#WWDC19

Testing in Xcode

Ana Calinov, Xcode Engineer
Stuart Montgomery, XCTest Engineer
Ethan Vaughan, XCTest Engineer

Introduction to XCTest

Test plans

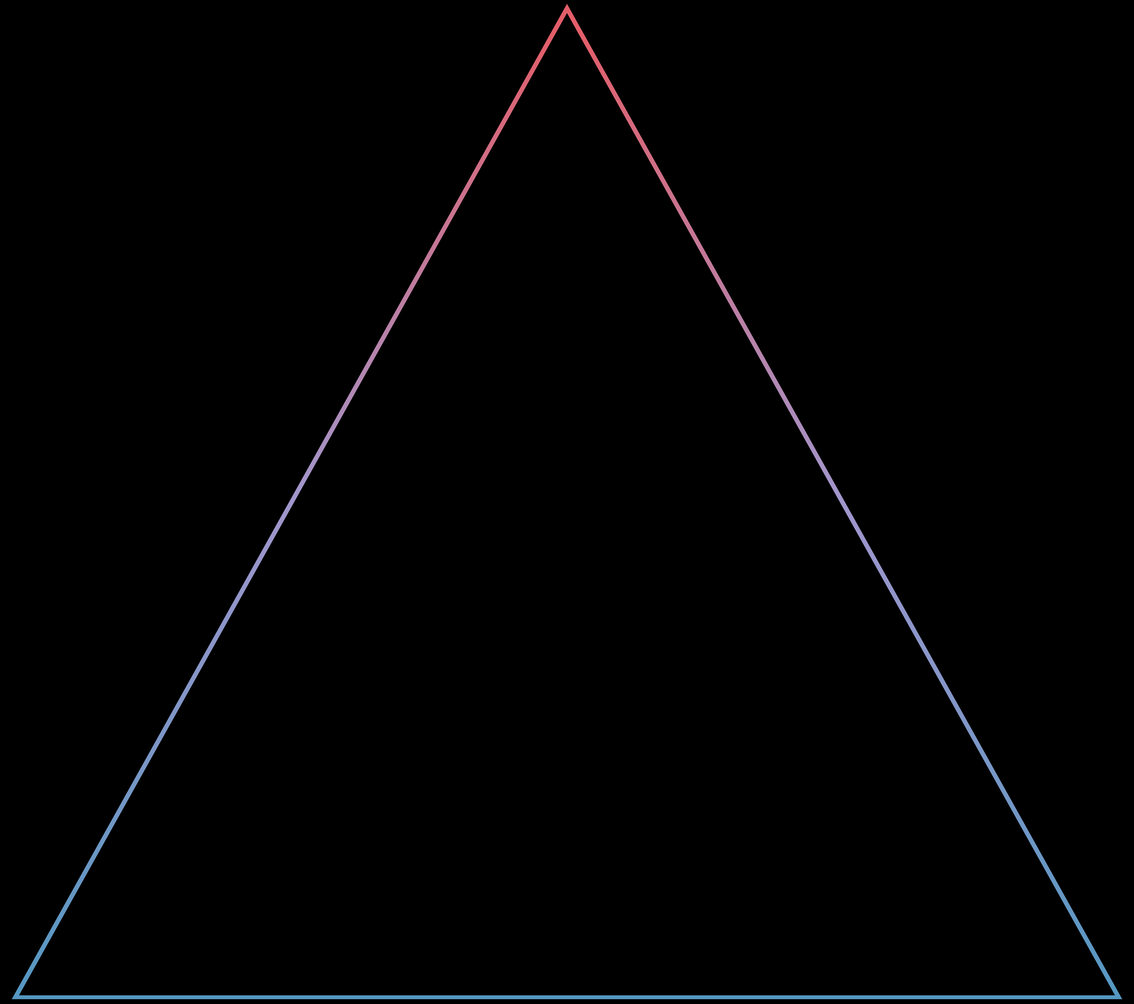
Continuous integration workflows

Introduction to XCTest

XCTest

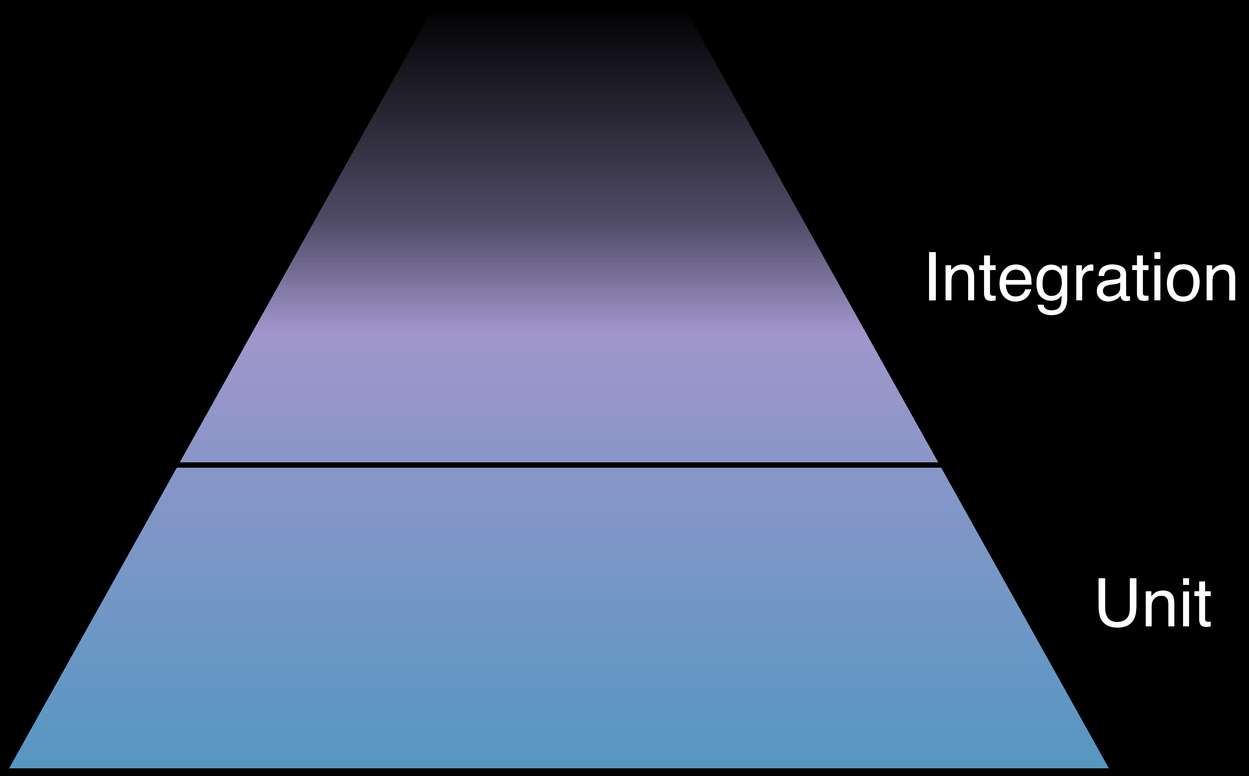
Test Pyramid

Testing fundamentals

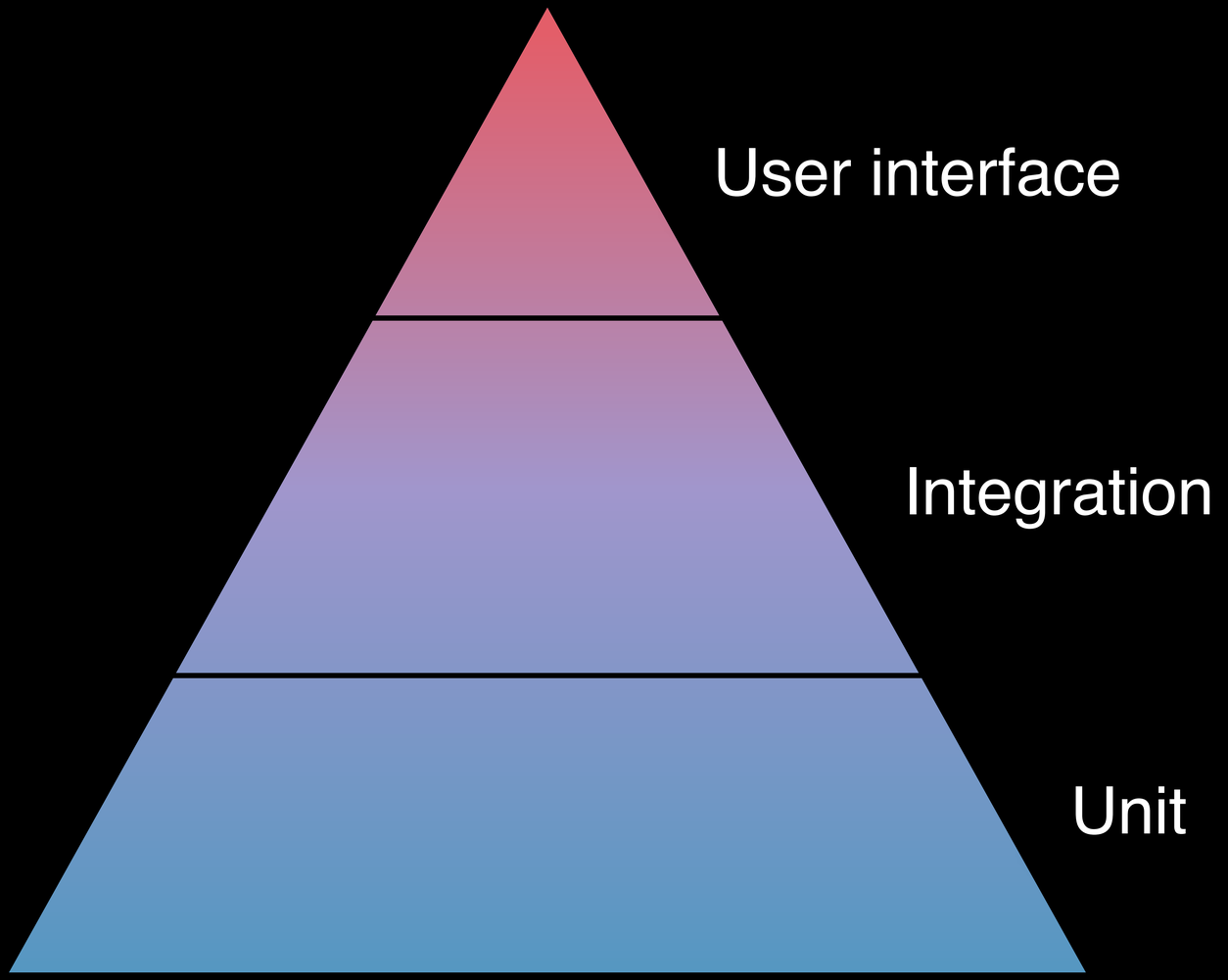


Test Pyramid

Test Pyramid



Test Pyramid



Types of Tests in XCTest

Types of Tests in XCTest

	Source code	User perspective	Workflow	Speed

Types of Tests in XCTest

	Source code	User perspective	Workflow	Speed
Unit Tests	✅			

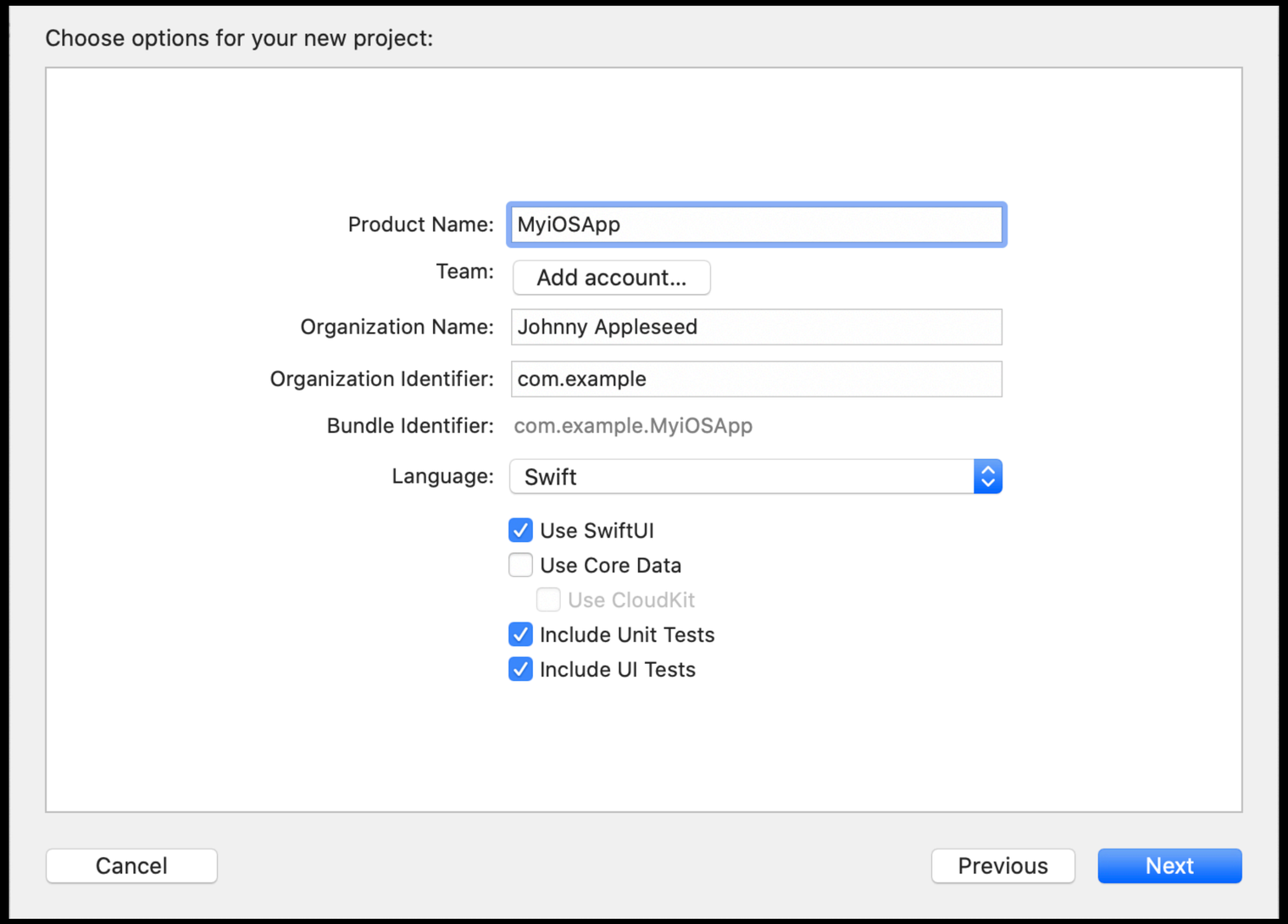
Types of Tests in XCTest

	Source code	User perspective	Workflow	Speed
Unit Tests	✅			
UI Tests		✅	✅	

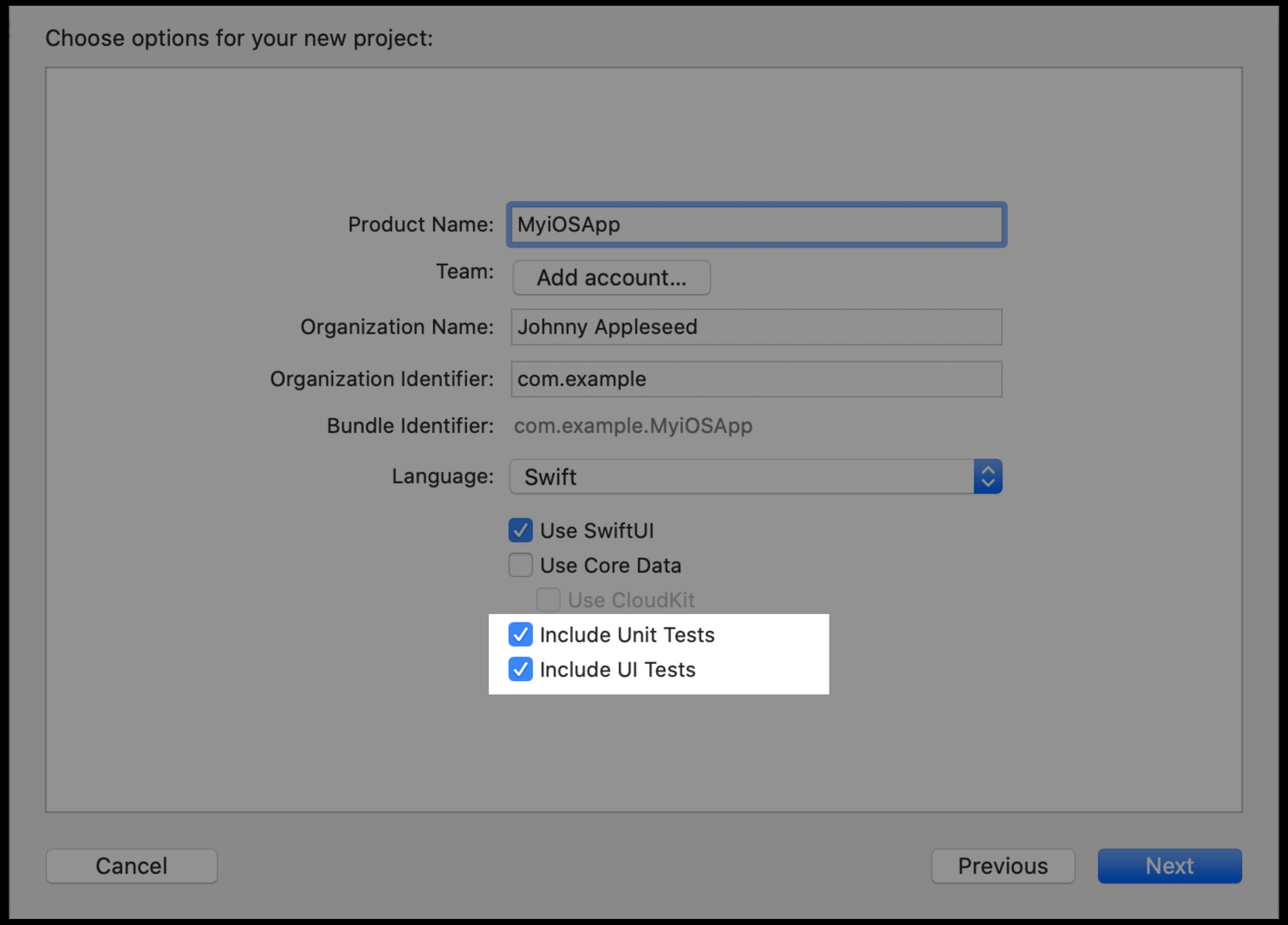
Types of Tests in XCTest

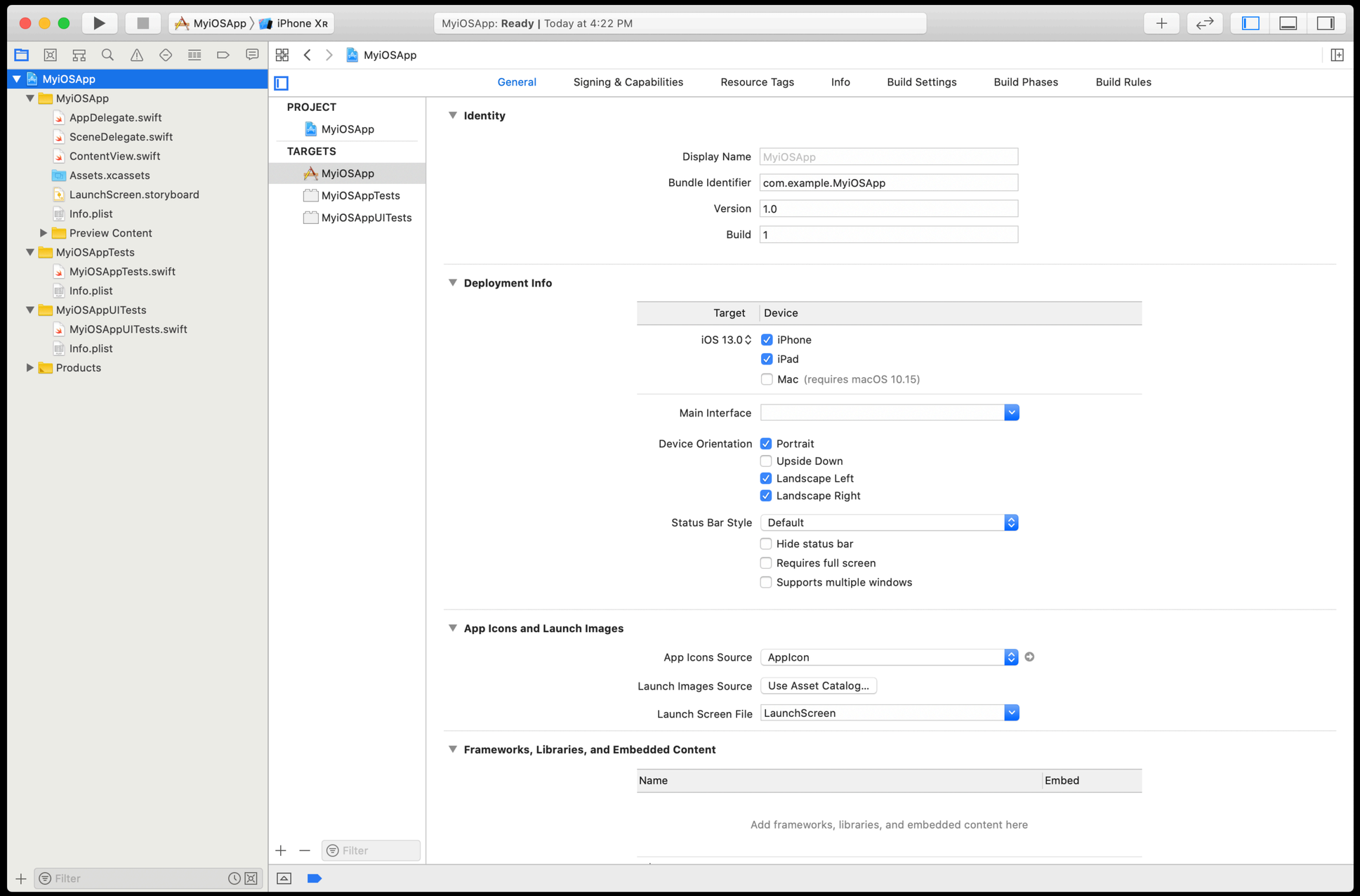
	Source code	User perspective	Workflow	Speed
Unit Tests	✅			
UI Tests		✅	✅	
Performance Tests	✅	✅		

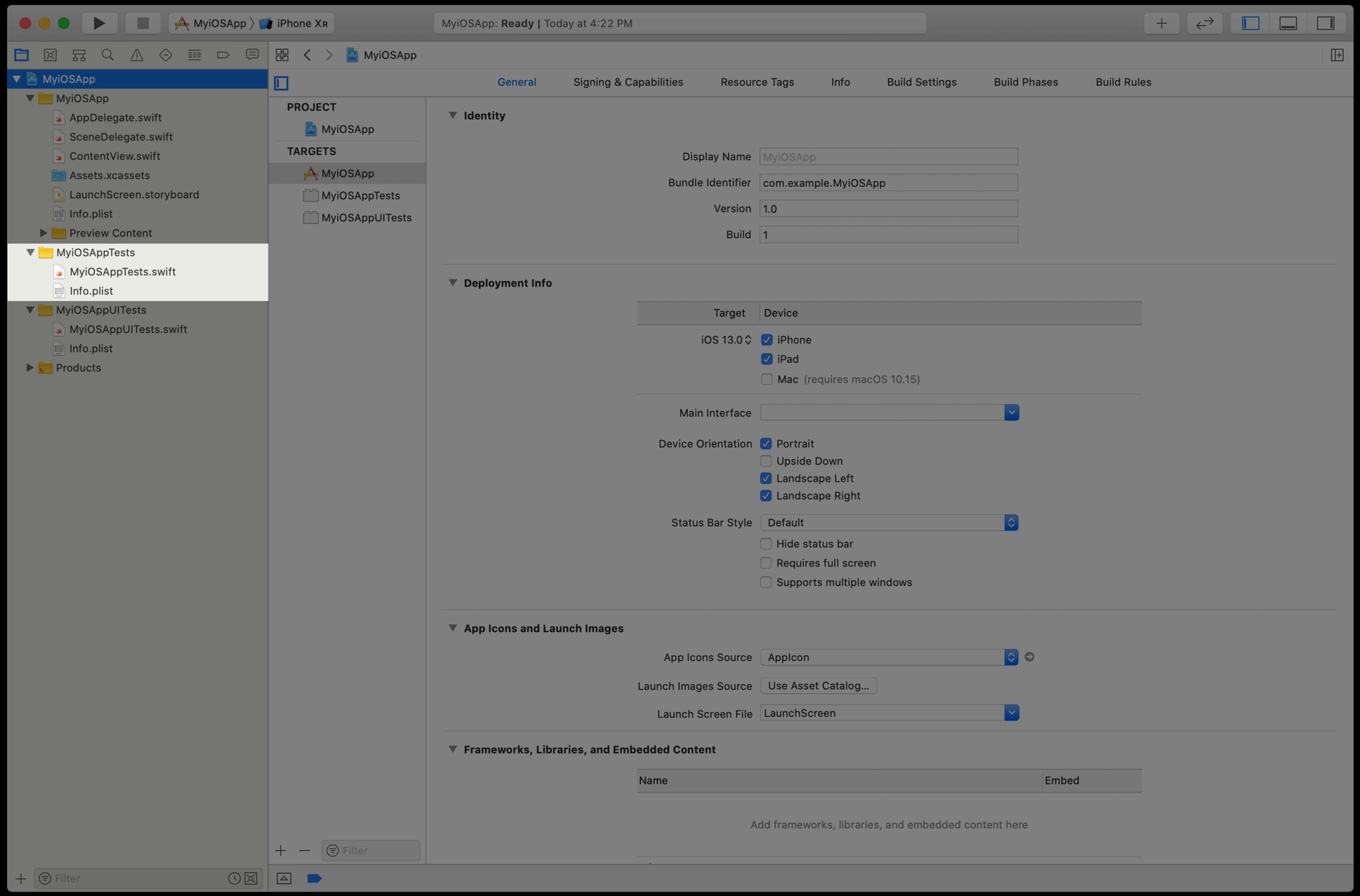
Testing from Scratch

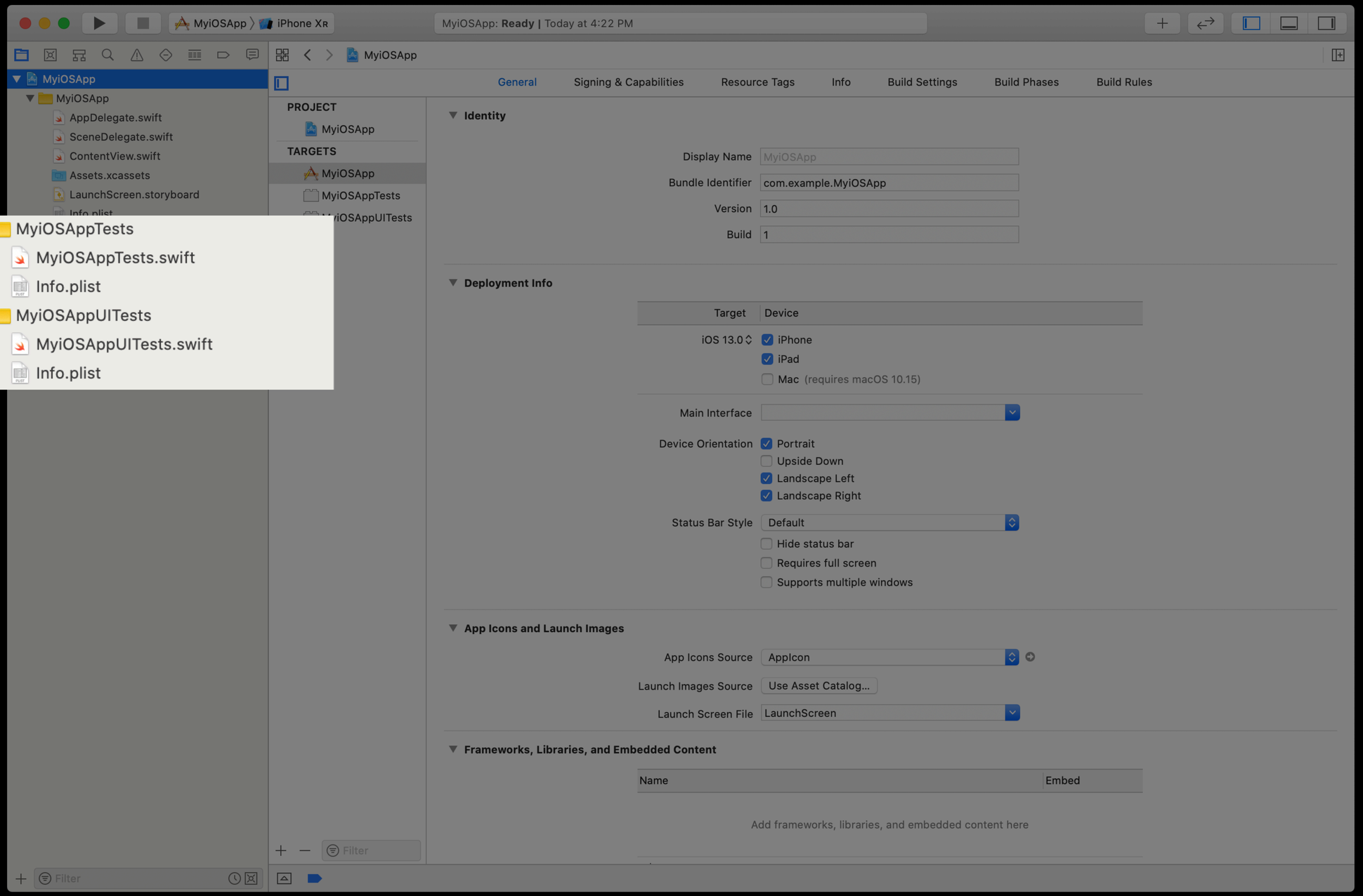


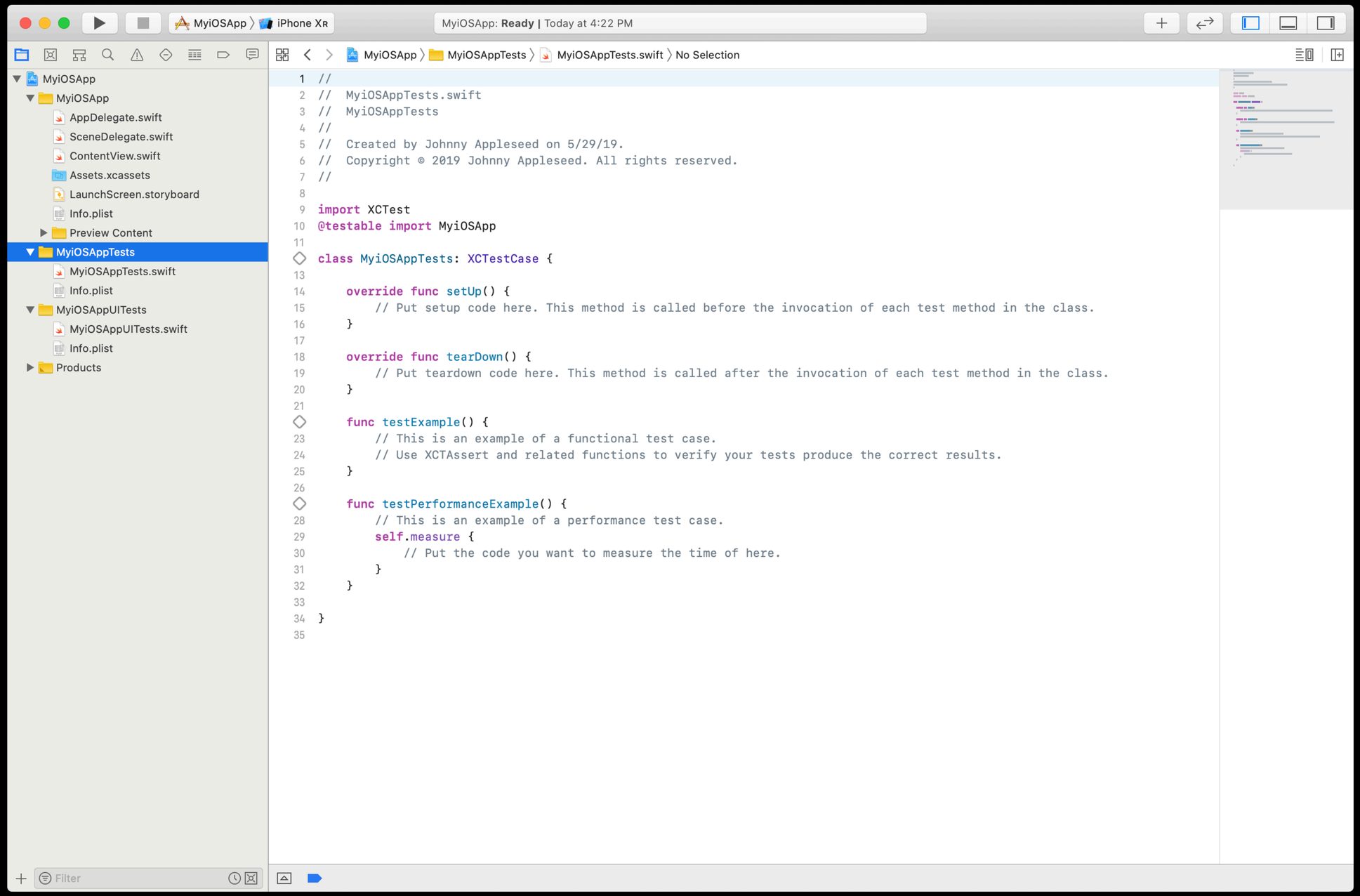
Testing from Scratch




MyiOSApp 〉 iPhone XR
MyiOSApp: Ready | Today at 4:22 PM
MyiOSApp
General
Signing & Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
MyiOSApp
TARGETS
MyiOSApp
MyiOSAppTests
MyiOSAppUITests
Identity
Display Name
MyiOSApp
Bundle Identifier
com.example.MyiOSApp
Version
1.0
Build
1
Deployment Info
Target
Device
iOS 13.0
iPhone
iPad
Mac (requires macOS 10.15)
Main Interface
Device Orientation
Portrait
Upside Down
Landscape Left
Landscape Right
Status Bar Style
Default
Hide status bar
Requires full screen
Supports multiple windows
App Icons and Launch Images
App Icons Source
AppIcon
Launch Images Source
Use Asset Catalog...
Launch Screen File
LaunchScreen
Frameworks, Libraries, and Embedded Content
Name
Embed
Add frameworks, libraries, and embedded content here
Filter
MyiOSApp
AppDelegate.swift
SceneDelegate.swift
ContentView.swift
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Preview Content
MyiOSAppTests
MyiOSAppTests.swift
Info.plist
MyiOSAppUITests
MyiOSAppUITests.swift
Info.plist
Products
Filter


