The case for macOS

Why native apps are worth fighting for

Charley Ho - Aug 1, 2023

Questions? Corrections? Just interrupt me!

Who am I?

I’m Charley, and I make Multi.app

- Formerly, I made Remotion - a virtual office for your hybrid team, right on your desktop
- I’ve been building native macOS apps full time for the past 4 years
- Before that, I was a more “traditional” engineer working on web/server/js in BigCo
- https://www.linkedin.com/in/charley-ho
- charleyho.com

First: A very rough history of macOS

1976: Apple Computer, Inc

- Apple II, Macintosh

1985: NeXT → OSX

- Objective C
- BSD
- Appkit: NS- classes

2007: iOS

- UIKit

2014: Swift

macOS is dead

- Apple killed it so that iOS could live
- Electron has taken over the desktop world
  - Spotify
  - Slack
  - Figma
  - VSCode
- lame example: Threads “desktop” is likely a web page
  - Messenger → electron
  - Instagram: Not even an iPad app.

https://medium.com/source-and-buggy/elegy-for-the-native-mac-app-39ee92cc37ba

Building macOS apps is hard

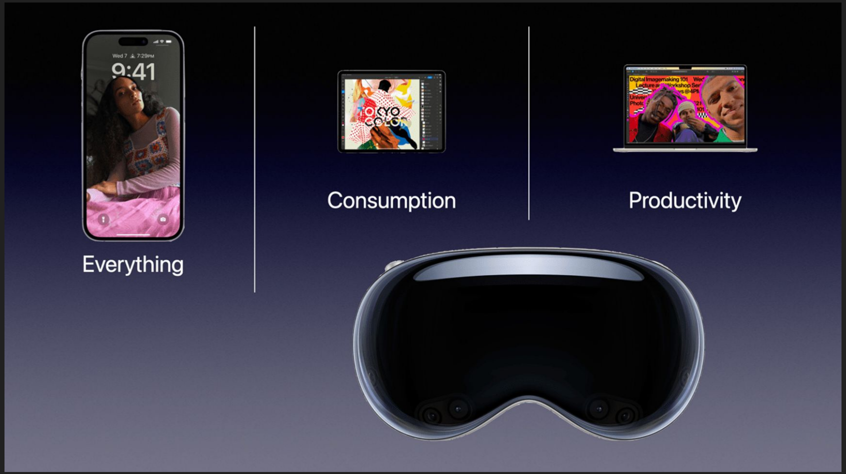
- Historically difficult: legacy, documentation, updates
- App store = bad, optional
- Market share: still a tough sell
- Apple doesn’t even care!
    - Mac Pro ???

2023: Things are different now

- Windows & Linux are stuck
- Apple Silicon >> Intel
- In a world of disposable software, Quality matters on Mac.
- macOS development is easier now
  - Catalyst, since macOS 10.15
  - SwiftUI
  - Universal hardware: Macs are basically just ipads with keyboards

Side note: But what about Vision Pro?

- https://stratechery.com/2023/apple-vision/
  - Everything? Consumption? Productivity?
- My personal bet for top use: larger screens for my mac
- Mobile computers have not replaced desktop software



The native difference

- Consistency
- Performance
- Access to “Apple Magic”
- Quality

Consistency

- Default Behaviors
  - Hover states
  - Dark mode
- Integration with macOS
  - Notifications
  - Again, Dark mode
  - accessibility
- Apple HIG guidelines
- The small things “feel right”
  - ie: Padding
  - ie: cmd+w, cmd+q, cmd+shift+/

Performance

- Smaller app sizes
- Smaller memory footprint
- Only the code you intend to write needs to run. Less overhead
- More control of the code - access all the way into the OS
- Objective-C and Swift are fast languages, especially compared to Electron & JS.

“Apple Magic”

- Access to Apple custom hardware, like “AI” APIs that run on the neural engine
- Access to special Apple APIs
- Apple developer documentation, tooling, and resources
  - Ie. Xcode, for better or worse
  - WWDC
  - Swift conferences & dev community

Why do I like macOS development?

