What's New in Testing

Session 409

Wil Addario-Turner, Xcode Engineer

What’s new in testing?

Enhancements

Enhancements

Async testing

Enhancements

Async testing

Multi-app testing

Enhancements

Async testing

Multi-app testing

UI testing performance

Enhancements

Async testing

Multi-app testing

UI testing performance

Activities, attachments, and screenshots

Enhancements

UI Testing in Xcode 8.3

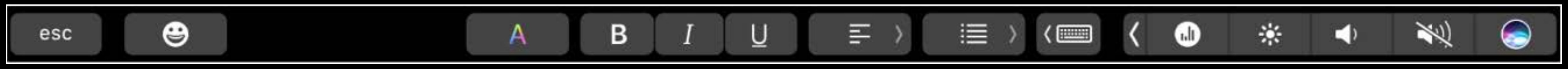
UI Testing in Xcode 8.3

`XCUISiriService`

UI Testing in Xcode 8.3

`XCUISiriService`

`XCUIElement.Type.touchBar`



XCTest in Xcode 9

XCTest in Xcode 9

Swift 4

XCTest in Xcode 9

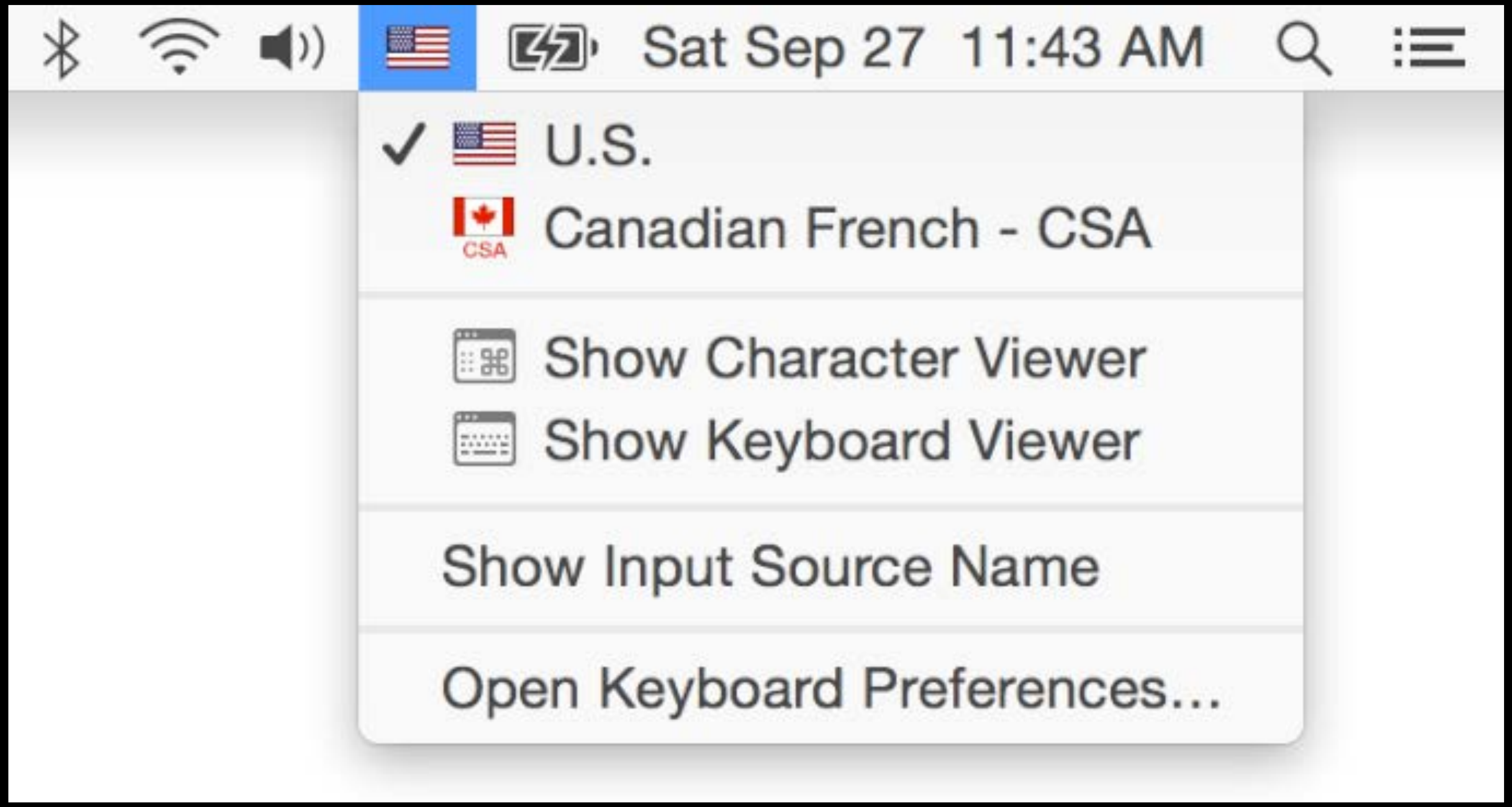
Swift 4

Block-based test teardown

UI Testing

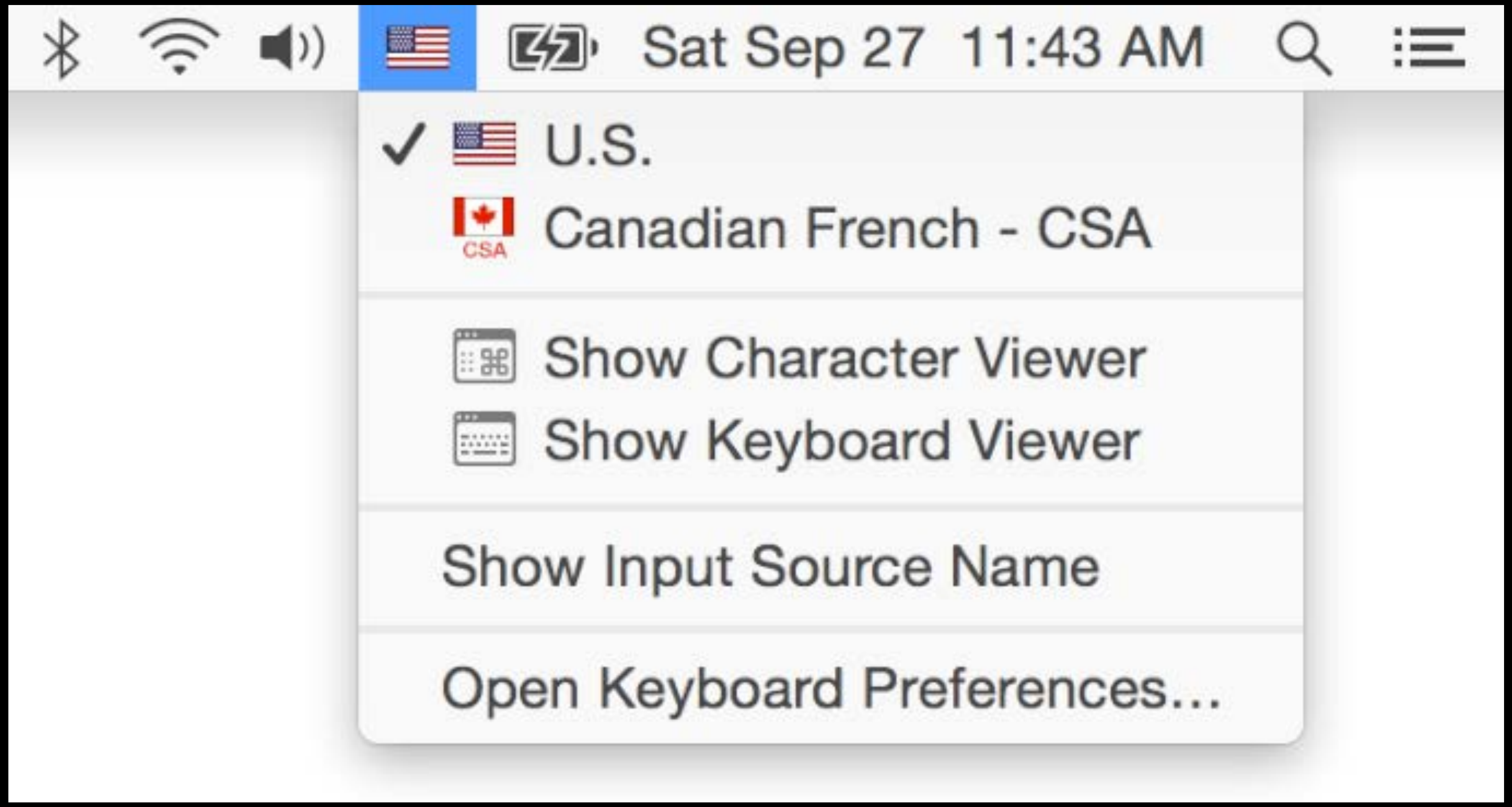
UI Testing

```
XCUIElement.Type.statusItem
```



