XcodeGen
Why & How?

Sebastian Osiński

Senior iOS Engineer

@

WARNING!

⚠️ No Swift Code Ahead ⚠️

⚠️ YAMLing in progress ⚠️

Why?

The pains of Xcode

Collaboration

Working with other developers

Expectation:

Working with other developers

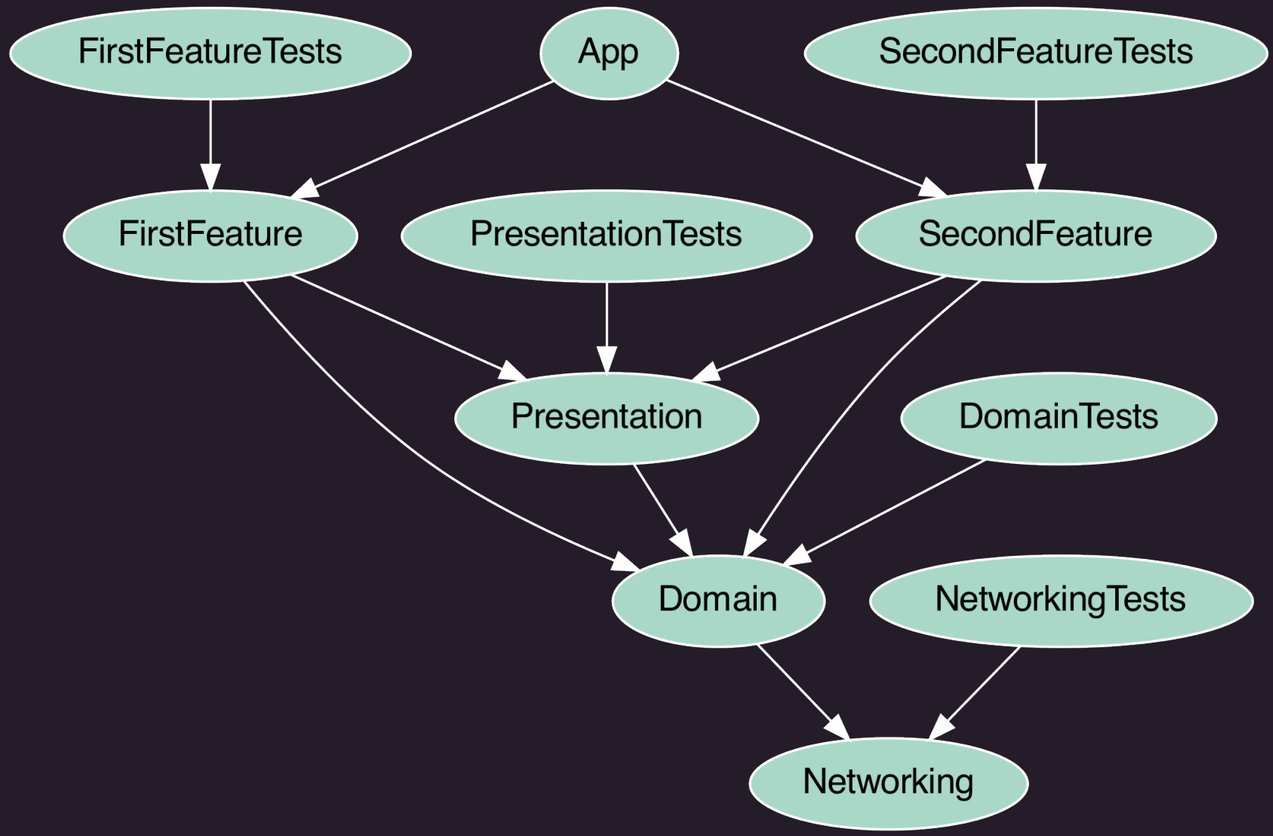
Reality:

```
<<<<<<< HEAD
        91E0475B282D7FDF00536ADC /* Never.swift in Sources */
        91E0475D282D7FEC00536ADC /* Gonna.swift in Sources */
        91E0475F282D7FF200536ADC /* Give.swift in Sources */
        91E04761282D7FF900536ADC /* You.swift in Sources */ =
        91E04763282D7FFF00536ADC /* Up.swift in Sources */ =
=======
        91E04765282D802400536ADC /* Never.swift in Sources */
        91E04767282D803200536ADC /* Gonna.swift in Sources */
        91E04769282D803800536ADC /* Let.swift in Sources */ =
        91E0476B282D803D00536ADC /* You.swift in Sources */ =
        91E0476D282D804300536ADC /* Down.swift in Sources */
>>>>>>> other-branch
```

