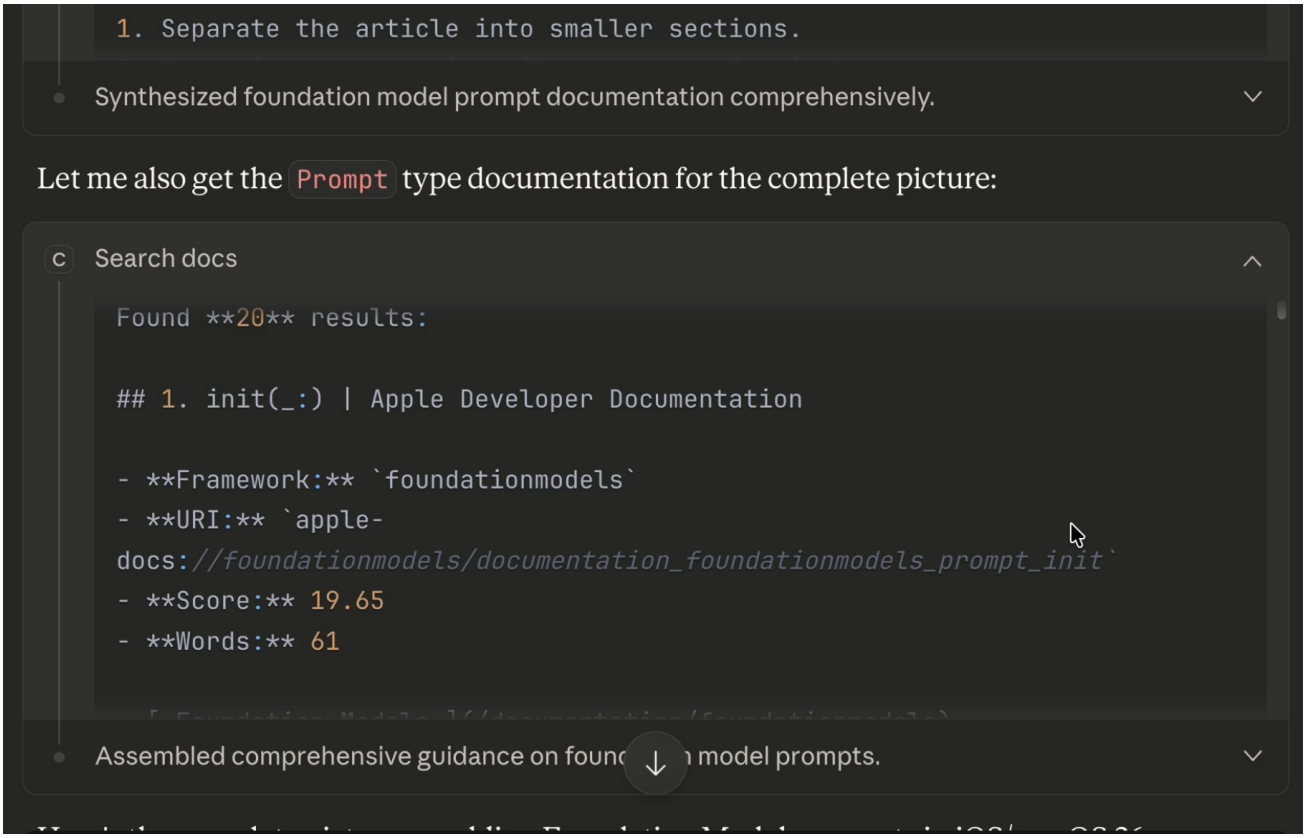
Cupertino: 27 Issues Down, Just Getting Started



I'm currently working on two macOS apps, and I rely heavily on Claude Code to help me ship faster. But here's the thing—I kept hitting the same wall: hallucinated APIs, outdated patterns, confident suggestions for methods that don't exist.

I'm not alone. Developers everywhere report the same issues: AI suggesting `@ObservableObject` and `@Published` when Apple now recommends `@Observable`. Xcode 26's AI integration hallucinating 100% of the time when asked about new 2025 frameworks. One developer described fixing AI-generated Swift code as "whack-a-mole"—fix one obsolete API, another pops up.

That frustration is why I built Cupertino.

In my previous post, I introduced it—an MCP server that gives Claude offline access to 22,000+ Apple documentation pages. That was version 0.1.x. Since then, things have evolved rapidly: nine releases in 72 hours, two new companion repositories, and a complete ecosystem for AI-assisted Apple development.

I need this tool. I use it every day. And I'm building it in public because I know I'm not the only one frustrated with AI that hallucinates Apple APIs.

What Changed: v0.1.6 ? v0.2.3

The original Cupertino solved the core problem—searching Apple docs locally. But real-world usage revealed gaps: ranking wasn't smart enough, storage was bloated, and legacy documentation was missing. Here's what shipped:

Intelligent Ranking (v0.2.1)

The initial BM25 search treated all documents equally. A query for "View" might return some random extension before `SwiftUI.View` itself. Eight ranking heuristics now prioritize:

Core types over extensions - `View` ranks above `View+Accessibility`
URL depth analysis - `/documentation/swiftui/view` beats `/documentation/swiftui/view/some/nested/thing`
Title pattern detection - Exact matches surface first
Modern over deprecated - Current APIs rank higher

The result: sub-100ms queries that actually return what you're looking for.