UI Testing

```
XCUIElement.Type.statusItem
XCUIElement.waitForExistence()
```



xcodebuild

xcodebuild

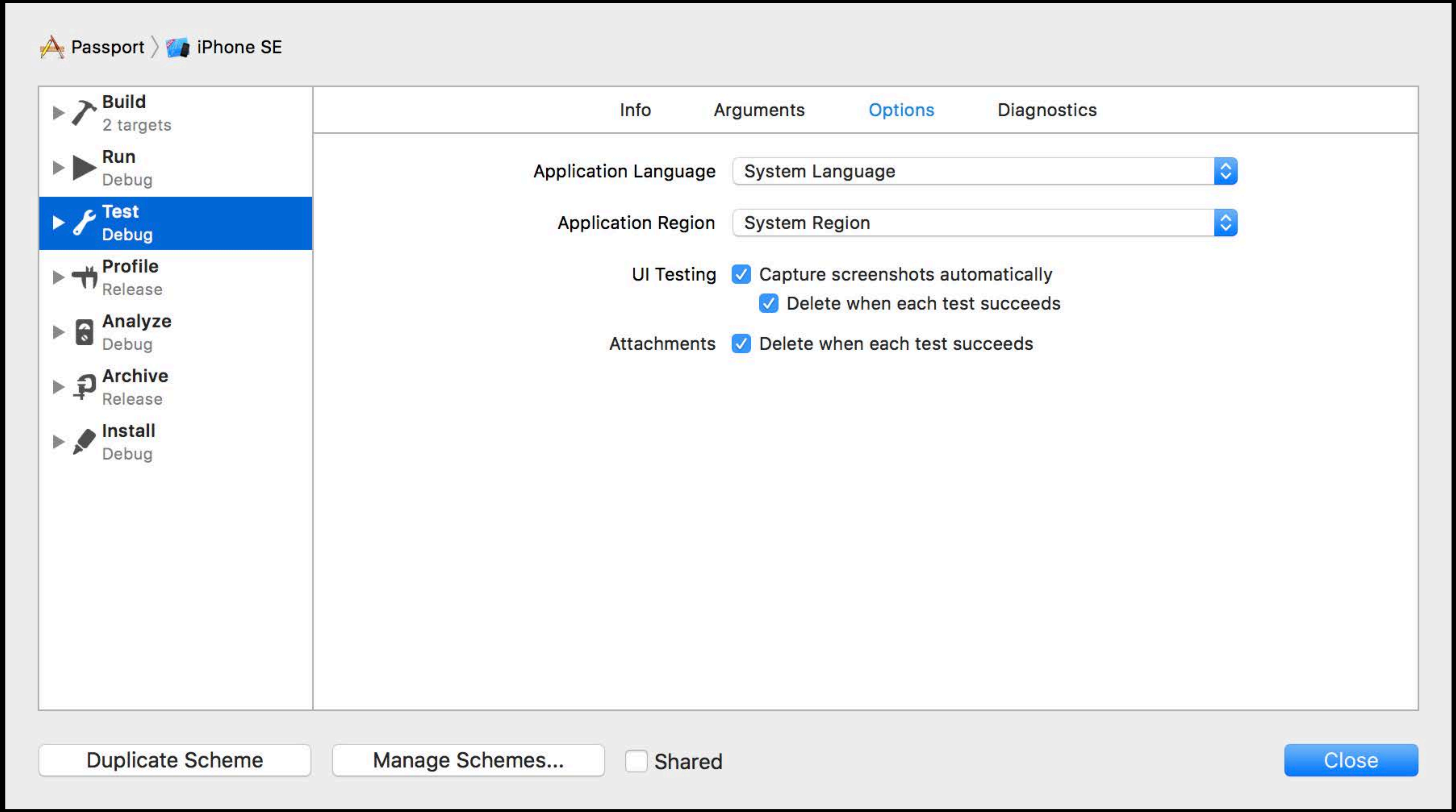
CoreSimulator

xcodebuild

CoreSimulator

Parallel testing

Localization



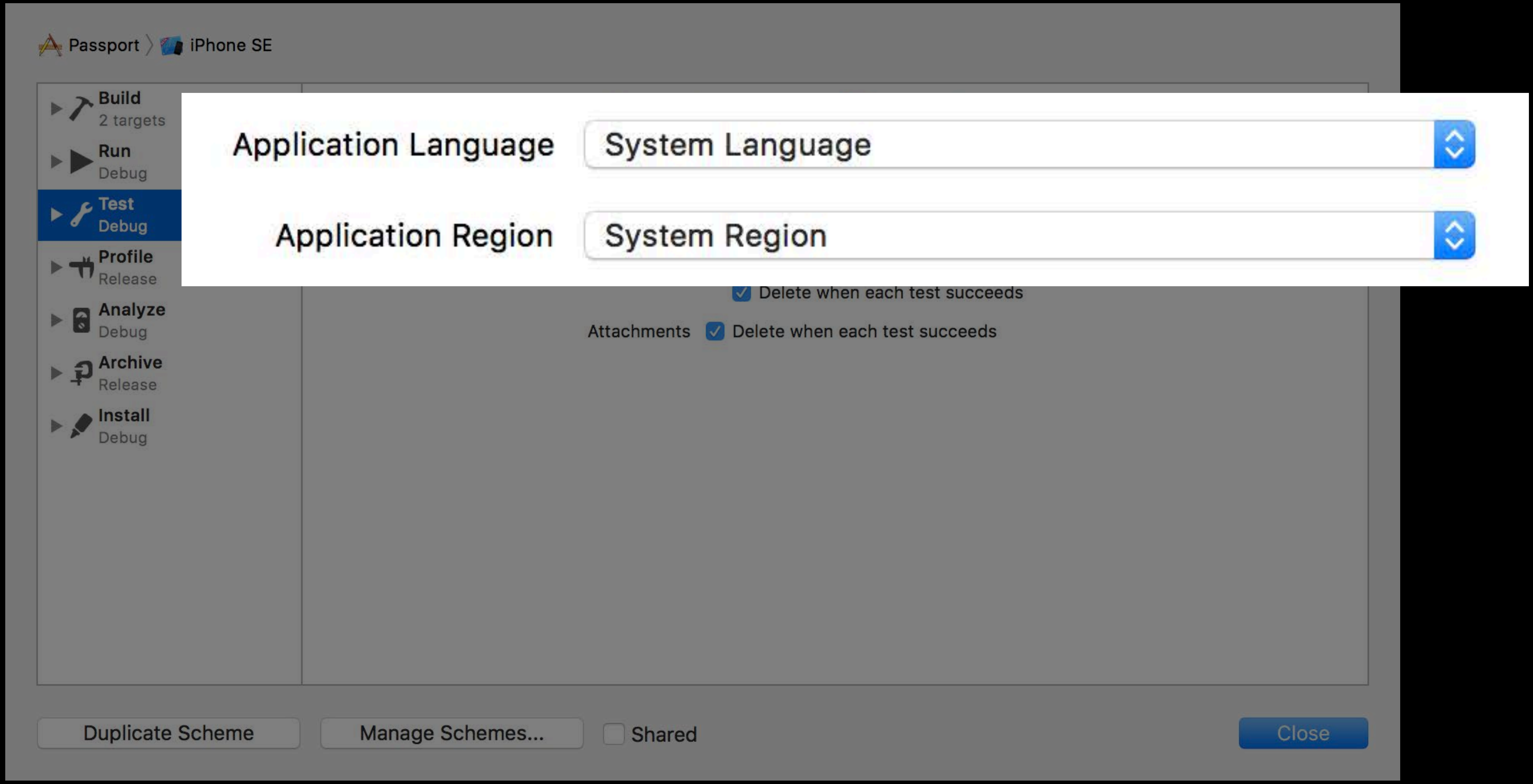
Application Language: System Language

Application Region: System Region

