

# **Mobile Development on the iOS Platform**

**Master's Degree in Computer Science  
Mobile Programming and Multimedia  
University of Padua**

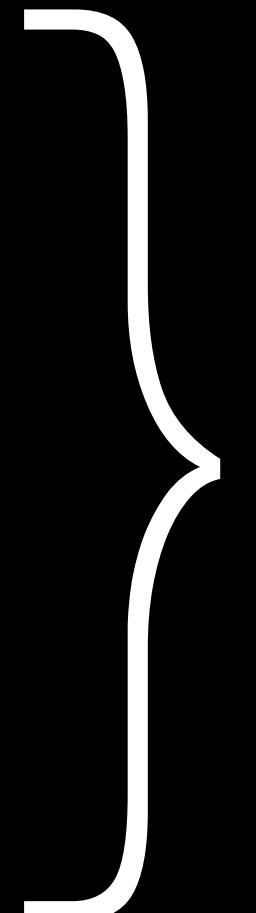
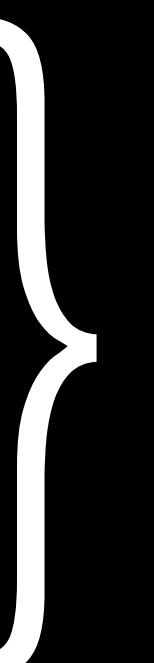
**Alberto De Bortoli - 03/05/2022**

# Who I am

- University of Padua alumni (Master's degree in 2011)
- Worked in the mobile space since then
- Currently working in London at Just Eat Takeaway.com as Principal Software Engineer
- More info at [albertodebortoli.com](http://albertodebortoli.com)



# Today's agenda

- History of native iOS development
  - Languages
  - Frameworks
  - Tools and Package Managers
  - Documentation and Resources
  - Demo
  - Q&A
- 
- 
- ~45 min
- ~45 min

# History of native iOS development

# History of native iOS development

## Operating System and devices

- Drastically evolved over the years
- Started in 2008 with iPhone OS 2
- Very limited set of features
- iPhone OS rebranded as iOS in 2010



# History of native iOS development

## Operating System and devices

- New iPhone models and new iOS major releases every year
- Ever increasing set of features



# History of native iOS development

## Operating System and devices

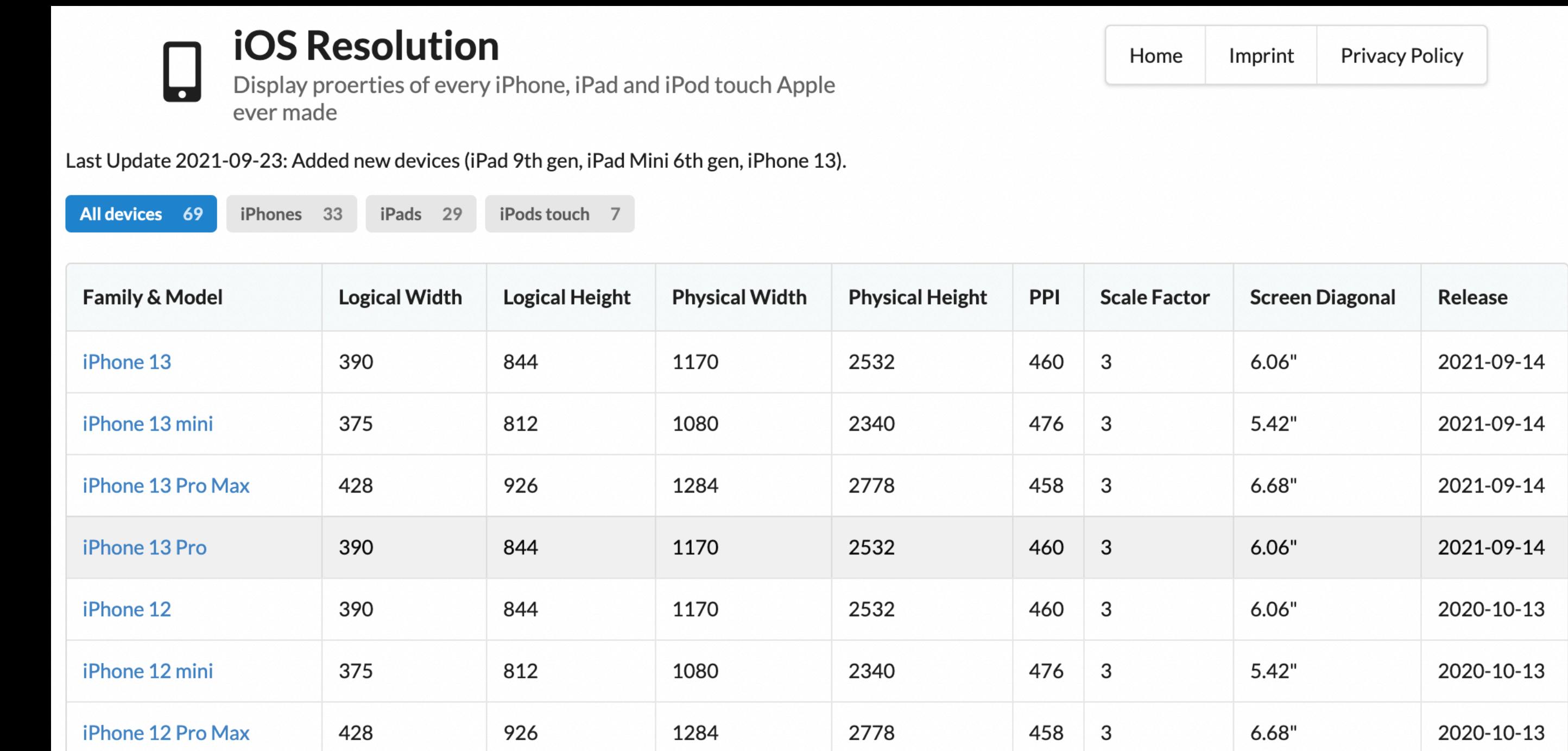
- iPadOS introduced in 2019
- Same as iOS, just a marketing branding choice



# History of native iOS development

## Operating System and devices

- Device fragmentation not particularly relevant anymore
- Developers don't usually care much about specific resolutions, more about layouting rules



The screenshot shows a website titled "iOS Resolution" which displays the properties of every iPhone, iPad, and iPod touch Apple ever made. The page includes a navigation bar with links to Home, Imprint, and Privacy Policy. A message at the top indicates the last update was on 2021-09-23, adding new devices like the iPad 9th gen, iPad Mini 6th gen, and iPhone 13. Below this, there are four tabs: All devices (69), iPhones (33), iPads (29), and iPods touch (7). The main content is a table with the following columns: Family & Model, Logical Width, Logical Height, Physical Width, Physical Height, PPI, Scale Factor, Screen Diagonal, and Release. The table lists several models from the iPhone 13 series down to the iPhone 12 mini, along with their respective dimensions and release dates.

Family & Model	Logical Width	Logical Height	Physical Width	Physical Height	PPI	Scale Factor	Screen Diagonal	Release
iPhone 13	390	844	1170	2532	460	3	6.06"	2021-09-14
iPhone 13 mini	375	812	1080	2340	476	3	5.42"	2021-09-14
iPhone 13 Pro Max	428	926	1284	2778	458	3	6.68"	2021-09-14
iPhone 13 Pro	390	844	1170	2532	460	3	6.06"	2021-09-14
iPhone 12	390	844	1170	2532	460	3	6.06"	2020-10-13
iPhone 12 mini	375	812	1080	2340	476	3	5.42"	2020-10-13
iPhone 12 Pro Max	428	926	1284	2778	458	3	6.68"	2020-10-13