Storage Cleanup (v0.1.8, v0.2.0)

The initial crawl produced ~27GB of data. Most of it was cruft: `.git` folders from sample code downloads, `.DS_Store` files, Xcode user data. A new `cleanup` command reduced storage to 2-3GB—a 90% reduction.

Version 0.2.0 fixed a critical bug where cleanup was accidentally deleting source code instead of just metadata. Now 99.8% of sample ZIPs retain intact source.

Language Filtering (v0.1.9)

Not everyone wants Objective-C results. The CLI and MCP tools now accept a `language` parameter:

```
cupertino search "NSObject" --language swift
```

Claude can filter too—just ask for "Swift-only results."

Apple Archive Support (v0.2.3)

Apple's modern documentation is great, but some topics only exist in the legacy archive: Core Animation Programming Guide, Quartz 2D Programming Guide, Core Text Programming Guide. These are still the authoritative sources for low-level graphics work.

Cupertino now crawls `developer.apple.com/library/archive/` and integrates results with smart ranking that prioritizes modern docs while keeping legacy content searchable.

The Ecosystem: Three Repositories

Cupertino grew from one repo to three:

cupertino — The MCP Server

The main Swift package. Crawls, indexes, and serves documentation via MCP protocol. Install it, point Claude at it, done.

```
# Quick setup with Claude Code
claude mcp add cupertino --scope user -- /usr/local/bin/cupertino
```

cupertino-docs — Pre-Crawled Documentation

Don't want to crawl everything yourself? Clone this repo:

```
git clone https://github.com/mihaelamj/cupertino-docs ~/.cupertino
```

Currently includes:

Source	Content	Status
`swift-evolution/`	All Swift proposals (~400)	Ready
`swift-org/`	Swift.org language docs	Ready
`packages/`	Swift Package Index metadata	Ready
`docs/`	Apple Developer Documentation (13K+ pages)	WIP - requires manual crawl
`archive/`	Legacy Apple guides	WIP - requires manual crawl

Swift Evolution, Swift.org, and package docs are pre-indexed and ready. Apple Developer Documentation and Archive still require running `cupertino fetch` yourself due to size—but having the infrastructure in place is a start. The tooling exists; full pre-crawled distribution is coming.

cupertino-sample-code — 606 Apple Samples

This is the fun one. Every official Apple sample code project, cleaned and ready to build:

100+ frameworks covered: SwiftUI, ARKit, RealityKit, Metal, CoreML, Vision, AVFoundation, HealthKit, HomeKit, and more
606 projects: From "Hello World" to complex GPU shaders
Build-ready: No `.git` folders, no xcuserdata, no cruft
MIT Licensed: Use them however you want

When you ask Claude "show me how Apple implements ARKit face tracking," it can now search actual Apple sample code—not hallucinate an approximation.

Example projects include:

`arkit-*` — Augmented reality implementations
`swiftui-*` — Modern UI patterns
`metal-*` — GPU programming examples
`coreml-*` — Machine learning integrations
`avfoundation-*` — Video and audio capture

The 72-Hour Sprint

Nine releases shipped in three days. Here's the changelog:

Version	Key Changes
v0.1.6	JSON-first crawling, WKWebView memory fixes
v0.1.7	Swift Book content retrieval fixes
v0.1.8	Cleanup command (27GB ? 2-3GB)
v0.1.9	Language filtering (Swift/ObjC)
v0.2.0	Critical fix: source code retention
v0.2.1	Eight ranking heuristics
v0.2.2	URL depth analysis, resume fixes
v0.2.3	Apple Archive legacy docs

Each release addressed real issues discovered while using Cupertino with Claude Code. Fast iteration, real-world testing.

Using the Full Ecosystem

Here's the quickest path to AI-assisted Apple development with accurate documentation:

```
# 1. Get pre-crawled docs (skip the 20-hour crawl)
git clone https://github.com/mihaelamj/cupertino-docs ~/.cupertino

# 2. Build the MCP server
git clone https://github.com/mihaelamj/cupertino
cd cupertino
swift build -c release
sudo cp .build/release/cupertino /usr/local/bin/

# 3. Configure Claude Code
claude mcp add cupertino --scope user -- /usr/local/bin/cupertino

# 4. (Optional) Get sample code for reference
git clone https://github.com/mihaelamj/cupertino-sample-code ~/Apple-Samples
```

Now when you ask Claude about Apple APIs, it searches 22,000+ pages of real documentation. No hallucinations. No deprecated patterns. Just accurate, current information.

What's Next

The ranking heuristics work well but aren't perfect. Planned improvements:

- `make install` — One command to install everything: the MCP server, pre-crawled docs from cupertino-docs, and sample code from cupertino-sample-code
- **Semantic search** — Embeddings-based similarity for conceptual queries
- **Version awareness** — Filter by iOS/macOS version
- **Cross-reference linking** — "See also" connections between related docs
- **Sample code search** — Full-text search across all 606 projects

Join Me

If you're an Apple developer tired of AI hallucinations, give Cupertino a try. If you find bugs or have ideas, open an issue—27 closed so far, plenty more to go.

This isn't a polished product yet. It's a tool I'm building because I need it, and I'm sharing it because you might need it too.

Repositories

- cupertino — MCP Server (Swift)
- cupertino-docs — Pre-crawled documentation
- cupertino-sample-code — 606 Apple samples (MIT)

No more AI hallucinations about Apple APIs. Just real documentation that compiles.