MyiOSApp 〉 iPhone XR
MyiOSApp: Ready | Today at 4:22 PM
MyiOSApp
General
Signing & Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
MyiOSApp
AppDelegate.swift
SceneDelegate.swift
ContentView.swift
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Preview Content
MyiOSAppTests
MyiOSAppTests.swift
Info.plist
MyiOSAppUITests
MyiOSAppUITests.swift
Info.plist
Products
PROJECT
MyiOSApp
TARGETS
MyiOSApp
MyiOSAppTests
MyiOSAppUITests
Filter
Identity
Display Name
MyiOSApp
Bundle Identifier
com.example.MyiOSApp
Version
1.0
Build
1
Deployment Info
Target
Device
iOS 13.0
iPhone
iPad
Mac (requires macOS 10.15)
Main Interface
Device Orientation
Portrait
Upside Down
Landscape Left
Landscape Right
Status Bar Style
Default
Hide status bar
Requires full screen
Supports multiple windows
App Icons and Launch Images
App Icons Source
AppIcon
Launch Images Source
Use Asset Catalog...
Launch Screen File
LaunchScreen
Frameworks, Libraries, and Embedded Content
Name
Embed
Add frameworks, libraries, and embedded content here


MyiOSApp 〉 iPhone XR
MyiOSApp: Ready | Today at 4:22 PM
MyiOSApp
General
Signing & Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rules
PROJECT
MyiOSApp
TARGETS
MyiOSApp
MyiOSAppTests
MyiOSAppUITests
Identity
Display Name
MyiOSApp
Bundle Identifier
com.example.MyiOSApp
Version
1.0
Build
1
Deployment Info
Target
Device
iOS 13.0
iPhone
iPad
Mac (requires macOS 10.15)
Main Interface
Device Orientation
Portrait
Upside Down
Landscape Left
Landscape Right
Status Bar Style
Default
Hide status bar
Requires full screen
Supports multiple windows
App Icons and Launch Images
App Icons Source
AppIcon
Launch Images Source
Use Asset Catalog...
Launch Screen File
LaunchScreen
Frameworks, Libraries, and Embedded Content
Name
Embed
Add frameworks, libraries, and embedded content here
Filter
MyiOSApp
AppDelegate.swift
SceneDelegate.swift
ContentView.swift
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MyiOSAppTests
MyiOSAppTests.swift
Info.plist
MyiOSAppUITests
MyiOSAppUITests.swift
Info.plist
Filter


MyiOSApp 〉 iPhone XR
MyiOSApp: Ready | Today at 4:22 PM
MyiOSApp 〉 MyiOSAppTests 〉 MyiOSAppTests.swift 〉 No Selection
MyiOSApp
MyiOSApp
AppDelegate.swift
SceneDelegate.swift
ContentView.swift
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Preview Content
MyiOSAppTests
MyiOSAppTests.swift
Info.plist
MyiOSAppUITests
MyiOSAppUITests.swift
Info.plist
Products
Filter
//
// MyiOSAppTests.swift
// MyiOSAppTests
//
// Created by Johnny Appleseed on 5/29/19.
// Copyright © 2019 Johnny Appleseed. All rights reserved.
//
import XCTest
@testable import MyiOSApp
class MyiOSAppTests: XCTestCase {
override func setUp() {
// Put setup code here. This method is called before the invocation of each test method in the class.
}
override func tearDown() {
// Put teardown code here. This method is called after the invocation of each test method in the class.
}
func testExample() {
// This is an example of a functional test case.
// Use XCTAssert and related functions to verify your tests produce the correct results.
}
func testPerformanceExample() {
// This is an example of a performance test case.
self.measure {
// Put the code you want to measure the time of here.
}
}
}

```swift
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

```swift
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

```
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

```
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

```
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

```swift
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

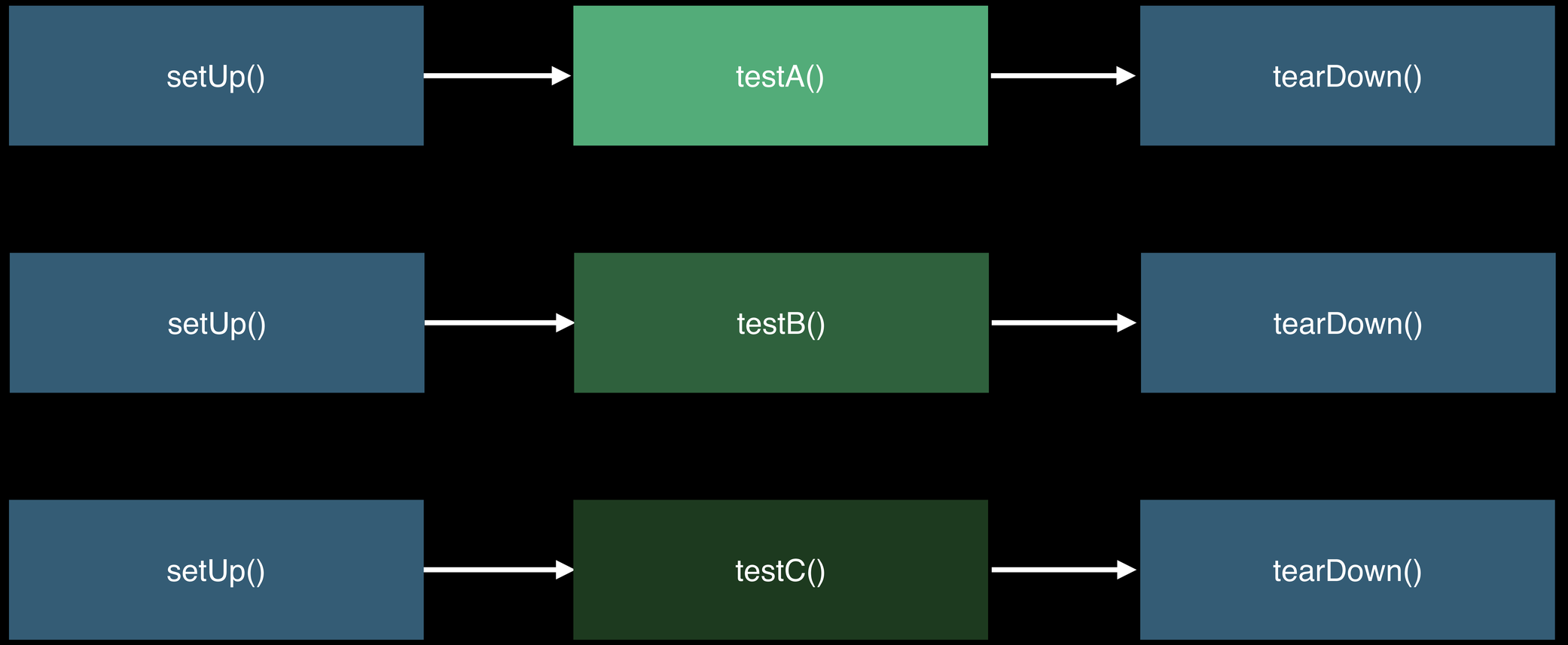
```swift
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

XCTAssertEqual failed: ("2") is not equal to ("1")
- Counter was not incremented:
<MyiOSApp.Counter: 0x600002dd1090>

```swift
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    func testIncrementCounter() {
        var counter = Counter()
        counter.increment()

        XCTAssertEqual(counter.value, 1,
                       "Counter was not incremented: \(counter)")
    }
}
```

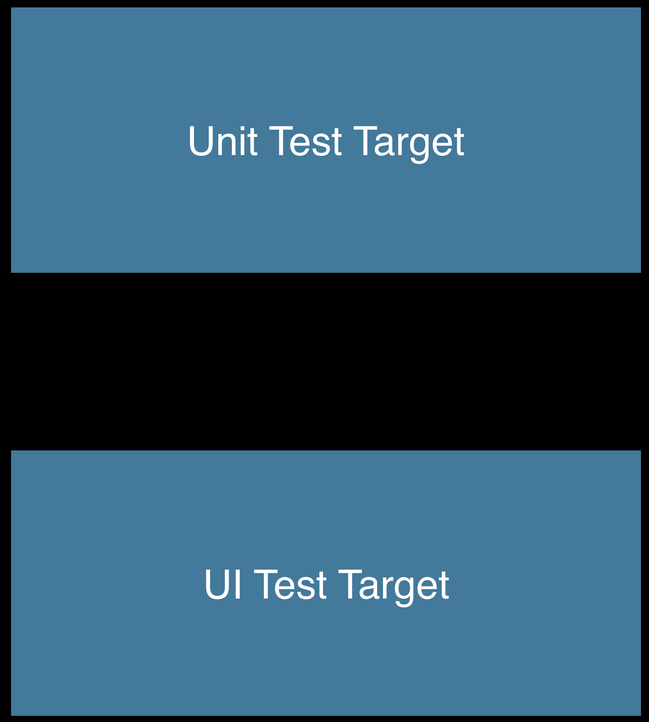
```
// How to use XCTest