Source: <https://www.ios-resolution.com/>

# History of native iOS development

## Operating System and devices

### Capabilities:

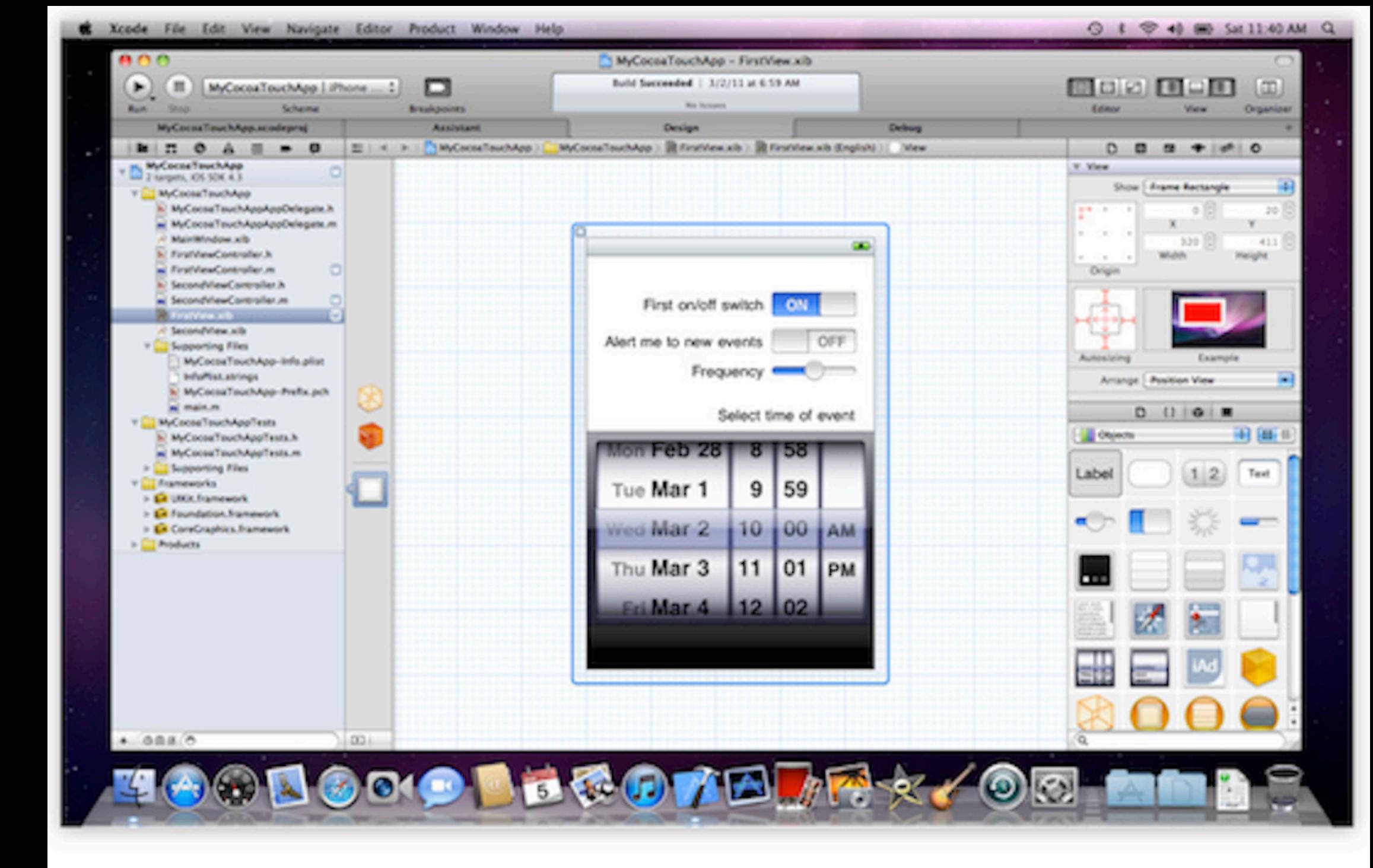
- Ever growing list
- Enabled per app in App Store Connect
- Defined in an Entitlements file (XML) in the app project

Source: <https://appstoreconnect.apple.com/>

Capabilities	App Services
ENABLED	NAME
<input type="checkbox"/>	Access WiFi Information ⓘ
<input type="checkbox"/>	App Attest ⓘ
<input type="checkbox"/>	App Groups ⓘ
<input type="checkbox"/>	Apple Pay Payment Processing ⓘ
<input type="checkbox"/>	Associated Domains ⓘ
<input type="checkbox"/>	AutoFill Credential Provider ⓘ
<input checked="" type="checkbox"/>	ClassKit ⓘ
<input type="checkbox"/>	Communication Notifications ⓘ
<input type="checkbox"/>	Custom Network Protocol ⓘ
<input type="checkbox"/>	Data Protection ⓘ ○ Complete Protection ○ Protected Unless Open ○ Protected Until First User Authentication
<input type="checkbox"/>	Extended Virtual Address Space ⓘ
<input type="checkbox"/>	Family Controls ⓘ
<input type="checkbox"/>	FileProvider TestingMode ⓘ
<input type="checkbox"/>	FONT ⓘ
<input checked="" type="checkbox"/>	Game Center ○ iOS ○ macOS
<input type="checkbox"/>	HealthKit ⓘ
<input type="checkbox"/>	HealthKit Estimate Recalibration ⓘ
<input type="checkbox"/>	HLS Interstitial Previews ⓘ
<input type="checkbox"/>	HomeKit ⓘ
<input type="checkbox"/>	Hotspot ⓘ
<input type="checkbox"/>	iCloud ⓘ ○ Include CloudKit support (requires Xcode 6) ○ Compatible with Xcode 5
<input checked="" type="checkbox"/>	In-App Purchase
<input type="checkbox"/>	Increased Memory Limit ⓘ
<input type="checkbox"/>	Inter-App Audio ⓘ
<input type="checkbox"/>	Low Latency HLS ⓘ
<input type="checkbox"/>	Maps ⓘ
<input type="checkbox"/>	MDM Managed Associated Domains ⓘ
<input type="checkbox"/>	Multipath ⓘ
<input type="checkbox"/>	Network Extensions ⓘ
<input type="checkbox"/>	NFC Tag Reading ⓘ
<input type="checkbox"/>	Personal VPN ⓘ
<input type="checkbox"/>	Push Notifications ⓘ
<input type="checkbox"/>	Sign In with Apple ⓘ

# History of native iOS development IDE

- Xcode as the official IDE
- New major releases almost every year
- Requires Apple's hardware



2010: Xcode 4 was a major turning point in Xcode's history, and the successive improvements were all built on the changes made during this era.

# History of native iOS development IDE

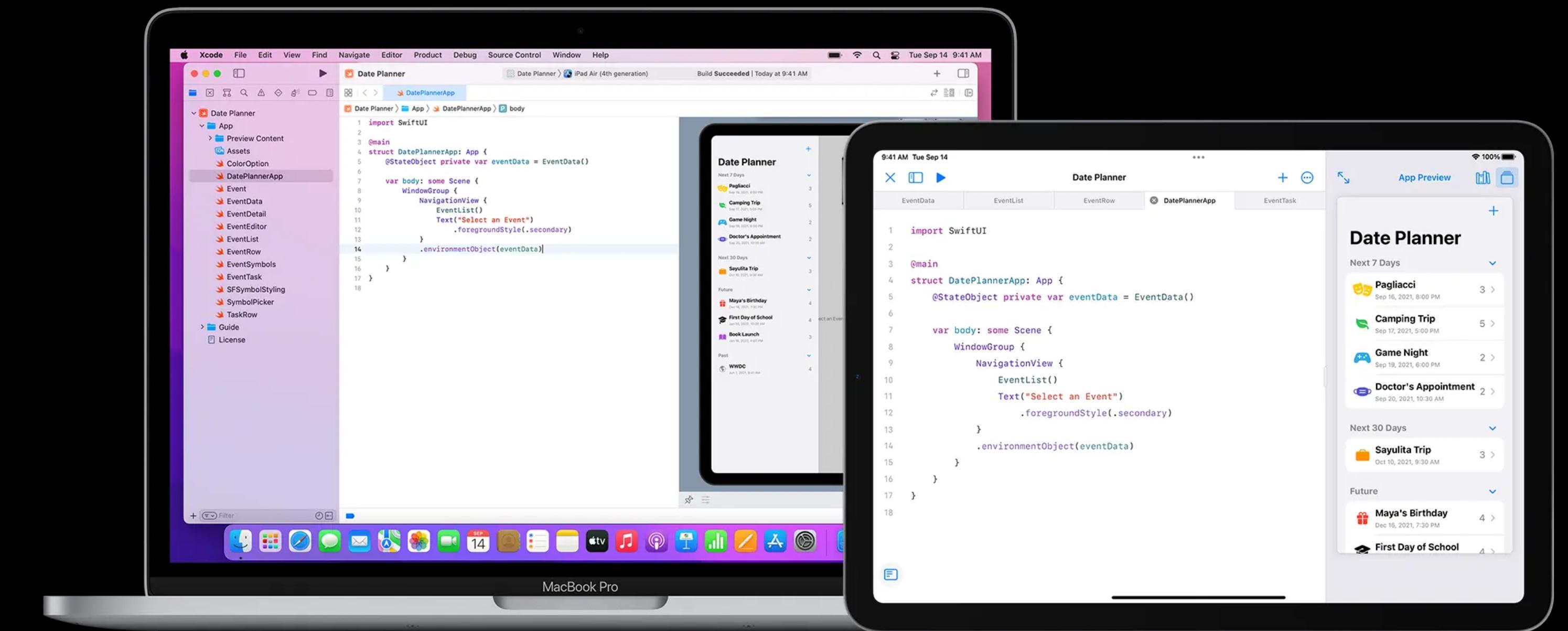
- Xcode as the official IDE
- New major releases almost every year
- Requires Apple's hardware