Modularization

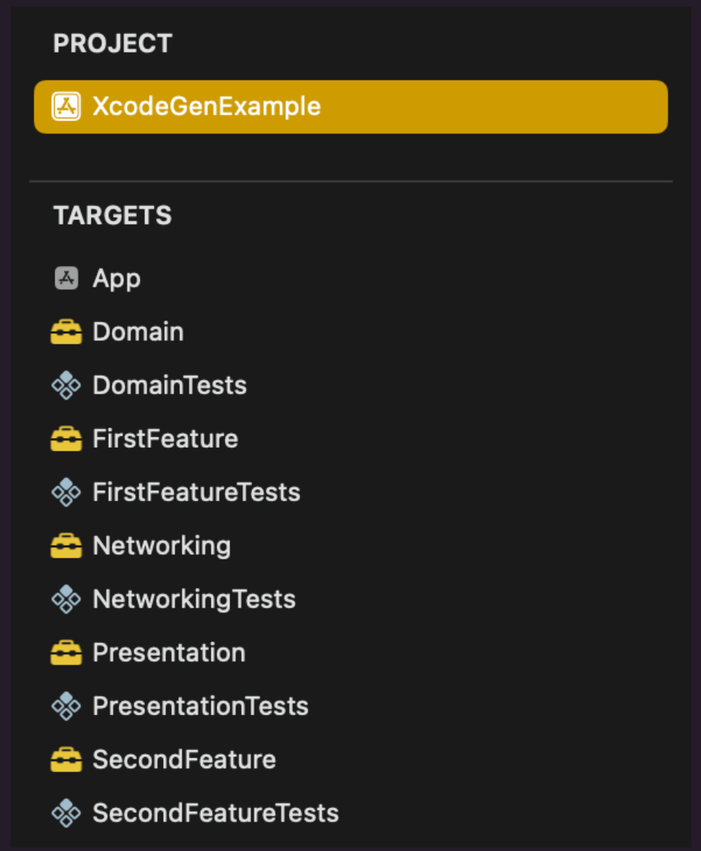
Modularized projects

Expectation:



Modularized projects

Reality:



Modularized projects

Reality:

Assets

Name	App	Domain	DomainTests	FirstFeature	FirstFeature...	Networking	Networking...	Presentation	Presentatio...	SecondFeat...	SecondFeatu...
ASSET_PACK_MANIFEST_UR...											
EMBED_ASSET_PACKS_IN_P...	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
ENABLE_ON_DEMAND_RES...	YES	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
ON_DEMAND_RESOURCES_I...											
ON_DEMAND_RESOURCES_...											

Build Locations

Name	App	Domain	DomainTests	FirstFeature	FirstFeature...	Networking	Networking...	Presentation	Presentatio...	SecondFeat...	SecondFeatu...
OBJROOT	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)	$(SYMROOT)
SHARED_PRECOMPS_DIR	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...	$(OBJROOT)/S...
SYMROOT	build	build	build	build	build	build	build	build	build	build	build

Build Options

Name	App	Domain	DomainTests	FirstFeature	FirstFeature...	Networking	Networking...	Presentation	Presentatio...	SecondFeat...	SecondFeatu...
ALLOW_TARGET_PLATFORM...	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
ALWAYS_EMBED_SWIFT_ST...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...	$(EMBEDDED_...
APPLICATION_EXTENSION_A...	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO
BUILD_LIBRARY_FOR_DISTRI...	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO	NO

How?

Solution:

XcodeGen 

Goodbye .xcodeproj 

Welcome project.yml 

XcodeGen

- ✦ generates .xcodeproj for you based on YAML/JSON config file and project's file structure
- ✦ config file **editable**, **readable** and **mergable** by humans