UI Testing: ✓ Capture screenshots automatically
✓ Delete when each test succeeds

Attachments: ✓ Delete when each test succeeds

Localization



Passport 〉 iPhone SE

Application Language: System Language

Application Region: System Region

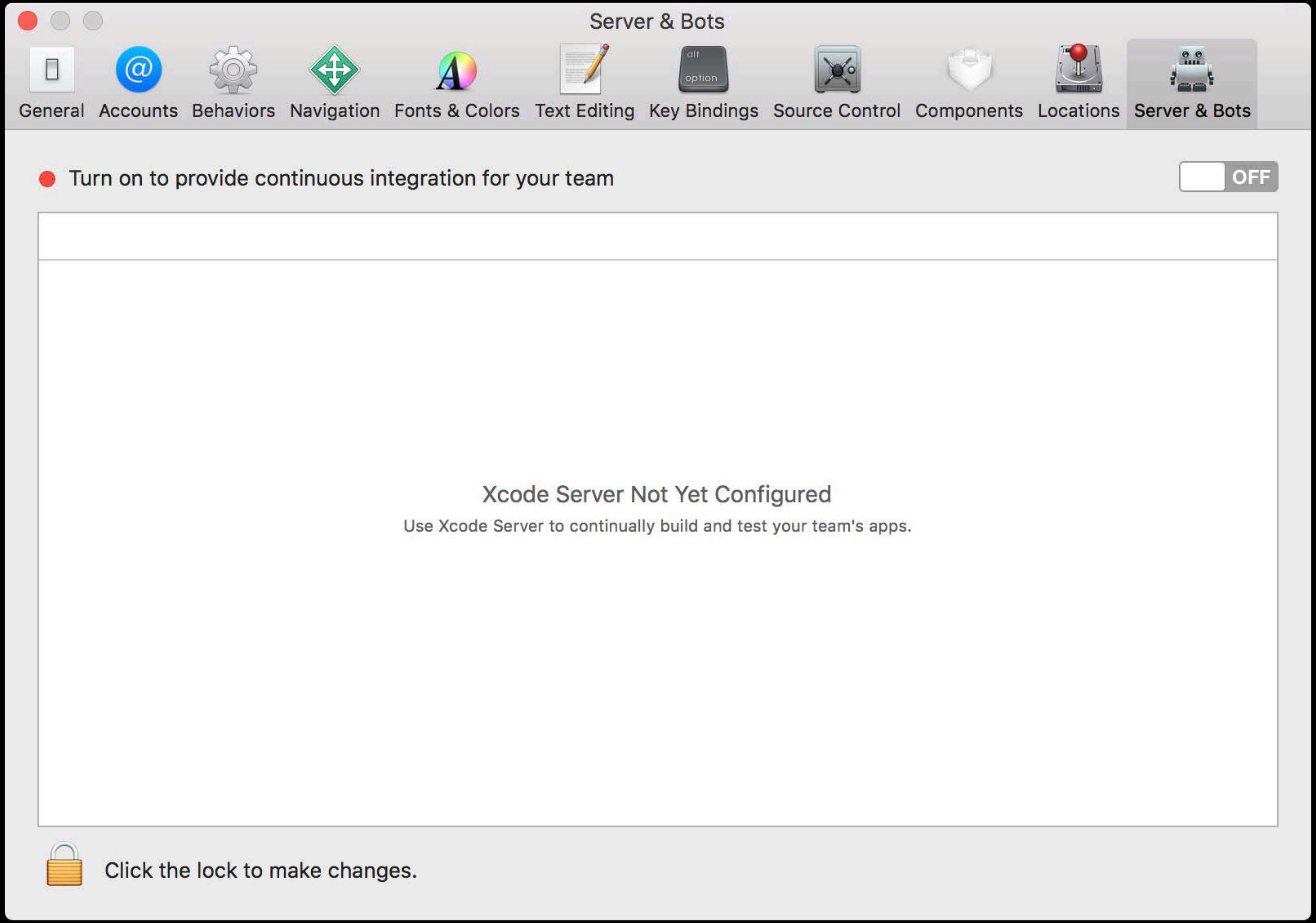
✓ Delete when each test succeeds