import XCTest
@testable import MyCounterApp

class MyCounterTests: XCTestCase {

    override func setUp() {
        // Put setup code here.
    }

    override func tearDown() {
        // Put teardown code here.
    }
}
```

Test Execution Flow

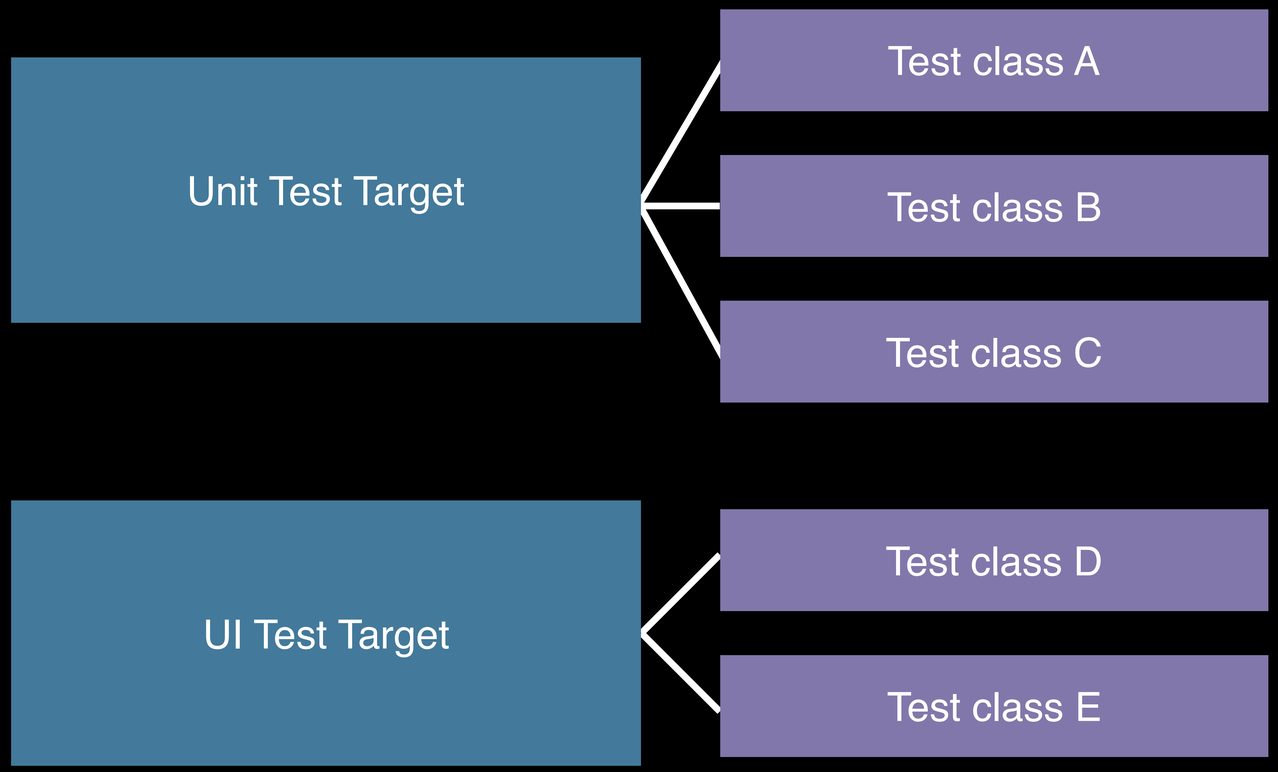


Demo

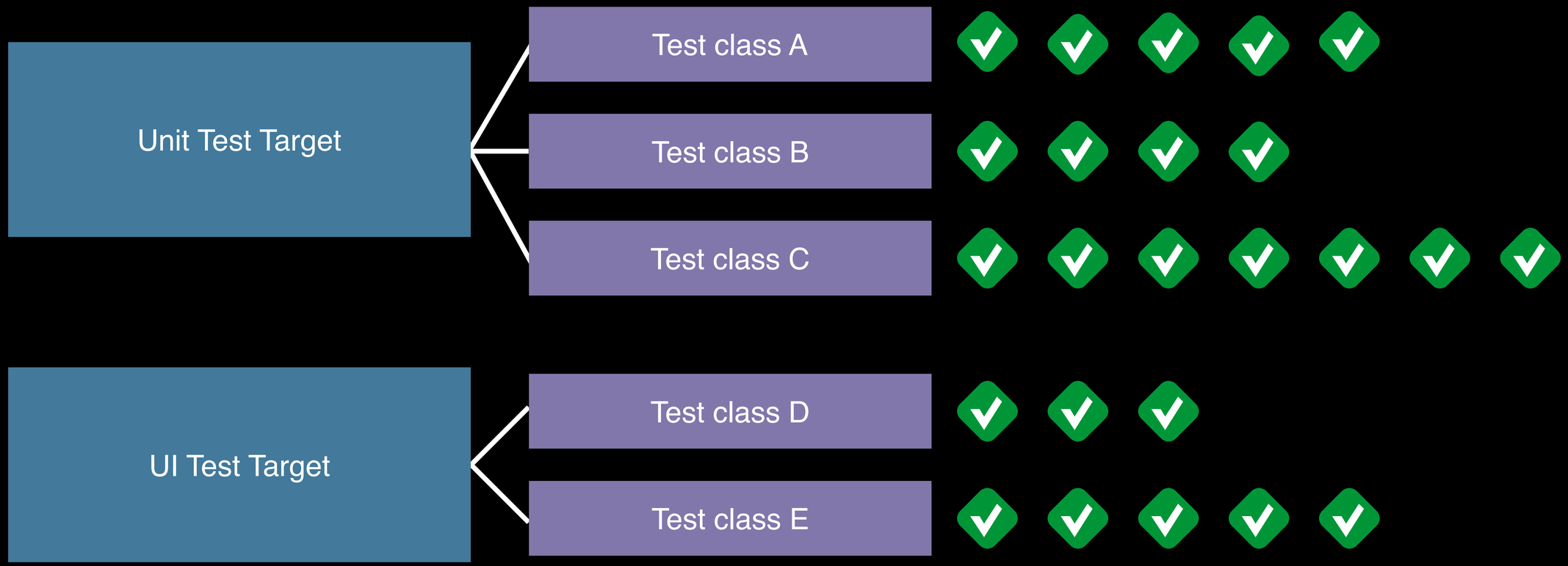
Test Organization



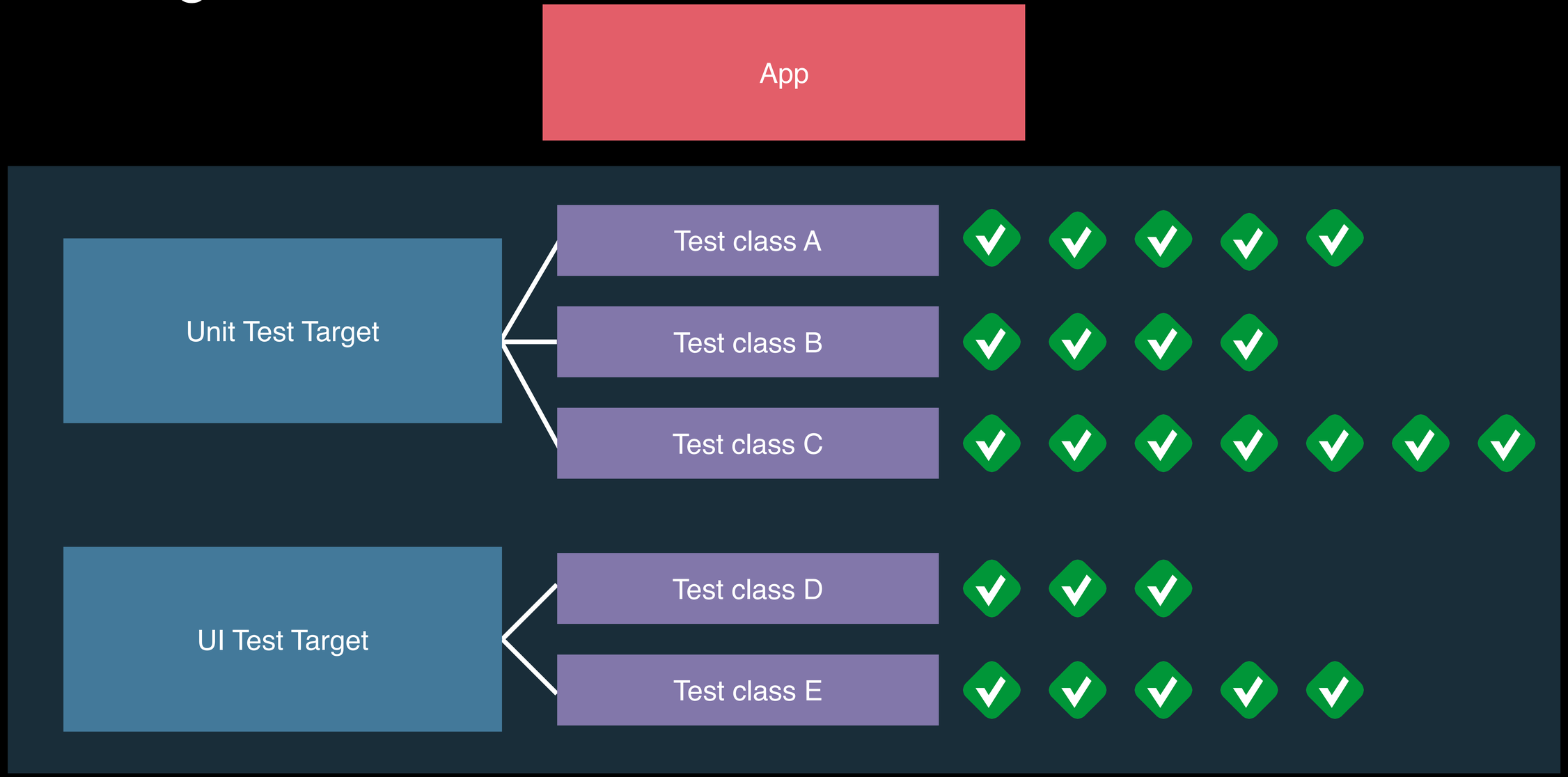
Test Organization



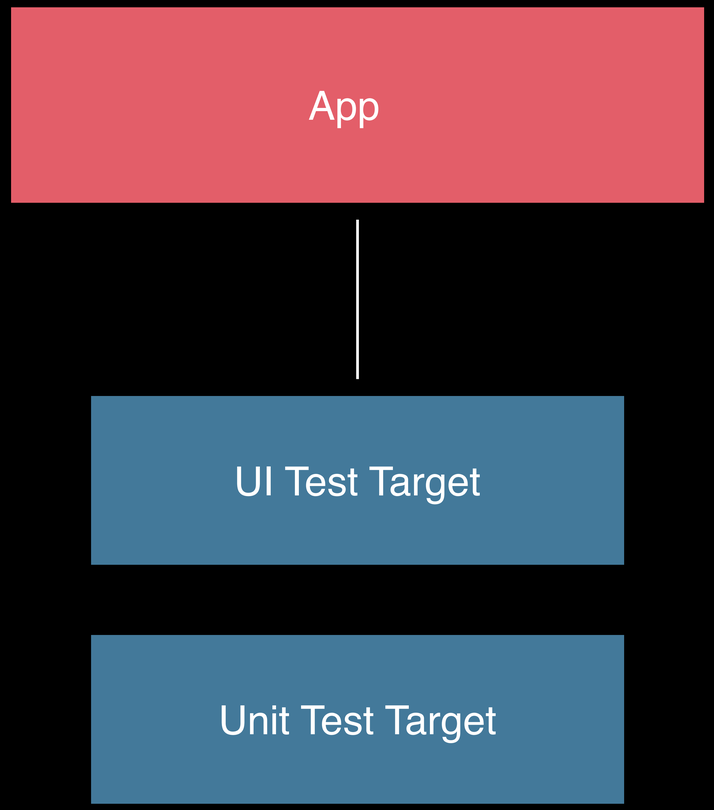
Test Organization



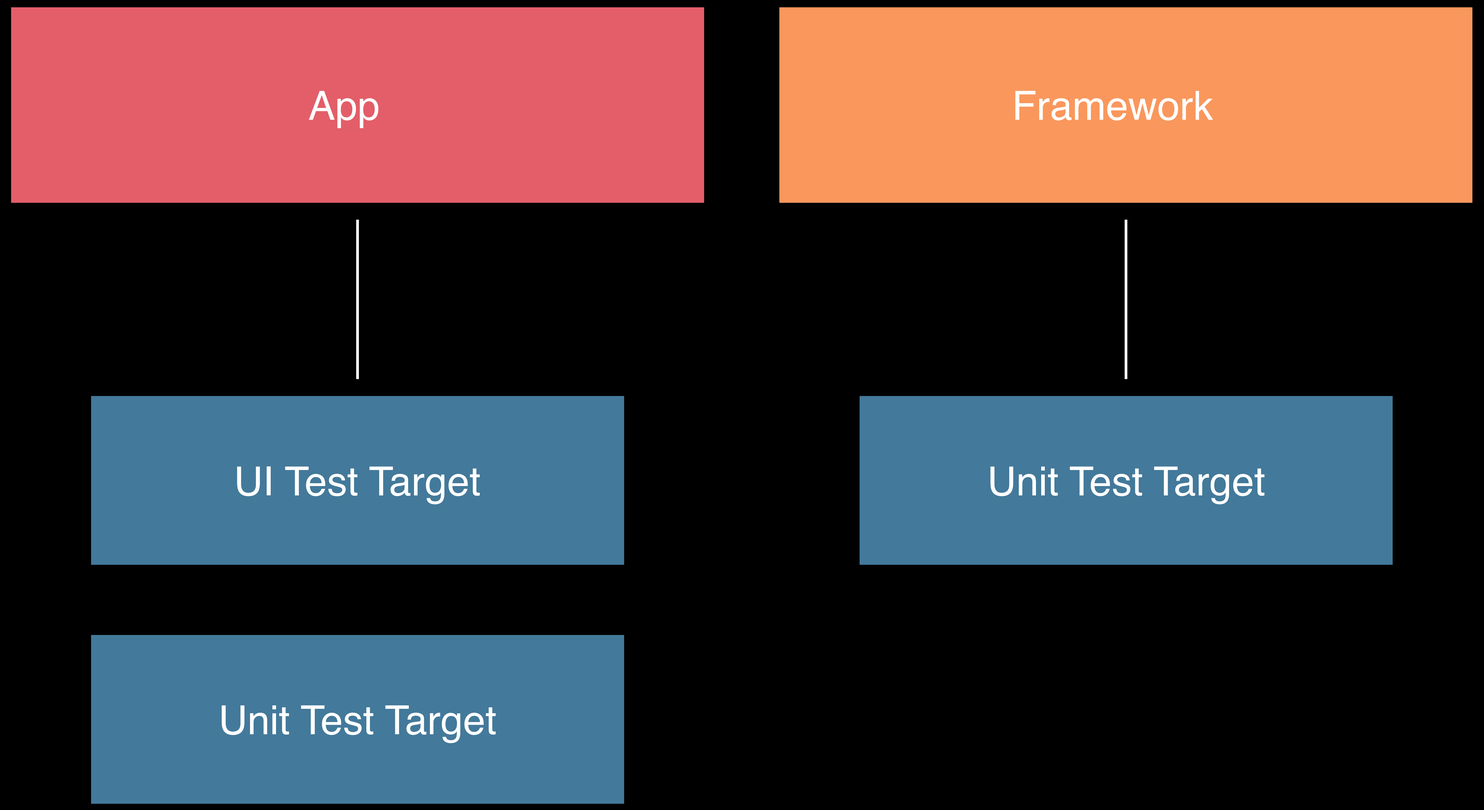
Test Organization



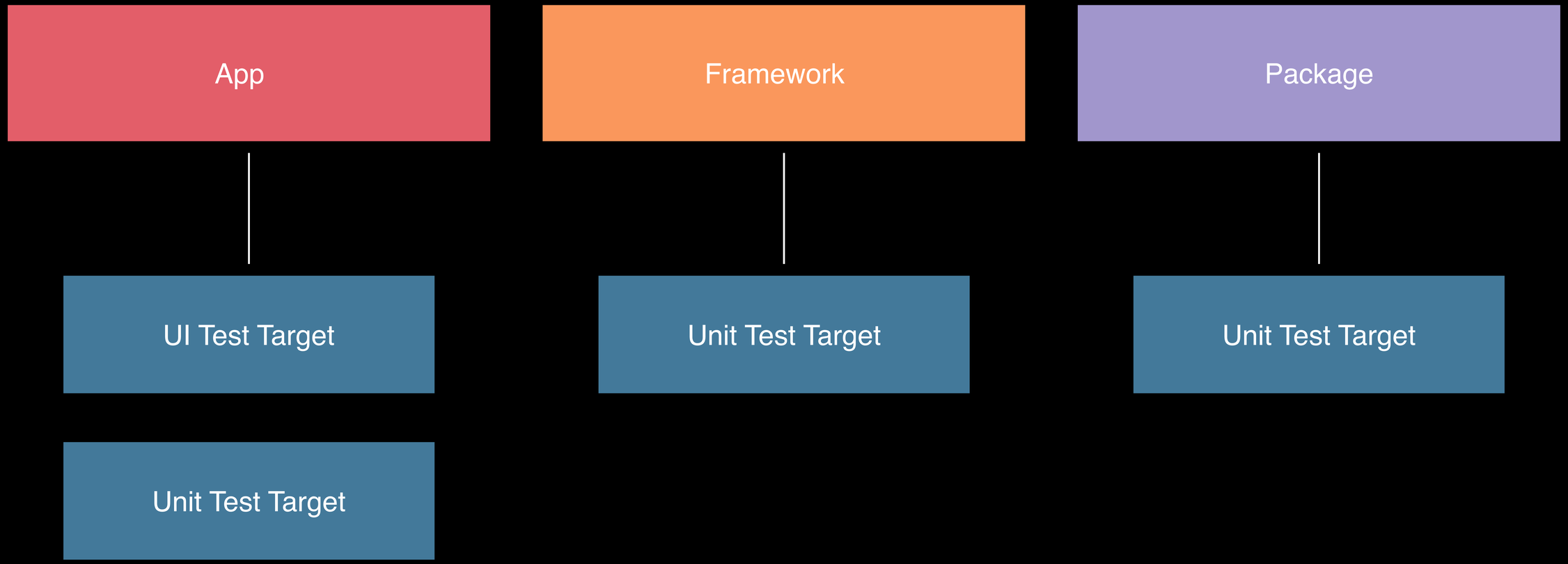
Test Organization



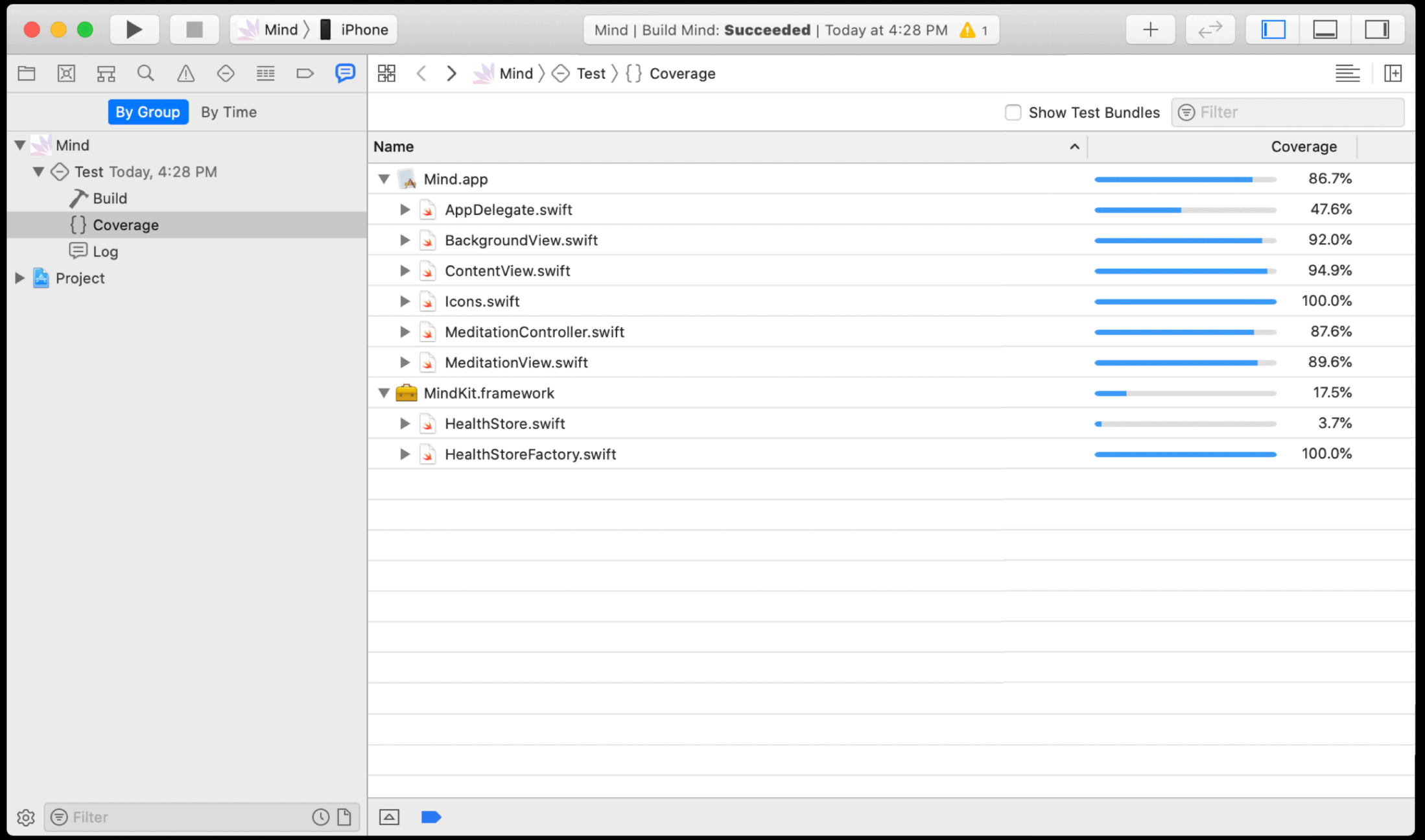
Test Organization

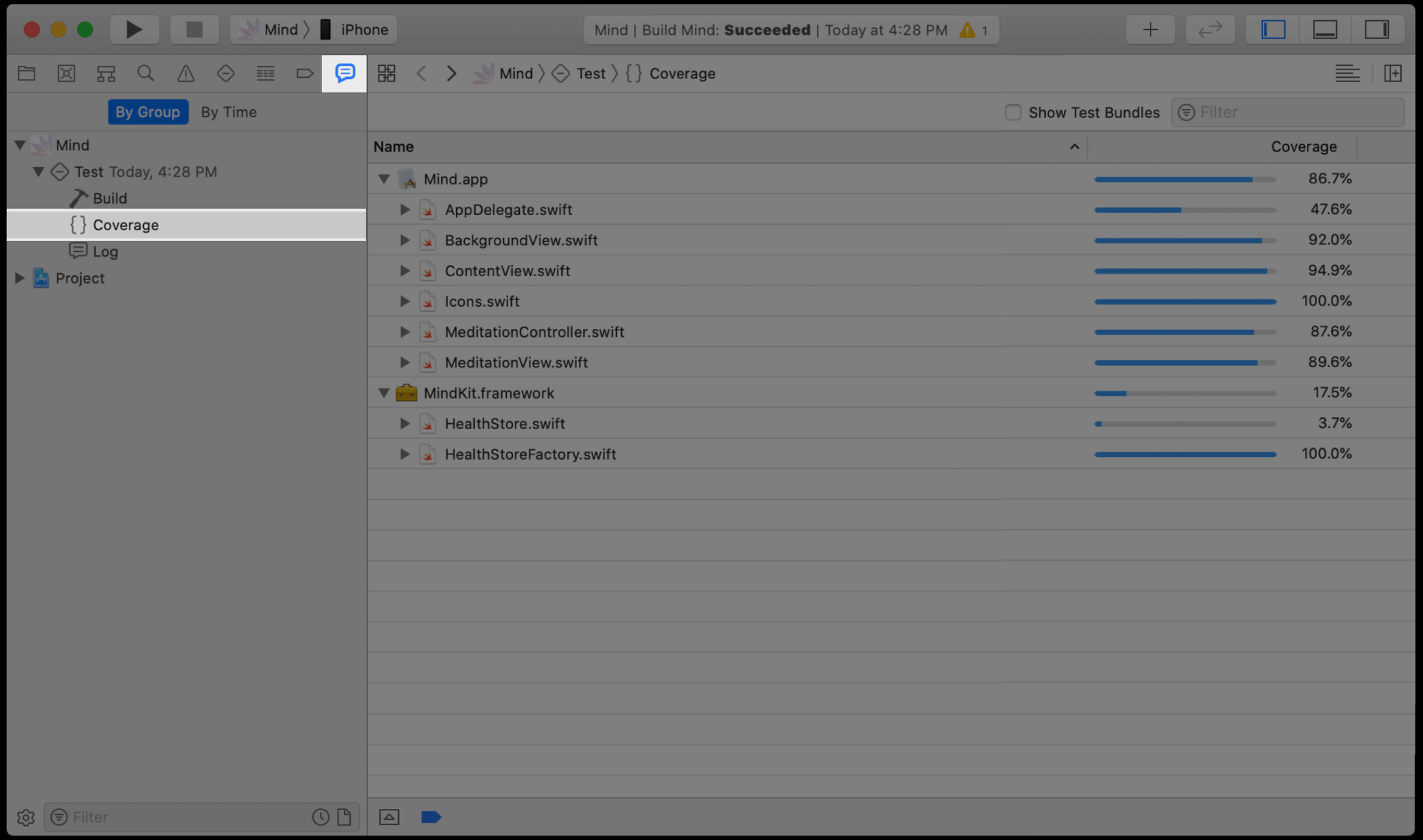


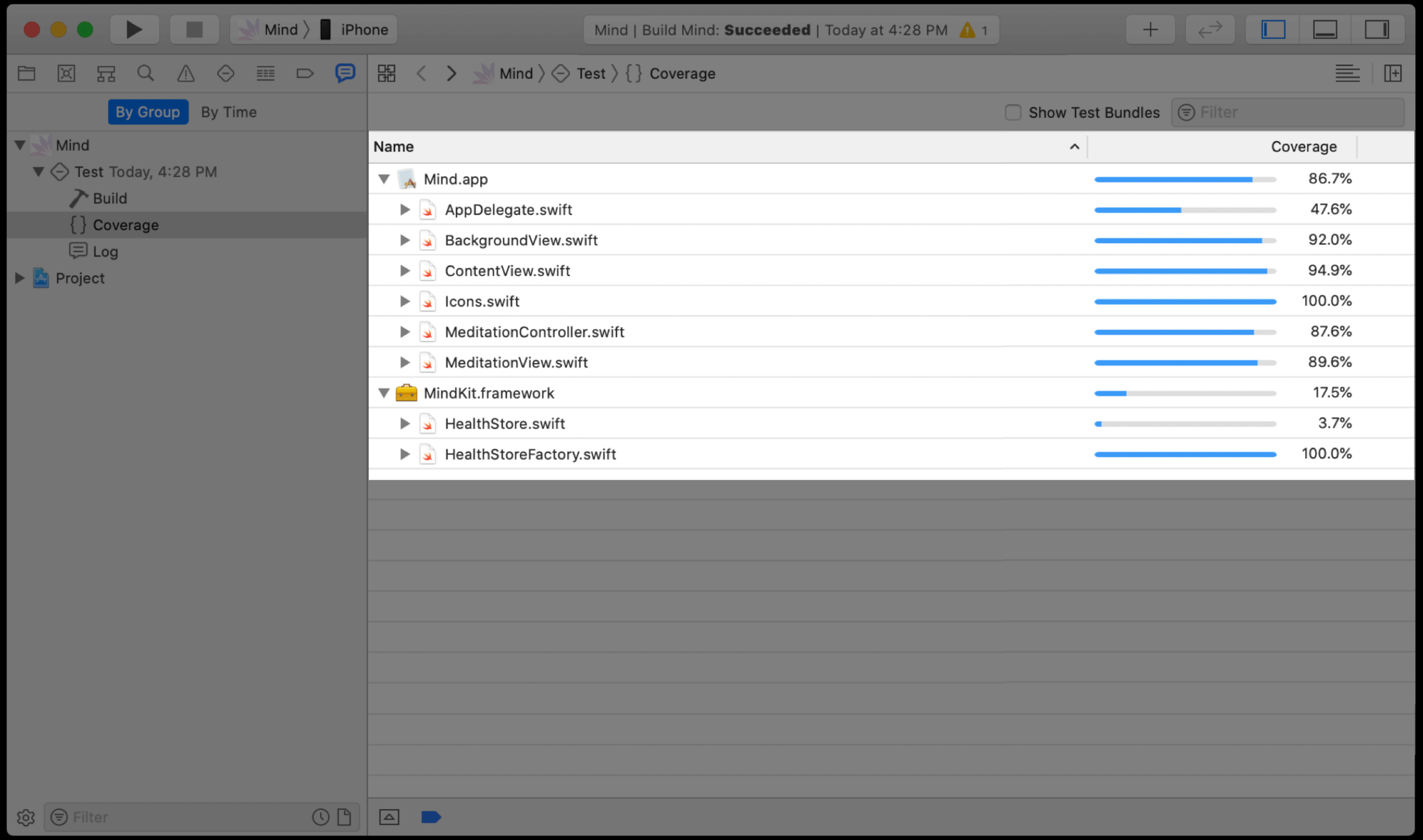
Test Organization

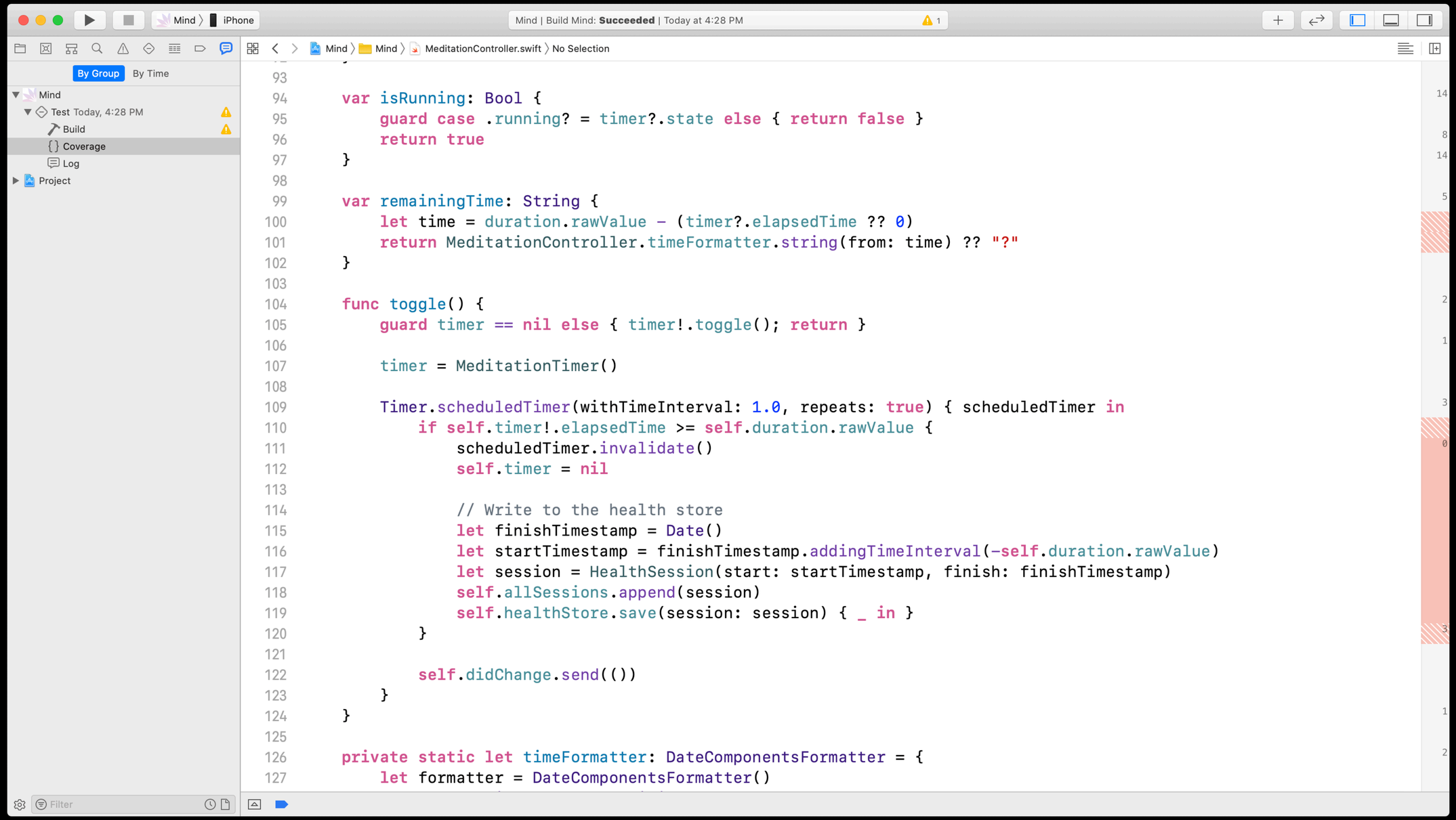


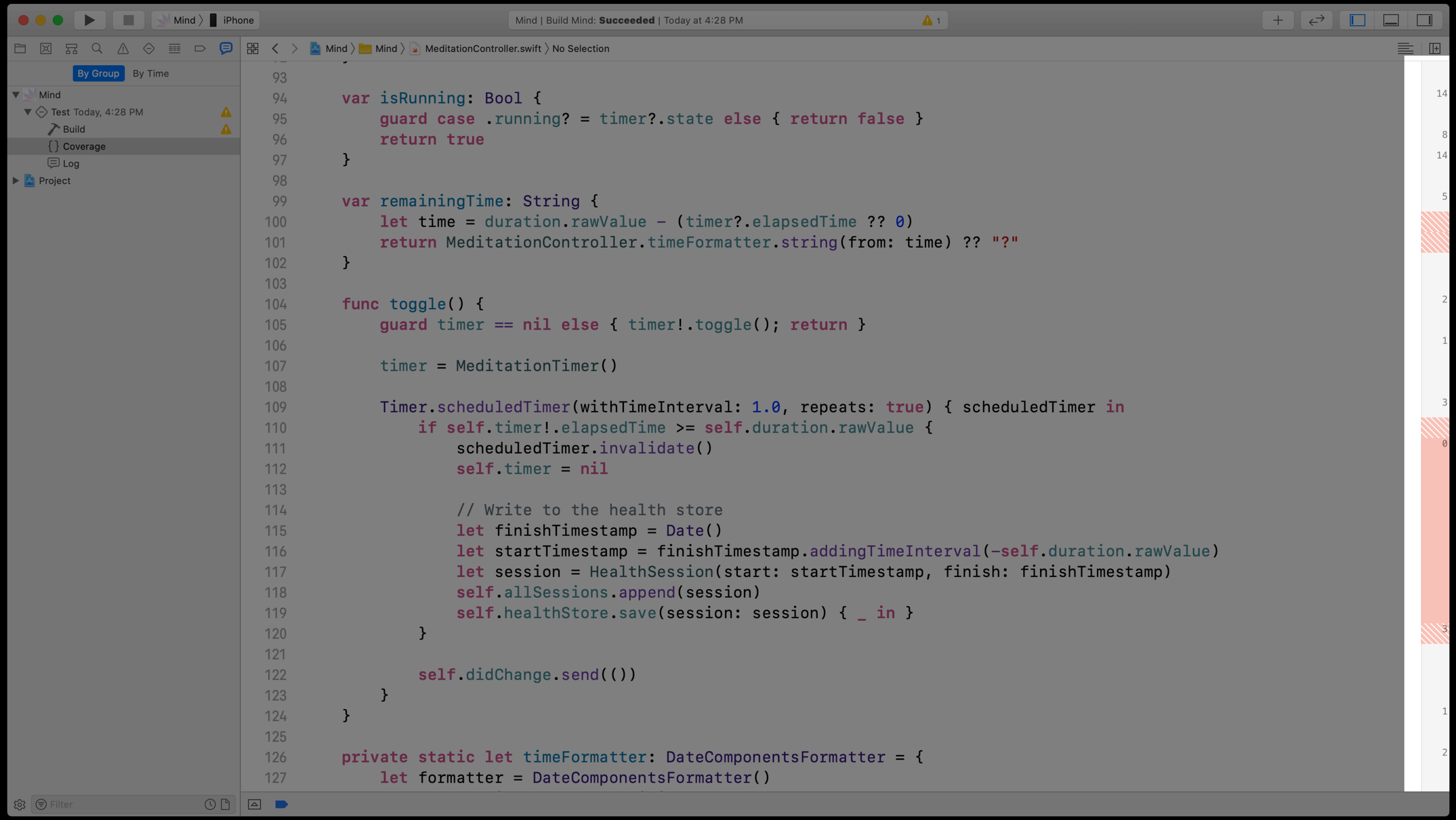
Code Coverage

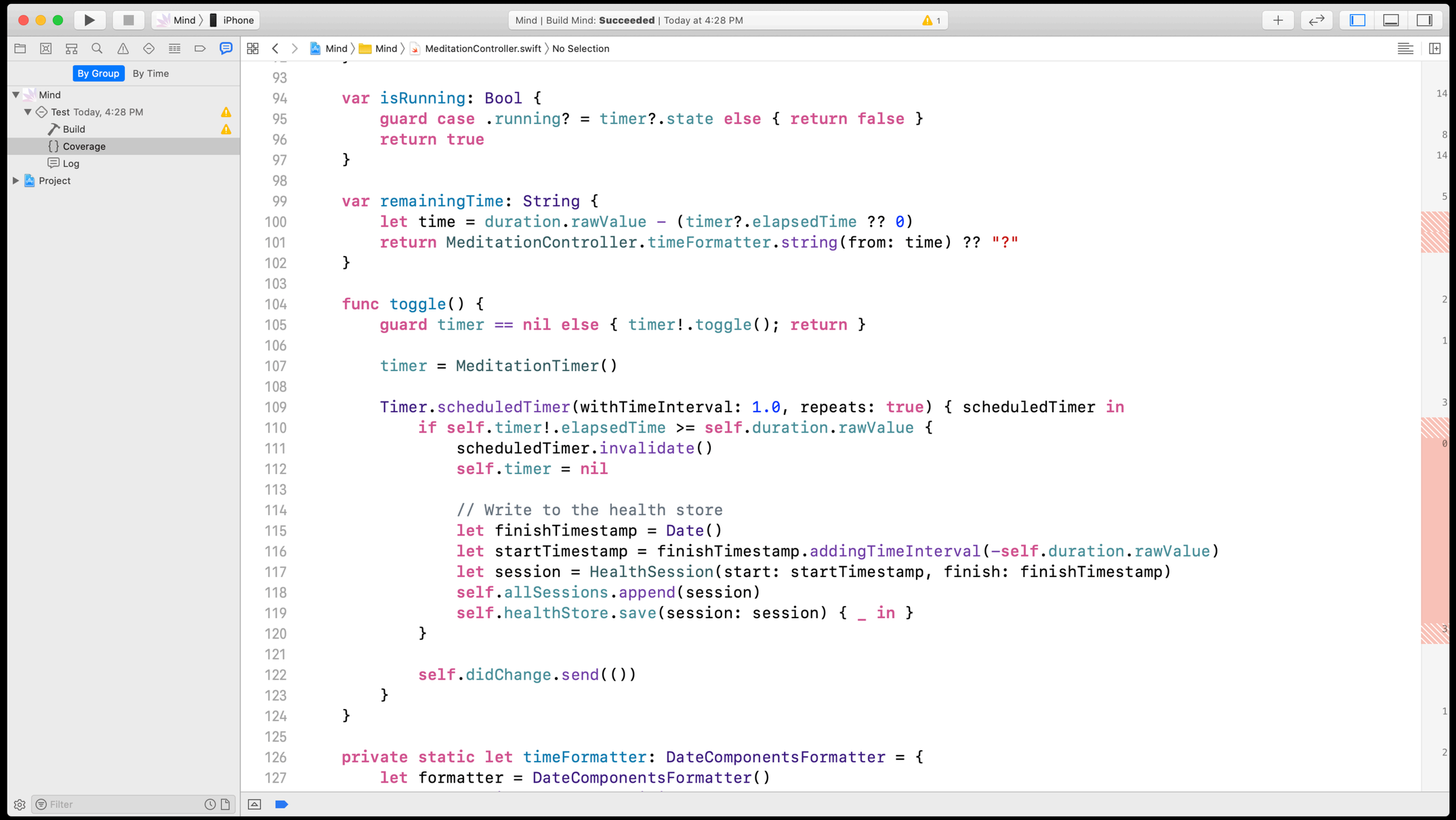

Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM 1
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Filter
Mind 〉 Test 〉 Coverage
Show Test Bundles
Filter
Name
Coverage
Mind.app 86.7%
AppDelegate.swift 47.6%
BackgroundView.swift 92.0%
ContentView.swift 94.9%
Icons.swift 100.0%
MeditationController.swift 87.6%
MeditationView.swift 89.6%
MindKit.framework 17.5%
HealthStore.swift 3.7%
HealthStoreFactory.swift 100.0%

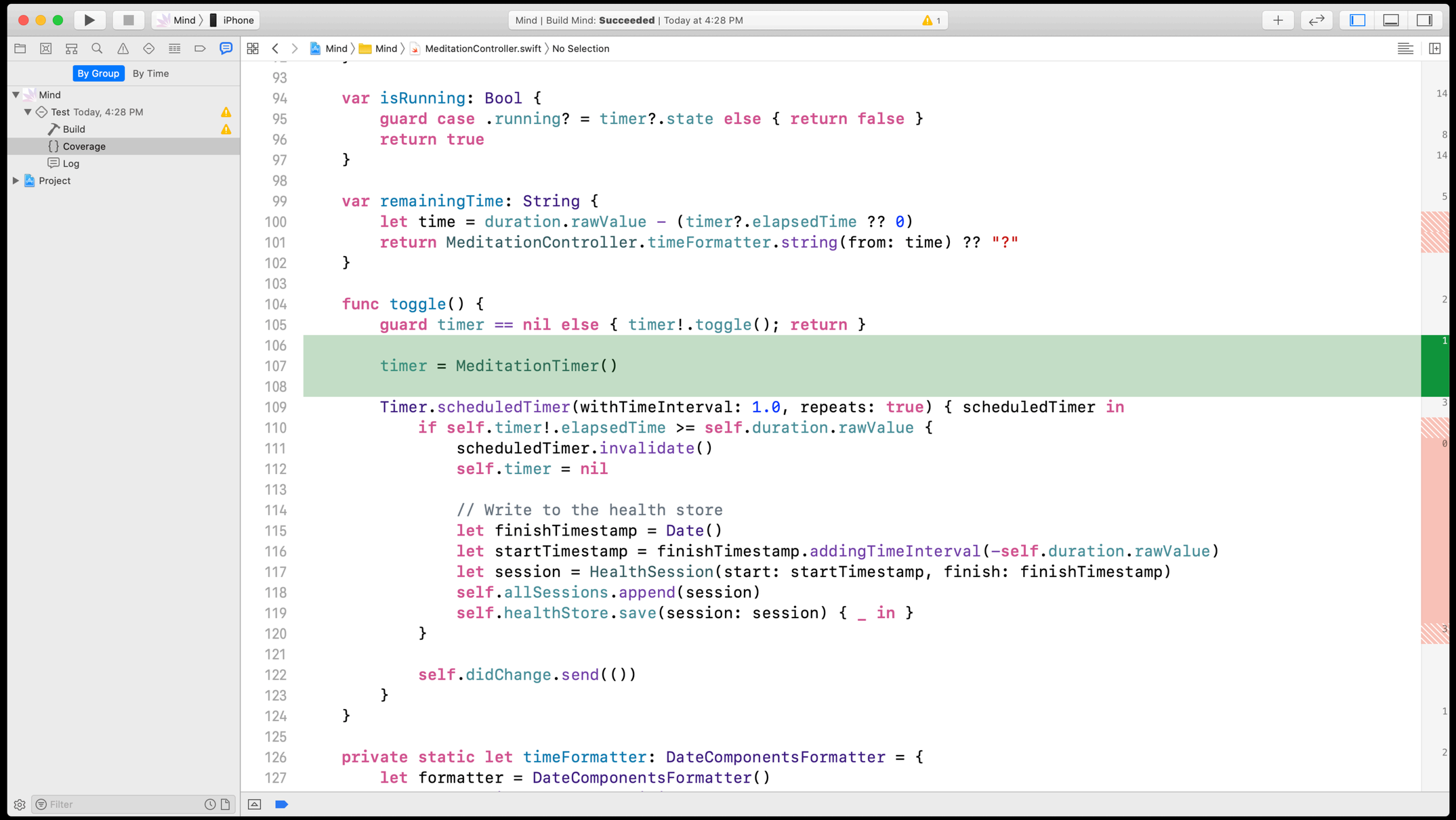

Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM 1
Mind 〉 Test 〉 Coverage
By Group
By Time
Show Test Bundles
Filter
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Name
Coverage
Mind.app 86.7%
AppDelegate.swift 47.6%
BackgroundView.swift 92.0%
ContentView.swift 94.9%
Icons.swift 100.0%
MeditationController.swift 87.6%
MeditationView.swift 89.6%
MindKit.framework 17.5%
HealthStore.swift 3.7%
HealthStoreFactory.swift 100.0%
Filter

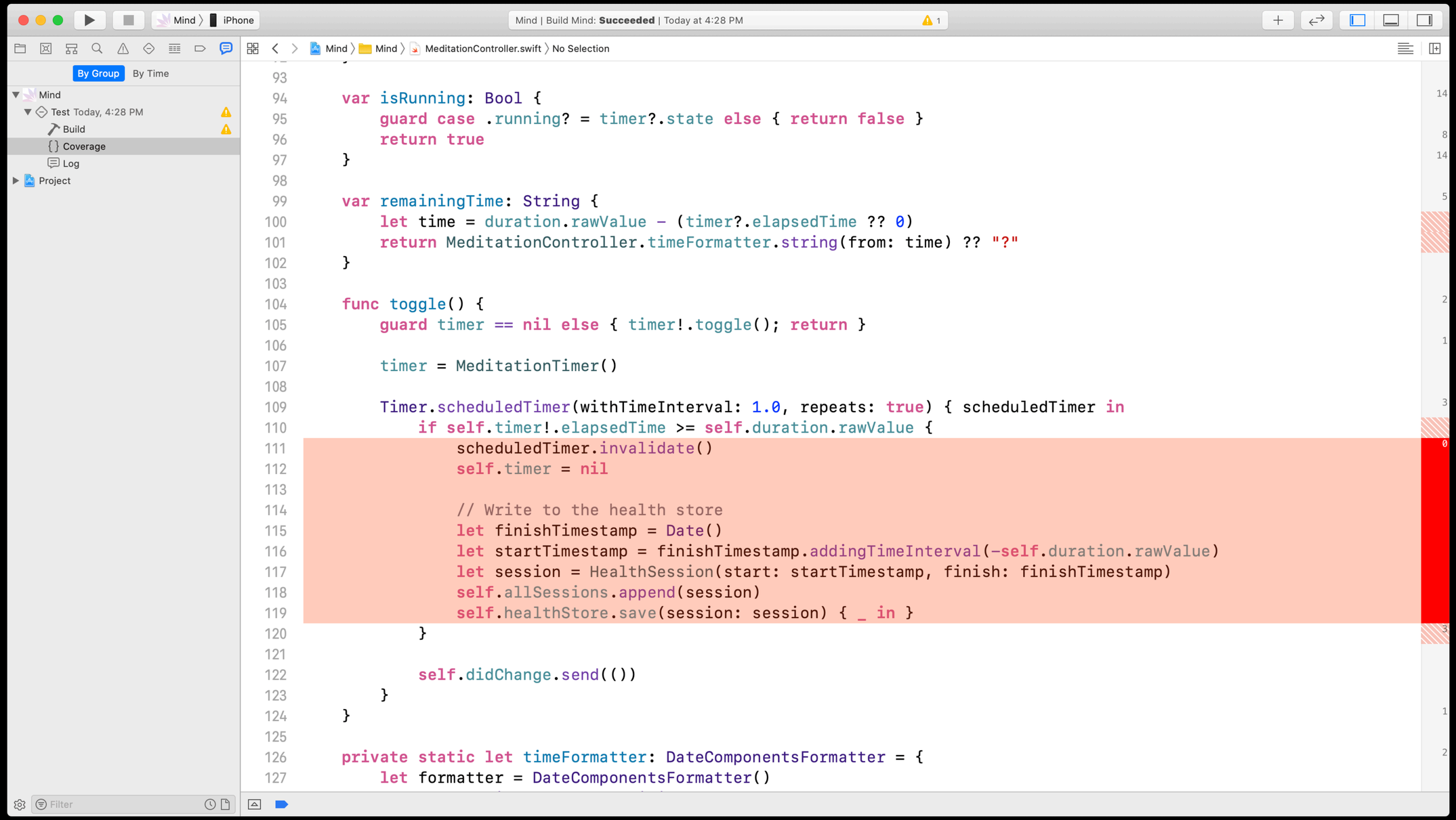

Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM 1
Mind 〉 Test 〉 { } Coverage
By Group
By Time
Show Test Bundles
Filter
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Name
Coverage
Mind.app 86.7%
AppDelegate.swift 47.6%
BackgroundView.swift 92.0%
ContentView.swift 94.9%
Icons.swift 100.0%
MeditationController.swift 87.6%
MeditationView.swift 89.6%
MindKit.framework 17.5%
HealthStore.swift 3.7%
HealthStoreFactory.swift 100.0%
Filter

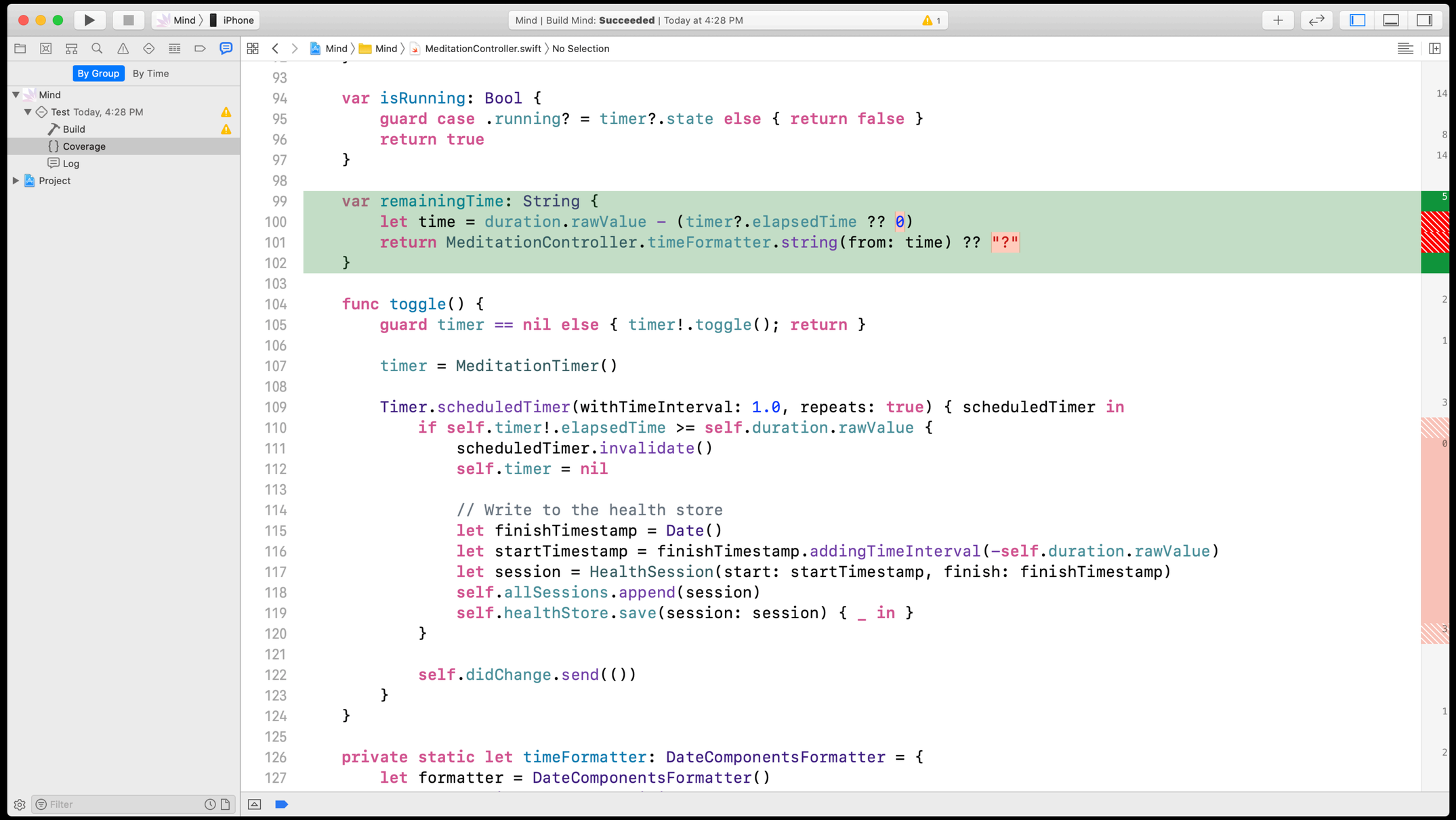

Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM
Mind 〉 Mind 〉 MeditationController.swift 〉 No Selection
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Filter
var isRunning: Bool {
guard case .running? = timer?.state else { return false }
return true
}
var remainingTime: String {
let time = duration.rawValue - (timer?.elapsedTime ?? 0)
return MeditationController.timeFormatter.string(from: time) ?? "?"
}
func toggle() {
guard timer == nil else { timer!.toggle(); return }
timer = MeditationTimer()
Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { scheduledTimer in
if self.timer!.elapsedTime >= self.duration.rawValue {
scheduledTimer.invalidate()
self.timer = nil
// Write to the health store
let finishTimestamp = Date()
let startTimestamp = finishTimestamp.addingTimeInterval(-self.duration.rawValue)
let session = HealthSession(start: startTimestamp, finish: finishTimestamp)
self.allSessions.append(session)
self.healthStore.save(session: session) { _ in }
}
self.didChange.send(())
}
}
private static let timeFormatter: DateComponentsFormatter = {
let formatter = DateComponentsFormatter()


Mind | iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM
Mind | Mind | MeditationController.swift | No Selection
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Filter
var isRunning: Bool {
guard case .running? = timer?.state else { return false }
return true
}
var remainingTime: String {
let time = duration.rawValue - (timer?.elapsedTime ?? 0)
return MeditationController.timeFormatter.string(from: time) ?? "?"
}
func toggle() {
guard timer == nil else { timer!.toggle(); return }
timer = MeditationTimer()
Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { scheduledTimer in
if self.timer!.elapsedTime >= self.duration.rawValue {
scheduledTimer.invalidate()
self.timer = nil
// Write to the health store
let finishTimestamp = Date()
let startTimestamp = finishTimestamp.addingTimeInterval(-self.duration.rawValue)
let session = HealthSession(start: startTimestamp, finish: finishTimestamp)
self.allSessions.append(session)
self.healthStore.save(session: session) { _ in }
}
self.didChange.send(())
}
}
private static let timeFormatter: DateComponentsFormatter = {
let formatter = DateComponentsFormatter()


Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM
Mind 〉 Mind 〉 MeditationController.swift 〉 No Selection
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Filter
93
94 var isRunning: Bool {
95 guard case .running? = timer?.state else { return false }
96 return true
97 }
98
99 var remainingTime: String {
100 let time = duration.rawValue - (timer?.elapsedTime ?? 0)
101 return MeditationController.timeFormatter.string(from: time) ?? "?"
102 }
103
104 func toggle() {
105 guard timer == nil else { timer!.toggle(); return }
106
107 timer = MeditationTimer()
108
109 Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { scheduledTimer in
110 if self.timer!.elapsedTime >= self.duration.rawValue {
111 scheduledTimer.invalidate()
112 self.timer = nil
113
114 // Write to the health store
115 let finishTimestamp = Date()
116 let startTimestamp = finishTimestamp.addingTimeInterval(-self.duration.rawValue)
117 let session = HealthSession(start: startTimestamp, finish: finishTimestamp)
118 self.allSessions.append(session)
119 self.healthStore.save(session: session) { _ in }
120 }
121
122 self.didChange.send(())
123 }
124 }
125
126 private static let timeFormatter: DateComponentsFormatter = {
127 let formatter = DateComponentsFormatter()

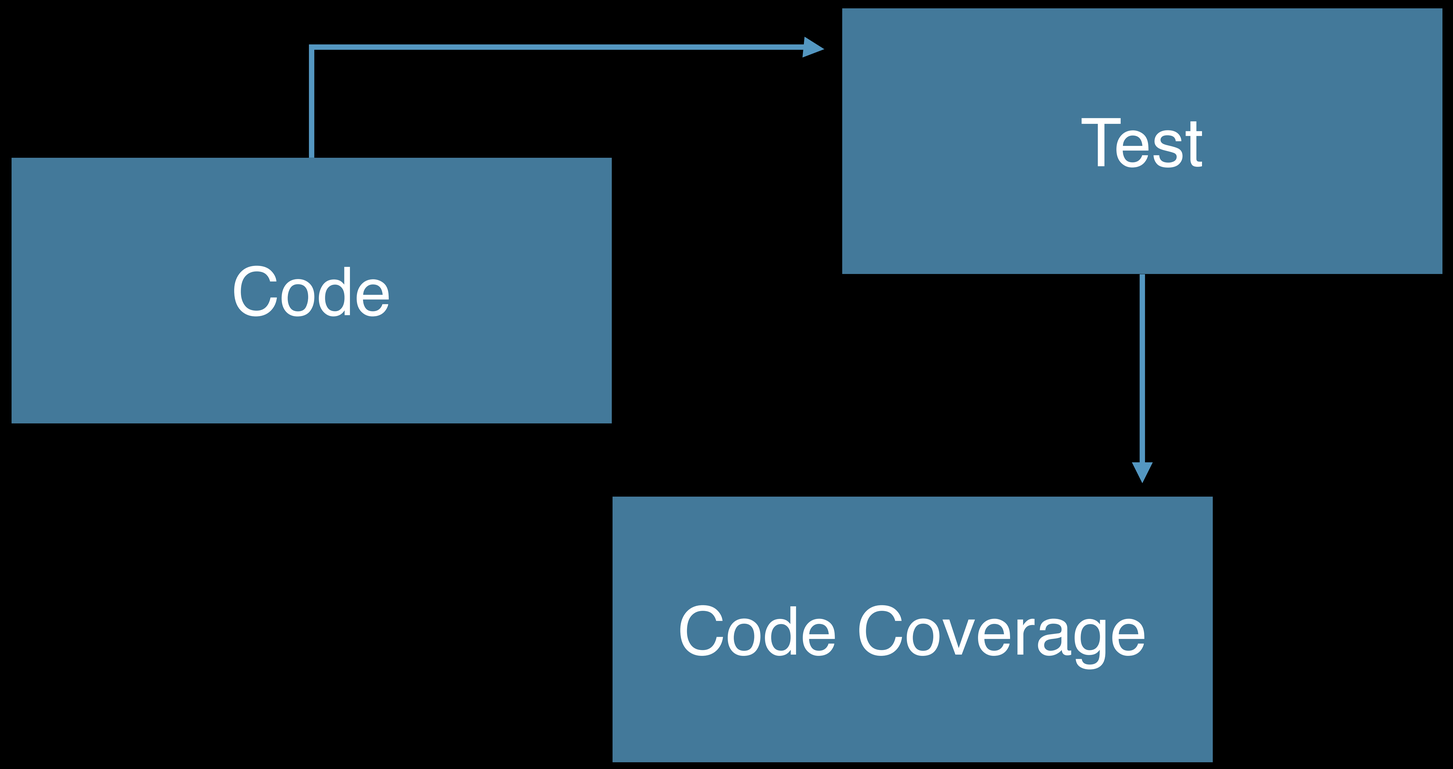

Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM
1
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Mind 〉 Mind 〉 MeditationController.swift 〉 No Selection
93
94 var isRunning: Bool {
95 guard case .running? = timer?.state else { return false }
96 return true
97 }
98
99 var remainingTime: String {
100 let time = duration.rawValue - (timer?.elapsedTime ?? 0)
101 return MeditationController.timeFormatter.string(from: time) ?? "?"
102 }
103
104 func toggle() {
105 guard timer == nil else { timer!.toggle(); return }
106
107 timer = MeditationTimer()
108
109 Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { scheduledTimer in
110 if self.timer!.elapsedTime >= self.duration.rawValue {
