SwiftUI On All Devices

Jeff Nadeau, macOS Frameworks
Ada Turner, tvOS Frameworks
Meghna Sapre, watchOS Frameworks
The shortest path to building great apps on every device
The shortest path to building great apps on every device
The shortest path to building great apps on every device
AppKit

UIKit

TVUIKit

UIKit

WatchKit
Designed for Every Platform

Designed to accommodate many UI paradigms

Platform support out of the box

A common toolkit that you can learn once and use anywhere
Common Elements

Basic controls

- iOS: Bluetooth (On)
- macOS: Bluetooth (On)
- tvOS: Bluetooth (On)
- watchOS: Bluetooth (On)
Common Elements
Basic controls

```
Toggle(isOn: $btEnabled) {
    Text("Bluetooth")
}
```
Common Elements

Layout

HStack {
    Text("Label")
    Spacer()

    Button(action: /* */) {
        Text("Delete")
    }

    Button(action: /* */) {
        Text("Cancel")
    }
}
.padding()
**Common Elements**

**Advanced controls**

```swift
Picker(selection: /* */ *, label: /* */ *) {
    ForEach(allFruits) { fruit in
        Text(verbatim: fruit.name)
    }
}
```

<table>
<thead>
<tr>
<th>macOS</th>
<th>Blueberry</th>
</tr>
</thead>
<tbody>
<tr>
<td>iOS</td>
<td>Apple</td>
</tr>
<tr>
<td></td>
<td>Apricot</td>
</tr>
<tr>
<td></td>
<td>Banana</td>
</tr>
<tr>
<td></td>
<td>Blackberry</td>
</tr>
<tr>
<td></td>
<td>Blueberry</td>
</tr>
<tr>
<td>watchOS</td>
<td>Blackberry</td>
</tr>
<tr>
<td></td>
<td>Blueberry</td>
</tr>
<tr>
<td></td>
<td>Dragonfruit</td>
</tr>
</tbody>
</table>
Common Elements
Advanced controls

```swift
Picker(selection: /* */ , label: /* */ ) {
    ForEach(allFruits) { fruit in
        Text(verbatim: fruit.name)
    }
}
```

macOS
Blueberry

iOS
Apple
Apricot
Banana
Blackberry
Blueberry

watchOS
Blackberry
Blueberry
Dragonfruit
Of course not
One Size Does Not Fit All
One Size Does Not Fit All

There’s no such thing as a “one size fits all app”
One Size Does Not Fit All

There’s no such thing as a “one size fits all app”

Make the most of each platform’s strengths
One Size Does Not Fit All

There’s no such thing as a “one size fits all app”
Make the most of each platform’s strengths
Share skillset and toolset
One Size Does Not Fit All

There’s no such thing as a “one size fits all app”

Make the most of each platform’s strengths

Share skillset and toolset

Share code where it makes sense
Write once, run anywhere
Write once, run anywhere
once, anywhere
Learn once, apply anywhere
We built an app
We built four apps
Landmarks
Landmarks

Research and visit landmarks
Landmarks

Research and visit landmarks

Look at photos and visitor info
Landmarks

Research and visit landmarks
Look at photos and visitor info
Get maps and directions
Landmarks

Research and visit landmarks
Look at photos and visitor info
Get maps and directions
Mark favorites
Landmarks for Apple TV
Landmarks for Apple TV

Couch surfing for places to go
Landmarks for Apple TV

Couch surfing for places to go

Together with family or friends
Landmarks for Apple TV

Couch surfing for places to go
Together with family or friends
Mark favorites to research later
Landmarks for Mac
Landmarks for Mac

Research many options in detail
Landmarks for Mac

Research many options in detail

Compare information
Landmarks for Mac

Research many options in detail

Compare information

Sort and filter
Landmarks for Mac

Research many options in detail

Compare information

Sort and filter

Read up on the details
Landmarks for iOS
Landmarks for iOS

Quick information about each landmark
Landmarks for iOS

Quick information about each landmark

Get driving directions
Landmarks for iOS

Quick information about each landmark
Get driving directions
Place a phone call
Landmarks for Apple Watch
Landmarks for Apple Watch

Get at-a-glance info about the current location
Landmarks for Apple Watch

- Get at-a-glance info about the current location
- Receive important notifications
Learning Experience
Learning Experience