2019-2021: Xcode 11/12/13 running SwiftUI code with the visual inspector opened for debugging.

# History of native iOS development IDE

- Xcode as the official IDE
- New major releases almost every year
- Requires Apple's hardware



2021 onwards: Swift Playgrounds 4 running on macOS and iPadOS, sharing the same codebase for the same app. Apps built with Swift Playgrounds 4 on iPadOS can be run and delivered to the App Store for publication.

# Languages

# Languages

## Objective-C

- Objective-C as the historical language used at Apple/NeXT
- Written by Brad Cox in the early '80 (been around for ~40 years)
- Very dynamic and allows introspection
- Takes inspiration from Smalltalk (message sending instead of function calls)
- No support for namespaces: classes and symbols prefixed with NS (e.g. NSString)
- Manual memory management (reference counting) in first incarnations

```
NSString *str = [NSString alloc] initWithString: @"Hello";
NSString *combinedStr = [str stringByAppendingString:@" World"];
NSLog(@"Combined string: %@", combinedStr);
```

# Languages

## Swift

- Swift was announced in 2014
- Originally written by Chris Lattner at Apple
- Latest version 5.6 (5.7 in development)
- Strongly typed
- Cutting-edge set of features
- Open-sourced by Apple on GitHub (<https://github.com/apple/swift>)
- Evolution in the open (<https://github.com/apple/swift-evolution>)



```
var str = "Hello"  
str.append(" World")  
print(str)
```

# Frameworks

# Frameworks

## Native Apple frameworks

- Huge list of frameworks from Apple
- Fundamentals:
  - Foundation
  - UIKit
  - SwiftUI

### iOS SDK Frameworks

The following lists the frameworks that are part of the [iOS SDK](#) as of version 13.0.

- ARKit
- AVFoundation.AVFAudio
- AVFoundation
- AVKit
- Accelerate
- Accelerate.vImage
- Accelerate.vecLib
- Accounts
- AdSupport
- AddressBook
- AddressBookUI
- AssetsLibrary
- AudioToolbox
- AudioUnit
- AuthenticationServices
- BackgroundTasks
- BusinessChat
- CFNetwork
- CallKit
- CarPlay
- ClassKit
- CloudKit
- Contacts
- ContactsUI
- CoreAudio
- CoreAudioKit
- CoreAudioTypes
- CoreBluetooth
- CoreData
- CoreFoundation – lower-level C System APIs
- CoreGraphics
- CoreHaptics
- CoreImage
- CoreLocation

# Frameworks

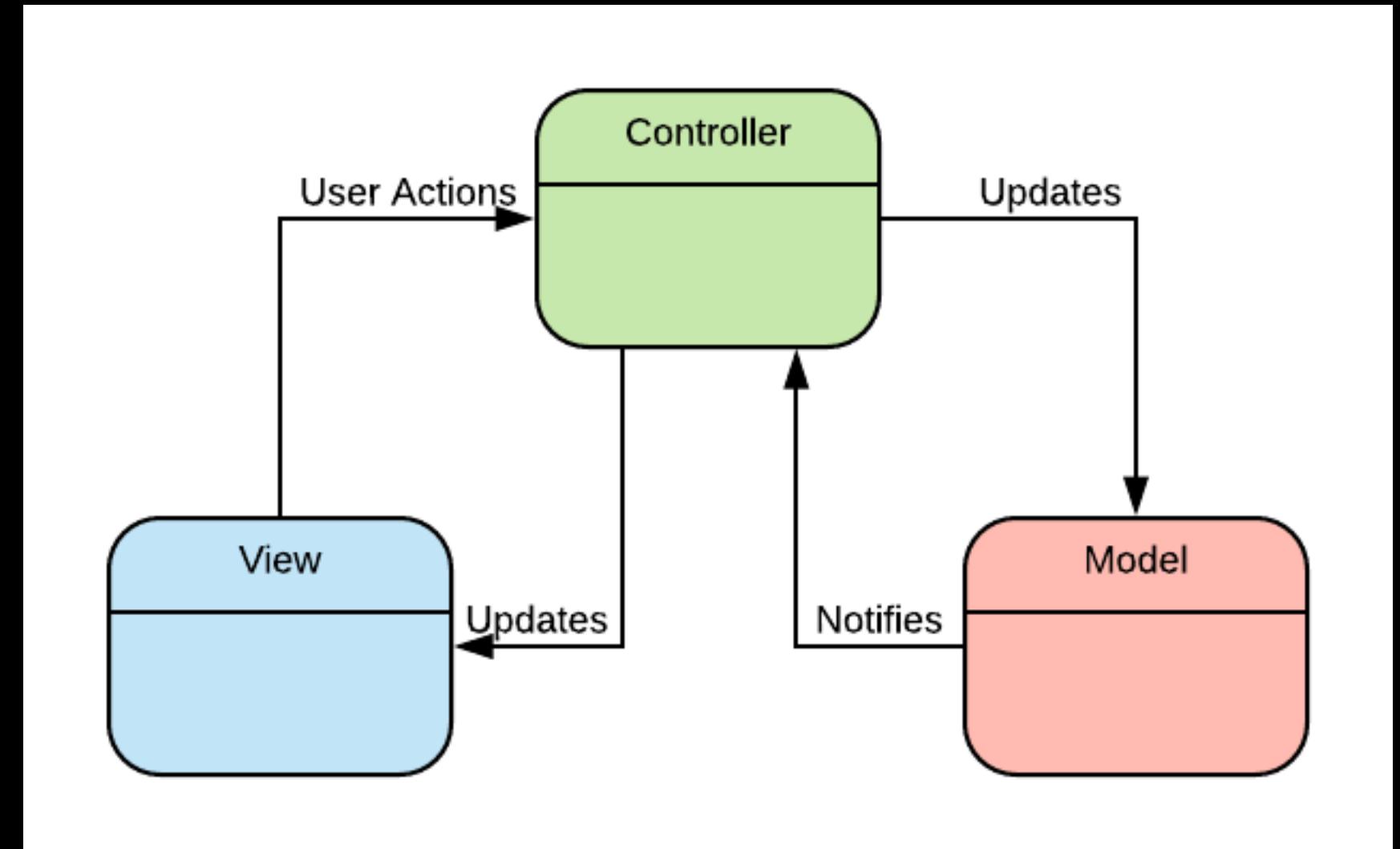
## Foundation and the Swift standard library

- Essential data types, collections, dates, time calculation, text processing, sorting, networking, and operating-system services
- Foundation could be considered the “standard library” of Objective-C (NSArray, NSDictionary, NSString, etc.)
- The Swift standard library contains the fundamental data types
- File system access, calendar and time-zone-aware date handling, localization functionality etc. are not included in the Swift standard library and remain in Foundation.

