#WWDC18

Building Faster in Xcode

Session 408

David Owens, Xcode Engineer
Jordan Rose, Swift Engineer

Building Faster in Xcode

Building Faster in Xcode

Increasing Build Efficiency

Reducing the Work on Rebuilds

Building Faster in Xcode

Increasing Build Efficiency	Reducing the Work on Rebuilds
Parallelizing your build process	Declaring script inputs and outputs
Measuring your build time	

Building Faster in Xcode

Increasing Build Efficiency	Reducing the Work on Rebuilds
Parallelizing your build process	Declaring script inputs and outputs
Measuring your build time	Understanding dependencies in Swift
Dealing with complex expressions	Limiting your Objective-C/Swift interface

Parallelizing Your Build

Increasing build efficiency

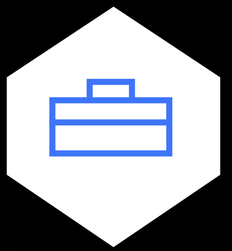
Xcode’s Targets and Dependencies

Xcode’s Targets and Dependencies

Target specifies a product to build

- iOS App
- Framework
- Unit Tests

Game



Framework

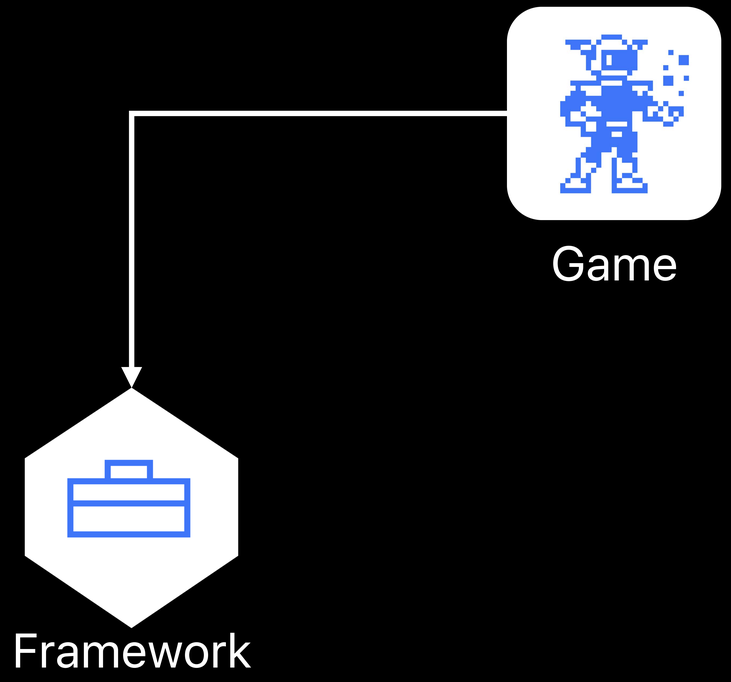
Xcode’s Targets and Dependencies

Target specifies a product to build

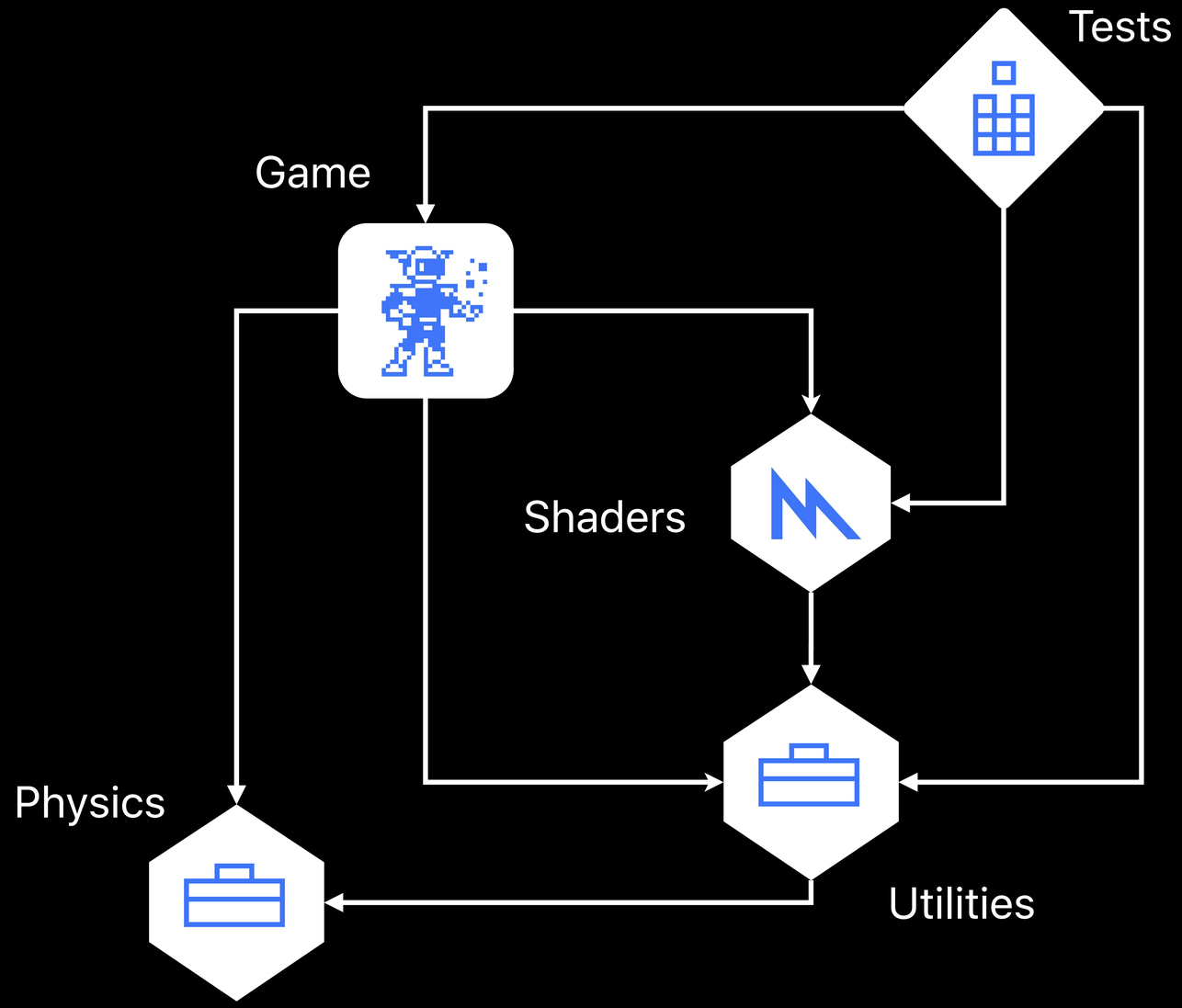
- iOS App
- Framework
- Unit Tests

Target dependency requires another target

- Explicit via Target Dependencies
- Implicit via Link Binary with Libraries

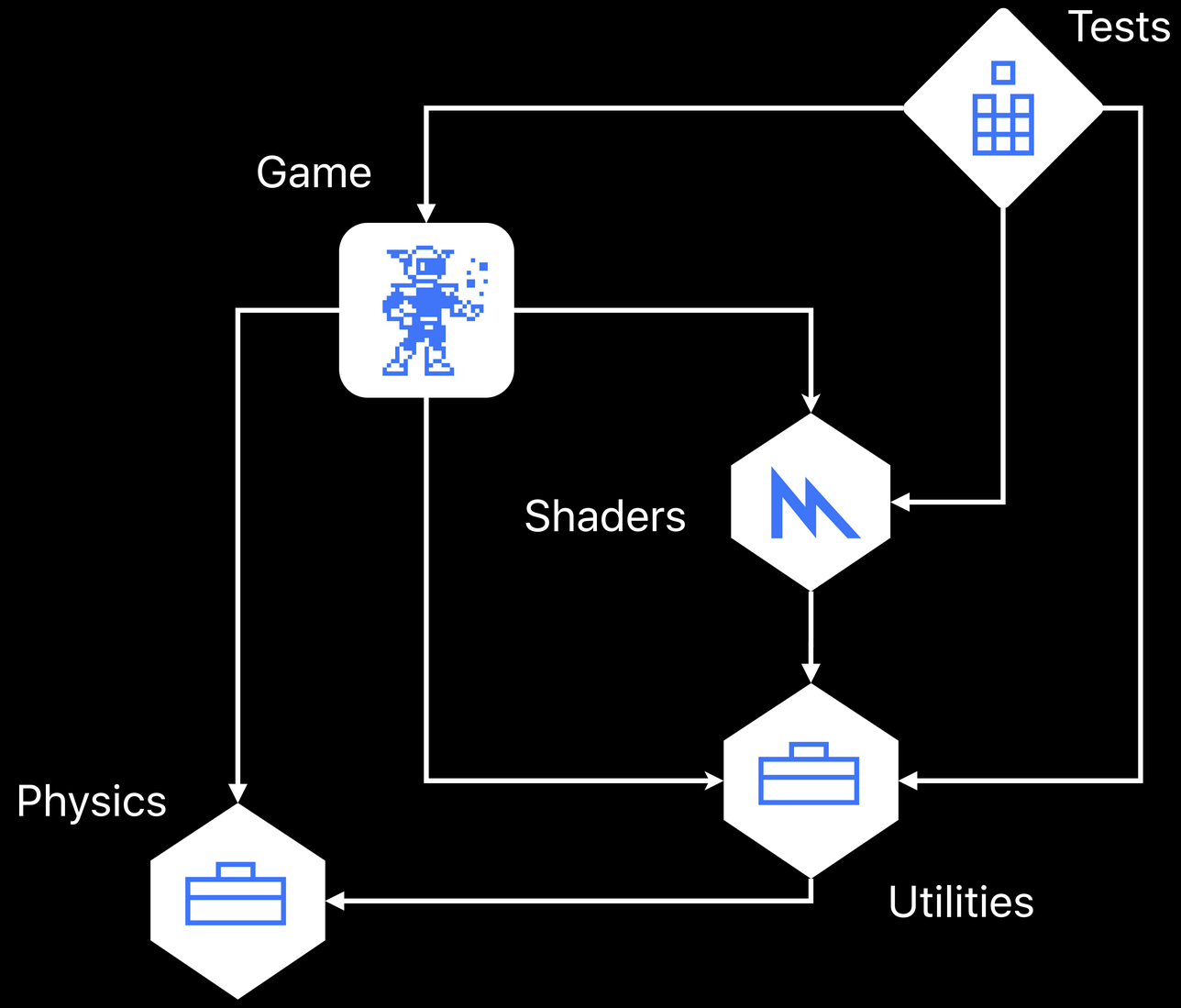


Game Dependency Graph



Game Dependency Graph

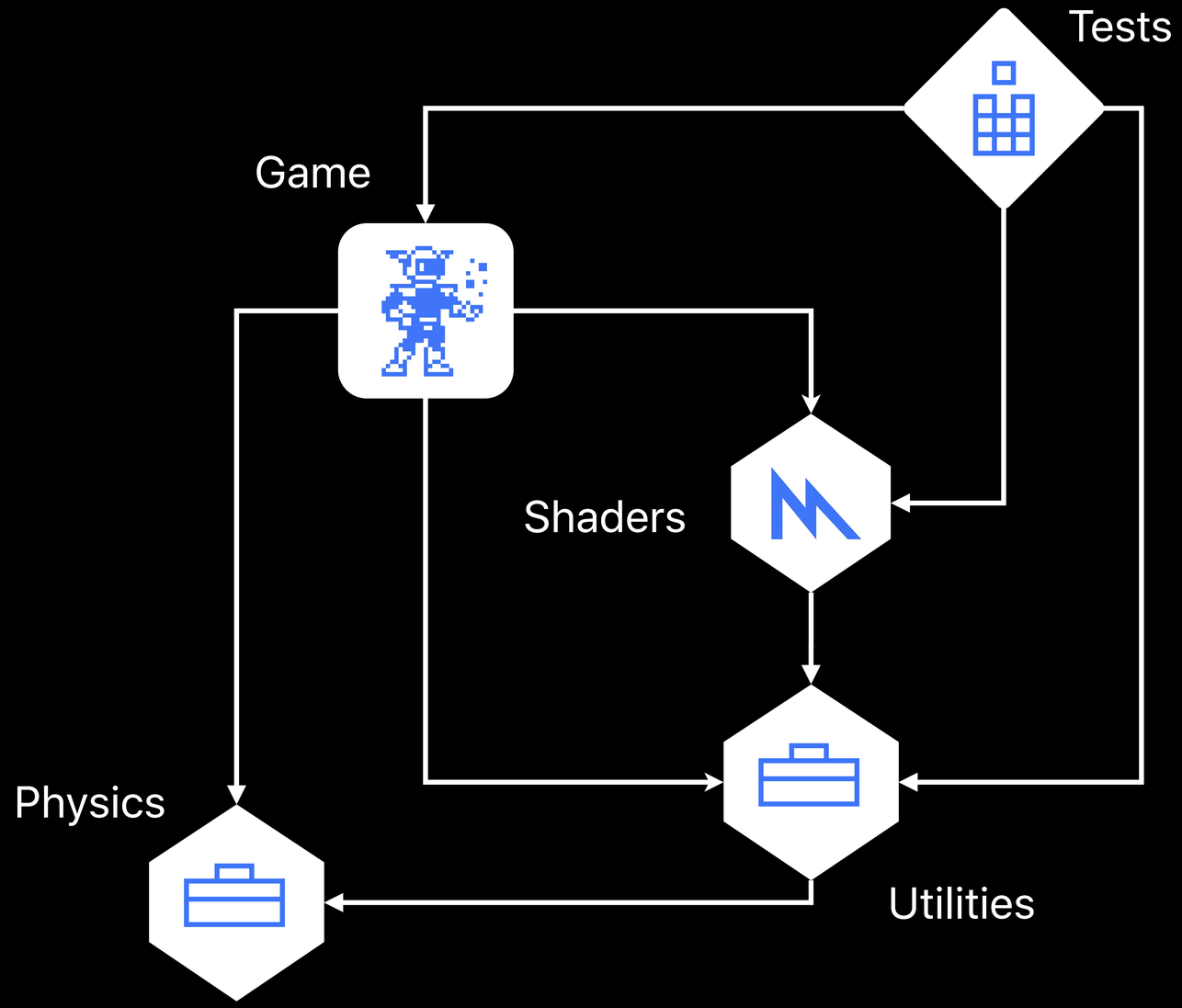
List of all targets to build



Game Dependency Graph

List of all targets to build

Dependency between targets

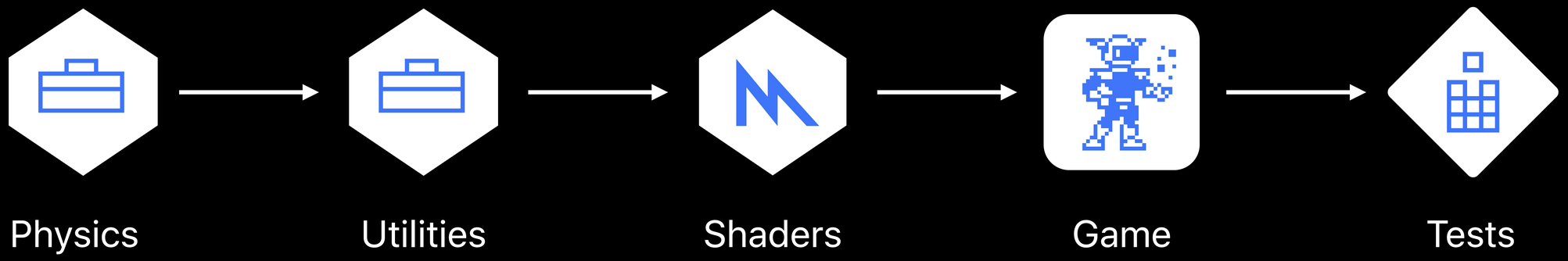


Game Dependency Graph

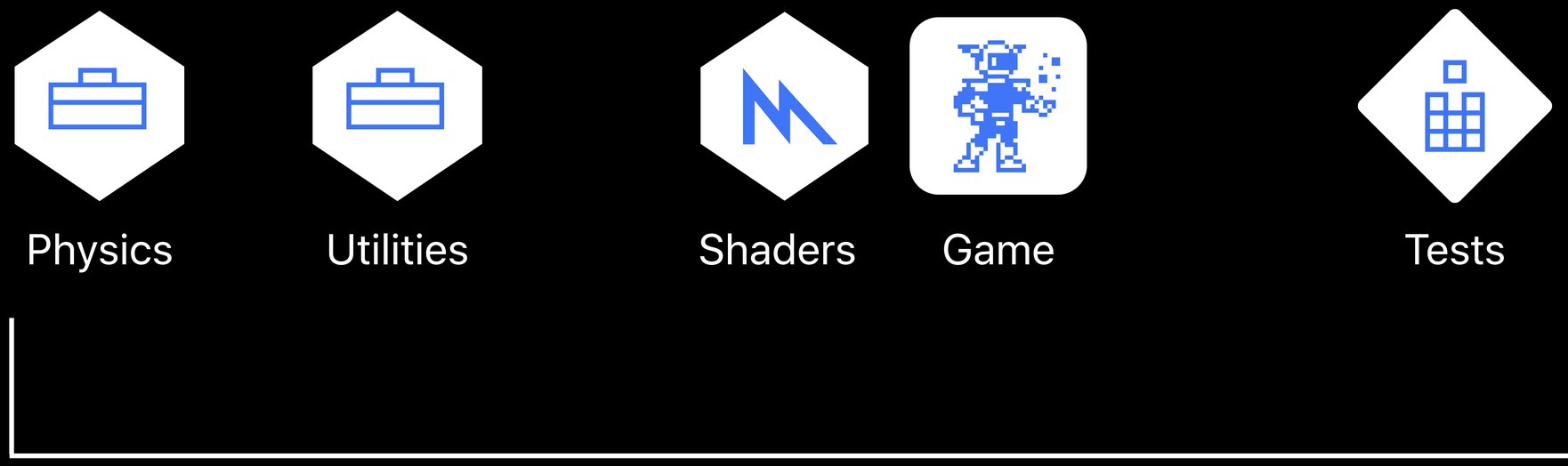
List of all targets to build

Dependency between targets

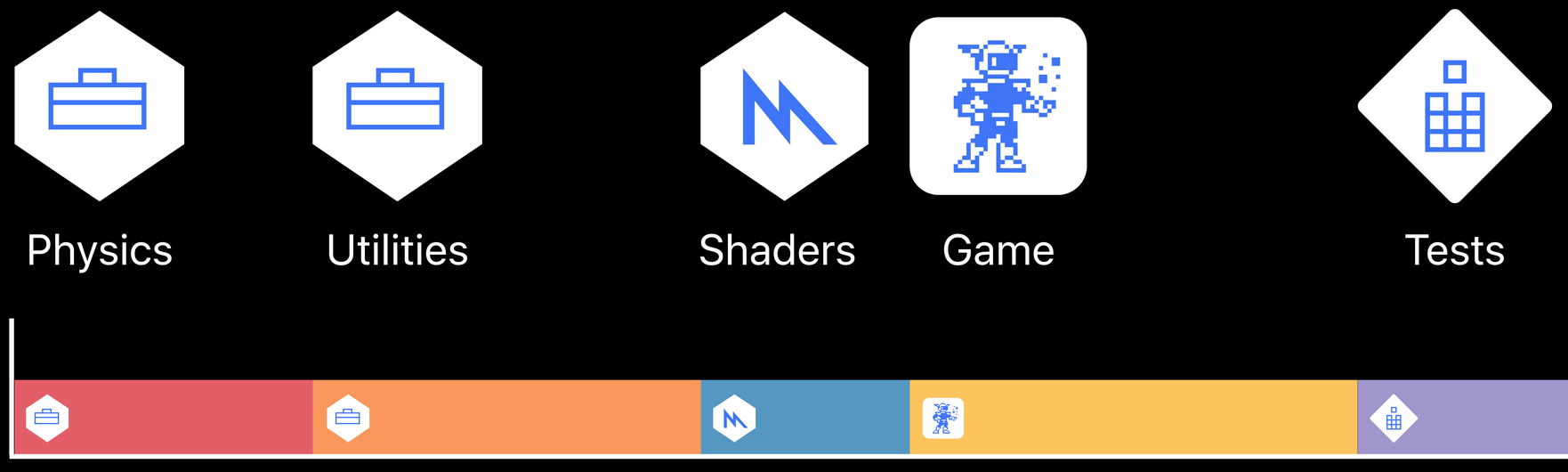
Build order can be derived



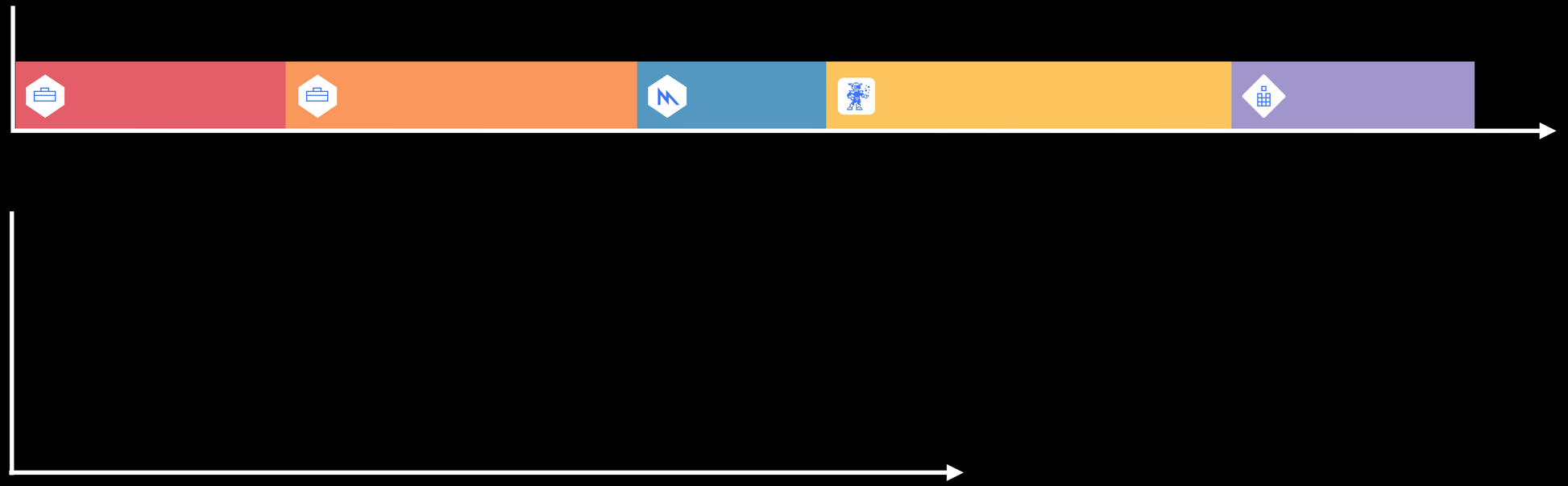
Serialized Build Timeline



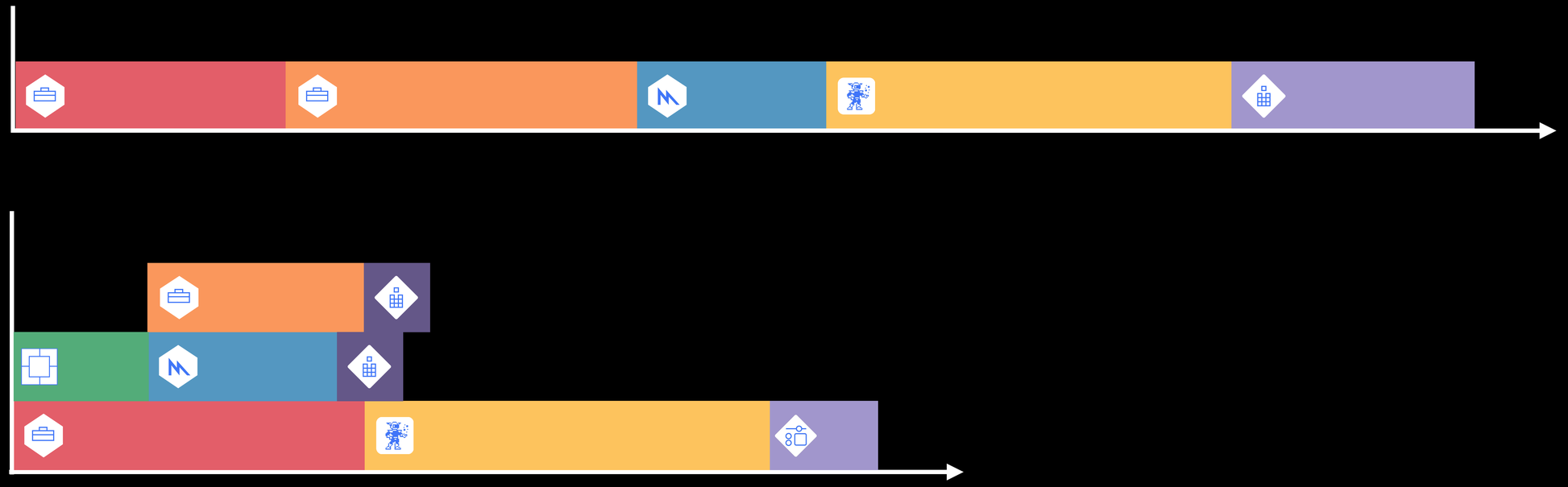
Serialized Build Timeline



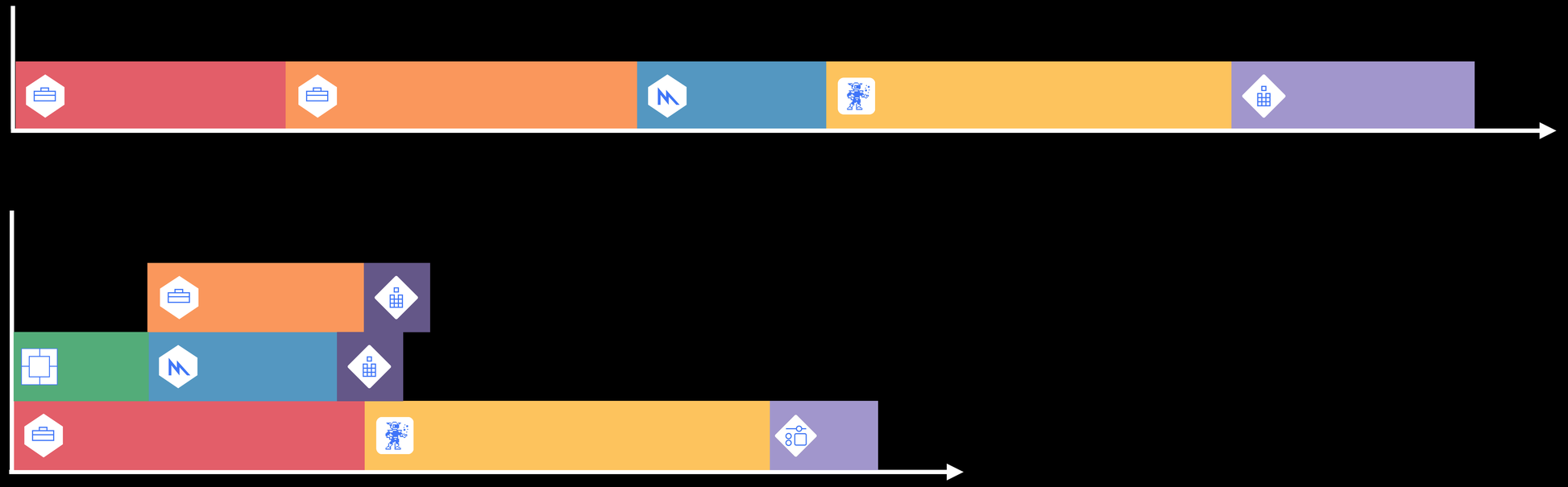