Installation

```
> brew install xcodegen
```

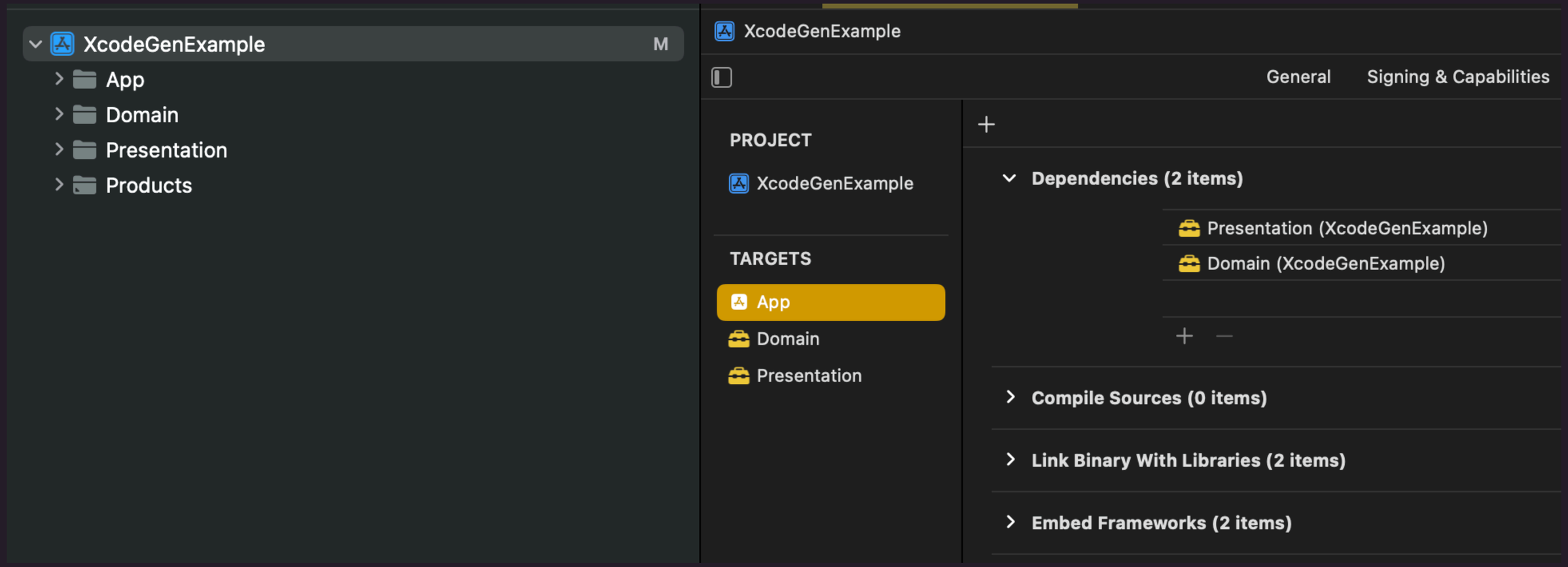
project.yml

```yaml
name: XcodeGenExample
options:
  bundleIdPrefix: dev.osinski
  deploymentTarget: "15.0"
targets:
  App:
    type: application
    platform: iOS
    sources:
      - App
    dependencies:
      - target: Presentation
      - target: Domain
  Presentation:
    type: framework
    platform: iOS
    sources:
      - Presentation
  Domain:
    type: framework
    platform: iOS
    sources:
      - Domain
```

Usage

```
> xcodegen generate
⚙️  Generating plists...
⚙️  Generating project...
⚙️  Writing project...
Created project at XcodeGenExample/XcodeGenExample.xcodeproj
```

Generated project



Problem:

Kinda repetitive

Solution:

Templates

Templates

```
targetTemplates:
  BaseModule:
    platform: iOS
    sources:
      - ${target_name}
  FrameworkModule:
    templates:
      - BaseModule
    type: framework
```

project.yml revisited

```yaml
include:
  - templates.yml
targets:
  App:
    templates:
      - BaseModule
    type: application
    dependencies:
      - target: Presentation
      - target: Domain
  Presentation:
    templates:
      - FrameworkModule
  Domain:
    templates:
      - FrameworkModule
```

What else can it do?

Build settings

```
settings:
  base:
    SWIFT_OBJC_BRIDGING_HEADER: BridgingHeader.h
    OTHER_LDFLAGS:
      - "-ObjC"
    USER_HEADER_SEARCH_PATHS:
      - $(PROJECT_DIR)/Headers/**
    # Very weird bug haunting only this app
    # See rdar://123456789
    VERY_OBSCURE_BUT_IMPORTANT_SETTING:
      - "-Foobarize"
```

Build settings

```
settings:
  base:
    SWIFT_OBJC_BRIDGING_HEADER: BridgingHeader.h
  configs:
    Debug:
      SOME_SETTING: "Foo"
    Release:
      SOME_SETTING: "Bar"
```

Info.plist

```
info:
  path: Info.plist
  properties:
    CFBundleDisplayName: Awesome App
    ITSAppUsesNonExemptEncryption: false
    NSLocationWhenInUseUsageDescription: We want to know
    UIBackgroundModes:
      - remote-notification
    MyOwnInfoPlistKey: SomeValue
```

Entitlements

```
entitlements:
  path: App.entitlements
  properties:
    aps-environment: development
    com.apple.developer.associated-domains:
      - webcredentials:osinski.dev
      - applinks:osinski.dev
```

Build phase scripts

```
preBuildScripts:
  - path: scripts/pre_build_script.sh
    name: First Script
    inputFiles:
      - $(SRCROOT)/input
    outputFiles:
      - $(DERIVED_FILE_DIR)/output
postCompileScripts:
  - script: some_shell_command -a
    name: My Command Script
postBuildScripts:
  - path: script/post_build_script.sh
    name: Final Script
```