# Frameworks

## UIKit

- Construct and manage a graphical, event-driven user interface
- Manages life-cycle events and scenes
- Provides classes for UI components as well as utilities to manage interactions/gestures and resources
- MVC and delegation as some of the main patterns (implicitly) promoted by Apple
- Imperative UI framework



# Frameworks

## SwiftUI

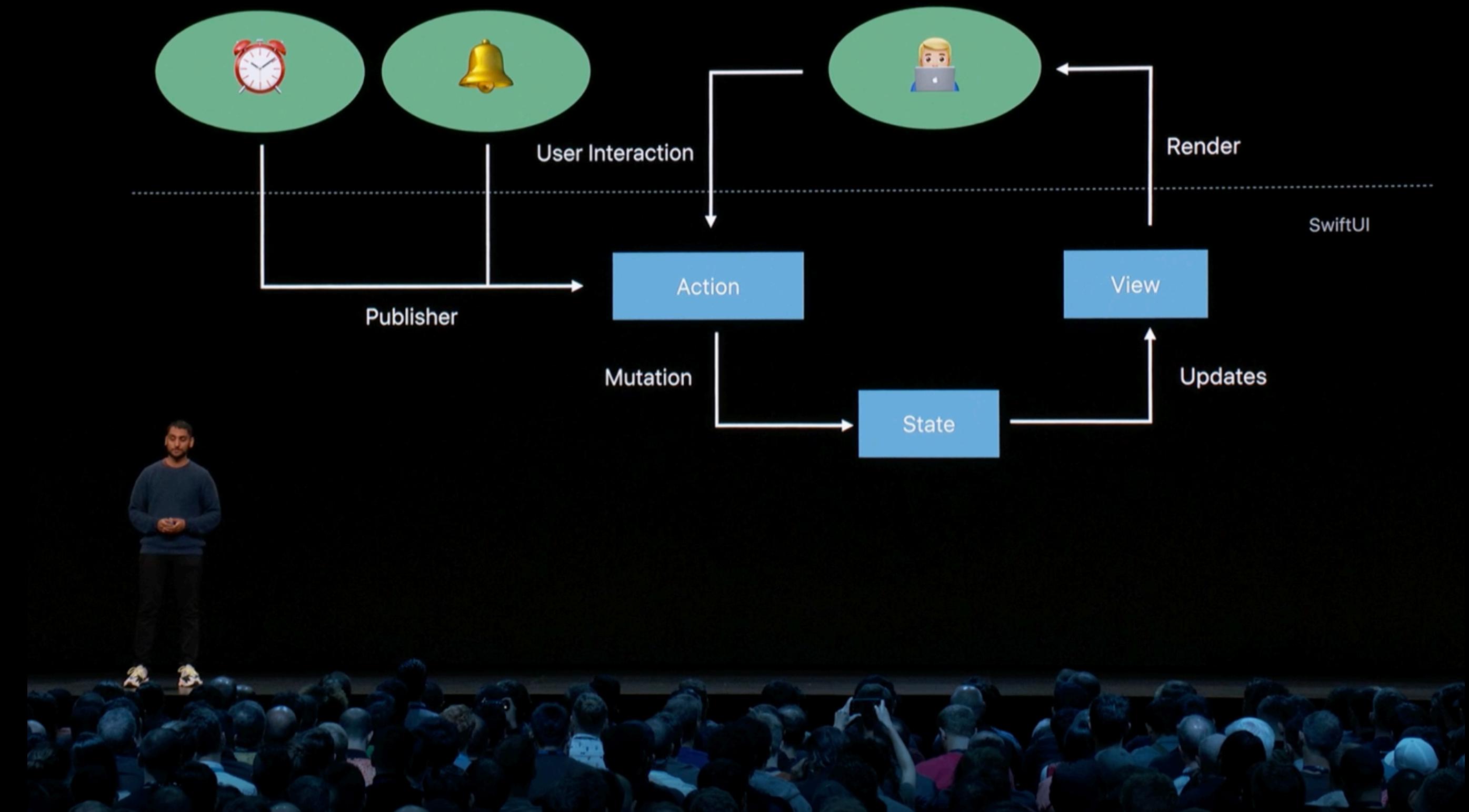
- Massive leap forward
- Swift evolved to enable SwiftUI
- The way forward to write interfaces
- Cross OSes (tvOS, watchOS, macOS, iOS/iPadOS)
- Localisations, dynamic type and accessibility almost for free



# Frameworks

## SwiftUI

- Data Flow Through SwiftUI
- Declarative UI framework



Unnamed pattern (on purpose) but in reality it's a reincarnation of Flux/Redux

Source: <https://developer.apple.com/videos/play/wwdc2019/226/>

# Frameworks

## SwiftUI

### Layouting rules

#### 1. Parent Proposes Size for Child

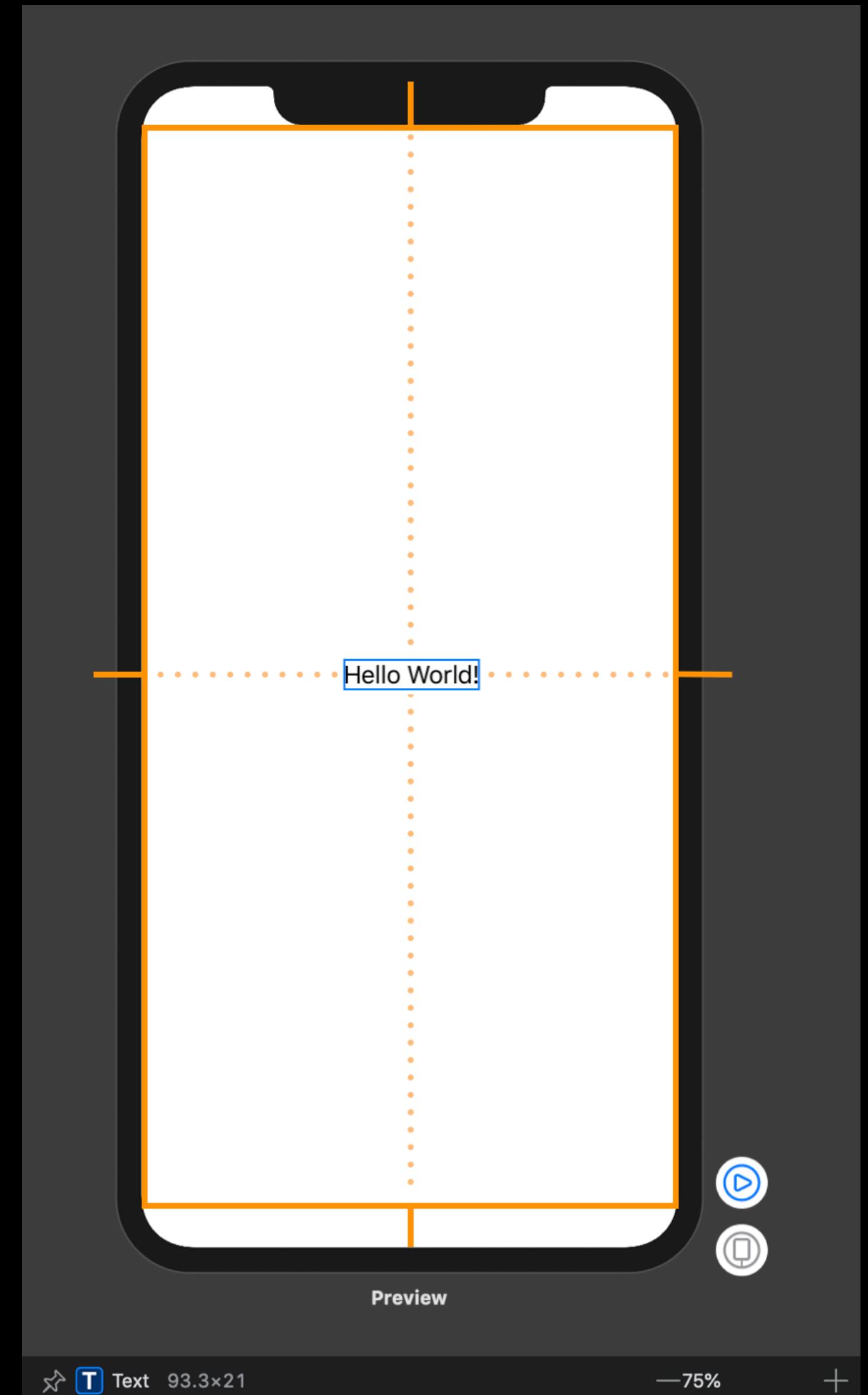
First, the root view offers the text a proposed size – in this case, the entire safe area of the screen, represented by an orange rectangle.