Parallelized Build Timeline



Parallelized Build Timeline

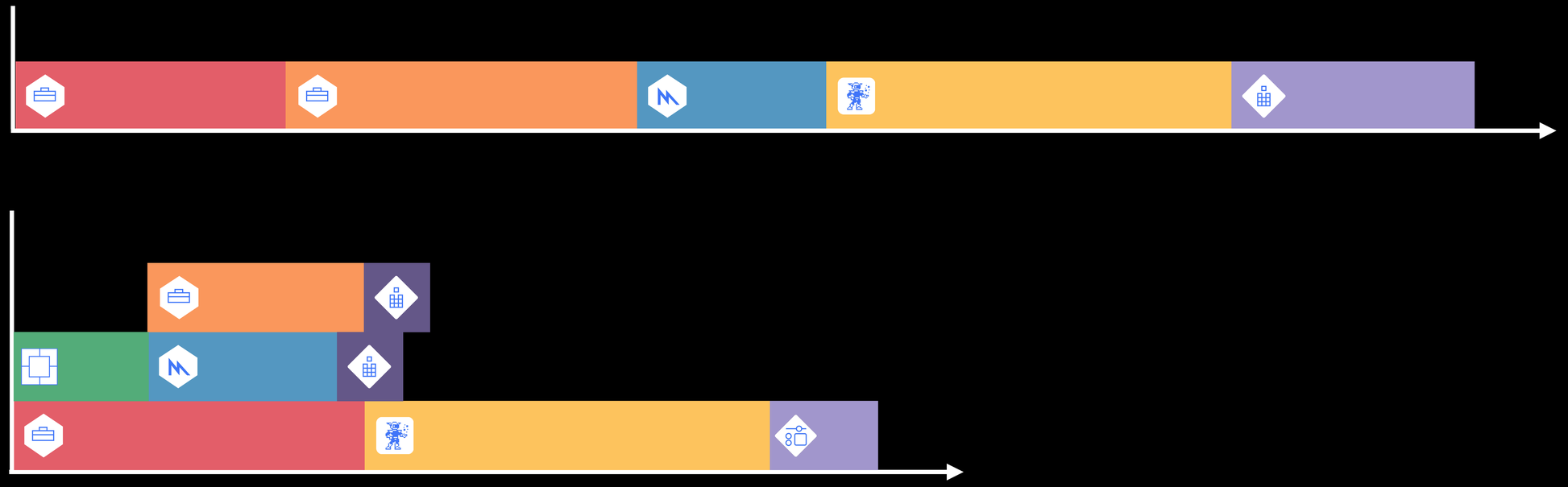


Parallelized Build Timeline



Amount of work did not change

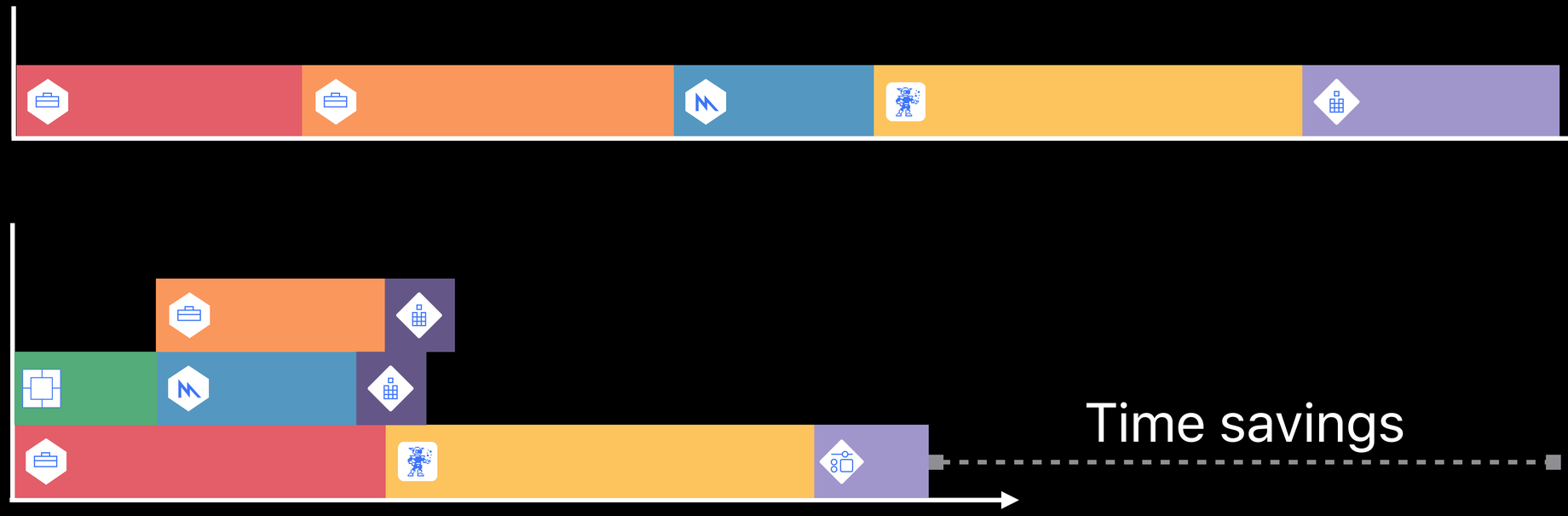
Parallelized Build Timeline



Amount of work did not change

Time to build decreased

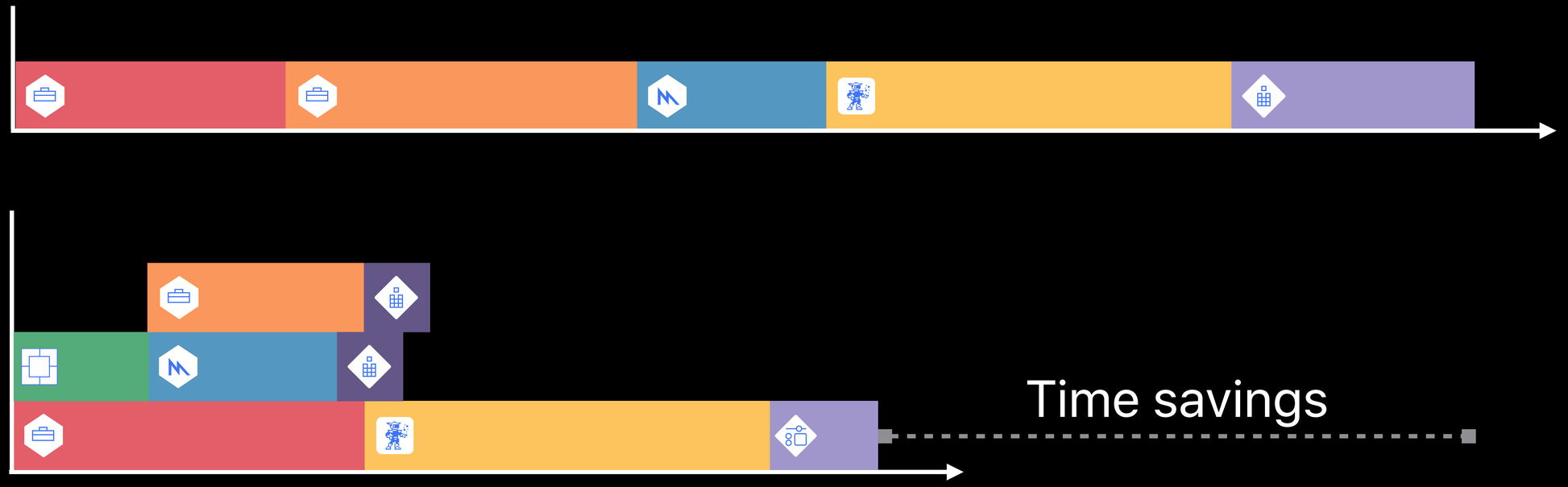
Parallelized Build Timeline



Amount of work did not change

Time to build decreased

Parallelized Build Timeline

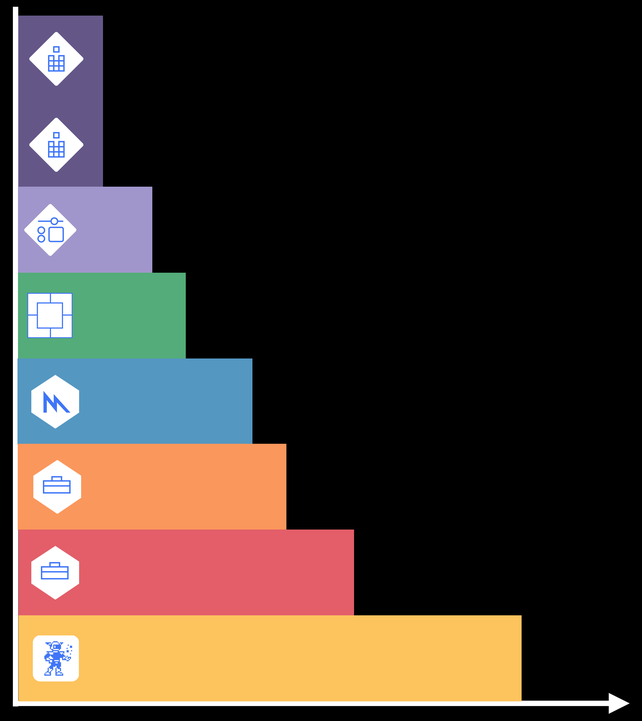


Amount of work did not change

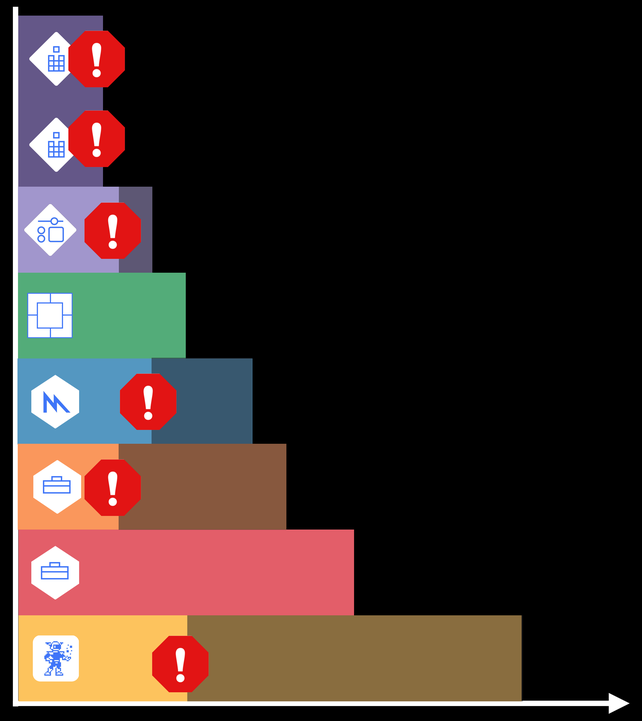
Time to build decreased

Increased hardware utilization

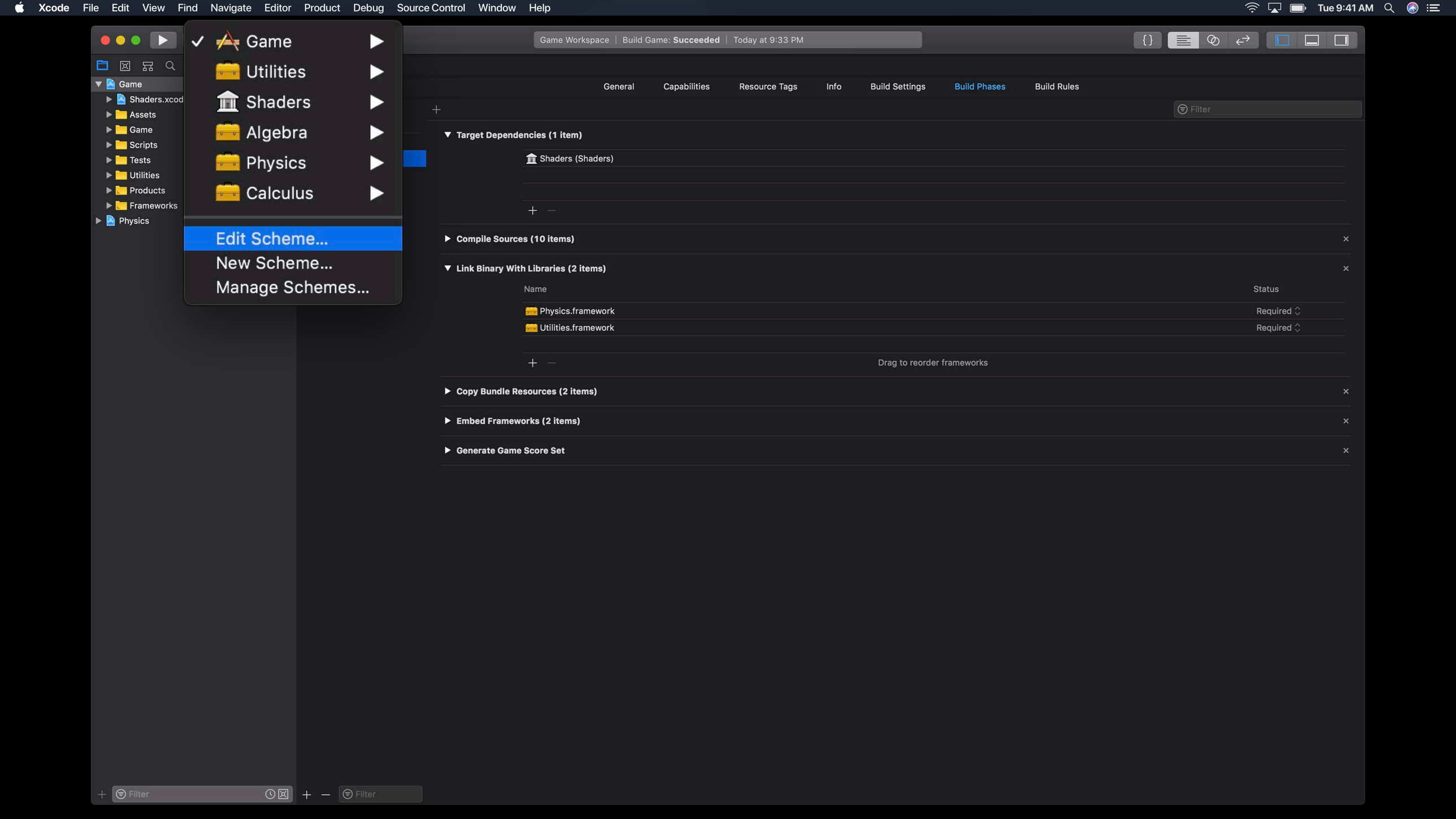
Extreme Parallelization

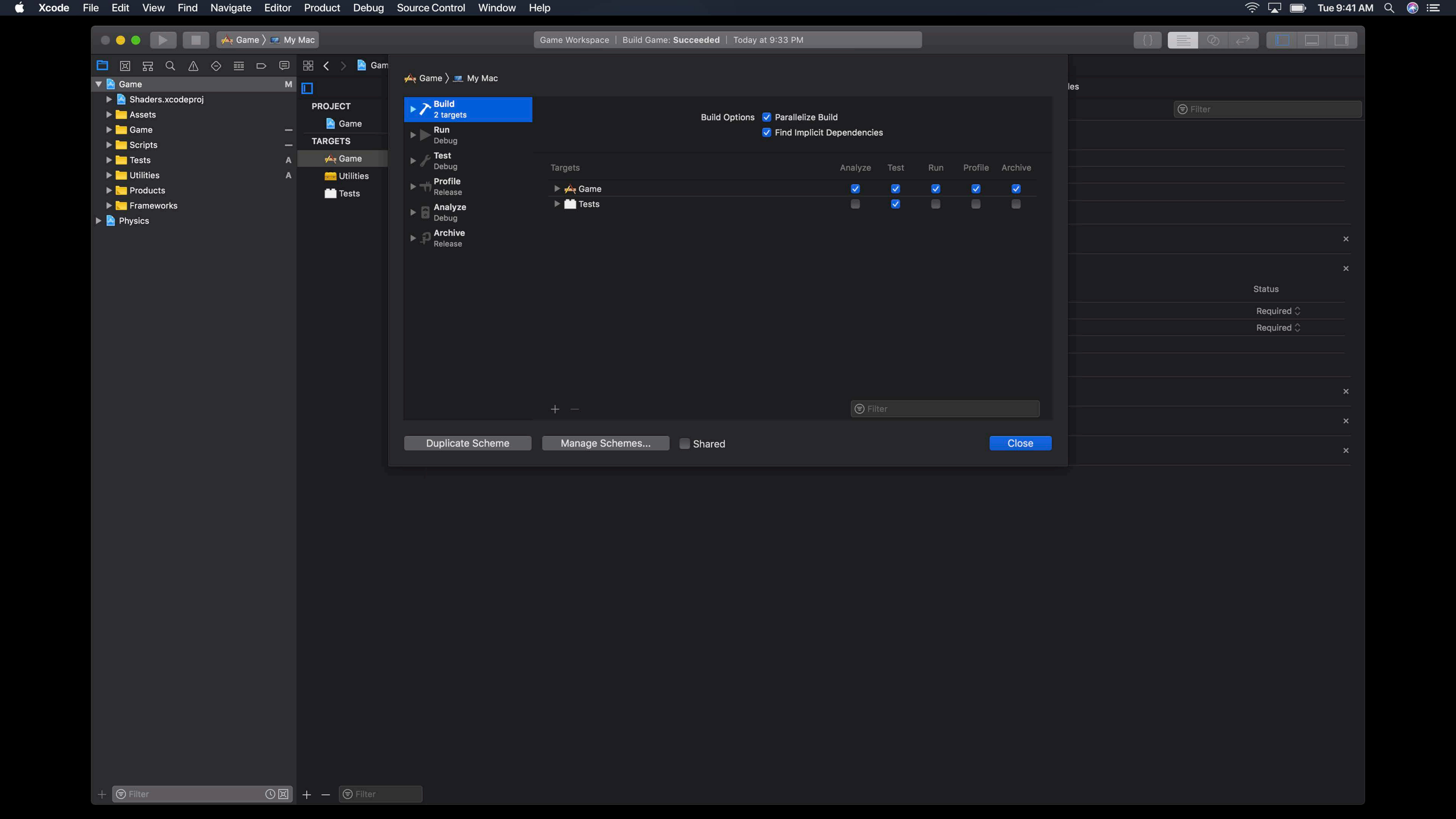


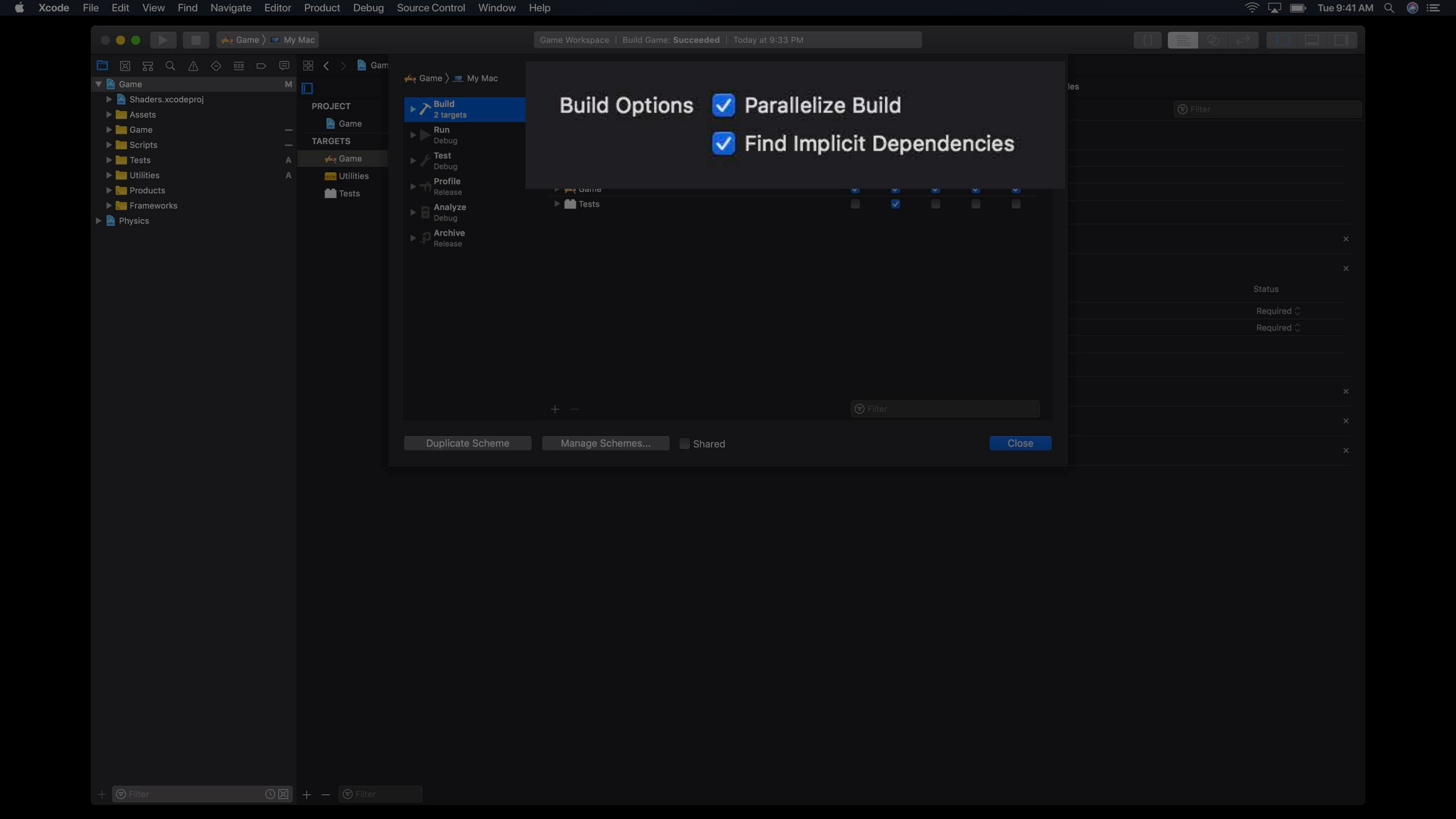
Extreme Parallelization



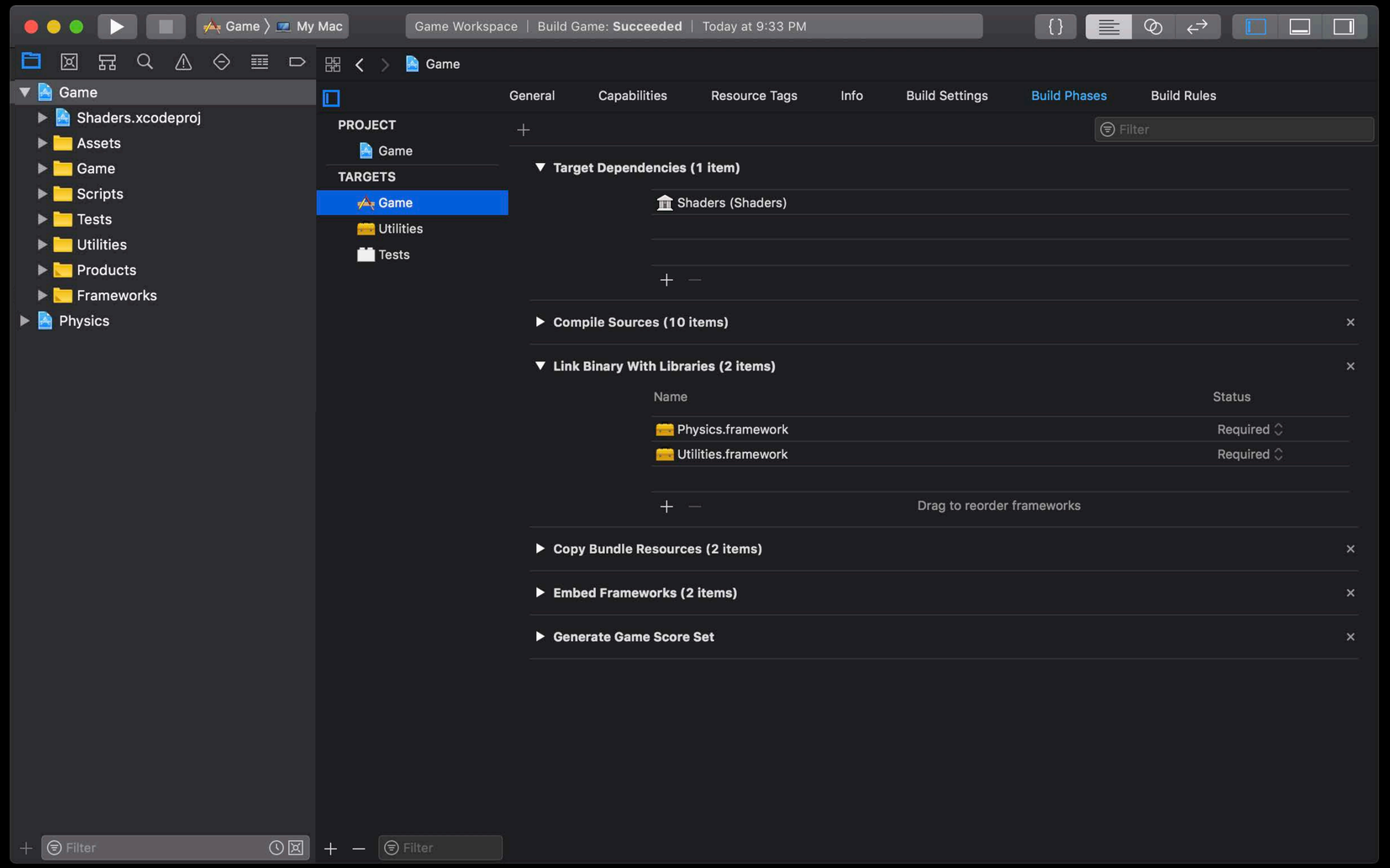
How do we get there?

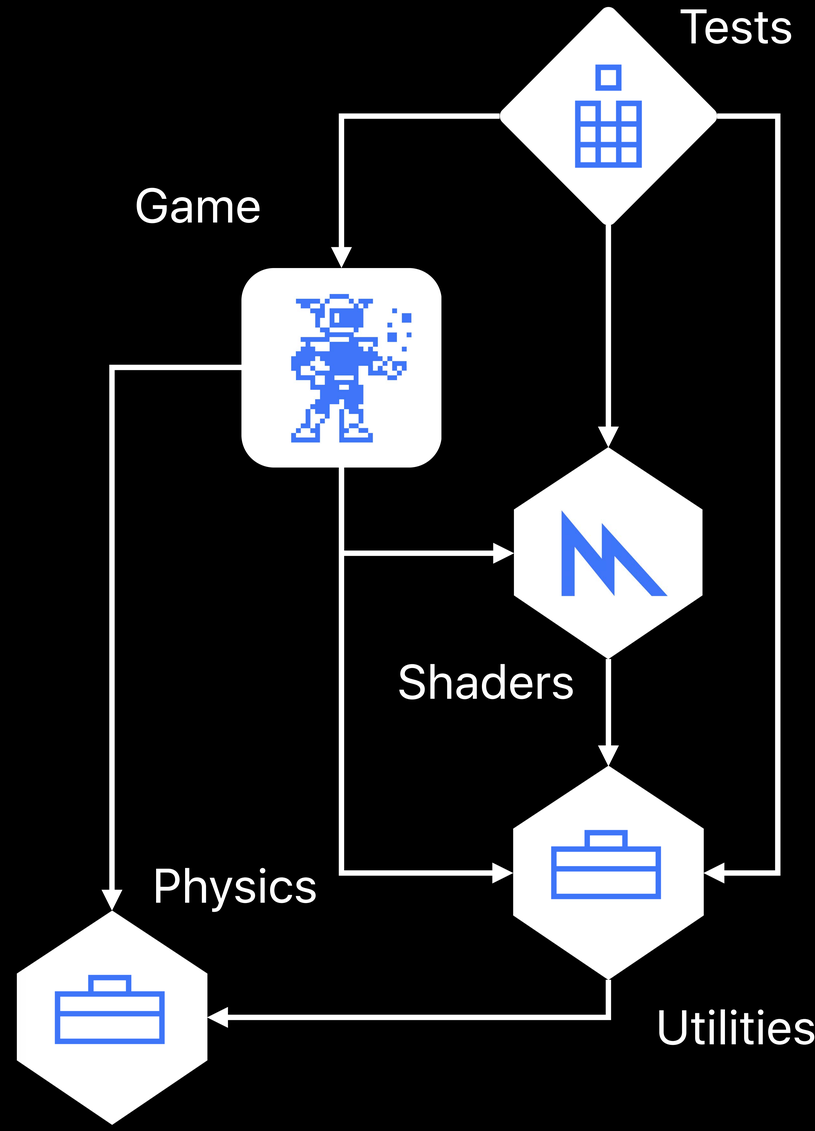

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game
Utilities
Shaders
Algebra
Physics
Calculus
Edit Scheme...
New Scheme...
Manage Schemes...
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Shaders (Shaders)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Name
Status
Physics.framework
Required
Utilities.framework
Required
Drag to reorder frameworks
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Score Set
Game
Shaders.xcod
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Build Options
Parallelize Build
Find Implicit Dependencies
Targets
Analyze
Test
Run
Profile
Archive
Game
Tests
Filter
Duplicate Scheme
Manage Schemes...
Shared
Close
Status
Required
Required

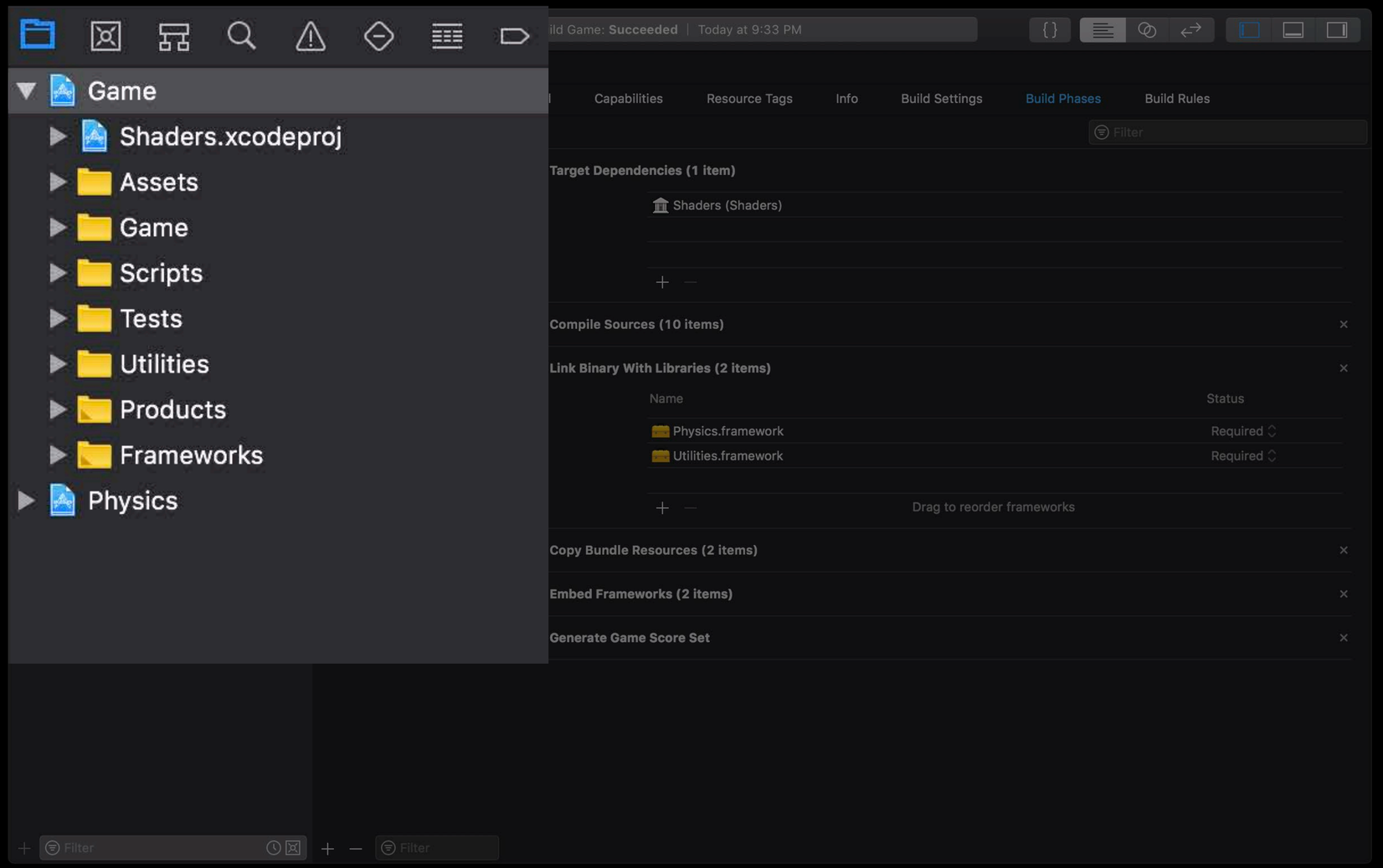

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
Build
2 targets
Run
Debug
Test
Debug
Profile
Release
Analyze
Debug
Archive
Release
Build Options
Parallelize Build
Find Implicit Dependencies
Tests
Filter
Duplicate Scheme
Manage Schemes...
Shared
Close
Status
Required
Required

