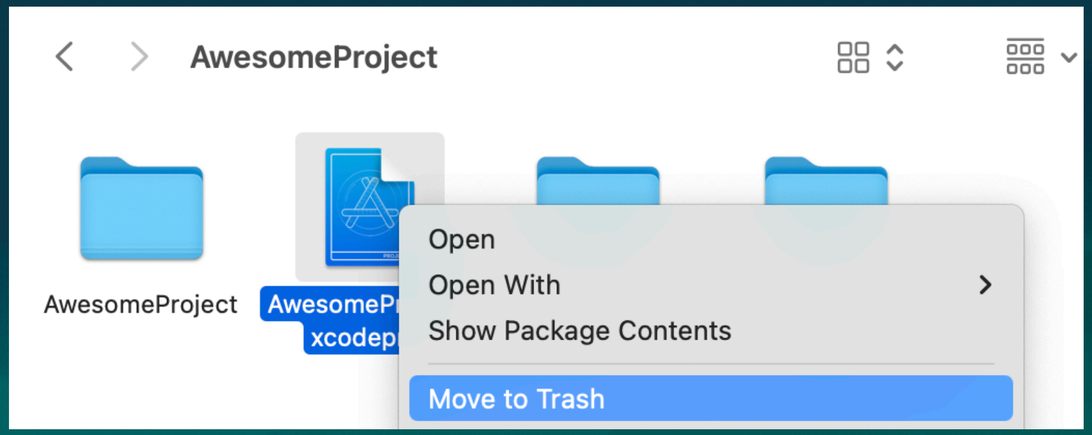
From .xcodeproj to Tuist: Modernizing Xcode Project Setup

Jonas Schlabertz, 24.04.2025

Why use Tuist?

- Avoid merge conflicts in .xcodeproj and .xcworkspace

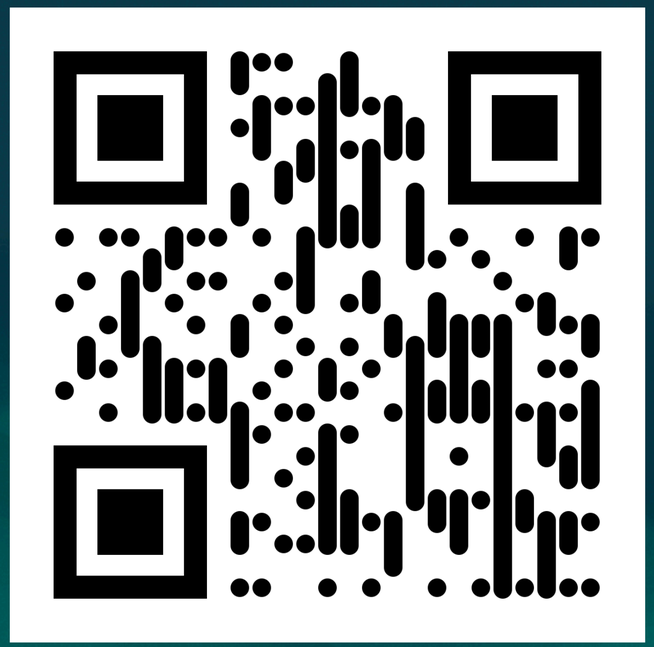
How?




```
~/Dev/AwesomeProject tuist generate
Loading and constructing the graph
It might take a while if the cache is empty
Using cache binaries for the following targets:
Generating workspace AwesomeProject.xcworkspace
Generating project AwesomeProject
Total time taken: 1.078s
✔ Success
Project generated.
```

What is Tuist?

- Idea: Delete your xcworkspace and xcodeproj folders, generate them instead when needed
- Avoid merge conflicts in Xcode project files
- Readable and maintainable specification of build parameters
- https://tuist.dev/
- Alternative: XcodeGen



Project.swift - Basic Setup

```
import ProjectDescription

let project = Project(
    name: "AwesomeProject",
    targets: [
        .target(
            name: "AwesomeProject",
            destinations: .iOS,
            product: .app,
            bundleId: "de.schlabertz.AwesomeProject",
            deploymentTargets: .iOS("18.0"),
            infoPlist: .file(path: "AwesomeProject/Info.plist"),
            sources: ["AwesomeProject/Sources/**"],
            resources: ["AwesomeProject/Resources/**"],
            dependencies: []
        )
    ]
)
```

Commands Overview

- `tuist init`
  - Generate new project, templates available for iOS, macOS, tvOS, watchOS
  - visionOS can be set as a destination later
- `tuist edit`
  - Opens a temporary Xcode project to edit the current Project.swift file
- `tuist generate`
  - Generates the Xcode project files and opens them in Xcode
  - `--no-open` parameter to suppress opening Xcode after generation

Custom Build Phases

- Defined per target
- Two types: `.pre` (build) and `.post` (build)

```
.target(
    …
    scripts: [
        .pre(script: """
                ./do-something.sh
            """,
            name: "Pre-Build-Script",
            inputPaths: ["$(PROJECT_DIR)/some-input-file.txt"],
            outputPaths: ["$(PROJECT_DIR)/some-output-file.txt"],
            basedOnDependencyAnalysis: true,
            runForInstallBuildsOnly: false)
    ],
    …
)
```

Dependencies

```
let project = Project(
    name: "AwesomeProject",
    packages: [
        .package(url: "https://github.com/onevcat/Kingfisher",
                 .upToNextMajor(from: "8.3.2"))
    ],
    targets: [
        .target(
            name: "AwesomeProject",
            …
            dependencies: [
                .package(product: "Kingfisher", type: .runtime),
                .target(someOtherTarget)
            ]
        )
    ]
)
```

Some tips for migrating

- Extract .xcconfig from .xcodeproj

```
✘ ~/Dev/AwesomeProject tuist migration settings-to-xcconfig  -p AwesomeProject.xcodeproj -x AwesomeProject.xcconfig
✔ Success
Build settings successfully extracted into /Users/schlabbi/Dev/AwesomeProject/AwesomeProject.xcconfig
```

```
let project = Project(
    name: "AwesomeProject",
    settings: .settings(configurations: [
        .debug(name: "Debug", xcconfig: "./AwesomeProject.xcconfig"),
        .release(name: "Release", xcconfig: "./AwesomeProject.xcconfig")
    ])
    ...
)
```

- Troubleshooting: Open original and generated .xcodeproj/project.pbxproj in text editor and compare

Any Questions?

Jonas Schlabertz
jonas@schlabertz.de
CocoaHeads Aachen, 24.04.2025