#### 2. Child Chooses its Size

Text only requires that much size to draw its content. The parent has to respect the child's choice. It doesn't stretch or compress the child.

#### 3. Parent Places Child in Parent's Coordinate Space

And now the root view has to put the child somewhere, so it puts it right in the middle.

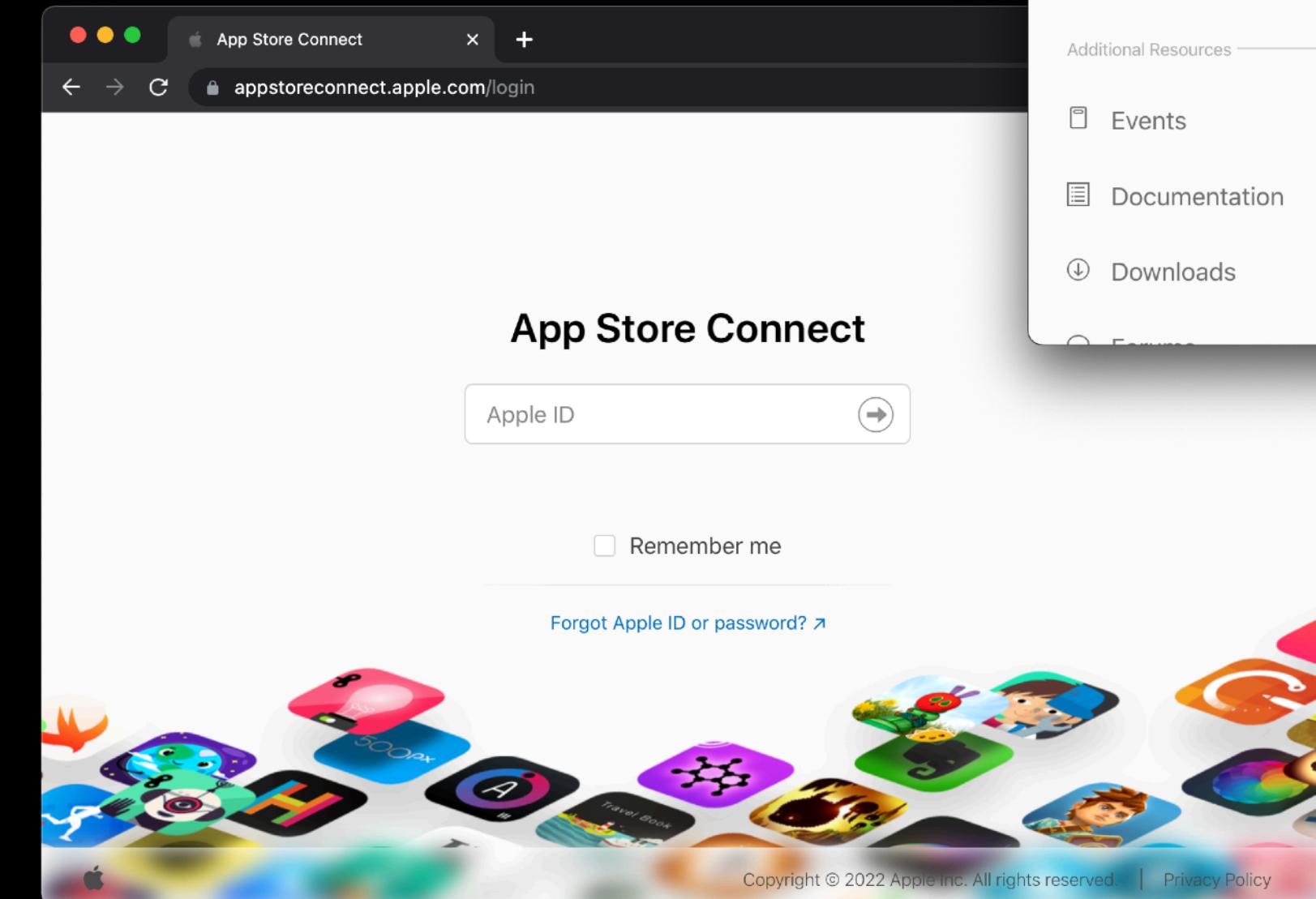


# Tools and Package Managers

# Tools

## App Store Connect and Developer Portal

- The portals to setup and configure all the things for your apps
- Developer License \$99



A screenshot of the Apple Developer account overview page. The top navigation bar shows the Apple logo, the text "Account - Apple Developer", and the URL "developer.apple.com/account/#!/overview". On the right, there's a user profile for "Alberto De Bortoli" and a "Incognito" button. The main content area has a sidebar with links like "Overview", "Membership", "Certificates, IDs &amp; Profiles", "App Store Connect", "CloudKit Console", "Servers", "Code-Level Support", "Events", "Documentation", "Downloads", and "Forums". To the right of the sidebar, there are two main sections: "Certificates, Identifiers &amp; Profiles" (with a gear icon) and "App Store Connect" (with a stylized 'A' icon). Both sections have brief descriptions and links to manage them.

References:

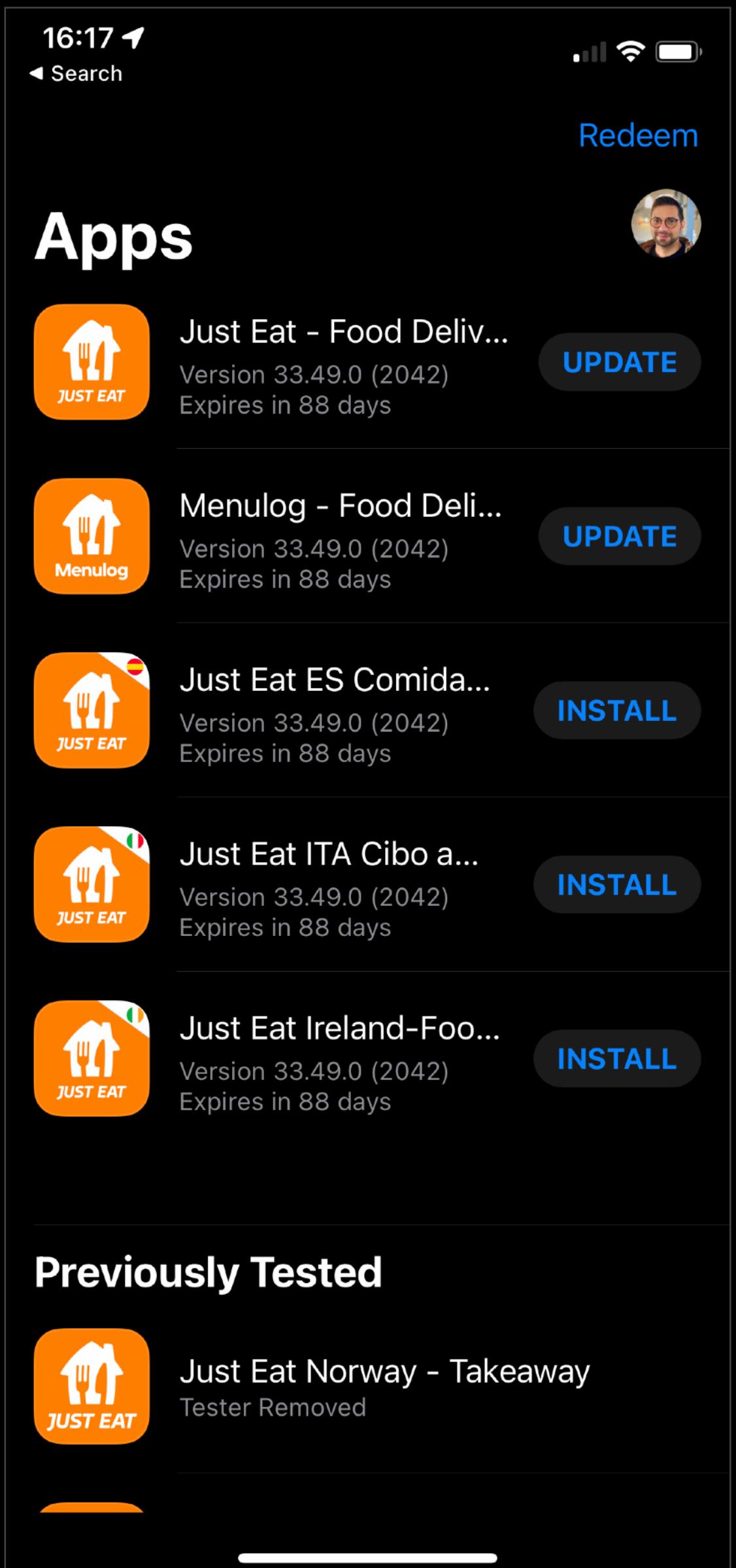
<https://appstoreconnect.apple.com/>  
<https://developer.apple.com/>

