

SwiftUI Minimal Project Structure

Single view SwiftUI app. Ideal for learning, prototypes, and small utility apps.

#swiftui #swift #ios #macos #apple #minimal

PNG

PDF

Copy

Prompt

Project Directory

MyApp/

- > **MyApp/** Main app target
 - MyAppApp.swift App entry point
 - ContentView.swift Main view
 - > **Assets.xcassets/** Images and colo...
 - AccentColor.colorset/**
 - AppIcon.appiconset/**
 - Preview Content/** Preview assets
- MyApp.xcodeproj Xcode project
- .gitignore

Why This Structure?

SwiftUI at its simplest—one view, one app file. Xcode generates this structure by default. **ContentView.swift** holds your entire UI. Perfect for learning SwiftUI fundamentals before adding complexity.

Key Directories

MyAppApp.swift - **@main** entry point with **WindowGroup**

ContentView.swift - Your app's main view hierarchy

Assets.xcassets/ - App icon, colors, and image assets

Preview Content/ - Assets only used in Xcode previews

Getting Started

- Open Xcode → File → New → Project
- Select iOS/macOS App template
- Choose SwiftUI for Interface
- Build and run on simulator or device

When To Use This

- Learning SwiftUI for the first time
- Quick prototypes and experiments
- Single-screen utility apps
- Widgets and app extensions
- Tutorial projects

When To Upgrade

- More than 3-4 views in ContentView
- Need to share state across views
- Adding navigation or tabs
- Need testable business logic
- Working with a team

Trade-offs

No architecture - View and logic mixed together

Hard to test - No separation of concerns

State sprawl - **@State** everywhere becomes messy

App Entry Point

```
// MyAppApp.swift
@main
struct MyAppApp: App {
    var body: some Scene {
        WindowGroup {
            ContentView()
        }
    }
}
```