#WWDC18

What's New in Cocoa Touch

Session 202

Josh Shaffer
Eliza Block

Framework updates

API enhancements

Siri shortcuts

Framework Updates

Performance

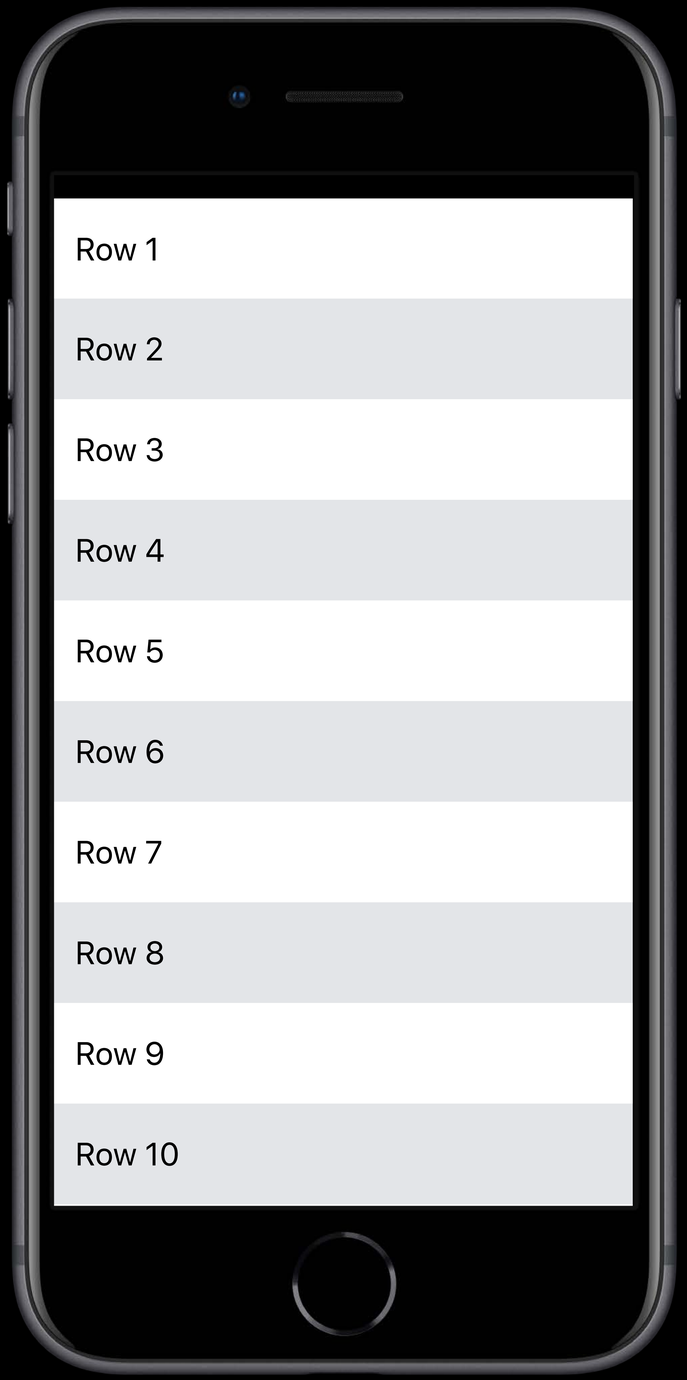
Framework Updates

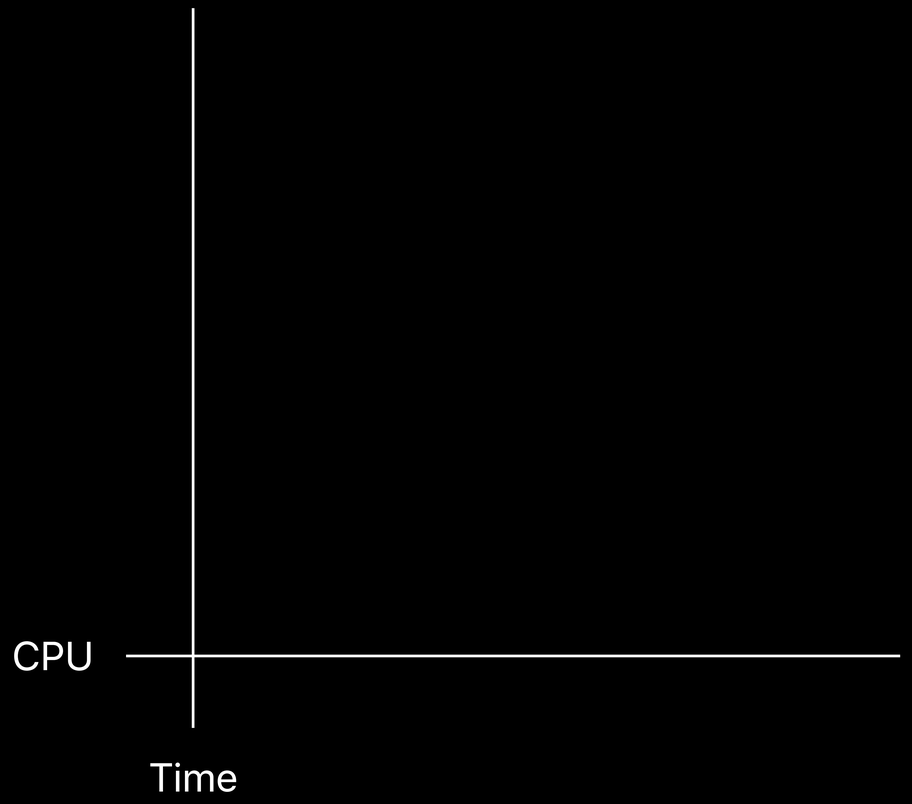
Performance

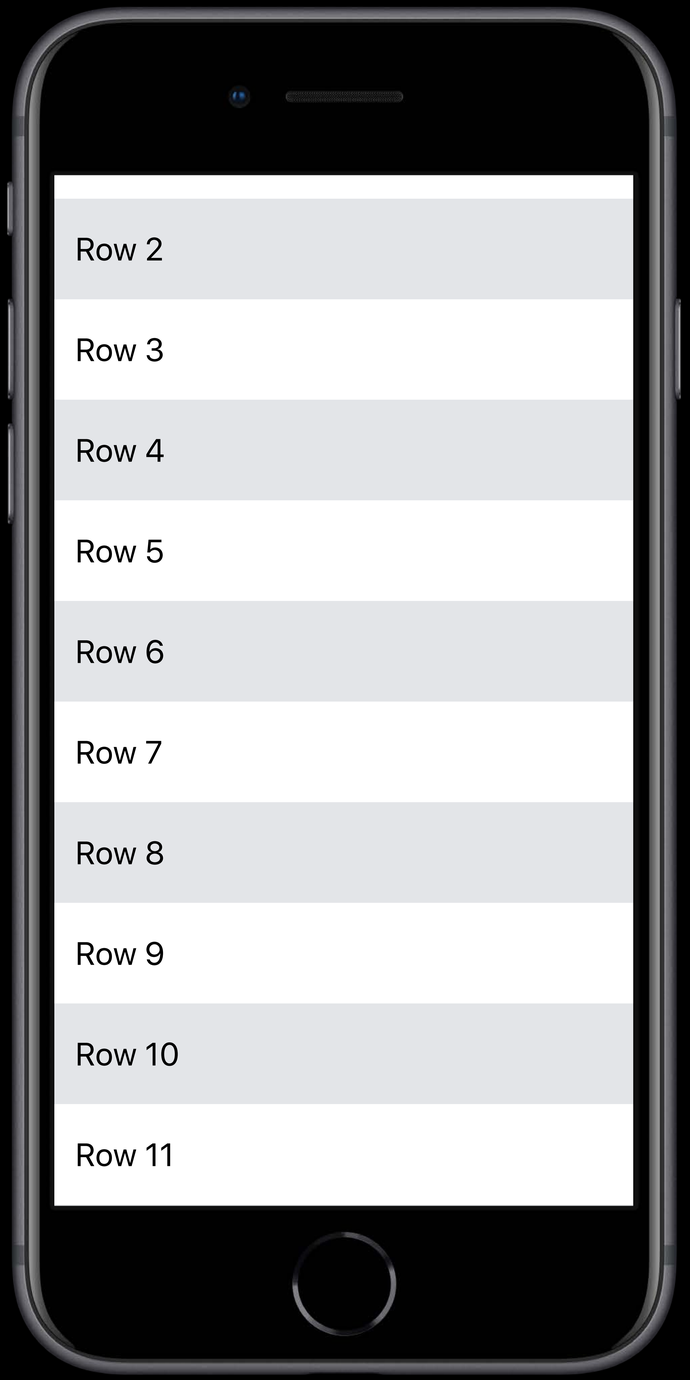
Scrolling

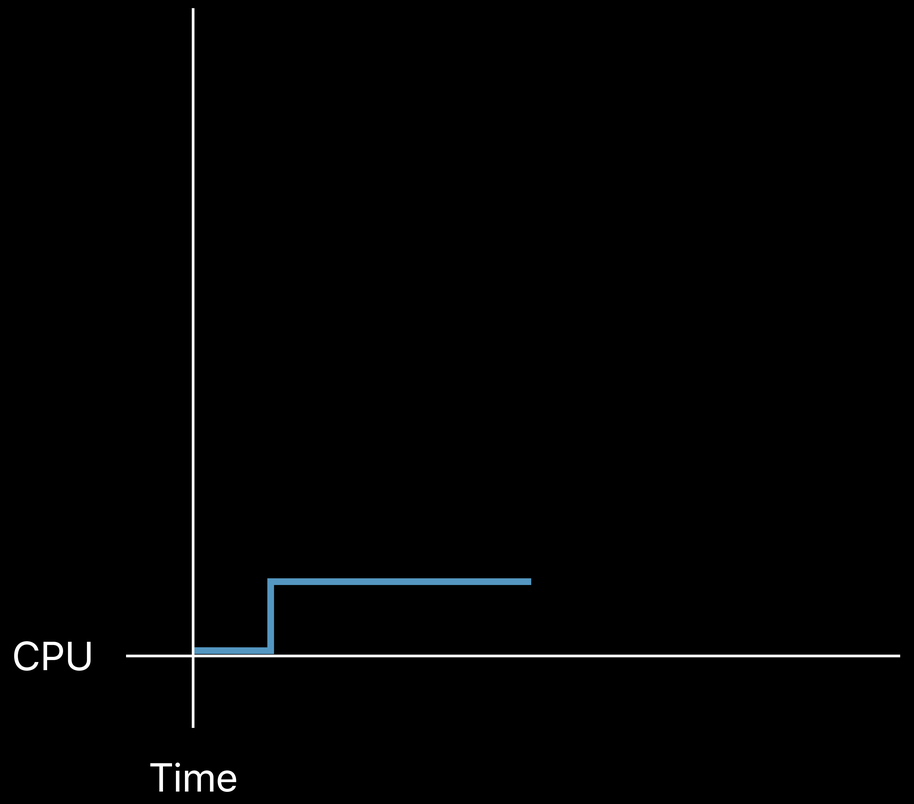
Memory

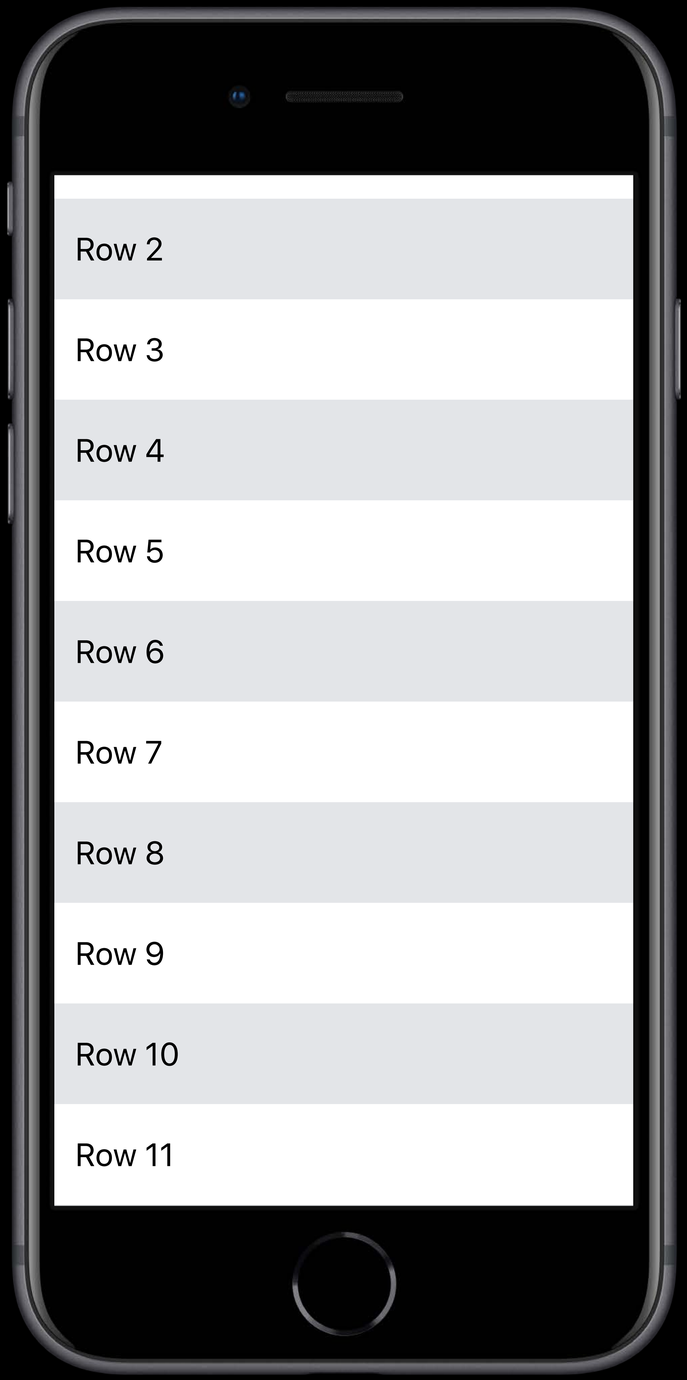
Auto Layout

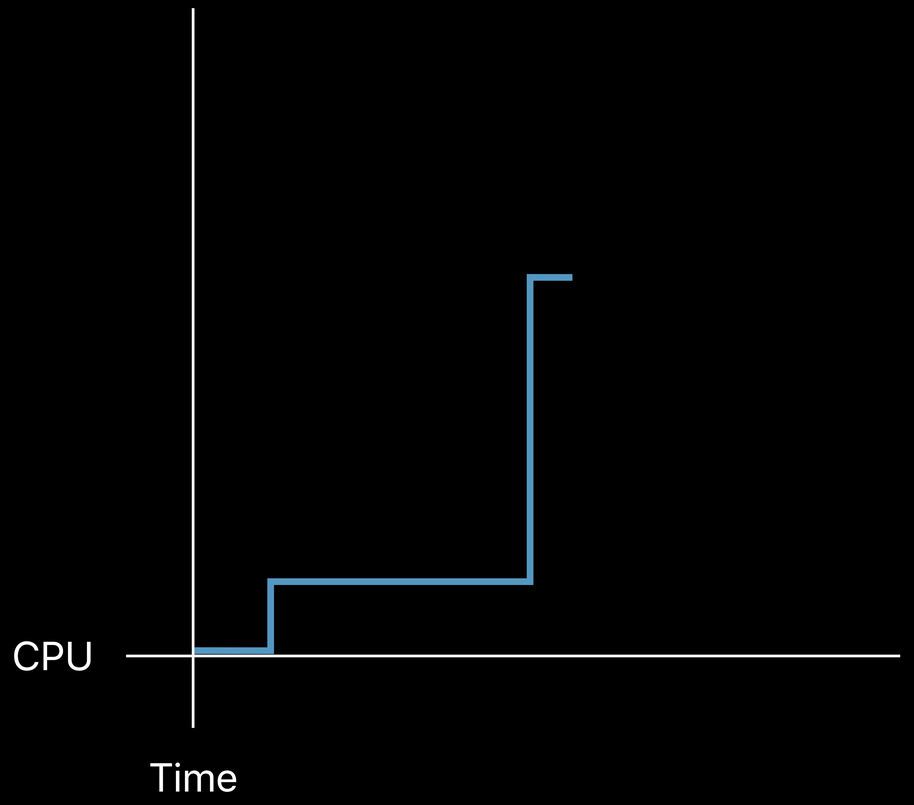

Row 1
Row 2
Row 3
Row 4
Row 5
Row 6
Row 7
Row 8
Row 9
Row 10

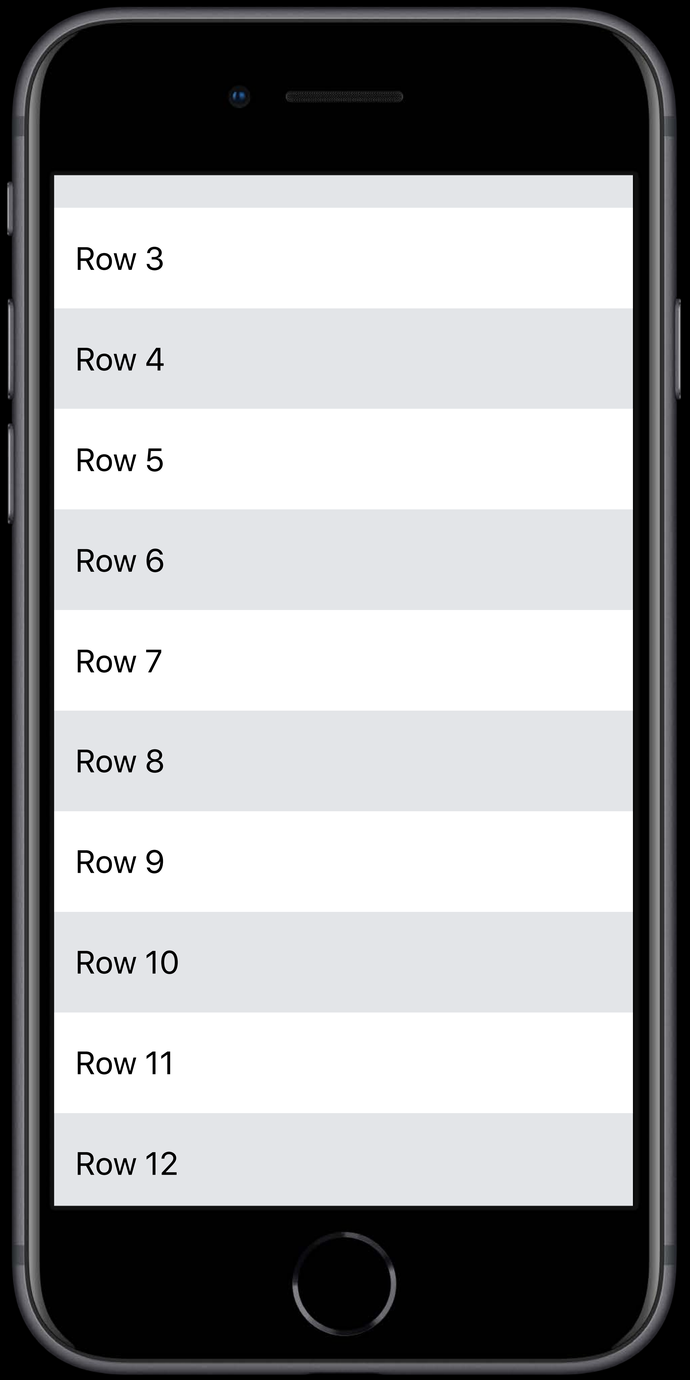

CPU
Time

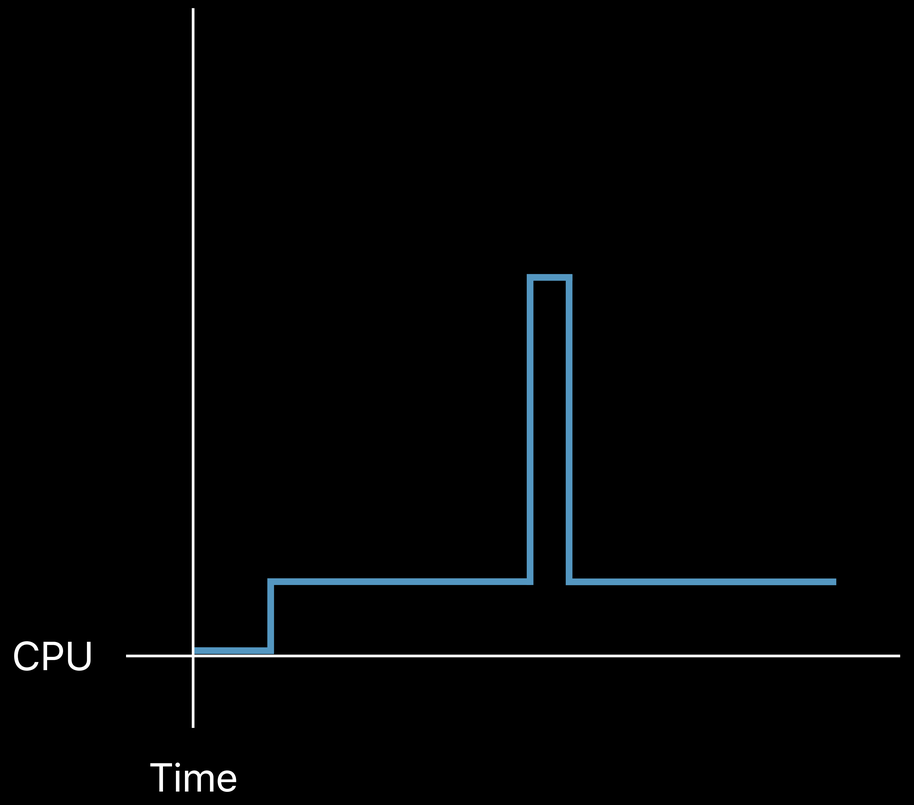

Row 2
Row 3
Row 4
Row 5
Row 6
Row 7
Row 8
Row 9
Row 10
Row 11


CPU
Time


Row 2
Row 3
Row 4
Row 5
Row 6
Row 7
Row 8
Row 9
Row 10
Row 11


CPU
Time


Row 3
Row 4
Row 5
Row 6
Row 7
Row 8
Row 9
Row 10
Row 11
Row 12


CPU
Time

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}
```

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}
```