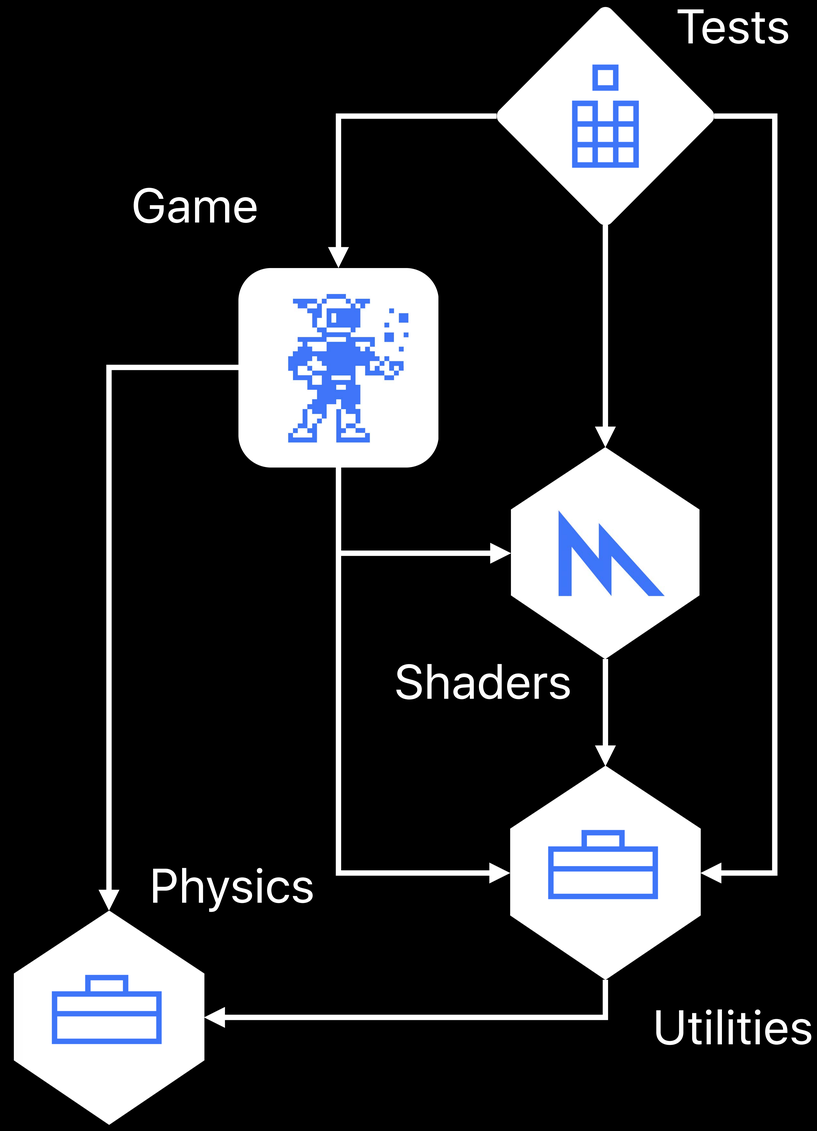
Project Configuration



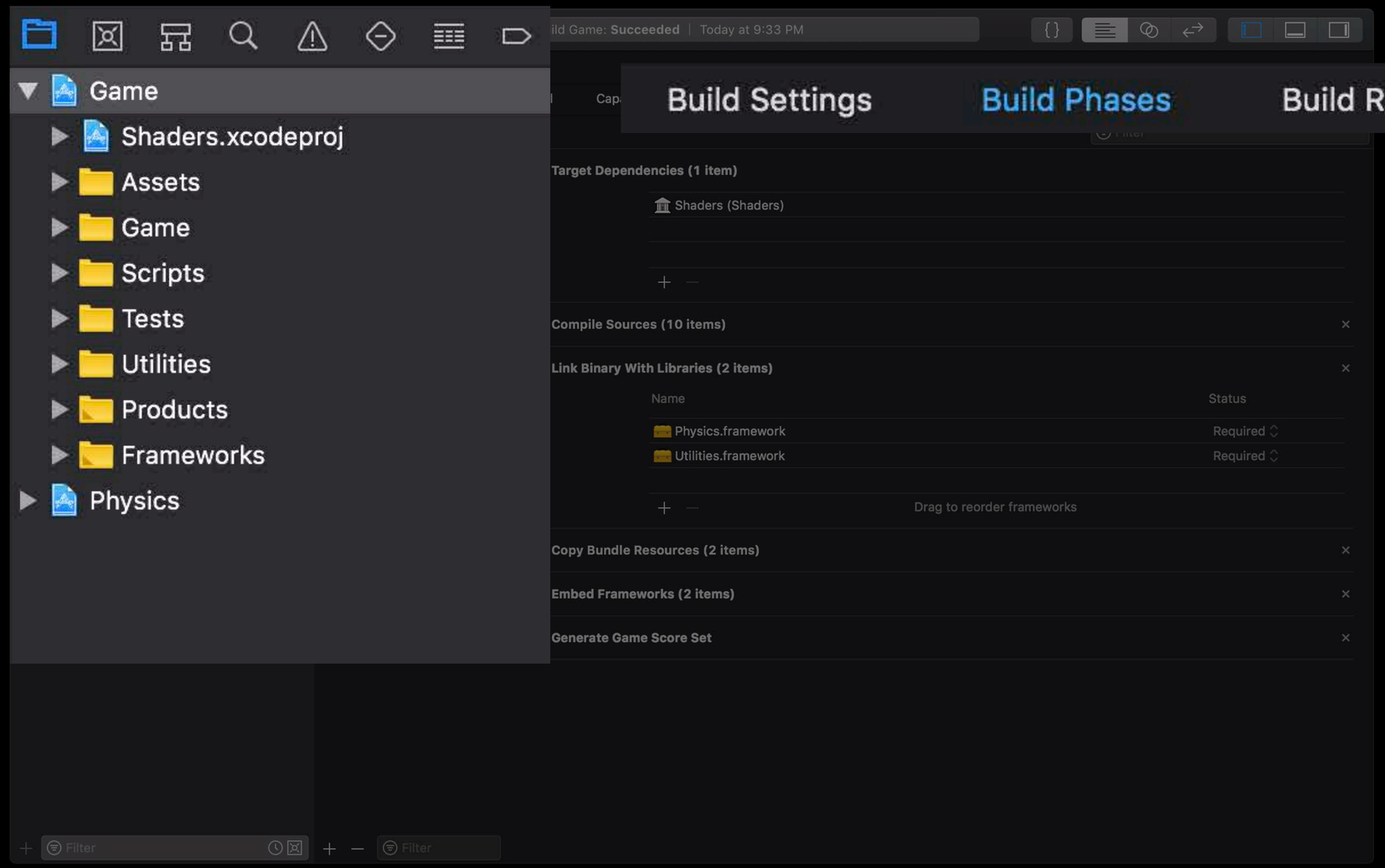


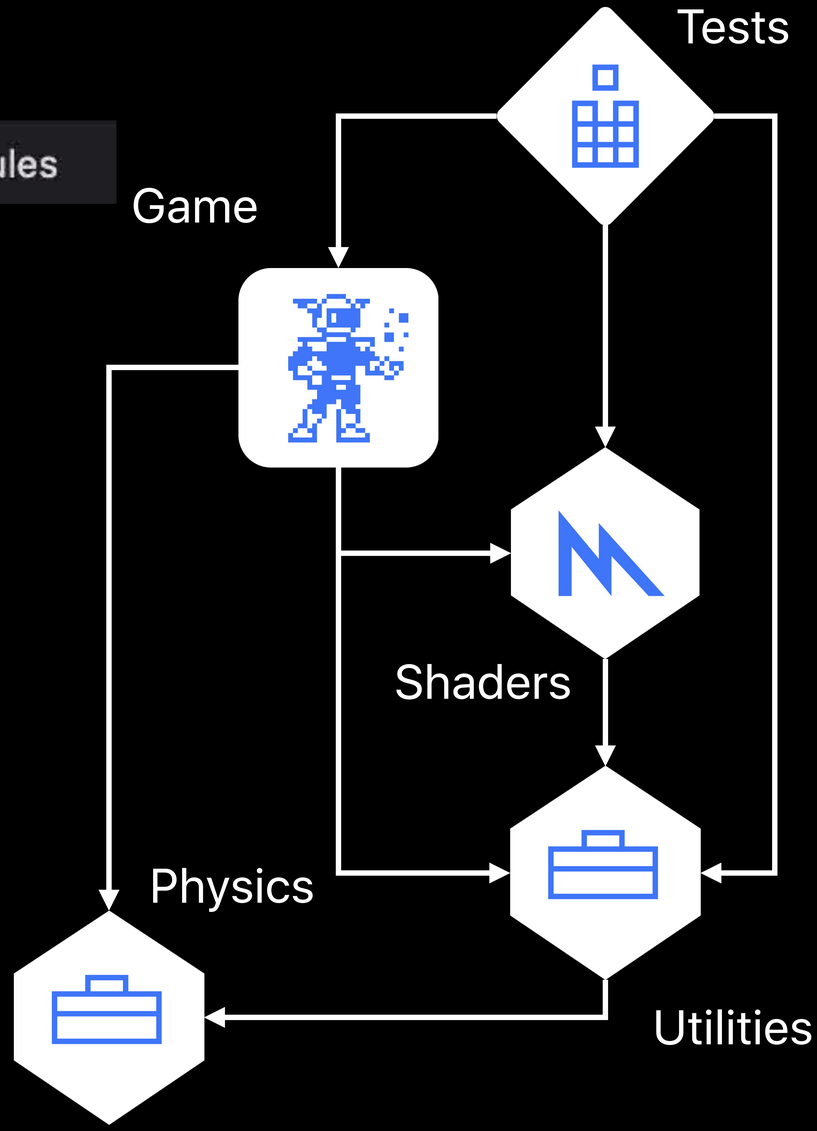
Project Configuration



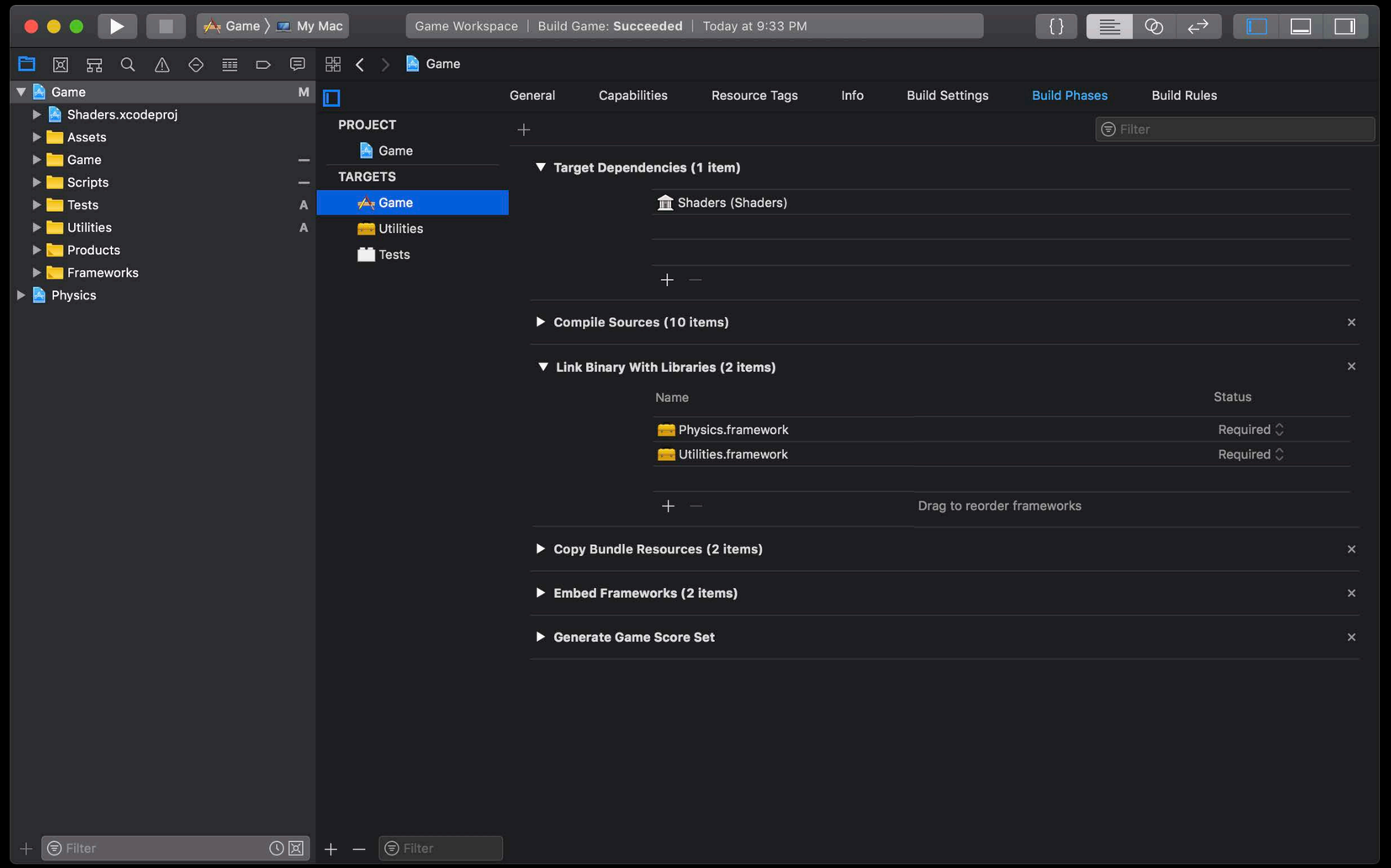


Project Configuration



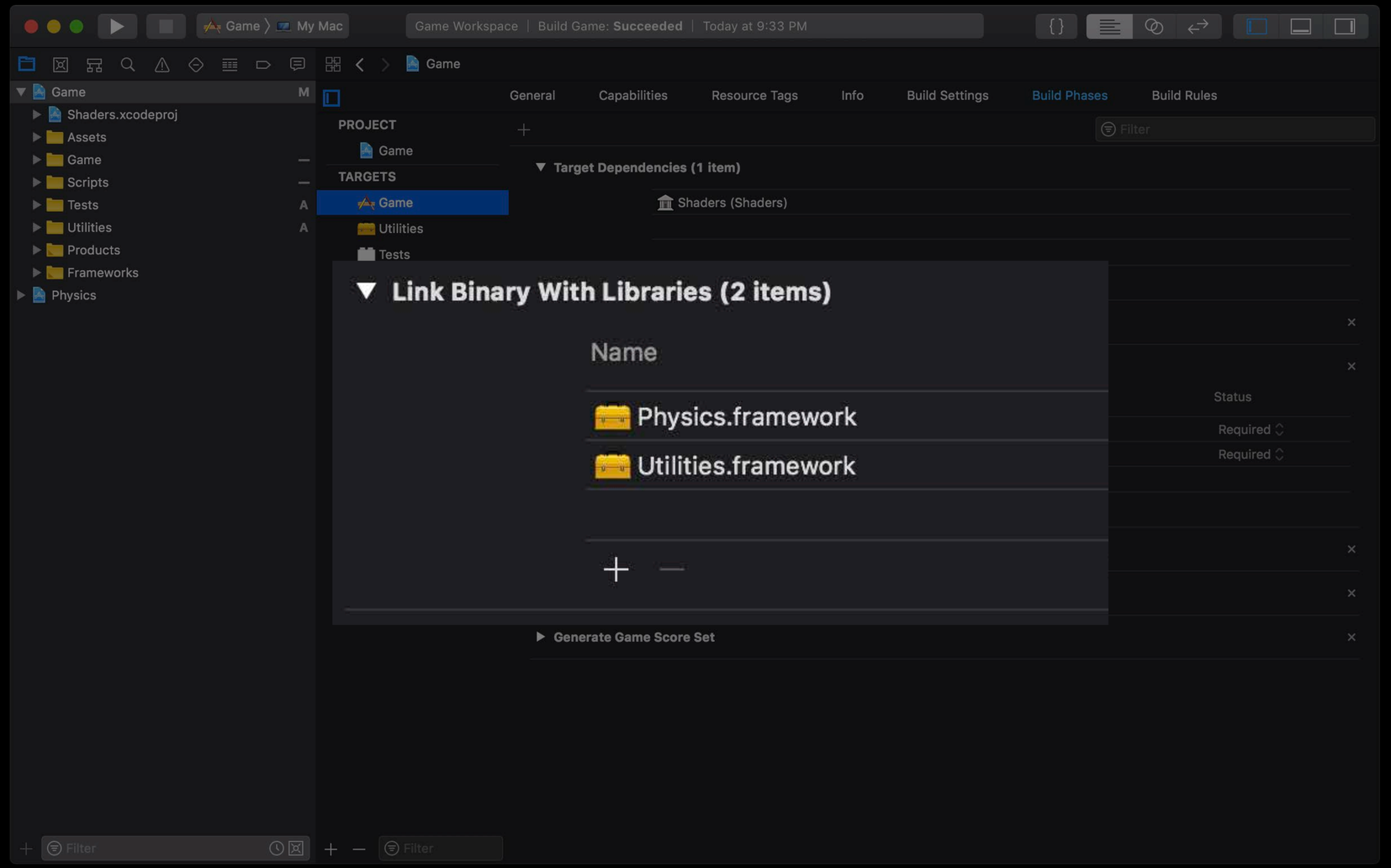


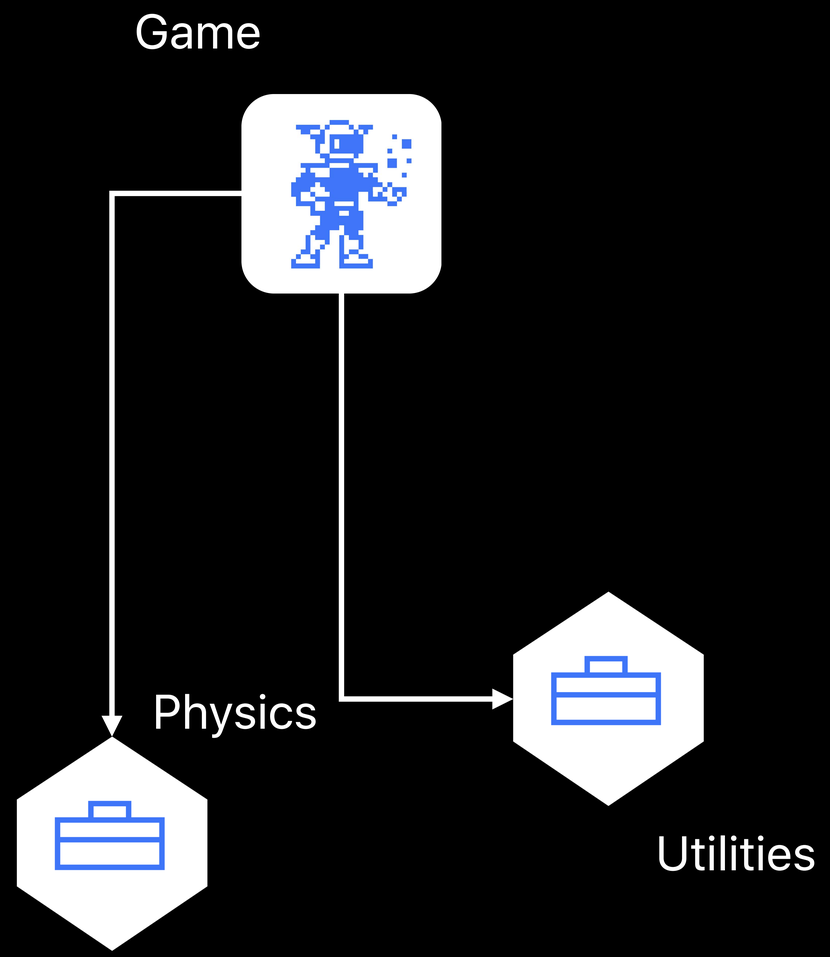
Game Target Dependencies



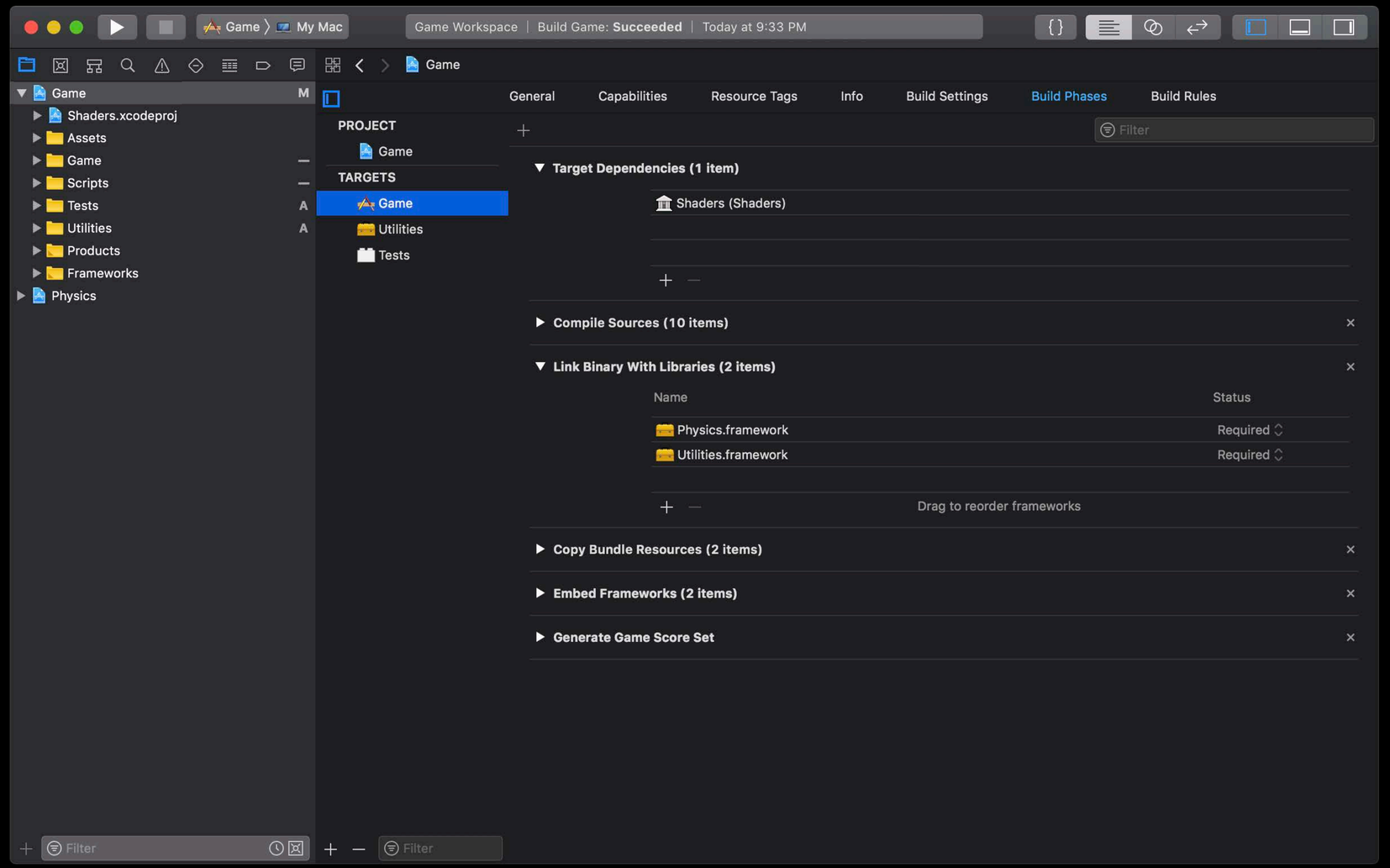
Game

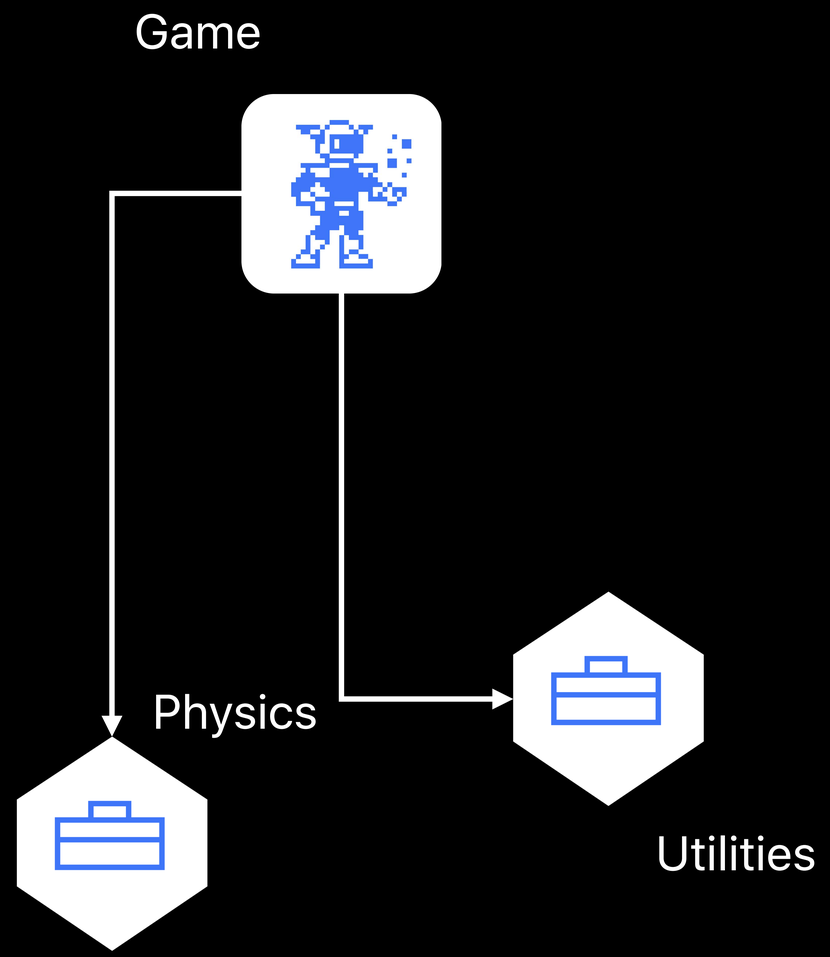
Game Target Dependencies



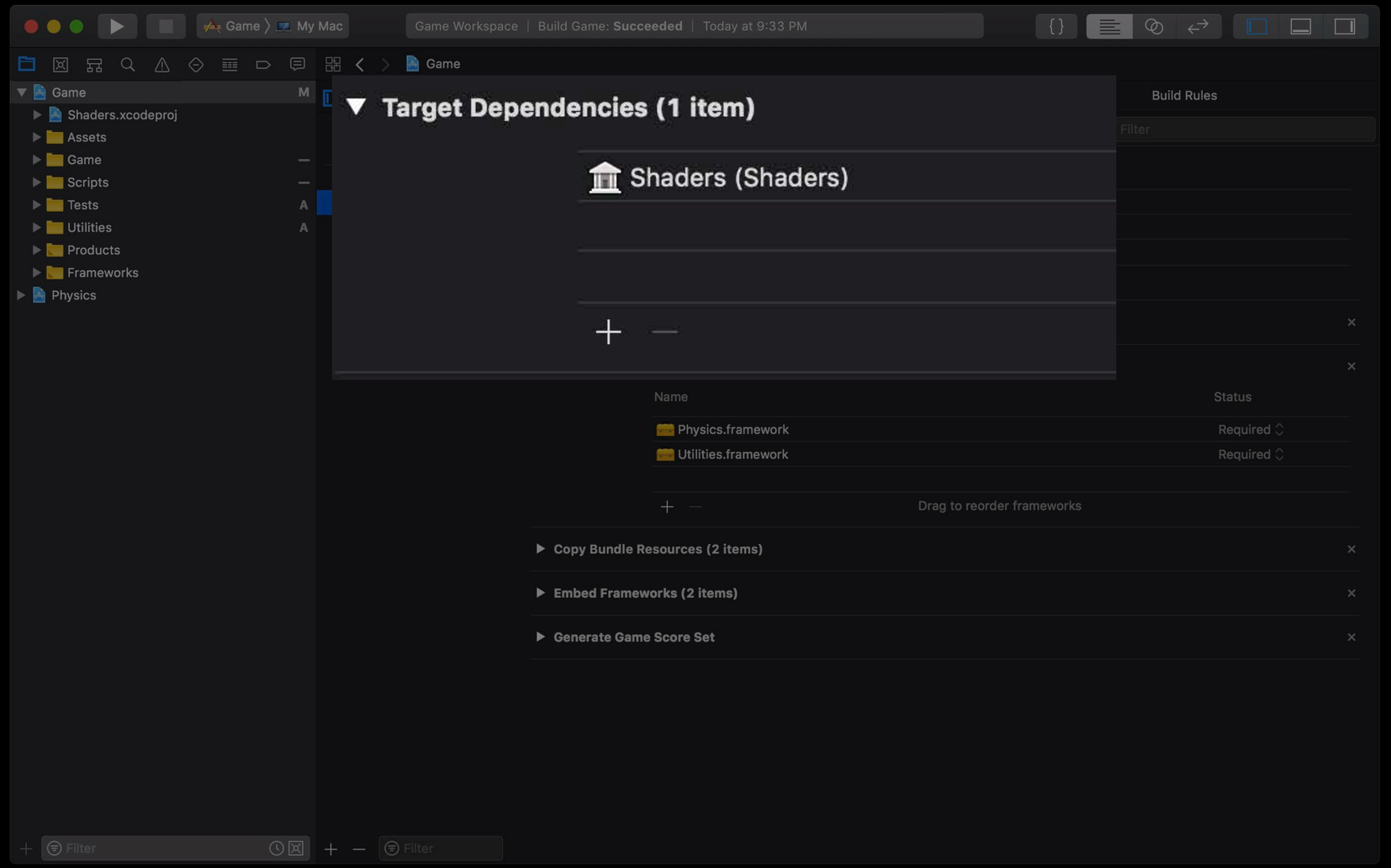


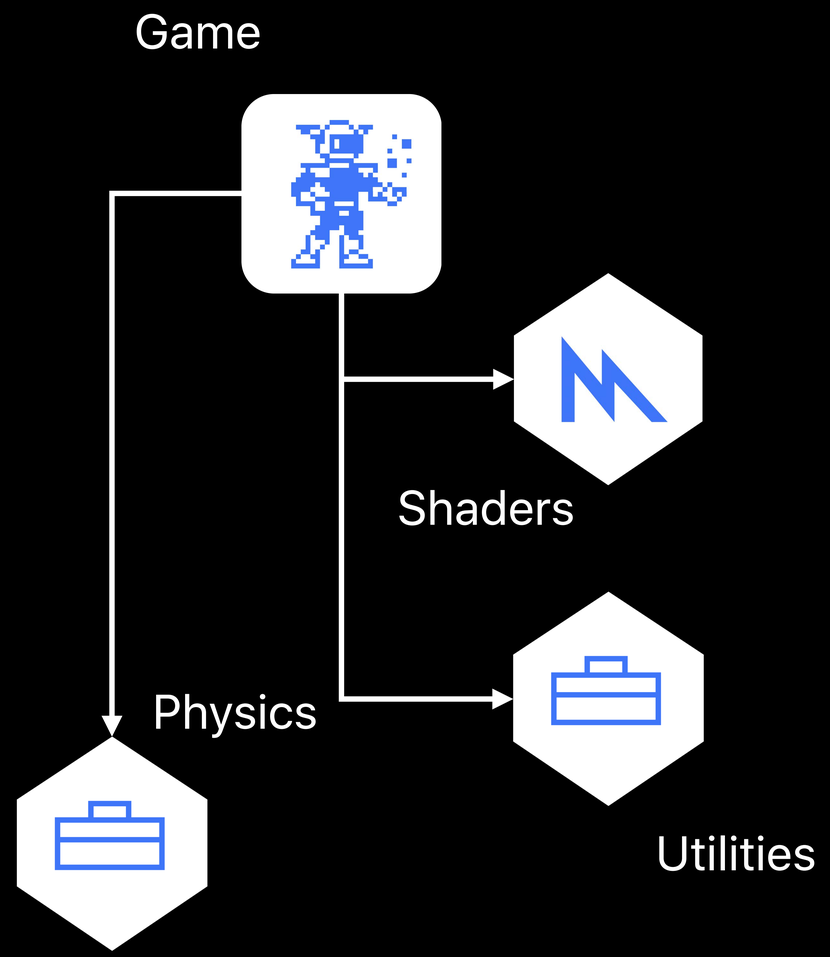
Game Target Dependencies



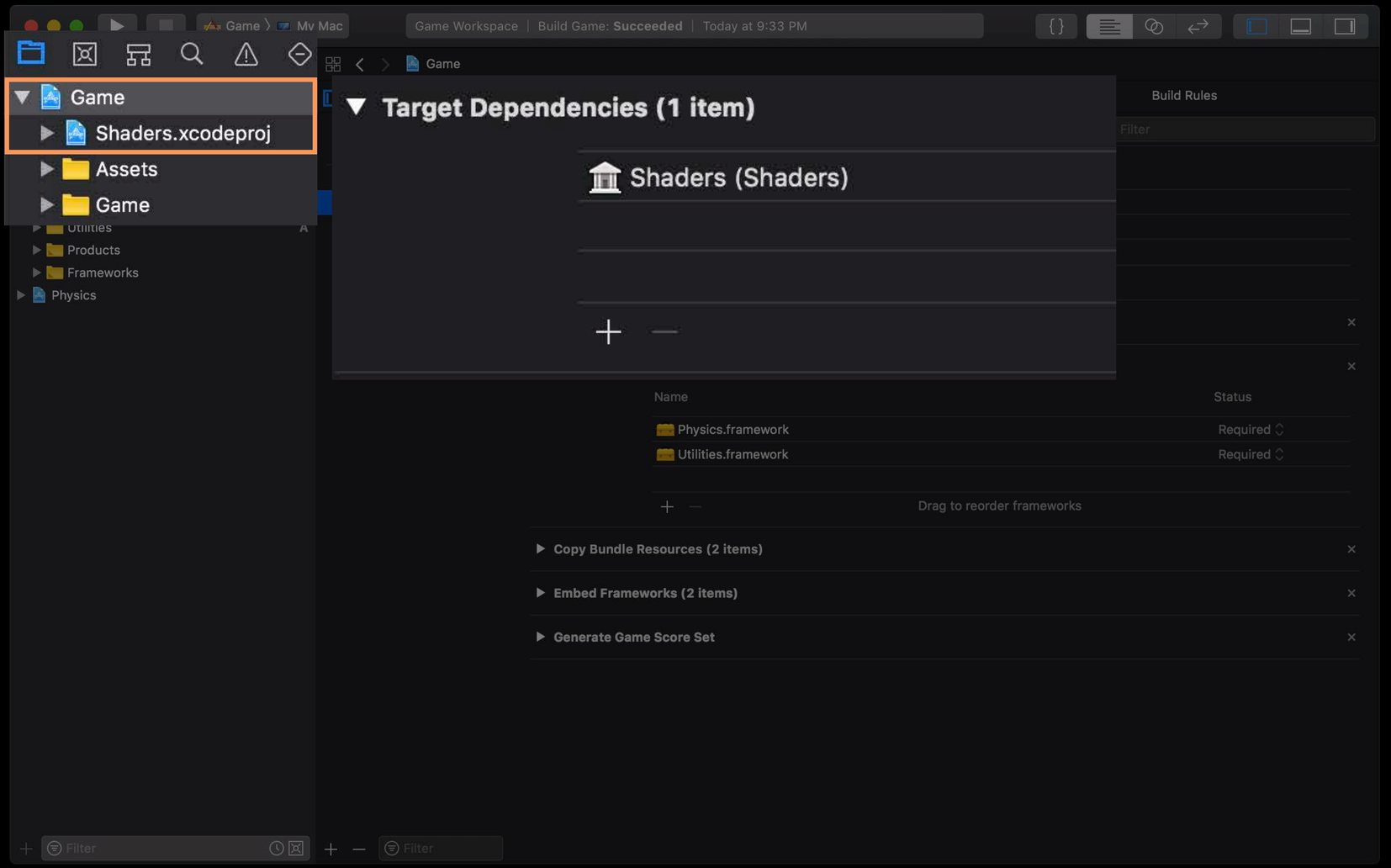


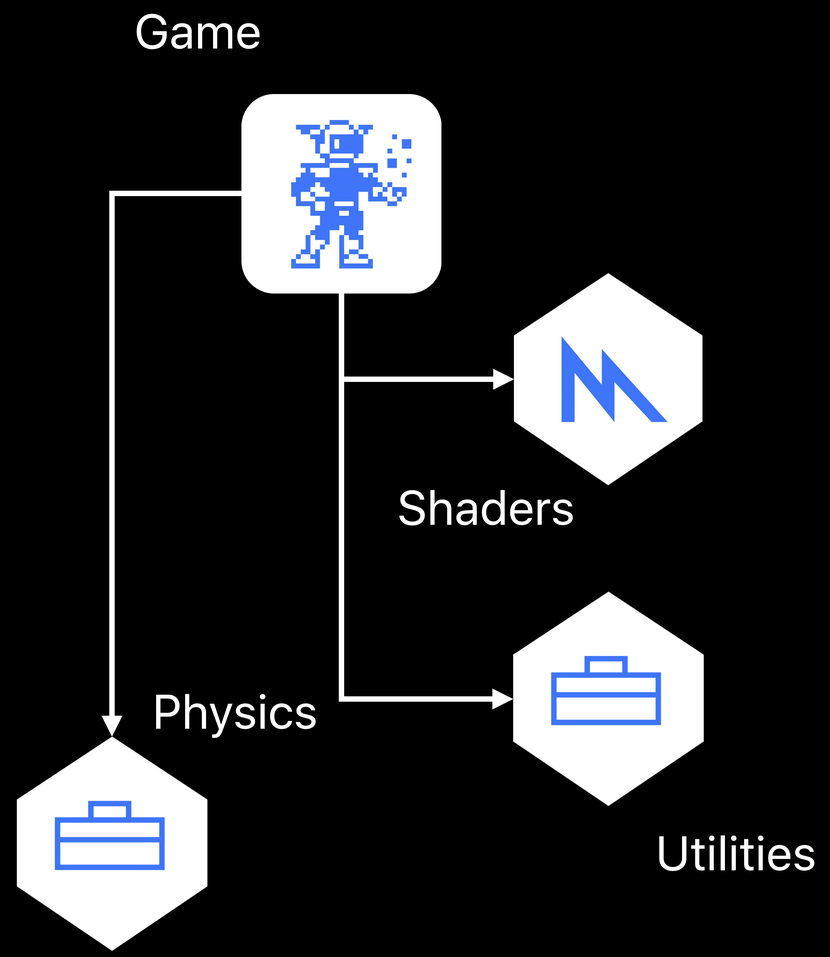
Game Target Dependencies



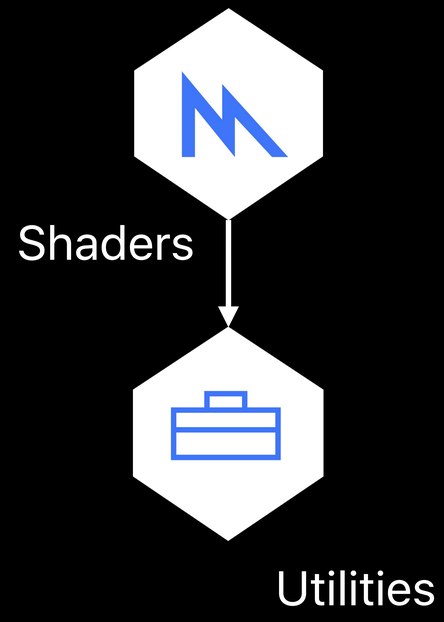


Game Target Dependencies



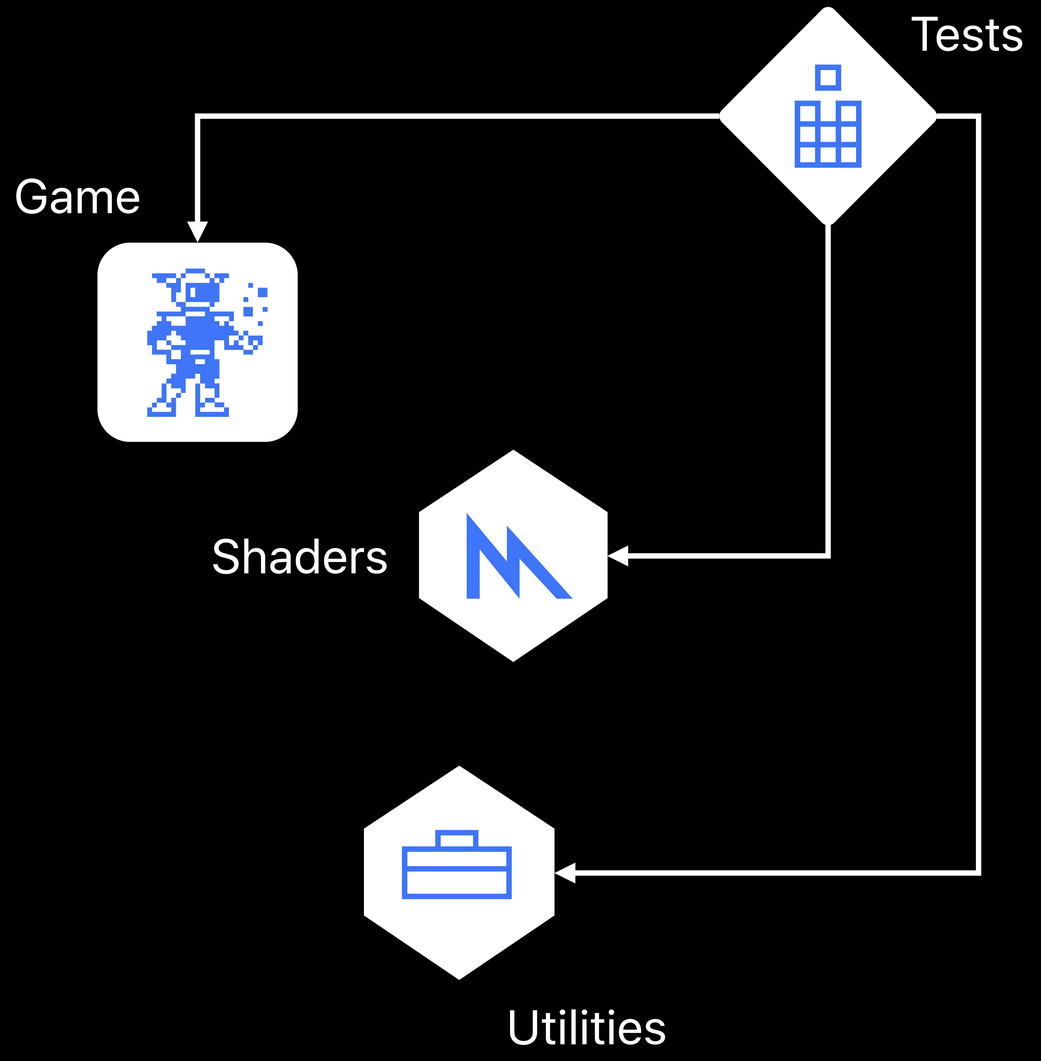


Shaders Target Dependencies

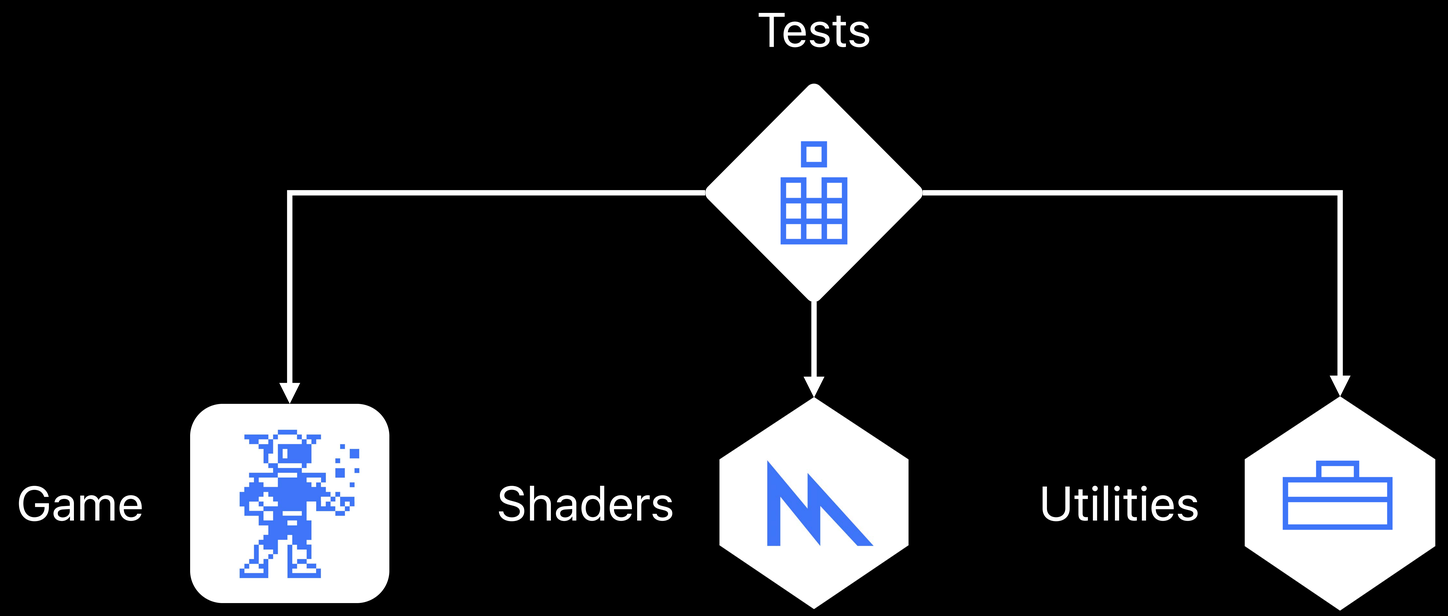


Utilities Target Dependencies

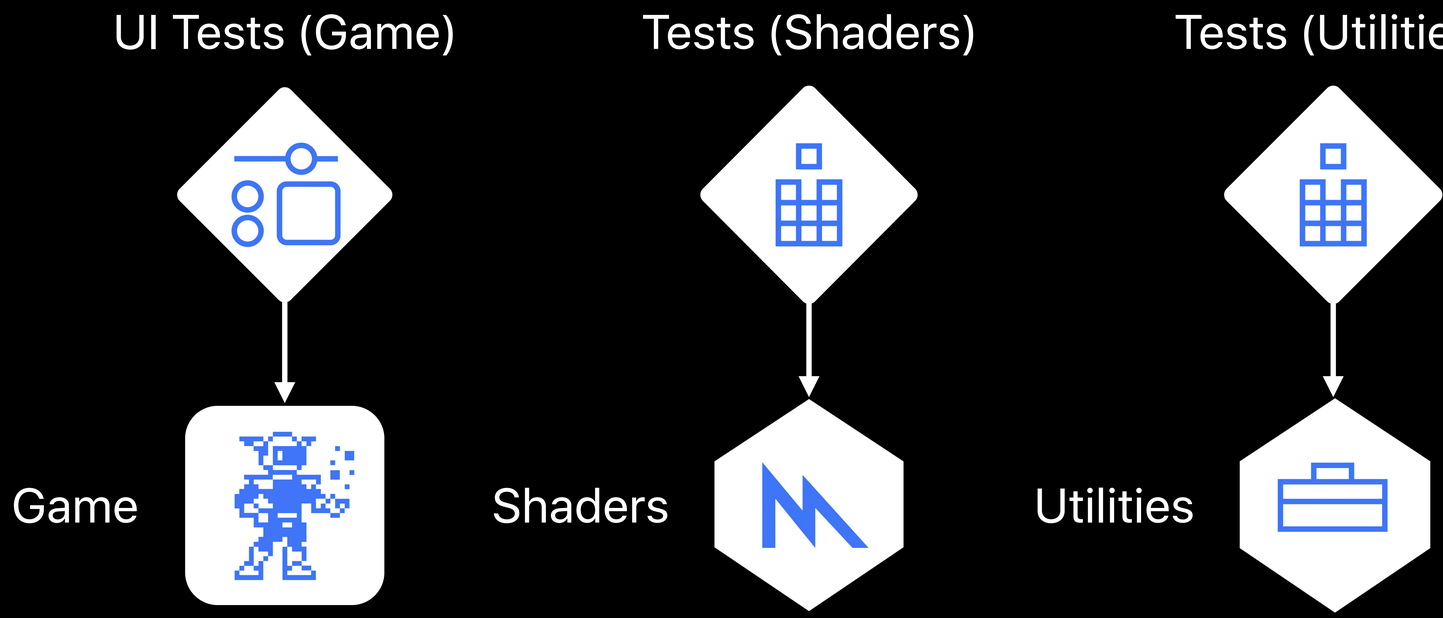
Tests Target Dependencies



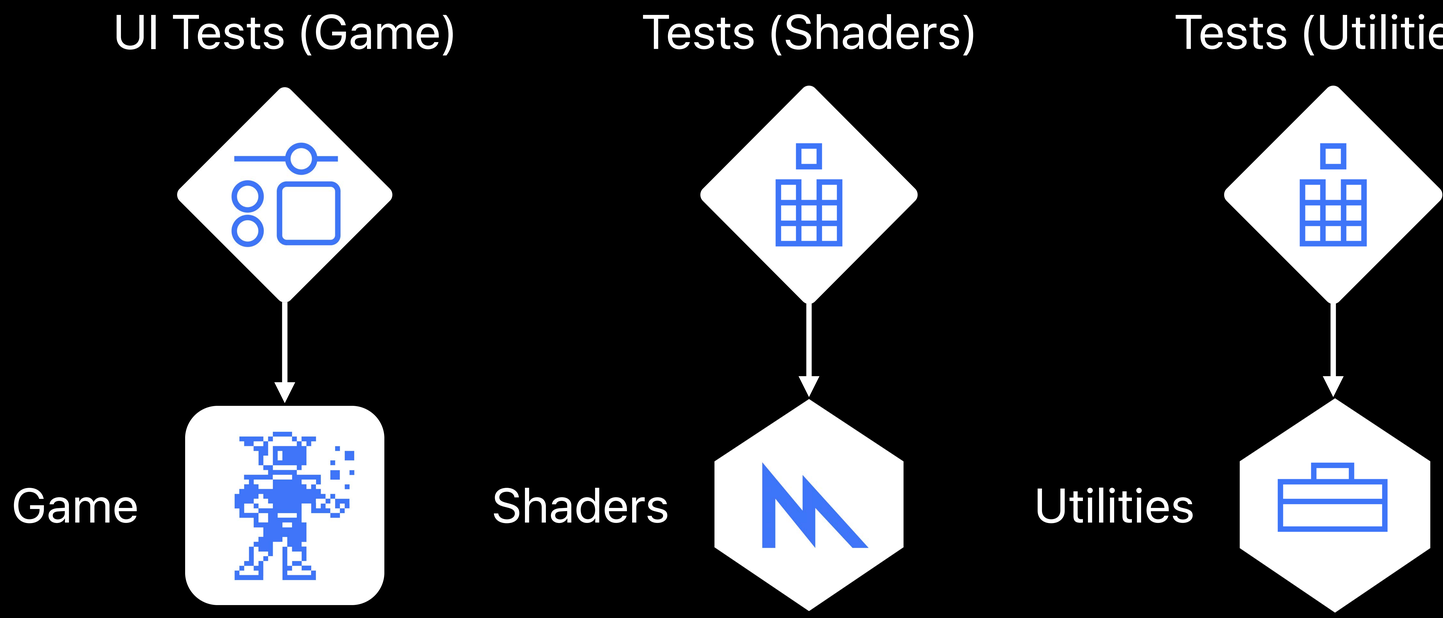
Examine Tests Dependencies



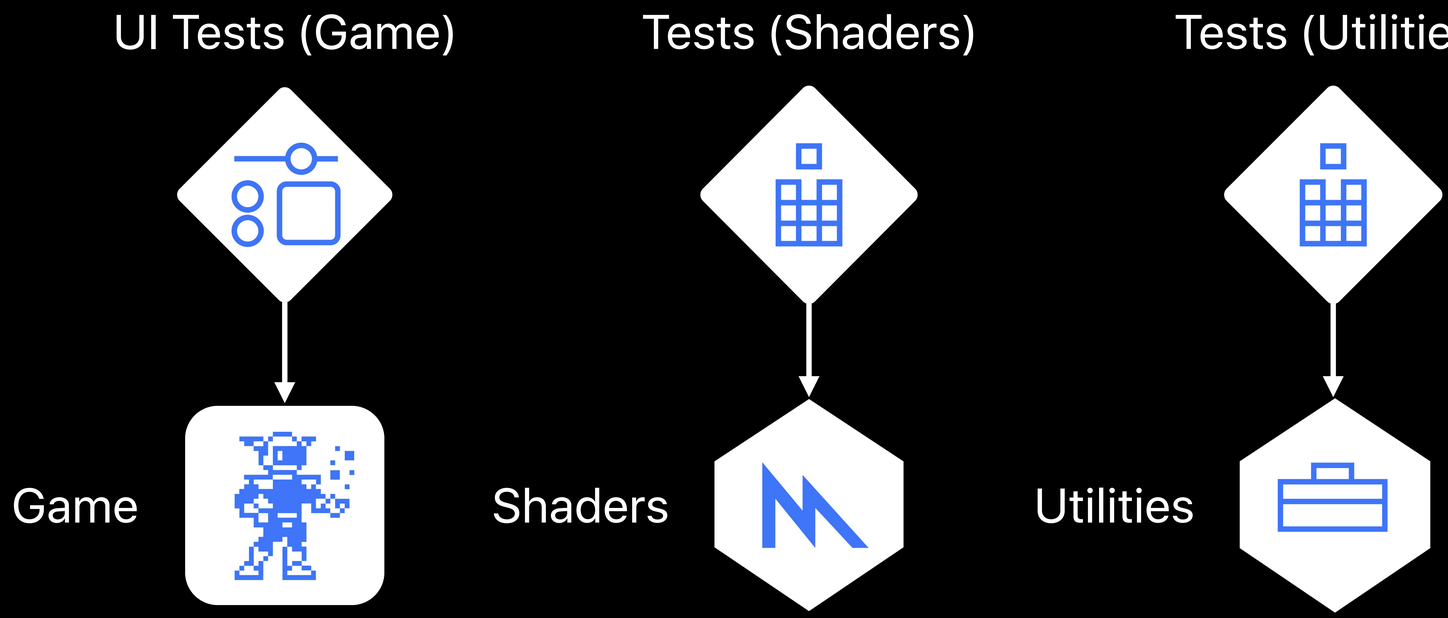
Examine Tests Dependencies



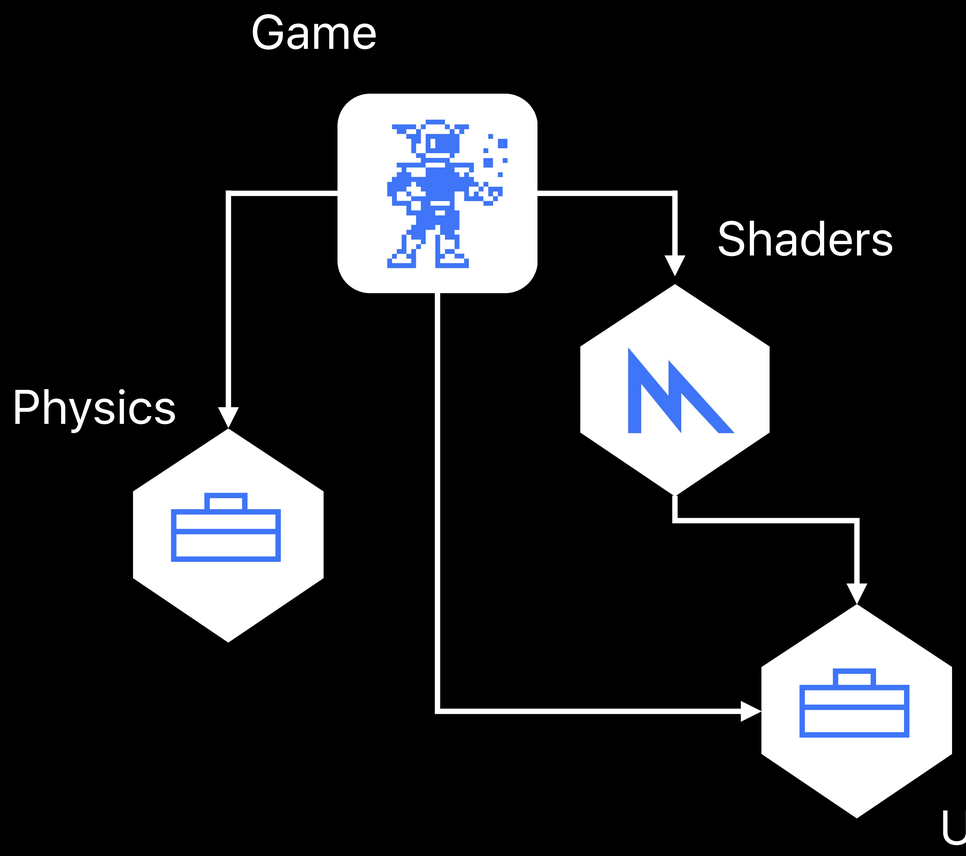
Examine Tests Dependencies



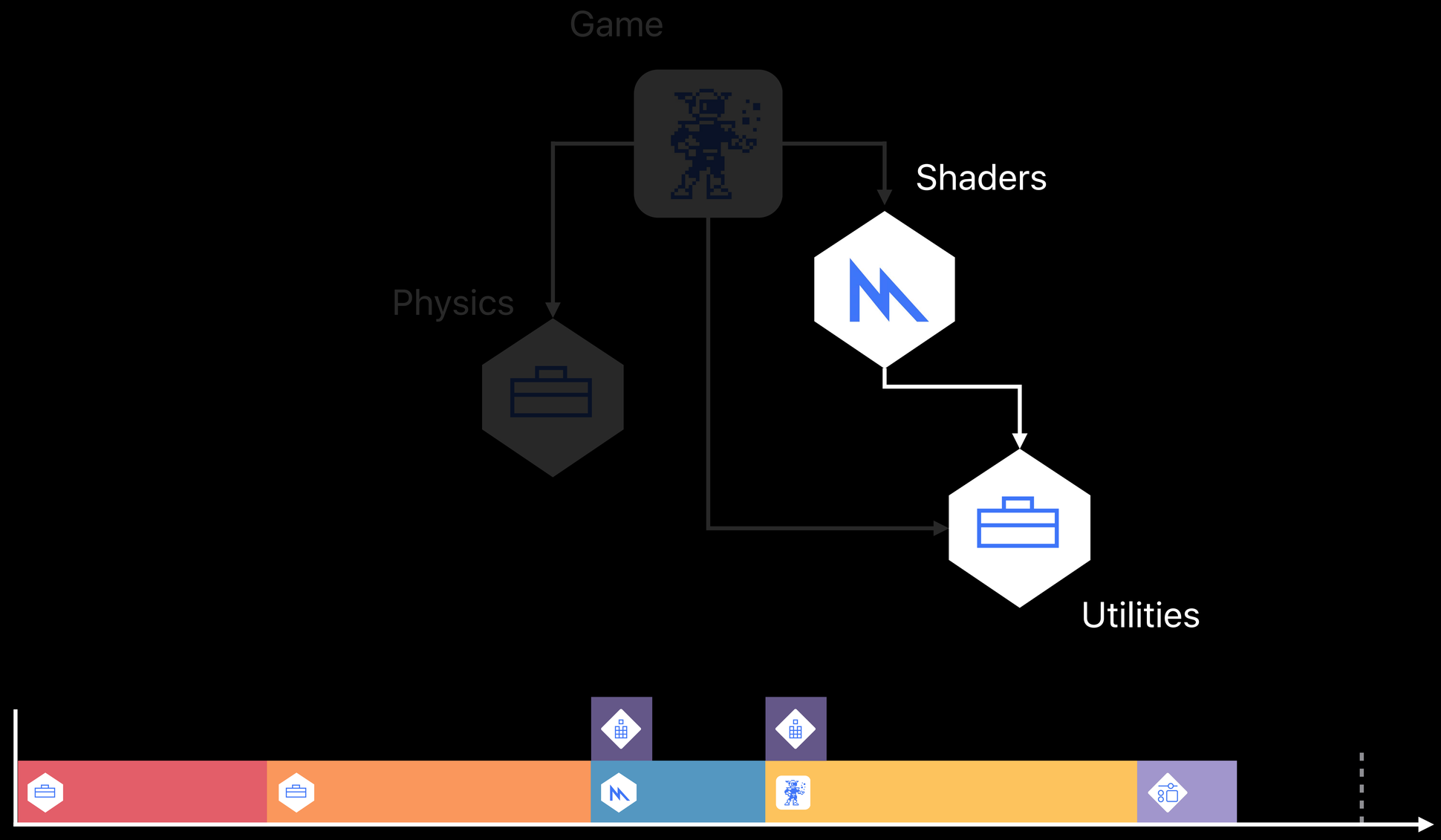
Examine Tests Dependencies



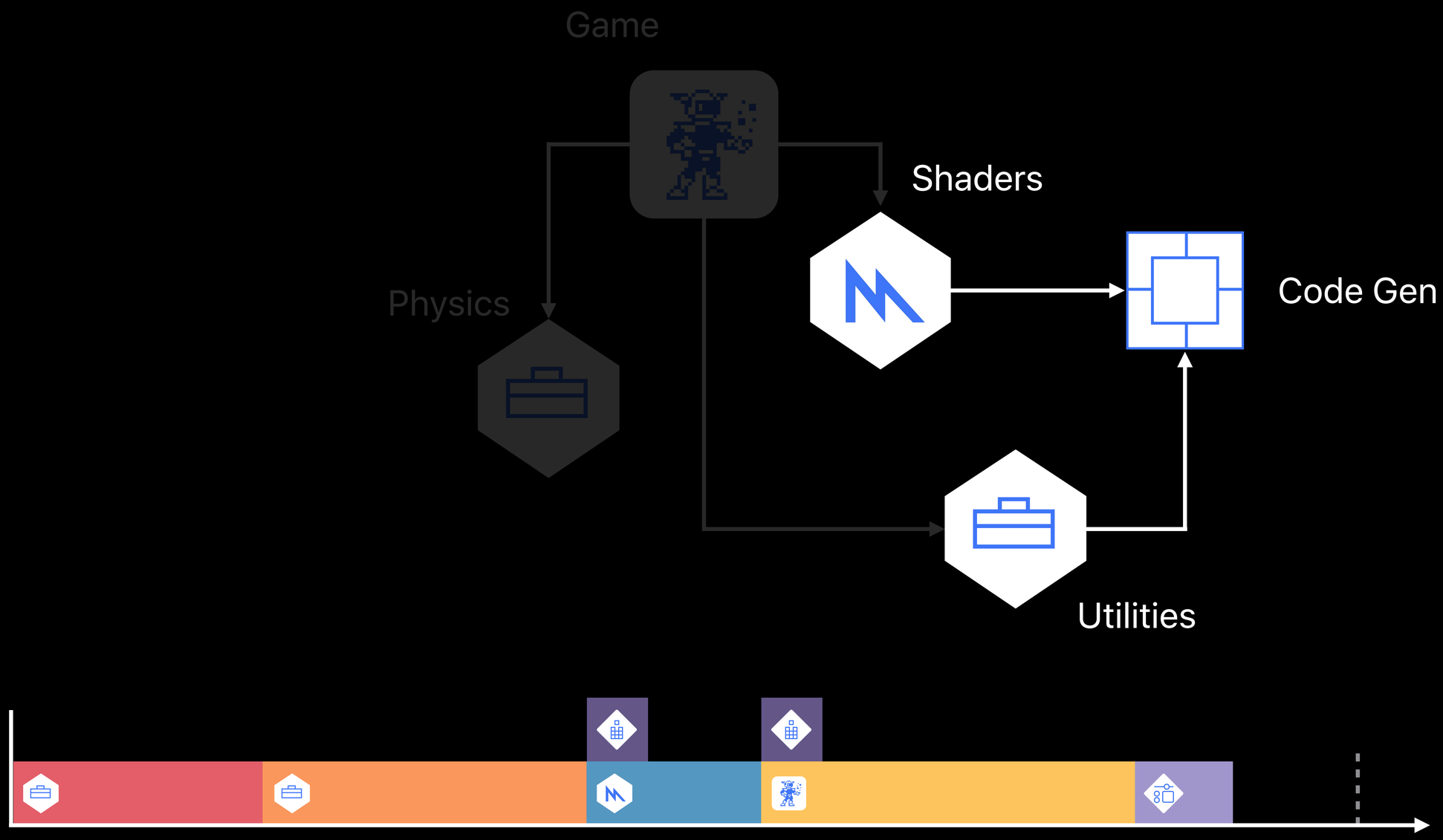
Reduce Dependency Exposure



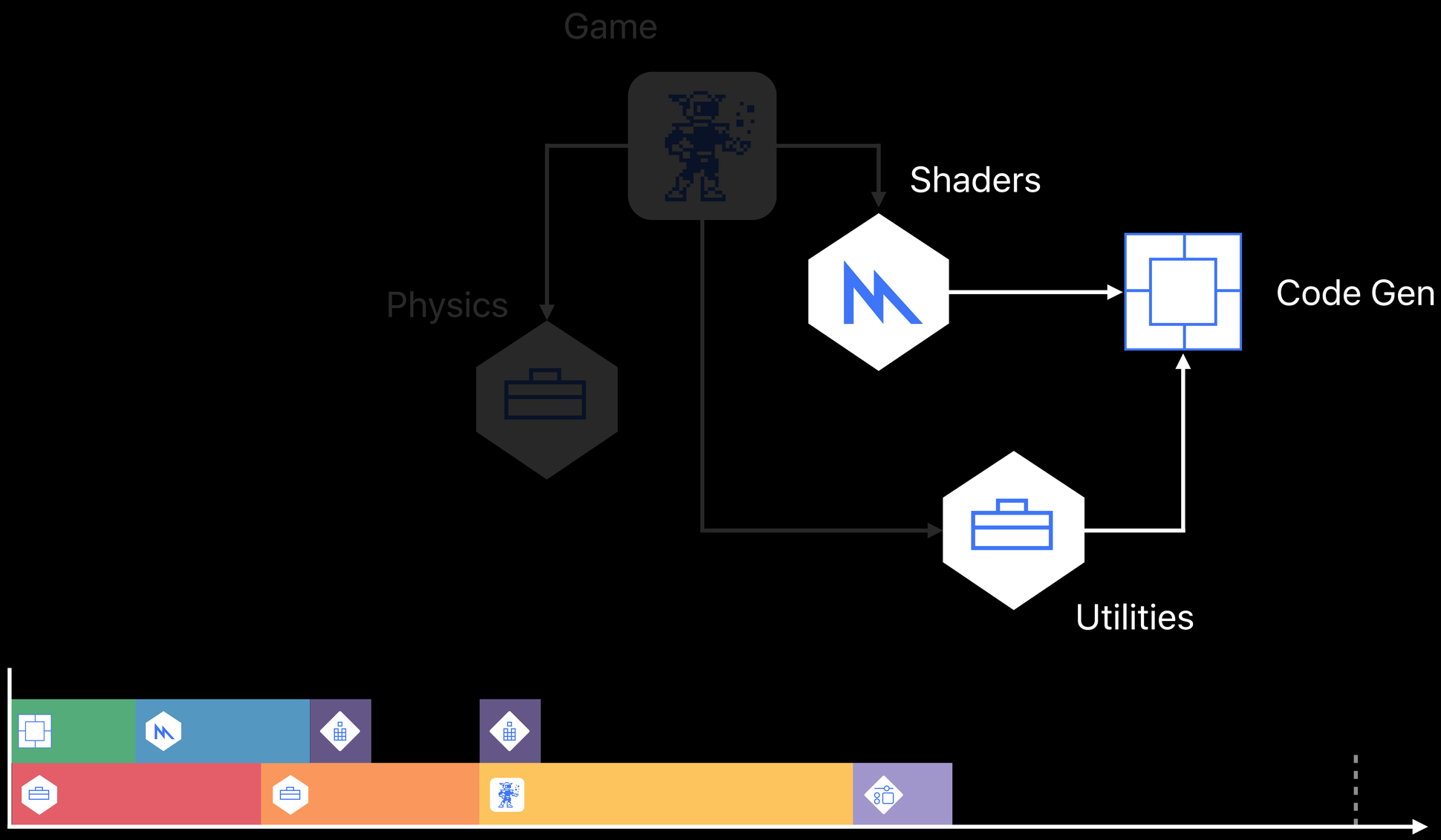
Reduce Dependency Exposure



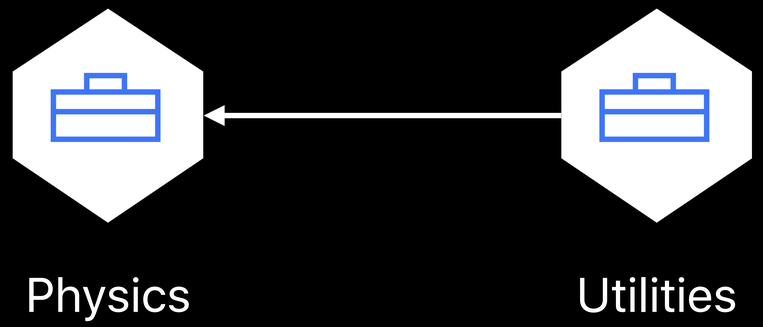
Reduce Dependency Exposure



Reduce Dependency Exposure



Examine Unused Dependencies



Examine Unused Dependencies

Physics

Utilities

Examine Unused Dependencies

Physics

Utilities

Target Build Process

Parallelized Target Build Process

NEW

Parallelized Target Build Process

NEW

Parallelized Target Build Process

NEW

Compile sources can start earlier

Parallelized Target Build Process

NEW

Compile sources can start earlier

Waits only for what it needs

Parallelized Target Build Process

NEW

Compile sources can start earlier

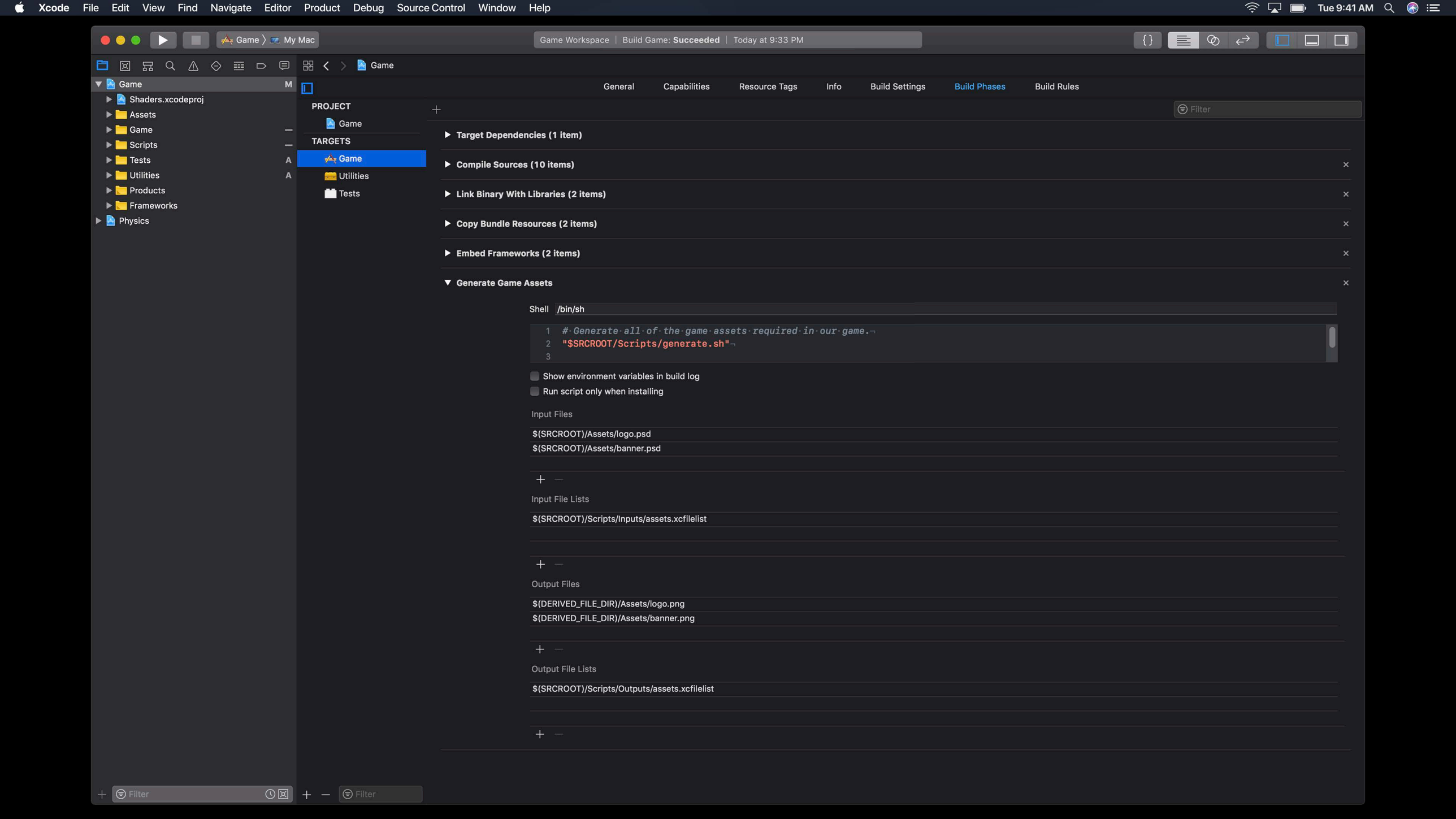
Waits only for what it needs

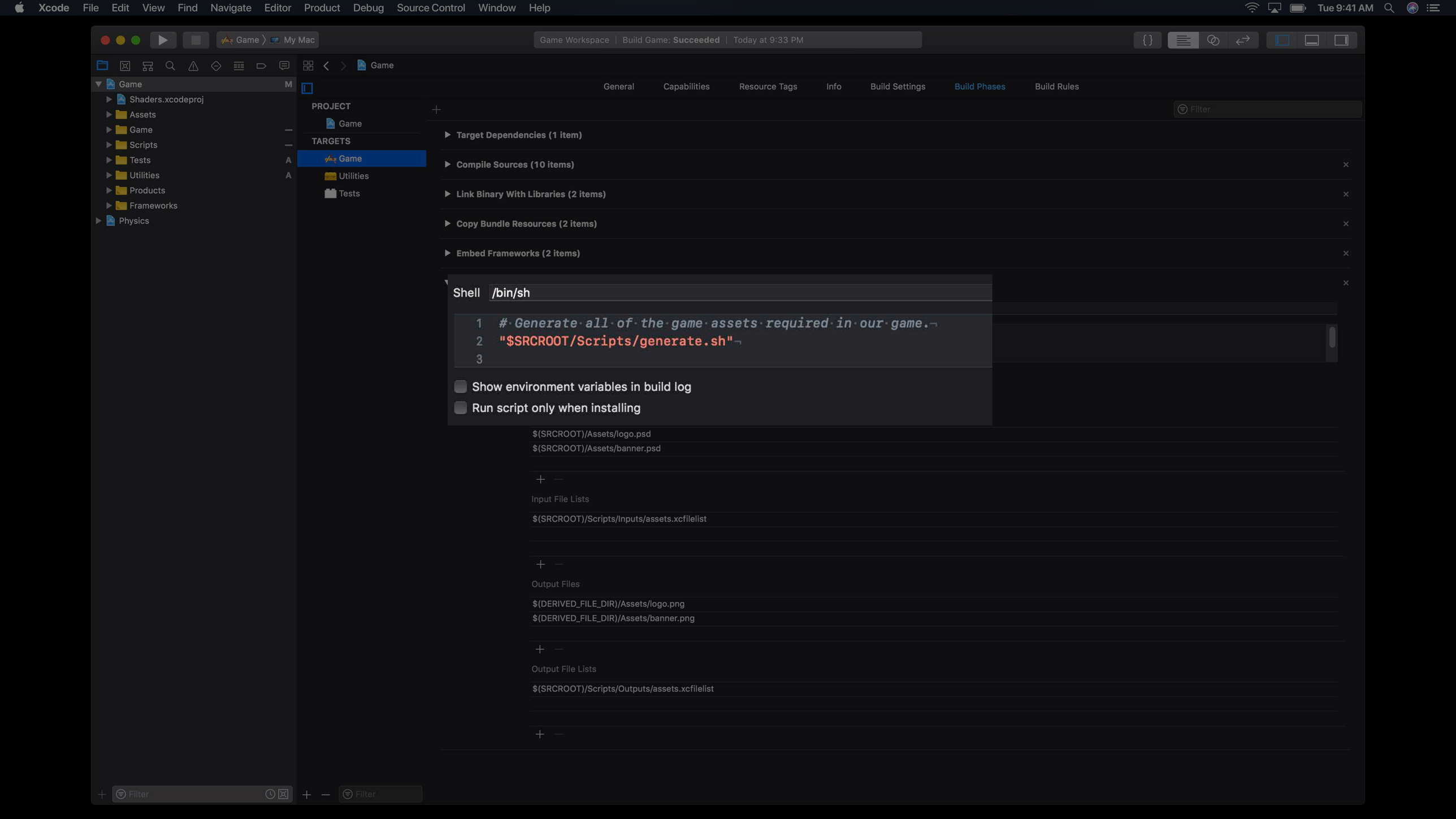
Must wait for Run Script phases

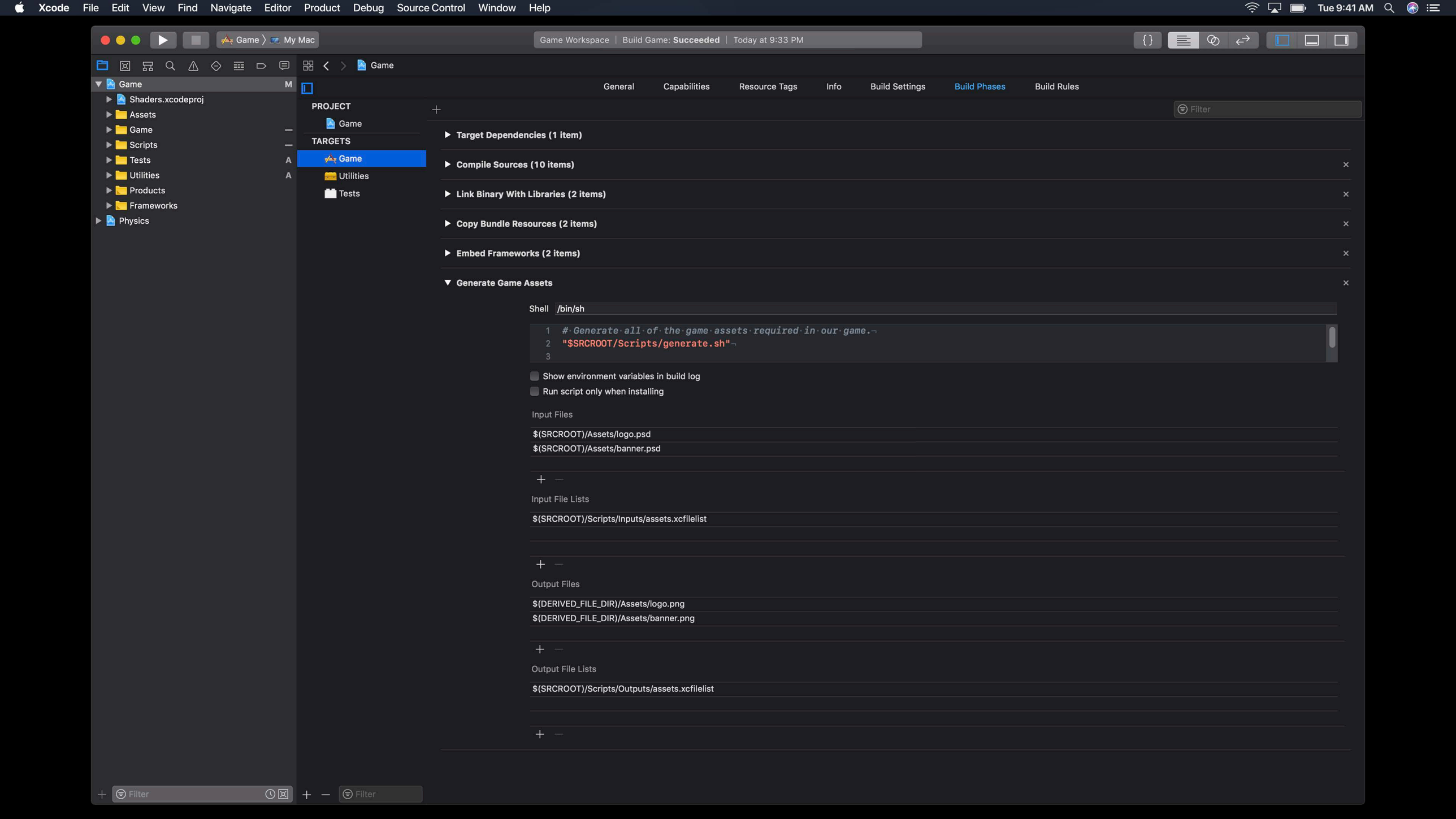
Run Script Phases

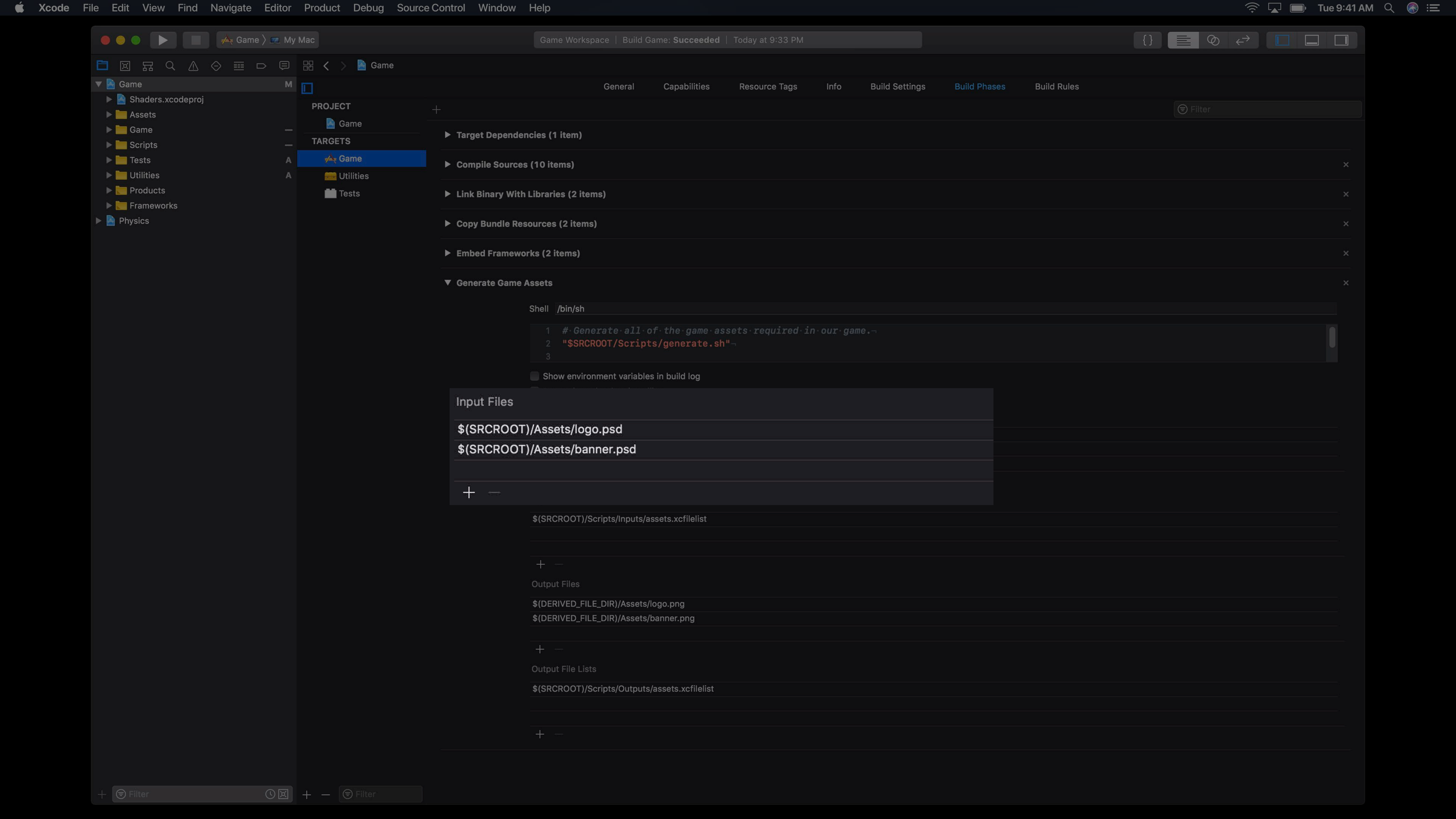
Reducing the work on rebuilds

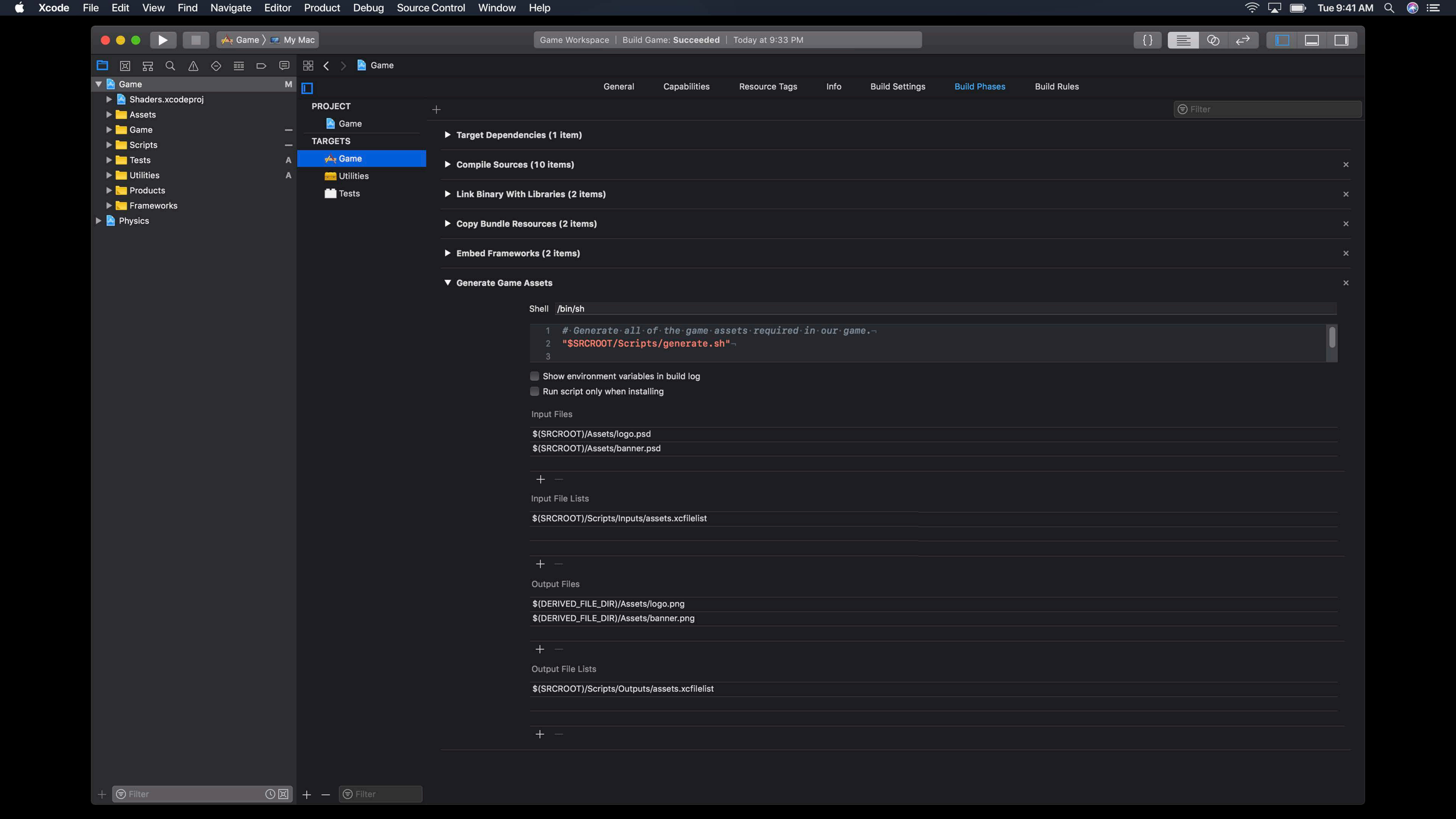