New documentation series
Learning Experience

New documentation series

Build the app step-by-step
Learning Experience

New documentation series

Build the app step-by-step

Available for download today
SwiftUI on Apple TV

Ada Turner, tvOS Frameworks
The biggest, boldest screen
The Biggest, Boldest Screen
The Biggest, Boldest Screen

10-foot experience
The Biggest, Boldest Screen

10-foot experience

Focus and the Siri Remote
The Biggest, Boldest Screen

10-foot experience

Focus and the Siri Remote

Streamlined navigation
What is a 10-foot experience?
10-Foot Experience
10-Foot Experience

Large screen
10-Foot Experience

Large screen

Long viewing distance
10-Foot Experience

Large screen

Long viewing distance

Extended periods of use
10-Foot Experience

Large screen
Long viewing distance
Extended periods of use
Multiple viewers at the same time
Landmarks — What Makes the Cut?
Landmarks — What Makes the Cut?

Beautiful photos of landmarks
Landmarks — What Makes the Cut?

- Beautiful photos of landmarks
- Adding favorites
Landmarks — What Makes the Cut?

- Beautiful photos of landmarks
- Adding favorites
- Basic tourism information
Landmarks — What Makes the Cut?

- Beautiful photos of landmarks
- Adding favorites
- Basic tourism information

- Lengthy historical information
Landmarks — What Makes the Cut?

- Beautiful photos of landmarks
- Adding favorites
- Basic tourism information

- Lengthy historical information
- Advanced sorting and filtering
Landmarks — What Makes the Cut?

- Beautiful photos of landmarks
- Adding favorites
- Basic tourism information

- Lengthy historical information
- Advanced sorting and filtering
- Location-based notifications
Focus and the Siri Remote
Focus and the Siri Remote
Focus and the Siri Remote

Entire interface must support focus
Focus and the Siri Remote

Entire interface must support focus

SwiftUI supports focus by default
// Custom Focusable Views

struct MyFocusableView: View {
    let canBecomeFocused: Bool
    var body: some View {
        return Text("Hello WWDC")
            .focusable(canBecomeFocused) { isFocused in
                // Focus changed
            }
            .onPlayPauseCommand {
                // Play/pause button pressed
            }
            .onExitCommand {
                // Menu button pressed
            }
    }
}
// Custom Focusable Views

struct MyFocusableView: View {

    let canBecomeFocused: Bool

    var body: some View {
        return Text("Hello WWDC")
            .focusable(canBecomeFocused) { isFocused in
            // Focus changed
            }
            .onPlayPauseCommand {
            // Play/pause button pressed
            }
            .onExitCommand {
            // Menu button pressed
            }
    }
}
// Custom Focusable Views

struct MyFocusableView: View {
    let canBecomeFocused: Bool

    var body: some View {
        return Text("Hello WWDC")
            .focusable(canBecomeFocused) { isFocused in
                // Focus changed
            }
            .onPlayPauseCommand {
                // Play/pause button pressed
            }
            .onExitCommand {
                // Menu button pressed
            }
    }
}
// Custom Focusable Views

struct MyFocusableView: View {
    let canBecomeFocused: Bool
    var body: some View {
        return Text("Hello WWDC")
            .focusable(canBecomeFocused) { isFocused in
                // Focus changed
            }
            .onPlayPauseCommand {
                // Play/pause button pressed
            }
            .onExitCommand {
                // Menu button pressed
            }
    }
}
Streamlined navigation
// TabbedView and NavigationView on tvOS

struct MainView: View {
    enum Tab { case explore, hikes, tours }
    @State var selection: Tab
    var body: some View {
        return NavigationView {
            TabbedView(selection: $selection) {
                ExploreView().tabItemLabel(Text("Explore")) .tag(Tab.explore)
                HikesView().tabItemLabel(Text("Hikes")) .tag(Tab.hikes)
                ToursView().tabItemLabel(Text("Tours")) .tag(Tab.tours)
            }
        }
    }
}
struct MainView: View {
    enum Tab { case explore, hikes, tours }
    @State var selection: Tab
    var body: some View {
        return NavigationView {
            TabbedView(selection: $selection) {
                ExploreView().tabItemLabel(Text("Explore")).tag(Tab.explore)
                HikesView().tabItemLabel(Text("Hikes")).tag(Tab.hikes)
                ToursView().tabItemLabel(Text("Tours")).tag(Tab.tours)
            }
        }
    }
}