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}
```

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}

// call layoutSubviews on UIViews in the cell
// call draw() on UIViews in the cell
```

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}

// call layoutSubviews on UIViews in the cell
// call draw() on UIViews in the cell
```

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}

// call layoutSubviews on UIViews in the cell
// call draw() on UIViews in the cell
```

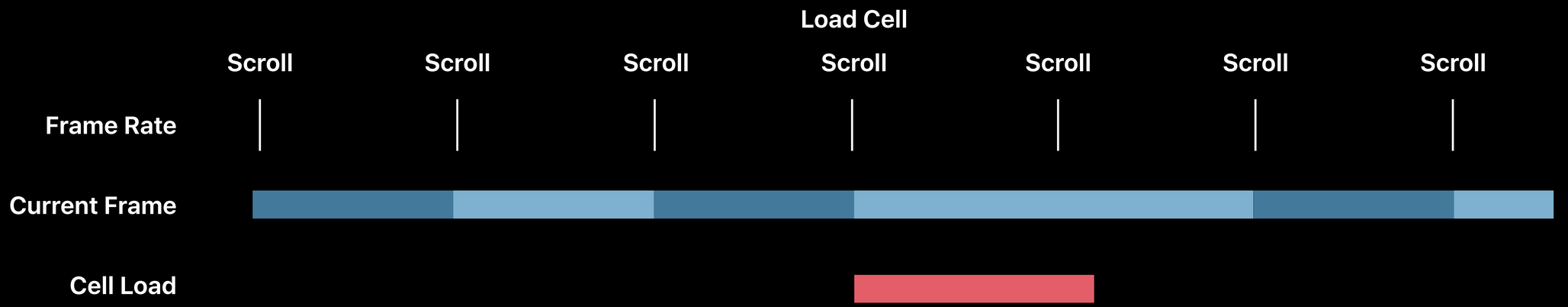
```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}

// call layoutSubviews on UIViews in the cell
// call draw() on UIViews in the cell
```

```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}

// call layoutSubviews on UIViews in the cell
// call draw() on UIViews in the cell
```

```
// UITableView Pre-Fetching

protocol UITableViewDataSourcePrefetching {
    func tableView(_ tableView: UITableView, prefetchRowsAt indexPaths: [IndexPath])
    func tableView(_ tableView: UITableView,
                   cancelPrefetchingForRowsAt: indexPaths [IndexPath])
}
```

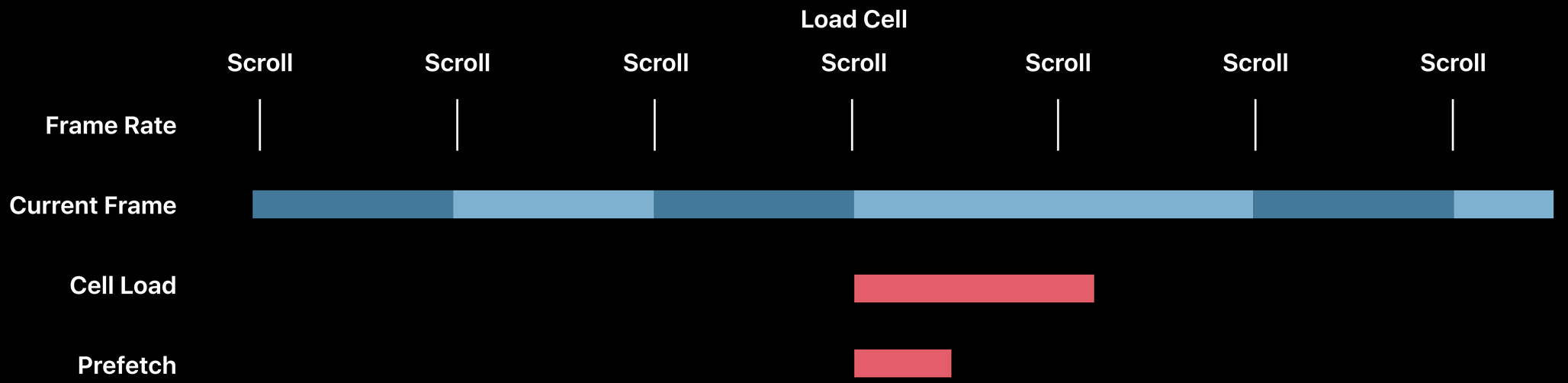
Scrolling

Prefetch Serialization



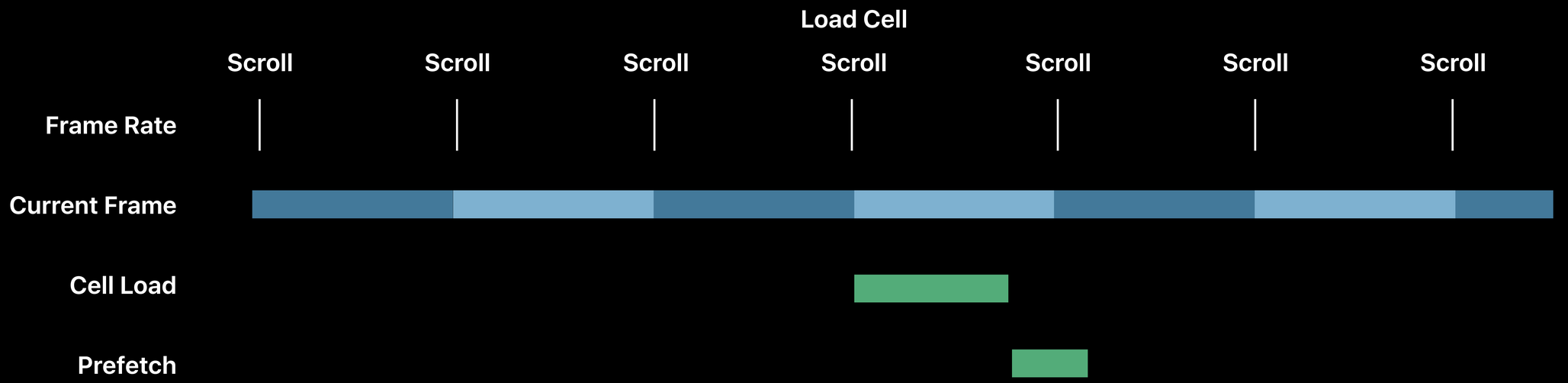
Scrolling

Prefetch Serialization



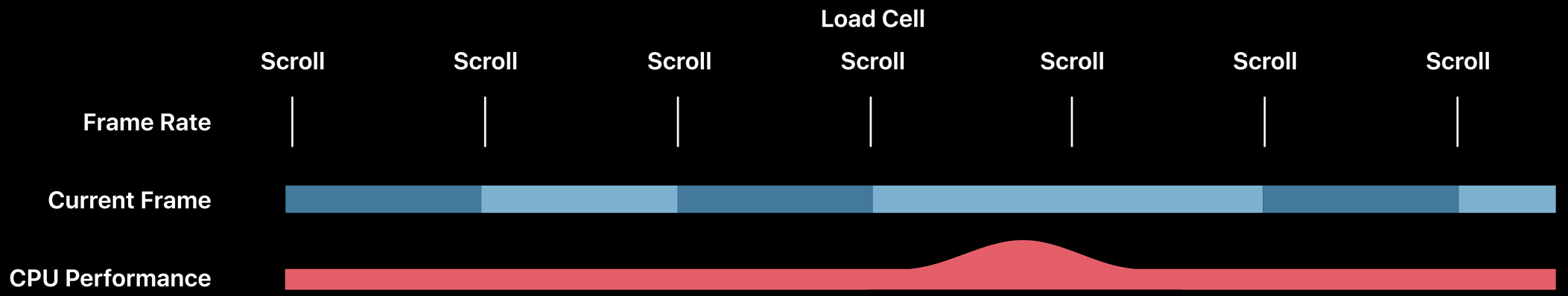
Scrolling

Prefetch Serialization



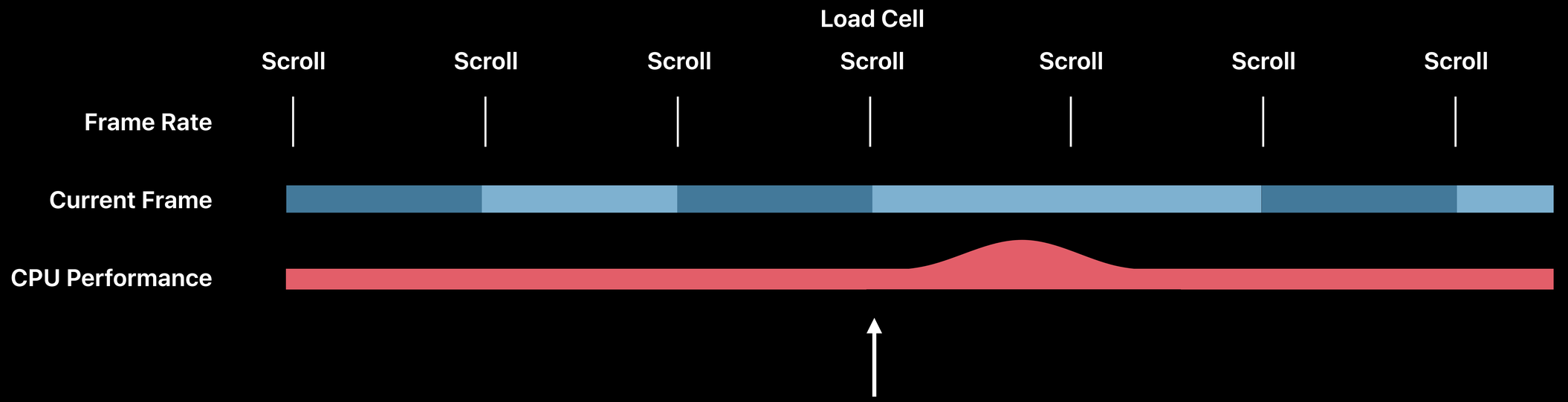
Scrolling

CPU Performance



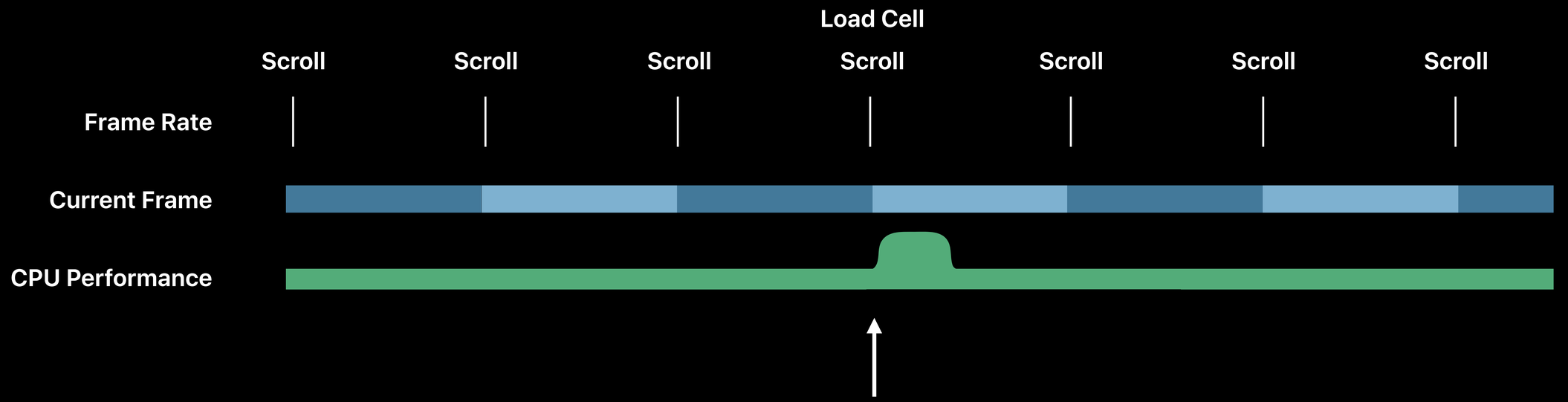
Scrolling

CPU Performance



Scrolling

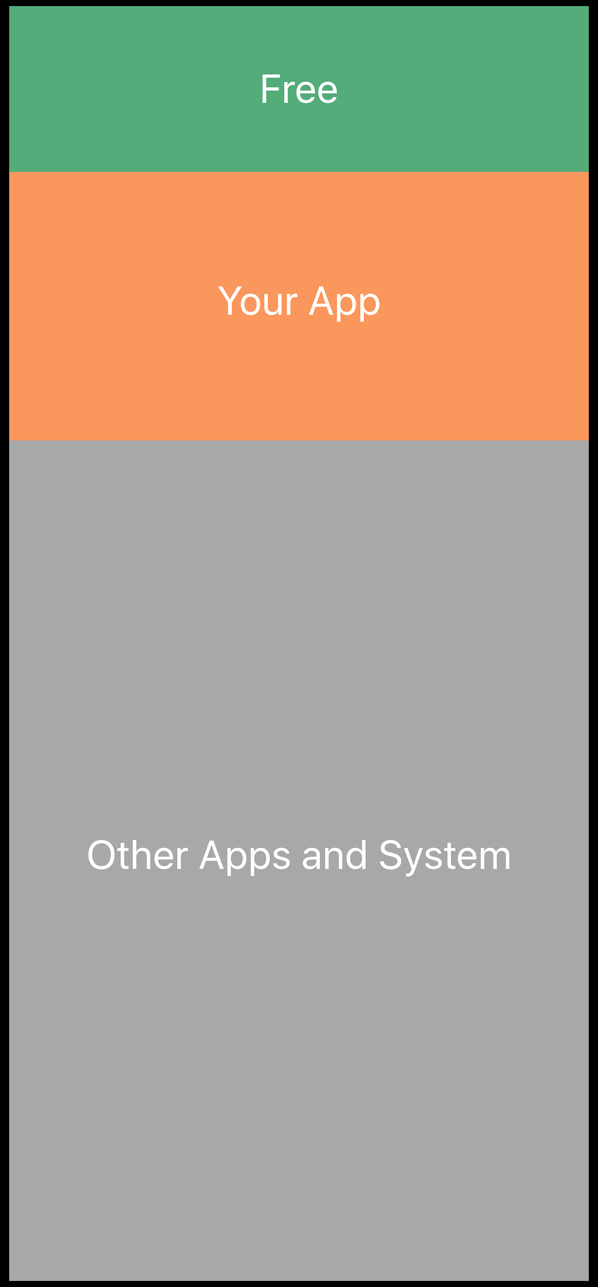
CPU Performance

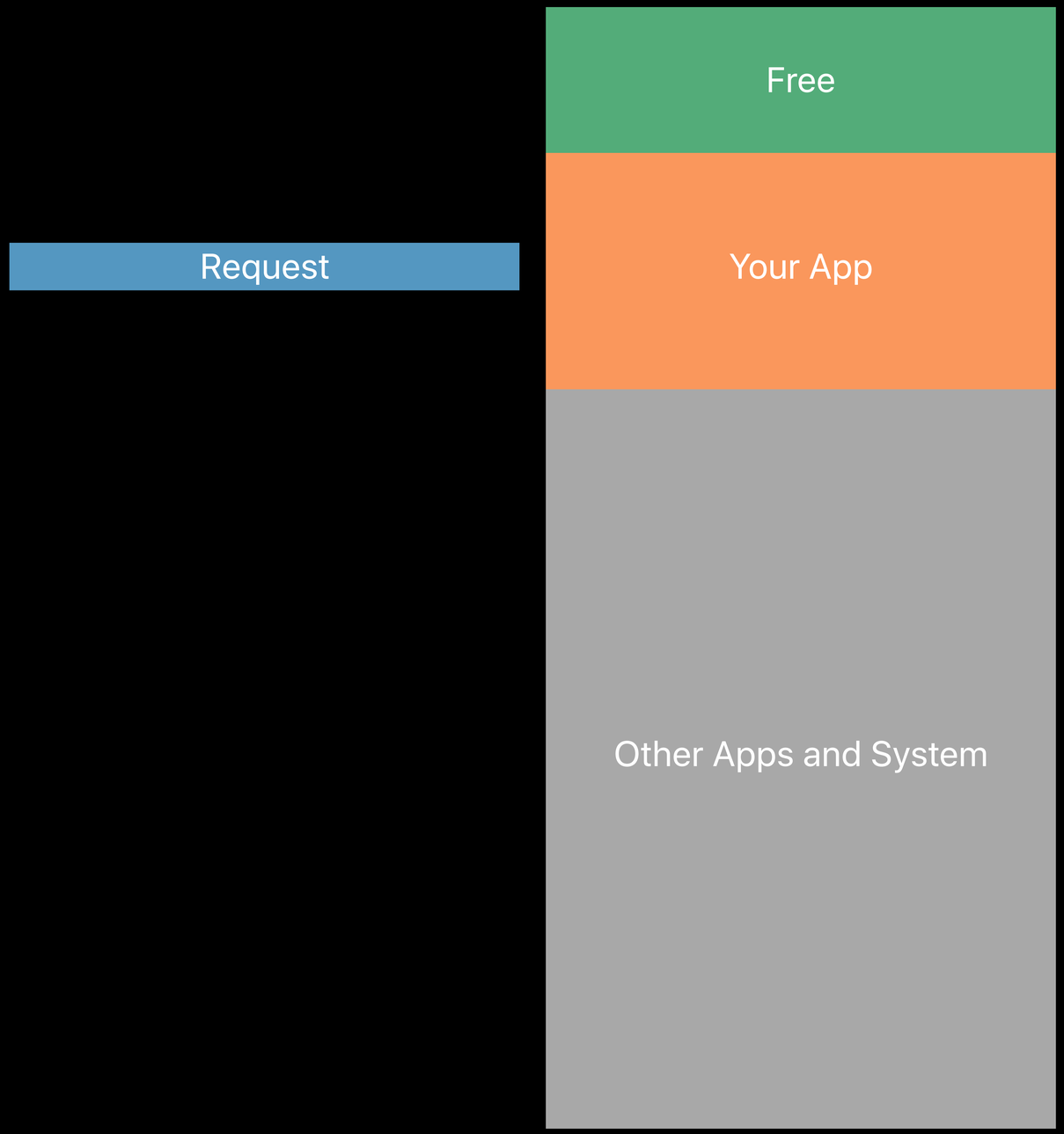


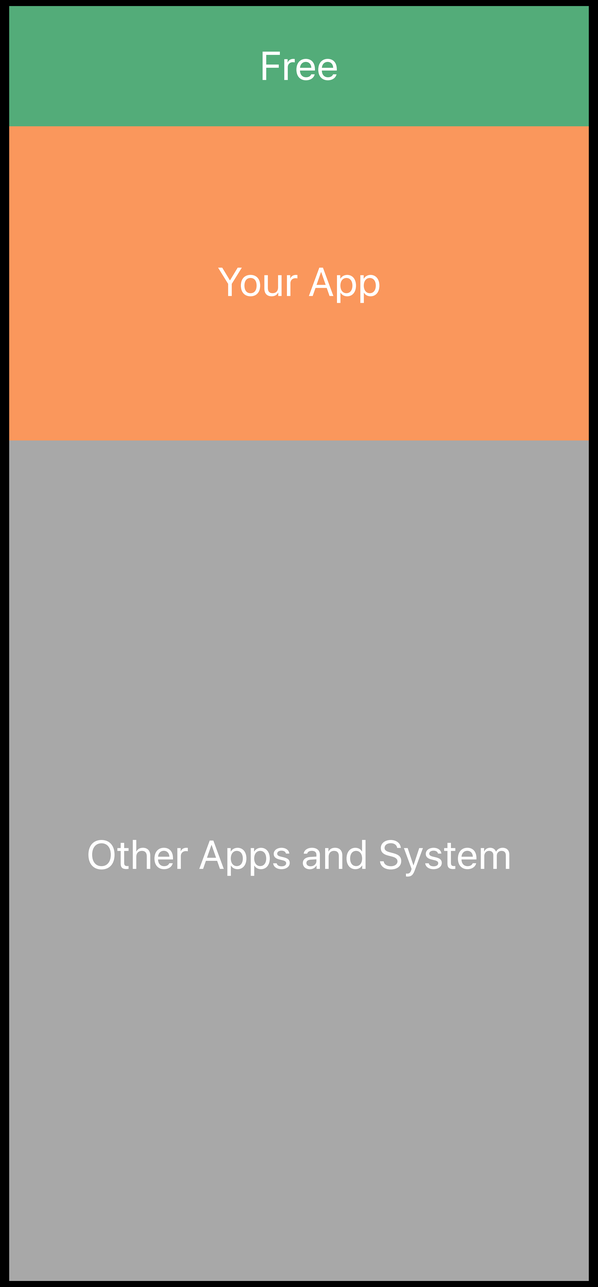
```
// UITableView Cell Load Cost

import UIKit

class MyTableViewDataSource {
    func tableView(_ tableView: UITableView,
       cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        // Dequeue or allocate cell
        // Populate cell with model data

        return myCell
    }
}

// compute layout
// call draw() on UIViews
```