Allows you to customize your build process for your exact needs.

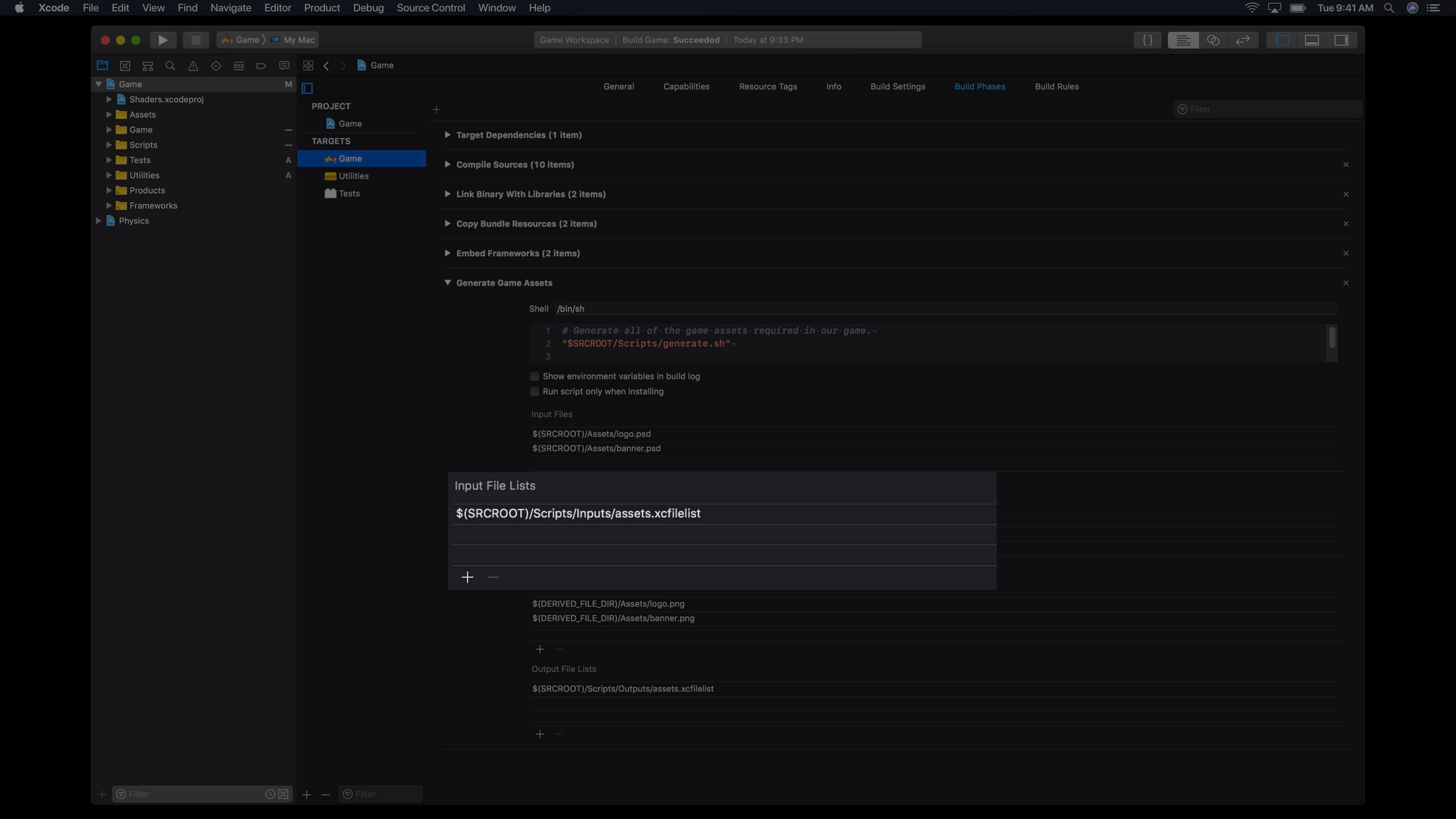

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Game M
Shaders.xcodeproj
Assets
Game
Scripts
Tests A
Utilities A
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
1 # Generate all of the game assets required in our game.
2 "$SRCROOT/Scripts/generate.sh"
3
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist
Filter
Filter
Filter


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Shell /bin/sh
1 #·Generate·all·of·the·game·assets·required·in·our·game.¬
2 "$SRCROOT/Scripts/generate.sh"¬
3
Show environment variables in build log
Run script only when installing
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Game M
Shaders.xcodeproj
Assets
Game
Scripts
Tests A
Utilities A
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
1 # Generate all of the game assets required in our game.
2 "$SRCROOT/Scripts/generate.sh"
3
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist
Filter
Filter
Filter


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
Generate all of the game assets required in our game.
"$SRCROOT/Scripts/generate.sh"
Show environment variables in build log
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Game M
Shaders.xcodeproj
Assets
Game
Scripts
Tests A
Utilities A
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
1 # Generate all of the game assets required in our game.
2 "$SRCROOT/Scripts/generate.sh"
3
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist
Filter
Filter
Filter


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
1 # Generate all of the game assets required in our game.
2 "$SRCROOT/Scripts/generate.sh"
3
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist

File Lists

```
#  assets.xcfilelist
#  Game

# Asset masters
$(SRCROOT)/Assets/game.psd
$(SRCROOT)/Assets/player.psd
$(SRCROOT)/Assets/hud.psd
```

File Lists

Newline separated

```
#  assets.xcfilelist
#  Game

# Asset masters
$(SRCROOT)/Assets/game.psd
$(SRCROOT)/Assets/player.psd
$(SRCROOT)/Assets/hud.psd
```

File Lists

Newline separated

Support for build setting variables

```
#  assets.xcfilelist
#  Game

# Asset masters
$(SRCROOT)/Assets/game.psd
$(SRCROOT)/Assets/player.psd
$(SRCROOT)/Assets/hud.psd
```