# Tools

## TestFlight

- The way to distribute apps to testers using the Apple Developer license
- Dashboard on App Store Connect (per app)
- Dedicated app on the App Store

Reference: <https://developer.apple.com/testflight/>



# Tools

## Fastlane

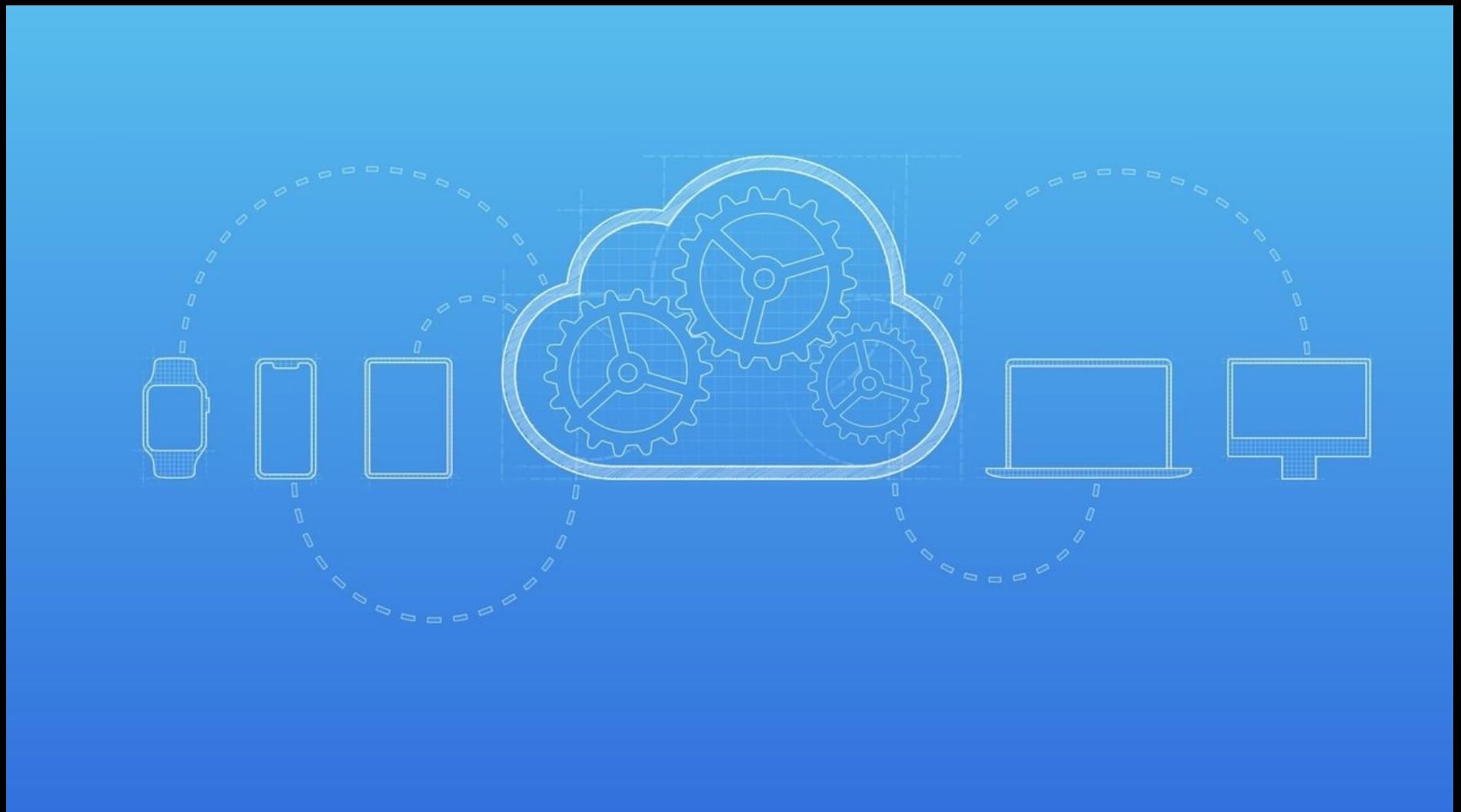


- Fastlane simplifies the tools and processes provided by Apple
  - Automate screenshots and metadata
  - Beta/App Store deployment
  - Code signing, provisioning profiles and private keys management
  - `xcodebuild` commands
- Fastlane is the only third-party tool that became a standard de facto

# Tools

## Xcode Cloud

- Continuous integration and delivery service
- New solution from Apple
- Integrates with the Apple ecosystem
- Meant to be easy
- Announced in 2021, still in beta



# Package Managers

## CocoaPods

〈COCOAPODS〉

- Third-party package manager to overcome the lack from Apple (not the case anymore)
- The historical de facto standard for managing dependencies
- Automated solution
- The frameworks (called pods) are defined in a manifest called “podspec”
- Ruby-based

# Package Managers

## Swift Package Manager



- Long-awaited native dependency manager
- Gradually taking over CocoaPods
- Integrated with Xcode
- The frameworks (called packages) are defined in a manifest named `Package.swift`
- Swift-based

The screenshot shows the Xcode interface for managing Swift packages. On the left, there's a sidebar with 'Recently Used' and a 'Collections' section where 'Apple Swift Packages' is selected. The main area displays the contents of the 'swift-algorithms' package, which includes:

- swift-algorithms**: Commonly used sequence and collection algorithms for Swift.
- swift-argument-parser**: Straightforward, type-safe argument parsing for Swift.
- swift-atomics**: Low-level atomic operations for Swift.
- swift-collections**: Commonly used data structures for Swift.
- swift-crypto**: Implements most of Apple's CryptoKit API for use across multiple platforms.
- swift-nio**: Supports development of asynchronous event-driven network applications that maintain high performance.
- swift-numerics**: Advanced mathematical types and functions for Swift.
- swift-system**: Low-level system calls and types for Swift.
- SwiftProtobuf**: Implements a runtime and plugin for using protobuf serialization technology with Swift.

On the right, detailed information for the **swift-algorithms** package is shown:

- Version**: 1.0.0
- Authors**: natecook1000 and 29 others
- Release Date**: 8 Sep 2021
- License**: Apache v2.0
- Repository**: [github.com/apple/swift-algorithms](https://github.com/apple/swift-algorithms)

Below this, there are sections for **Dependency Rule** (set to "Up to Next Major Version"), **Add to Project** (set to "SwiftUI-Weather"), **Description**, and **Release History**.

### Swift Algorithms

**Swift Algorithms** is an open-source package of sequence and collection algorithms, along with their related types. Read more about the package, and the intent behind it, in the [announcement on swift.org](#).

### Contents

#### Combinations / permutations

- `combinations(ofCount:)` : Combinations of particular sizes of the elements in a collection.
- `permutations(ofCount:)` : Permutations of a particular size of the elements in a collection, or of the full collection.
- `uniquePermutations(ofCount:)` : Permutations of a collection's elements, skipping any duplicate permutations.

#### Mutating algorithms

- `rotate(toStartAt:)`, `rotate(subrange:toStartAt:)` : In-place rotation of elements.
- `stablePartition(by:)`, `stablePartition(subrange:by:)` : A partition that preserves the relative order of the resulting prefix and suffix.