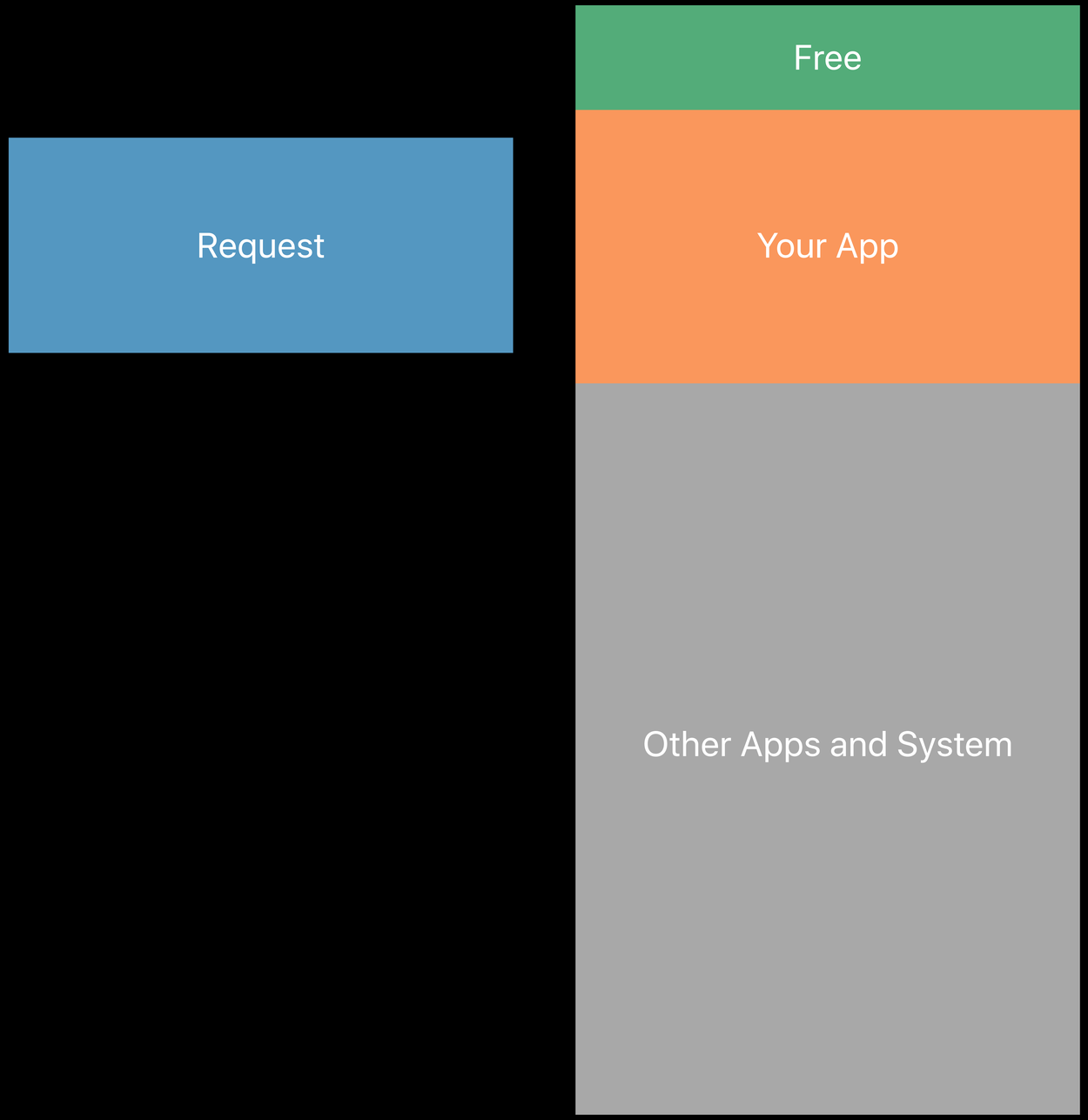
Memory

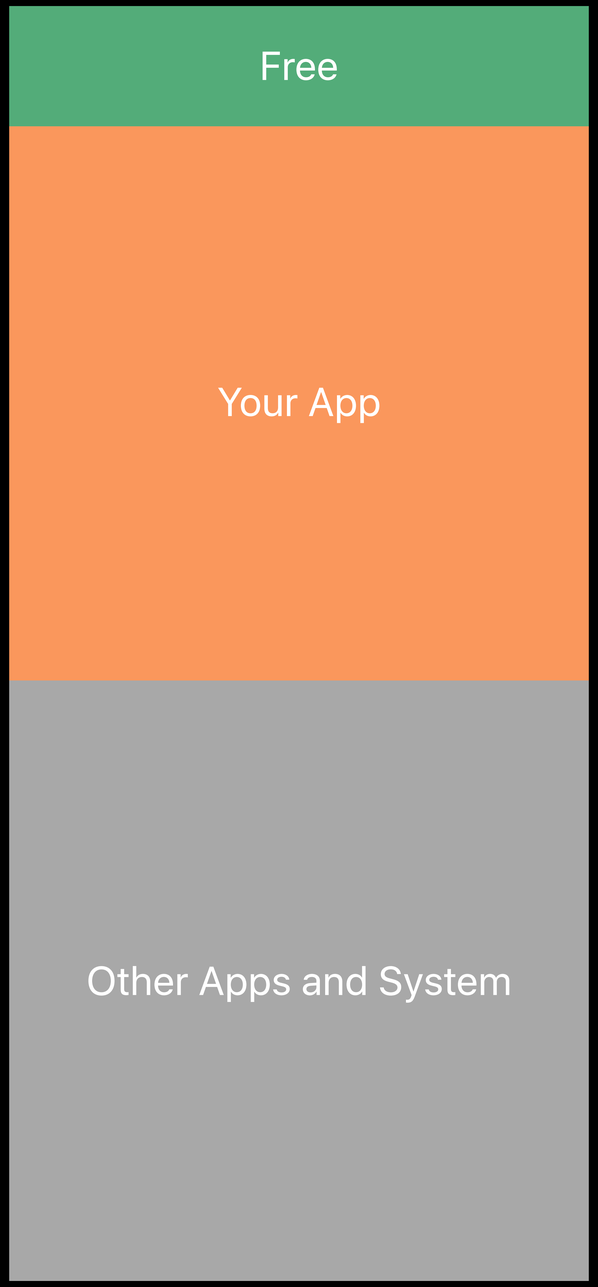
Memory Is Performance


Free
Your App
Other Apps and System


Free
Your App
Request
Other Apps and System


Free
Your App
Other Apps and System


Free
Request
Your App
Other Apps and System


Free
Your App
Other Apps and System

Automatic Backing Store

Automatic Backing Store

375

250

Automatic Backing Store

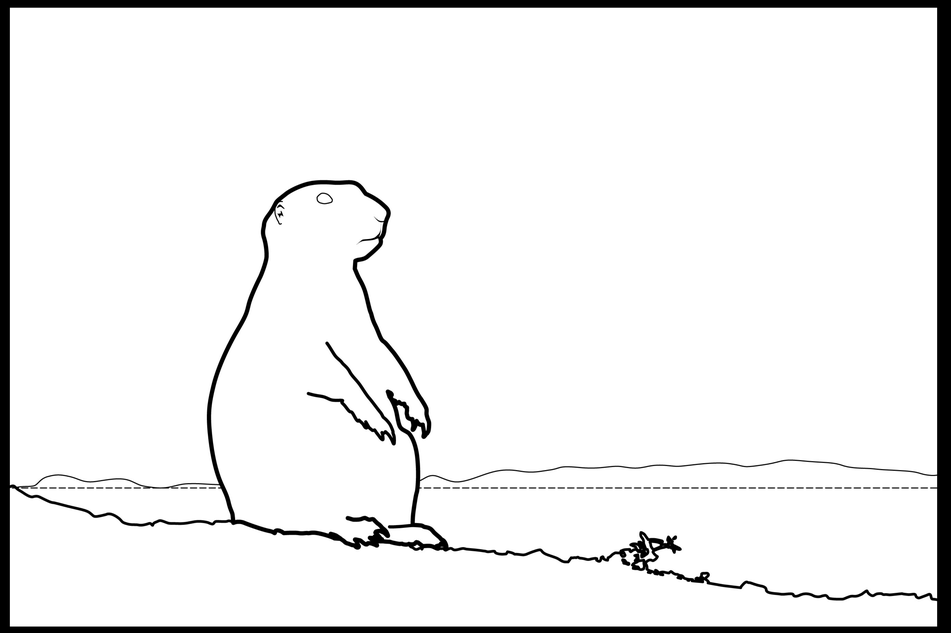
375

250

375w x 250h x @3x x 64 bpp = **2.2 megabytes**

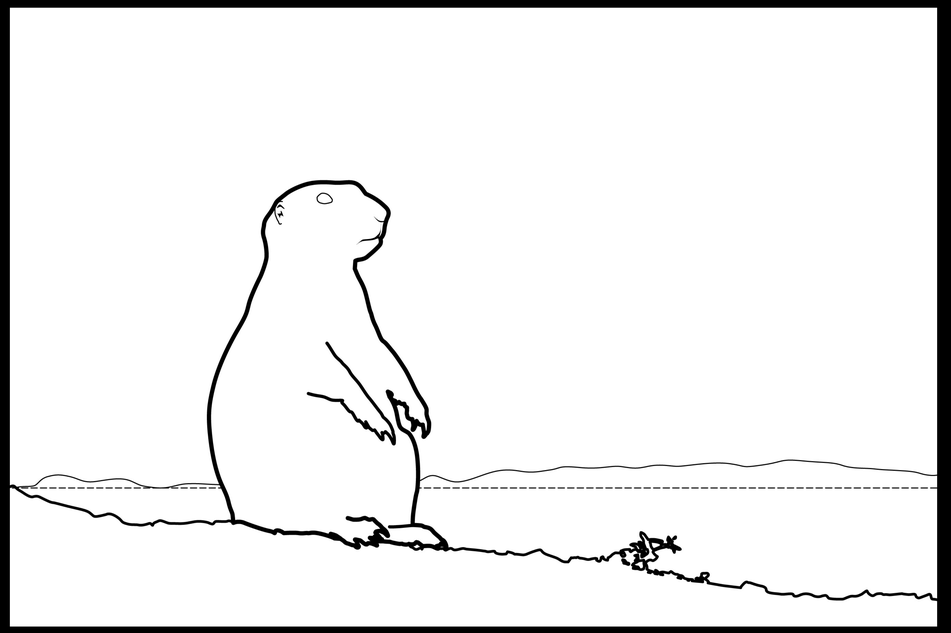
Automatic Backing Store

375w x 250h x @3x x 64 bpp = **2.2 megabytes**



Automatic Backing Store

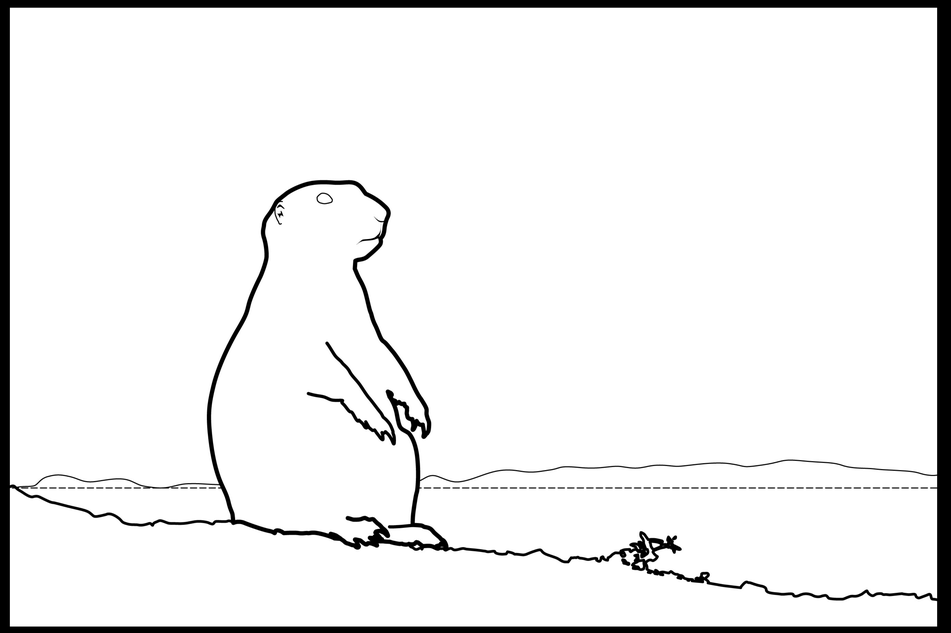
375w x 250h x @3x x 64 bpp = **2.2 megabytes**



375w x 250h x @3x x 64 bpp = **2.2 megabytes**

Automatic Backing Store

375w x 250h x @3x x 64 bpp = **2.2 megabytes**



375w x 250h x @3x x **8 bpp** = **275 kilobytes**

Automatic Backing Store

Automatic is now default

- `UIView.draw()`
- `UIGraphicsImageRenderer`
- `UIGraphicsImageRendererFormat.Range`