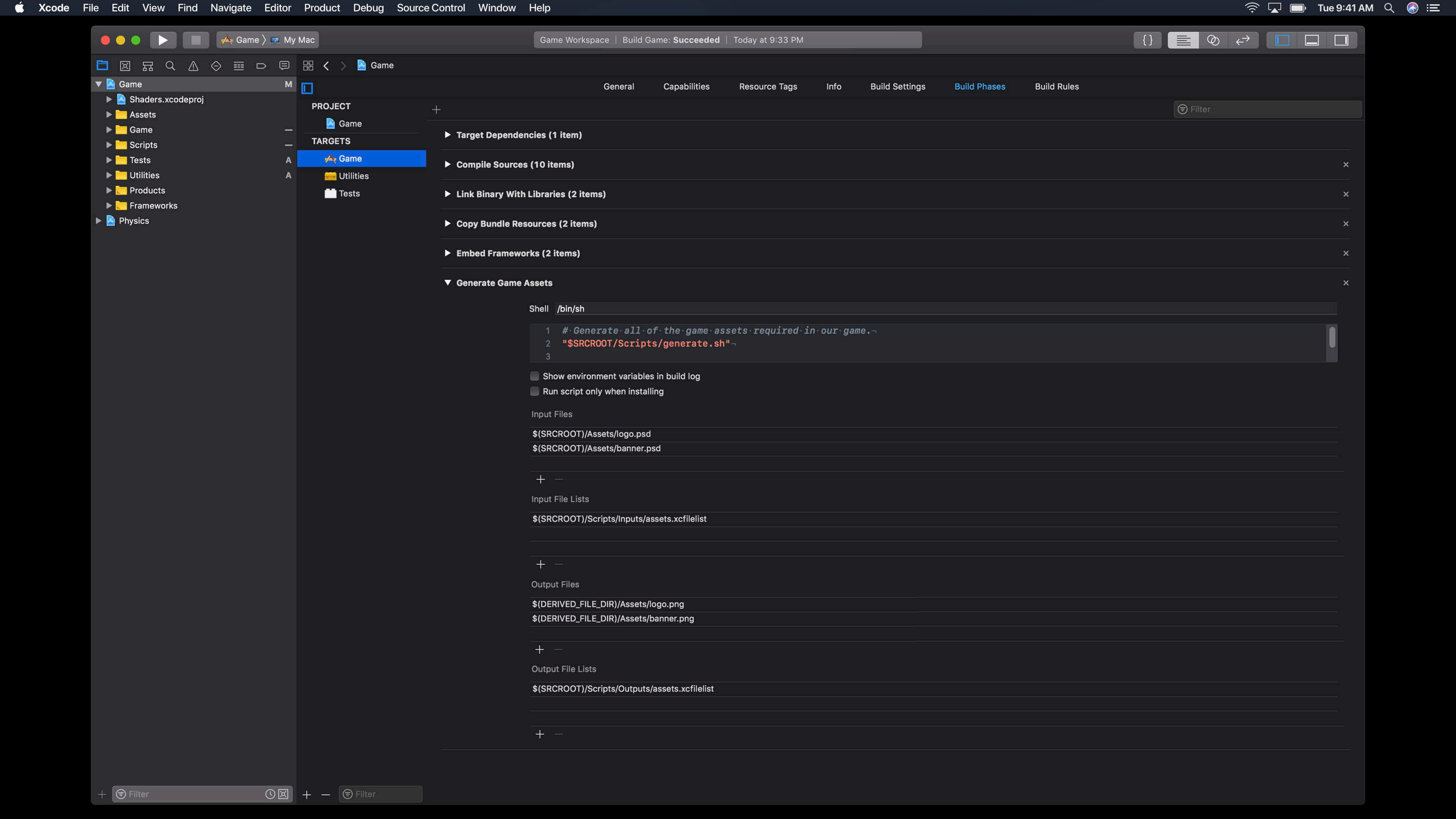
File Lists

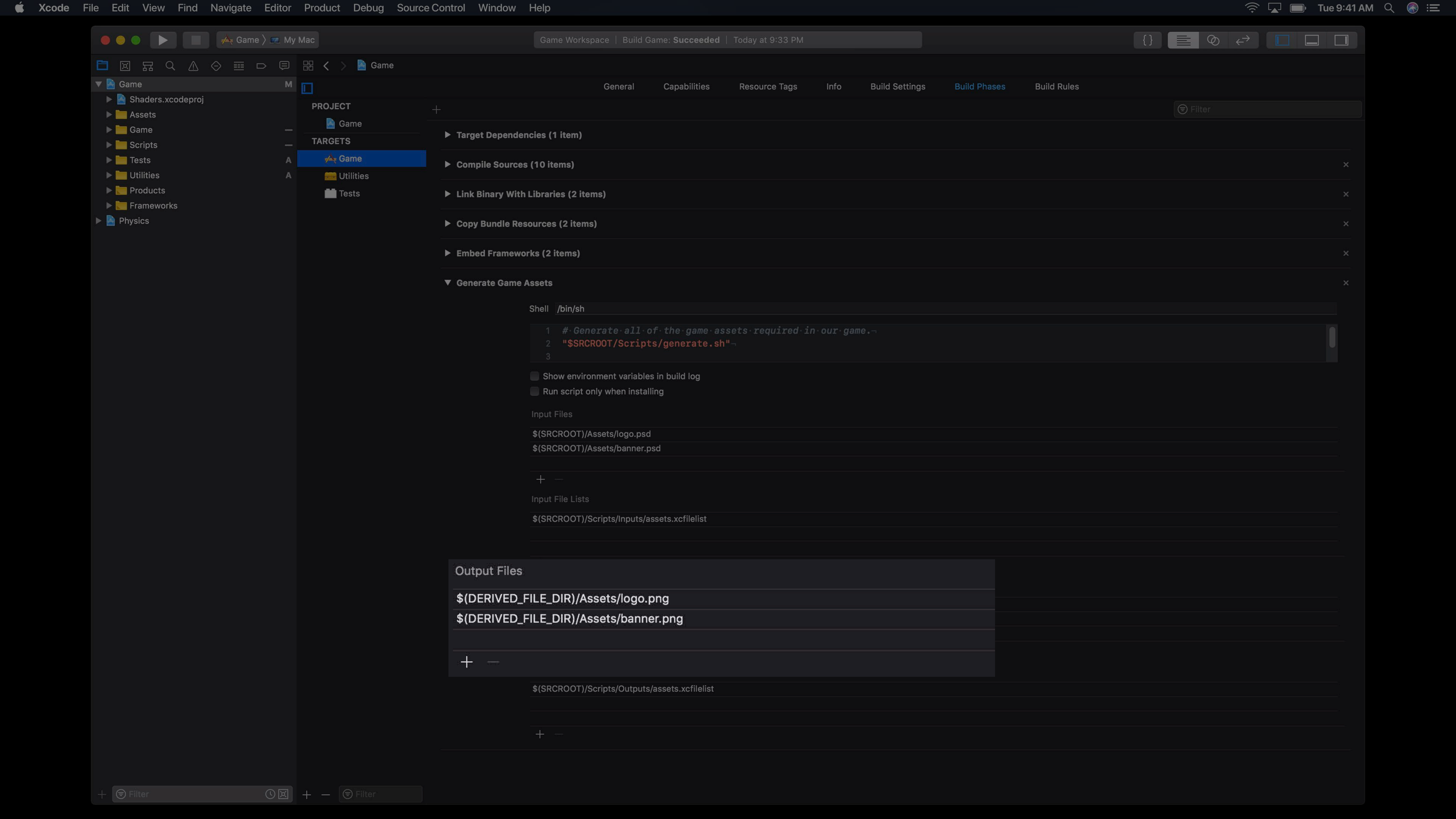
Newline separated

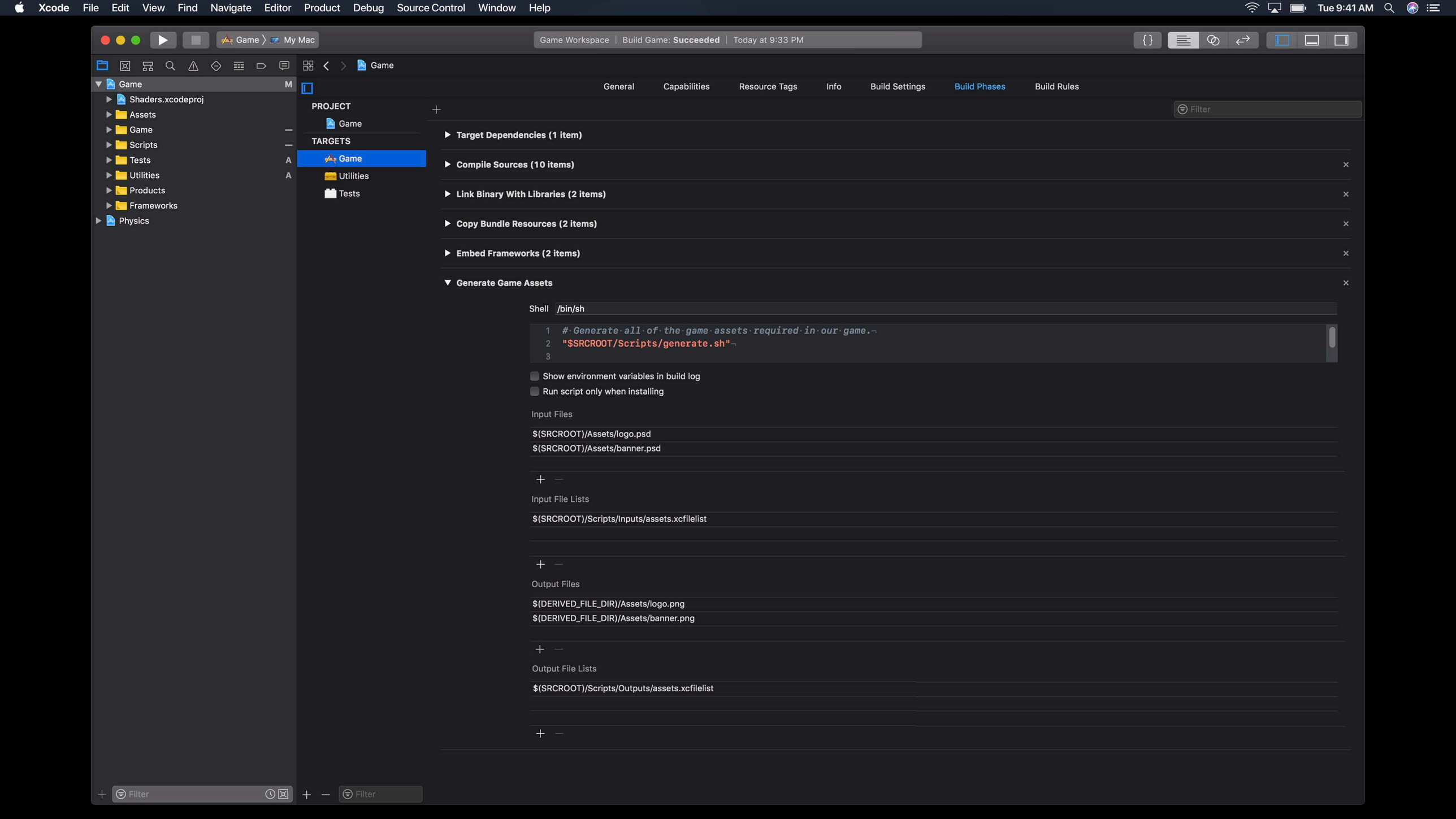
Support for build setting variables

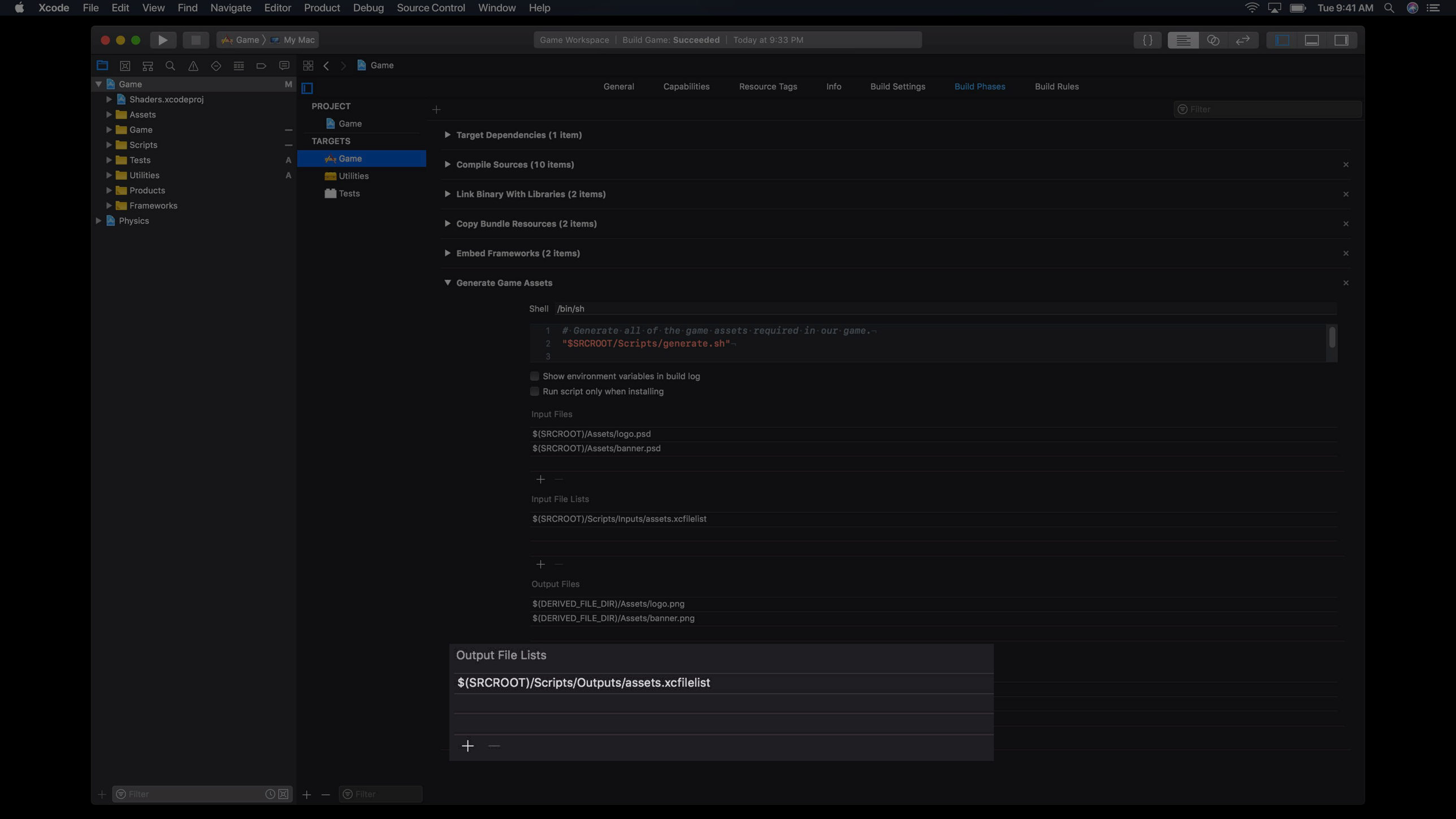
Cannot be generated during the build

```
#  assets.xcfilelist
#  Game

# Asset masters
$(SRCROOT)/Assets/game.psd
$(SRCROOT)/Assets/player.psd
$(SRCROOT)/Assets/hud.psd
```


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
Generate all of the game assets required in our game.
"$SRCROOT/Scripts/generate.sh"
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
Generate all of the game assets required in our game.
"$SRCROOT/Scripts/generate.sh"
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Game M
Shaders.xcodeproj
Assets
Game
Scripts
Tests A
Utilities A
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
1 # Generate all of the game assets required in our game.
2 "$SRCROOT/Scripts/generate.sh"
3
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist
Filter
Filter
Filter


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 9:33 PM
Game
Shaders.xcodeproj
Assets
Game
Scripts
Tests
Utilities
Products
Frameworks
Physics
PROJECT
Game
TARGETS
Game
Utilities
Tests
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
Filter
Target Dependencies (1 item)
Compile Sources (10 items)
Link Binary With Libraries (2 items)
Copy Bundle Resources (2 items)
Embed Frameworks (2 items)
Generate Game Assets
Shell /bin/sh
Generate all of the game assets required in our game.
"$SRCROOT/Scripts/generate.sh"
Show environment variables in build log
Run script only when installing
Input Files
$(SRCROOT)/Assets/logo.psd
$(SRCROOT)/Assets/banner.psd
Input File Lists
$(SRCROOT)/Scripts/Inputs/assets.xcfilelist
Output Files
$(DERIVED_FILE_DIR)/Assets/logo.png
$(DERIVED_FILE_DIR)/Assets/banner.png
Output File Lists
$(SRCROOT)/Scripts/Outputs/assets.xcfilelist