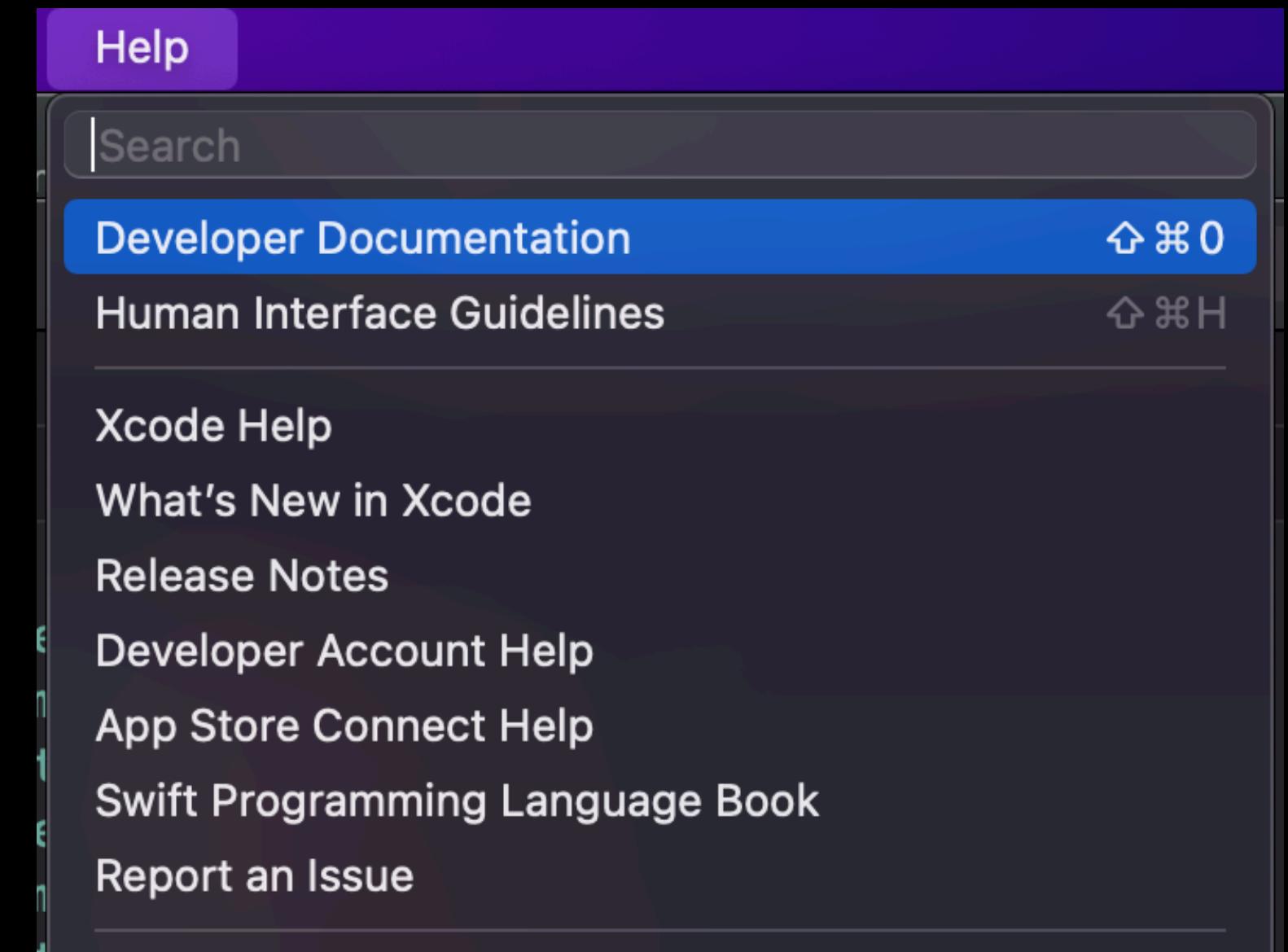
**Add Local...** **Cancel** **Add Package**

# Documentation and Resources

# Documentation and Resources

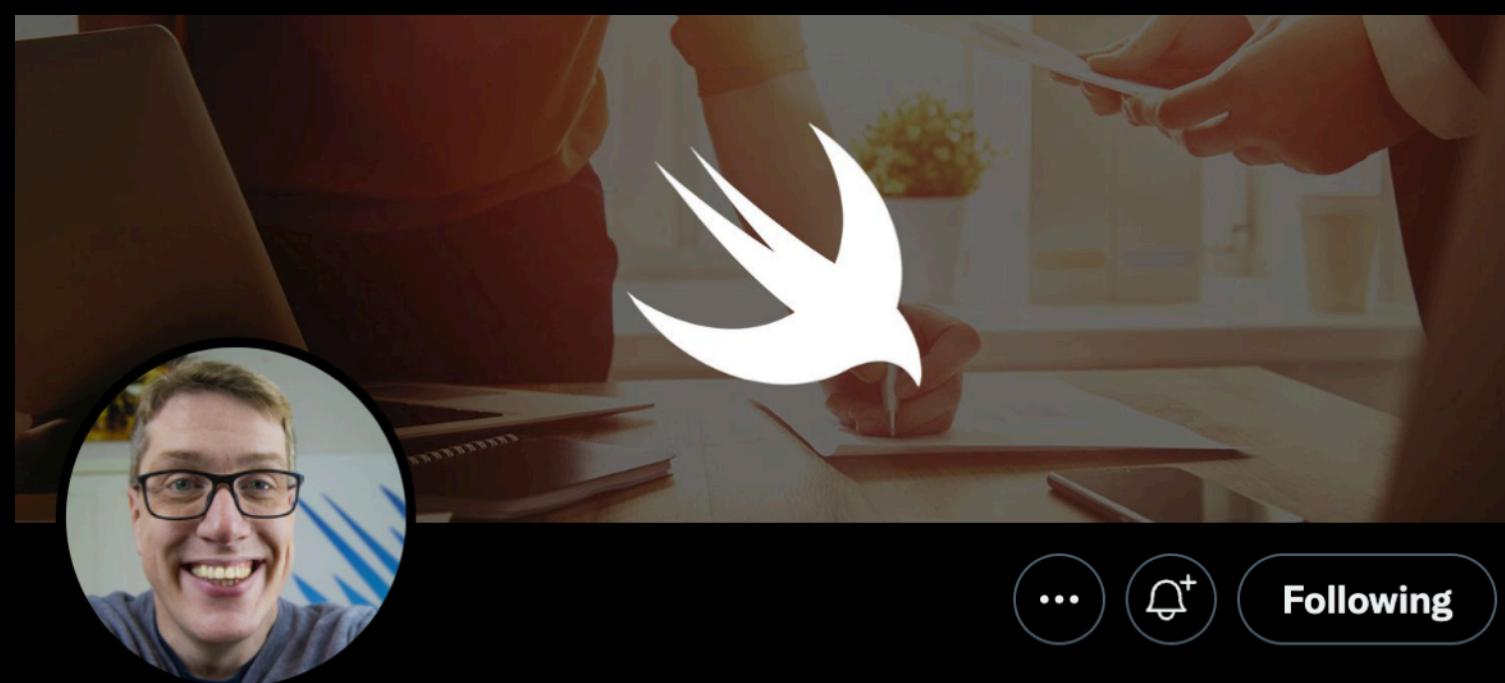
## Official documentation

- Apple has become extremely good in providing documentation, examples and tutorials for all level.
- Main places to go are [swift.org](http://swift.org) and [developer.apple.com](http://developer.apple.com)
- Easily accessible from within Xcode