111 scheduledTimer.invalidate()
112 self.timer = nil
113
114 // Write to the health store
115 let finishTimestamp = Date()
116 let startTimestamp = finishTimestamp.addingTimeInterval(-self.duration.rawValue)
117 let session = HealthSession(start: startTimestamp, finish: finishTimestamp)
118 self.allSessions.append(session)
119 self.healthStore.save(session: session) { _ in }
120 }
121
122 self.didChange.send(())
123 }
124 }
125
126 private static let timeFormatter: DateComponentsFormatter = {
127 let formatter = DateComponentsFormatter()
Filter


Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Mind 〉 Mind 〉 MeditationController.swift 〉 No Selection
var isRunning: Bool {
guard case .running? = timer?.state else { return false }
return true
}
var remainingTime: String {
let time = duration.rawValue - (timer?.elapsedTime ?? 0)
return MeditationController.timeFormatter.string(from: time) ?? "?"
}
func toggle() {
guard timer == nil else { timer!.toggle(); return }
timer = MeditationTimer()
Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { scheduledTimer in
if self.timer!.elapsedTime >= self.duration.rawValue {
scheduledTimer.invalidate()
self.timer = nil
// Write to the health store
let finishTimestamp = Date()
let startTimestamp = finishTimestamp.addingTimeInterval(-self.duration.rawValue)
let session = HealthSession(start: startTimestamp, finish: finishTimestamp)
self.allSessions.append(session)
self.healthStore.save(session: session) { _ in }
}
self.didChange.send(())
}
}
private static let timeFormatter: DateComponentsFormatter = {
let formatter = DateComponentsFormatter()
Filter

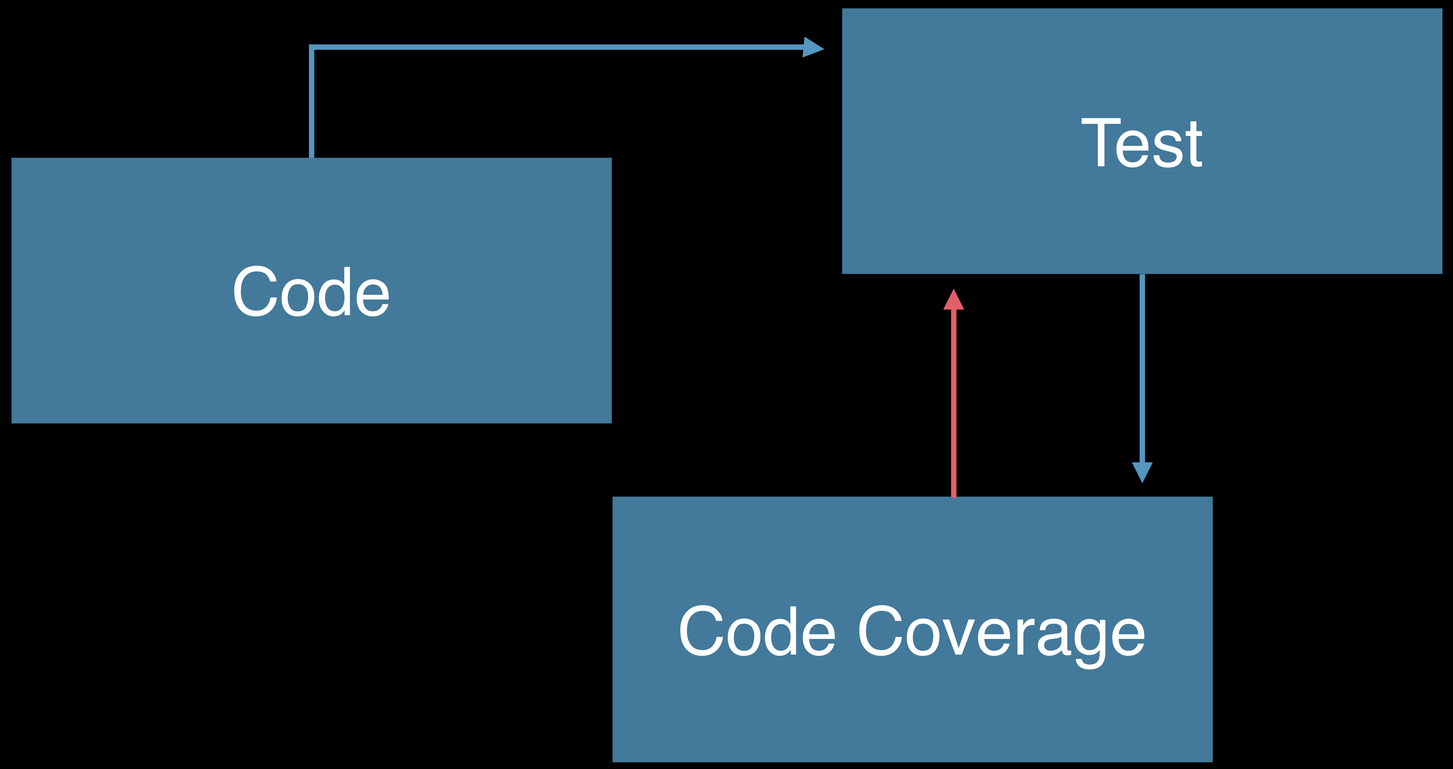

Mind 〉 iPhone
Mind | Build Mind: Succeeded | Today at 4:28 PM
By Group
By Time
Mind
Test Today, 4:28 PM
Build
Coverage
Log
Project
Mind 〉 Mind 〉 MeditationController.swift 〉 No Selection
var isRunning: Bool {
guard case .running? = timer?.state else { return false }
return true
}
var remainingTime: String {
let time = duration.rawValue - (timer?.elapsedTime ?? 0)
return MeditationController.timeFormatter.string(from: time) ?? "?"
}
func toggle() {
guard timer == nil else { timer!.toggle(); return }
timer = MeditationTimer()
Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) { scheduledTimer in
if self.timer!.elapsedTime >= self.duration.rawValue {
scheduledTimer.invalidate()
self.timer = nil
// Write to the health store
let finishTimestamp = Date()
let startTimestamp = finishTimestamp.addingTimeInterval(-self.duration.rawValue)
let session = HealthSession(start: startTimestamp, finish: finishTimestamp)
self.allSessions.append(session)
self.healthStore.save(session: session) { _ in }
}
self.didChange.send(())
}
}
private static let timeFormatter: DateComponentsFormatter = {
let formatter = DateComponentsFormatter()
Filter

Test Early, Test Often

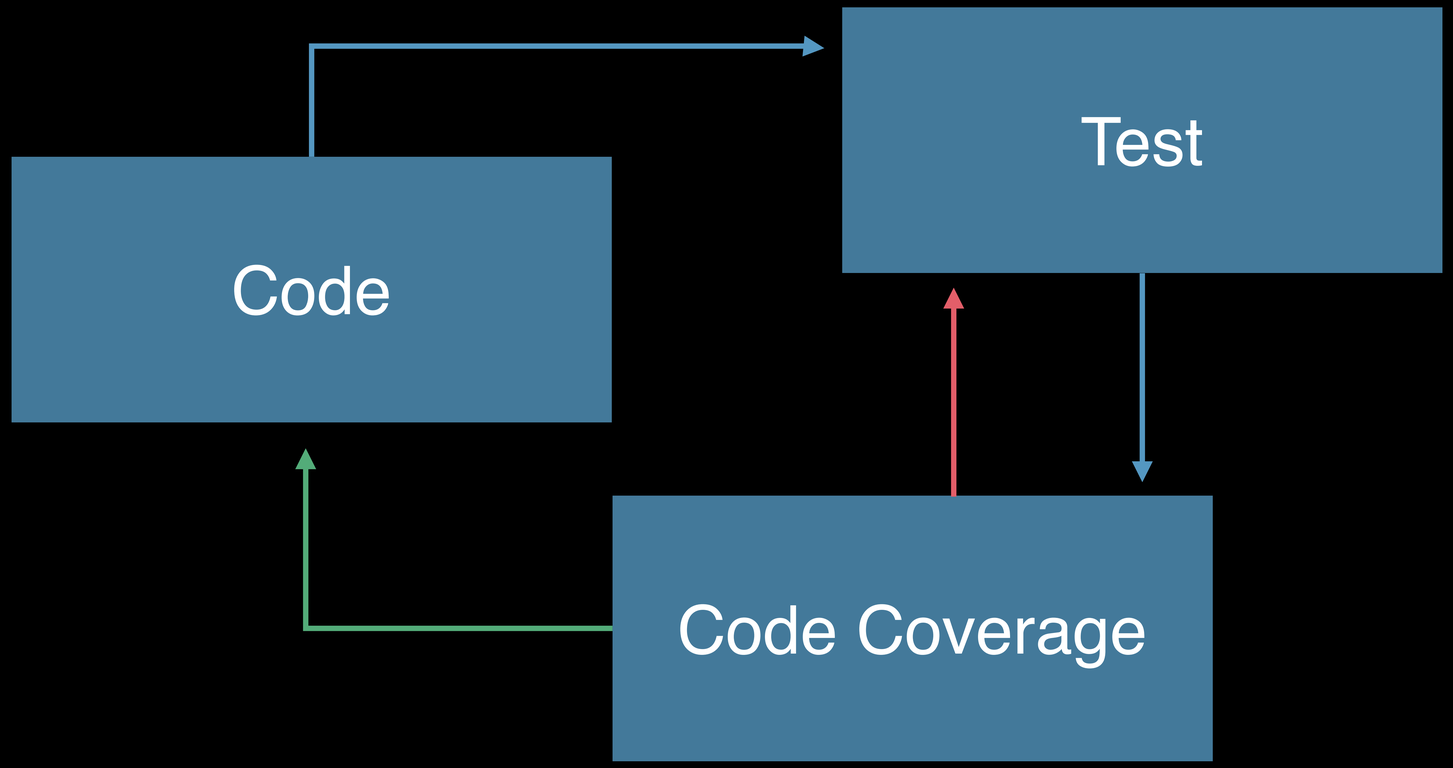
Test Early, Test Often



Test Early, Test Often



Test Early, Test Often



Test Plans

Stuart Montgomery, XCTest Engineer

Getting the Most Out of Your Tests

Run test suite multiple ways to catch more bugs

Leverage more of Xcode’s testing options


9:41
Journal
May, 2019
San Francisco
May, 2019
New York City
Discover
Plan
Journal


9:41
Journal
May, 2019
San Francisco
May, 2019
New York City
Discover
Plan
Journal


9:41
Journal
May, 2019
San Francisco
May, 2019
New York City
Discover
Plan
Journal


9:41
Tagebuch
Mai, 2019
VACATION_TITLE
Mai, 2019
VACATION_TITLE
Entdecken
Plan
Tagebuch


9:41
Tagebuch
Mai, 2019
VACATION_TITLE
Mai, 2019
VACATION_TITLE
Entdecken
Plan
Tagebuch


9:41
Tagebuch
Mai, 2019
Sommer in SF
Mai, 2019
Sommer in NY
Entdecken
Plan
Tagebuch

Getting the Most Out of Your Tests

Running multiple ways

Different localizations

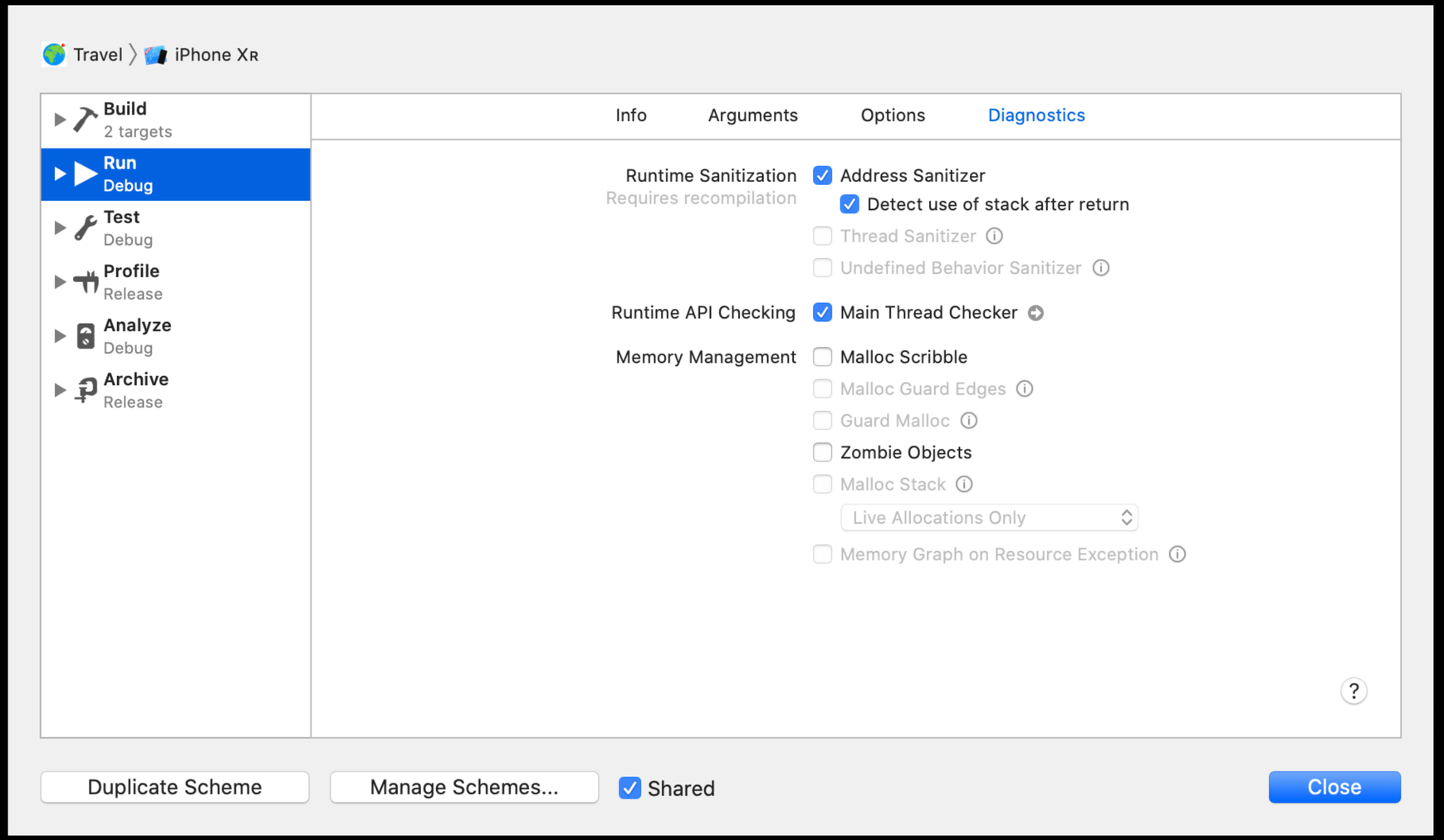
Random order

Using sanitizers

Arguments or environment variables

Editing App Launch Settings

Scheme editor, run action



Test Plans

Test Plans
Overview

NEW

Allows running tests more than once with different settings

Defines all testing variants in one place

Can share between multiple schemes

Supported in Xcode and xcodebuild for CI / Xcode Server

Easy to adopt in existing projects

Demo

Using test plans

Test Plan File

JSON file with .xctestplan extension

Contains

- Tests to run
- Test configurations

Included in project structure

Referenced by schemes

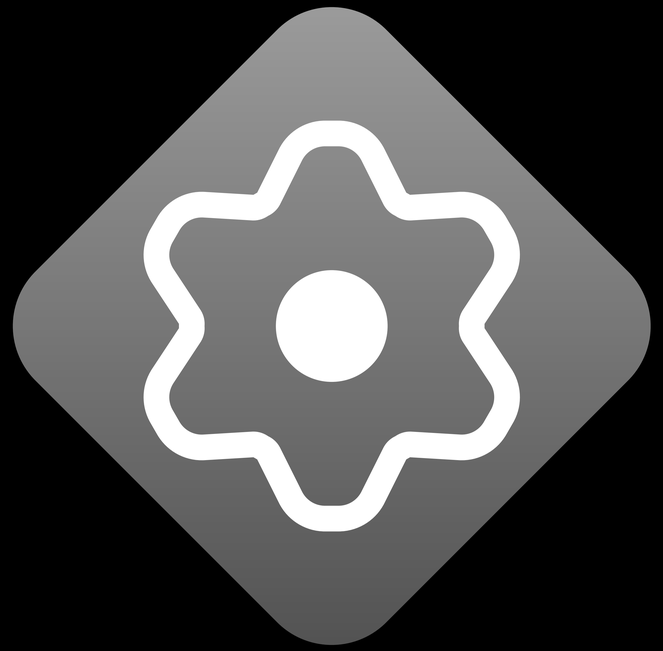
Test Configuration

Describes a single run of the plan’s tests

Has a customizable, unique name

Includes options for how to build and run tests

Inherits shared settings



Test Configuration

Available options

Command-line arguments and environment variables

Language and region

Sanitizers (ASan, TSan, UBSan) and Main Thread Checker

Simulated location

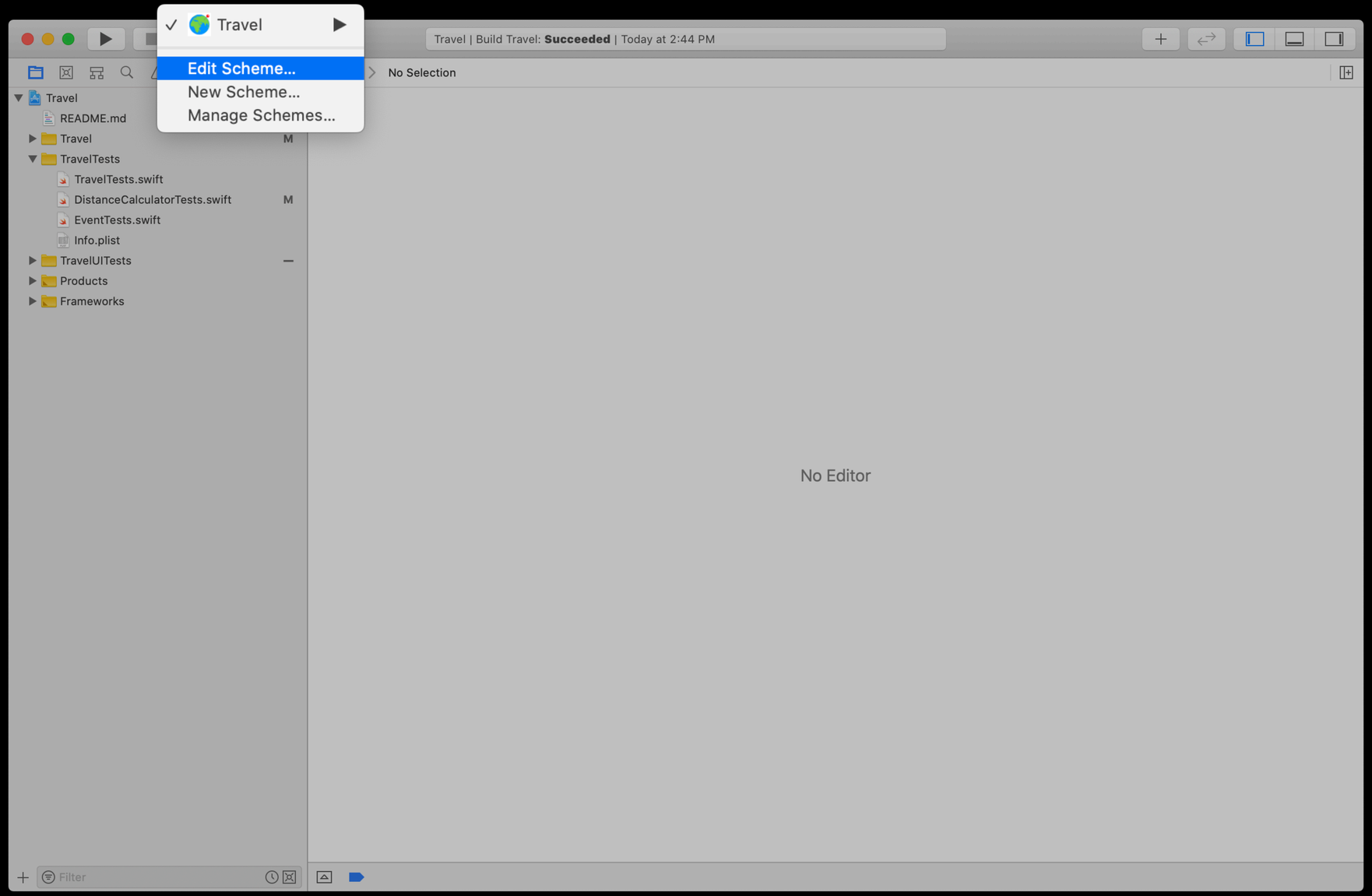
UI testing screenshots and test attachments lifetime

Randomized test execution order

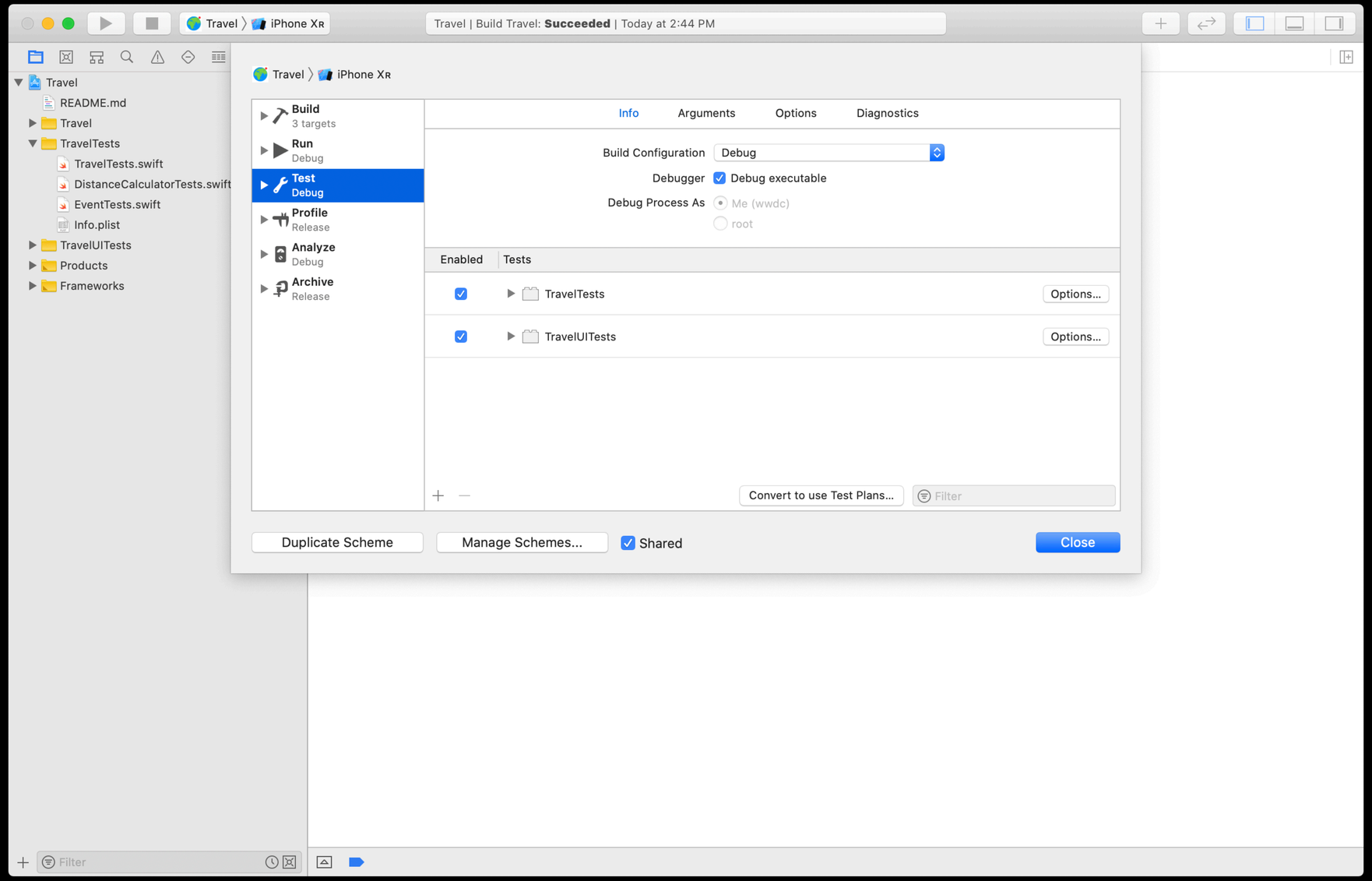
Memory diagnostics (e.g. MallocStackLogging, NSZombies)

Converting a Scheme to Use Test Plans

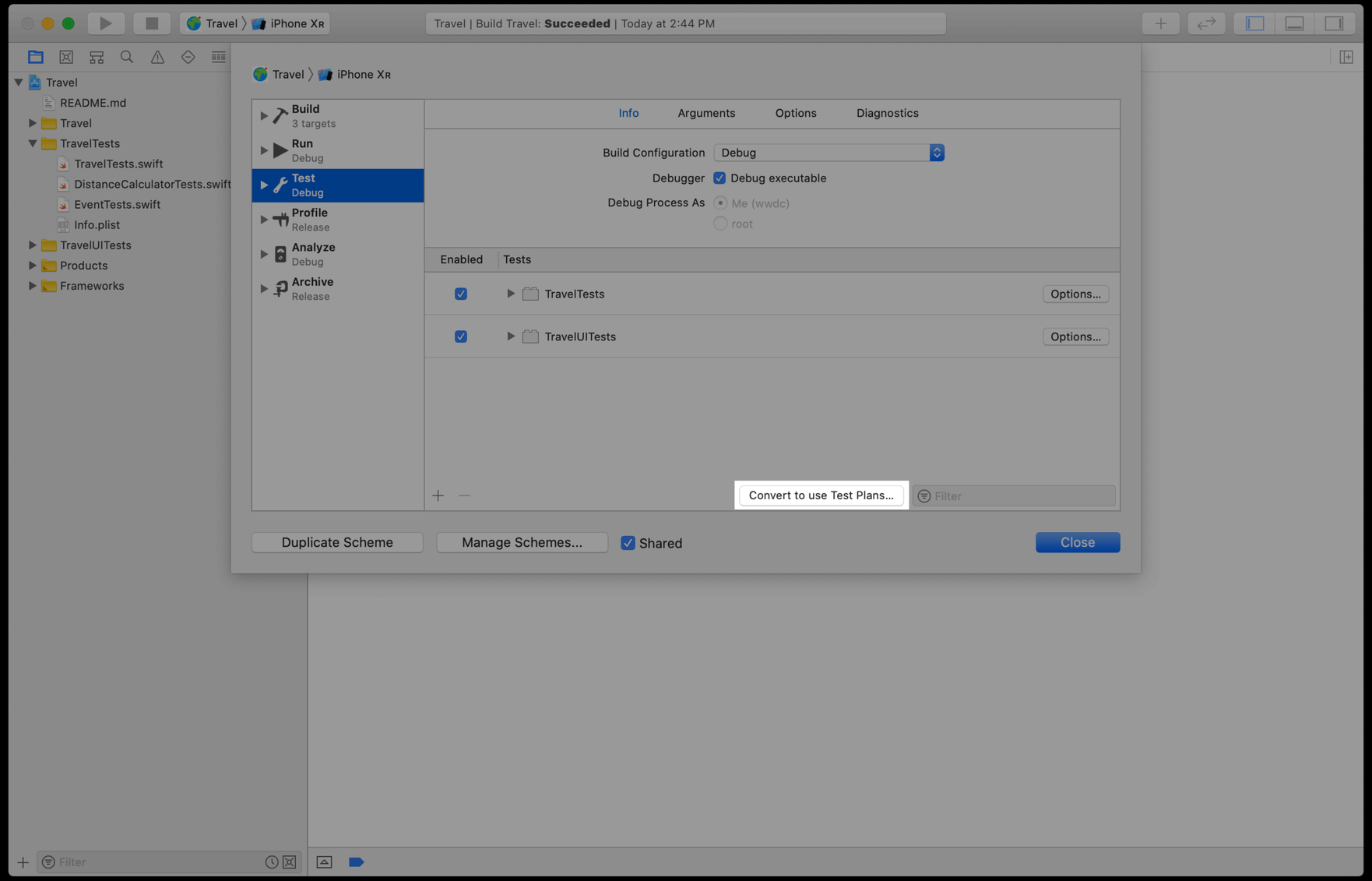
Converting a Scheme to Use Test Plans



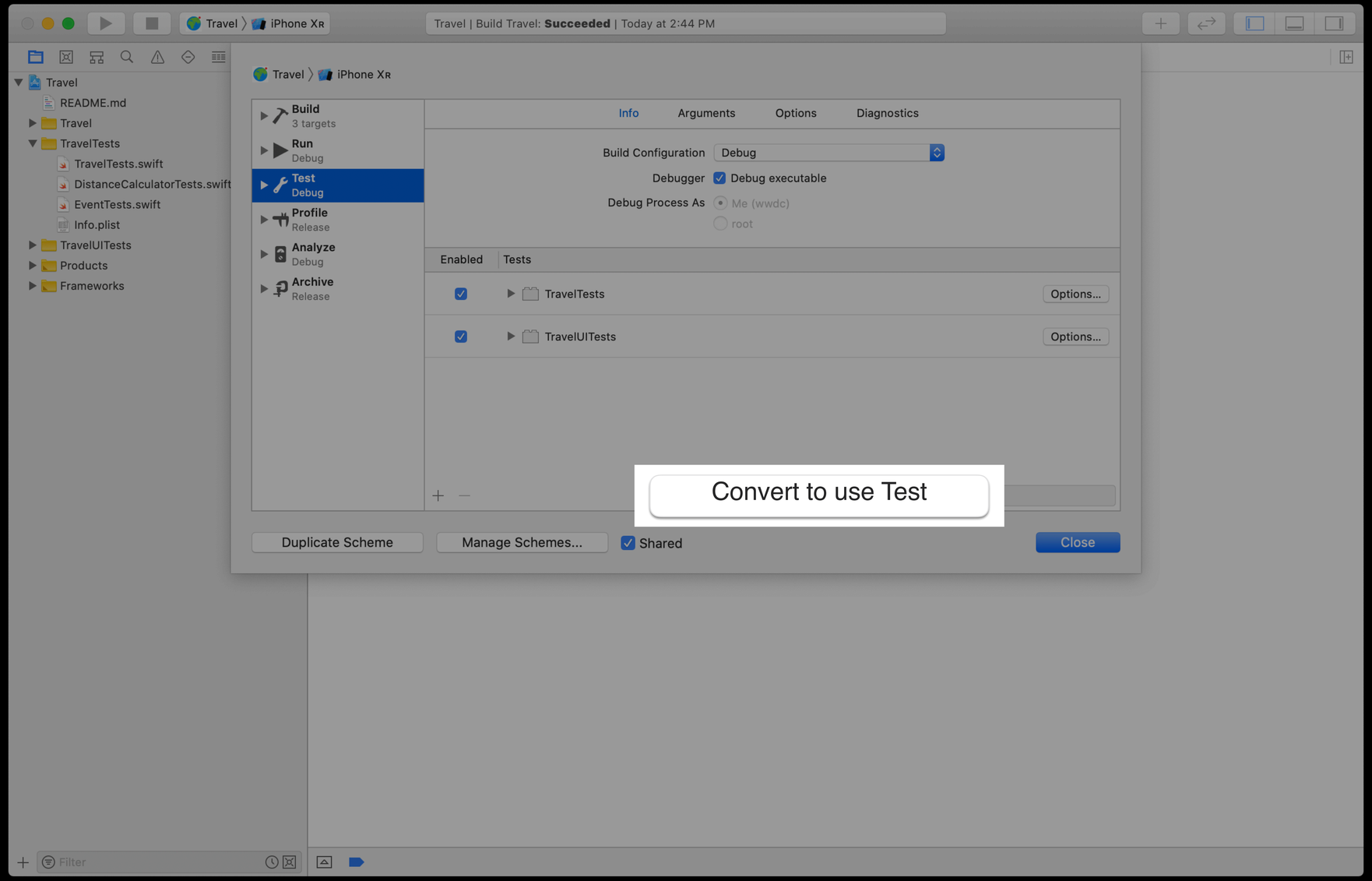
Converting a Scheme to Use Test Plans



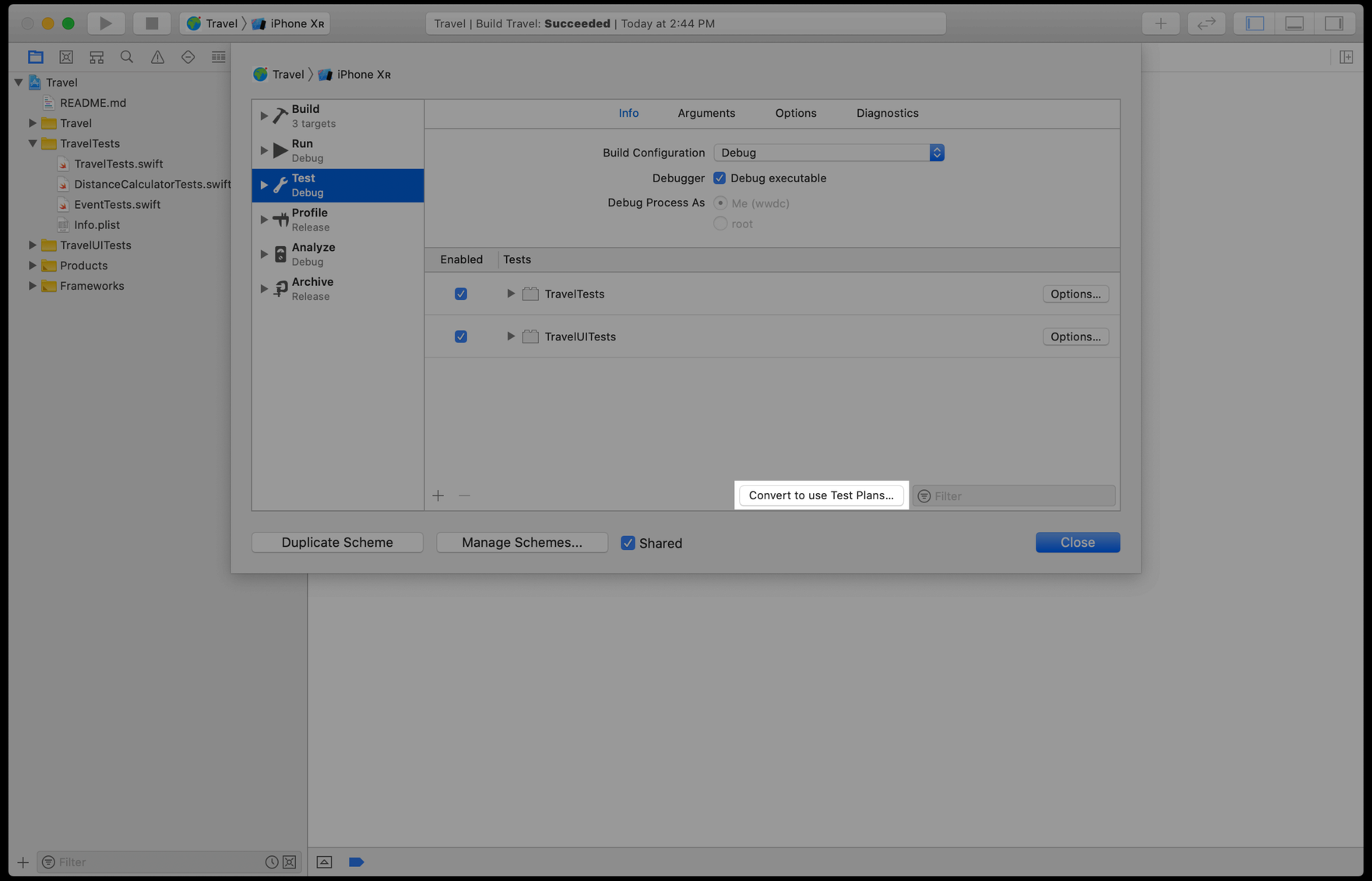
Converting a Scheme to Use Test Plans



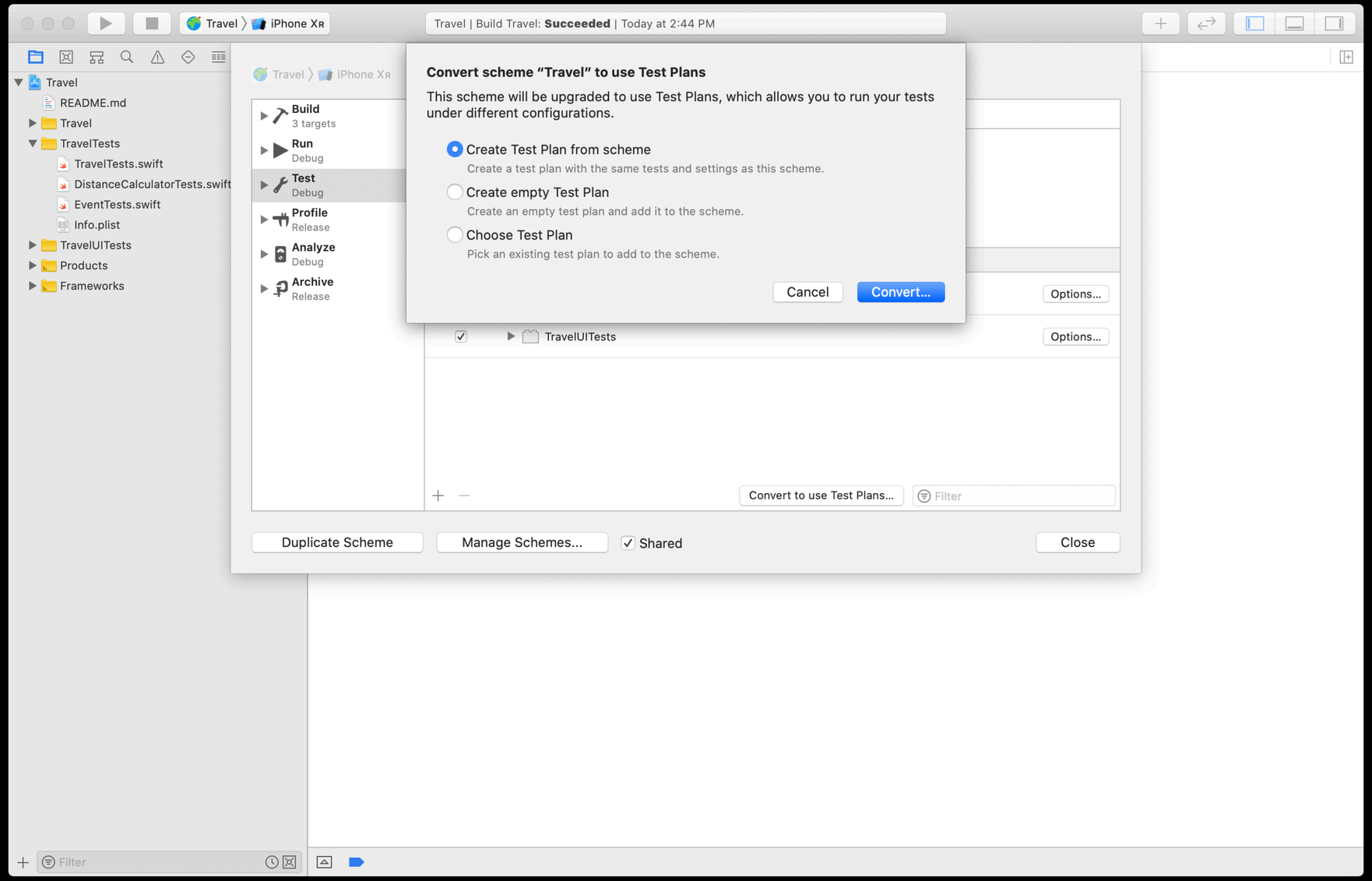
Converting a Scheme to Use Test Plans



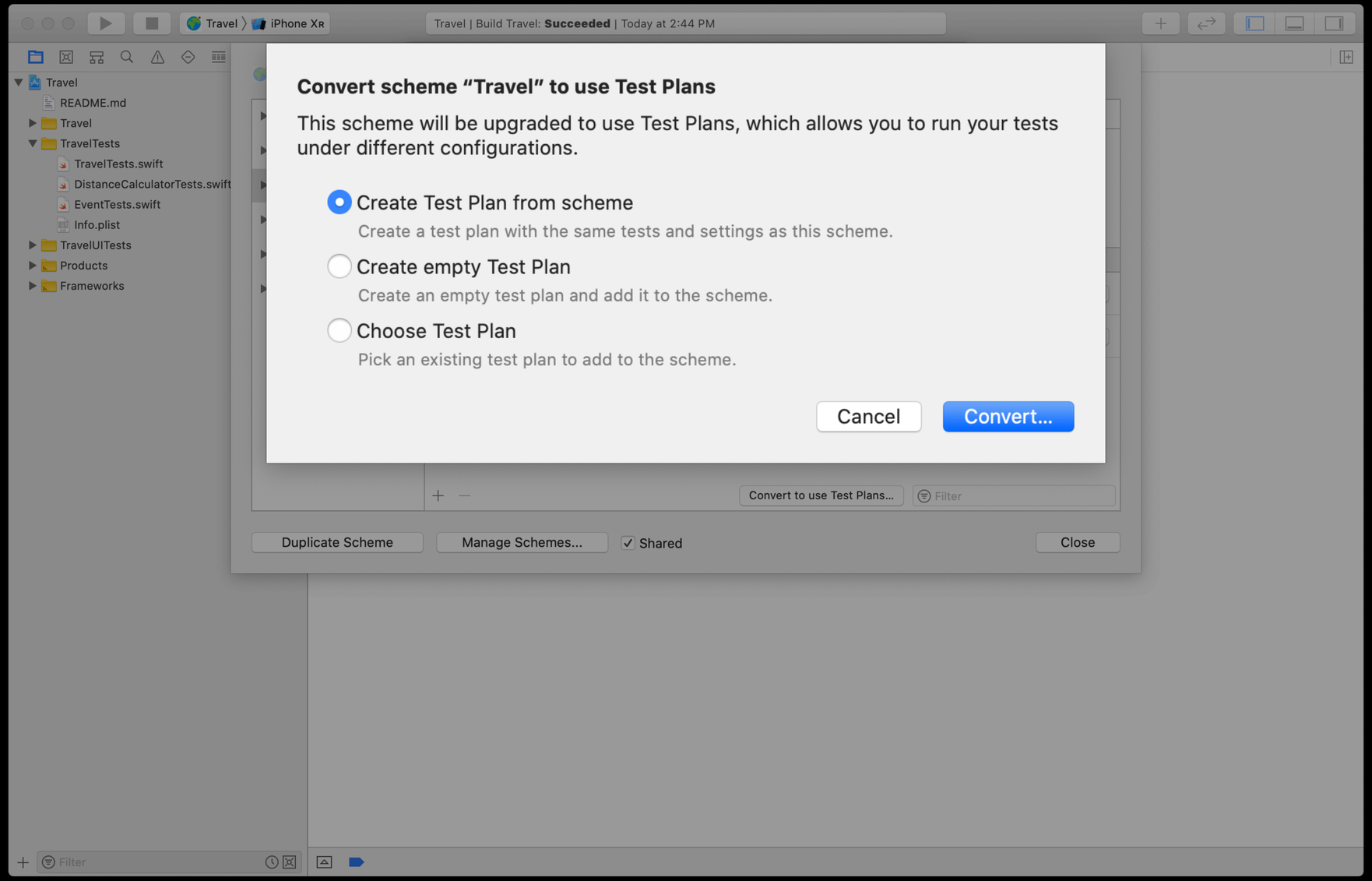
Converting a Scheme to Use Test Plans



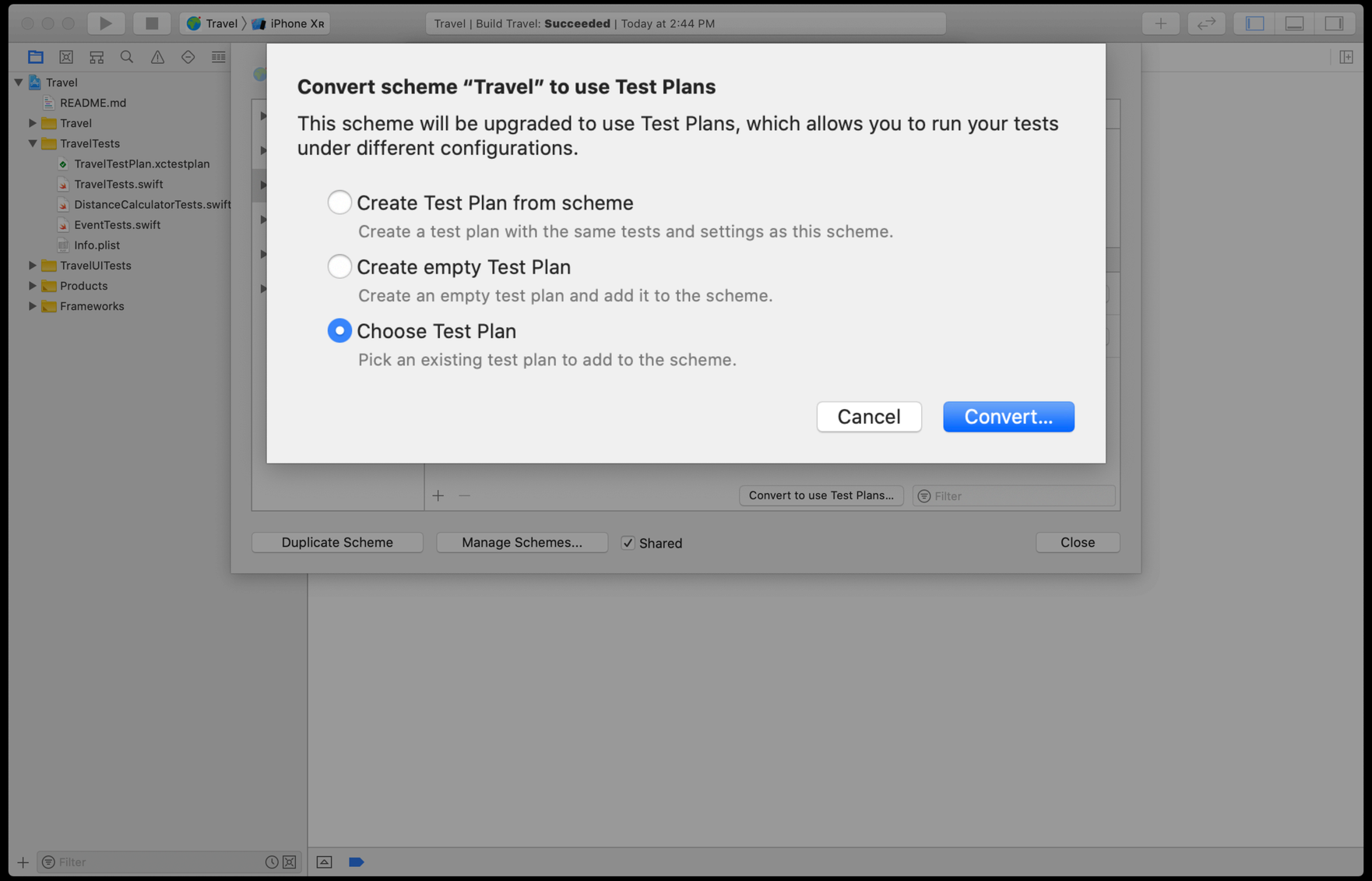
Converting a Scheme to Use Test Plans



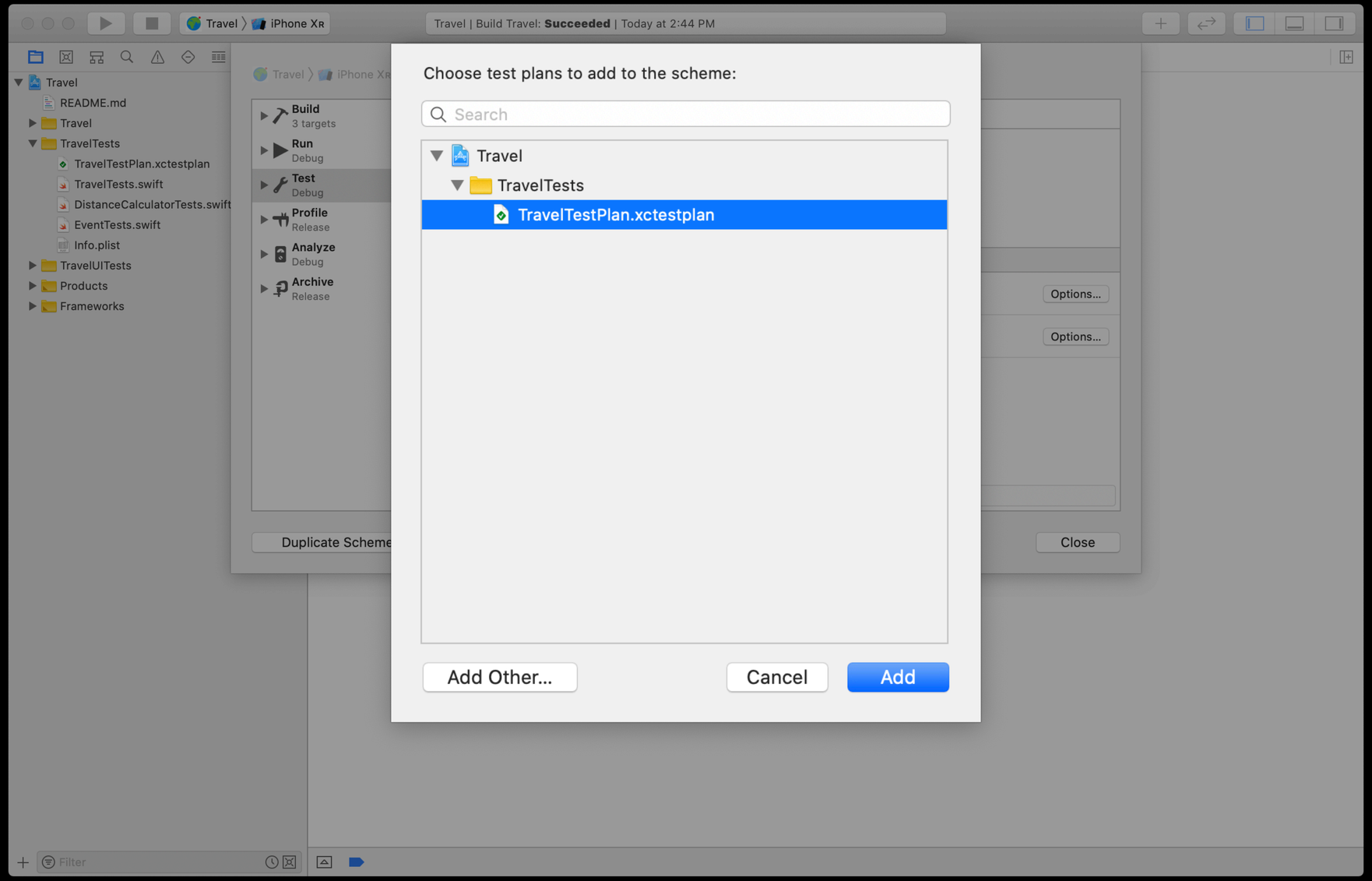
Converting a Scheme to Use Test Plans



Converting a Scheme to Use Test Plans



Converting a Scheme to Use Test Plans



Potential Use Cases

Potential Use Cases

Different sanitizers

ASan

Address Sanitizer

TSan

Thread Sanitizer

Potential Use Cases

Different sanitizers

ASan + UBSan

Address Sanitizer

Undefined Behavior Sanitizer

TSan + UBSan

Thread Sanitizer

Undefined Behavior Sanitizer

Potential Use Cases

Different sanitizers

Shared Settings

ASan + UBSan

Address Sanitizer

Undefined Behavior Sanitizer

TSan + UBSan

Thread Sanitizer

Undefined Behavior Sanitizer

Potential Use Cases

Different sanitizers

Shared Settings

Undefined Behavior Sanitizer

ASan + UBSan

Address Sanitizer

Undefined Behavior Sanitizer

TSan + UBSan

Thread Sanitizer

Undefined Behavior Sanitizer

Potential Use Cases

Simulated languages, locations, and locales

English / United States  $

Korean / South Korea  ₩

Italian / Italy  €

Potential Use Cases

Simulated languages, locations, and locales

Shared Settings

English / United States  $