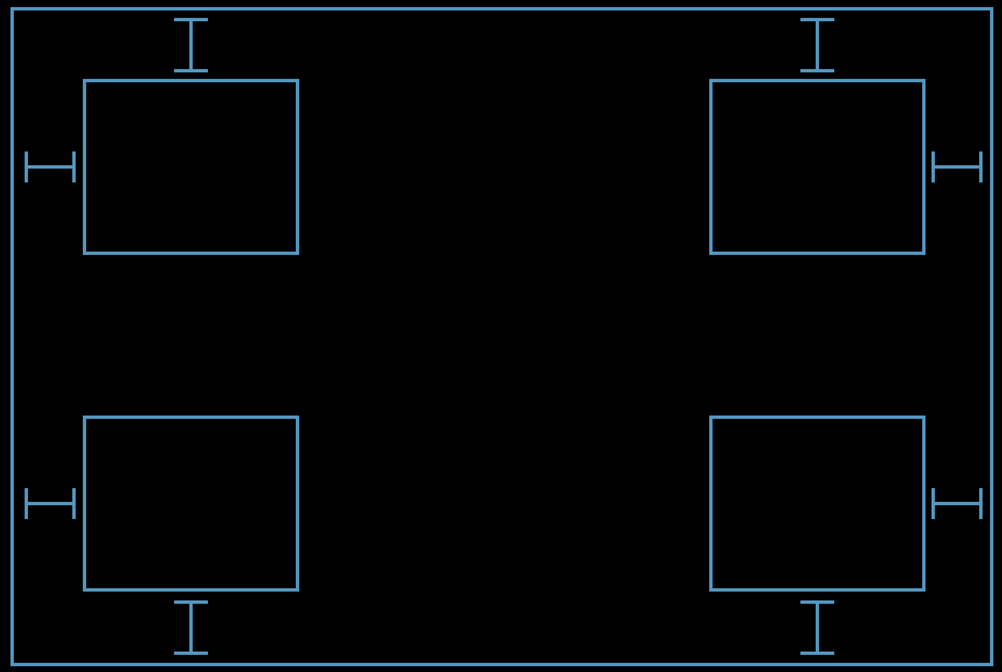
Auto Layout

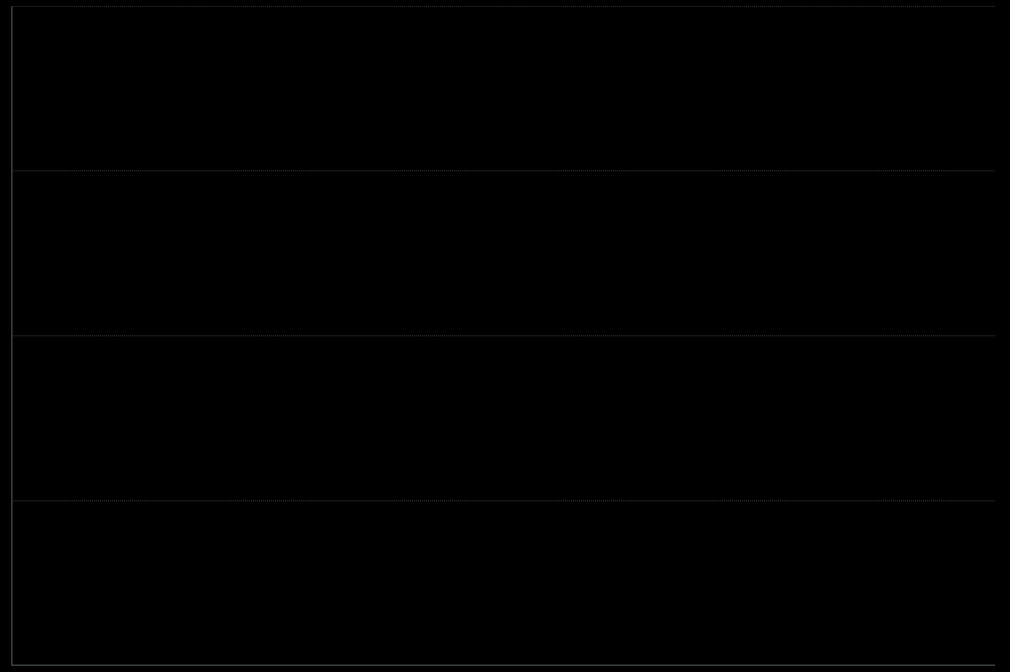
Faster by default

Simple best practices

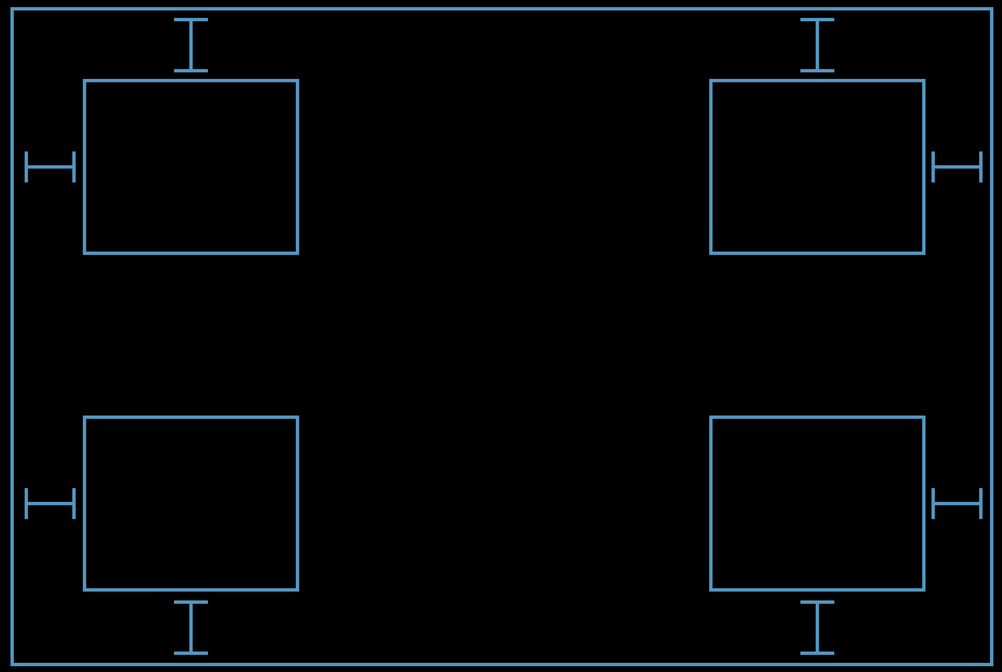
Independent Sibling Views

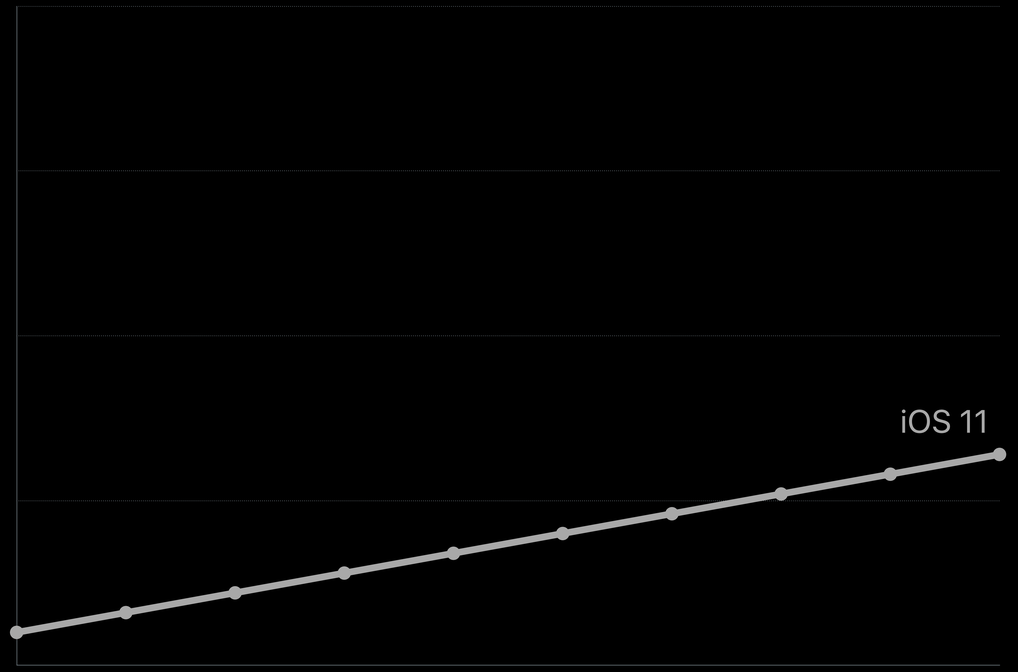
Independent Sibling Views



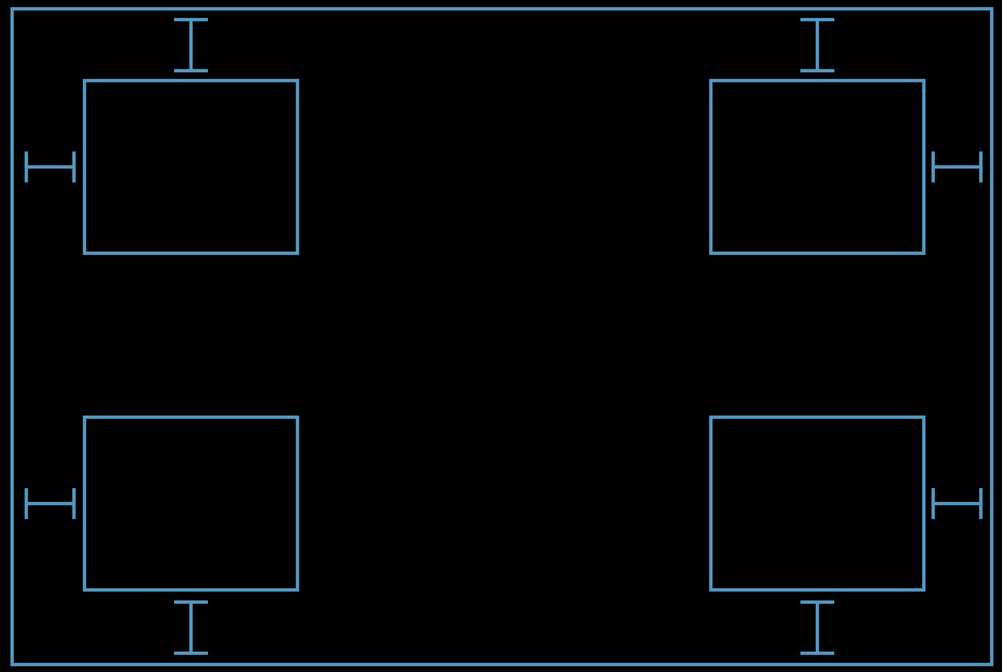


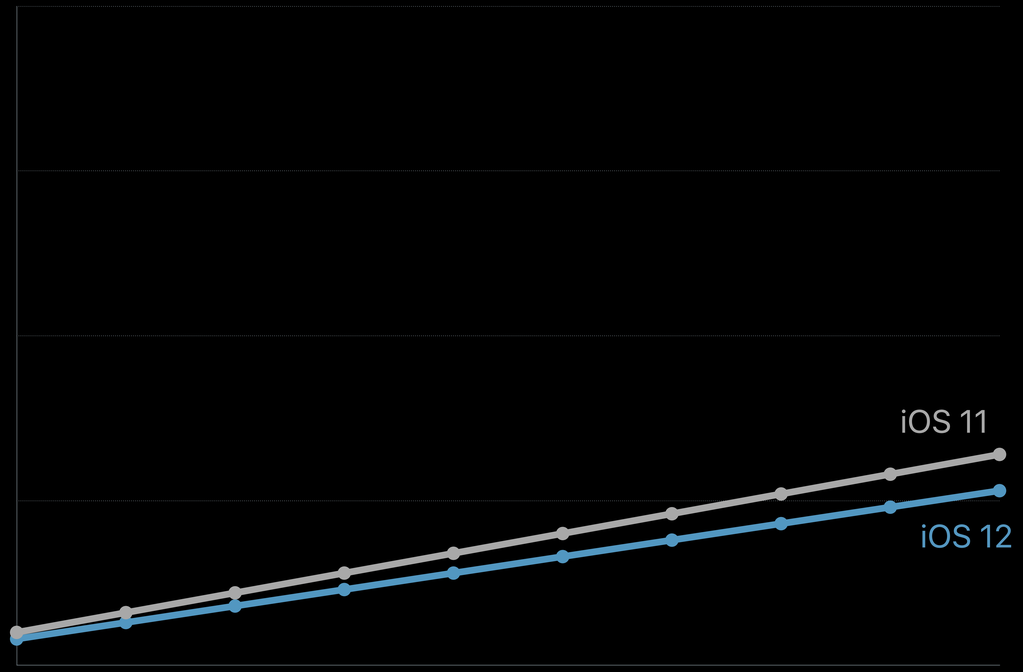
Independent Sibling Views





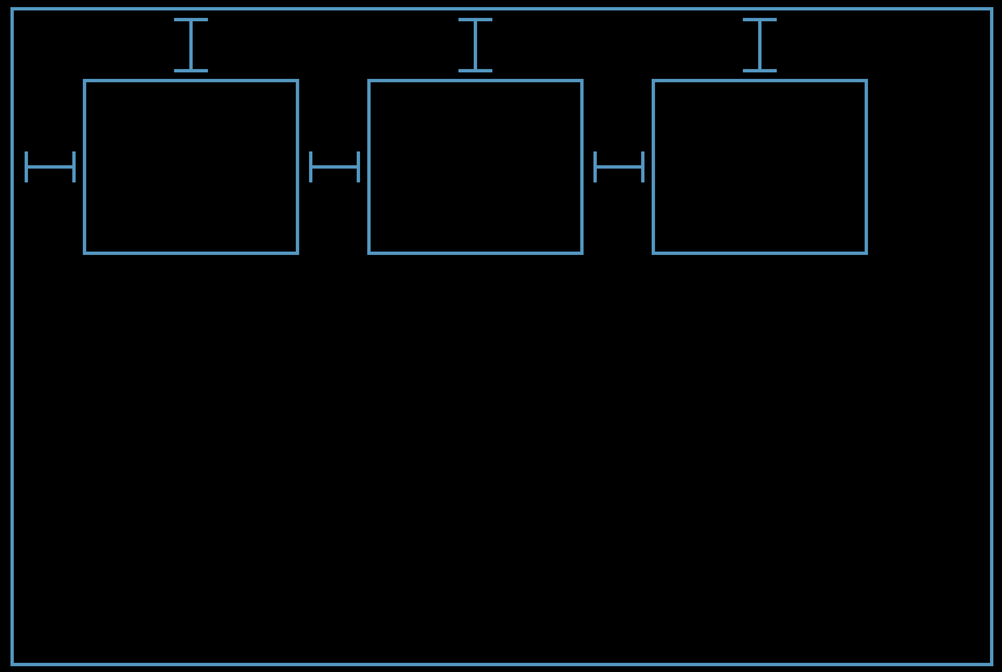
Independent Sibling Views

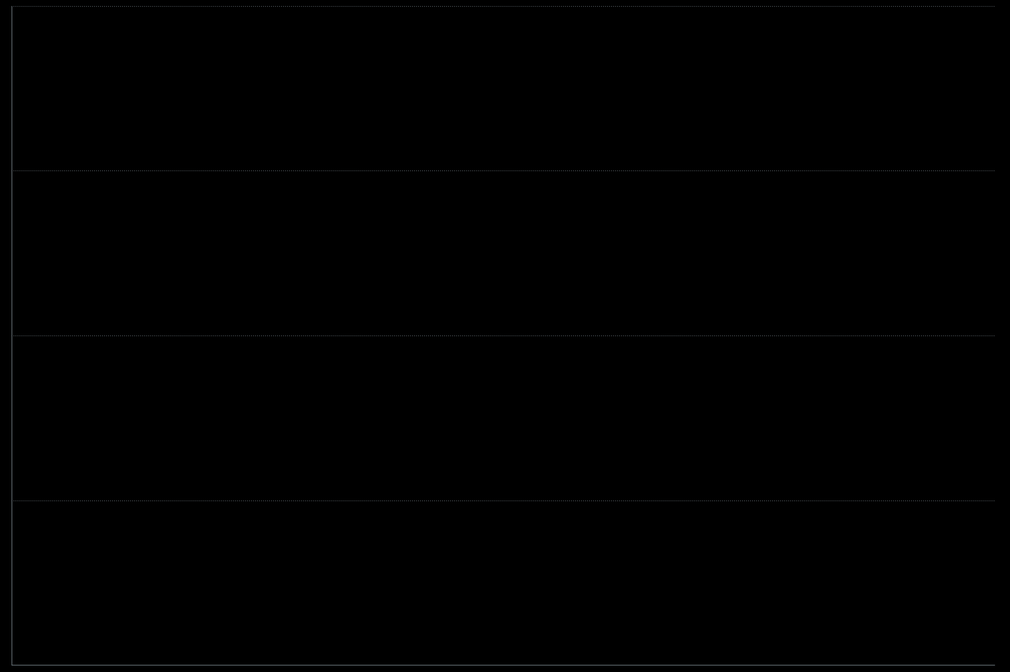




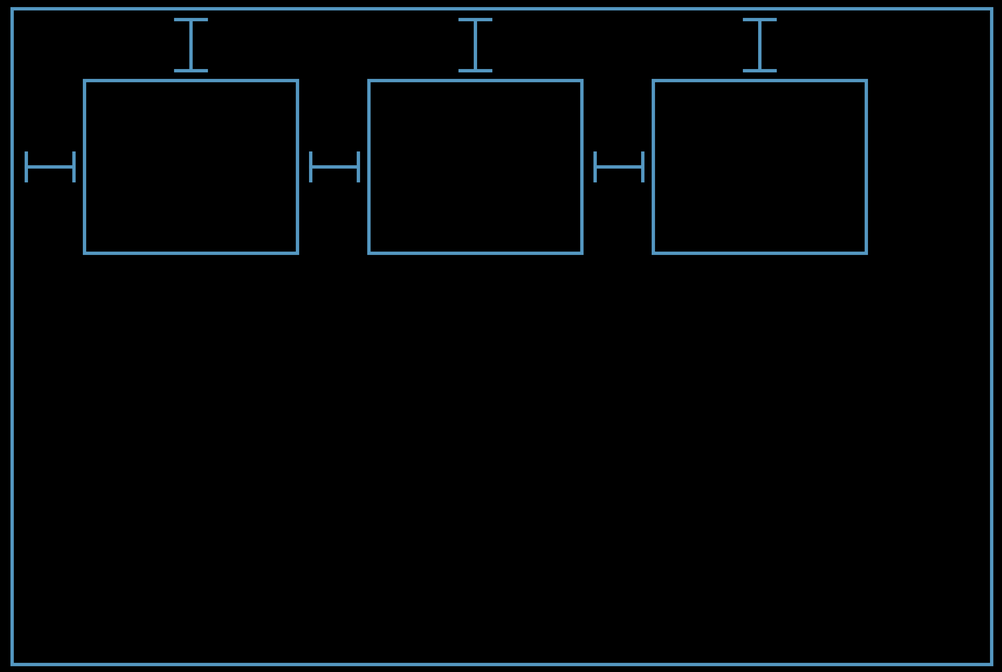
Dependent Sibling Views

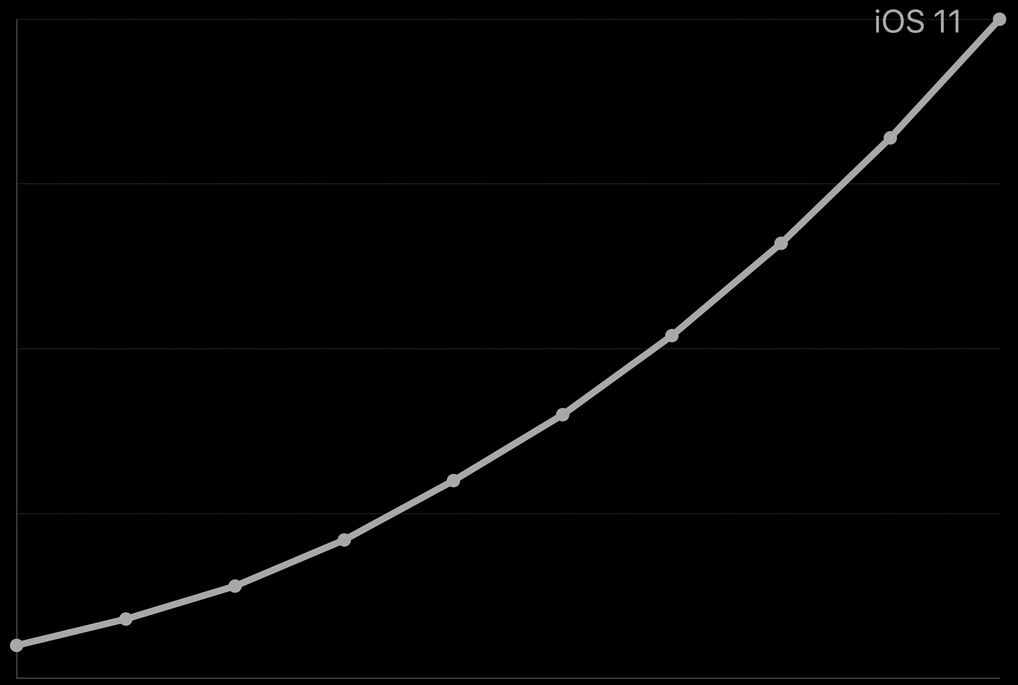
Dependent Sibling Views



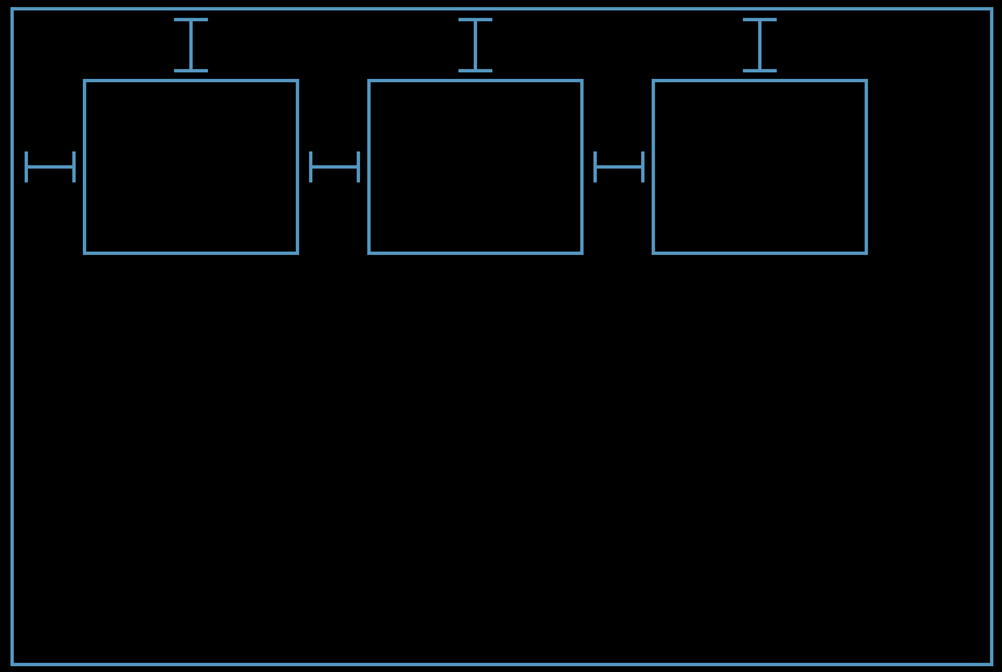


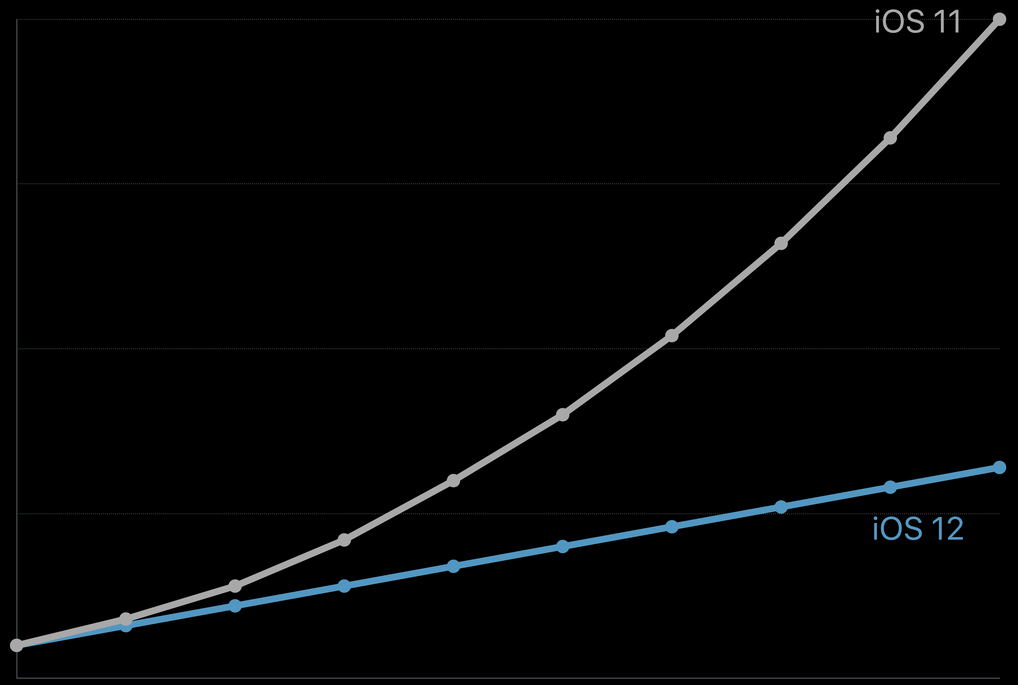
Dependent Sibling Views





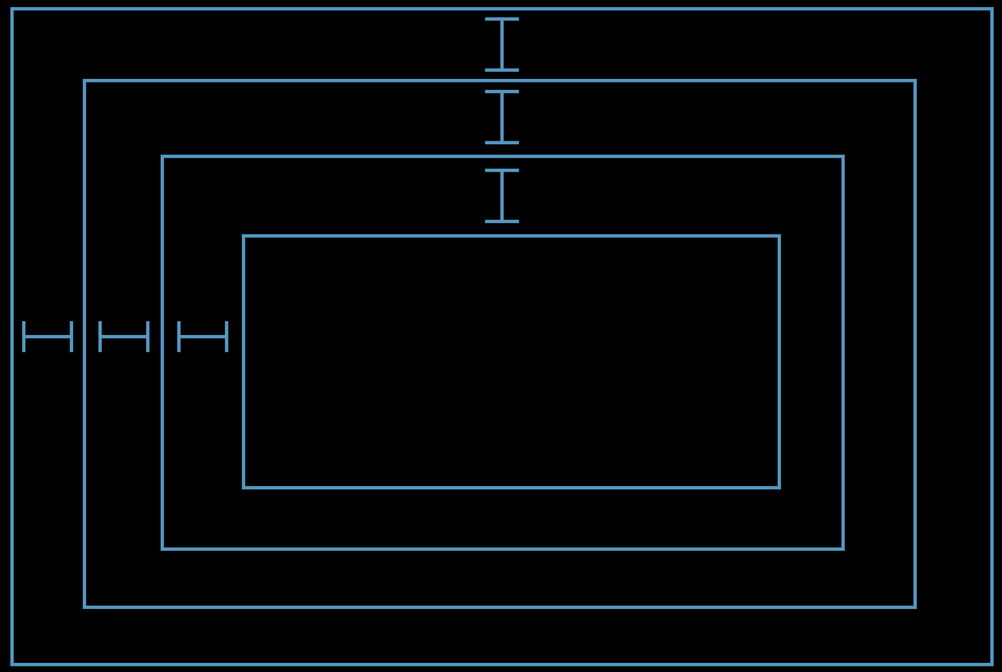
Dependent Sibling Views

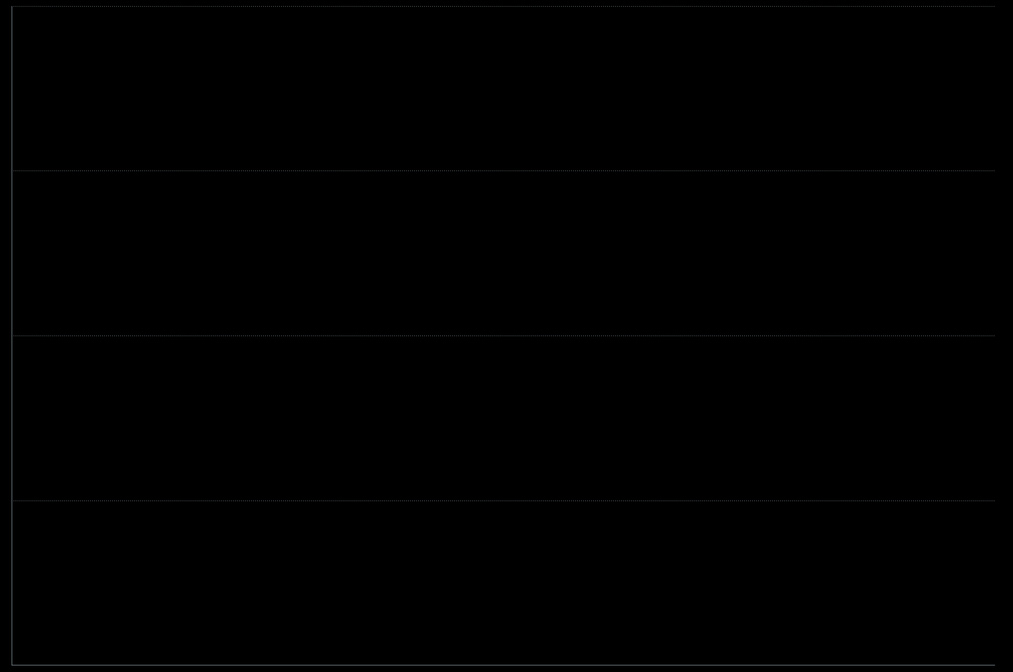




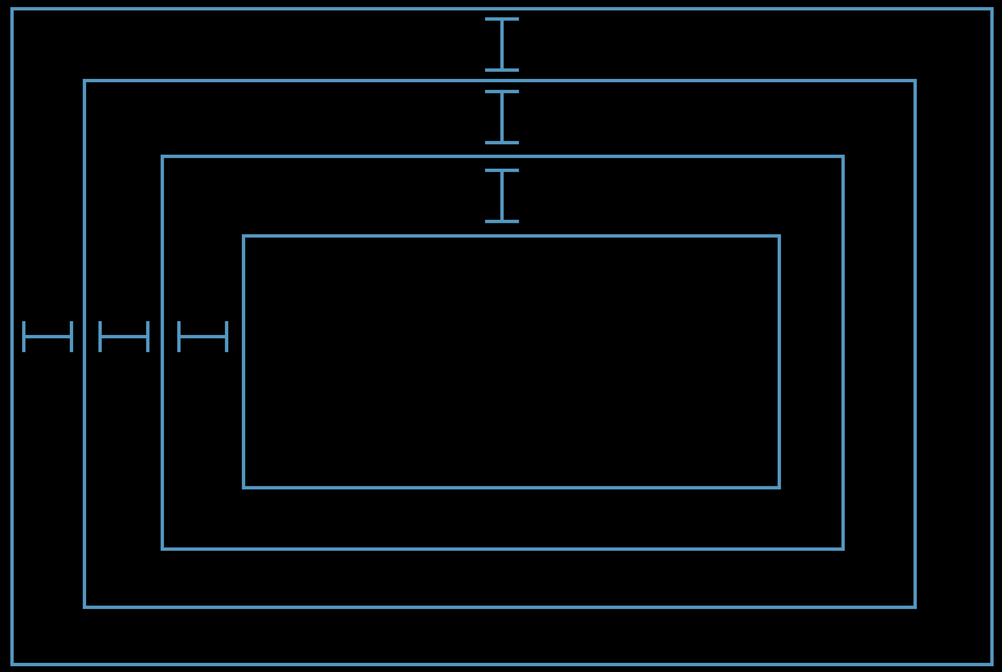
Nested Views

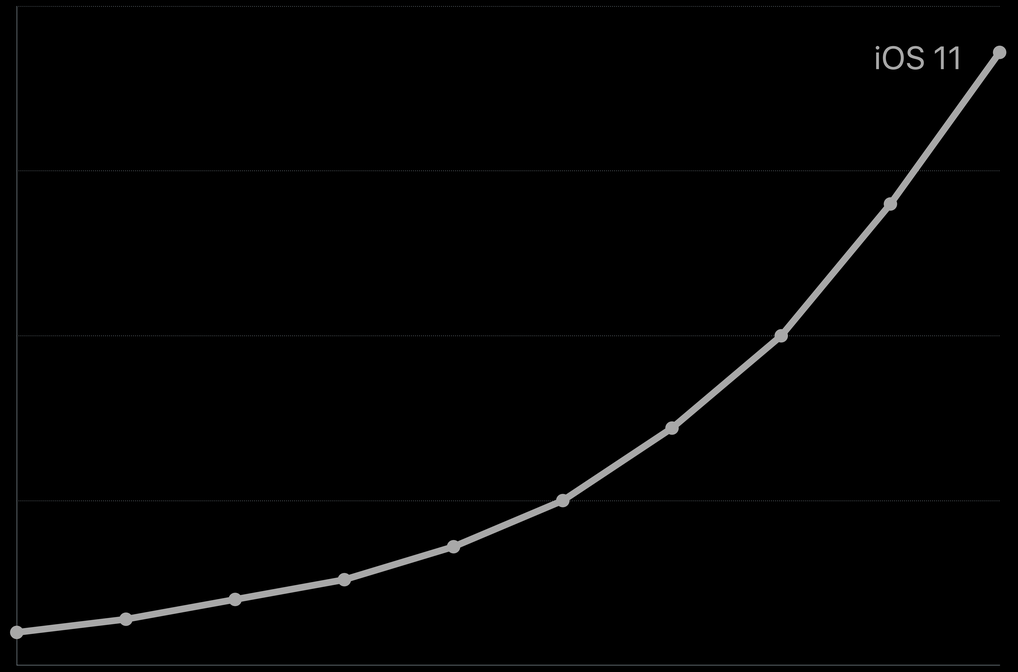
Nested Views



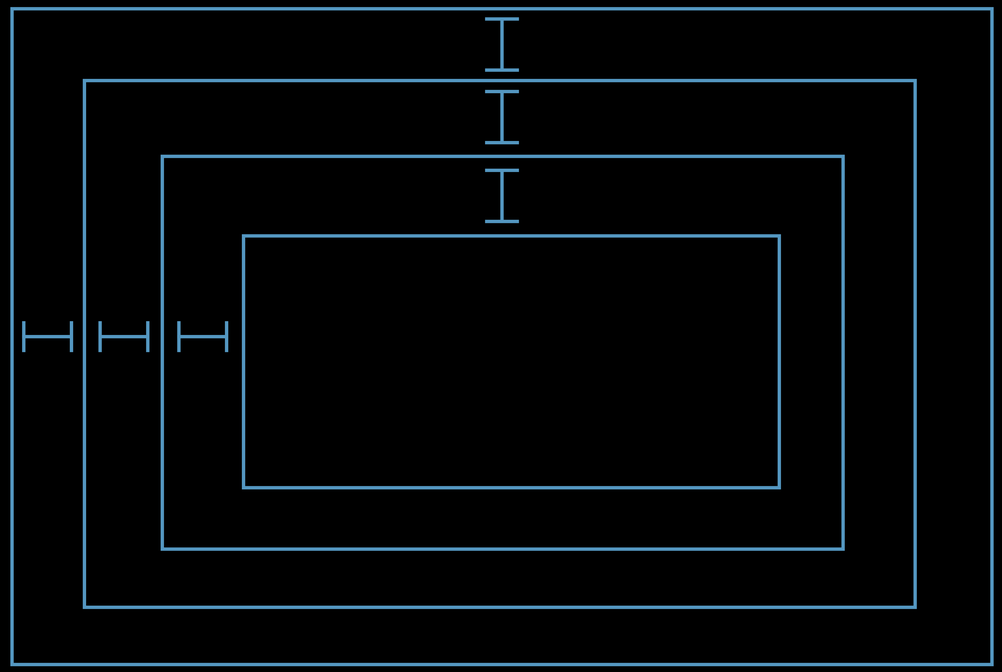


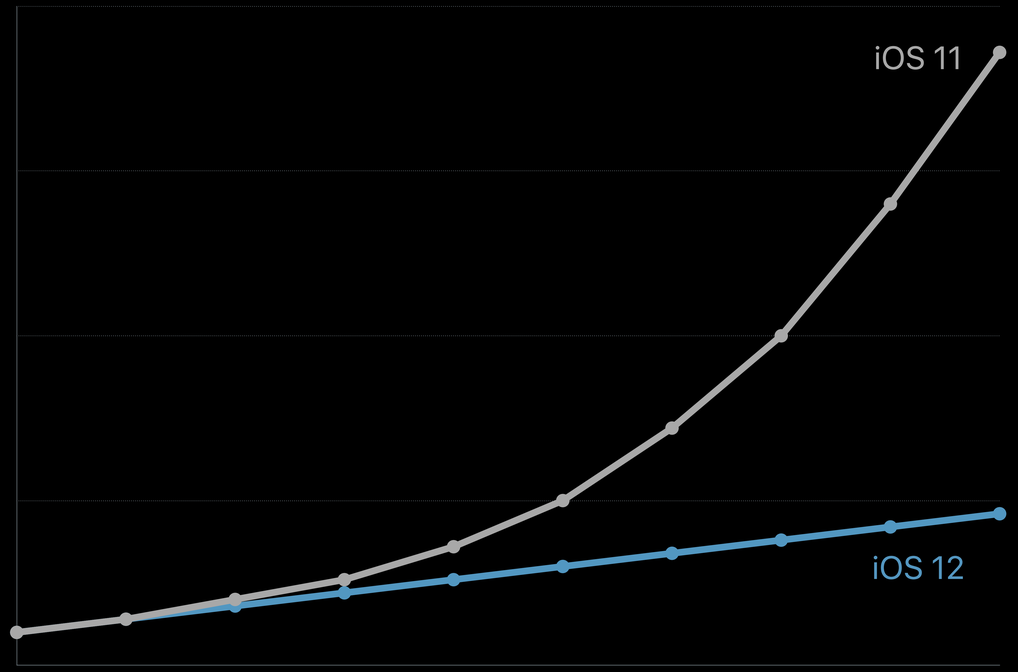
Nested Views



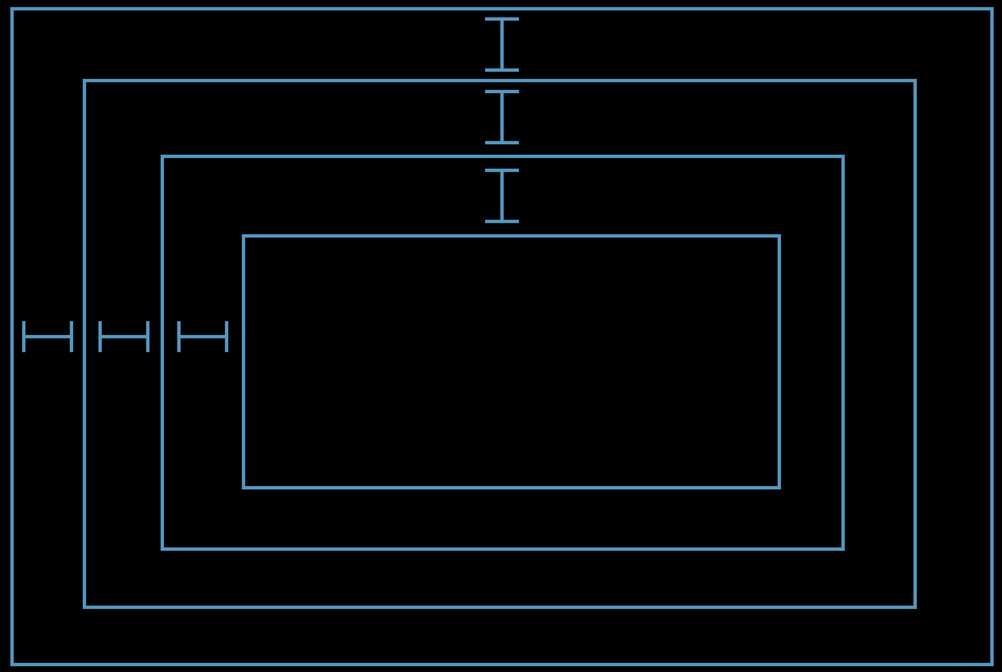


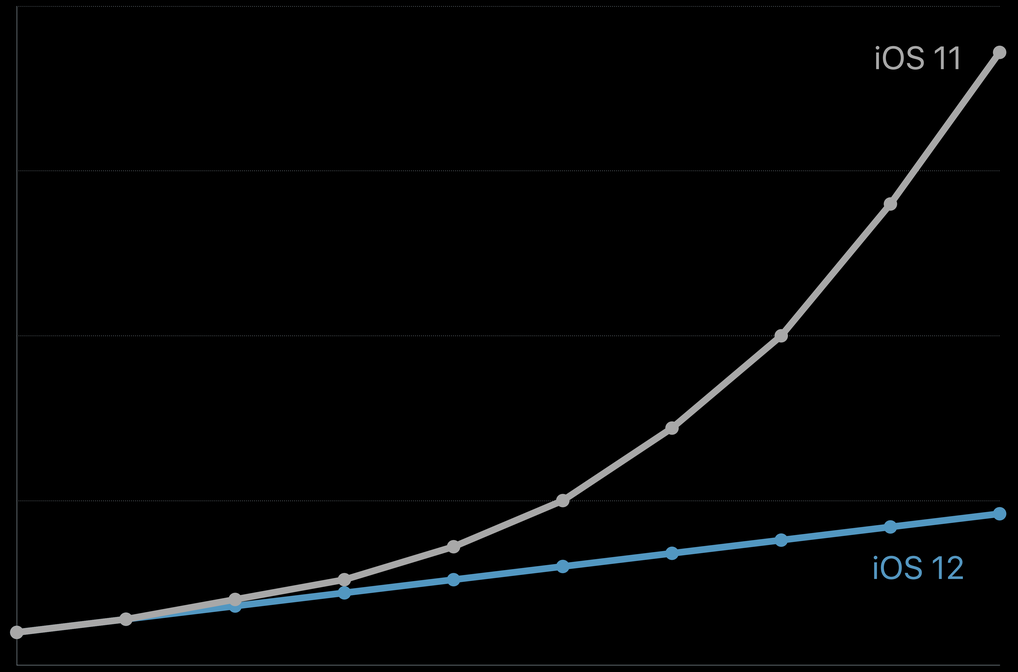
Nested Views





Nested Views





Framework Updates

Swiftification

Nesting

- Types
- Constants
- Functions

```
// Nested Types

// Swift 4
enum UIApplicationState {
    case active
    case inactive
    case background
}
```

```
// Nested Types
```

```
// Swift 4
enum UIApplicationState {
    case active
    case inactive
    case background
}
```

```
// Swift 4.2
class UIApplication : UIResponder {
    enum State {
        case active
        case inactive
        case background
    }
}
```

```
// Nested Types

// Swift 4
enum UITabBarItemPositioning {
    case automatic
    case fill
    case centered
}
```

```
// Nested Types
```

```
// Swift 4
enum UITabBarItemPositioning {
    case automatic
    case fill
    case centered
}
```

```
// Swift 4.2
class UITabBar : UIView {
    enum ItemPositioning {
        case automatic
        case fill
        case centered
    }
}
```

```
// Nested Constants

// Swift 4
class NSNotification : NSObject {
    struct Name {
        class let didChangeStatusBarOrientation: NSNotification.Name
    }
}

let UIApplicationStatusBarOrientationUserInfoKey: String
```

```
// Nested Constants

// Swift 4
class NSNotification : NSObject {
    struct Name {
        class let didChangeStatusBarOrientation: NSNotification.Name
    }
}

let UIApplicationStatusBarOrientationUserInfoKey: String

// Swift 4.2
class UIApplication : UIResponder {
    class let didChangeStatusBarOrientationNotification: NSNotification.Name
    class let statusBarOrientationUserInfoKey: String
}
```

```
// Nested Constants

// Swift 4
let UIFloatRangeZero: UIFloatRange
let UIFloatRangeInfinite: UIFloatRange
```

```
// Nested Constants

// Swift 4