When the Run Script Phase is Run

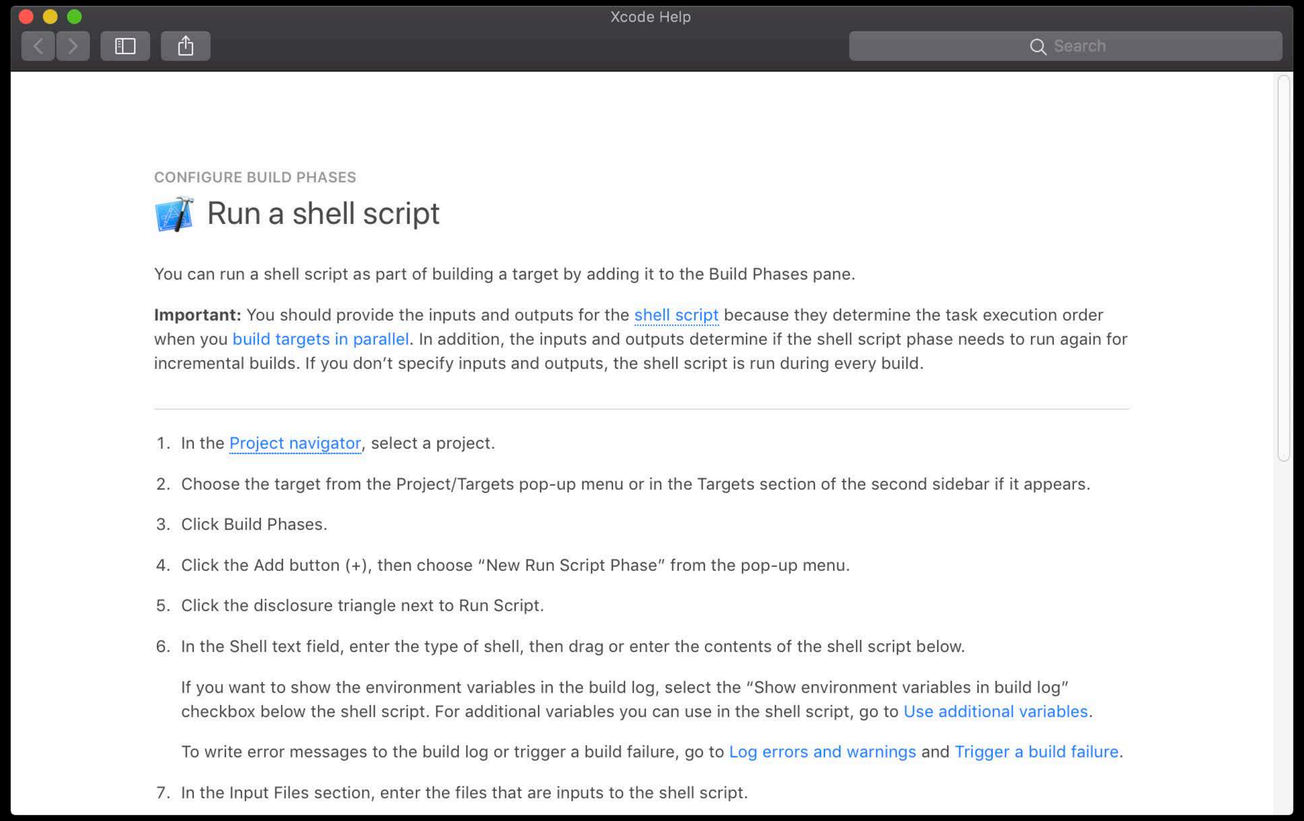
No input files declared

Input files changed

Output files missing

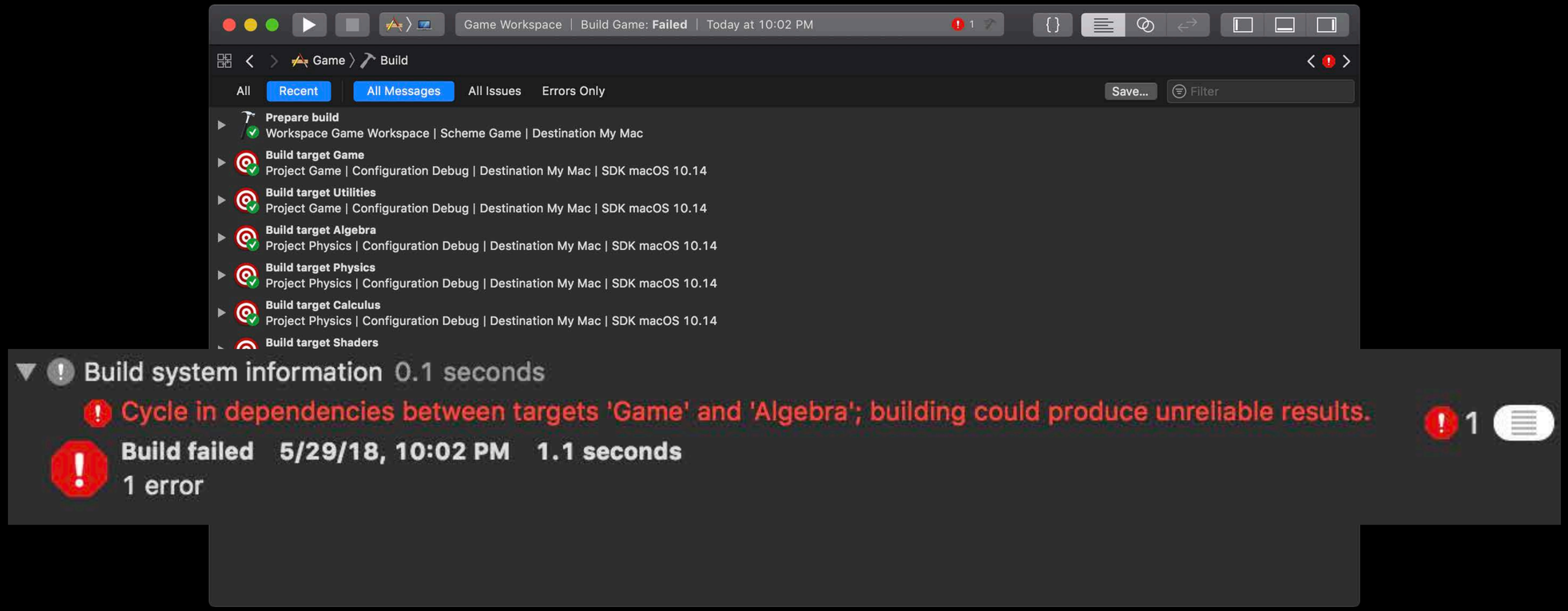
Documentation

NEW



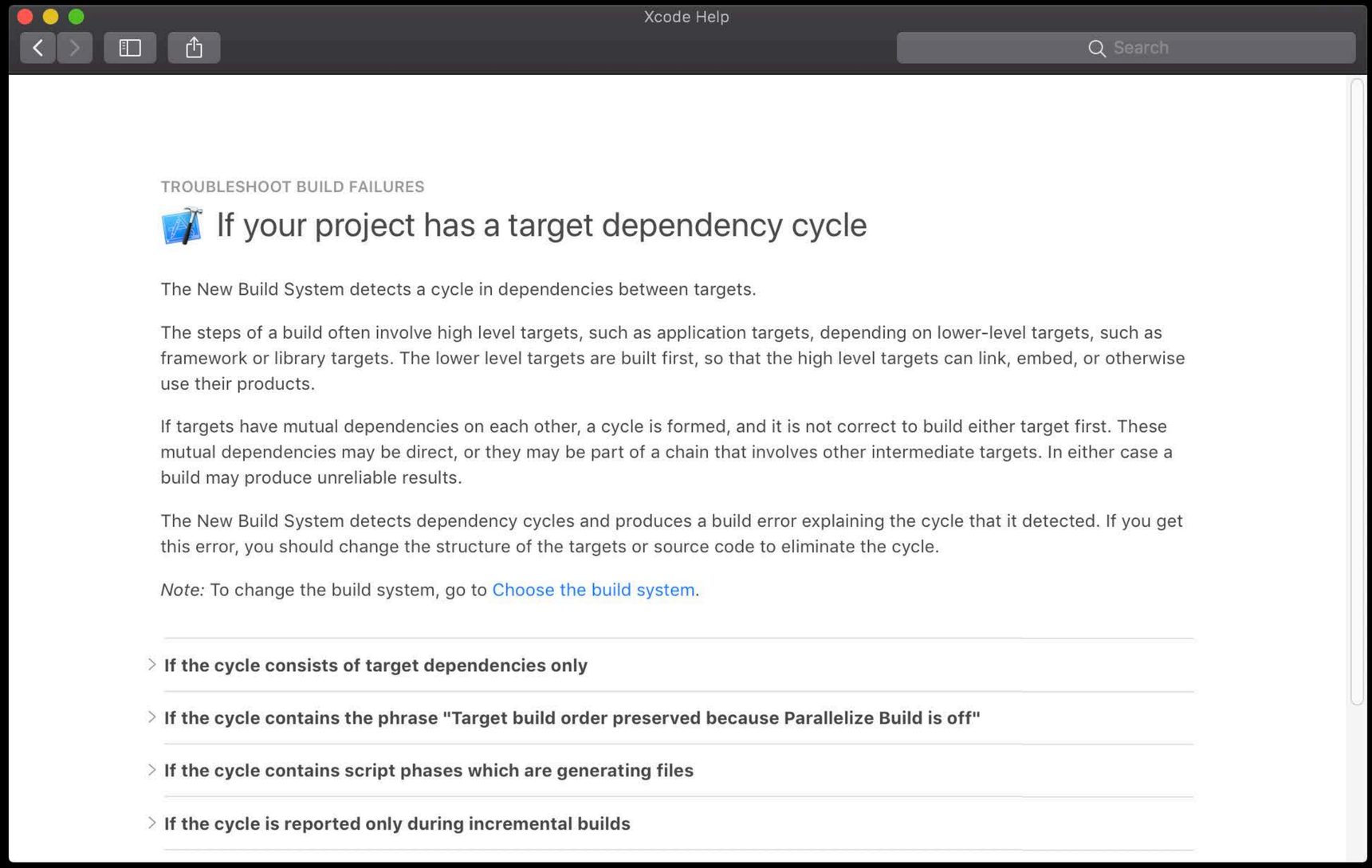
Dependency Cycle Detection

NEW



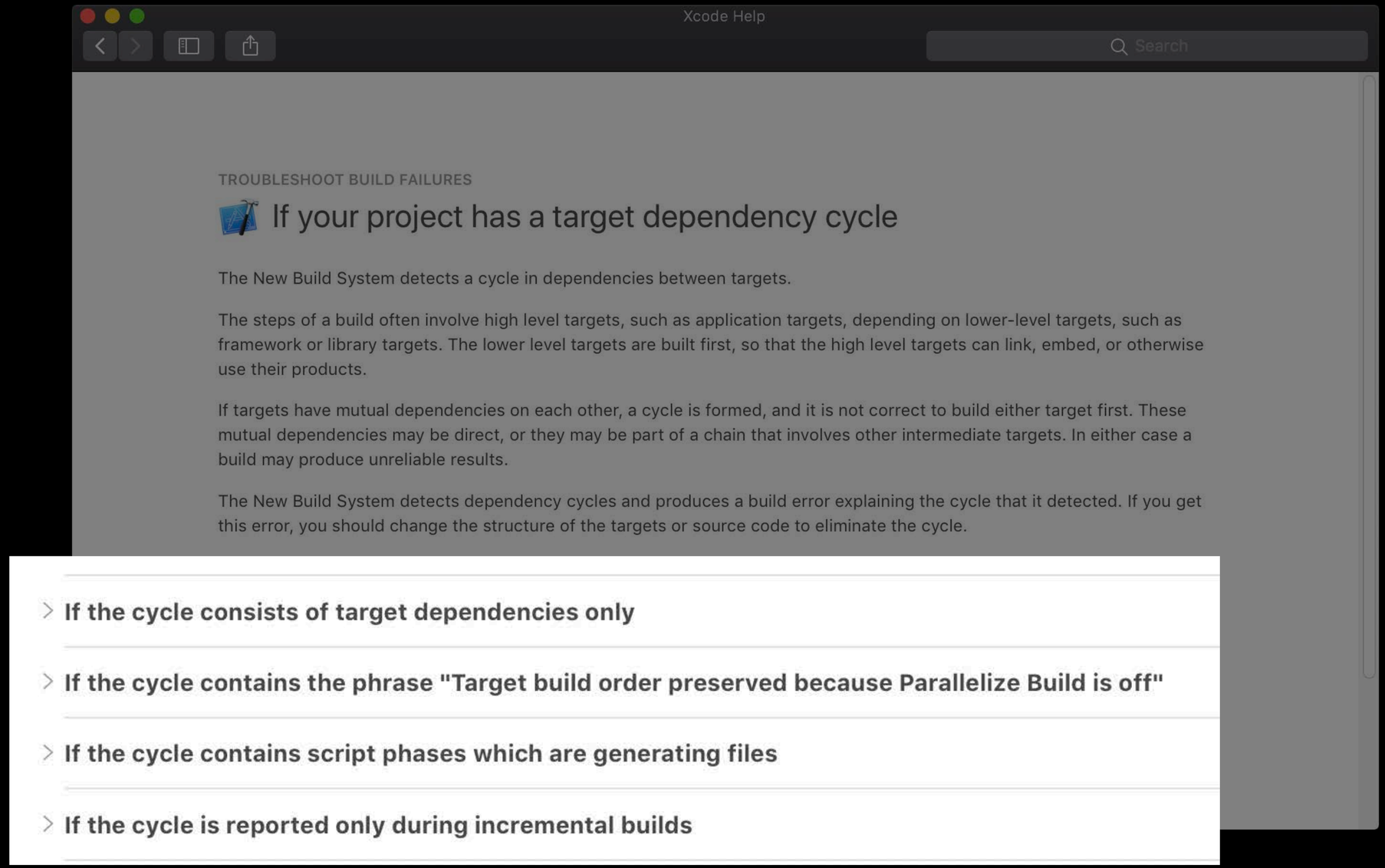
More on Dependency Cycles

NEW



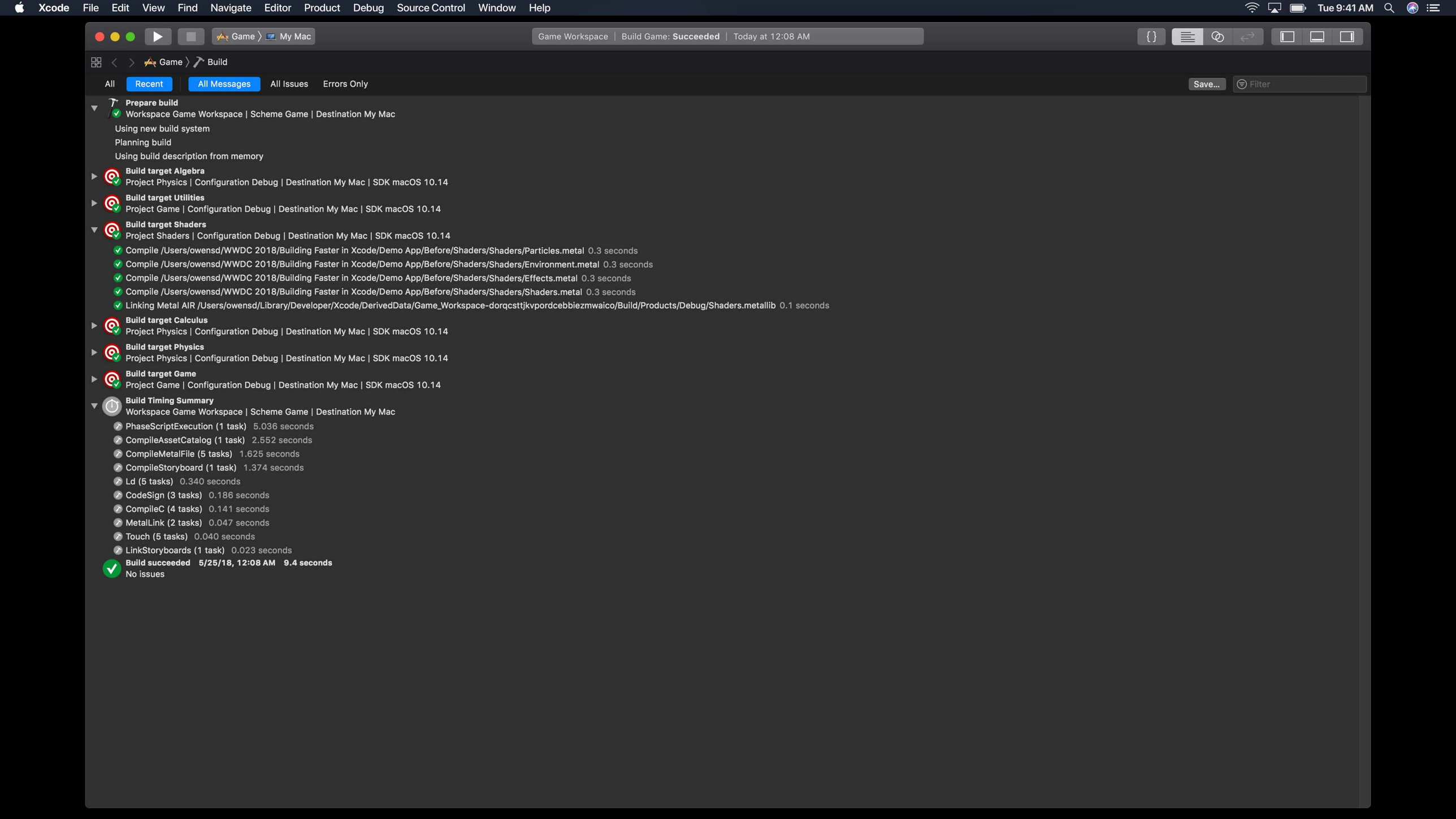
More on Dependency Cycles

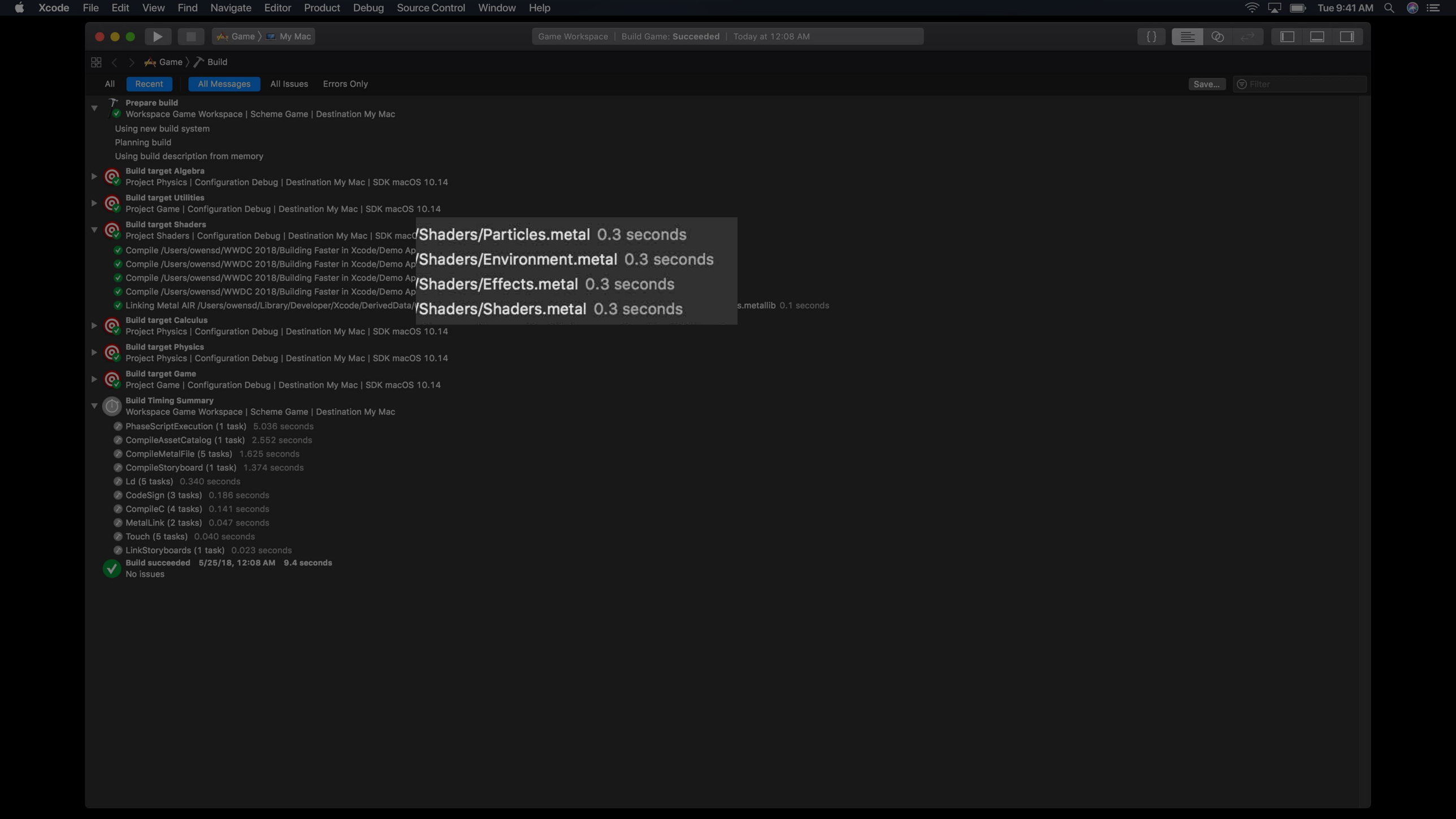
NEW



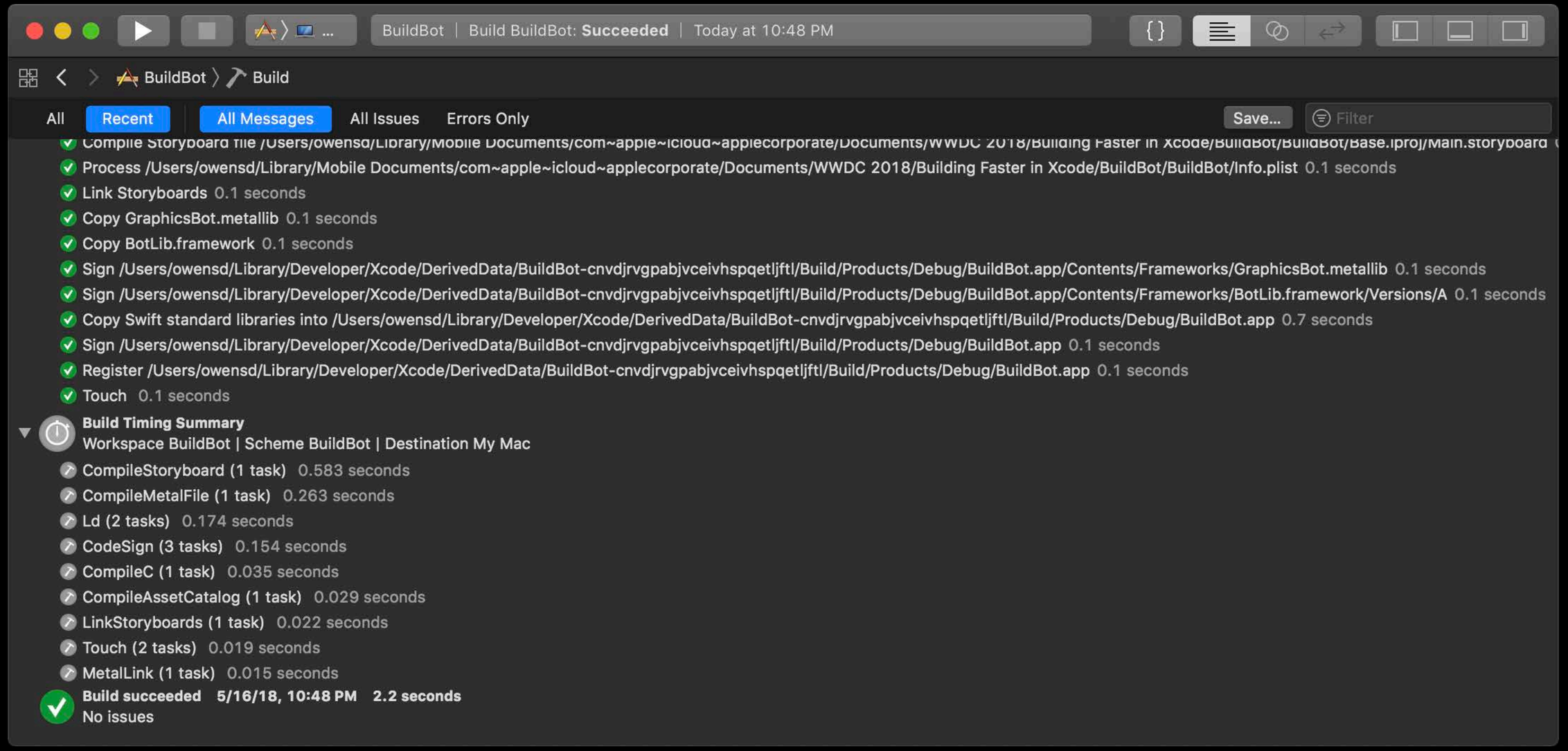
Measuring Build Time

Increasing build efficiency

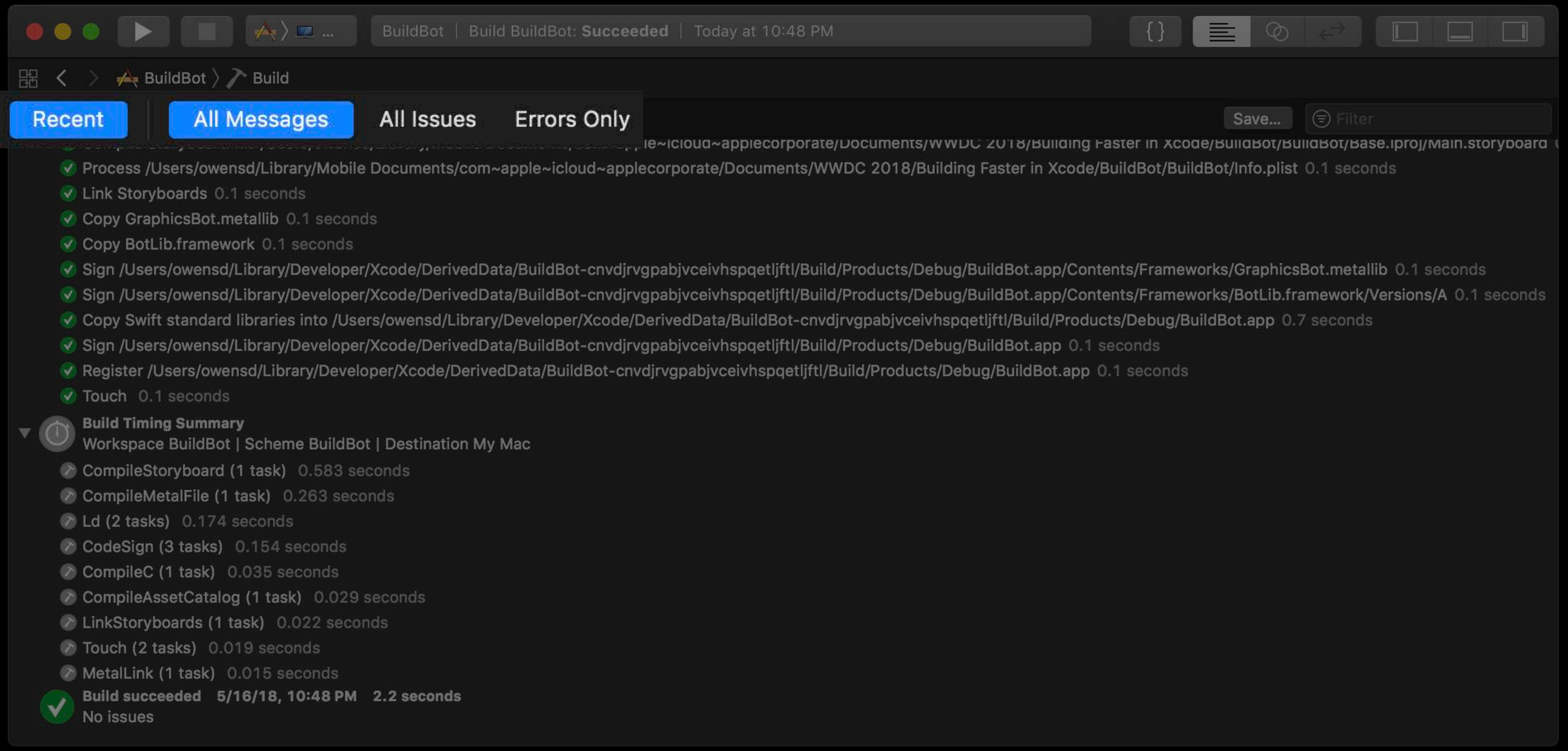

Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 12:08 AM
Game 〉 Build
All Recent All Messages All Issues Errors Only
Save...
Filter
Prepare build
Workspace Game Workspace | Scheme Game | Destination My Mac
Using new build system
Planning build
Using build description from memory
Build target Algebra
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Utilities
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Shaders
Project Shaders | Configuration Debug | Destination My Mac | SDK macOS 10.14
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Particles.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Environment.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Effects.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Shaders.metal 0.3 seconds
Linking Metal AIR /Users/owensd/Library/Developer/Xcode/DerivedData/Game_Workspace-dorqcsttjkvpordcebbiezmwaico/Build/Products/Debug/Shaders.metallib 0.1 seconds
Build target Calculus
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Physics
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Game
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build Timing Summary
Workspace Game Workspace | Scheme Game | Destination My Mac
PhaseScriptExecution (1 task) 5.036 seconds
CompileAssetCatalog (1 task) 2.552 seconds
CompileMetalFile (5 tasks) 1.625 seconds
CompileStoryboard (1 task) 1.374 seconds
Ld (5 tasks) 0.340 seconds
CodeSign (3 tasks) 0.186 seconds
CompileC (4 tasks) 0.141 seconds
MetalLink (2 tasks) 0.047 seconds
Touch (5 tasks) 0.040 seconds
LinkStoryboards (1 task) 0.023 seconds
Build succeeded 5/25/18, 12:08 AM 9.4 seconds
No issues


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game My Mac
Game Workspace | Build Game: Succeeded | Today at 12:08 AM
Game Build
All Recent All Messages All Issues Errors Only
Save...
Filter
Prepare build
Workspace Game Workspace | Scheme Game | Destination My Mac
Using new build system
Planning build
Using build description from memory
Build target Algebra
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Utilities
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Shaders
Project Shaders | Configuration Debug | Destination My Mac | SDK macO
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo Ap
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo Ap
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo Ap
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo Ap
Linking Metal AIR /Users/owensd/Library/Developer/Xcode/DerivedData/
/Shaders/Particles.metal 0.3 seconds
/Shaders/Environment.metal 0.3 seconds
/Shaders/Effects.metal 0.3 seconds
/Shaders/Shaders.metal 0.3 seconds
s.metallib 0.1 seconds
Build target Calculus
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Physics
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Game
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build Timing Summary
Workspace Game Workspace | Scheme Game | Destination My Mac
PhaseScriptExecution (1 task) 5.036 seconds
CompileAssetCatalog (1 task) 2.552 seconds
CompileMetalFile (5 tasks) 1.625 seconds
CompileStoryboard (1 task) 1.374 seconds
Ld (5 tasks) 0.340 seconds
CodeSign (3 tasks) 0.186 seconds
CompileC (4 tasks) 0.141 seconds
MetalLink (2 tasks) 0.047 seconds
Touch (5 tasks) 0.040 seconds
LinkStoryboards (1 task) 0.023 seconds
Build succeeded 5/25/18, 12:08 AM 9.4 seconds
No issues

Build Log Filters

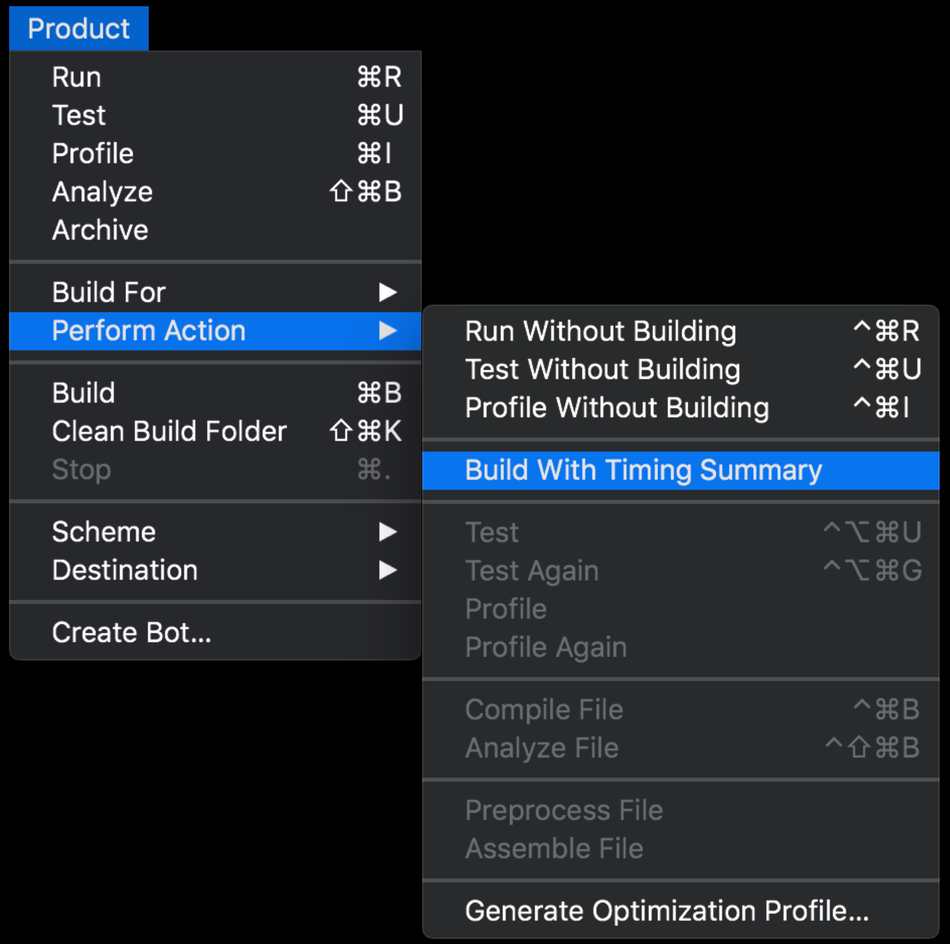


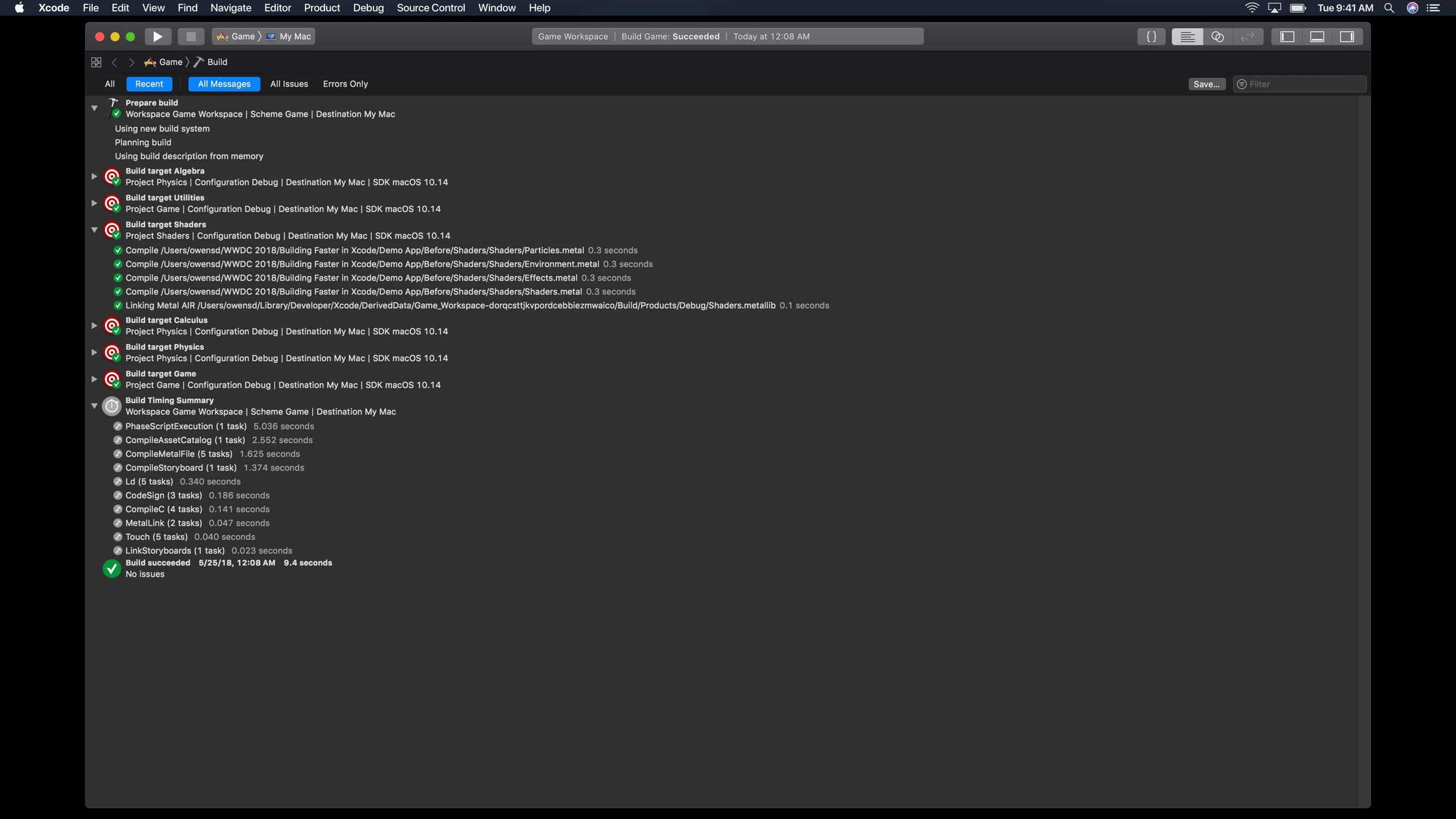
Build Log Filters



Build With Timing Summary

NEW




Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Tue 9:41 AM
Game 〉 My Mac
Game Workspace | Build Game: Succeeded | Today at 12:08 AM
Game 〉 Build
All Recent All Messages All Issues Errors Only
Save...
Filter
Prepare build
Workspace Game Workspace | Scheme Game | Destination My Mac
Using new build system
Planning build
Using build description from memory
Build target Algebra
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Utilities
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Shaders
Project Shaders | Configuration Debug | Destination My Mac | SDK macOS 10.14
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Particles.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Environment.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Effects.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Shaders.metal 0.3 seconds
Linking Metal AIR /Users/owensd/Library/Developer/Xcode/DerivedData/Game_Workspace-dorqcsttjkvpordcebbiezmwaico/Build/Products/Debug/Shaders.metallib 0.1 seconds
Build target Calculus
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Physics
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build target Game
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14
Build Timing Summary
Workspace Game Workspace | Scheme Game | Destination My Mac
PhaseScriptExecution (1 task) 5.036 seconds
CompileAssetCatalog (1 task) 2.552 seconds
CompileMetalFile (5 tasks) 1.625 seconds
CompileStoryboard (1 task) 1.374 seconds
Ld (5 tasks) 0.340 seconds
CodeSign (3 tasks) 0.186 seconds
CompileC (4 tasks) 0.141 seconds
MetalLink (2 tasks) 0.047 seconds
Touch (5 tasks) 0.040 seconds
LinkStoryboards (1 task) 0.023 seconds
Build succeeded 5/25/18, 12:08 AM 9.4 seconds
No issues

Build target Algebra
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Utilities
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Shaders
Project Shaders | Configuration Debug | Destination My Mac | SDK macOS 10.14

- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Particles.metal 0.3 seconds
- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Environment.metal 0.3 seconds
- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Effects.metal 0.3 seconds
- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Shaders.metal 0.3 seconds
- Linking Metal AIR /Users/owensd/Library/Developer/Xcode/DerivedData/Game_Workspace-dorqcsttjkvpordcebbiezmwaico/Build/Products/Debug/Shaders.metallib 0.1 seconds

Build target Calculus
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Physics
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Game
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build Timing Summary
Workspace Game Workspace | Scheme Game | Destination My Mac

- PhaseScriptExecution (1 task) 5.036 seconds
- CompileAssetCatalog (1 task) 2.552 seconds
- CompileMetalFile (5 tasks) 1.625 seconds
- CompileStoryboard (1 task) 1.374 seconds
- Ld (5 tasks) 0.340 seconds
- CodeSign (3 tasks) 0.186 seconds
- CompileC (4 tasks) 0.141 seconds
- MetalLink (2 tasks) 0.047 seconds
- Touch (5 tasks) 0.040 seconds
- LinkStoryboards (1 task) 0.023 seconds

Build succeeded 5/25/18, 12:08 AM 9.4 seconds
No issues

Build target Algebra
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Utilities
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Shaders
Project Shaders | Configuration Debug | Destination My Mac | SDK macOS 10.14

Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Particles.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Environment.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Effects.metal 0.3 seconds
Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Shaders.metal 0.3 seconds
Linking Metal AIR /Users/owensd/Library/Developer/Xcode/DerivedData/Game_Workspace-dorqcsttjkvpordcebbiezmwaico/Build/Products/Debug/Shaders.metallib 0.1 seconds

Build target Calculus
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Physics
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build Timing Summary
Workspace Game Workspace | Scheme Game | Destination My Mac

PhaseScriptExecution (1 task) 5.036 seconds
CompileAssetCatalog (1 task) 2.552 seconds
CompileMetalFile (5 tasks) 1.625 seconds
CompileStoryboard (1 task) 1.374 seconds
Ld (5 tasks) 0.340 seconds
CodeSign (3 tasks) 0.186 seconds
CompileC (4 tasks) 0.141 seconds
MetalLink (2 tasks) 0.047 seconds
Touch (5 tasks) 0.040 seconds
LinkStoryboards (1 task) 0.023 seconds

Build target Algebra
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Utilities
Project Game | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Shaders
Project Shaders | Configuration Debug | Destination My Mac | SDK macOS 10.14

- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Particles.metal 0.3 seconds
- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Environment.metal 0.3 seconds
- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Effects.metal 0.3 seconds
- Compile /Users/owensd/WWDC 2018/Building Faster in Xcode/Demo App/Before/Shaders/Shaders/Shaders.metal 0.3 seconds
- Linking Metal AIR /Users/owensd/Library/Developer/Xcode/DerivedData/Game_Workspace-dorqcsttjkvpordcebbiezmwaico/Build/Products/Debug/Shaders.metallib 0.1 seconds

Build target Calculus
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build target Physics
Project Physics | Configuration Debug | Destination My Mac | SDK macOS 10.14

Build Timing Summary
Workspace Game Workspace | Scheme Game | Destination My Mac

- PhaseScriptExecution (1 task) 5.036 seconds
- CompileAssetCatalog (1 task) 2.552 seconds
- CompileMetalFile (5 tasks) 1.625 seconds
- CompileStoryboard (1 task) 1.374 seconds
- Ld (5 tasks) 0.340 seconds
- CodeSign (3 tasks) 0.186 seconds
- CompileC (4 tasks) 0.141 seconds
- MetalLink (2 tasks) 0.047 seconds
- Touch (5 tasks) 0.040 seconds
- LinkStoryboards (1 task) 0.023 seconds

```
$ xcodebuild -showBuildTimingSummary

Build Timing Summary

CompileSwiftSources (1 task) | 5.434 seconds
PhaseScriptExecution (1 task) | 5.046 seconds
CompileAssetCatalog (1 task) | 2.788 seconds
CompileStoryboard (1 task) | 1.880 seconds
CompileMetalFile (5 tasks) | 1.735 seconds
CopySwiftLibs (1 task) | 0.740 seconds
Ld (2 tasks) | 0.306 seconds
CodeSign (3 tasks) | 0.177 seconds
CompileC (1 task) | 0.170 seconds
MetalLink (2 tasks) | 0.046 seconds
Ditto (4 tasks) | 0.032 seconds
LinkStoryboards (1 task) | 0.023 seconds
```

```
$ xcodebuild -showBuildTimingSummary
```

```
Build Timing Summary

CompileSwiftSources (1 task) | 5.434 seconds
PhaseScriptExecution (1 task) | 5.046 seconds
CompileAssetCatalog (1 task) | 2.788 seconds
CompileStoryboard (1 task) | 1.880 seconds
CompileMetalFile (5 tasks) | 1.735 seconds
CopySwiftLibs (1 task) | 0.740 seconds
Ld (2 tasks) | 0.306 seconds
CodeSign (3 tasks) | 0.177 seconds
CompileC (1 task) | 0.170 seconds
MetalLink (2 tasks) | 0.046 seconds
Ditto (4 tasks) | 0.032 seconds
LinkStoryboards (1 task) | 0.023 seconds
```

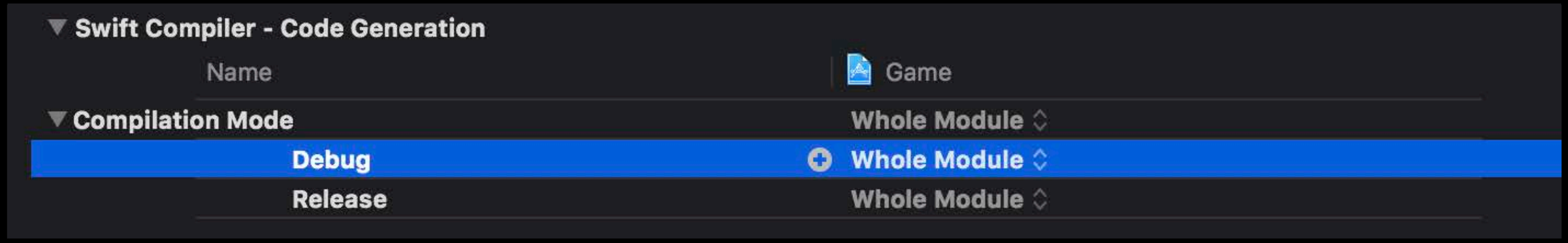
Source-Level Improvements

Jordan Rose, Swift Engineer

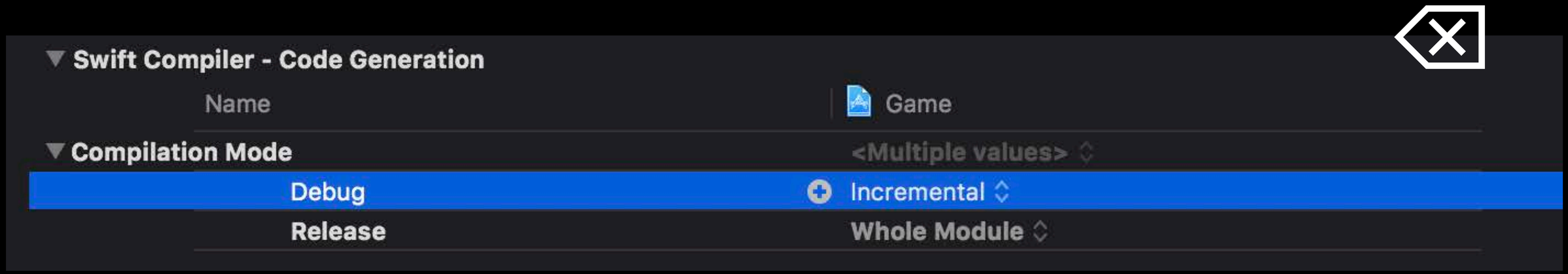
Remove this build time workaround...

Reducing the work on rebuilds

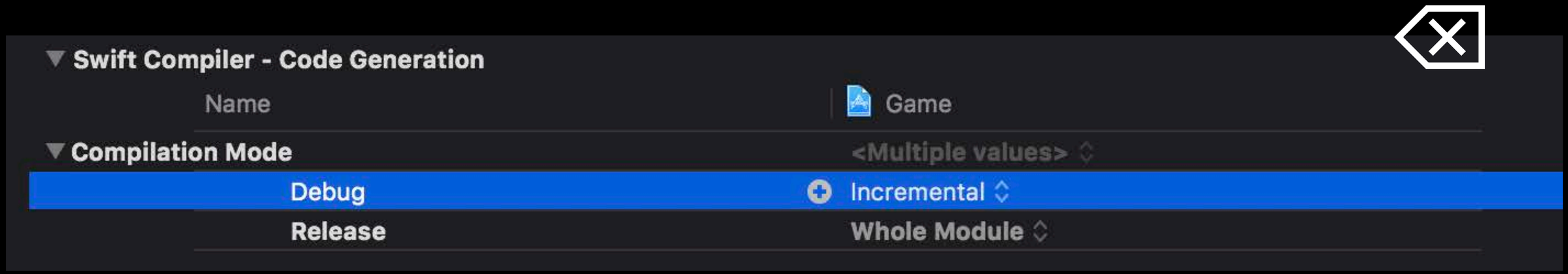
Turn off Whole Module Mode For Debug Builds



Turn off Whole Module Mode For Debug Builds



Turn off Whole Module Mode For Debug Builds



What’s New in Swift		WWDC 2018
Behind the Scenes of the Xcode Build Process	Hall 2	Friday 2:00PM

Increasing Build Efficiency	Reducing the Work on Rebuilds
Parallelizing your build process	Declaring script inputs and outputs
Measuring your build time	Understanding dependencies in Swift
Dealing with complex expressions	Limiting your Objective-C/Swift interface

Dealing with complex expressions

Understanding dependencies in Swift

Limiting your Objective-C/Swift interface

When a build takes a long time,
there is often a key piece of information
you can add to make it better.

Dealing with Complex Expressions

Increasing build efficiency

Use Explicit Types for Complex Properties

```
struct ContrivedExample {
  var bigNumber = [4, 3, 2].reduce(1) {
    soFar, next in
    pow(next, soFar)
  }
}
```

Use Explicit Types for Complex Properties

```
struct ContrivedExample {
  var bigNumber: Double = [4, 3, 2].reduce(1) {
    soFar, next in
    pow(next, soFar)
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in



  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

error: the compiler is unable to type-check this expression in reasonable time

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    (soFar: Int?, next: Int?) -> Int? in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Provide Types in Complex Closures

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    soFar != nil && next != nil ? soFar! + next! :
      (soFar != nil ? soFar! : (next != nil ? next! : nil))
  }
}
```

Break Apart Complex Expressions

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in




  }
}
```

Break Apart Complex Expressions

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    if soFar != nil && next != nil { return soFar! + next! }
    if soFar != nil { return soFar! }
    if next != nil { return next! }
    return nil
  }
}
```

Break Apart Complex Expressions

```
func sumNonOptional(i: Int?, j: Int?, k: Int?) -> Int? {
  return [i, j, k].reduce(0) {
    soFar, next in
    if let soFar = soFar {
      if let next = next { return soFar + next }
      return soFar
    } else {
      return next
    }
  }
}
```

Use `AnyObject` Methods and Properties Sparingly

```
weak var delegate: AnyObject?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}
```

Use `AnyObject` Methods and Properties Sparingly

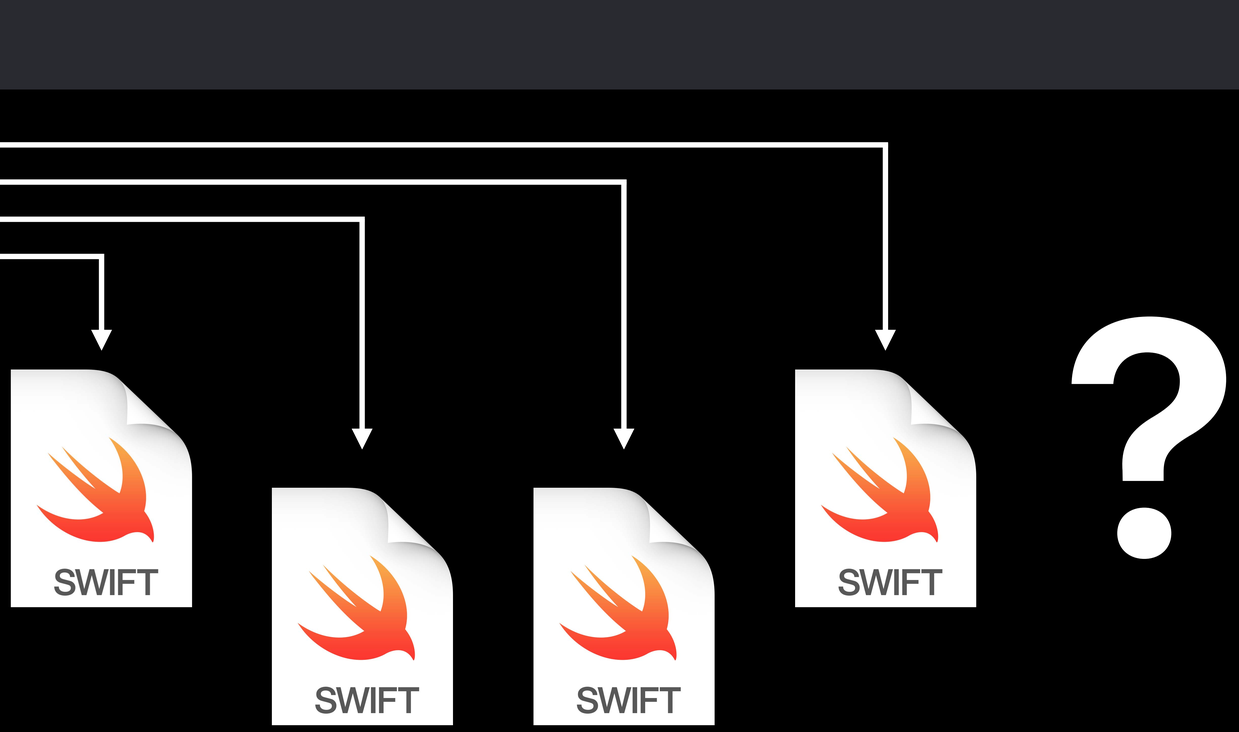
```
weak var delegate: AnyObject?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}
```

Use `AnyObject` Methods and Properties Sparingly

```
weak var delegate: AnyObject?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}
```

Use `AnyObject` Methods and Properties Sparingly

```
weak var delegate: AnyObject?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}
```



Use `AnyObject` Methods and Properties Sparingly

Define a protocol instead

```
weak var delegate: AnyObject?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}
```

```
protocol MyOperationDelegate: class {
  func myOperationDidSucceed(_ operation: MyOperation)
}
```

Use `AnyObject` Methods and Properties Sparingly

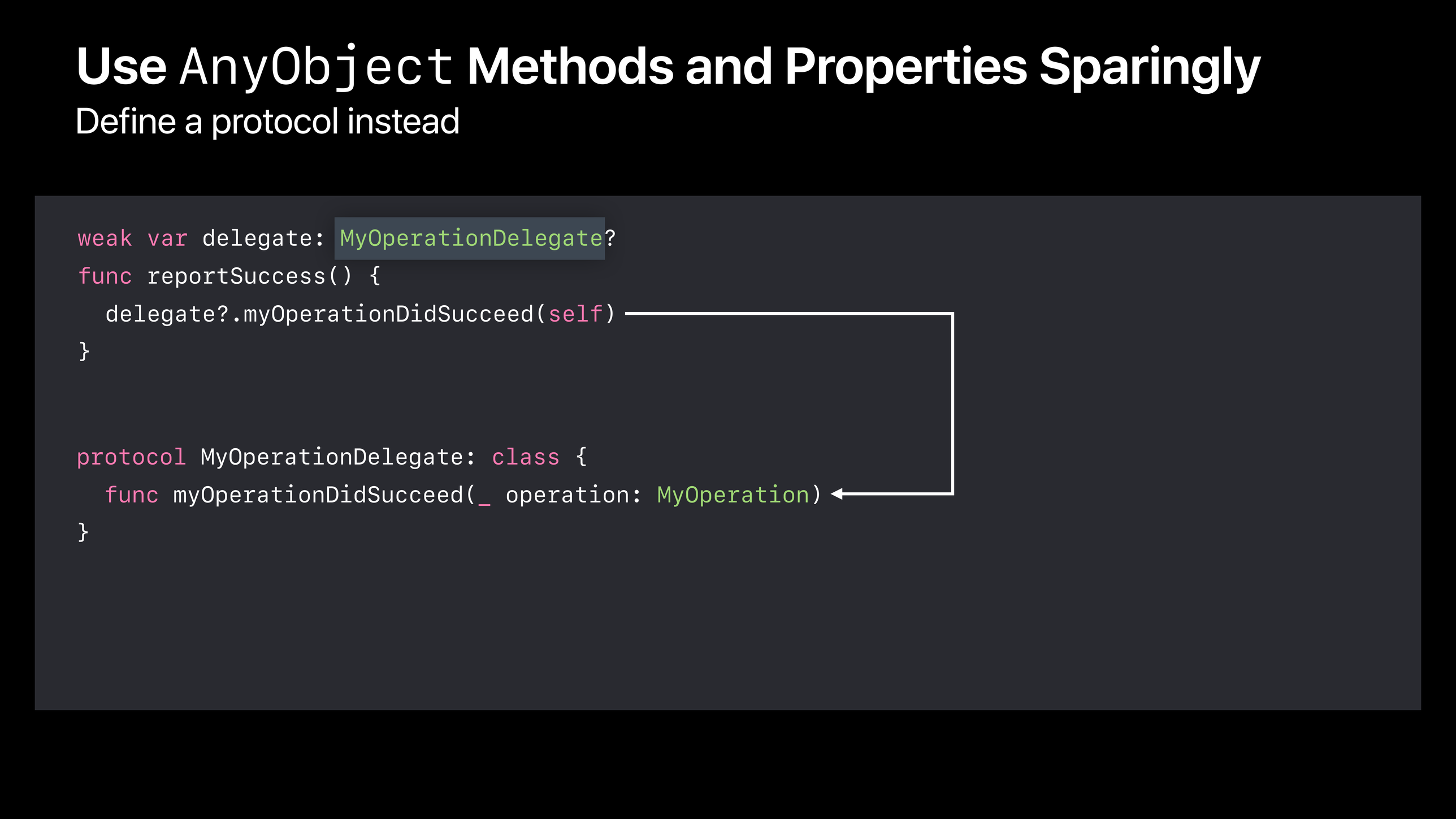
Define a protocol instead

```
weak var delegate: MyOperationDelegate?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}


protocol MyOperationDelegate: class {
  func myOperationDidSucceed(_ operation: MyOperation)
}
```

Use `AnyObject` Methods and Properties Sparingly

Define a protocol instead

```
weak var delegate: MyOperationDelegate?
func reportSuccess() {
  delegate?.myOperationDidSucceed(self)
}


protocol MyOperationDelegate: class {
  func myOperationDidSucceed(_ operation: MyOperation)
}
```



Understanding Dependencies in Swift

Reducing the work on rebuilds

Incremental Builds Are File-Based

```
struct Point {
  var x, y: Double
}
```

Incremental Builds Are File-Based

```
struct Point {
  var x, y: Double
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

Incremental Builds Are File-Based

```
struct Point {
  var x, y: Double
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

Incremental Builds Are File-Based

```
struct Point {
  var x, y: Double
  init(x: Double, y: Double) {
    assert(x.isNormal)
    assert(y.isNormal)
    self.x = x
    self.y = y
  }
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

Incremental Builds Are File-Based

```
struct Point {
  var x, y: Double
  init(x: Double, y: Double) {
    assert(x.isNormal)
    assert(y.isNormal)
    self.x = x
    self.y = y
  }
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

Incremental Builds Are File-Based

Changes in function bodies do not affect other files

```swift
struct Point {
  var x, y: Double
  init(x: Double, y: Double) {
    assert(x.isFinite)
    assert(y.isFinite)
    self.x = x
    self.y = y
  }
}
```


```swift
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

Incremental Builds Are File-Based

Changes in function bodies do not affect other files

```
struct Point {
  var x, y: Double
  init(x: Double, y: Double) {
    assert(x.isFinite)
    assert(y.isFinite)
    self.x = x
    self.y = y
  }
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