Korean / South Korea  ₩

Italian / Italy  €

Potential Use Cases

Simulated languages, locations, and locales

Shared Settings

Localization Screenshots

English / United States

$

Localization Screenshots

Korean / South Korea

₩

Localization Screenshots

Italian / Italy

€

Localization Screenshots

Potential Use Cases

Simulated languages, locations, and locales

Shared Settings

Localization Screenshots

English / United States

$

Localization Screenshots

Korean / South Korea

₩

Localization Screenshots

Potential Use Cases

Mix-and-match

Memory Checking

Concurrency

Extra Diagnostics

Potential Use Cases

Mix-and-match

Memory Checking

Address Sanitizer

Zombie Objects

Concurrency

Extra Diagnostics

Potential Use Cases

Mix-and-match

Memory Checking

Address Sanitizer

Zombie Objects

Concurrency

Thread Sanitizer

Undefined Behavior Sanitizer

Random Order

Extra Diagnostics

Potential Use Cases

Mix-and-match

Memory Checking

Address Sanitizer

Zombie Objects

Concurrency

Thread Sanitizer

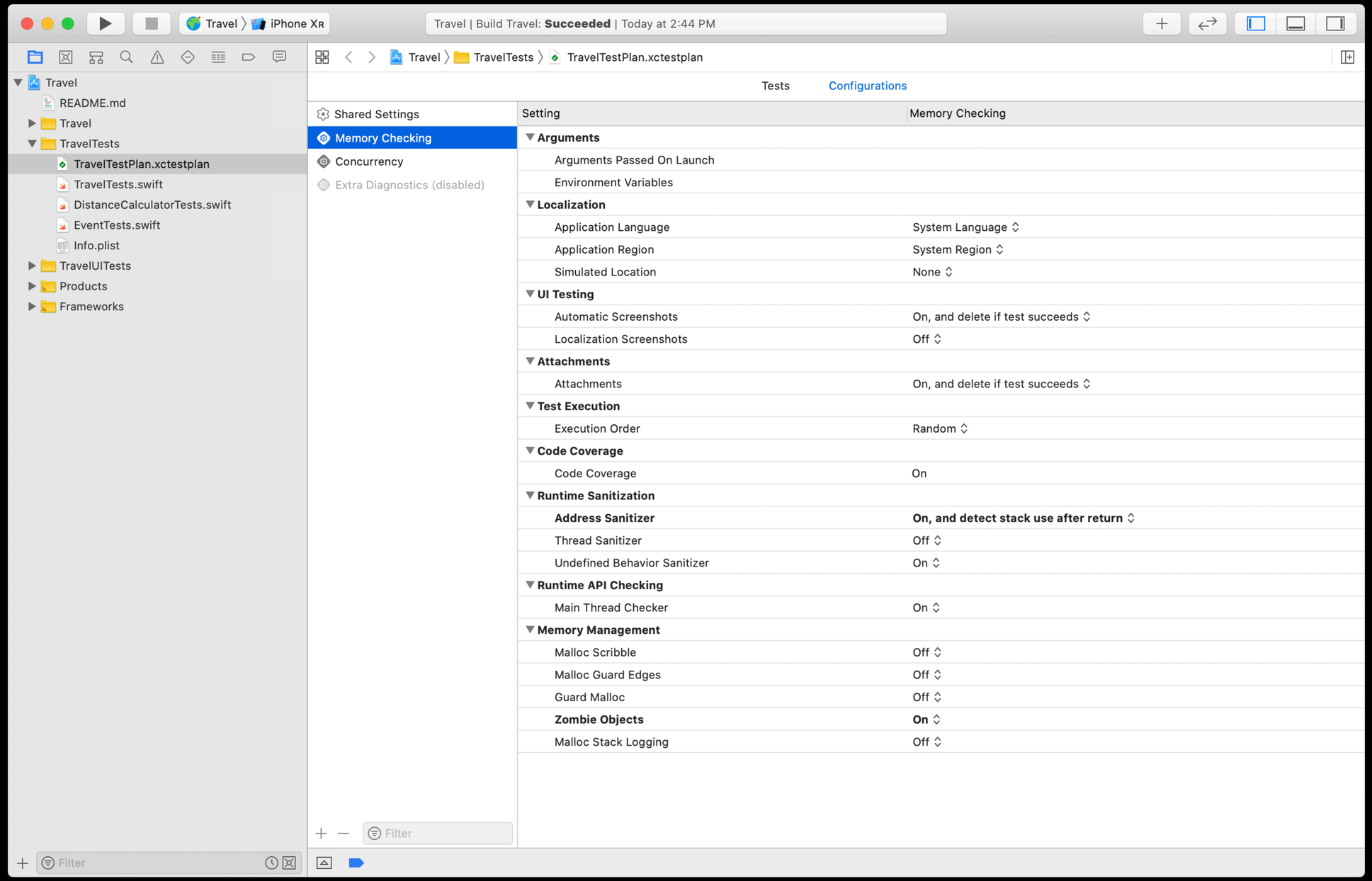
Undefined Behavior Sanitizer

Random Order

Extra Diagnostics

ENABLE_LOGGING=1

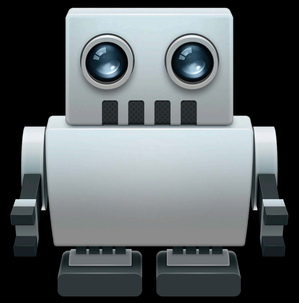
Keep Attachments


Travel 〉 iPhone XR
Travel | Build Travel: Succeeded | Today at 2:44 PM
Travel
README.md
Travel
TravelTests
TravelTestPlan.xctestplan
TravelTests.swift
DistanceCalculatorTests.swift
EventTests.swift
Info.plist
TravelUITests
Products
Frameworks
Filter
Travel 〉 TravelTests 〉 TravelTestPlan.xctestplan
Tests
Configurations
Shared Settings
Memory Checking
Concurrency
Extra Diagnostics (disabled)
Filter
Setting
Memory Checking
Arguments
Arguments Passed On Launch
Environment Variables
Localization
Application Language
System Language
Application Region
System Region
Simulated Location
None
UI Testing
Automatic Screenshots
On, and delete if test succeeds
Localization Screenshots
Off
Attachments
Attachments
On, and delete if test succeeds
Test Execution
Execution Order
Random
Code Coverage
Code Coverage
On
Runtime Sanitization
Address Sanitizer
On, and detect stack use after return
Thread Sanitizer
Off
Undefined Behavior Sanitizer
On
Runtime API Checking
Main Thread Checker
On
Memory Management
Malloc Scribble
Off
Malloc Guard Edges
Off
Guard Malloc
Off
Zombie Objects
On
Malloc Stack Logging
Off

Continuous Integration Workflows

Ethan Vaughan, XCTest Engineer

CI Workflows

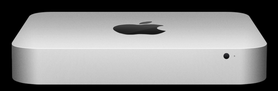


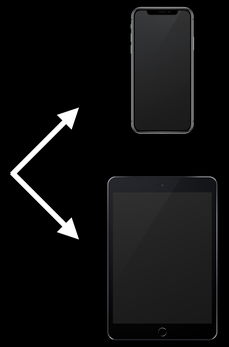
Xcode Server

Custom

Build Your Own CI

Step 1: Build tests

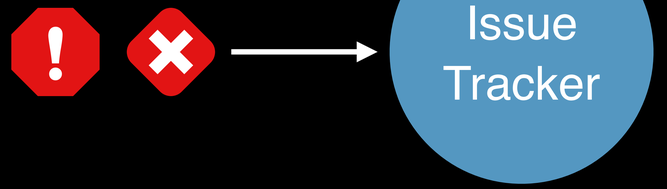




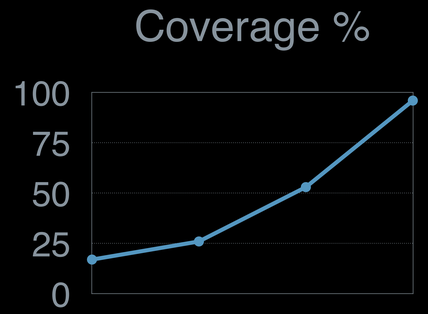
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

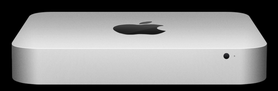


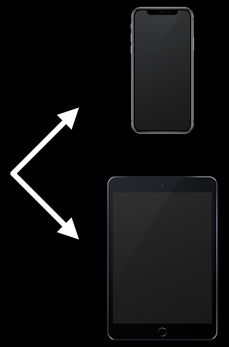
Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests

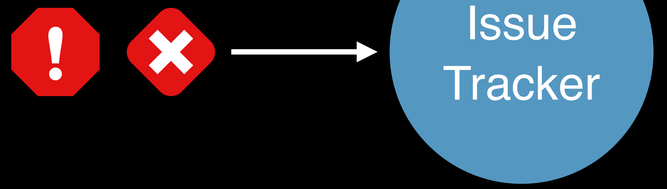




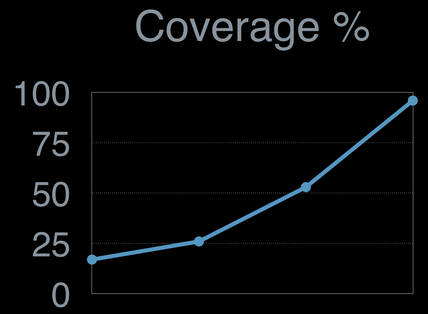
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

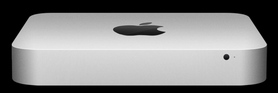


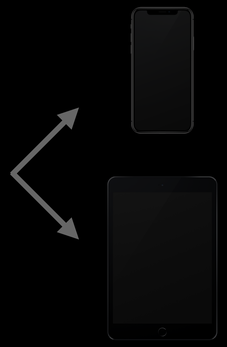
Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests



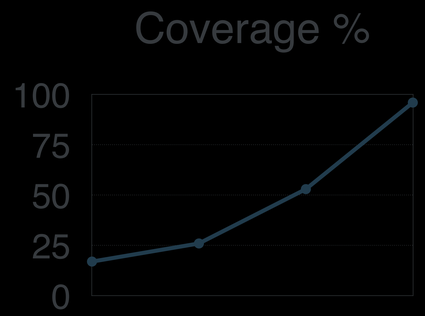


Step 2: Run tests

Build and test results

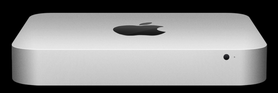
Step 3: Populate issue tracker

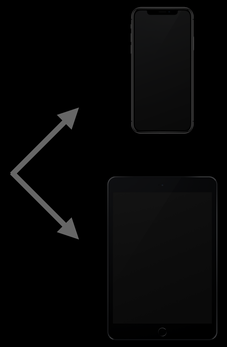
Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests



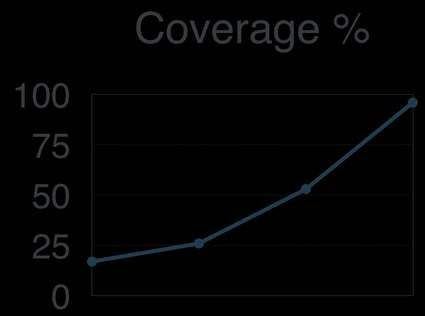


Step 2: Run tests

Build and test results

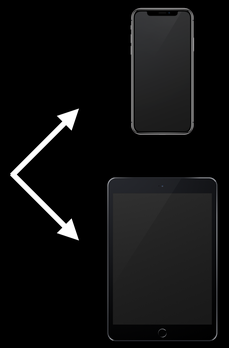
Step 3: Populate issue tracker

Step 4: Track code coverage



Build Your Own CI

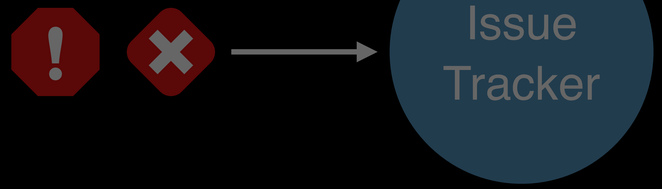
Step 1: Build tests



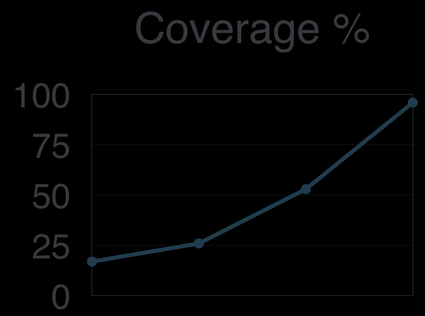
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

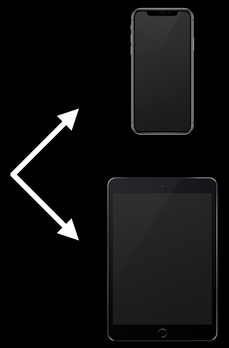


Step 4: Track code coverage



Build Your Own CI

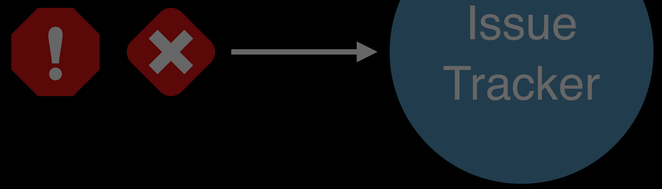
Step 1: Build tests



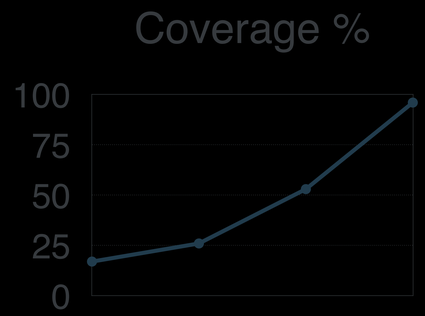
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

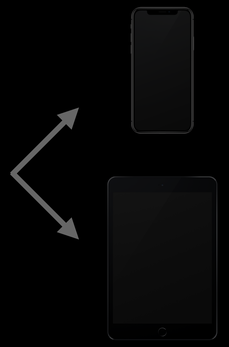


Step 4: Track code coverage



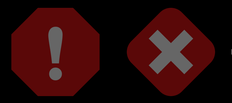
Build Your Own CI

Step 1: Build tests



Step 2: Run tests

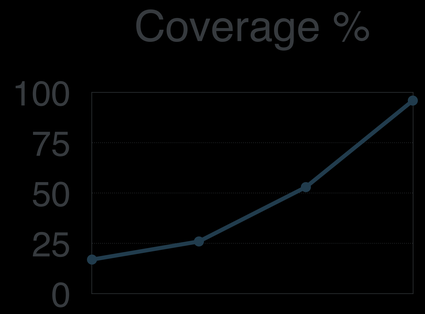
Build and test results



Issue Tracker

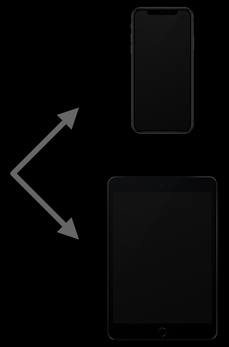
Step 3: Populate issue tracker

Step 4: Track code coverage



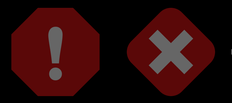
Build Your Own CI

Step 1: Build tests



Step 2: Run tests

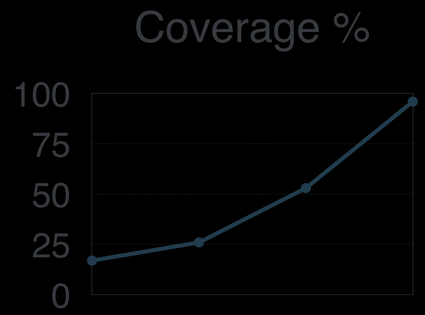
Build and test results



Issue Tracker

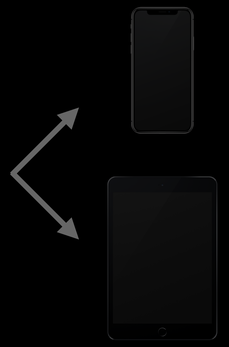
Step 3: Populate issue tracker

Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests



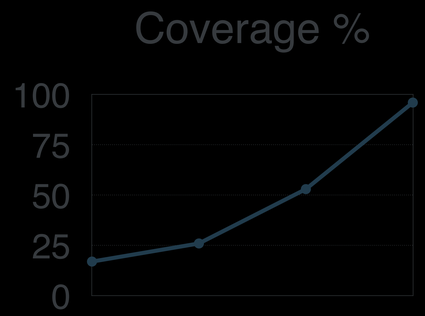
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

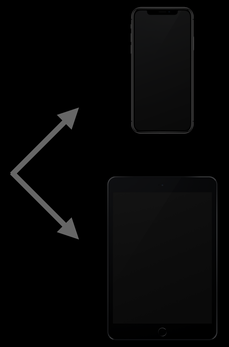
Issue Tracker

Step 4: Track code coverage



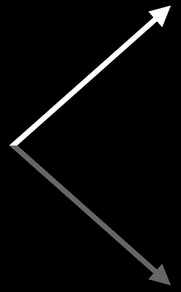
Build Your Own CI

Step 1: Build tests



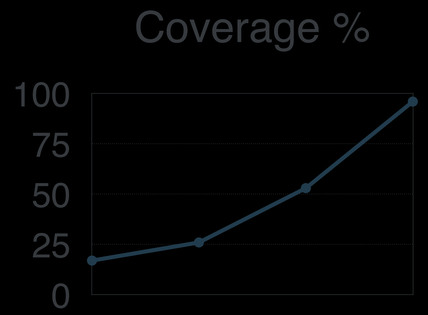
Step 2: Run tests

Build and test results



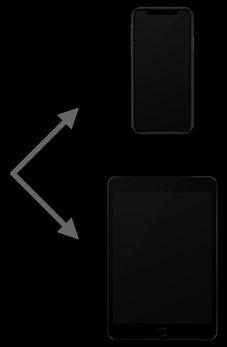
Step 3: Populate issue tracker

Step 4: Track code coverage



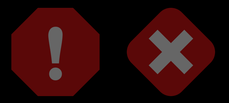
Build Your Own CI

Step 1: Build tests



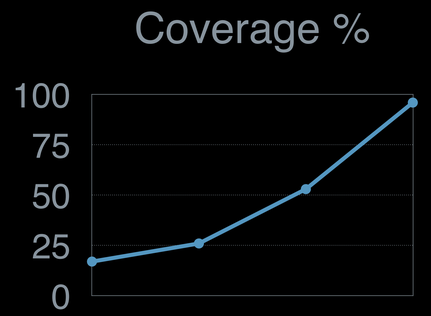
Step 2: Run tests

Build and test results



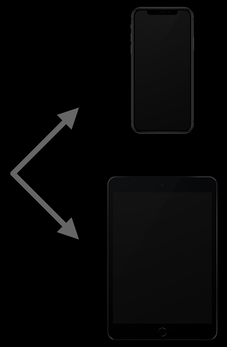
Step 3: Populate issue tracker

Step 4: Track code coverage



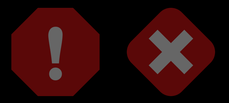
Build Your Own CI

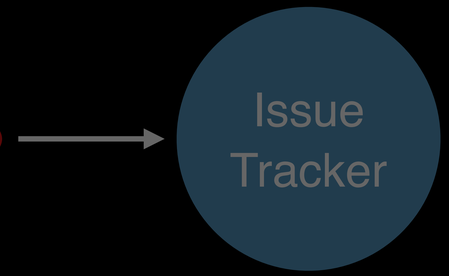
Step 1: Build tests



Step 2: Run tests

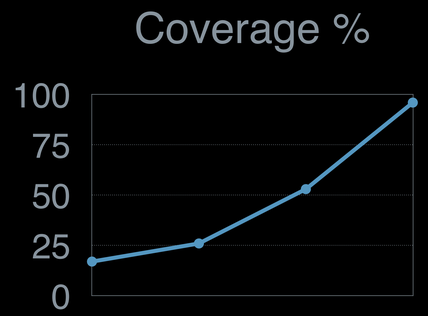
Build and test results





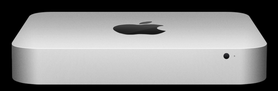
Step 3: Populate issue tracker

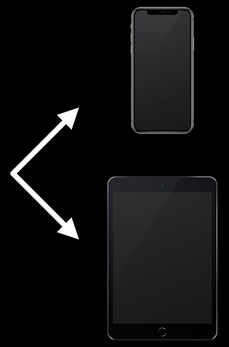
Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests

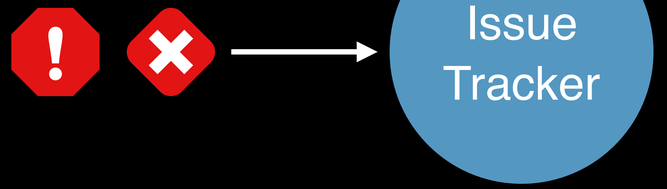




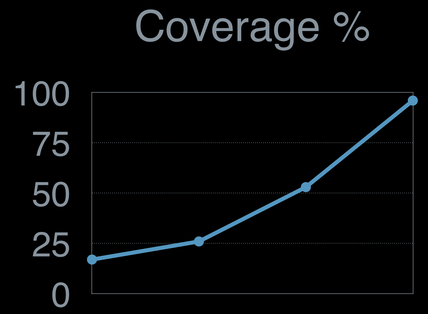
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

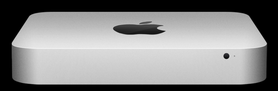


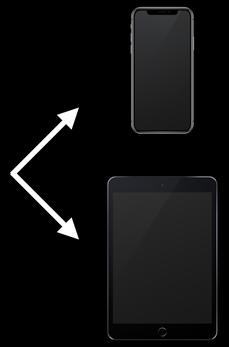
Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests

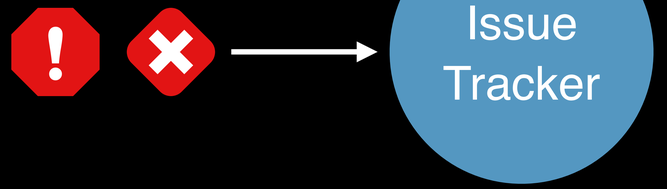




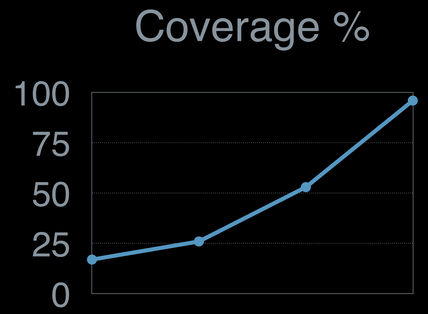
Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

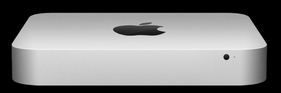


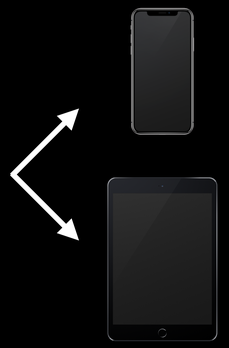
Step 4: Track code coverage



Build Your Own CI

Step 1: Build tests



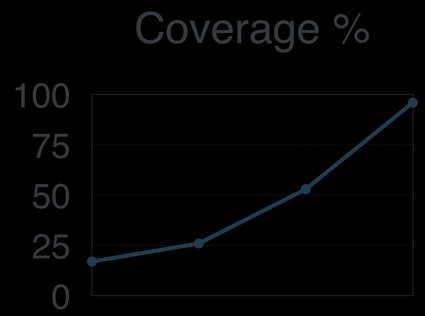


Step 2: Run tests

Build and test results

Step 3: Populate issue tracker

Step 4: Track code coverage



xcodebuild

Running Tests with xcodebuild

Build and test together