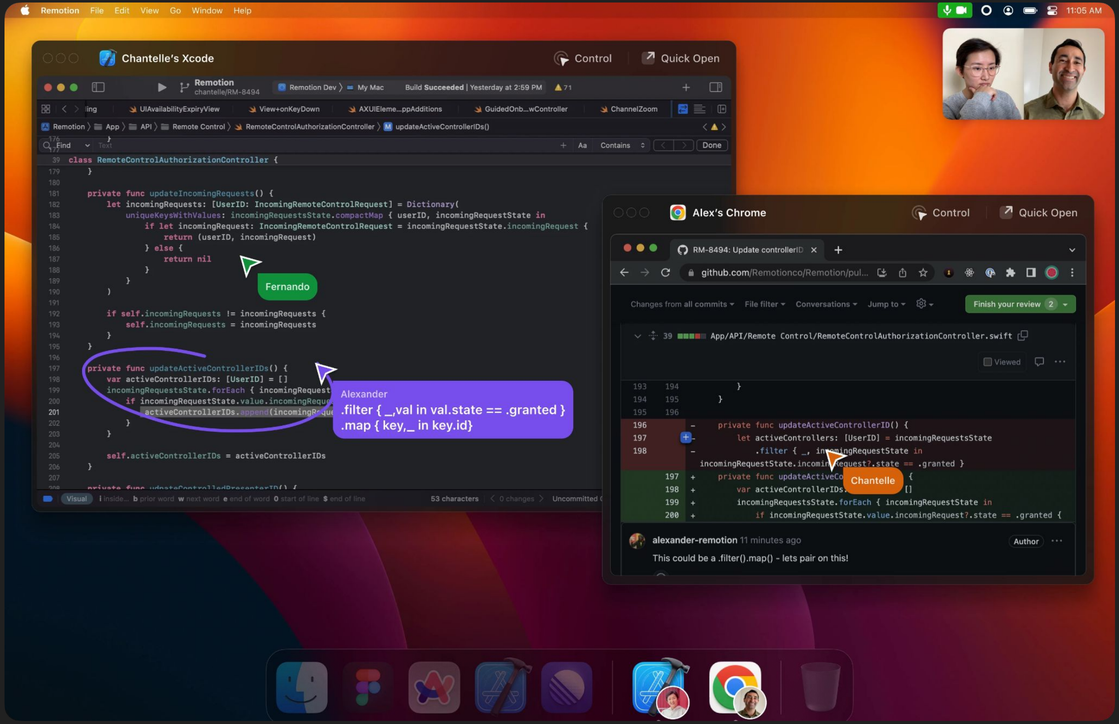
- I spend all my time in macOS. It’s where I get *work done*.
- Swift & Apple APIs are elegant to use.
- SwiftUI is *fast, easy,* and *stable* (or more stable than I originally anticipated).
- Most importantly, native development makes it possible to build to a ultra-high “Quality Bar”

Great native app examples

- Raycast
- Mimestream
- Arc
- Zed
- Warp
- Tuple

My app: Multi.app

- (formerly Remotion.app)
- multiplayer collaboration for macOS
- a fully native macOS app, built for pairing.



```swift
class RemoteControlAuthorizationController {
    }

    private func updateIncomingRequests() {
        let incomingRequests: [UserID: IncomingRemoteControlRequest] = Dictionary(
            uniqueKeysWithValues: incomingRequestsState.compactMap { userID, incomingRequestState in
                if let incomingRequest: IncomingRemoteControlRequest = incomingRequestState.incomingRequest {
                    return (userID, incomingRequest)
                } else {
                    return nil
                }
            }
        )

        if self.incomingRequests != incomingRequests {
            self.incomingRequests = incomingRequests
        }
    }

    private func updateActiveControllerIDs() {
        var activeControllerIDs: [UserID] = []
        incomingRequestsState.forEach { incomingRequest
            if incomingRequestState.value.incomingReque
                activeControllerIDs.append(incomingReque
            }
        }

        self.activeControllerIDs = activeControllerIDs
    }
```

Alexander

.filter { _,val in val.state == .granted }
.map { key,_ in key.id}

App/API/Remote Control/RemoteControlAuthorizationController.swift

```diff
             }
         }

-    private func updateActiveControllerID() {
-        let activeControllers: [UserID] = incomingRequestsState
-            .filter { _, incomingRequestState in
incomingRequestState.incomingRequest?.state == .granted }
+    private func updateActiveCo                 {
+        var activeControllerIDs:                []
+        incomingRequestsState.forEach { incomingRequestState in
+            if incomingRequestState.value.incomingRequest?.state == .granted {
```

alexander-remotion 11 minutes ago

This could be a .filter().map() - lets pair on this!