Incremental Builds Are File-Based

Unrelated changes outside function bodies can still result in rebuilding

```
struct Point {
  var x, y: Double
  init(x: Double, y: Double) {
    assert(x.isFinite)
    assert(y.isFinite)
    self.x = x
    self.y = y
  }
}
struct PathSegment {
  var start, end: Point
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

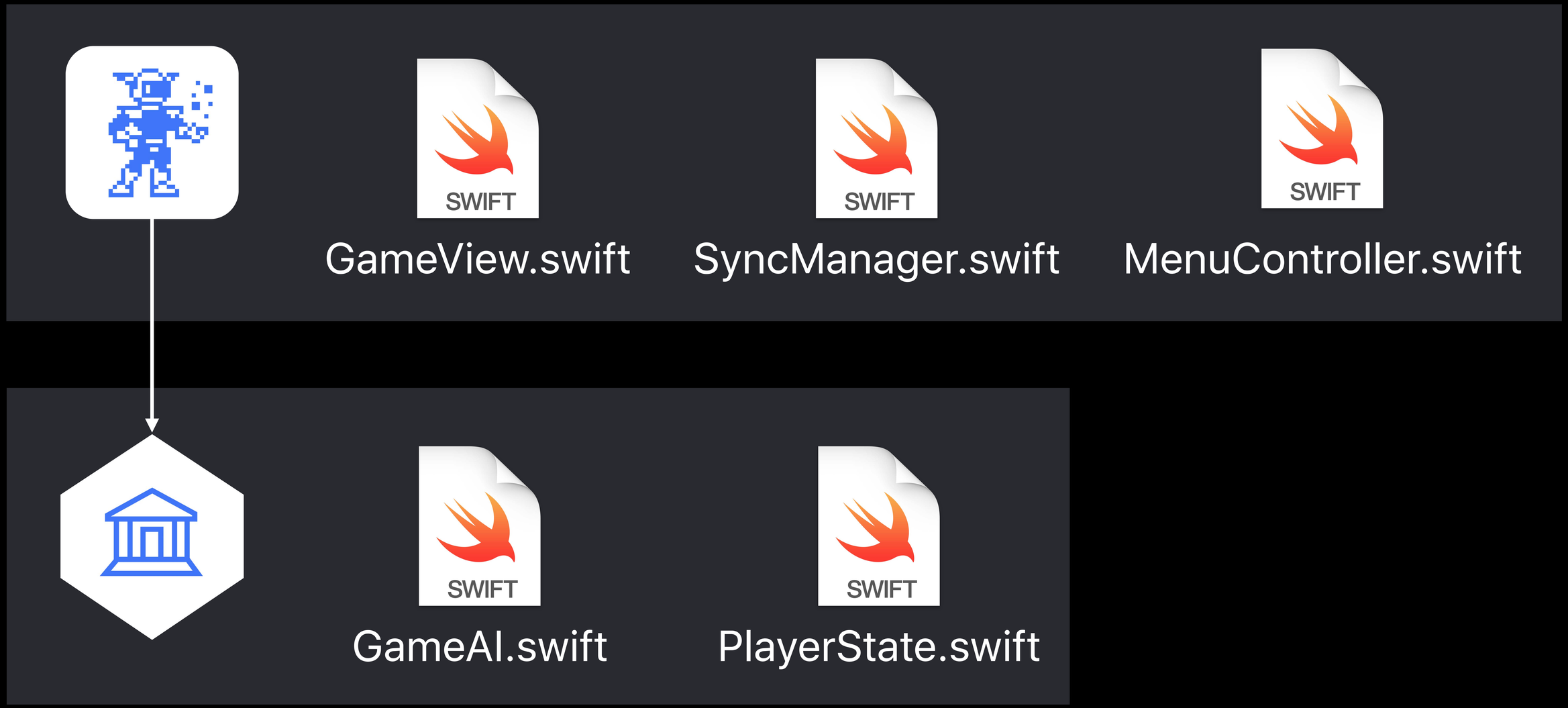
Incremental Builds Are File-Based

Unrelated changes outside function bodies can still result in rebuilding

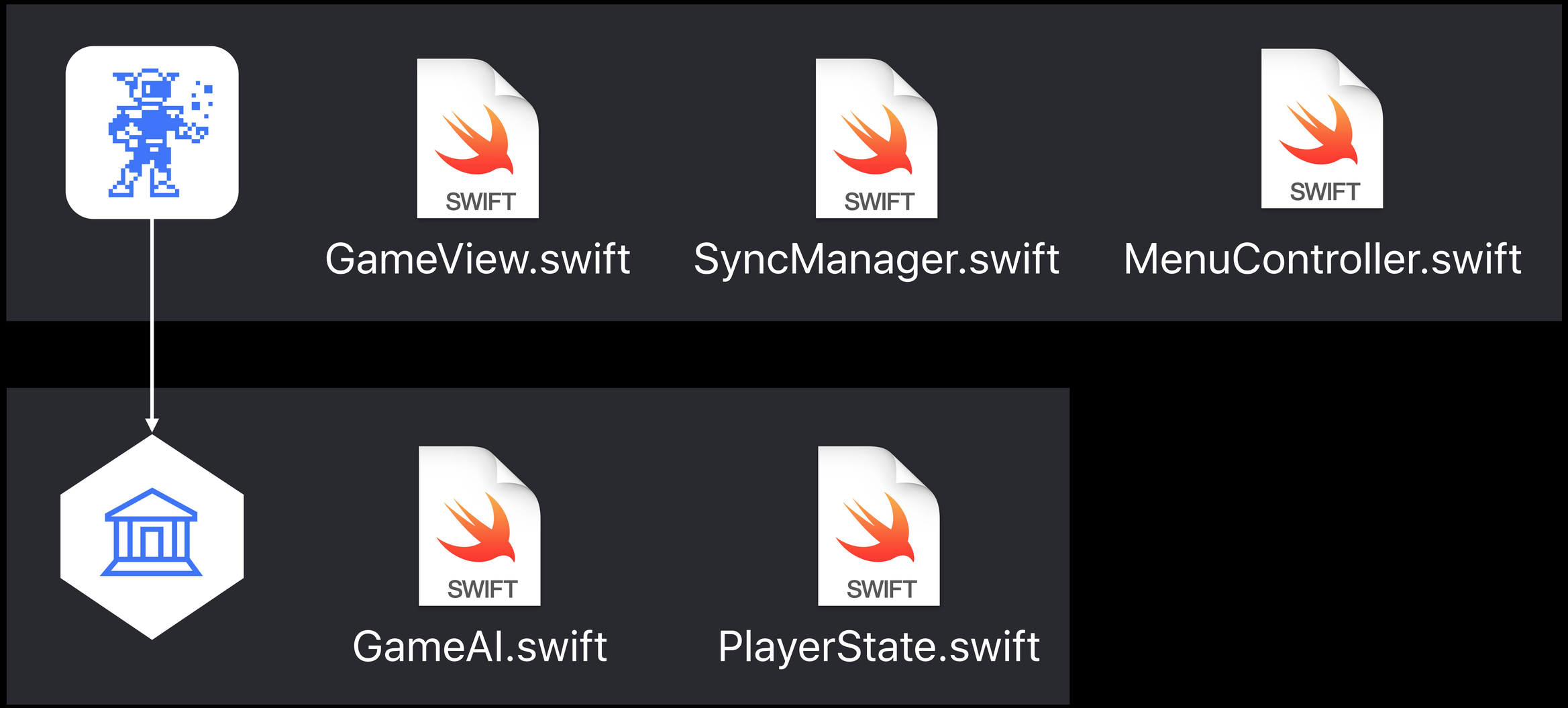
```
struct Point {
  var x, y: Double
  init(x: Double, y: Double) {
    assert(x.isFinite)
    assert(y.isFinite)
    self.x = x
    self.y = y
  }
}
struct PathSegment {
  var start, end: Point
}
```


```
let point = Point(x: 3.0, y: 4.0)

let distance =
  sqrt(point.x * point.x +
       point.y * point.y)
```

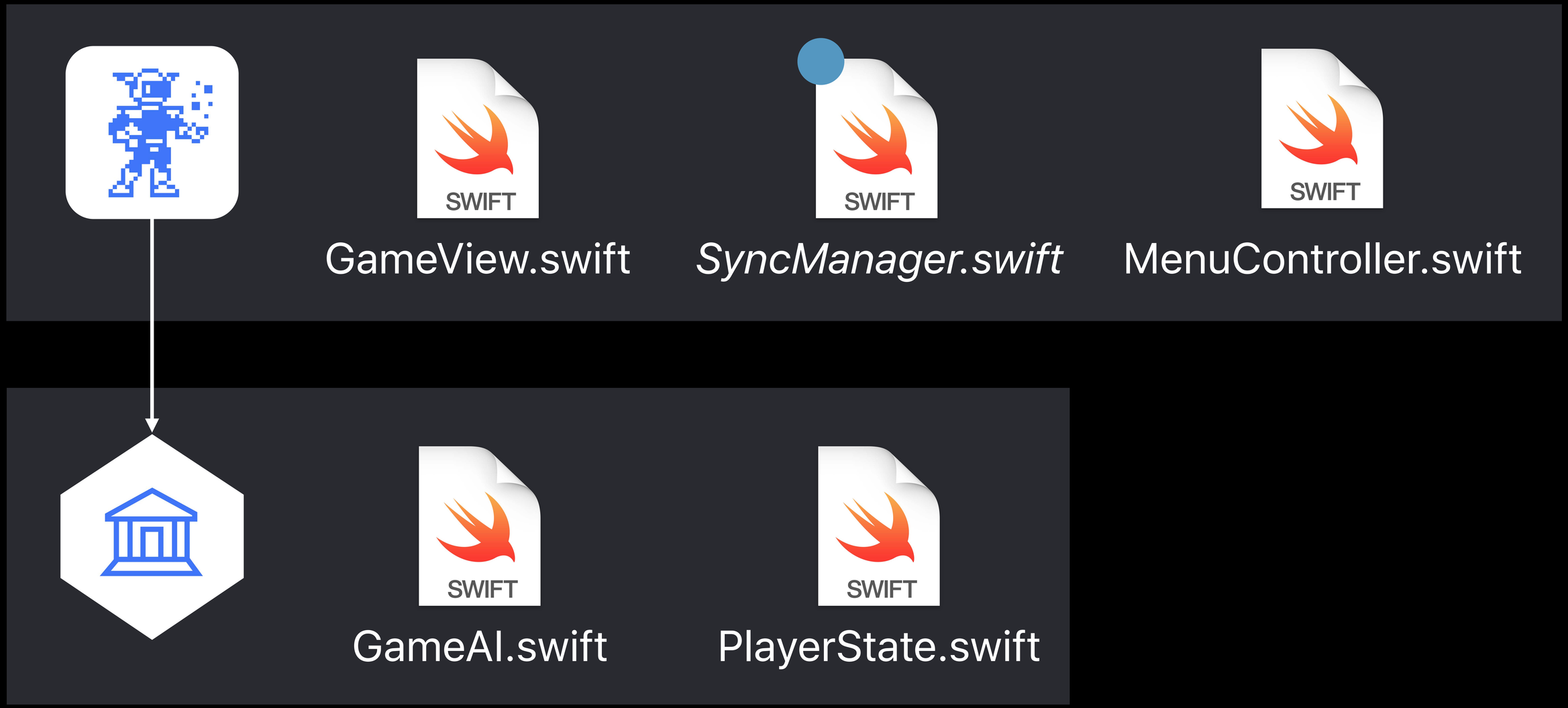
Dependencies Within a Target Are Per-File



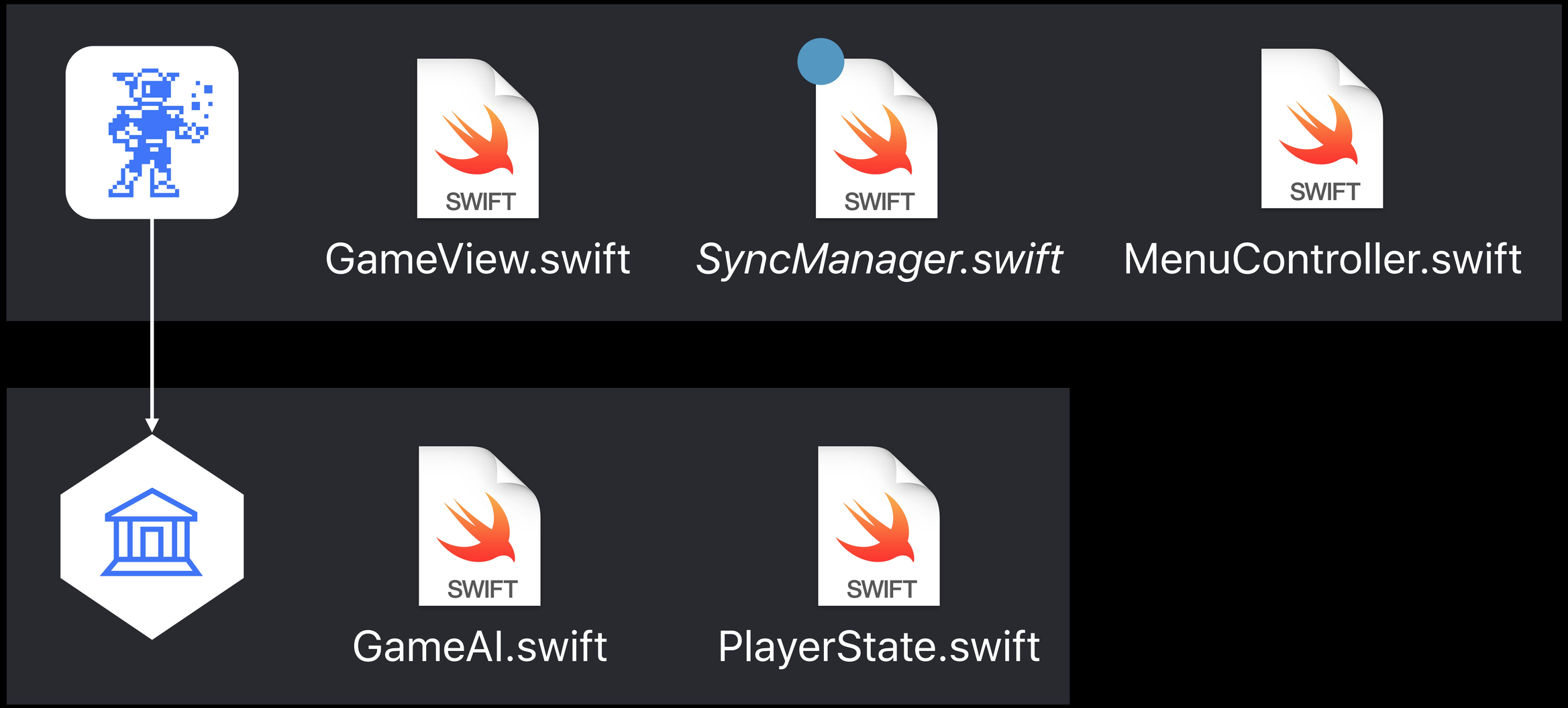
Dependencies Within a Target Are Per-File



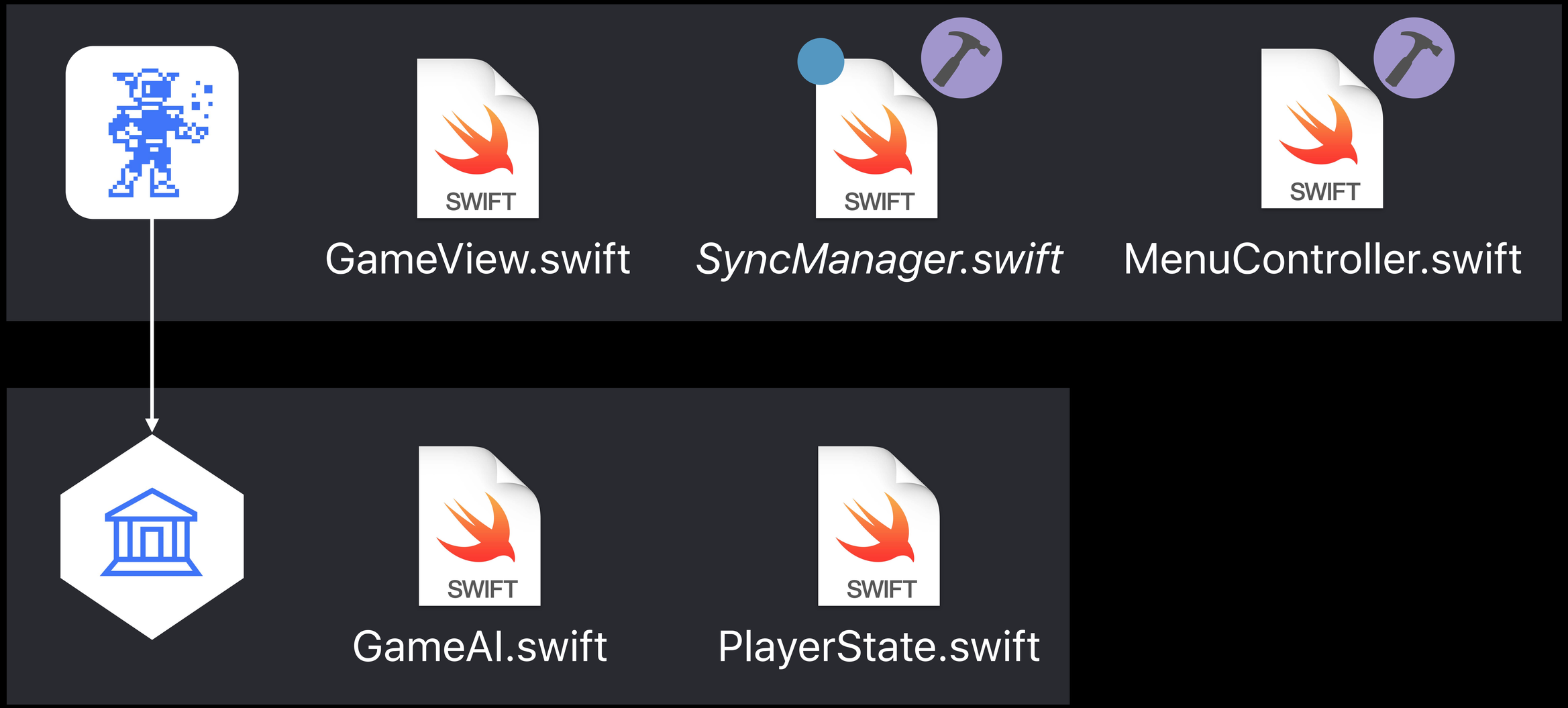
Dependencies Within a Target Are Per-File



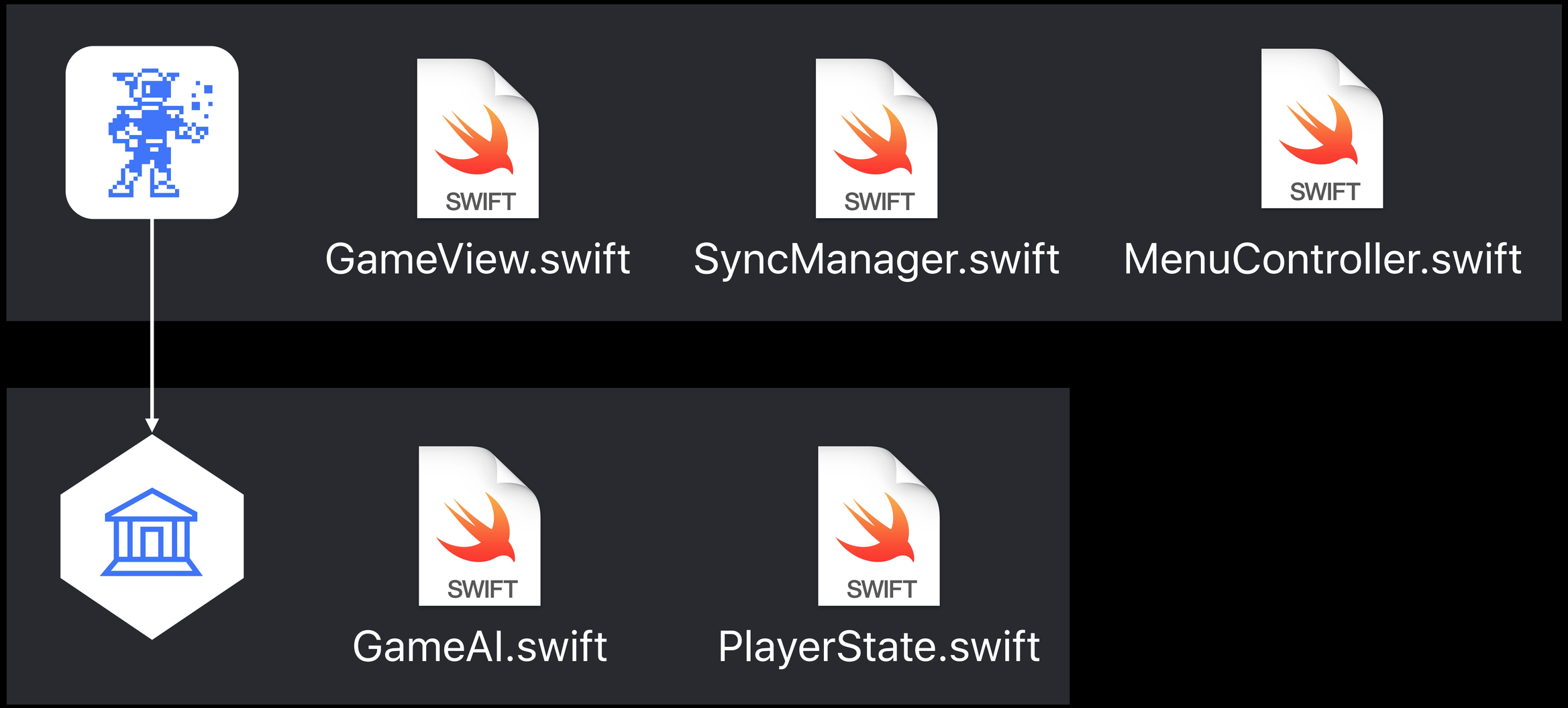
Dependencies Within a Target Are Per-File



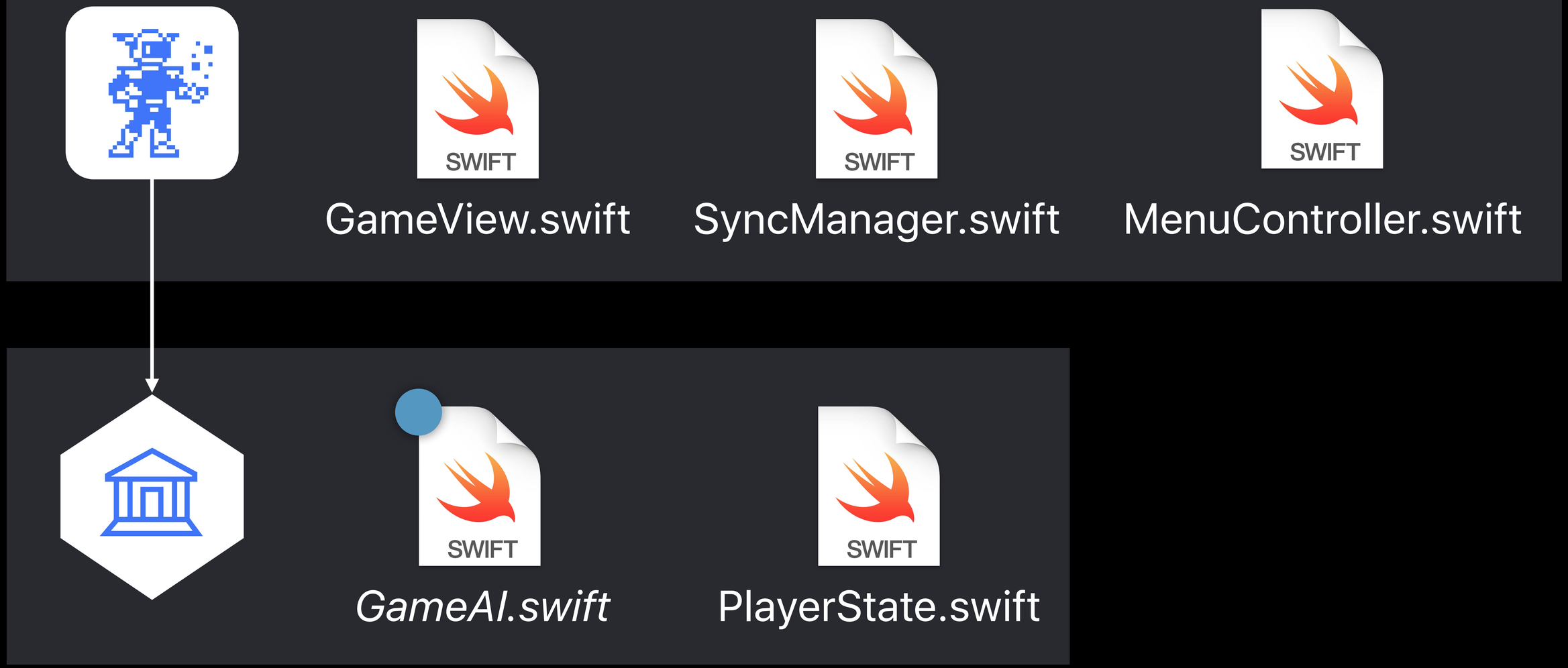
Dependencies Within a Target Are Per-File



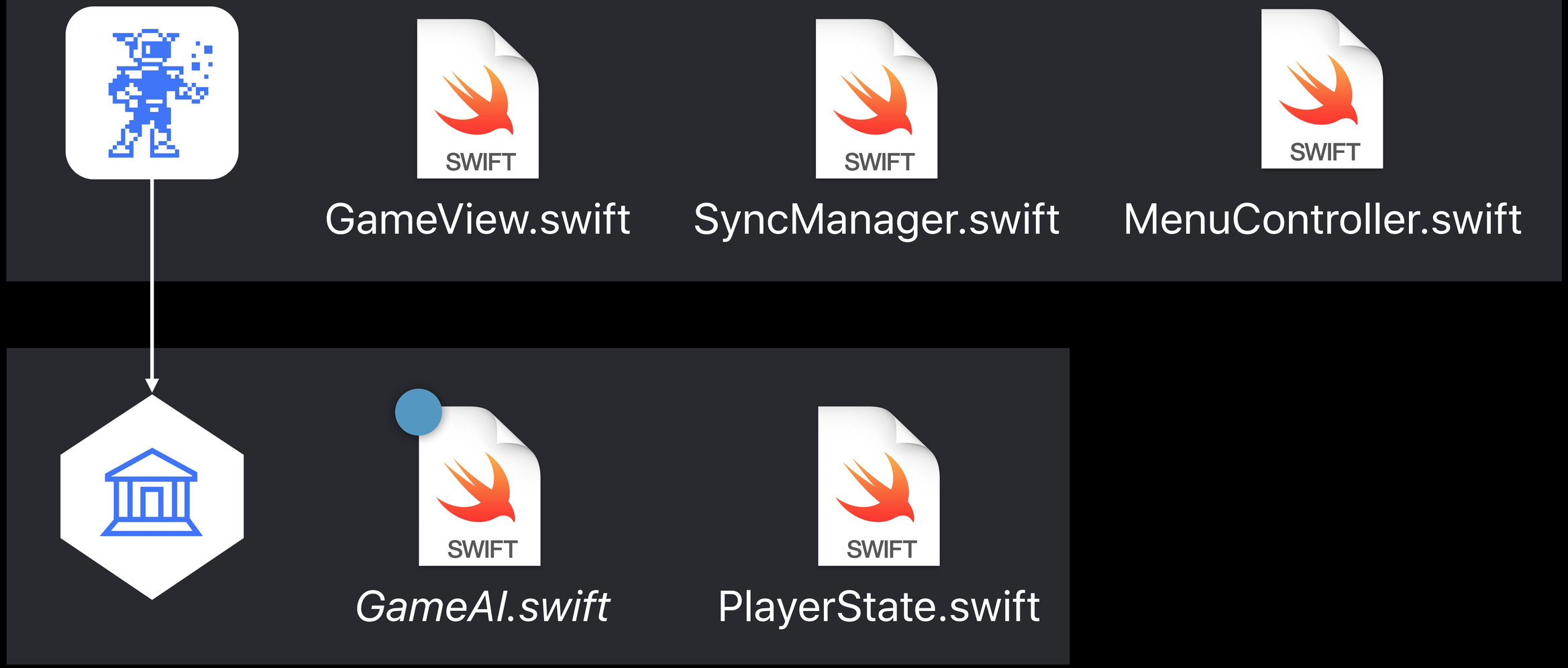
Cross-Target Dependencies Are Coarse-Grained



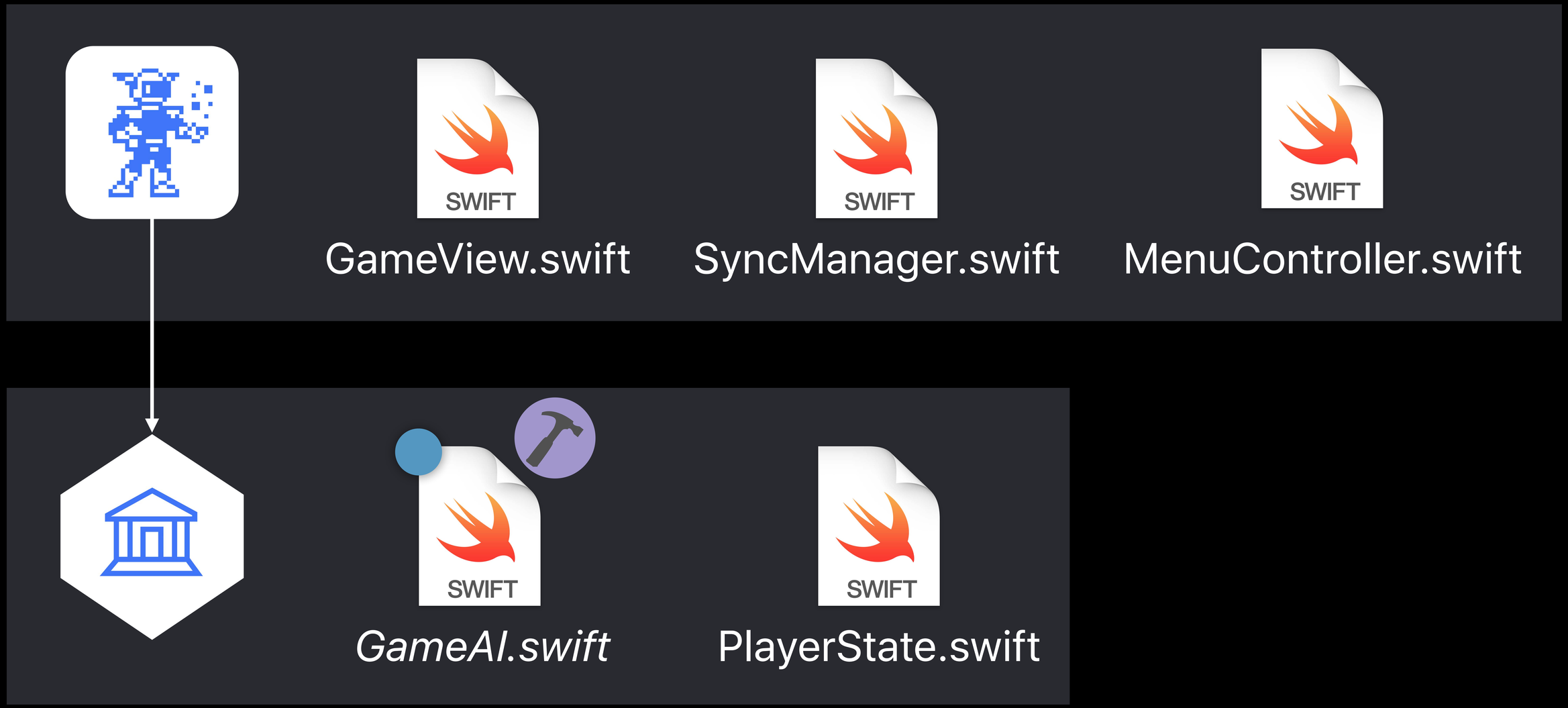
Cross-Target Dependencies Are Coarse-Grained



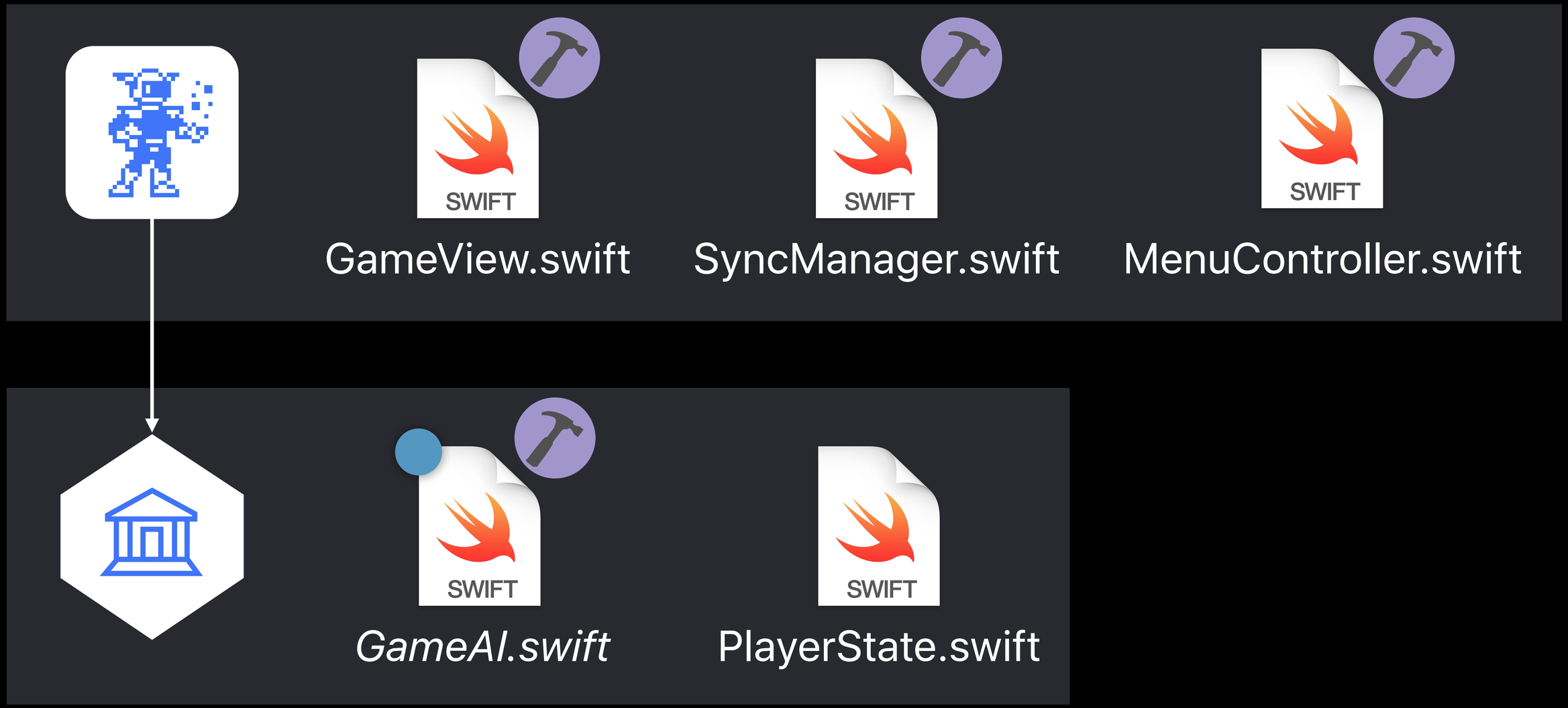
Cross-Target Dependencies Are Coarse-Grained



Cross-Target Dependencies Are Coarse-Grained



Cross-Target Dependencies Are Coarse-Grained



Swift Dependency Rules

Swift Dependency Rules

Compiler must be conservative

Swift Dependency Rules

Compiler must be conservative

Changes in function bodies do not affect the file’s interface

Swift Dependency Rules

Compiler must be conservative

Changes in function bodies do not affect the file’s interface

Dependencies within a module are per-file

Swift Dependency Rules

Compiler must be conservative

Changes in function bodies do not affect the file’s interface

Dependencies within a module are per-file

Dependencies across targets are for the whole target

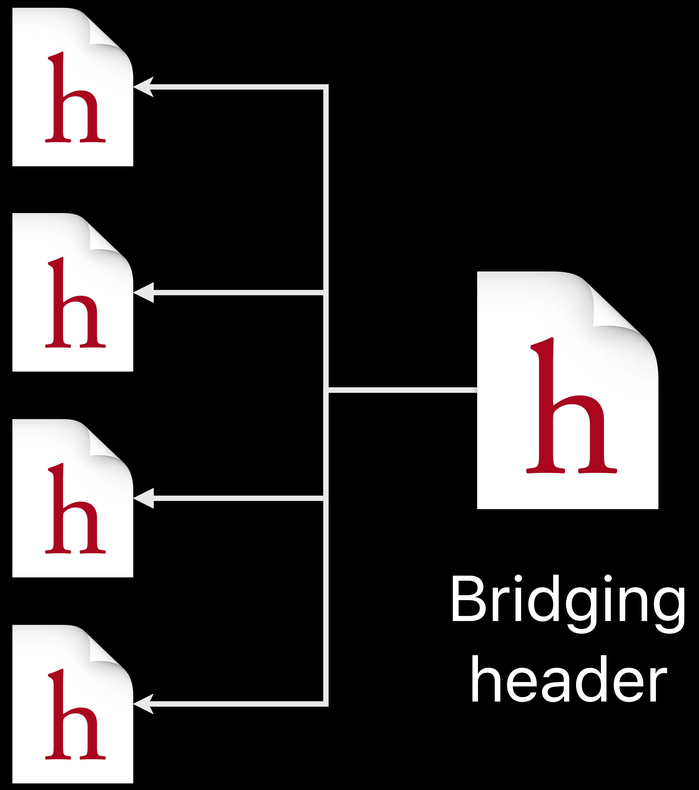
Limiting Your Objective-C/Swift Interface

Reducing the work on rebuilds

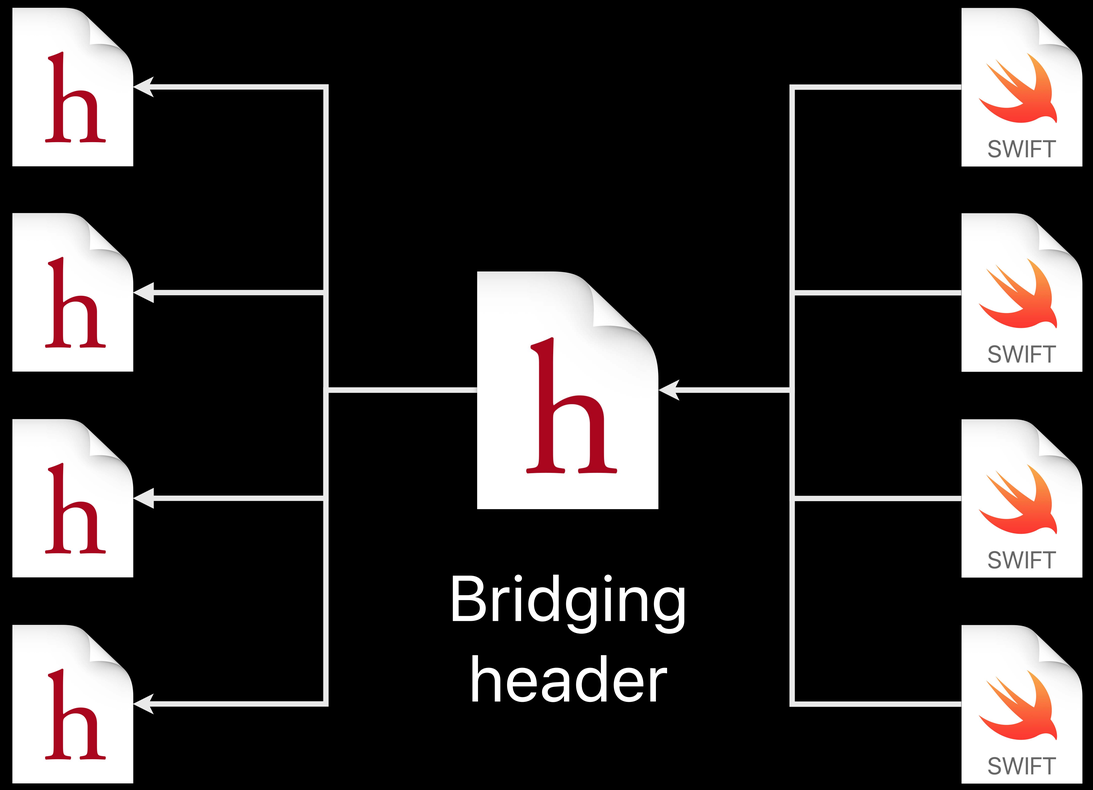
Mixed-Source App Targets

Mixed-Source App Targets

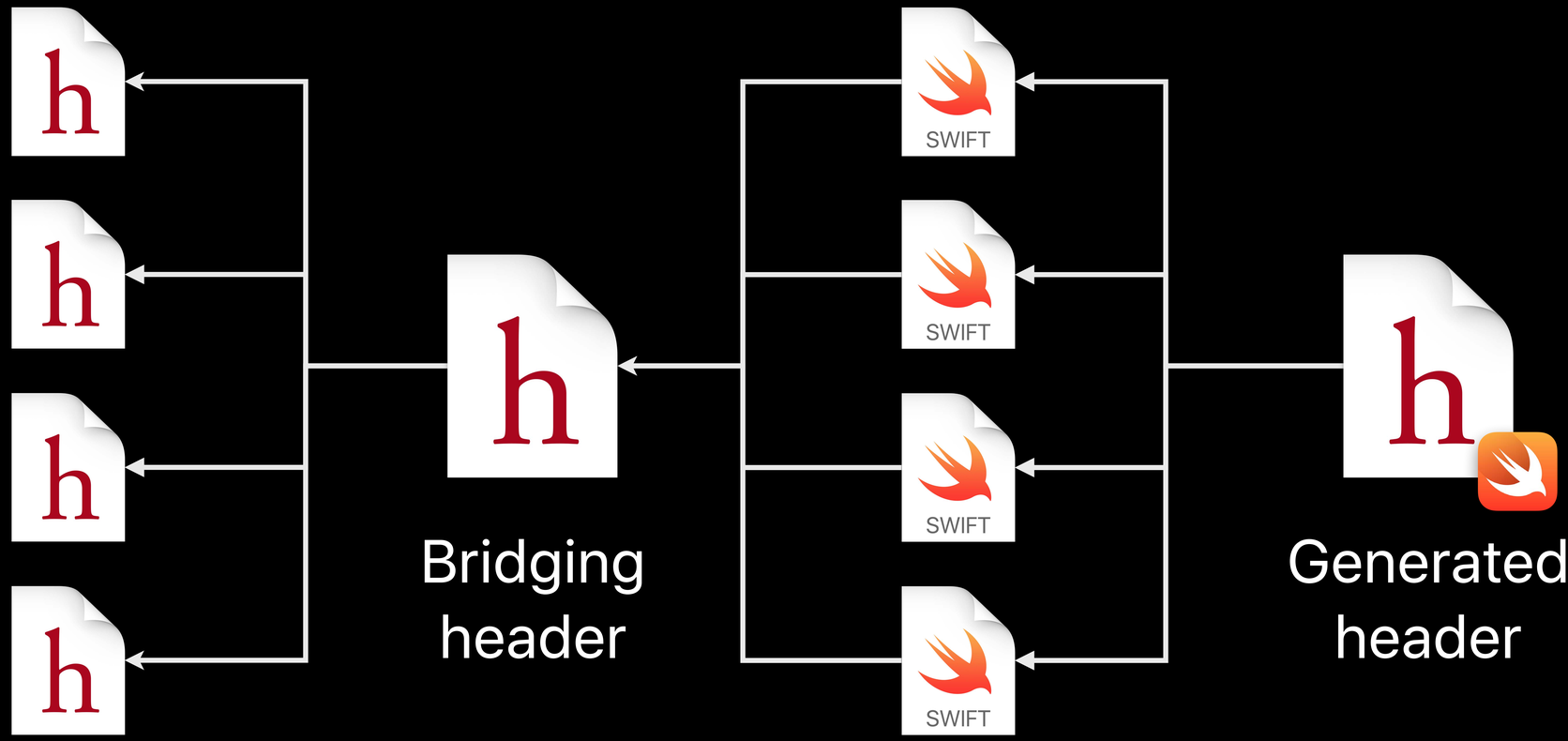
Mixed-Source App Targets



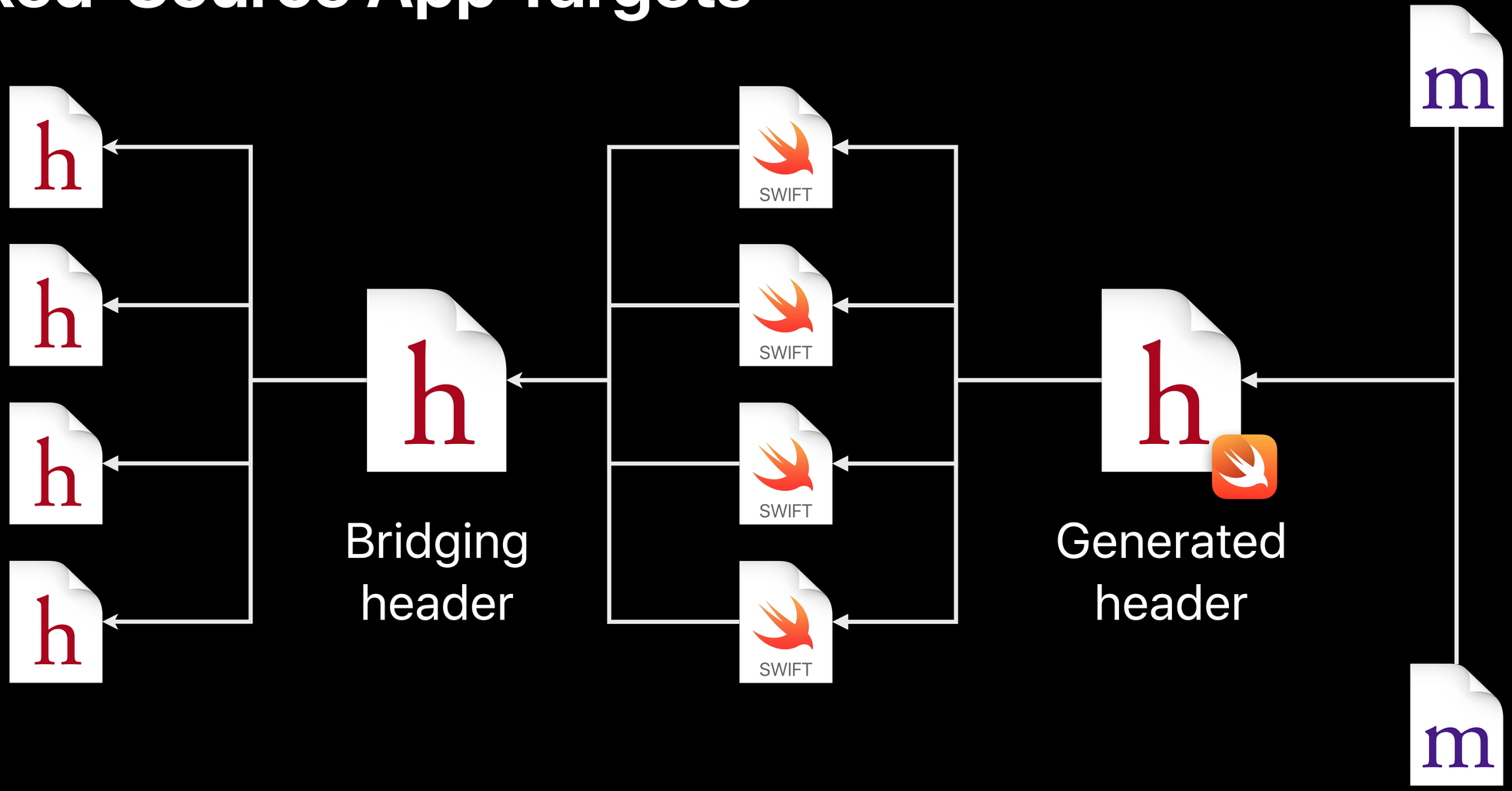
Mixed-Source App Targets



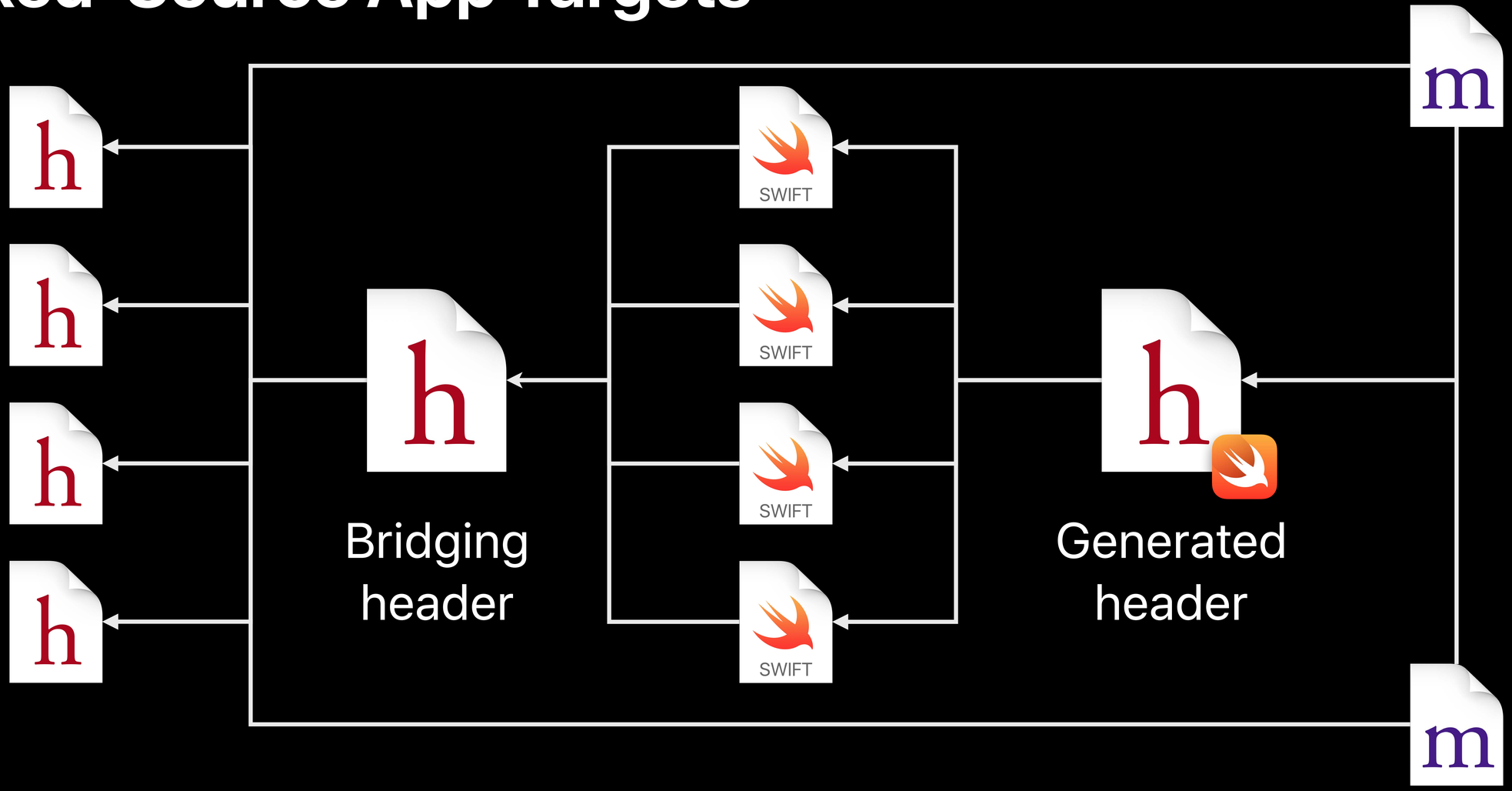
Mixed-Source App Targets



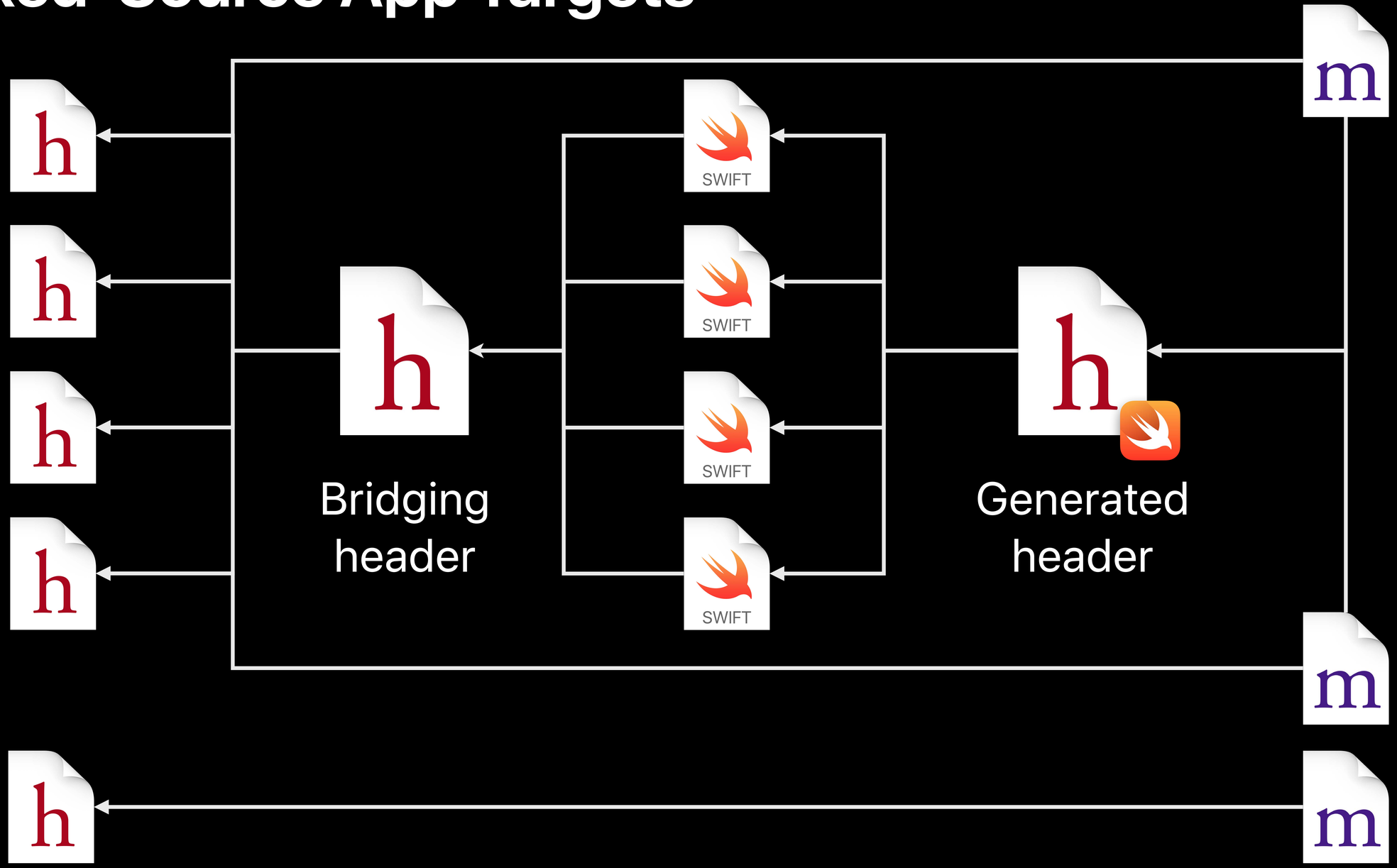
Mixed-Source App Targets



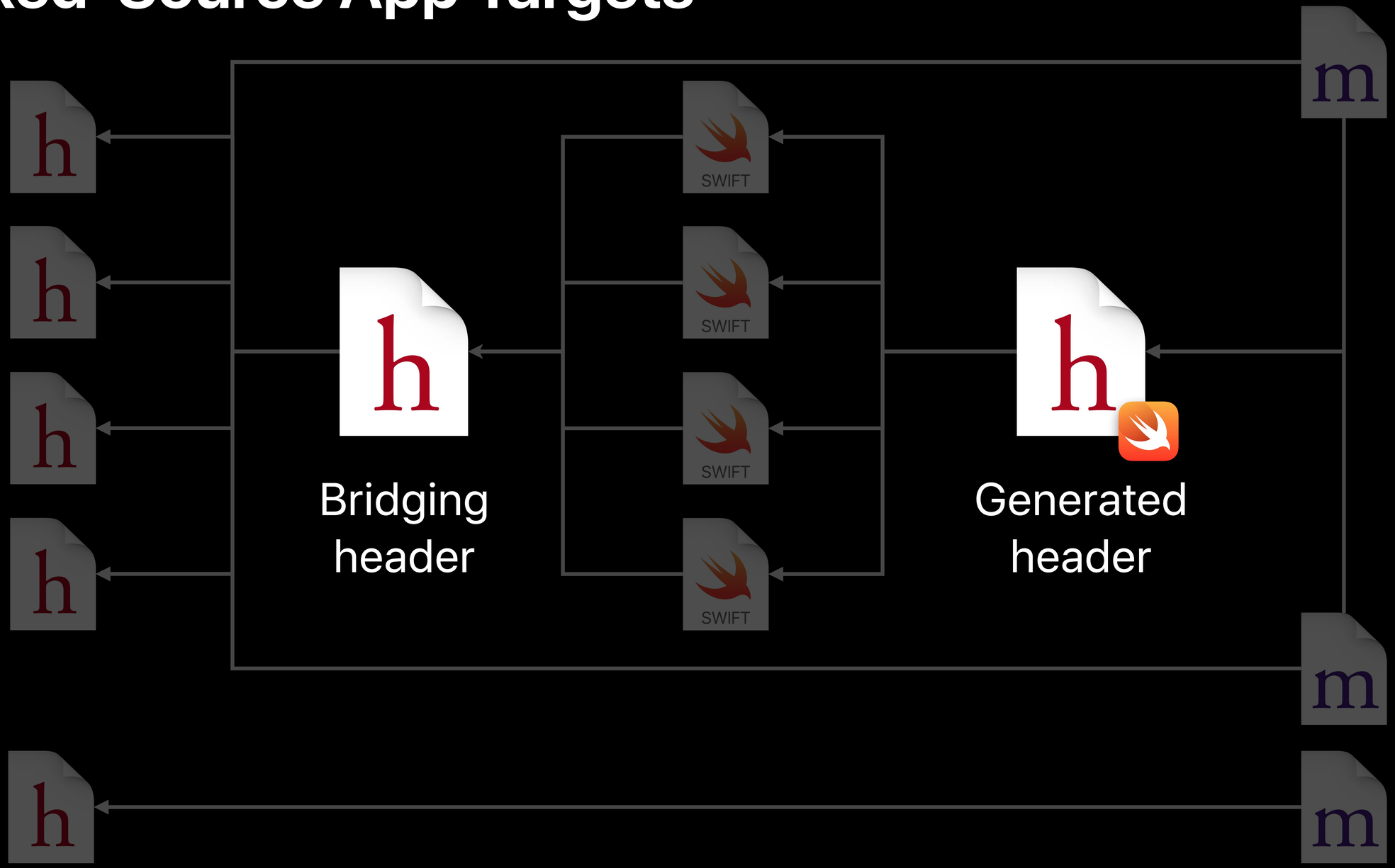
Mixed-Source App Targets



Mixed-Source App Targets



Mixed-Source App Targets



Keep Your Generated Header Minimal

Use `private` when possible