let UIFloatRangeZero: UIFloatRange
let UIFloatRangeInfinite: UIFloatRange

// Swift 4.2
struct UIFloatRange {
    static let zero: UIFloatRange
    static let infinite: UIFloatRange
}
```

```
// Nested Functions

let insets = UIEdgeInsets(top: 8, left: 0, bottom: 8, right: 0)
let image = UIImage(named: “Apple”)

// Swift 4
let insetRect = UIEdgeInsetsInsetRect(originalRect, insets)
let pngData = UIImagePNGRepresentation(image)
```

```
// Nested Functions

let insets = UIEdgeInsets(top: 8, left: 0, bottom: 8, right: 0)
let image = UIImage(named: “Apple”)

// Swift 4
let insetRect = UIEdgeInsetsInsetRect(originalRect, insets)
let pngData = UIImagePNGRepresentation(image)

// Swift 4.2
let insetRect = originalRect.insetBy(insets)
let pngData = image.pngData()
```

```
// Nested Functions — String Conversion

// Swift 4
NSStringFrom[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]
[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]FromString
```

```
// Nested Functions — String Conversion

// Swift 4
NSStringFrom[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]
[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]FromString

// Swift 4.2 - Codable Conformance
let encoded = JSONEncoder().encode(CGPoint(x: 0, y: 0))
let decoded = JSONDecoder().decode(CGPoint.self, from: encoded)
```

```
// Nested Functions — String Conversion

// Swift 4
NSStringFrom[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]
[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]FromString

// Swift 4.2 — Codable Conformance
let encoded = JSONEncoder().encode(CGPoint(x: 0, y: 0))
let decoded = JSONDecoder().decode(CGPoint.self, from: encoded)

// Swift 4.2 — Debug Printing
print(CGPoint(x: 0, y: 0))
print("Offset: \(UIOffset(horizontal: 10, vertical: 10))")
```

```
// Nested Functions — String Conversion

// Swift 4
NSStringFrom[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]
[CGPoint, CGRect, CGSize, CGVector, CGAffineTransform, UIEdgeInsets, UIOffset]FromString

// Swift 4.2 — Codable Conformance
let encoded = JSONEncoder().encode(CGPoint(x: 0, y: 0))
let decoded = JSONDecoder().decode(CGPoint.self, from: encoded)

// Swift 4.2 — Debug Printing
print(CGPoint(x: 0, y: 0))
print("Offset: \(UIOffset(horizontal: 10, vertical: 10))")

// Swift 4.2 — Legacy Encoding/Decoding
let encoded = NSCoder.string(for: CGPoint(x: 0, y: 0))
let decoded = NSCoder.cgPoint(for: encoded)
```

Framework Updates

NSSecureCoding

New secure-by-default encoding and decoding APIs

Older APIs now deprecated

API Enhancements

Notifications

Interaction

Grouping

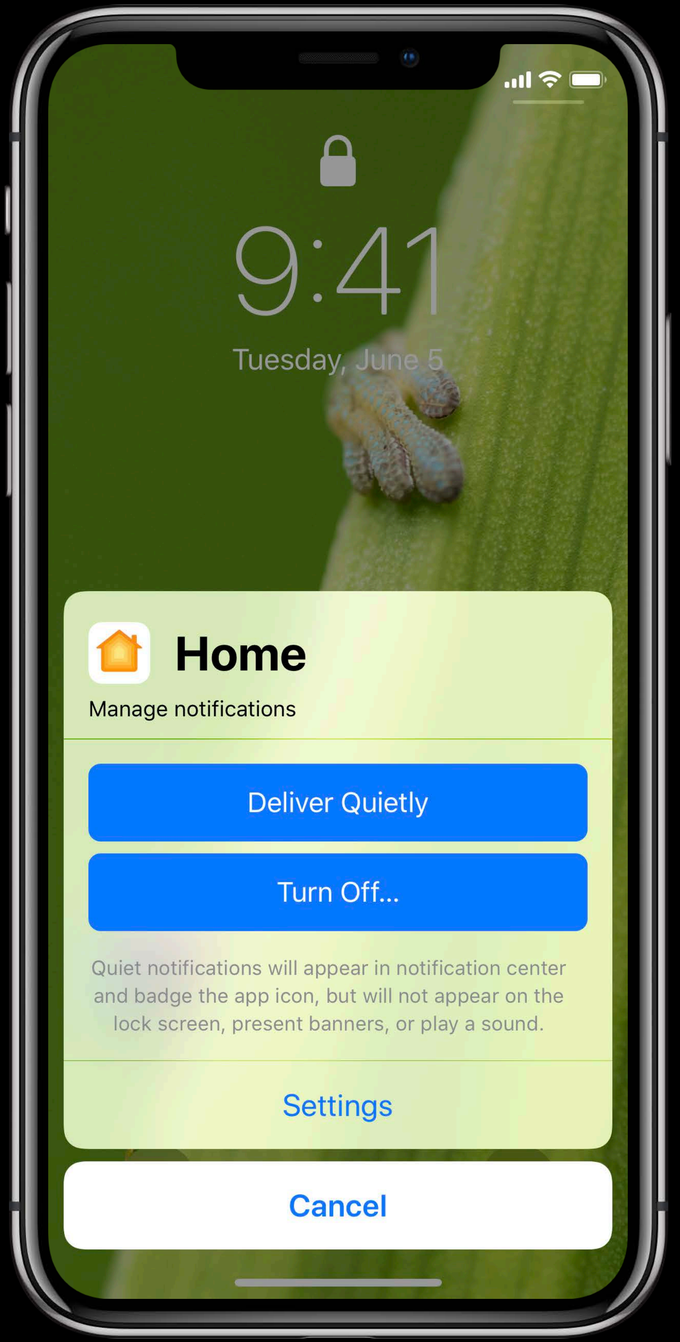
Settings

Interaction

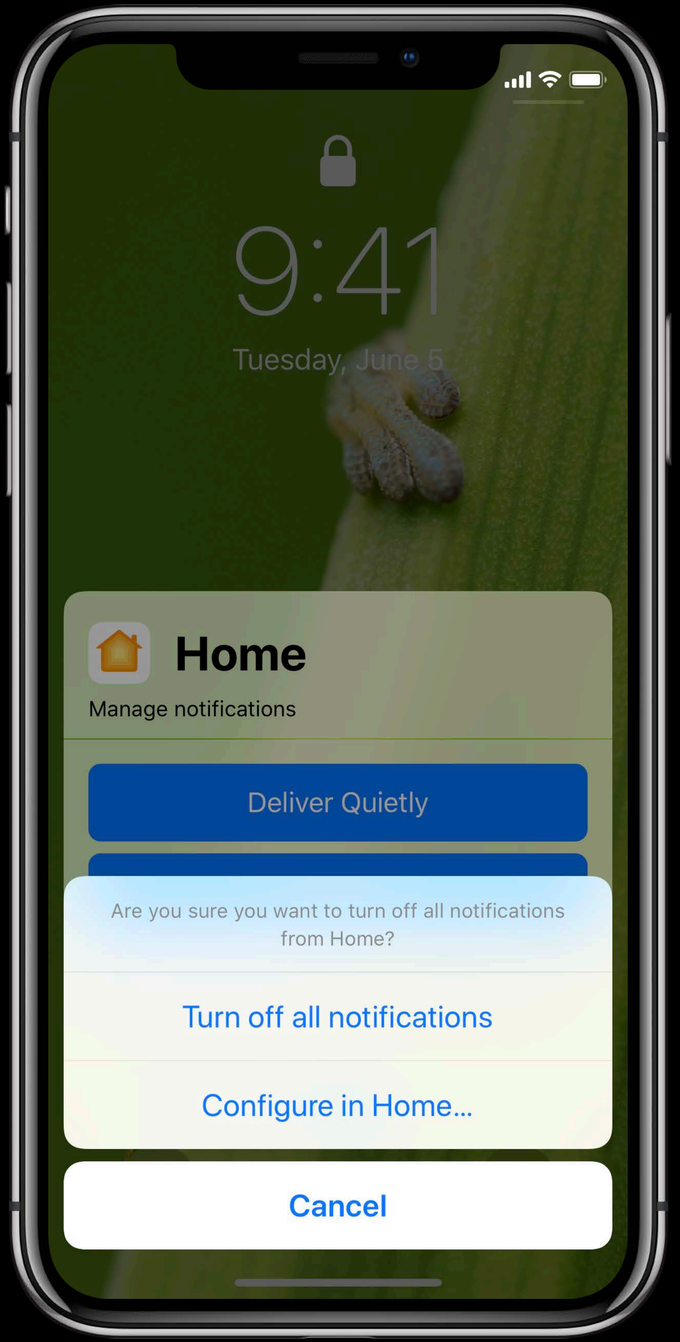
Grouping


9:41
Tuesday, June 5
Notification Center
HOME
27m ago
11 Notifications
CALENDAR
6/21, 10:00 AM
Invitation
PHOTOS
Sun 10:53 PM
2 Notifications