```
$ xcodebuild test
 -project MyProject.xcodeproj
 -scheme MyScheme
 -destination 'platform=iOS,name=My iPhone'
```

Running Tests with xcodebuild

Build, then test

```
$ xcodebuild build-for-testing
 -project MyProject.xcodeproj
 -scheme MyScheme
 -destination 'platform=iOS,name=My iPhone'


$ xcodebuild test-without-building
 -xctestrun MyProject.xctestrun
 -destination 'platform=iOS,name=My iPhone'
```

Running Tests with xcodebuild

Build, then test

```
$ xcodebuild build-for-testing
 -project MyProject.xcodeproj
 -scheme MyScheme
 -destination 'platform=iOS,name=My iPhone'
```

```
$ xcodebuild test-without-building
 -xctestrun MyProject.xctestrun
 -destination 'platform=iOS,name=My iPhone'
```

Running Tests with xcodebuild

Build, then test

```
$ xcodebuild build-for-testing
 -project MyProject.xcodeproj
 -scheme MyScheme
 -destination 'platform=iOS,name=My iPhone'
```

Build products

+

.xctestrun file

```
$ xcodebuild test-without-building
 -xctestrun MyProject.xctestrun
 -destination 'platform=iOS,name=My iPhone'
```

Running Tests with xcodebuild

Build, then test

```
$ xcodebuild build-for-testing
 -project MyProject.xcodeproj
 -scheme MyScheme
 -destination 'platform=iOS,name=My iPhone'
```

```
$ xcodebuild test-without-building
 -xctestrun MyProject.xctestrun
 -destination 'platform=iOS,name=My iPhone'
```

Build products

+

.xctestrun file

xctestrun File Format

```
$ man xcodebuild.xctestrun
```

```
NAME
     xcodebuild.xctestrun -- Test run parameters files for xcodebuild

DESCRIPTION
     This document details the parameters contained in an xctestrun file.
     …
```

Testing on Multiple Destinations

```
$ xcodebuild test
 -destination 'platform=iOS,name=My iPad'
 -destination 'platform=iOS,name=My iPad mini'
 -destination 'platform=iOS,name=My iPhone'
```

Testing on Multiple Destinations

```
$ xcodebuild test
 -destination 'platform=iOS,name=My iPad'
 -destination 'platform=iOS,name=My iPad mini'
 -destination 'platform=iOS,name=My iPhone'
```

xcodebuild and Test Plans

NEW

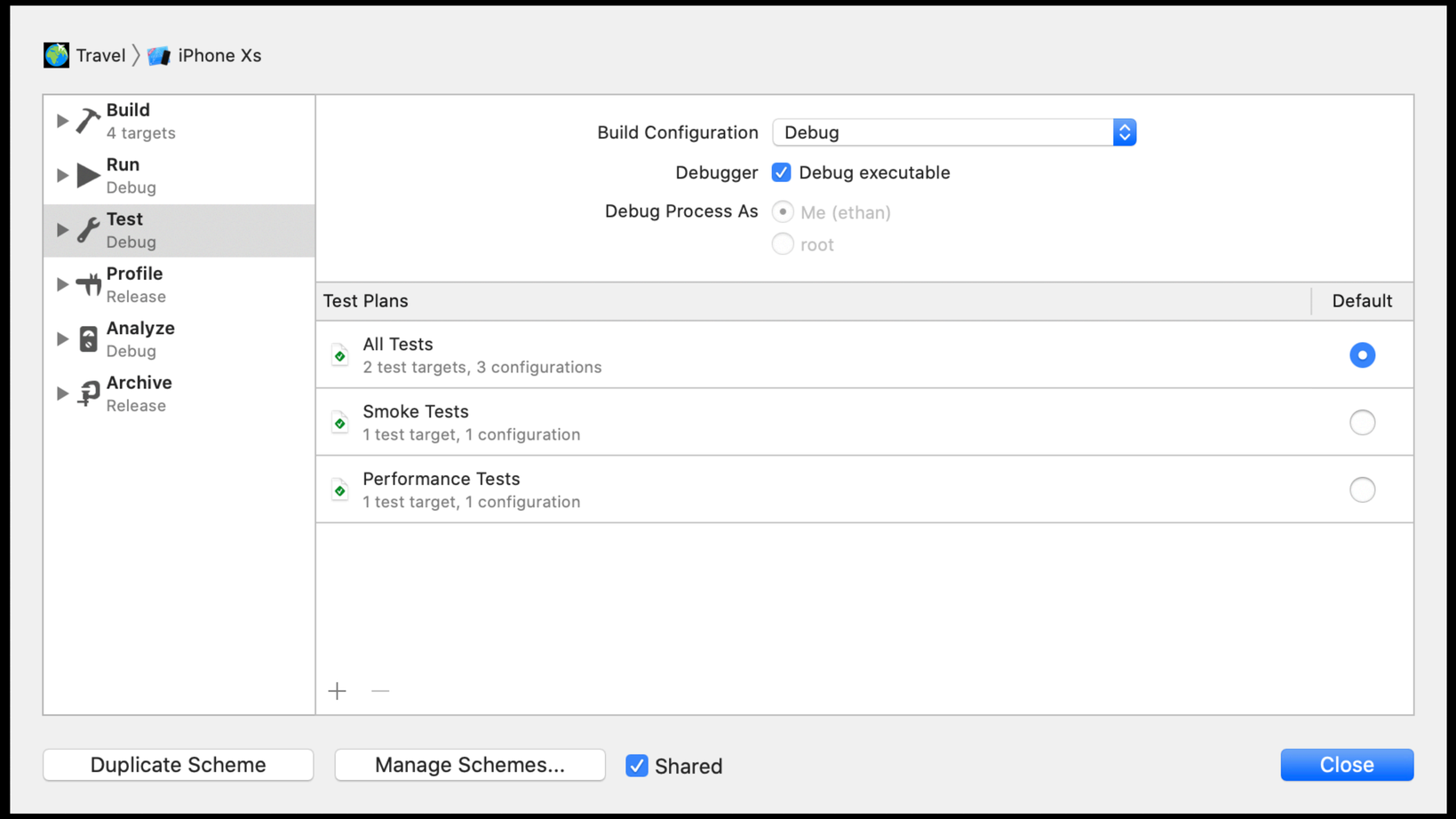
xcodebuild and Test Plans

NEW

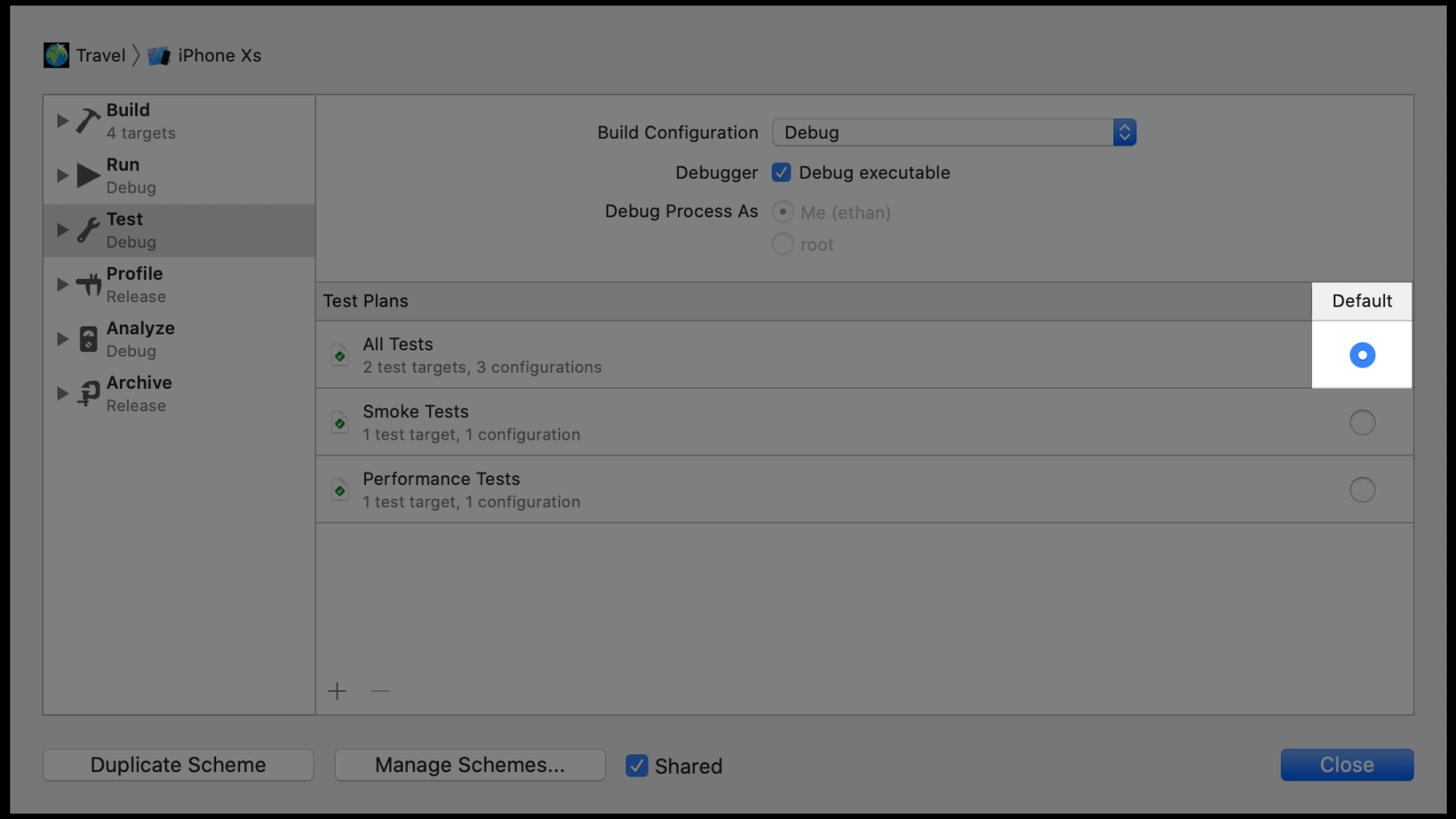
List a scheme’s test plans