# Documentation and Resources

## Authors and speakers



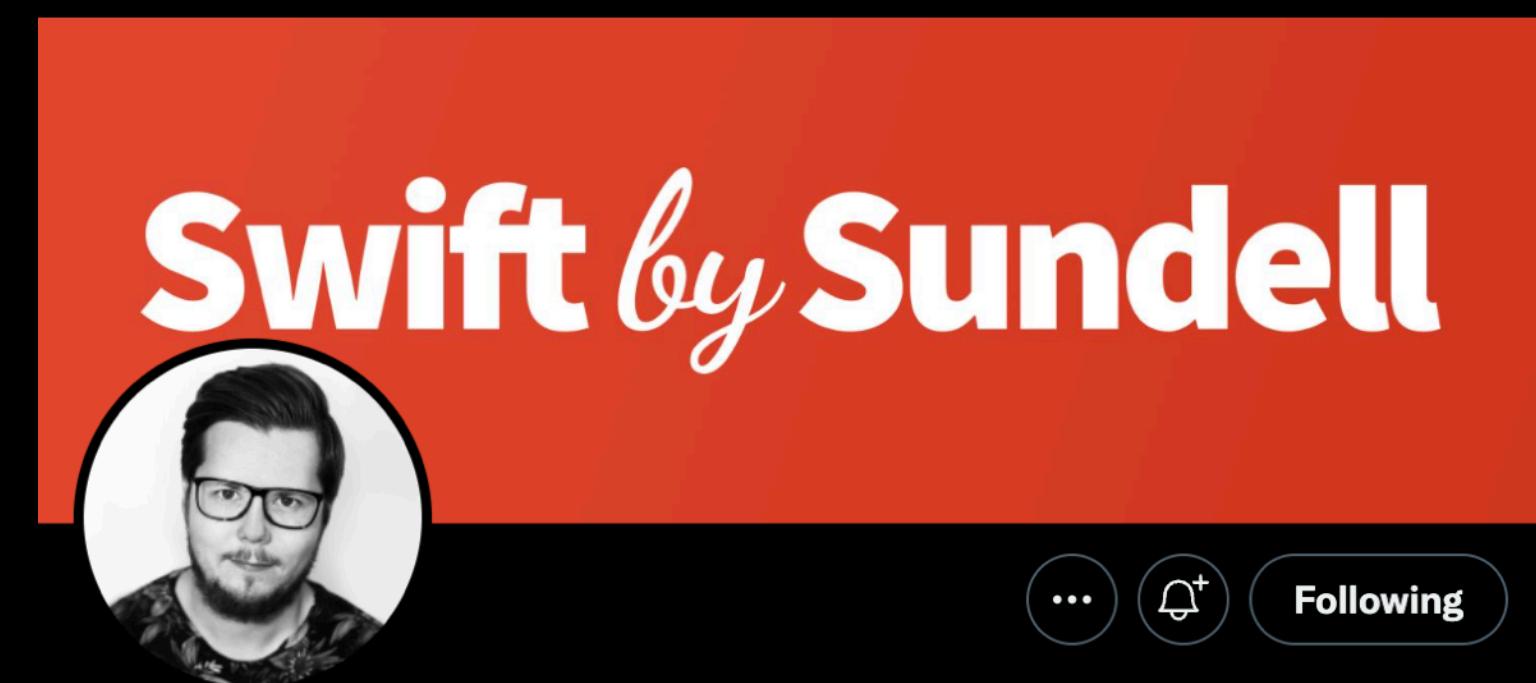
**Paul Hudson**

@twostraws

Learning and sharing at Hacking with Swift, author of Swift, SwiftUI, and iOS books, public speaker, Rubik's cube enthusiast, and herder of my kids. (He/him)

⌚ Bath, UK ⌚ [hackingwithswift.com](#) 📅 Joined July 2009

257 Following 68.3K Followers



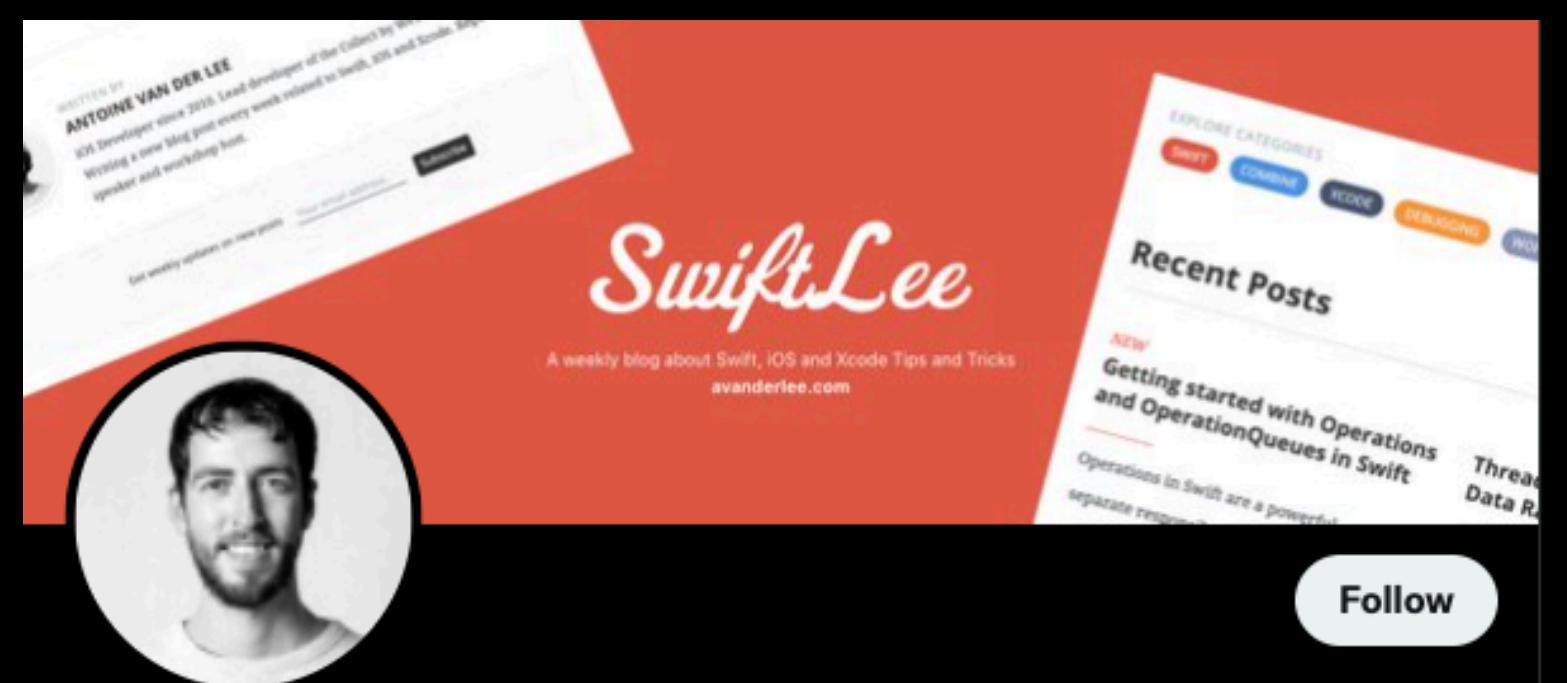
**John Sundell**

@johnsundell

Creator of [@swiftbysundell](#), co-host of [@stacktracepod](#), Swift developer, avid prototyper and amateur chef 🧑‍🍳🧑‍🍳

⌚ Gdańsk, Poland ⌚ [swiftbysundell.com](#) 📅 Joined February 2011

315 Following 46.7K Followers



**Antoine v.d. SwiftLee** 🐹

@twannl

🇳🇱 #iosdev #iOS @WeTransfer

#swift blog: [avanderlee.com](#)

Newsletter & discounts: [avanderlee.com/swiftleewee...](#)

Simulator enhanced: [rocketsim.app](#)

⌚ Amsterdam, The Netherlands ⌚ [avanderlee.com](#) 📅 Joined December 2009

711 Following 28.6K Followers

# Documentation and Resources

## Conferences

- WWDC (Worldwide Developer Conference) - Cupertino, CA  
<https://developer.apple.com/wwdc22/>
- #Pragma Conference - Bologna, IT  
<https://pragmaconference.com/>
- SwiftHeroes - Turin, IT  
<https://swiftheroes.com/>

# Demo

# Demo

## Xcode / SwiftUI / Documentation

- Make a start to build a weather app UI
- Time is extremely limited (~40 min)
- Basic interactions with Xcode
- Navigating the documentation



Source: <https://www.youtube.com/watch?v=HXoVSbwWUlk>

# Q&A