// TabbedView and NavigationView on tvOS
// TabbedView and NavigationView on tvOS

struct MainView: View {
    enum Tab { case explore, hikes, tours }
    @State var selection: Tab

    var body: some View {
        return NavigationView {
            TabbedView(selection: $selection) {
                ExploreView().tabItemLabel(Text("Explore")).tag(Tab.explore)
                HikesView().tabItemLabel(Text("Hikes")).tag(Tab.hikes)
                ToursView().tabItemLabel(Text("Tours")).tag(Tab.tours)
            }
        }
    }
}
// TabbedView and NavigationView on tvOS

struct MainView: View {

    enum Tab { case explore, hikes, tours }

    @State var selection: Tab

    var body: some View {
        return NavigationView {
            TabbedView(selection: $selection) {
                ExploreView().tabItemLabel(Text("Explore")).tag(Tab.explore)
                HikesView().tabItemLabel(Text("Hikes")).tag(Tab.hikes)
                ToursView().tabItemLabel(Text("Tours")).tag(Tab.tours)
            }
        }
    }
}
tvOS vs. iOS Navigation Hierarchies
tvOS vs. iOS Navigation Hierarchies

iOS

TabbedView
tvOS vs. iOS Navigation Hierarchies

iOS

- TabbedView
  - NavigationView
  - Content
  - NavigationView
tvOS vs. iOS Navigation Hierarchies

iOS

TabbedView

NavigationView

Content

NavigationView

Content
tvOS vs. iOS Navigation Hierarchies

iOS

TabbedView

NavigationView

Content

NavigationView

Content

Content

Content

Content
tvOS vs. iOS Navigation Hierarchies
tvOS vs. iOS Navigation Hierarchies

iOS

- TabbedView
  - NavigationView
    - Content
      - Content
    - Content
  - Content
- NavigationView
  - Content
    - Content

tvOS

- NavigationView
tvOS vs. iOS Navigation Hierarchies

iOS
- TabbedView
  - NavigationView
    - Content
      - Content
        - Content
  - Content
    - Content
      - Content

tvOS
- NavigationView
  - TabbedView
    - Content
    - Content
    - Content
tvOS vs. iOS Navigation Hierarchies

iOS
- TabbedView
  - NavigationView
    - Content
    - Content
  - Content

tvOS
- NavigationView
  - TabbedView
    - Content
    - Content
    - Content
    - Content
tvOS vs. iOS Navigation Hierarchies
tvOS vs. iOS Navigation Hierarchies

iOS

```swift
TabbedView(selection: $selection) {
    NavigationView {
        ExploreView()
    }
    .tabItemLabel(Text("Explore"))
    .tag(Tab.explore)
    NavigationView {
        HikesView()
    }
    .tabItemLabel(Text("Hikes"))
    .tag(Tab.hikes)
    NavigationView {
        ToursView()
    }
    .tabItemLabel(Text("Tours"))
    .tag(Tab.tours)
}
```

tvOS

```swift
TabbedView(selection: $selection) {
    ExploreView()
    .tabItemLabel(Text("Explore"))
    .tag(Tab.explore)
    HikesView()
    .tabItemLabel(Text("Hikes"))
    .tag(Tab.hikes)
    ToursView()
    .tabItemLabel(Text("Tours"))
    .tag(Tab.tours)
}
```
tvOS vs. iOS Navigation Hierarchies

iOS

TabbedView(selection: $selection) {
    NavigationView
        NavigationView
            NavigationView

}

tvOS

NavigationView {
    TabbedView(selection: $selection) {
    }
}
tvOS vs. iOS Navigation Hierarchies

**iOS**

```swift
TabbedView(selection: $selection) {
    NavigationView {
        ExploreView()
    } .tabItemLabel(Text("Explore")) .tag(Tab.explore)
    NavigationView { HikesView() }
        .tabItemLabel(Text("Hikes")) .tag(Tab.hikes)
    NavigationView { ToursView() }
        .tabItemLabel(Text("Tours")) .tag(Tab.tours)
}
```

**tvOS**

```swift
NavigationView {
    TabbedView(selection: $selection) {
        ExploreView()
            .tabItemLabel(Text("Explore")) .tag(Tab.explore)
        HikesView()
            .tabItemLabel(Text("Hikes")) .tag(Tab.hikes)
        ToursView()
            .tabItemLabel(Text("Tours")) .tag(Tab.tours)
    }
}
```
SwiftUI on Mac