```
$ xcodebuild -project ... -scheme ... -showTestPlans
```

Multiple Test Plans



Multiple Test Plans



xcodebuild and Test Plans

NEW

List a scheme’s test plans

```
$ xcodebuild -project ... -scheme ... -showTestPlans
```

xcodebuild and Test Plans

NEW

List a scheme’s test plans

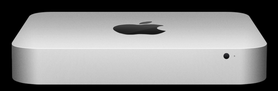
```
$ xcodebuild -project ... -scheme ... -showTestPlans
```

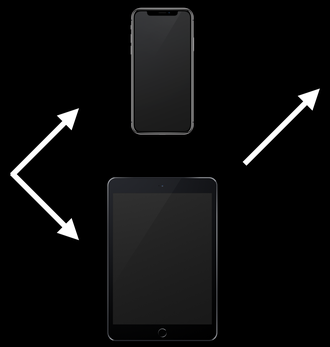
Override default test plan

```
$ xcodebuild test -project ... -scheme ... -testPlan 'Smoke Tests'
```

Build Your Own CI

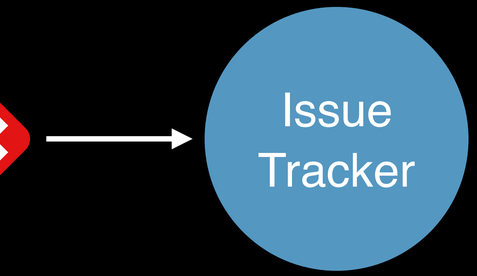
Step 1: Build tests





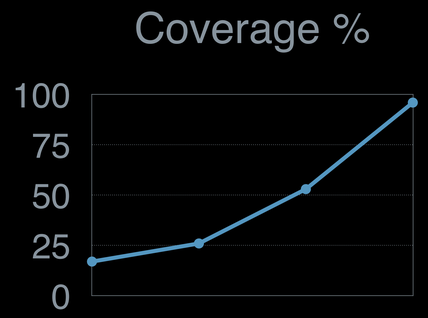
Step 2: Run tests

Build and test results



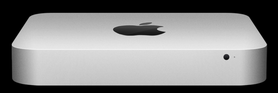
Step 3: Populate issue tracker

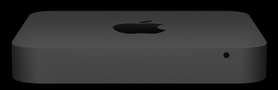
Step 4: Track code coverage

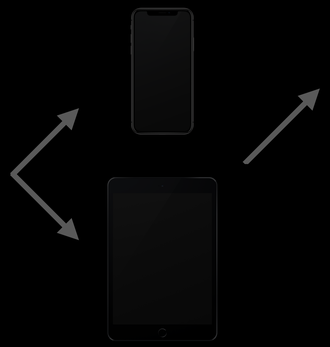


Build Your Own CI

Step 1: Build tests





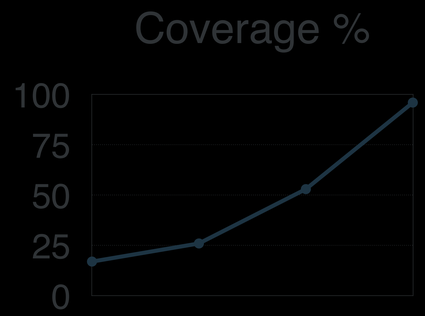


Step 2: Run tests

Build and test results

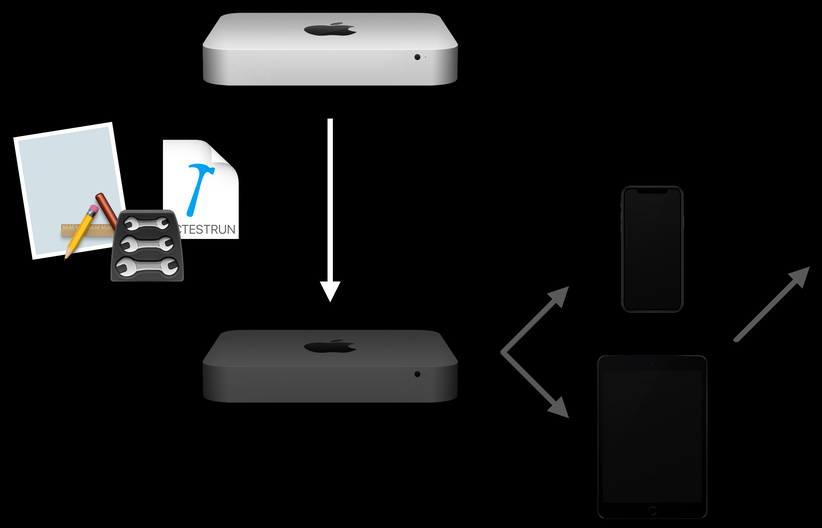
Step 3: Populate issue tracker

Step 4: Track code coverage



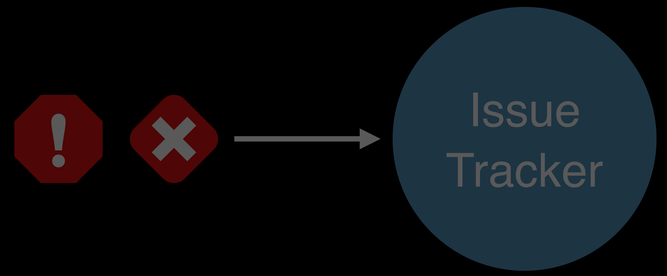
Build Your Own CI

```
xcodebuild build-for-testing
```



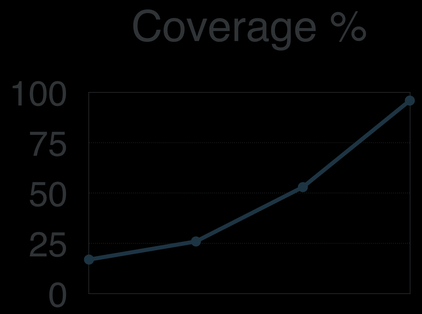
Step 2: Run tests

Build and test results



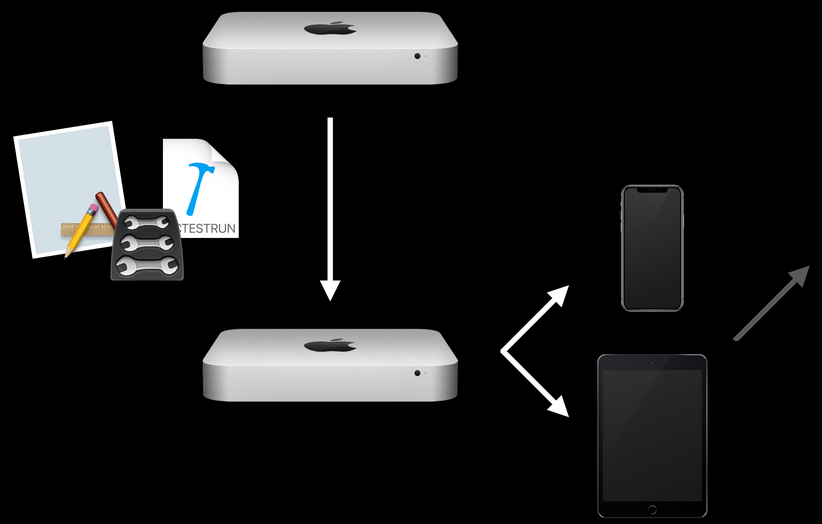
Step 3: Populate issue tracker

Step 4: Track code coverage



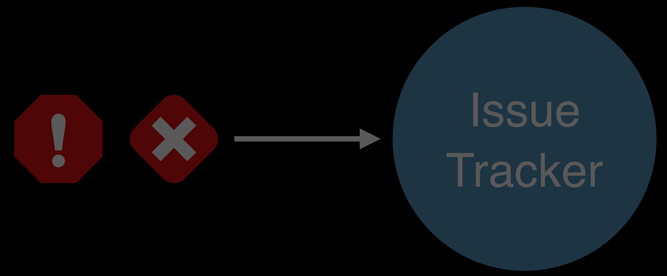
Build Your Own CI

```
xcodebuild build-for-testing
```



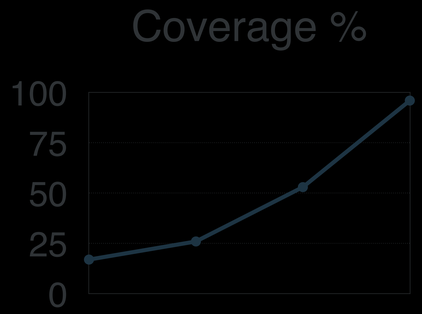
Step 2: Run tests

Build and test results



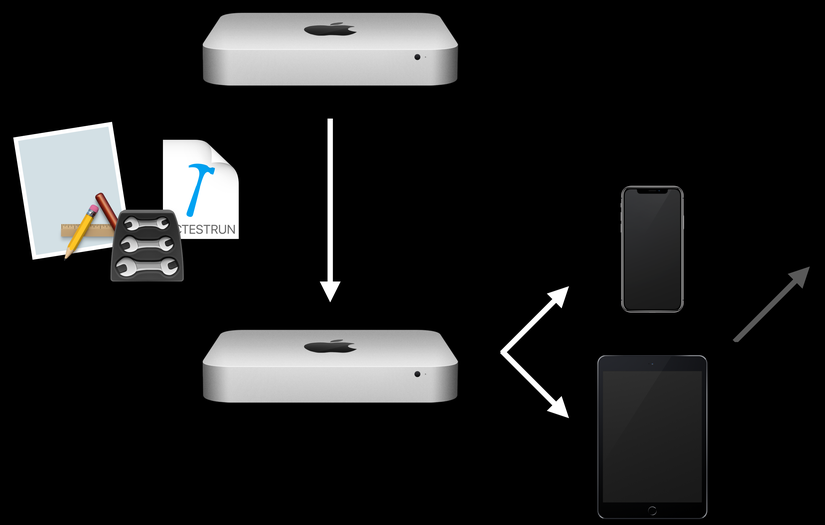
Step 3: Populate issue tracker

Step 4: Track code coverage



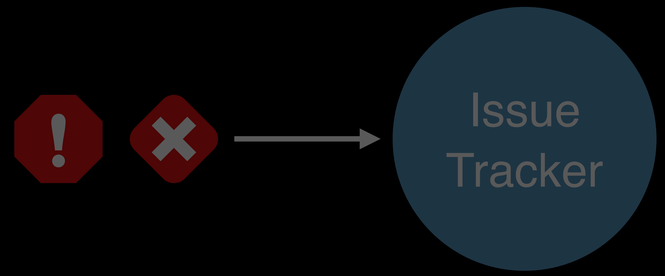
Build Your Own CI

```
xcodebuild build-for-testing
```



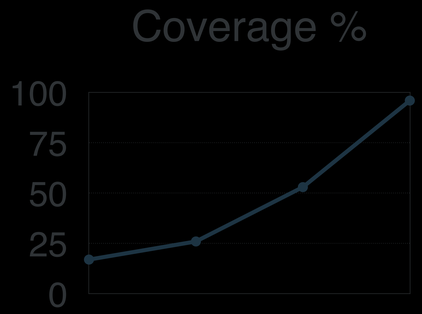

```
xcodebuild test-without-building
 -destination 'name=iPhone X'
 -destination 'name=iPad'
```

Build and test results



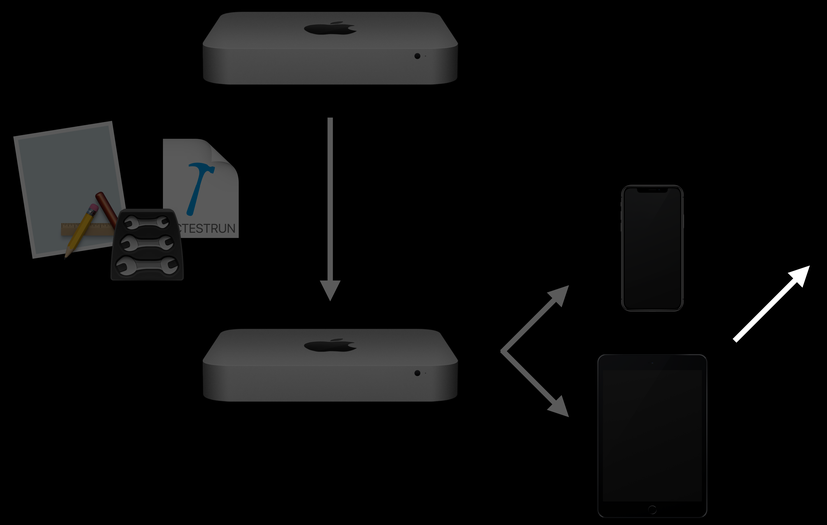
Step 3: Populate issue tracker

Step 4: Track code coverage



Build Your Own CI

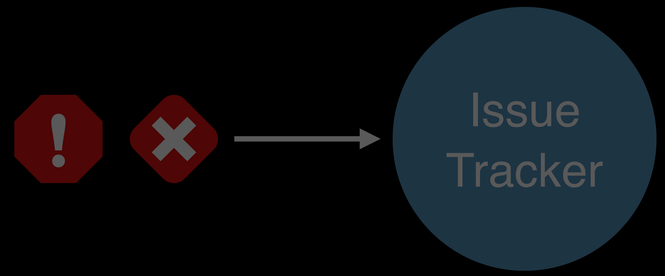
```
xcodebuild build-for-testing
```



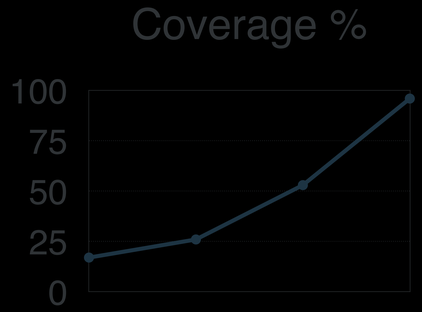

```
xcodebuild test-without-building
 -destination 'name=iPhone X'
 -destination 'name=iPad'
```

Build and test results

Step 3: Populate issue tracker



Step 4: Track code coverage



Result Bundles

What is a Result Bundle?

File containing the results of building and running tests

What is a Result Bundle?

File containing the results of building and running tests

- Build log

What is a Result Bundle?

File containing the results of building and running tests

- Build log
- Test report

What is a Result Bundle?

File containing the results of building and running tests

- Build log
- Test report
- Code coverage report

What is a Result Bundle?

File containing the results of building and running tests

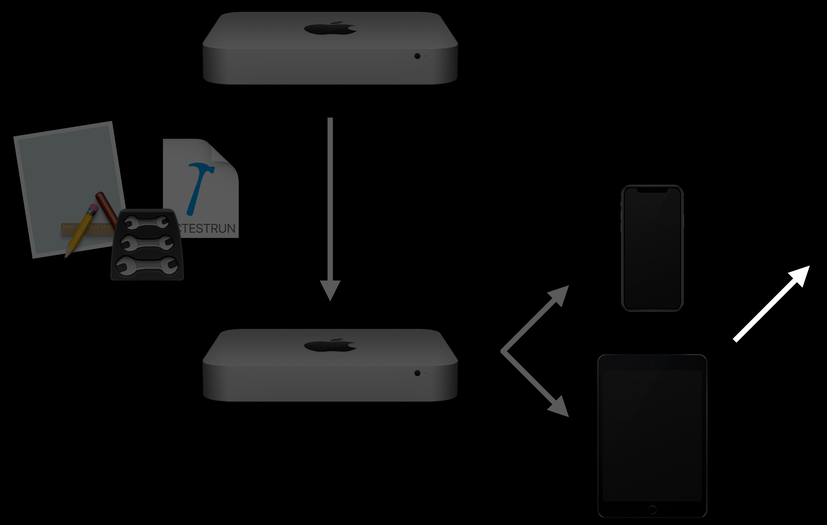
- Build log
- Test report
- Code coverage report
- Test attachments

Producing Result Bundles

```
$ xcodebuild test
-project MyProject.xcodeproj
-scheme MyScheme
-resultBundlePath /path/to/ResultBundle.xcresult
```

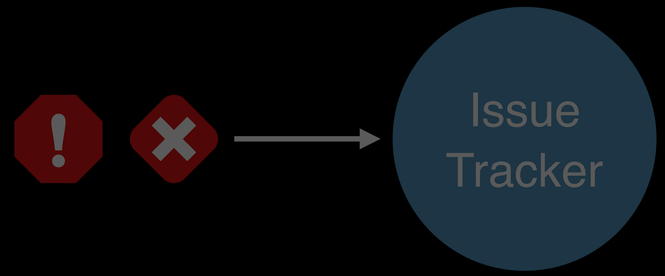
Build Your Own CI

```
xcodebuild build-for-testing
```



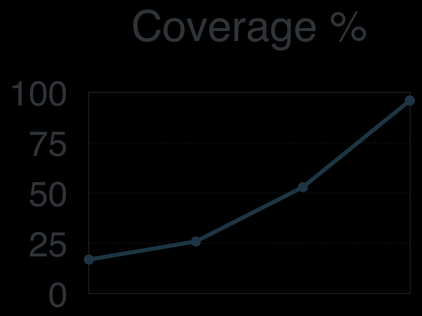

```
xcodebuild test-without-building
 -destination 'name=iPhone X'
 -destination 'name=iPad'
```

Build and test results



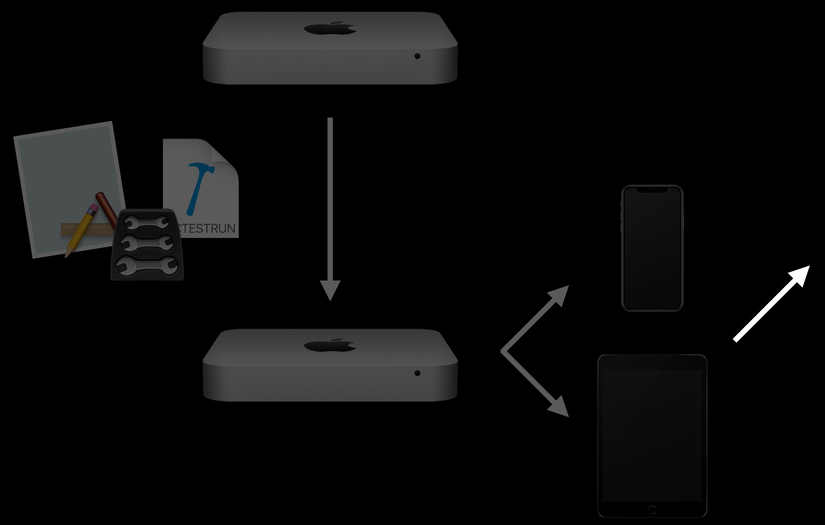
Step 3: Populate issue tracker

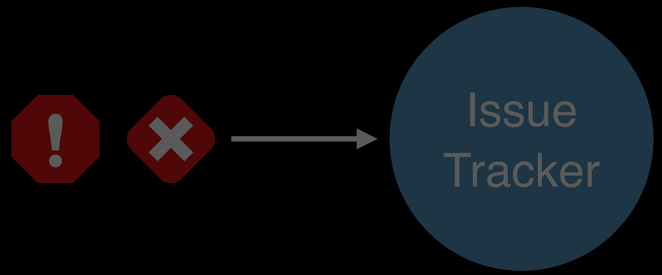
Step 4: Track code coverage



Build Your Own CI

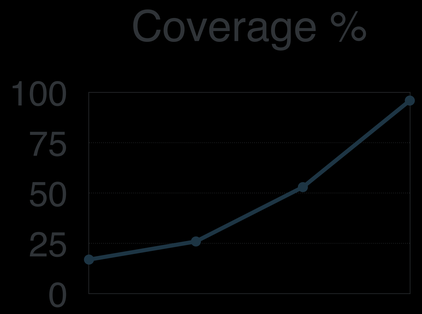
```
xcodebuild build-for-testing
```





Step 3: Populate issue tracker

Step 4: Track code coverage




```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Result Bundle Format

NEW

Result Bundle Format

NEW

Highly optimized, space-efficient format (~4x smaller)

Result Bundle Format

NEW

Highly optimized, space-efficient format (~4x smaller)

Viewable in Xcode

Result Bundle Format

NEW

Highly optimized, space-efficient format (~4x smaller)

Viewable in Xcode

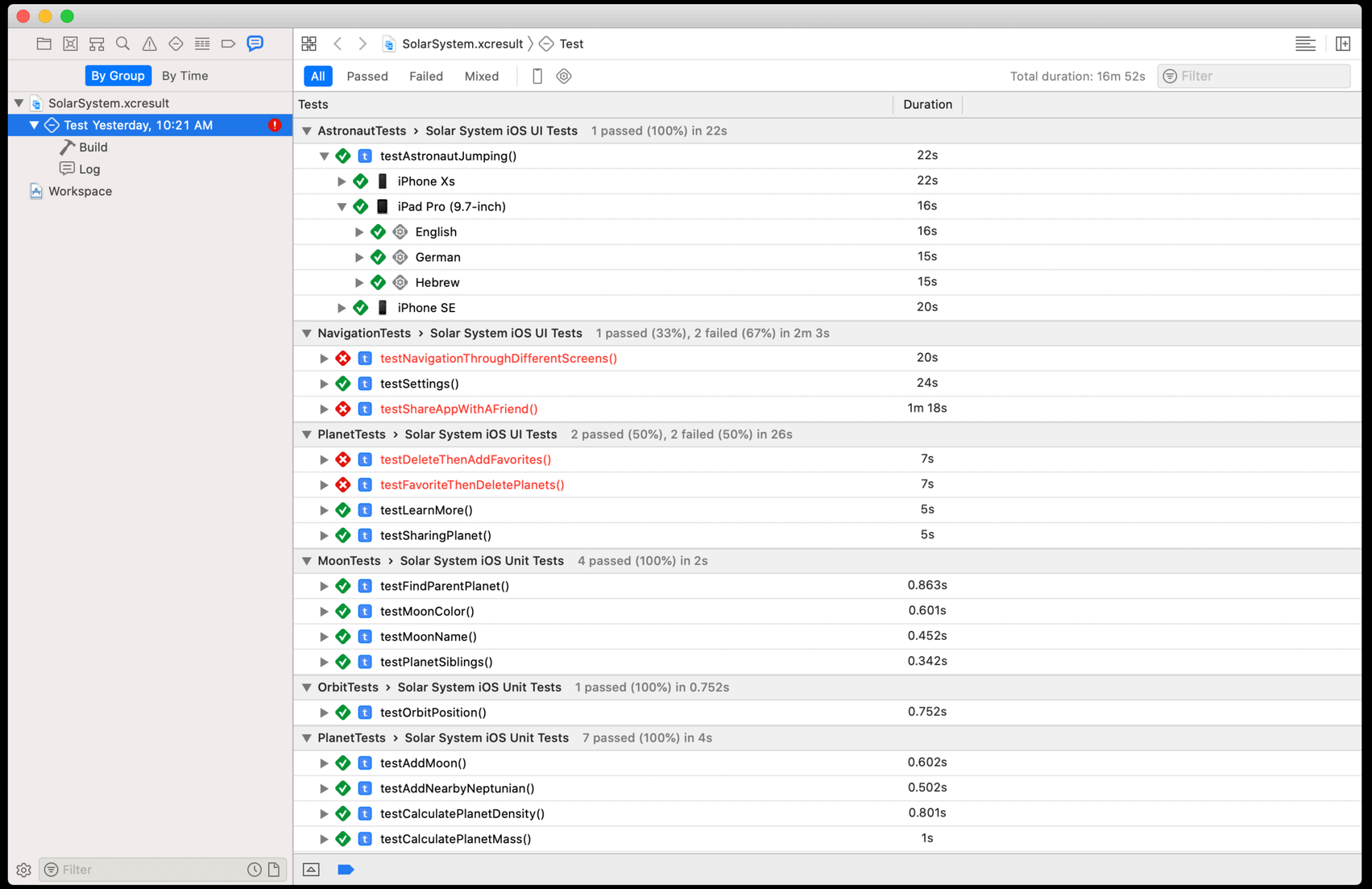
Programmatically accessible contents

Viewing Result Bundles

NEW

Viewing Result Bundles

NEW



xcresulttool

xcresulttool

NEW

xcresulttool

NEW

Programmatic access to result bundle contents

xcresulttool

NEW

Programmatic access to result bundle contents

Structured data available as JSON

xcresulttool

NEW

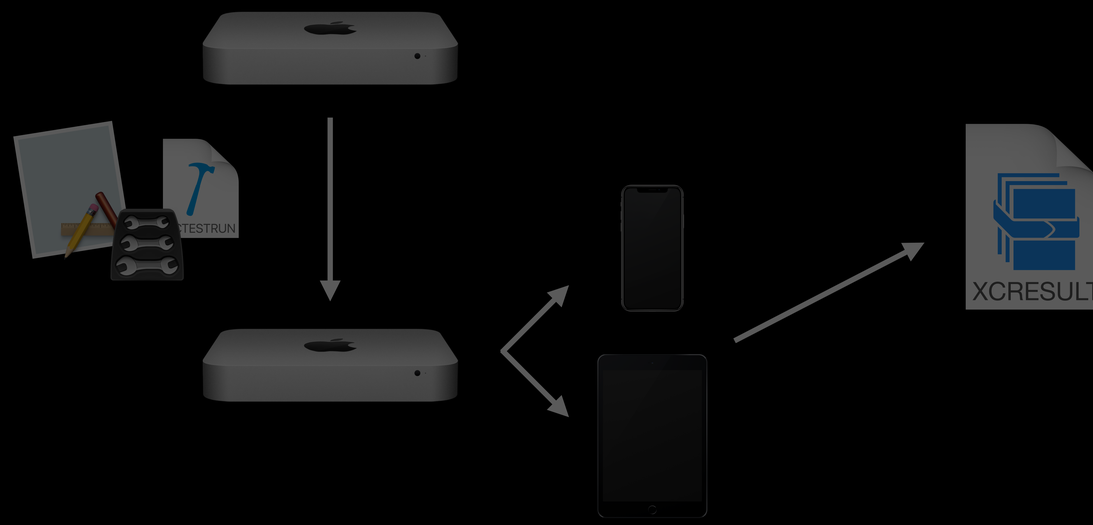
Programmatic access to result bundle contents

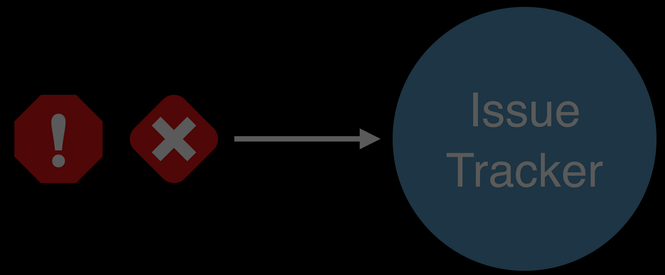
Structured data available as JSON

JSON schema is publicly documented

Build Your Own CI

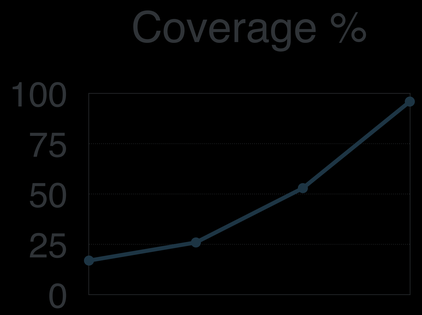
```
xcodebuild build-for-testing
```





Step 3: Populate issue tracker

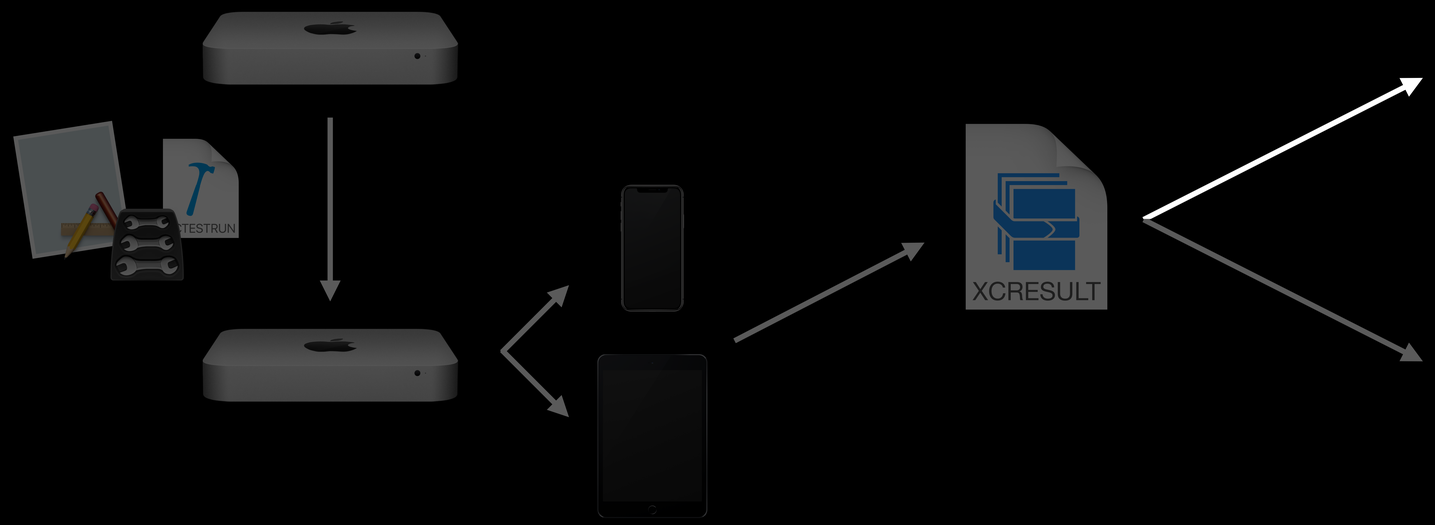
Step 4: Track code coverage

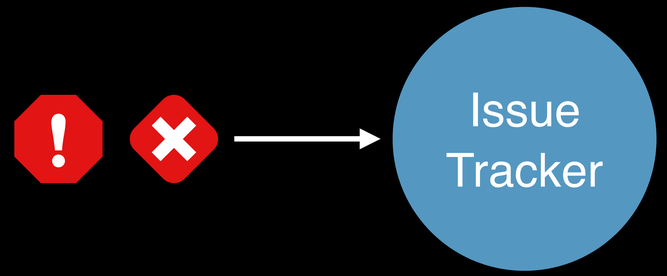



```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Build Your Own CI

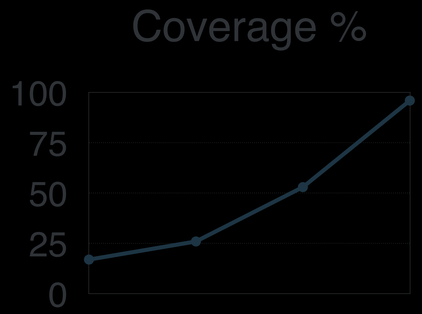
```
xcodebuild build-for-testing
```





Step 3: Populate issue tracker

Step 4: Track code coverage




```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Extracting Build and Test Failures

Extracting Build and Test Failures

```
$ xcrun xcresulttool get --path ResultBundle.xcresult --format json
```

Extracting Build and Test Failures

```
$ xcrun xcresulttool get --path ResultBundle.xcresult --format json
```

```
{
  "issues" : {
    "errorSummaries" : {
      "_values" : [
        {
          "message" : {
            "_value" : "Use of unresolved identifier 'garbage'"
          },
          ...
```