Demo!

How it’s built

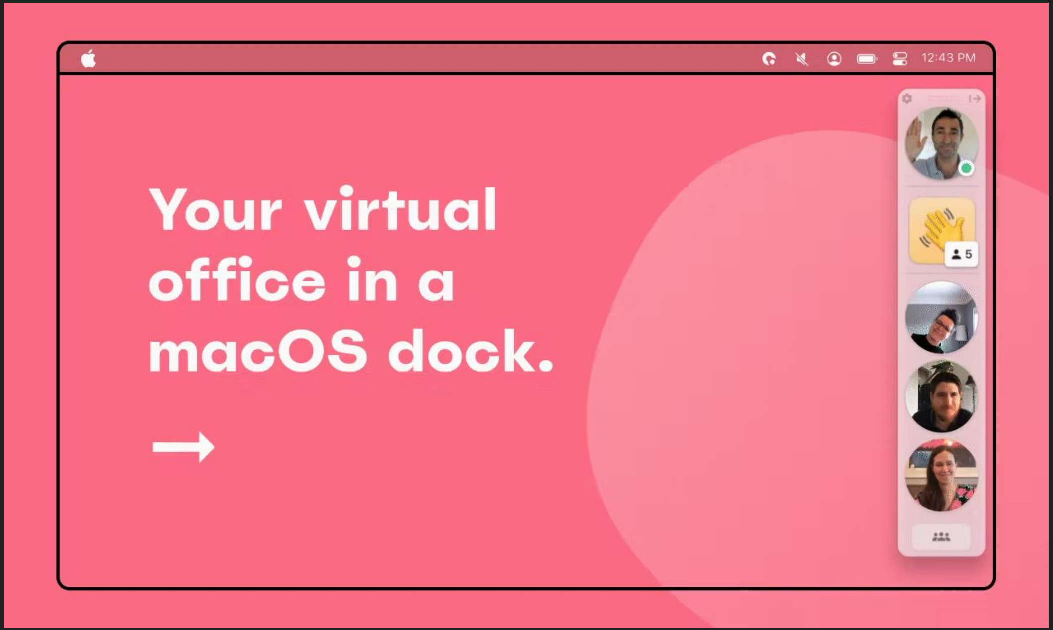
- Infra: GCP Firebase (Firestore, auth, storage, and much more)
- AV: Zoom + custom low level infra
  - ie. Metal renderer
  - ie. SCK
- SwiftUI
- Domain-based package architecture
- Combine & Async streams

Our Team:

- A small but mighty group with a deep passion for desktop software
- Design-forward execution
- Fully distributed globally

Remotion

- Our previous product, built over the previous 3 years.
- A dock for your team, in macOS
- quick calls: get into a chat with your team quickly.



Your virtual office in a macOS dock.

Appendix

- Elegy for native macOS apps
    - https://medium.com/source-and-buggy/elegy-for-the-native-mac-app-39ee92cc37ba
- Multi blog post
    - https://multi.app/blog/why-remotion-is-a-native-macos-app-not-electron
- Sketch
    - https://www.sketch.com/blog/part-of-your-world-why-we-re-proud-to-build-a-truly-native-mac-app/
- Stratechery
    - https://stratechery.com/2023/apple-vision/

Other topics to deep dive: Building for macOS

- Alternatives
- Build process
- Show some code
- Release process?
- Tooling

Alternatives to native desktop app development

- Electron
  - Most popular; huge tooling & dev community
  - Slowest, basically just a web wrapper
  - Origins: Atom browser.
- Tauri
  - New Rust-based electron alternative
- React Native for desktop
  - React → React Native → React Native for desktop
  - JS, but with native core components
  - Fast, but complex infra.
  - Run by Microsoft (!!!)
- Flutter
  - Write in Dart
  - Run by Google - so don’t trust it.
  - Probably the next codebase I would personally try.

Qs? charley@multi.app