Subprojects

```
projectReferences:
  SomeSubproject:
    path: "./Subproject/Subproject.xcodeproj"
```

This merging behaviour can be overridden on a value basis. If you wish to replace a whole value (set a new dictionary or new array instead of merging them) then just affix `:REPLACE` to the key

```
  ba  yml
  :  mSpec
targets:
  MyTarget: # target lives in base.yml
    sources:REPLACE:
      - my_new_sources
```

No ha g an ca sc chang by ing a e be to arg

Options

- **minimumXcodeGenVersion**: **String** - The minimum version of XcodeGen required.
- **carthageBuildPath**: **String** - The path to the carthage build directory. Defaults to `Carthage/Build`. This is used when specifying target arthage dependencies.
- ** thageE ut ePa S g** - T pat the carthage executable. Defaults to `carthage`. You can specify when you use custom uilt od i lle rt e us Mint for example.
- **createIntermediateGroups**: **Bool** - If this is specified and set to `true`, then intermediate groups will be created for every path component between the folder containing the source and next existing group it finds or the base path. For example, when enabled if a source path is specified as `Vendor/Foo/Hello.swift`, the group `Vendor` will created as a parent of the `Foo` group. This can be overriden in a specific Target source
- **bundleIdPrefix**: **String** - If this is specified then any target that doesn't have an `PRODUCT_BUNDLE_IDENTIFIER` (via all levels of build

It can do much much more...

XcodeGen & Package Managers

Swift Package Manager

```
packages:
  AwesomePackage:
    url: https://github.com/somebody/AwesomePackage
    from: 1.2.3
targets:
  TargetName:
    dependencies:
      - package: AwesomePackage
```

Carthage

```
TargetName:
  dependencies:
    - carthage: SomeAwesomeLibrary
```

CocoaPods

```
options:
  postGenCommand: pod install
```

CocoaPods

```
> xcodegen generate
⚙️  Generating plists...
⚙️  Generating project...
⚙️  Writing project...
Created project at XcodeGenExample/XcodeGenExample.xcodeproj
Analyzing dependencies
Downloading dependencies
Generating Pods project
Integrating client project
Pod installation complete!
```

XcodeGen & Existing projects

Solution:

xcdiff

xcdiff

```
> xcdiff -p1 original.xcodeproj -p2 xcodegen_generated.xcodeproj
❌ FILE_REFERENCES
❌ TARGETS > NATIVE targets
✅ TARGETS > AGGREGATE targets
✅ CONFIGURATIONS > Root project
✅ SETTINGS > Root project > "Debug" configuration > Base configuration
❌ SETTINGS > Root project > "Debug" configuration > Values
✅ SETTINGS > Root project > "Release" configuration > Base configuration
❌ SETTINGS > Root project > "Release" configuration > Values
❌ SOURCE_TREES > Root project
❌ ATTRIBUTES > Root project
✅ SWIFT_PACKAGES
```

xcdiff --verbose

```
> xcdiff -p1 original.xcodeproj -p2 xcodegen_generated.xcodeproj --verbose
 …
❌ SETTINGS > Root project > "Debug" configuration > Values
⚠️  Value mismatch (1):
  • IPHONEOS_DEPLOYMENT_TARGET
    ◦ 14.0
    ◦ 15.4
 …
❌ ATTRIBUTES > Root project
⚠️  Only in first (1):
  • ORGANIZATIONNAME = Sebastian Osiński
 …
```

Good practices

Use cache to speed project generation

```
> xcodegen generate --use-cache
Project has not changed since cache was written
```

Use git hooks

```
> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
⚙️  Generating plists ...
⚙️  Generating project ...
⚙️  Writing project ...
Created project at XcodeGenExample/XcodeGenExample.xcodeproj
```

Split project.yml into multiple files

```yaml
name: TheApp
include:
- xcodegen/templates.yml
- xcodegen/app.yml
- xcodegen/modules.yml
- xcodegen/ui_tests.yml
```

Getting rid of XcodeGen

Getting rid of XcodeGen

1. Run XcodeGen one last time 🛠️
2. Start tracking `.xcodeproj` file with git 💾
3. Stop using XcodeGen ✅

XcodeGen

vs

Tuist

XcodeGen

- YAML/JSON
- simple
- sometimes hard to debug
- fairly easy to integrate with complicated projects

Tuist

- Swift
- compile time safety
- history of major breaking changes
- integration problems due to its strictness

Thank you!

Resources

- https://github.com/yonaskolb/XcodeGen
- https://github.com/bloomberg/xcdiff

Getting in touch

 OSINSKI.DEV

 @seb_osinski

Q&A