Extracting Build and Test Failures

```
$ xcrun xcresulttool get --path ResultBundle.xcresult --format json
```

```
{
  "issues" : {
    "errorSummaries" : {
      "_values" : [
        {
          "message" : {
            "_value" : "Use of unresolved identifier 'garbage'"
          },
          ...
```

Extracting Build and Test Failures

```
$ xcrun xcresulttool get --path ResultBundle.xcresult --format json
```

```
{
  "issues" : {
    "testFailureSummaries" : {
      "_values" : [
        {
          "message" : {
            "_value" : "failed - ("1") is not equal to ("2")"
          },
          "testCaseName" : {
            "_value" : "Calculator.testAddition()"
          }
```

Extracting Build and Test Failures

```
$ xcrun xcresulttool get --path ResultBundle.xcresult --format json
```

```
{
  "issues" : {
    "testFailureSummaries" : {
      "_values" : [
        {
          "message" : {
            "_value" : "failed - ("1") is not equal to ("2")"
          },
          "testCaseName" : {
            "_value" : "Calculator.testAddition()"
          }
```

Publicly Documented Format

```
$ xcrun xcresulttool formatDescription get
```

```
Name: Xcode Result Types
Version: 3.18
Types:
  - ActionAbstractTestSummary
    * Kind: object
    * Properties:
      + name: String?
  - ActionDeviceRecord
    * Kind: object
    * Properties:
    ...
```

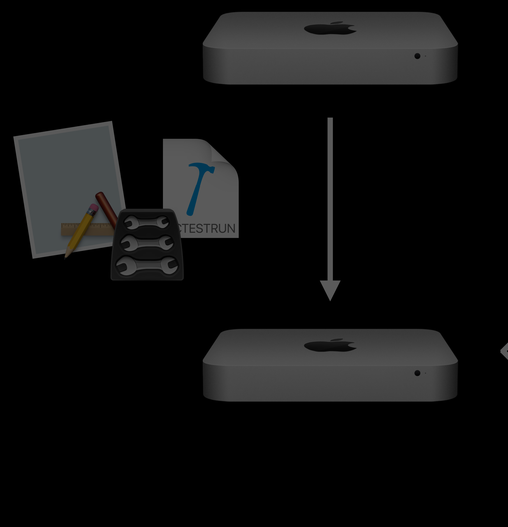
Further Documentation

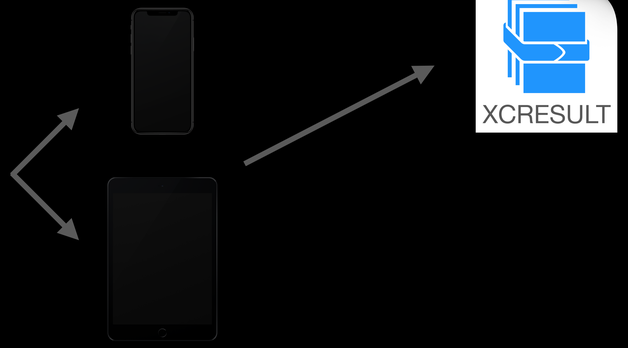
```
$ man xcresulttool
```

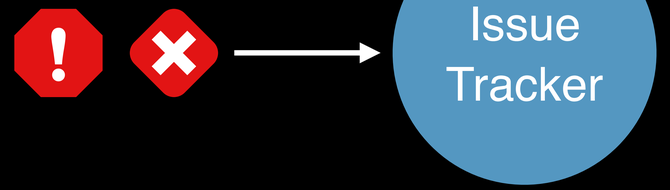
```
xcresulttool(1)

NAME
       xcresulttool – view Xcode result bundle data in a human-readable or machine-parseable
       format.

SYNOPSIS
       xcresulttool get --path Example.xcresult [--format json | raw] [--id ID]

       ...
```

Build Your Own CI

```
xcodebuild build-for-testing
```

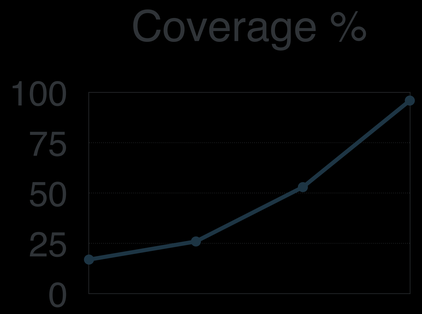






Step 3: Populate issue tracker

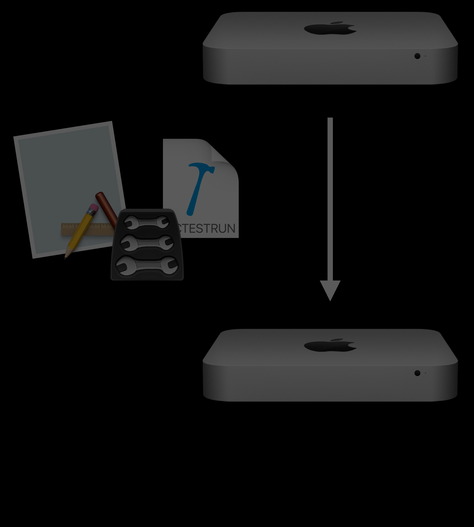
Step 4: Track code coverage

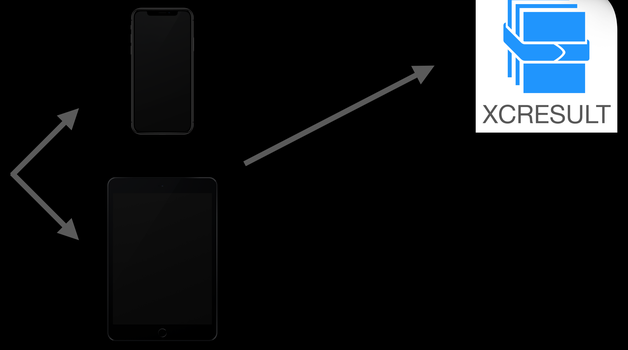


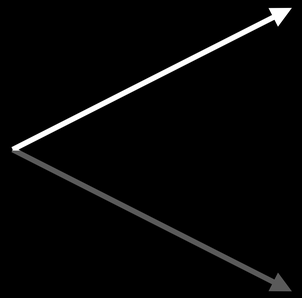

```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Build Your Own CI

```
xcodebuild build-for-testing
```



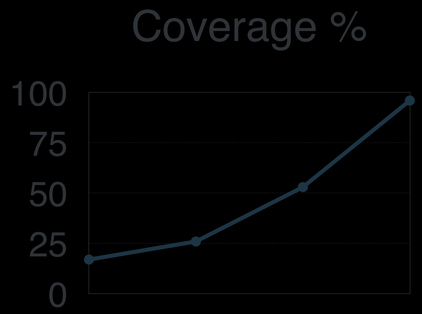




Issue Tracker

```
xcresulttool get
```

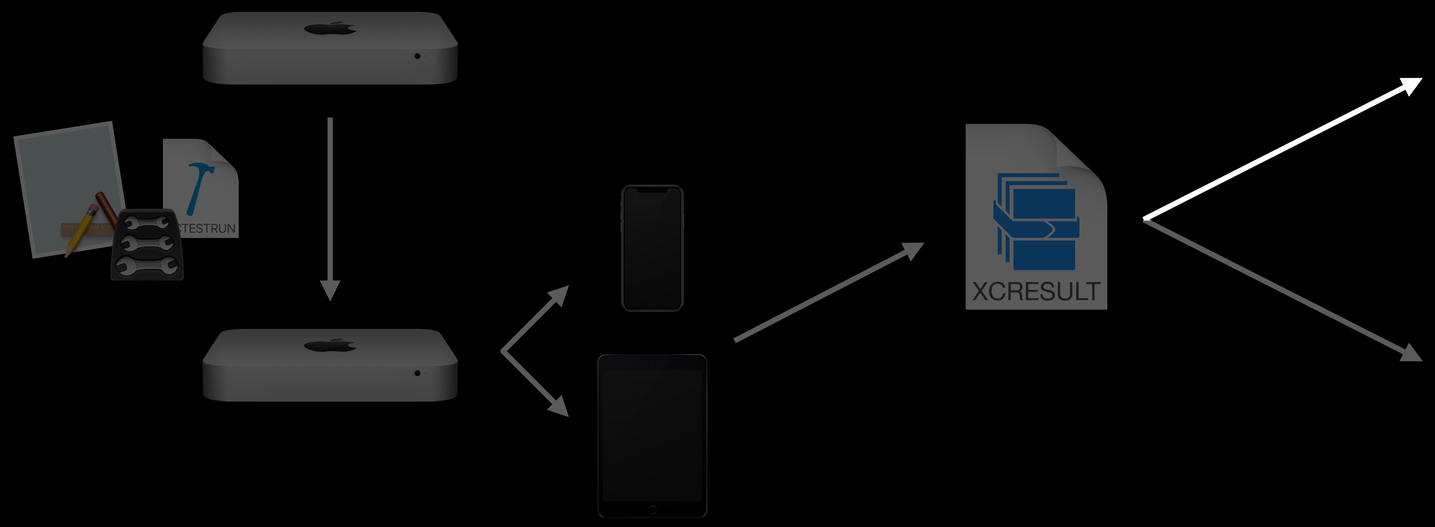
Step 4: Track code coverage

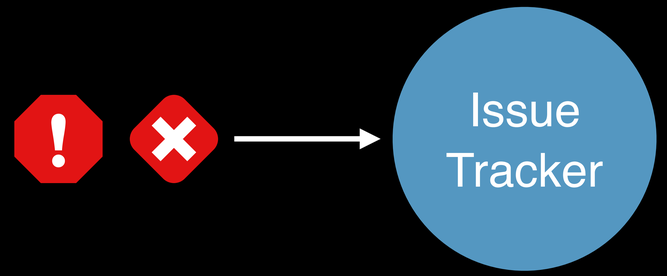



```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Build Your Own CI

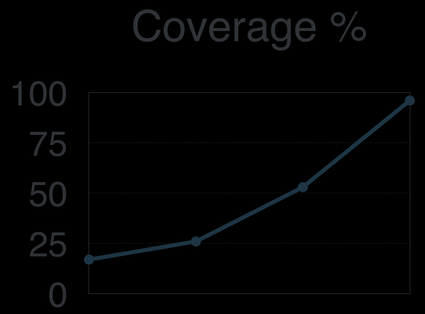
```
xcodebuild build-for-testing
```






```
xcresulttool get
```

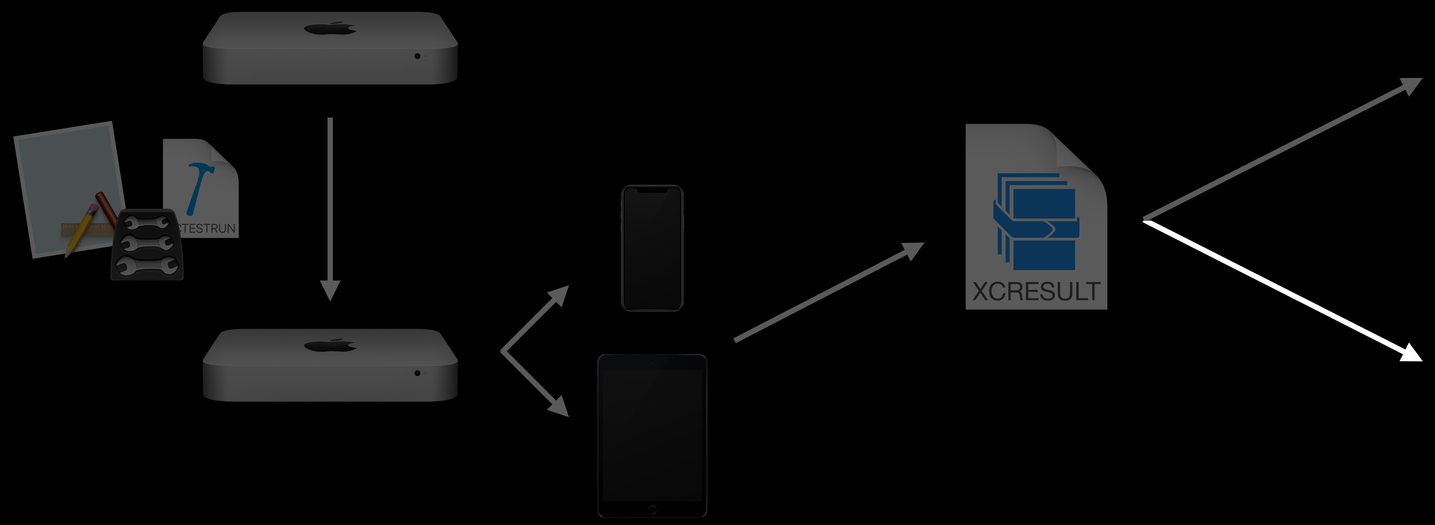
Step 4: Track code coverage

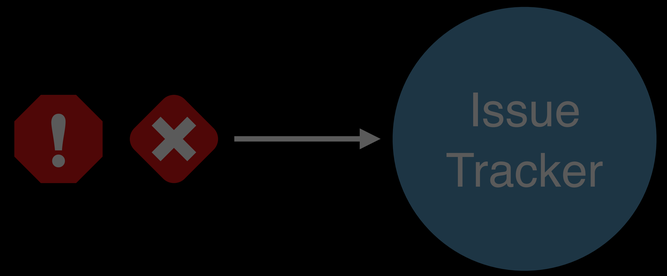



```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Build Your Own CI

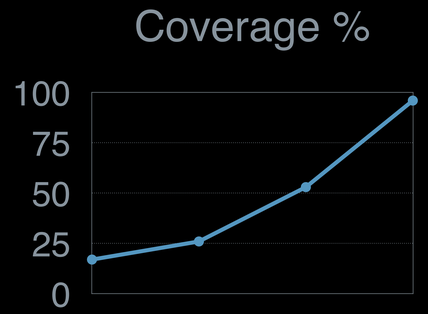
```
xcodebuild build-for-testing
```






```
xcresulttool get
```

Step 4: Track code coverage




```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Code Coverage Reports with xccov

Code Coverage Reports with xccov

```
$ xcrun xccov view --report ResultBundle.xcresult
```

Code Coverage Reports with xccov

```
$ xcrun xccov view --report ResultBundle.xcresult
```

```
Name                                                                                                Coverage
--------------------------------------------------------------------------------------------------- ---------------
HelloWorld.app                                                                                      57.14% (8/14)
    /tmp/HelloWorld/HelloWorld/AppDelegate.swift                                                    50.00% (3/6)
        AppDelegate.applicationDidFinishLaunching(Notification) -> ()                               100.00% (3/3)
        AppDelegate.applicationWillTerminate(Notification) -> ()                                    0.00% (0/3)
    /tmp/HelloWorld/HelloWorld/ViewController.swift                                                 62.50% (5/8)
        ViewController.viewDidLoad() -> ()                                                          100.00% (5/5)
        ViewController.representedObject.didset : Swift.Optional<Any>                               0.00% (0/3)
```

Code Coverage Reports with xccov

```
$ xcrun xccov view --report ResultBundle.xcresult
```

```
Name                                                                                                Coverage
--------------------------------------------------------------------------------------------------- --------------
HelloWorld.app                                                                                      57.14% (8/14)
    /tmp/HelloWorld/HelloWorld/AppDelegate.swift                                                    50.00% (3/6)
        AppDelegate.applicationDidFinishLaunching(Notification) -> ()                               100.00% (3/3)
        AppDelegate.applicationWillTerminate(Notification) -> ()                                    0.00% (0/3)
    /tmp/HelloWorld/HelloWorld/ViewController.swift                                                 62.50% (5/8)
        ViewController.viewDidLoad() -> ()                                                          100.00% (5/5)
        ViewController.representedObject.didset : Swift.Optional<Any>                               0.00% (0/3)
```

Code Coverage Reports with xccov

```
$ xcrun xccov view --report ResultBundle.xcresult
```

```
Name                                                                    Coverage
----------------------------------------------------------------------- ---------------
HelloWorld.app                                                          57.14% (8/14)
    /tmp/HelloWorld/HelloWorld/AppDelegate.swift                        50.00% (3/6)
        AppDelegate.applicationDidFinishLaunching(Notification) -> ()   100.00% (3/3)
        AppDelegate.applicationWillTerminate(Notification) -> ()        0.00% (0/3)
    /tmp/HelloWorld/HelloWorld/ViewController.swift                     62.50% (5/8)
        ViewController.viewDidLoad() -> ()                              100.00% (5/5)
        ViewController.representedObject.didset : Swift.Optional<Any>   0.00% (0/3)
```

Code Coverage Reports with xccov

```
$ xcrun xccov view --report ResultBundle.xcresult
```

```
Name                                                                    Coverage
----------------------------------------------------------------------- ---------------
HelloWorld.app                                                          57.14% (8/14)
    /tmp/HelloWorld/HelloWorld/AppDelegate.swift                        50.00% (3/6)
        AppDelegate.applicationDidFinishLaunching(Notification) -> ()   100.00% (3/3)
        AppDelegate.applicationWillTerminate(Notification) -> ()        0.00% (0/3)
    /tmp/HelloWorld/HelloWorld/ViewController.swift                     62.50% (5/8)
        ViewController.viewDidLoad() -> ()                              100.00% (5/5)
        ViewController.representedObject.didset : Swift.Optional<Any>   0.00% (0/3)
```

Comparing Code Coverage Reports

Comparing Code Coverage Reports

```
$ xcrun xccov diff --json Before.xcresult After.xcresult
```

Comparing Code Coverage Reports

```
$ xcrun xccov diff --json Before.xcresult After.xcresult
```

```
{
  "documentLocation": "\/tmp\/HelloWorld\/HelloWorld\/AppDelegate.swift",
  "lineCoverageDelta": {
    "executableLinesDelta": 0,
    "coveredLinesDelta": 3,
    "lineCoverageDelta": 0.5
  },
  "functionDeltas": [
    {
      "name": "AppDelegate.applicationWillTerminate(Notification) -> ()",
      ...
```

Comparing Code Coverage Reports

```
$ xcrun xccov diff --json Before.xcresult After.xcresult
```

```
{
  "documentLocation": "\/tmp\/HelloWorld\/HelloWorld\/AppDelegate.swift",
  "lineCoverageDelta": {
    "executableLinesDelta": 0,
    "coveredLinesDelta": 3,
    "lineCoverageDelta": 0.5
  },
  "functionDeltas": [
    {
      "name": "AppDelegate.applicationWillTerminate(Notification) -> ()",
      ...
```

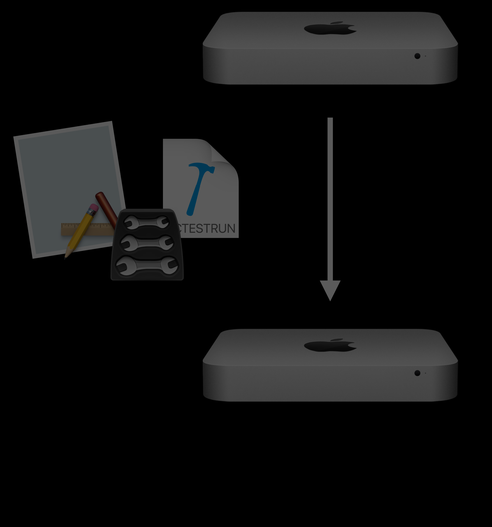
Further Documentation

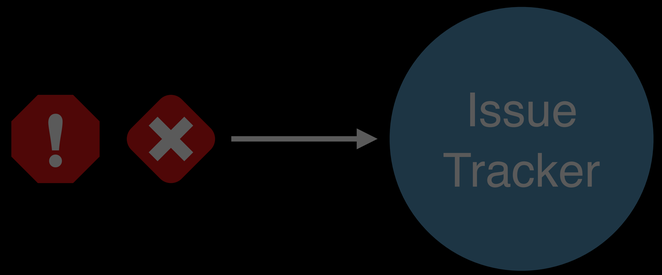
```
$ man xccov
```

```
xccov(1)

NAME
       xccov – view Xcode coverage data in human-readable or machine-parseable format.

SYNOPSIS
       xccov view [--only-targets | --files-for-target target_name | --functions-for-file
                  name_or_path] [--json] report.xccovreport

       ...
```

Build Your Own CI

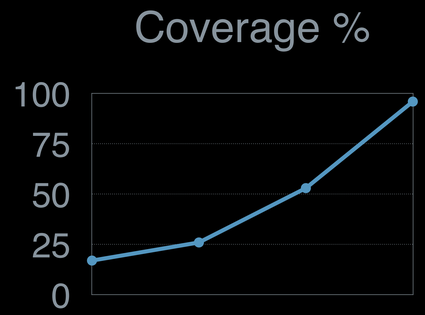
```
xcodebuild build-for-testing
```






```
xcresulttool get
```

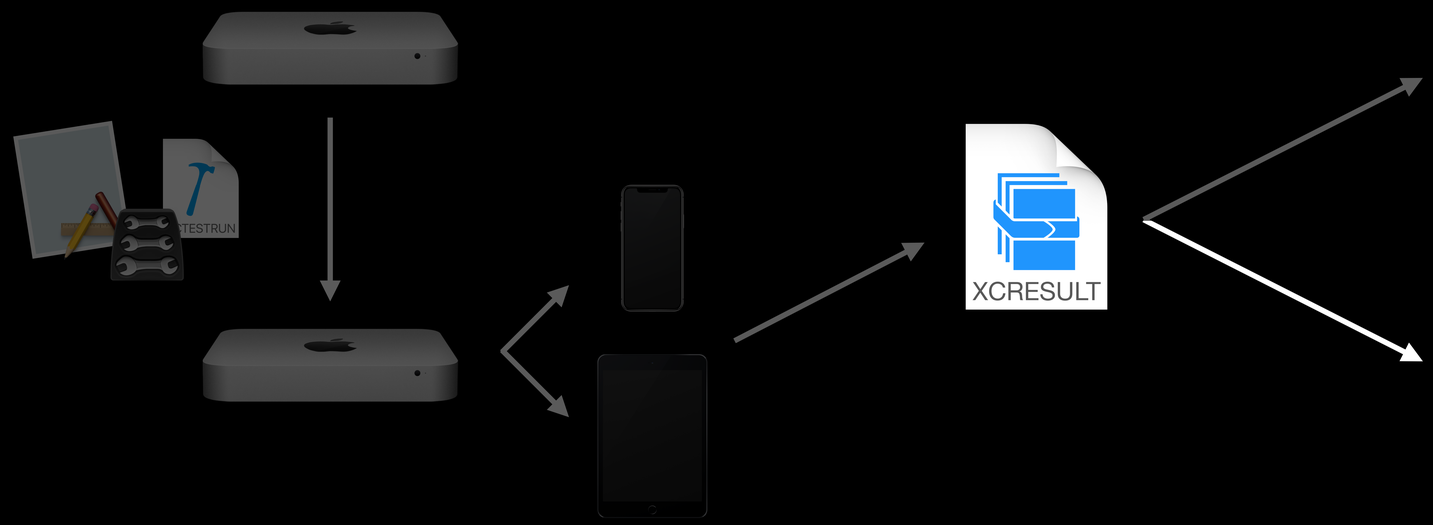
Step 4: Track code coverage



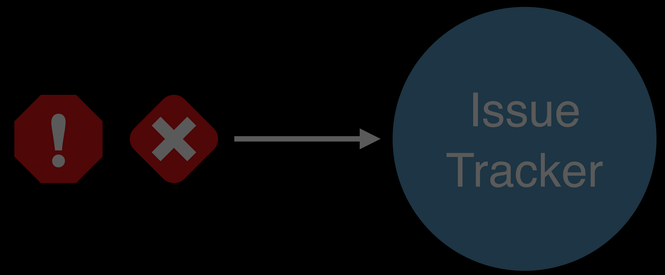

```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```

Build Your Own CI

```
xcodebuild build-for-testing
```



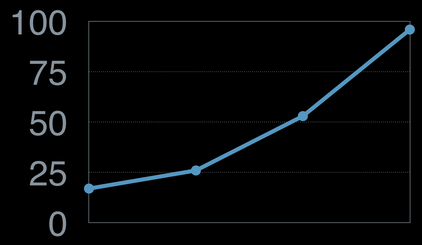

```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```




```
xcresulttool get
```

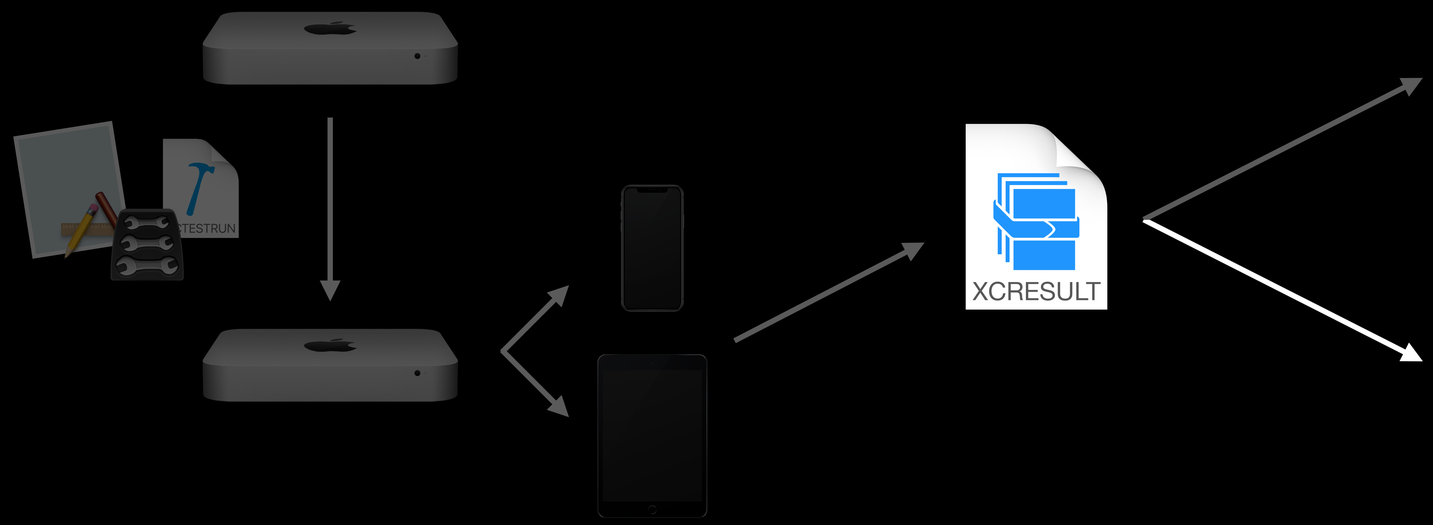
```
xccov view
```

Coverage %

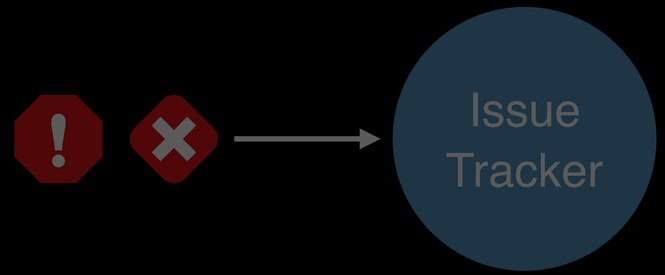


Build Your Own CI

```
xcodebuild build-for-testing
```



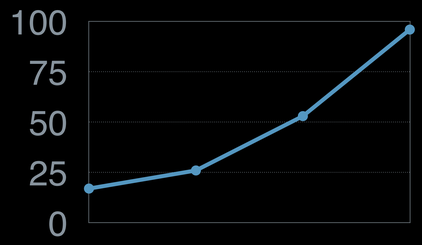

```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```




```
xcresulttool get
```

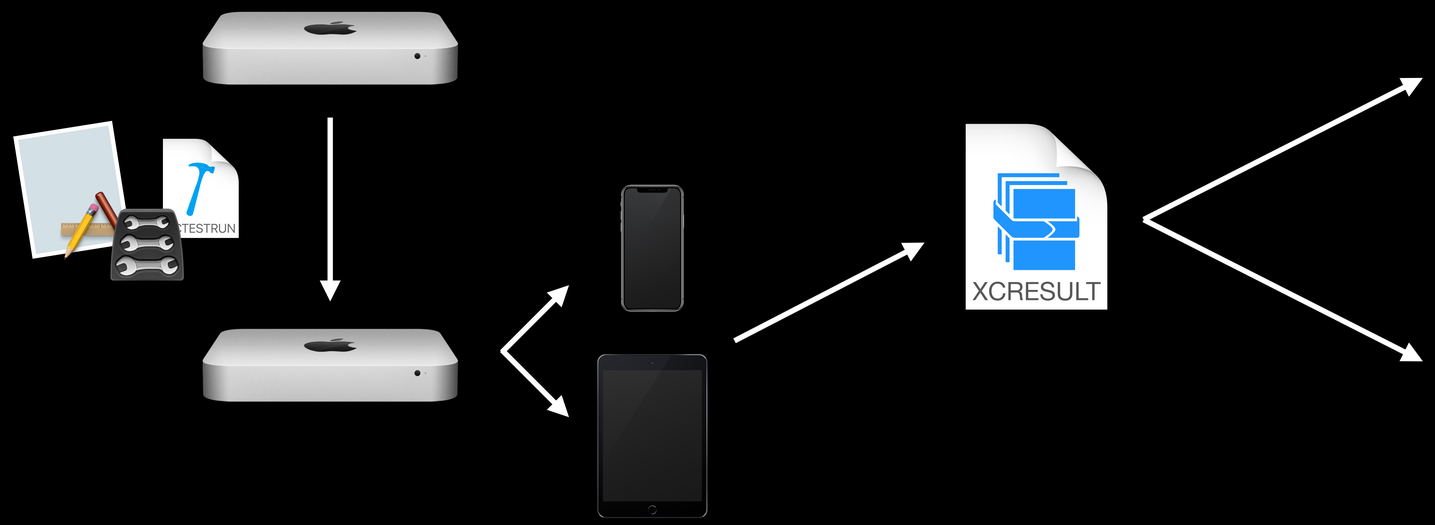
```
xccov view
```

Coverage %

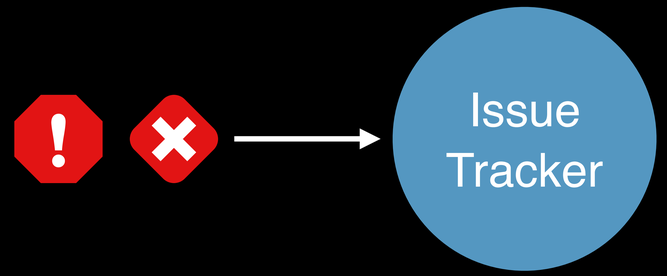


Build Your Own CI

```
xcodebuild build-for-testing
```



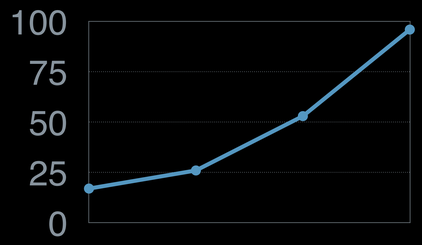

```
xcodebuild test-without-building
  ...
 -resultBundlePath ResultBundle.xcresult
```




```
xcresulttool get
```

```
xccov view
```

Coverage %



Summary

Summary

Write tests in Xcode using XCTest

Summary

Write tests in Xcode using XCTest

Run tests under different configurations with Test Plans

Summary

Write tests in Xcode using XCTest

Run tests under different configurations with Test Plans

Use xcodebuild, xcresulttool, and xccov in CI

More Information

developer.apple.com/wwdc19/413

Improving Battery Life and Performance	Thursday, 4:00
What’s New in Testing	WWDC 2018


WWDC19