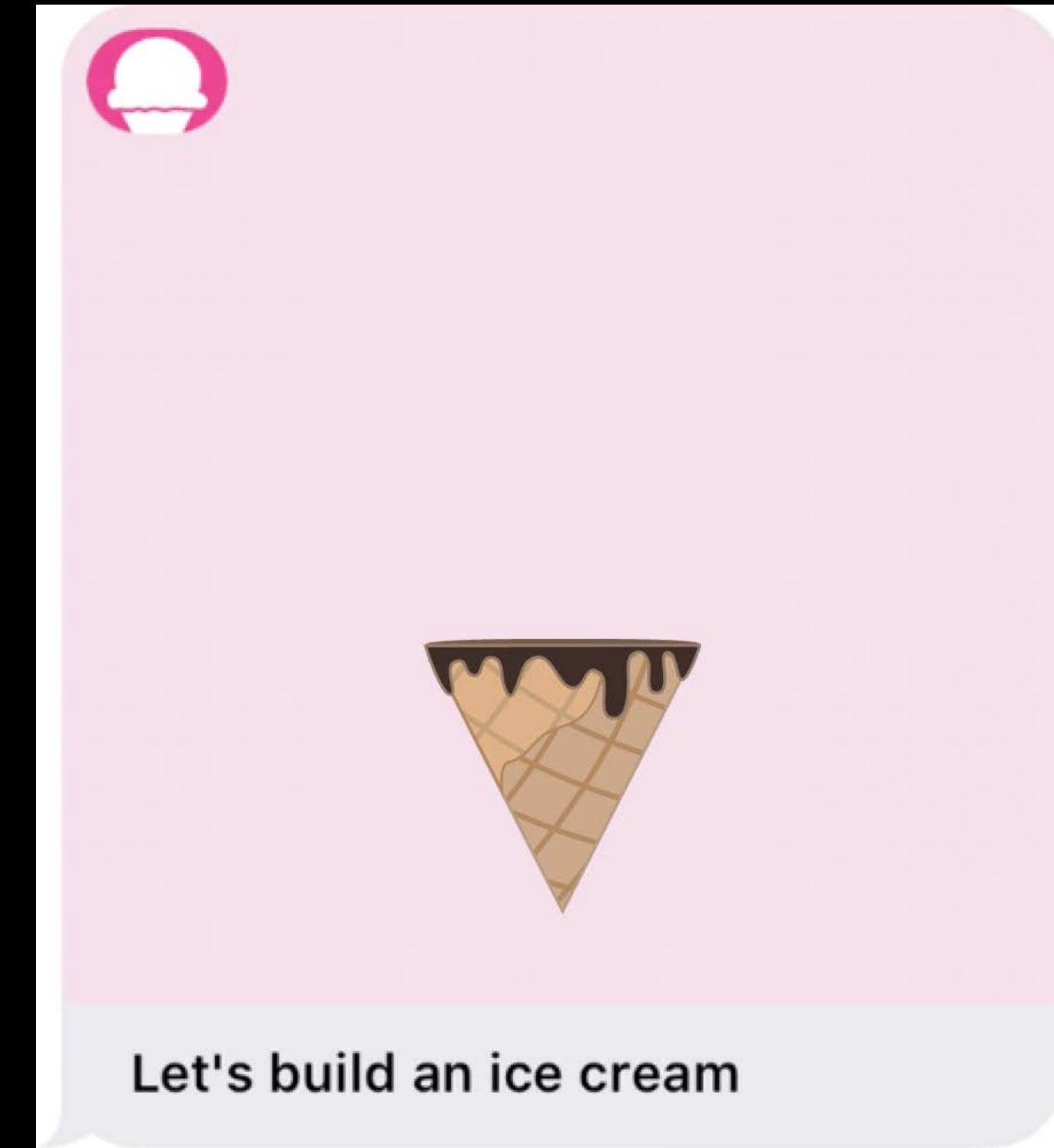
Message Extensions



iMessage Apps

Message Extensions

Interactive Messages

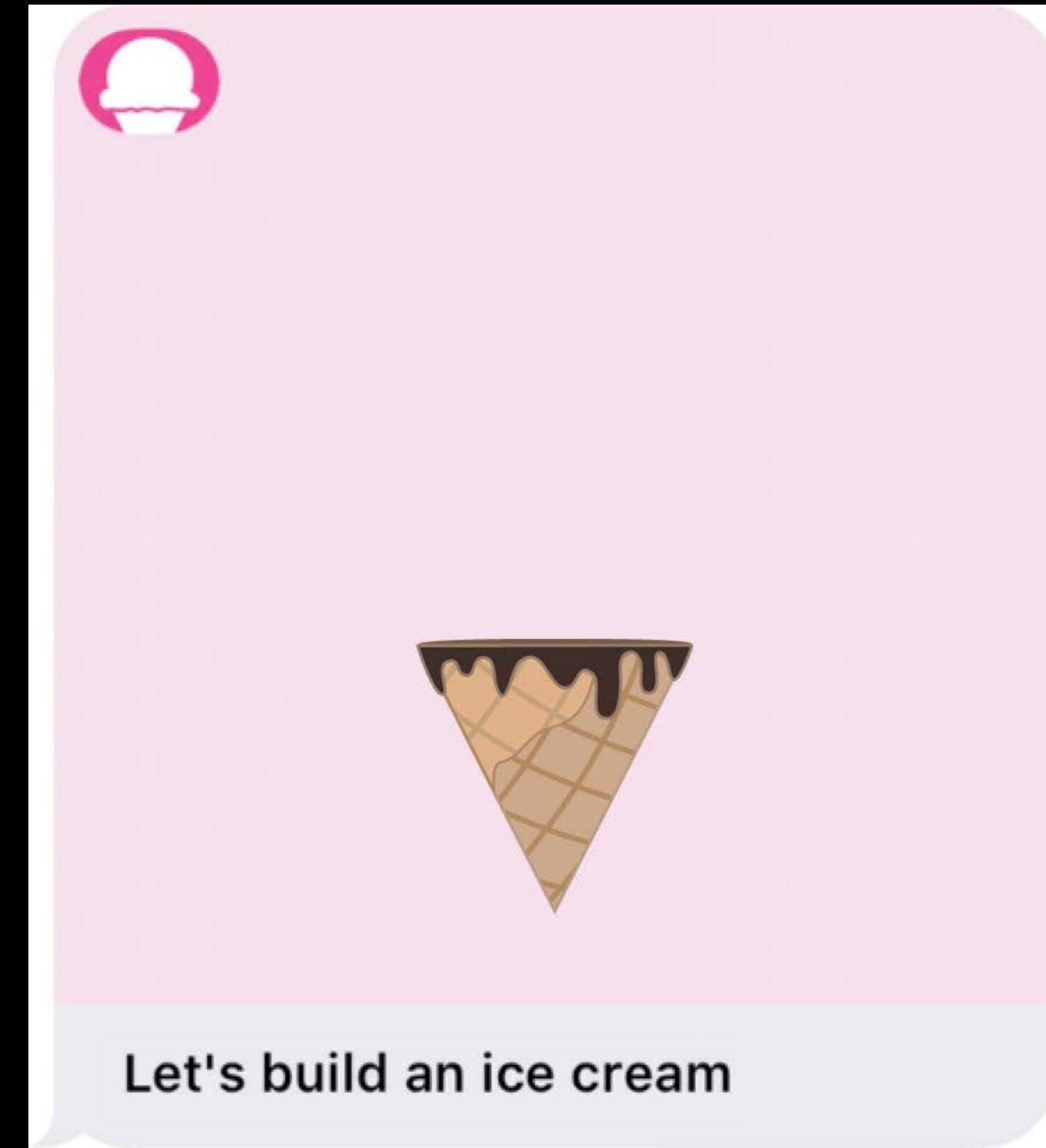


iMessage Apps

Message Extensions

Interactive Messages

Connect and integrate with a Messages session



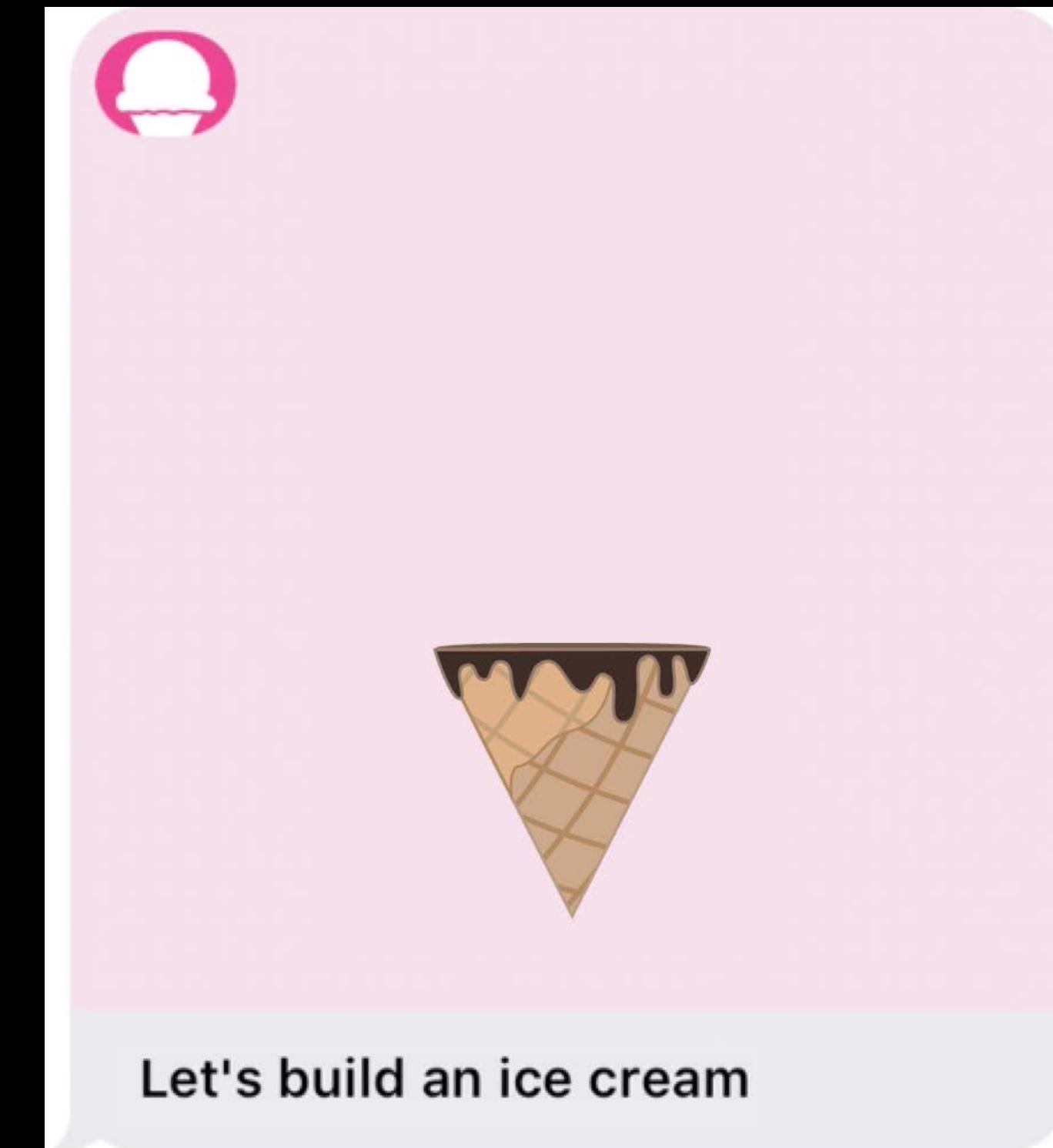
iMessage Apps

Message Extensions

Interactive Messages

Connect and integrate with a Messages session

Custom content



iMessage Apps

iMessage Apps

iMessage Apps and Stickers, Part 1

Presidio

Tuesday 11:00AM

iMessage Apps and Stickers, Part 2

Presidio

Thursday 1:40PM

More Information

<https://developer.apple.com/wwdc16/205>

Labs

Cocoa Touch Lab

Frameworks Lab D Tuesday 2:30PM

Cocoa Touch Lab

Frameworks Lab A Wednesday 3:00M

UIKit and UIKit Animations Lab

Frameworks Lab C Thursday 1:00PM

Cocoa Touch 3D Touch Lab

Frameworks Lab C Friday 10:30AM



W W D C 16