Attachments ✓ Delete when each test succeeds

Xcode Server

Xcode Server

Xcode Server



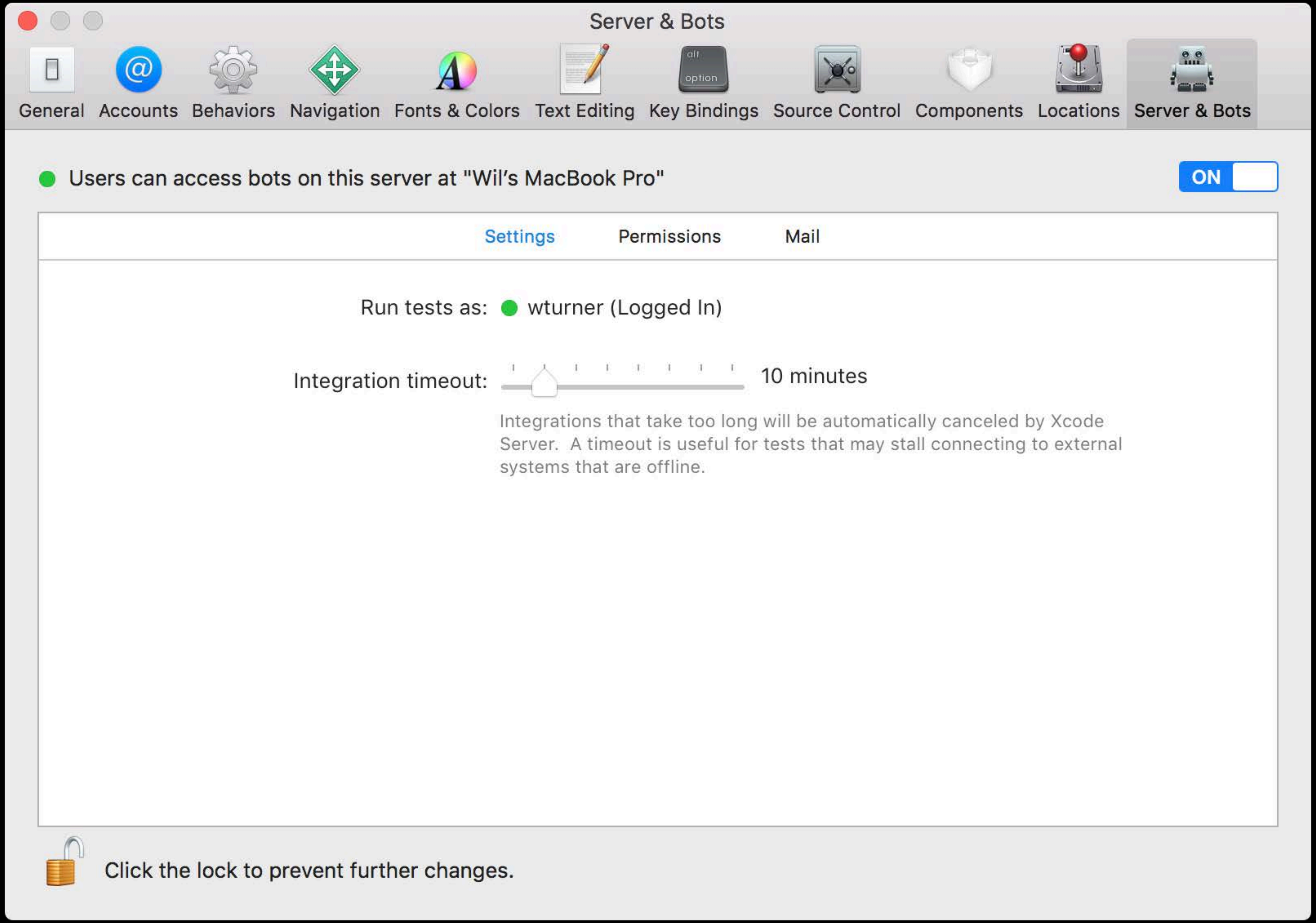
Turn on to provide continuous integration for your team

Xcode Server Not Yet Configured

Use Xcode Server to continually build and test your team's apps.

Click the lock to make changes.

Xcode Server



● Users can access bots on this server at "Wil's MacBook Pro"

Run tests as: ● wturner (Logged In)

Integration timeout: 10 minutes

Integrations that take too long will be automatically canceled by Xcode Server. A timeout is useful for tests that may stall connecting to external systems that are offline.

Click the lock to prevent further changes.