Settings


9:41
Tuesday, June 5
Home
Manage notifications
Deliver Quietly
Turn Off...
Quiet notifications will appear in notification center and badge the app icon, but will not appear on the lock screen, present banners, or play a sound.
Settings
Cancel

Settings


9:41
Tuesday, June 5
Home
Manage notifications
Deliver Quietly
Are you sure you want to turn off all notifications from Home?
Turn off all notifications
Configure in Home...
Cancel

Settings

What’s New in User Notifications	Hall 3	Wednesday 2:00PM
Using Grouped Notifications	Hall 3	Wednesday 3:00PM

Messages

Messages

Messages

```
// New Info.plist key

<key>MSSupportedPresentationContexts</key>
<array>
    <string>MSMessagesAppPresentationContextMessages</string>
    <string>MSMessagesAppPresentationContextMedia</string>
</array>
```

```
// New Info.plist key

<key>MSSupportedPresentationContexts</key>
<array>
    <string>MSMessagesAppPresentationContextMessages</string>
    <string>MSMessagesAppPresentationContextMedia</string>
</array>

var presentationContext: MSMessagesAppPresentationContext

enum MSMessagesAppPresentationContext : UInt {
    case messages
    case media
}
```

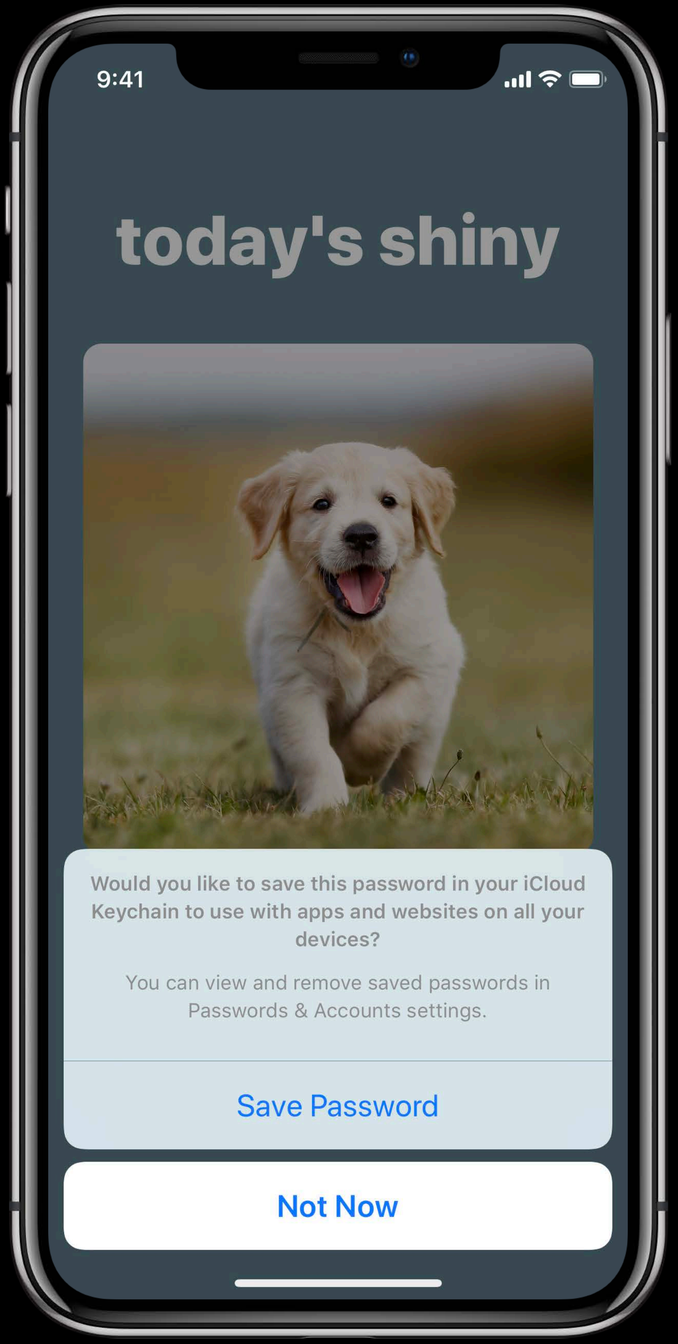

9:41
19
Eliza
iMessage
Pay

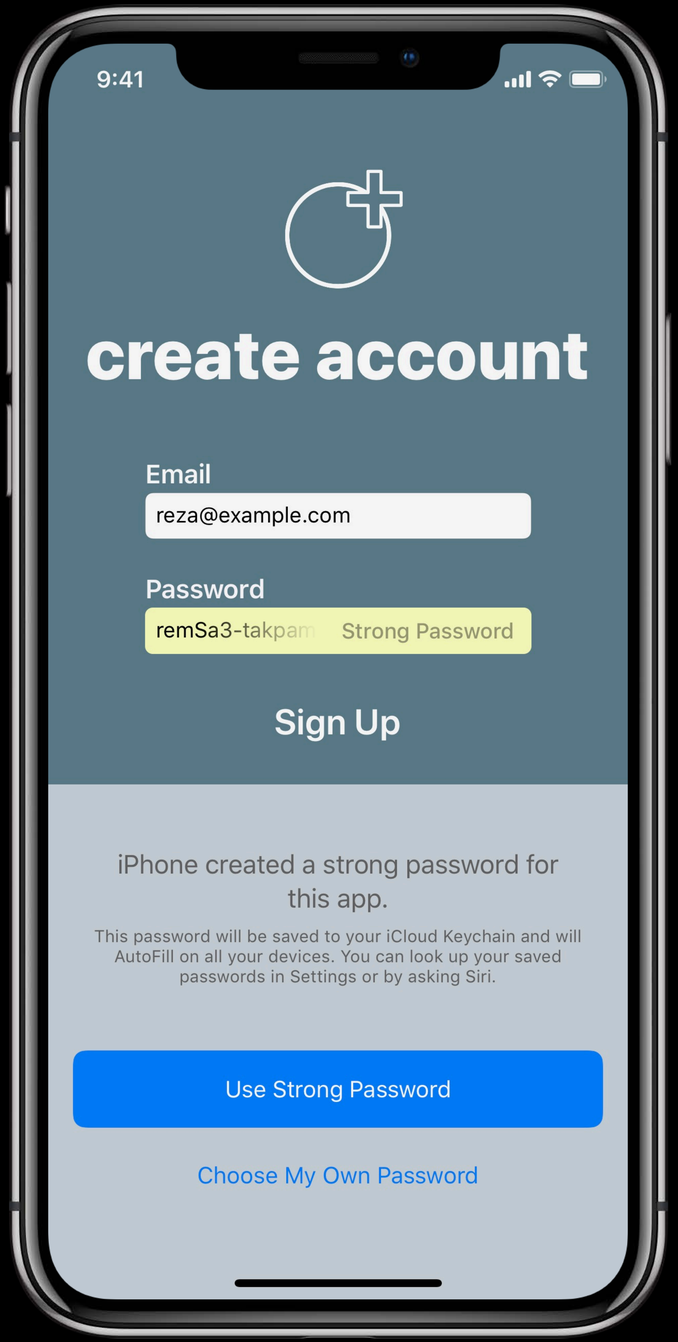

9:41
19
Eliza
iMessage
Pay

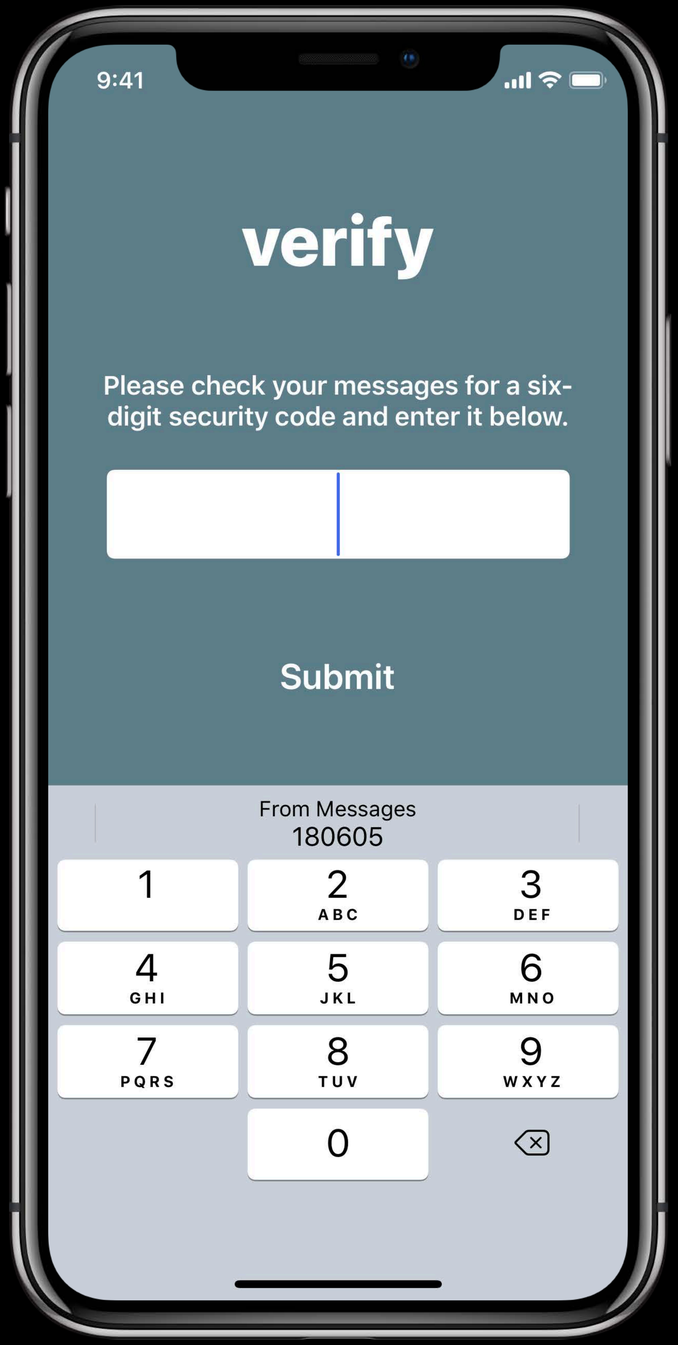

9:41
19
Eliza
iMessage
Pay

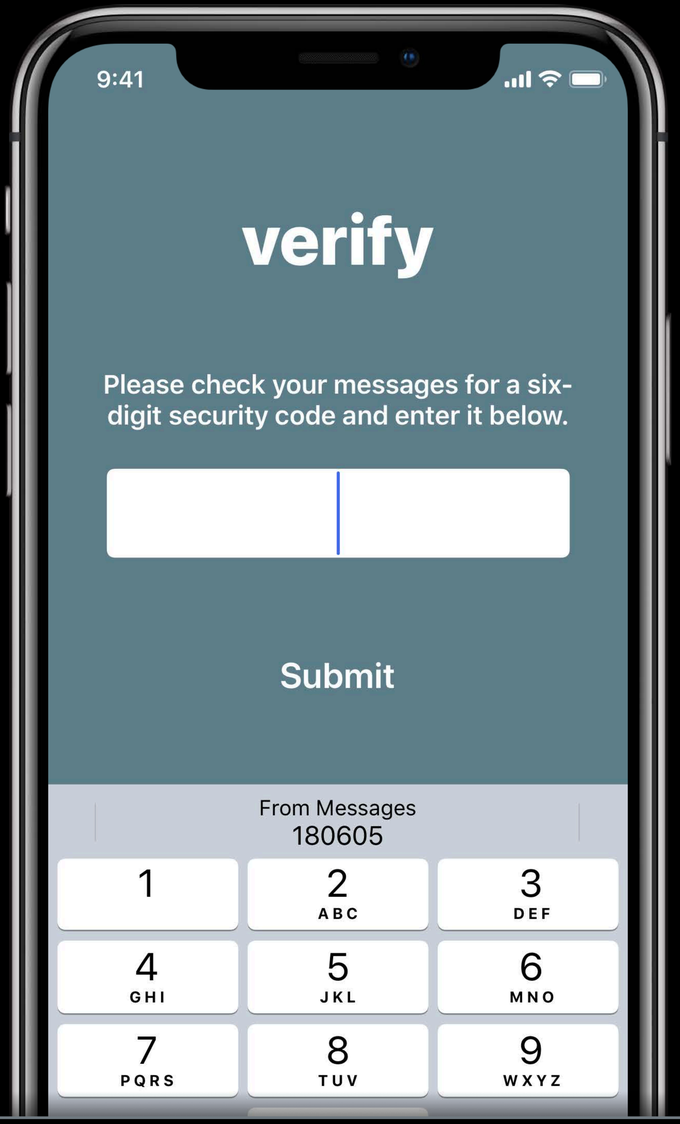
Automatic Passwords and Security Code AutoFill


9:41
shiny
Email
Password
Log In
password for shiny.example.com
chelsea@example.com
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
@
.
return


9:41
today's shiny
Would you like to save this password in your iCloud Keychain to use with apps and websites on all your devices?
You can view and remove saved passwords in Passwords & Accounts settings.
Save Password
Not Now

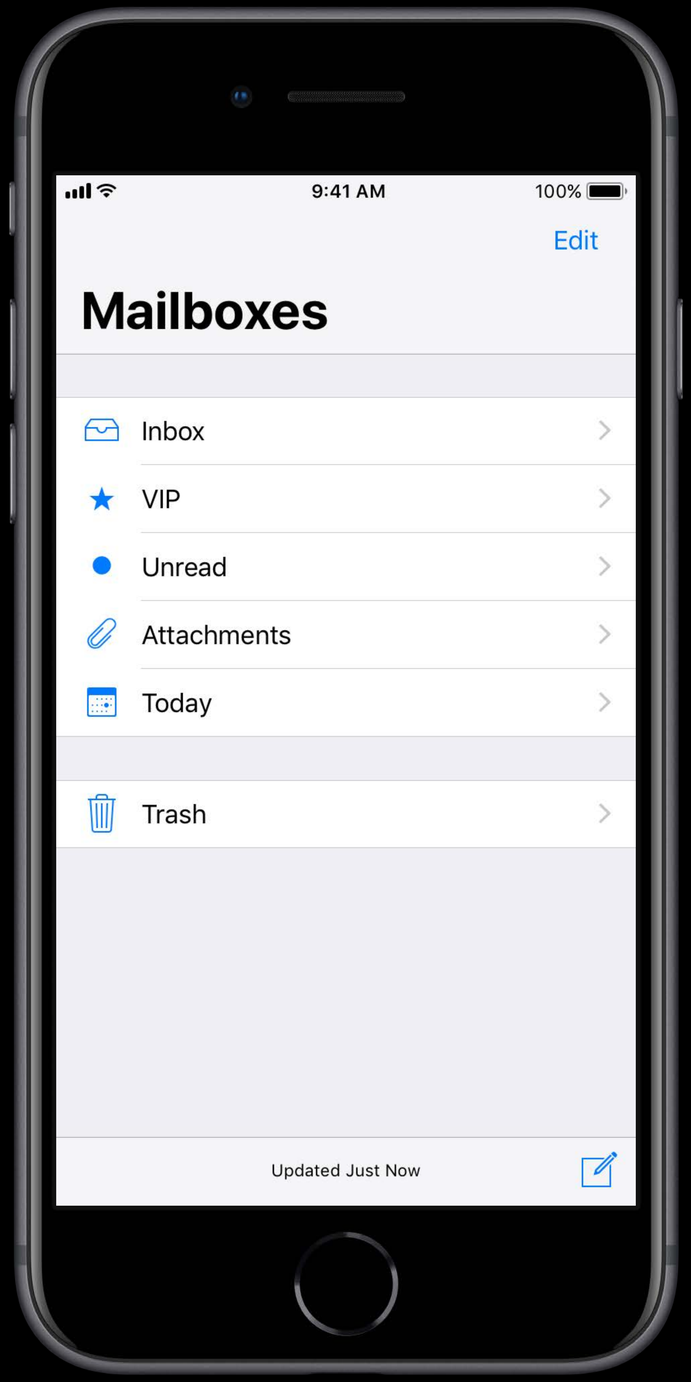

9:41
create account
Email
reza@example.com
Password
remSa3-takpam
Strong Password
Sign Up
iPhone created a strong password for this app.
This password will be saved to your iCloud Keychain and will AutoFill on all your devices. You can look up your saved passwords in Settings or by asking Siri.
Use Strong Password
Choose My Own Password


9:41
verify
Please check your messages for a six-digit security code and enter it below.
Submit
From Messages
180605
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
0

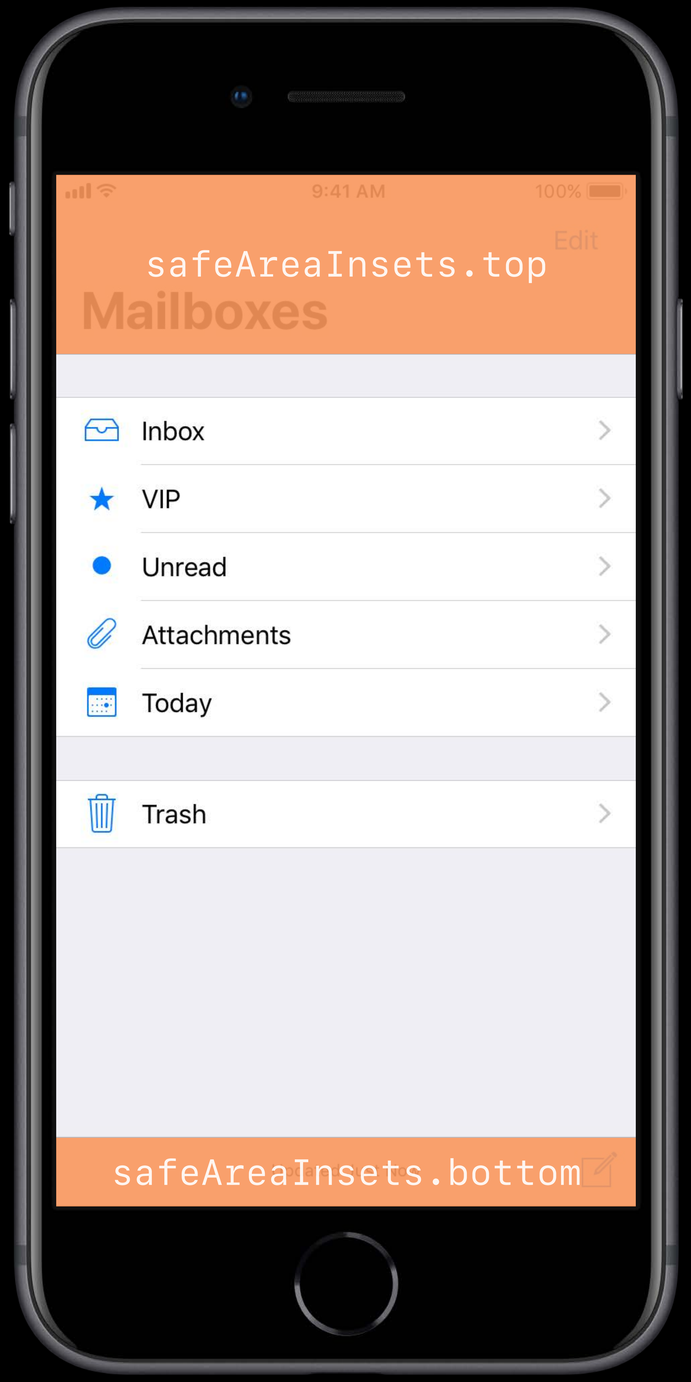


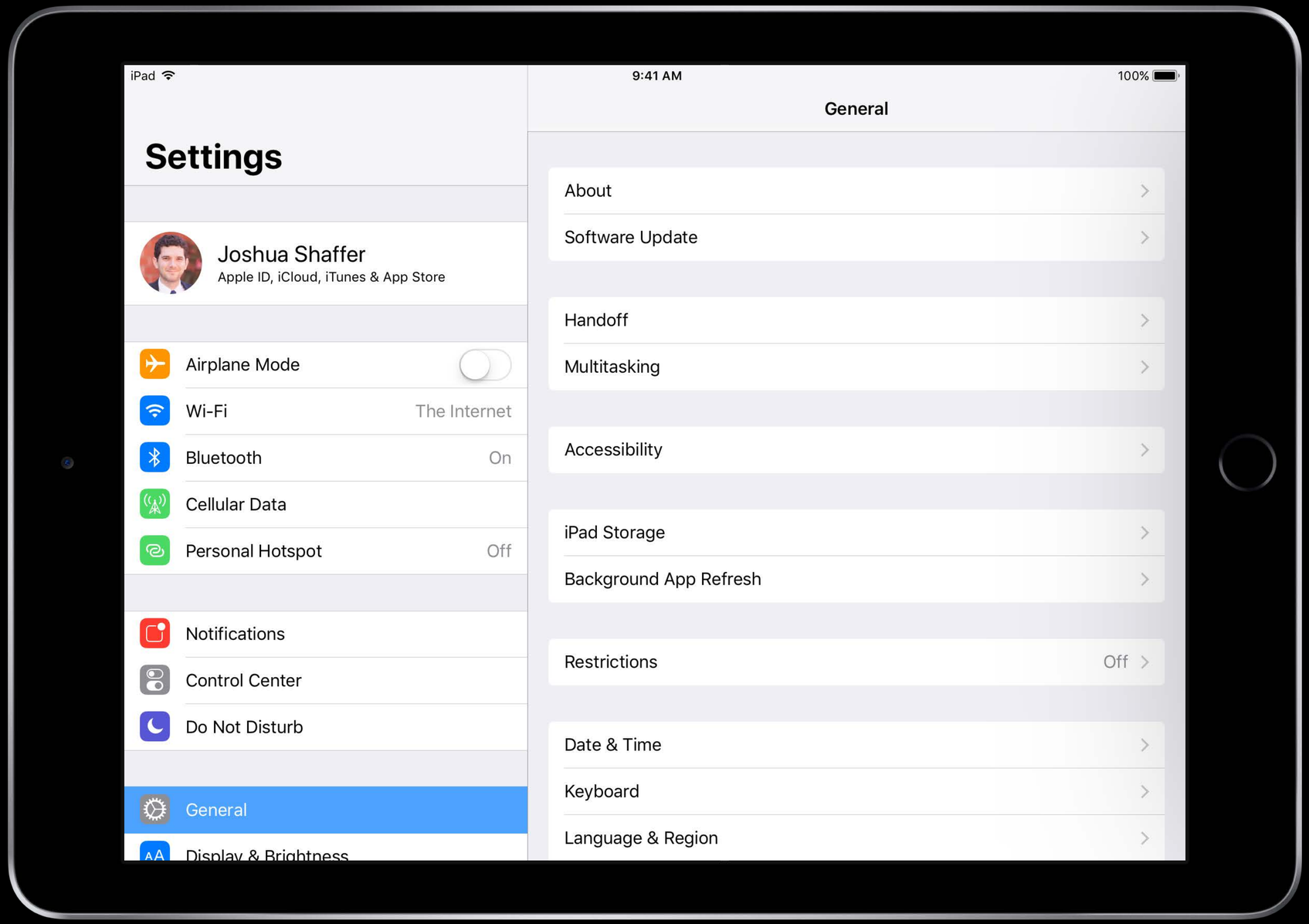
Automatic Strong Passwords and Security Code AutoFill | Hall 1 | Tuesday 11:00AM