```
class MainViewController: UIViewController {
  @IBOutlet var statusField: UITextField!
  @IBAction func close(_ sender: Any?) { … }
}
```

Keep Your Generated Header Minimal

Use `private` when possible

```swift
class MainViewController: UIViewController {
  @IBOutlet var statusField: UITextField!
  @IBAction func close(_ sender: Any?) { … }
}
```


```objc
@property IBOutlet UITextField *statusField;
- (IBAction)close:(id)sender;
```

Keep Your Generated Header Minimal

Use `private` when possible

```swift
class MainViewController: UIViewController {
  @IBOutlet private var statusField: UITextField!
  @IBAction private func close(_ sender: Any?) { … }
}
```


```objc
@property IBOutlet UITextField *statusField;
- (IBAction)close:(id)sender;
```

Keep Your Generated Header Minimal

Use `private` when possible

```
class MainViewController: UIViewController {
  @IBOutlet private var statusField: UITextField!
  @IBAction private func close(_ sender: Any?) { … }
}
```

Keep Your Generated Header Minimal

Use `private` when possible

```
class MainViewController: UIViewController {
  @IBOutlet private var statusField: UITextField!
  @IBAction private func close(_ sender: Any?) { … }
}
```

Keep Your Generated Header Minimal

Use `private` when possible

```
@objc func keyboardWillShow(_: Notification) {
  // Important keyboard setup code here.
}

// …
NotificationCenter.default.addObserver(self, selector: #selector(keyboardWillShow(_:)), …)
```

Keep Your Generated Header Minimal

Use `private` when possible

```
@objc private func keyboardWillShow(_: Notification) {
  // Important keyboard setup code here.
}

// …
NotificationCenter.default.addObserver(self, selector: #selector(keyboardWillShow(_:)), …)
```

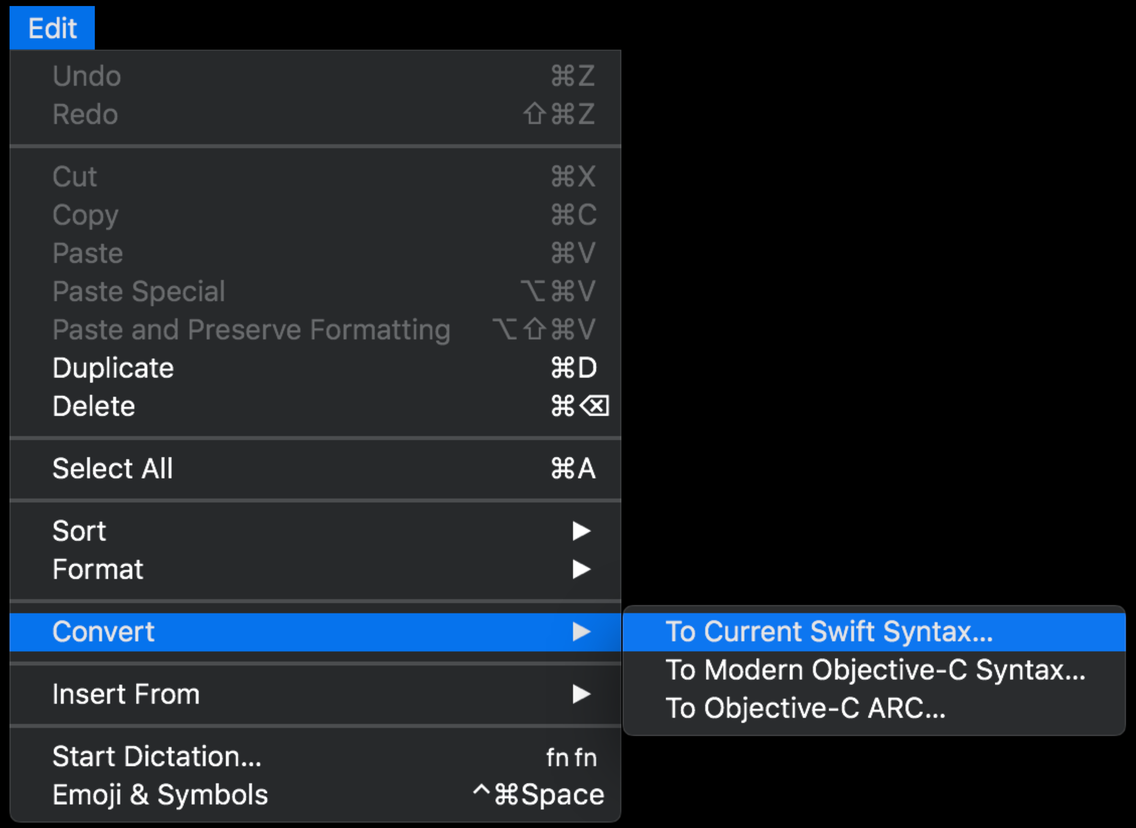
Keep Your Generated Header Minimal

Use block-based APIs

```
self.observer = NotificationCenter.default.addObserver(
    forName: UIKeyboardWillShow, object: nil, queue: nil) {
  // Important keyboard setup code here.
}
```

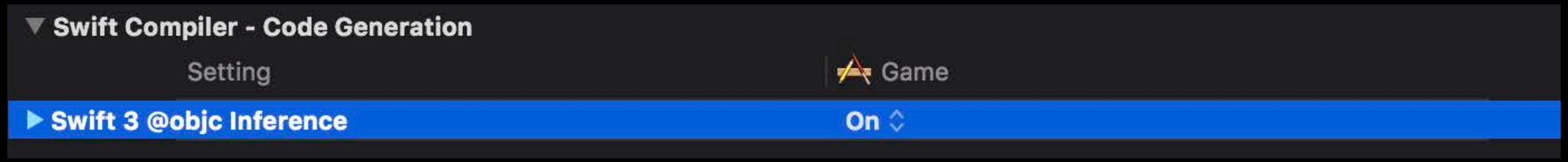
Keep Your Generated Header Minimal

Migrate to Swift 4



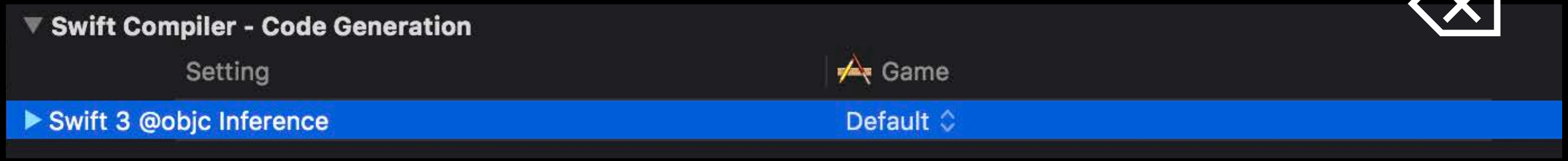
Keep Your Generated Header Minimal

Turn off “Swift 3 @objc Inference”



Keep Your Generated Header Minimal

Turn off “Swift 3 @objc Inference”



Keep Your Bridging Header Minimal

```
// Use this file to import your target's public headers that
// you would like to expose to Swift.

#import "MyBackendAPI.h"
#import "MyViewController.h"
#import "MyUserDefaultsHelpers.h"
```

MyApp-Bridging-Header.h

Keep Your Bridging Header Minimal

```
// Use this file to import your target's public headers that
// you would like to expose to Swift.

#import "MyBackendAPI.h"
#import "MyViewController.h"
#import "MyUserDefaultsHelpers.h"
```

MyApp-Bridging-Header.h

Keep Your Bridging Header Minimal

```
#import <UIKit/UIKit.h>
#import "MyNetworkManager.h"

@interface MyViewController: UIViewController
@property (nonnull) MyNetworkManager *networkManager;
// …
@end
```

Keep Your Bridging Header Minimal

```
#import <UIKit/UIKit.h>
#import "MyNetworkManager.h"

@interface MyViewController: UIViewController
@property (nonnull) MyNetworkManager *networkManager;
// …
@end
```

Keep Your Bridging Header Minimal

```
#import <UIKit/UIKit.h>
#import "MyNetworkManager.h"

@interface MyViewController: UIViewController
@property (nonnull) MyNetworkManager *networkManager;
// …
@end
```

Keep Your Bridging Header Minimal

Use categories to break up your interface

```
#import <UIKit/UIKit.h>
#import "MyNetworkManager.h"
@interface MyViewController: UIViewController
@property (nonnull) MyNetworkManager *networkManager;
// …
@end
```

Keep Your Bridging Header Minimal

Use categories to break up your interface

```
#import <UIKit/UIKit.h>
#import "MyNetworkManager.h"
@interface MyViewController: UIViewController
@property (nonnull) MyNetworkManager *networkManager;
// …
@end
```

MyViewController.h

```
#import "MyViewController.h"
@interface MyViewController ()
@end
```

MyViewController+Internal.h

Keep Your Bridging Header Minimal

Use categories to break up your interface

```
#import <UIKit/UIKit.h>
#import "MyNetworkManager.h"
@interface MyViewController: UIViewController
@property (nonnull) MyNetworkManager *networkManager;
// …
@end
```

MyViewController.h

```
#import "MyViewController.h"
@interface MyViewController ()
@end
```

MyViewController+Internal.h

Keep Your Bridging Header Minimal

Use categories to break up your interface

```
#import <UIKit/UIKit.h>
@interface MyViewController: UIViewController
// …
@end
```

MyViewController.h

```
#import "MyNetworkManager.h"
#import "MyViewController.h"
@interface MyViewController ()
@property (nonnull) MyNetworkManager *networkManager;
@end
```

MyViewController+Internal.h

Keep Your Bridging Header Minimal

Use categories to break up your interface

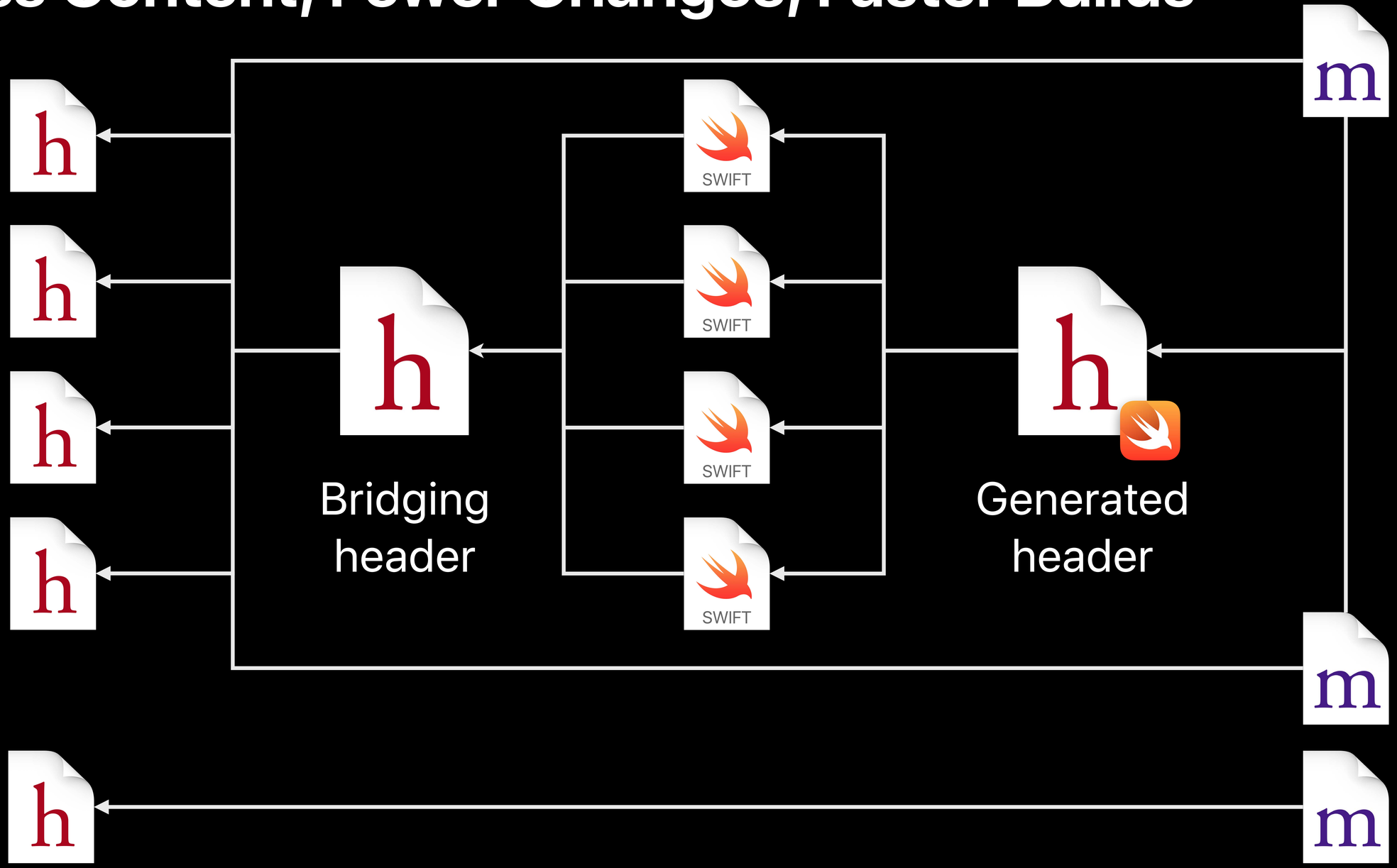
```
#import <UIKit/UIKit.h>
@interface MyViewController: UIViewController
// …
@end
```

MyViewController.h

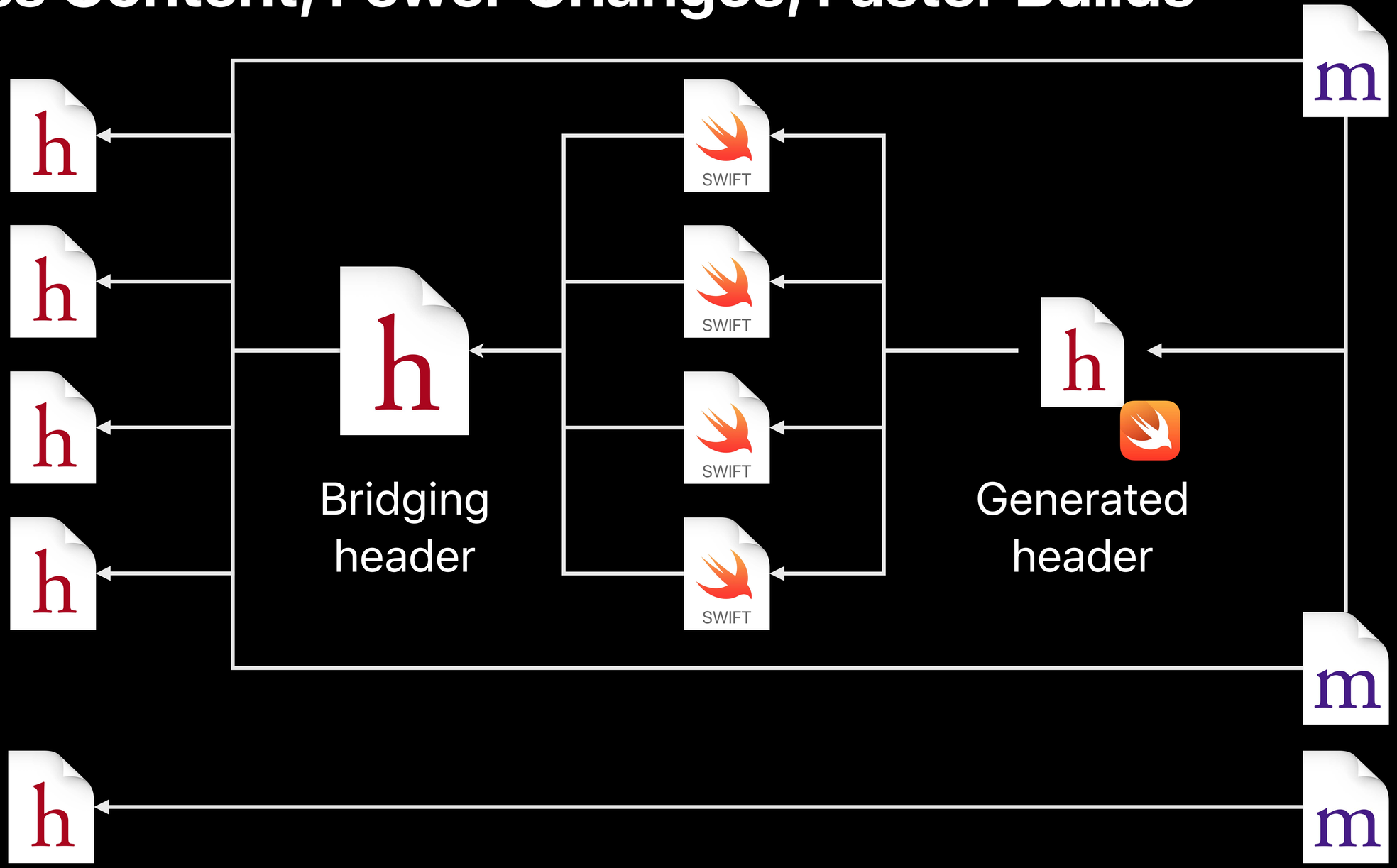
```
#import "MyNetworkManager.h"
#import "MyViewController.h"
@interface MyViewController ()
@property (nonnull) MyNetworkManager *networkManager;
@end
```

MyViewController.m

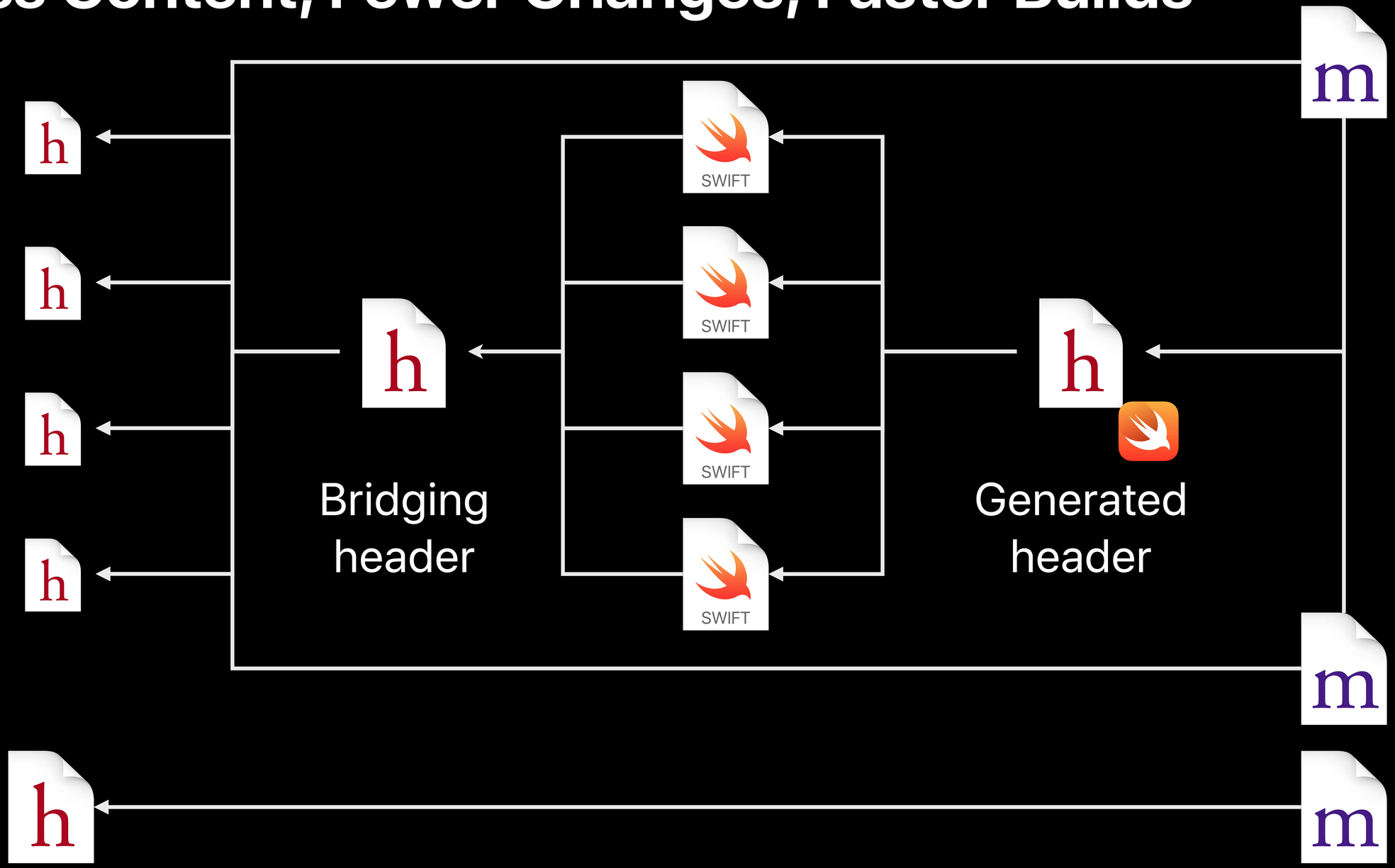
Less Content, Fewer Changes, Faster Builds



Less Content, Fewer Changes, Faster Builds



Less Content, Fewer Changes, Faster Builds



Summary

Building Faster in Xcode

Increasing Build Efficiency	Reducing the Work on Rebuilds
Parallelizing your build process	Declaring script inputs and outputs
Measuring your build time	Understanding dependencies in Swift
Dealing with complex expressions	Limiting your Objective-C/Swift interface

More Information

https://developer.apple.com/wwdc18/408

Building Your App with Xcode	Technology Lab 8	Thursday 12:00PM
Building Your App with Xcode	Technology Lab 8	Friday 3:00PM


WWDC18