Xcode Server

Xcode Server

Improved provisioning

Xcode Server

Improved provisioning

CoreSimulator

Xcode Server

Improved provisioning

CoreSimulator

Parallel testing

Xcode Server

Improved provisioning

CoreSimulator

Parallel testing

Localization control

Enhancements

Async testing

Multi-app testing

UI testing performance

Activities, attachments, and screenshots

Async Testing

Async Testing

Opening documents

Async Testing

Opening documents

Work on background threads

Async Testing

Opening documents

Work on background threads

Communicating with services and extensions

Async Testing

Opening documents

Work on background threads

Communicating with services and extensions

Network activity

Async Testing

Opening documents

Work on background threads

Communicating with services and extensions

Network activity

Animations

Async Testing

Opening documents

Work on background threads

Communicating with services and extensions

Network activity

Animations

UI test conditions

XCTestCase APIs

XCTestCase APIs

Introduced in Xcode 6

XCTestCase APIs

Introduced in Xcode 6

Create expectations

XCTestCase APIs

Introduced in Xcode 6

Create expectations

Wait for them to be “fulfilled”

```swift
let document = UIDocument(fileURL: documentURL)

let documentExpectation = expectation(description: "Document opened")

document.open() { success in
    XCTAssert(success, "Failed to open file")
    documentExpectation.fulfill()
}

waitForExpectations(timeout: 10)
```

```swift
let document = UIDocument(fileURL: documentURL)

let documentExpectation = expectation(description: "Document opened")

document.open() { success in
    XCTAssert(success, "Failed to open file")
    documentExpectation.fulfill()
}

waitForExpectations(timeout: 10)
```

```swift
let document = UIDocument(fileURL: documentURL)

let documentExpectation = expectation(description: "Document opened")

document.open() { success in
    XCTAssert(success, "Failed to open file")
    documentExpectation.fulfill()
}

waitForExpectations(timeout: 10)
```

```swift
let document = UIDocument(fileURL: documentURL)

let documentExpectation = expectation(description: "Document opened")

document.open() { success in
    XCTAssert(success, "Failed to open file")
    documentExpectation.fulfill()
}

waitForExpectations(timeout: 10)
```

Limitations

Limitations

Timeout is a test failure

Limitations

Timeout is a test failure

Waiting requires test object

Limitations

Timeout is a test failure

Waiting requires test object

Hard to factor out

Limitations

Timeout is a test failure

Waiting requires test object

Hard to factor out