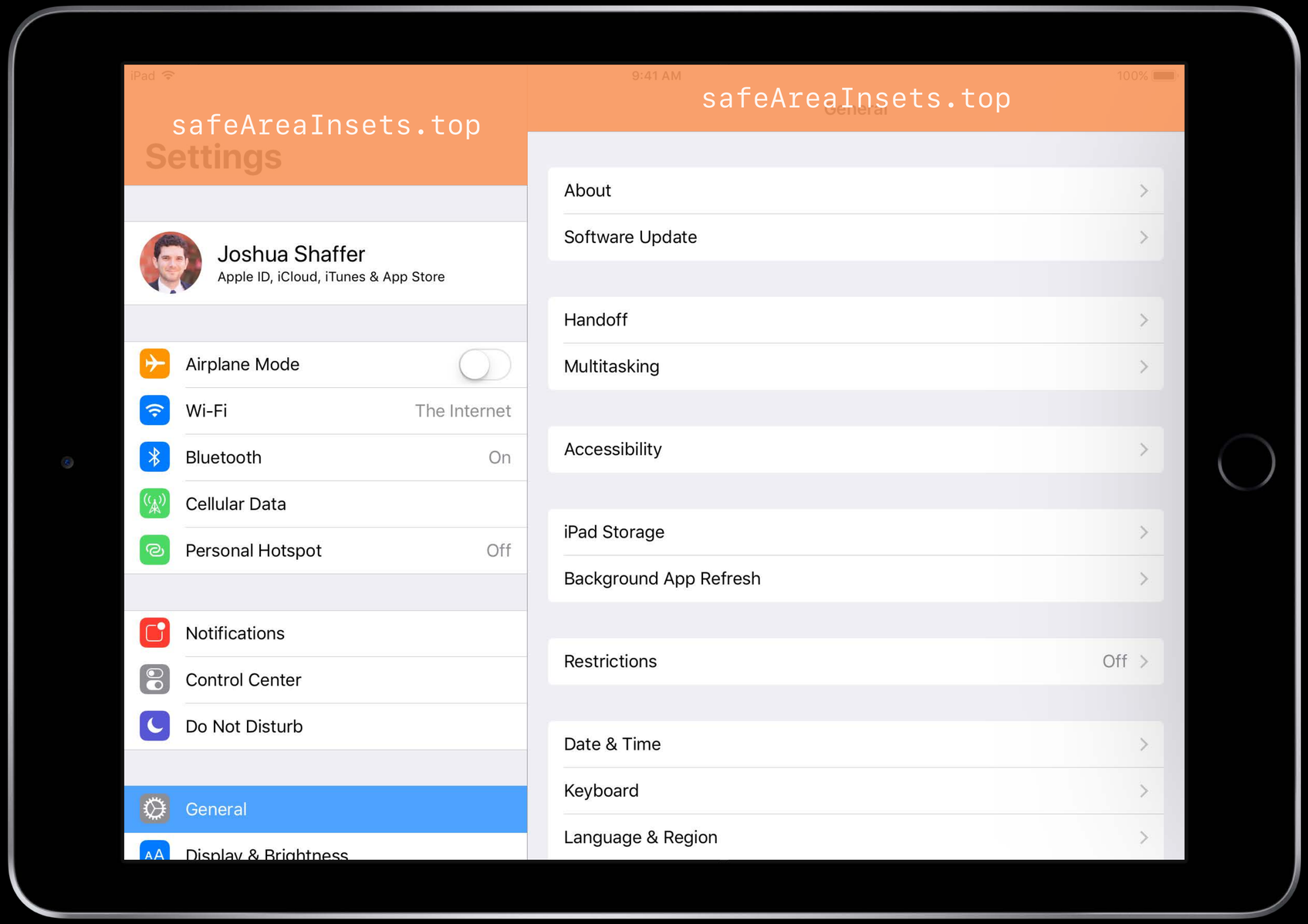
Safe Area

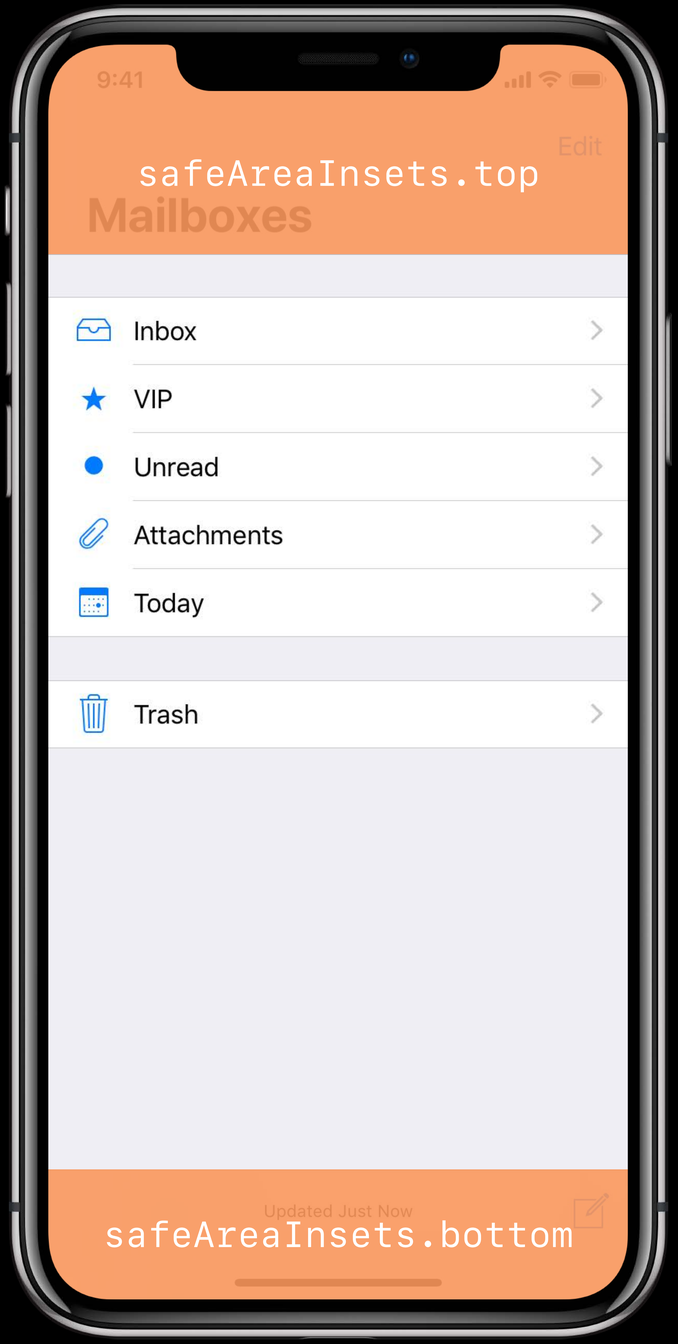

9:41 AM
100%
Edit
Mailboxes
Inbox
VIP
Unread
Attachments
Today
Trash
Updated Just Now

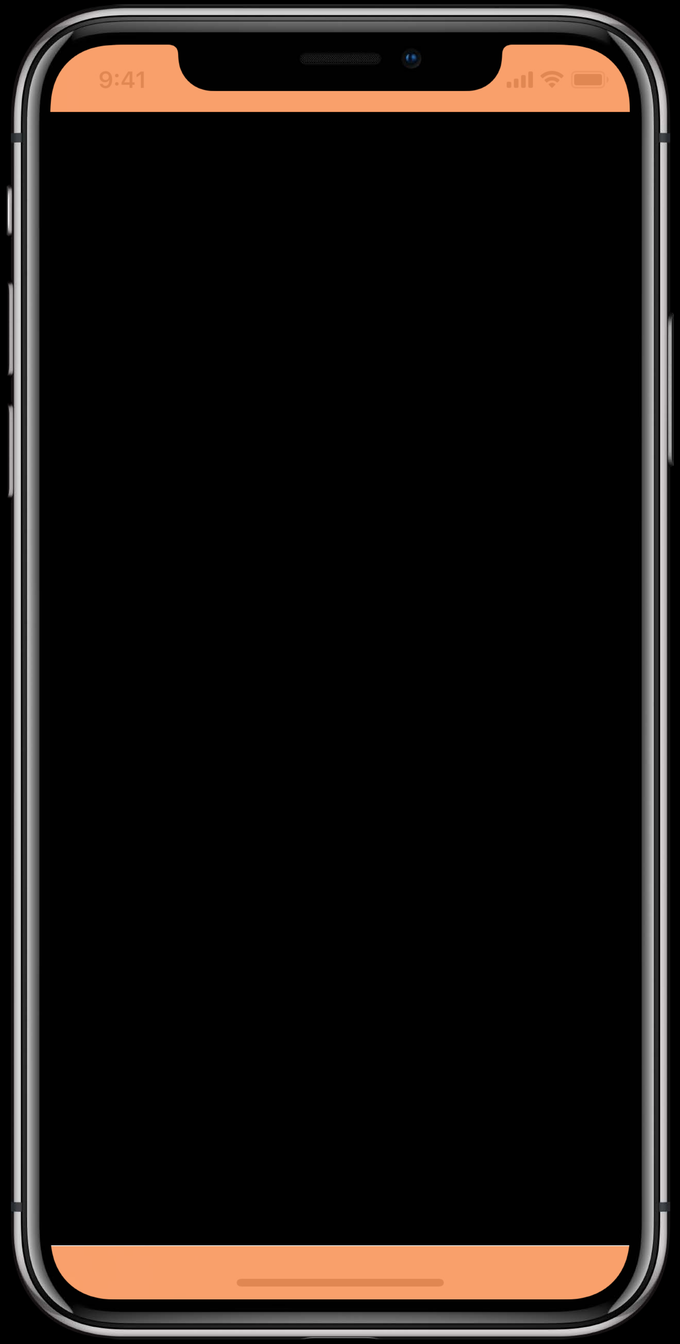
Safe Area

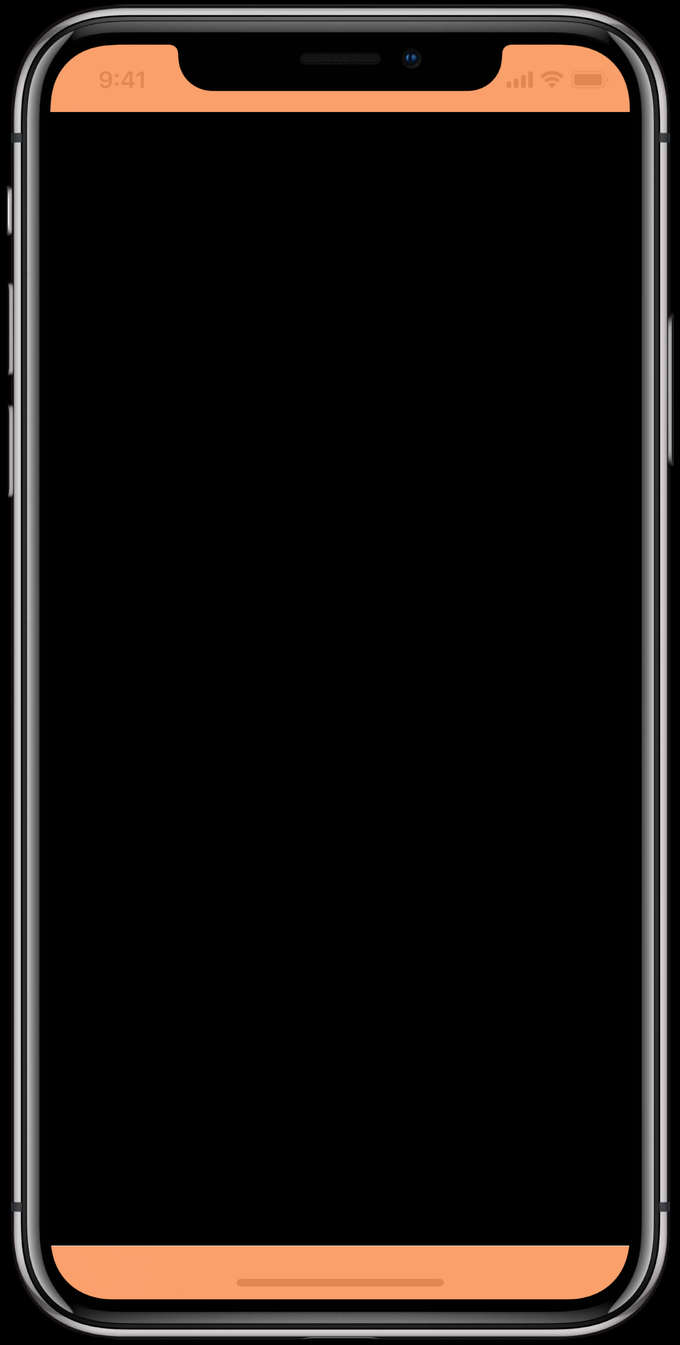

9:41 AM
100%
safeAreaInsets.top
Edit
Mailboxes
Inbox
VIP
Unread
Attachments
Today
Trash
safeAreaInsets.bottom


iPad
9:41 AM
100%
Settings
Joshua Shaffer
Apple ID, iCloud, iTunes & App Store
Airplane Mode
Wi-Fi
The Internet
Bluetooth
On
Cellular Data
Personal Hotspot
Off
Notifications
Control Center
Do Not Disturb
General
Display & Brightness
General
About
Software Update
Handoff
Multitasking
Accessibility
iPad Storage
Background App Refresh
Restrictions
Off
Date & Time
Keyboard
Language & Region


iPad
9:41 AM
100%
safeAreaInsets.top
Settings
safeAreaInsets.top
General
Joshua Shaffer
Apple ID, iCloud, iTunes & App Store
Airplane Mode
Wi-Fi
The Internet
Bluetooth
On
Cellular Data
Personal Hotspot
Off
Notifications
Control Center
Do Not Disturb
General
Display & Brightness
About
Software Update
Handoff
Multitasking
Accessibility
iPad Storage
Background App Refresh
Restrictions
Off
Date & Time
Keyboard
Language & Region

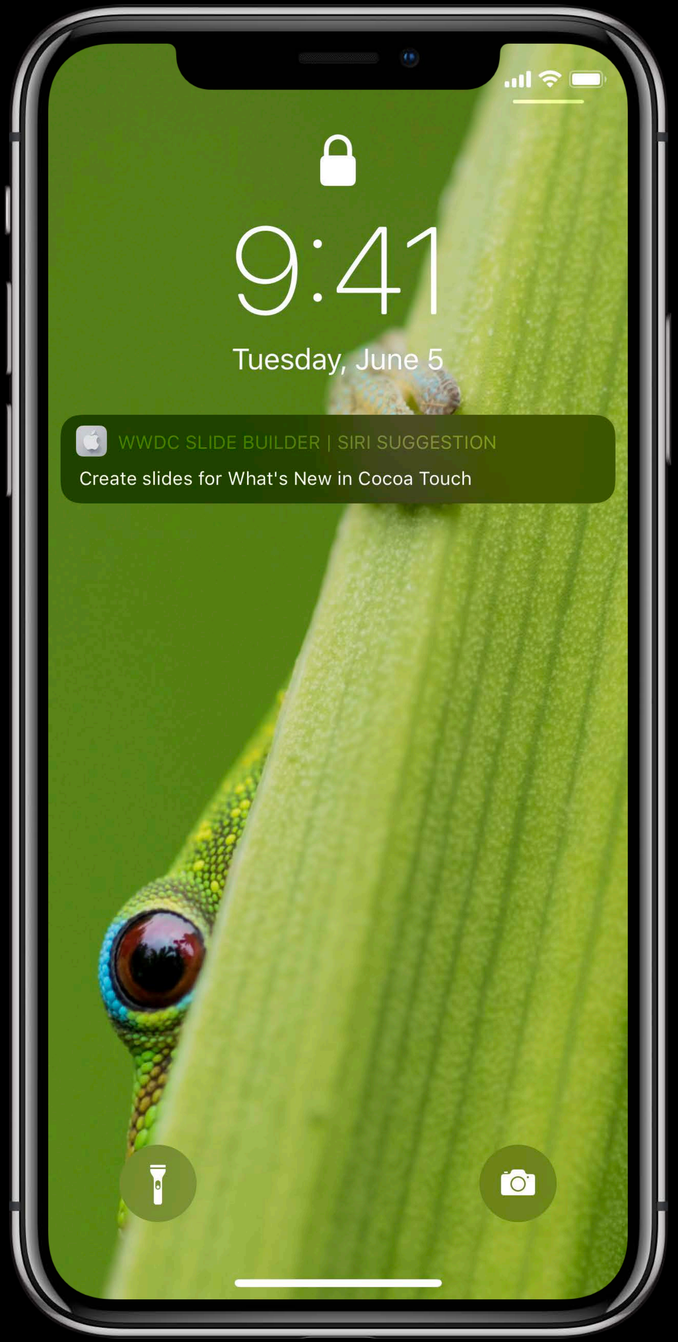

9:41
Edit
safeAreaInsets.top
Mailboxes
Inbox
VIP
Unread
Attachments
Today
Trash
Updated Just Now
safeAreaInsets.bottom


9:41


9:41

UIKit: Apps for Every Size and Shape Executive Ballroom Friday 3:20PM

Siri Shortcuts


9:41
Tuesday, June 5
WWDC SLIDE BUILDER | SIRI SUGGESTION
Create slides for What's New in Cocoa Touch


TUE 5
10:09
WWDC Slide Bu...
Create WWDC Slides
for What's New in C...
8:24PM
Sunset
10H 44M


9:41
Siri & Search
Shortcuts
Search
New Note
MESSAGES
See All
Send a Message to Megan
Send a Message to David
Send a Message to Eliza
NEWS
Today
PHONE
See All
Call Megan
mobile
Call Susan
mobile
Call David
mobile