Jeff Nadeau, macOS Frameworks
High information density
Multiple windows
Keyboard shortcuts
Touch Bar
High information density
Multiple windows
Keyboard shortcuts
Touch Bar
High Information Density

Provide more information at a glance
High Information Density

Provide more information at a glance

Precision pointing device allows smaller controls
High Information Density

Provide more information at a glance

Precision pointing device allows smaller controls
High Information Density

Provide more information at a glance

Precision pointing device allows smaller controls

Higher tolerance for large amounts of text
High Information Density

Provide more information at a glance

Precision pointing device allows smaller controls

Higher tolerance for large amounts of text
High Information Density
High Information Density

Spacing and padding automatically adapt
High Information Density

Spacing and padding automatically adapt

Use `controlSize()` for compact controls
High information density
Multiple windows
Keyboard shortcuts
Touch Bar
Multiple Windows
Multiple Windows

Compare content across windows side-by-side
Multiple Windows

Compare content across windows side-by-side

Focus on a single item in its own window
Multiple Windows

Compare content across windows side-by-side

Focus on a single item in its own window

Spatially organize windows around the Desktop and Spaces
High information density
Multiple windows
Keyboard shortcuts
Touch Bar
Keyboard Shortcuts
Keyboard Shortcuts

Quick access to common actions
Keyboard Shortcuts

Quick access to common actions

Lightning-fast navigation
// Adding Keyboard Shortcuts

TabbedView(selection: $selectedTab) {
    ExploreView().tag(Tab.explore).tabItemLabel(Text("Explore"))

    HikesView().tag(Tab.hikes).tabItemLabel(Text("Hikes"))

    ToursView().tag(Tab.tours).tabItemLabel(Text("Tours"))
}
Keyboard Shortcuts

- Explore: ⌘1
- Hikes: ⌘2
- Tours: ⌘3
- Show Toolbar: ⌘T
- Customize Toolbar...:
- Show Sidebar: ⌘S
- Enter Full Screen: ⌘F
extension Command {
    static let showExplore = Command(Selector("showExplore:"))
    static let showHikes = Command(Selector("showHikes:"))
    static let showTours = Command(Selector("showTours:"))
}
// Adding Keyboard Shortcuts

TabbedView(selection: $selectedTab) {
    ExploreView().tag(Tab.explore).tabItemLabel(Text("Explore"))

    HikesView().tag(Tab.hikes).tabItemLabel(Text("Hikes"))

    ToursView().tag(Tab.tours).tabItemLabel(Text("Tours"))
}
// Adding Keyboard Shortcuts

TabbedView(selection: $selectedTab) {
    ExploreView().tag(Tab.explore).tabItemLabel(Text("Explore"))

    HikesView().tag(Tab.hikes).tabItemLabel(Text("Hikes"))

    ToursView().tag(Tab.tours).tabItemLabel(Text("Tours"))
}

.onCommand(.showExplore) { self.selectedTab = .explore }
.onCommand(.showHikes) { self.selectedTab = .hikes }
.onCommand(.showTours) { self.selectedTab = .tours }
// Adding Keyboard Shortcuts

TabbedView(selection: $selectedTab) {
    ExploreView().tag(Tab.explore).tabItemLabel(Text("Explore"))
    HikesView().tag(Tab.hikes).tabItemLabel(Text("Hikes"))
    ToursView().tag(Tab.tours).tabItemLabel(Text("Tours"))
}
.onCommand(.showExplore) { self.selectedTab = .explore }
.onCommand(.showHikes) { self.selectedTab = .hikes }
.onCommand(.showTours) { self.selectedTab = .tours }
// Adding Keyboard Shortcuts

TabbedView(selection: $selectedTab) {
    ExploreView().tag(Tab.explore).tabItemLabel(Text("Explore"))

    HikesView().tag(Tab.hikes).tabItemLabel(Text("Hikes"))

    ToursView().tag(Tab.tours).tabItemLabel(Text("Tours"))
}
High information density
Multiple windows
Keyboard shortcuts
Touch Bar
// Adding Touch Bar Support

Touchable { }
// Adding Touch Bar Support

TouchBar {
    Button(action: toggleFavorite) {
        Image(isFavorite ? "favorite" : "favorite-empty")
    }
}
// Adding Touch Bar Support