Siri Shortcuts

NSUserActivity

Intents

- SiriKit
- Custom

Siri Shortcuts

NSUserActivity

Common API for Handoff and Spotlight

Set `eligibleForPrediction = true`

Siri Shortcuts

SiriKit Intents

Siri Shortcuts

SiriKit Intents

Car Commands

VoIP Calling

Photo Search

Notes

Messaging

Visual Codes

Lists

Ride Booking

CarPlay

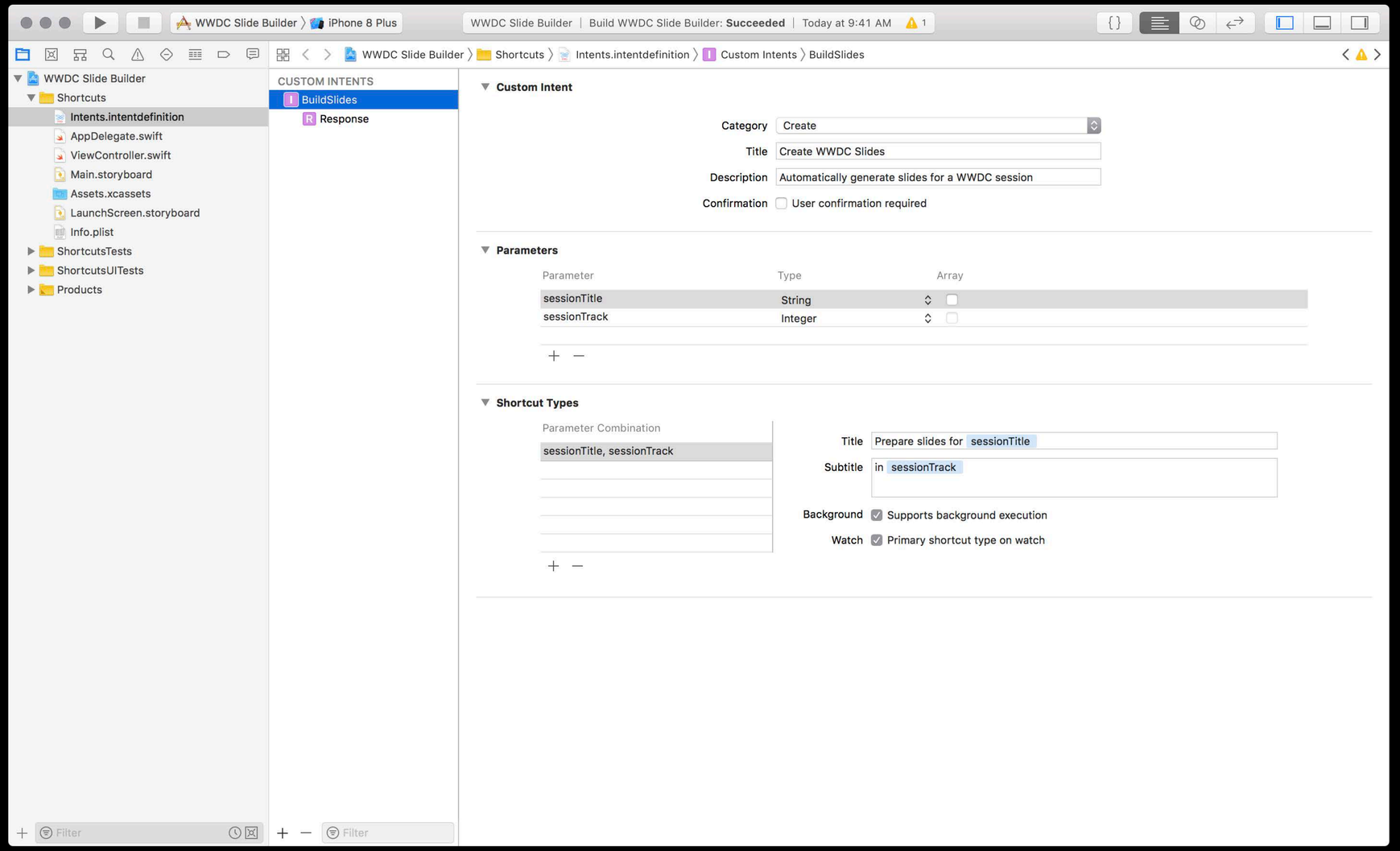
Workouts

Restaurant Reservations

Payments

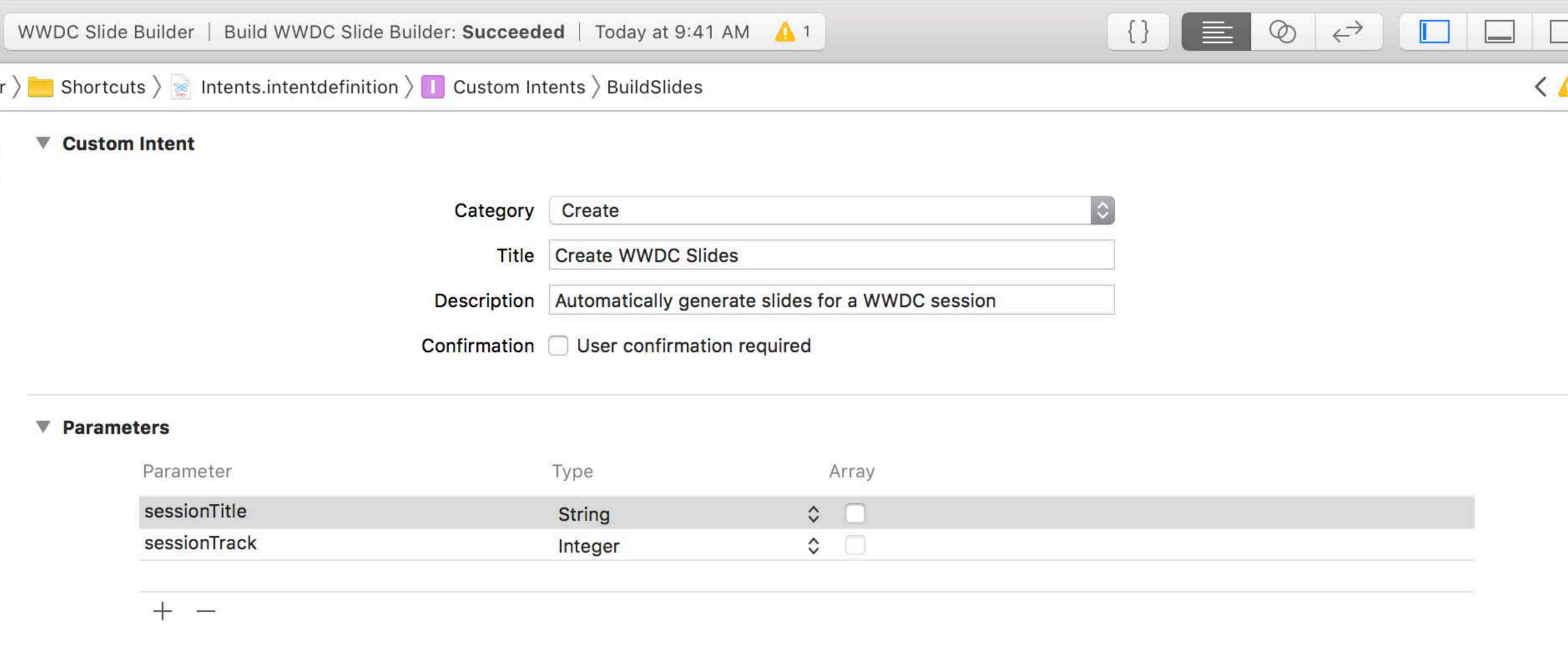
Siri Shortcuts

Custom Intents



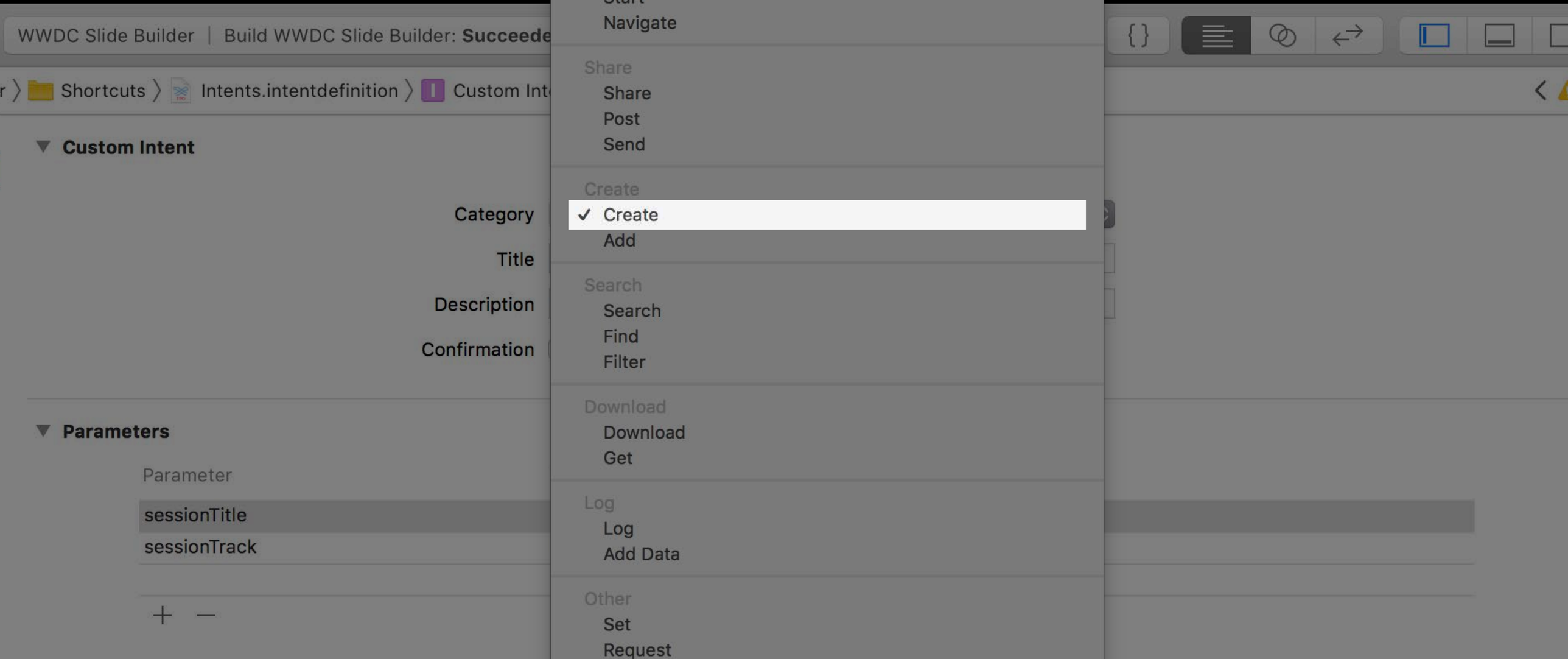
Siri Shortcuts

Custom Intents



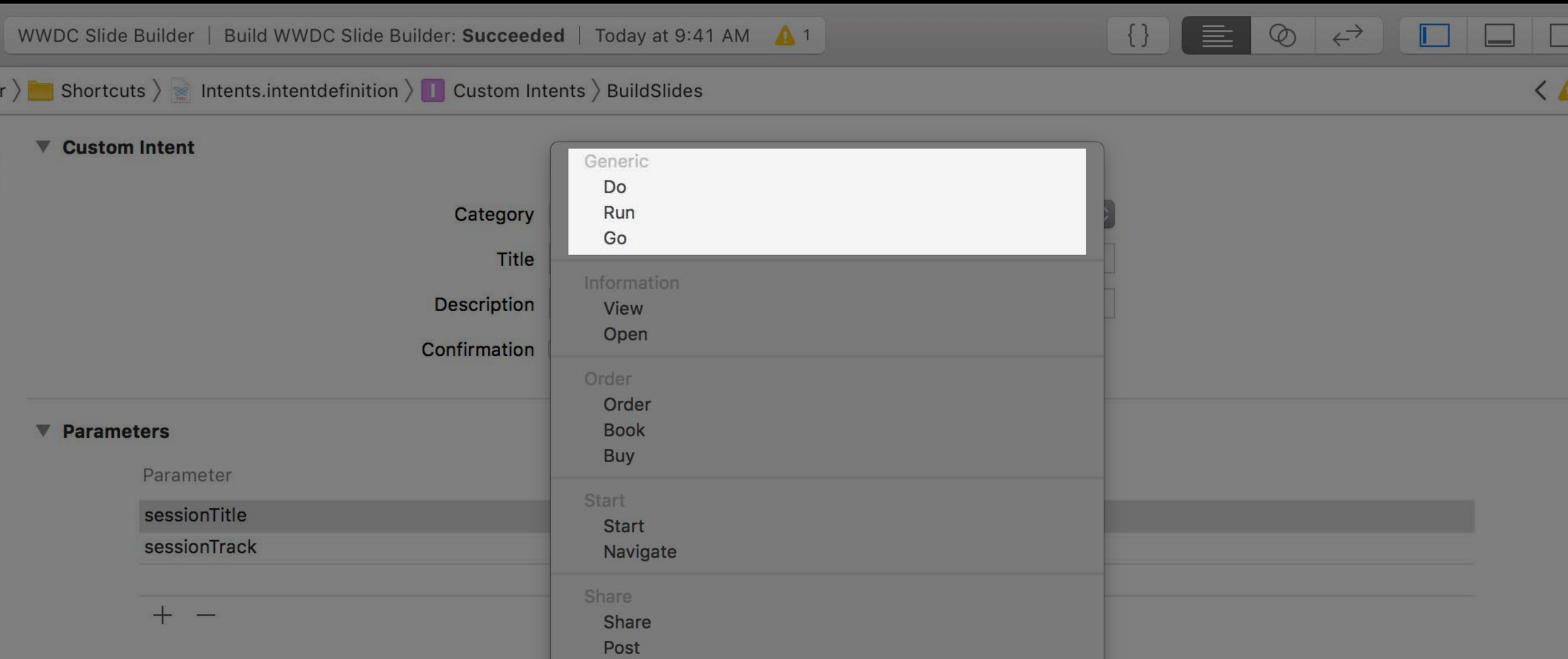
Siri Shortcuts

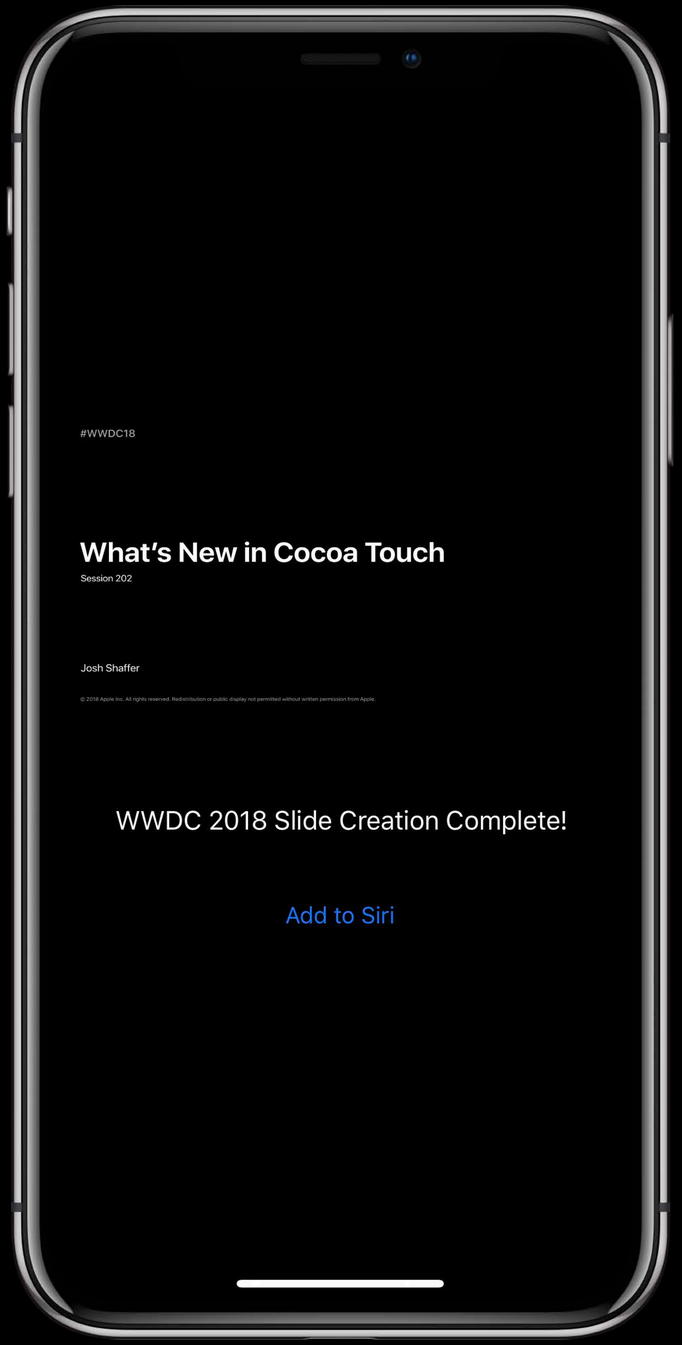
Custom Intents



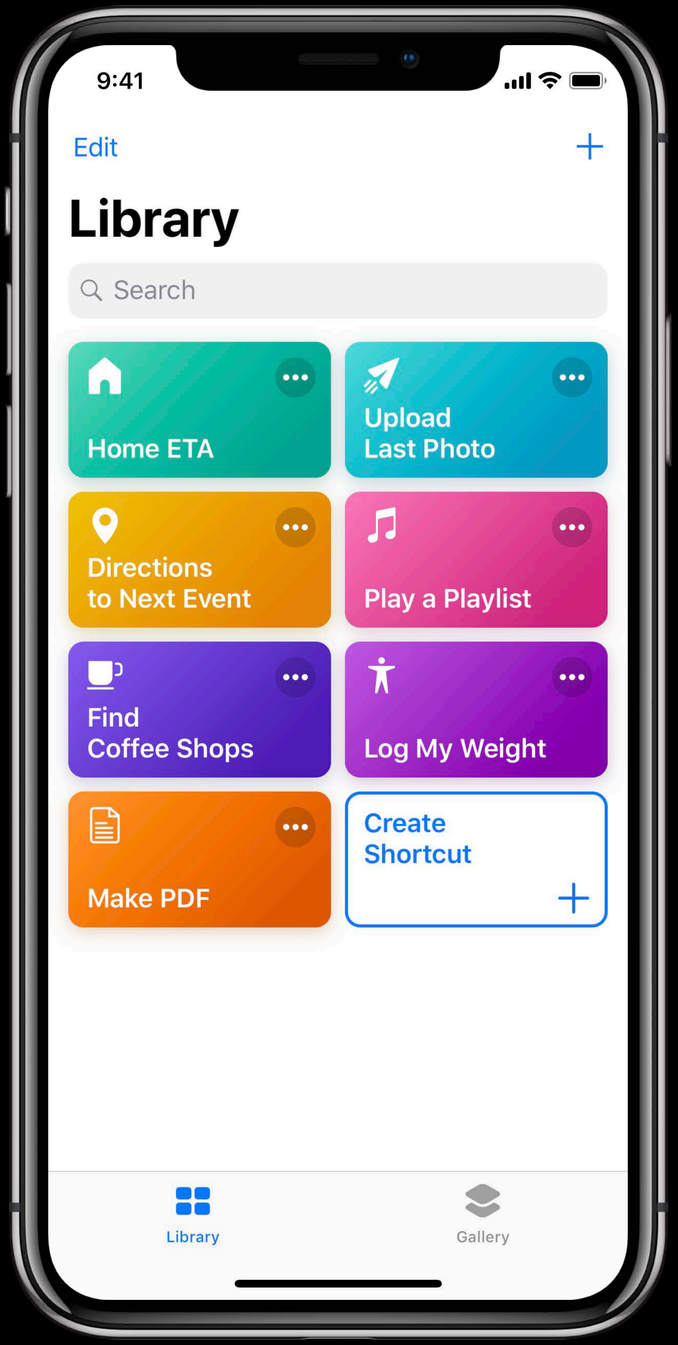
Siri Shortcuts

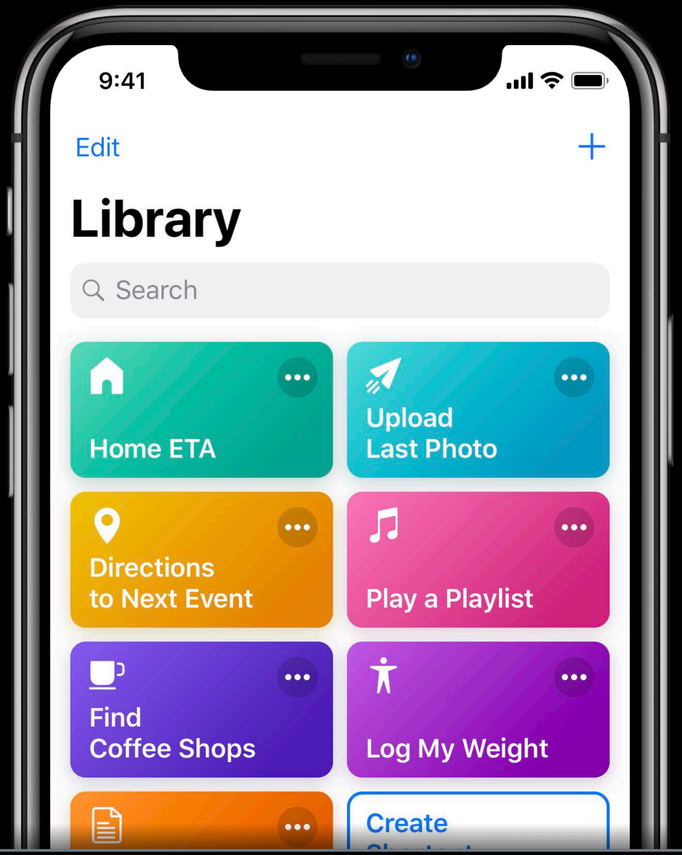
Custom Intents




#WWDC18
What’s New in Cocoa Touch
Session 202
Josh Shaffer
© 2018 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.
WWDC 2018 Slide Creation Complete!
Add to Siri


9:41
Cancel
Add to Siri
Create WWDC Slides
WWDC Slide Builder
You can say something like...
“Create WWDC Slides”
After you record your personalized phrase, Siri can use it to tell “WWDC Slide Builder” to run this shortcut.


9:41
Edit
Library
Search
Home ETA
Upload
Last Photo
Directions
to Next Event
Play a Playlist
Find
Coffee Shops
Log My Weight
Make PDF
Create
Shortcut
Library
Gallery



Introduction to Siri Shortcuts	Hall 2	Tuesday 5:00PM
Building for Voice with Siri Shortcuts	Hall 2	Wednesday 10:00AM
Siri Shortcuts on the Siri Watch Face	Hall 3	Wednesday 11:00AM

More Information

https://developer.apple.com/wwdc18/202

I Have This Idea For an App…	Hall 3	Tuesday 11:00AM
A Tour of UICollectionView	Hall 3	Thursday 2:00PM
Adding Delight to Your iOS App	Hall 3	Friday 2:00PM


WWDC18