TouchBar {
    Button(action: toggleFavorite) {
        Image(isFavorite ? "favorite" : "favorite-empty")
    }

    Button(action: placeCall) {
        Image("call")
    }
}

}
// Adding Touch Bar Support

TouchBar {
    Button(action: toggleFavorite) {
        Image(isFavorite ? "favorite" : "favorite-empty")
    }

    Button(action: placeCall) {
        Image("call")
    }

    Button(action: getDirections) {
        Text("Get Directions")
    }
}

// Adding Touch Bar Support

MyCustomView().touchBar(

    TouchBar {
        Button(action: toggleFavorite) {
            Image(isFavorite ? "favorite" : "favorite-empty")
        }

        Button(action: placeCall) {
            Image("call")
        }

        Button(action: getDirections) {
            Text("Get Directions")
        }
    }

)
// Adding Touch Bar Support

MyCustomView().touchBar{
    TouchBar {
        Button(action: toggleFavorite) {
            Image(isFavorite ? "favorite" : "favorite-empty")
        }
        Button(action: placeCall) {
            Image("call")
        }
        Button(action: getDirections) {
            Text("Get Directions")
        }
    }
}
SwiftUI on Apple Watch

Meghna Sapre, watchOS Frameworks
watchOS App

Interactive Notifications

Landmarks on Apple Watch
3
Taps or fewer
watchOS App

Your tour of Turtle Rock

Is confirmed for:
10:00 AM tomorrow

Change of Plans?
Just undate here!
watchOS App

Your tour of Turtle Rock
Is confirmed for:
10:00 AM tomorrow
Change of Plans?
Just update here!
watchOS App
watchOS App

.digitalCrownRotation
watchOS App
watchOS App
watchOS App

HStack
VStack
watchOS App
watchOS App

List Section

Landmarks
Rivers
Charley Rivers
Chincoteague
Lakes
Silver Salmon Creek
watchOS App
Interactive Notifications
Landmarks on Apple Watch
Interactive Notifications
Interactive Notifications

Provide useful information
Interactive Notifications

Provide useful information

Offer intuitive actions
Interactive Notifications

Provide useful information

Offer intuitive actions

Deliver at the right time
watchOS App
Interactive Notifications
Landmarks on Apple Watch
Landmarks on Apple Watch

Turtle Rock
7:00 AM - 7:00 PM

Tour: 10:00 AM
Landmarks on Apple Watch
Landmarks on Apple Watch
Landmarks on Apple Watch
Landmarks on Apple Watch

```swift
ForEach(landmarks) { landmark in
    WatchLandmarkDetails(landmark: landmark)
}
```
Landmarks on Apple Watch

// WatchLandmarkDetails

// Image
LandmarkImage(landmark: landmark)

// Details
NavigationButton(destination: WatchTourDetails(landmark: landmark)) {
    Text("Tour: 10:00 AM")
}

// Make a Call
Button(action: call) {
    Image(systemName: "phone.fill")
        .foregroundColor(.green)
}

// Tour: 10:00 AM

// Landmarks

// Turtle Rock
7:00 AM - 7:00 PM

// Tour: 10:00 AM

// Silver Salmon Creek
Landmarks on Apple Watch
Landmarks on Apple Watch
Landmarks on Apple Watch

```swift
var landmarks: [Landmark] {
    if showFavoritesOnly {
        return landmarkData.filter(isFavorite)
    }
    return landmarkData
}
```
var landmarks: [Landmark] {
    if showFavoritesOnly {
        return landmarkData.filter(isFavorite)
    }
    return landmarkData
}

Button(action: self.toggleFavoritesOnly) {
    Text(self.userData.showFavoritesOnly ?
         "Show All" : "Show Favorites")
}
return landmarkList

Landmarks on Apple Watch
Landmarks on Apple Watch

return landmarkList
Landmarks on Apple Watch

```swift
return landmarkList
  .listStyle(.carousel)
```
Interactive Notifications

- Landmarks
- Turtle Rock
  7:00 AM - 7:00 PM
- Tour: 10:00 AM
Interactive Notifications
Summary

Take a design-first approach
Summary

Take a design-first approach

Effortlessly share model code
Summary

Take a design-first approach

Effortlessly share model code

Be judicious when sharing view code
Summary

Take a design-first approach
Effortlessly share model code
Be judicious when sharing view code
Learn once, apply anywhere
More Information

developer.apple.com/wwdc19/240