No nested waiting

XCTWaiter

NEW

XCTWaiter

NEW

Extracted logic from XCTestCase

XCTWaiter

NEW

Extracted logic from XCTestCase

Explicit list of expectations

XCTWaiter

NEW

Extracted logic from XCTestCase

Explicit list of expectations

Calls back to `XCTWaiterDelegate`

XCTWaiter

NEW

Extracted logic from XCTestCase

Explicit list of expectations

Calls back to `XCTWaiterDelegate`

Returns `XCTWaiter.Result`

```swift
let document = UIDocument(fileURL: documentURL)

let documentExpectation = expectation(description: "Document opened")

document.open() { success in
    XCTAssert(success, "Failed to open file")
    documentExpectation.fulfill()
}

// Test case waits implicitly
waitForExpectations(timeout: 10)
```

```
// Test case waits implicitly
waitForExpectations(timeout: 10)
```

```swift
// Test case waits implicitly
waitForExpectations(timeout: 10)
```

```swift
// Test case waits explicitly
wait(for: [documentExpectation], timeout: 10)
```

```
// Test case waits implicitly
waitForExpectations(timeout: 10)

// Test case waits explicitly
wait(for: [documentExpectation], timeout: 10)

// Waiter instance delegates to test
XCTWaiter(delegate: self).wait(for: [documentExpectation], timeout: 10)
```

```swift
// Test case waits implicitly
waitForExpectations(timeout: 10)

// Test case waits explicitly
wait(for: [documentExpectation], timeout: 10)

// Waiter instance delegates to test
XCTWaiter(delegate: self).wait(for: [documentExpectation], timeout: 10)

// Waiter class returns result
let result = XCTWaiter.wait(for: [documentExpectation], timeout: 10)
if result == .timedOut {
    // handling the timeout…
}
```

XCTestExpectation

NEW

XCTestExpectation

NEW

Public initializer

XCTestExpectation

NEW

Public initializer

- Decoupled from XCTestCase

XCTestExpectation

NEW

Public initializer

- Decoupled from XCTestCase

Multiple fulfillments

XCTestExpectation

NEW

Public initializer

- Decoupled from XCTestCase

Multiple fulfillments

Inverted behavior

XCTestExpectation

NEW

Public initializer

- Decoupled from XCTestCase

Multiple fulfillments

Inverted behavior

Ordering enforcement

Async Testing

NEW

Async Testing

NEW

XCTWaiter manages expectations

Async Testing

NEW

XCTWaiter manages expectations

XCTestExpectation has new features

Async Testing

NEW

XCTWaiter manages expectations

XCTestExpectation has new features

Both decoupled from XCTestCase

Enhancements

Async testing

Multi-app testing

UI testing performance

Activities, attachments, and screenshots

XCUIApplication

XCUIApplication

XCUIApplication

Launch

XCUIApplication

Launch

Terminate

XCUIApplication

Launch

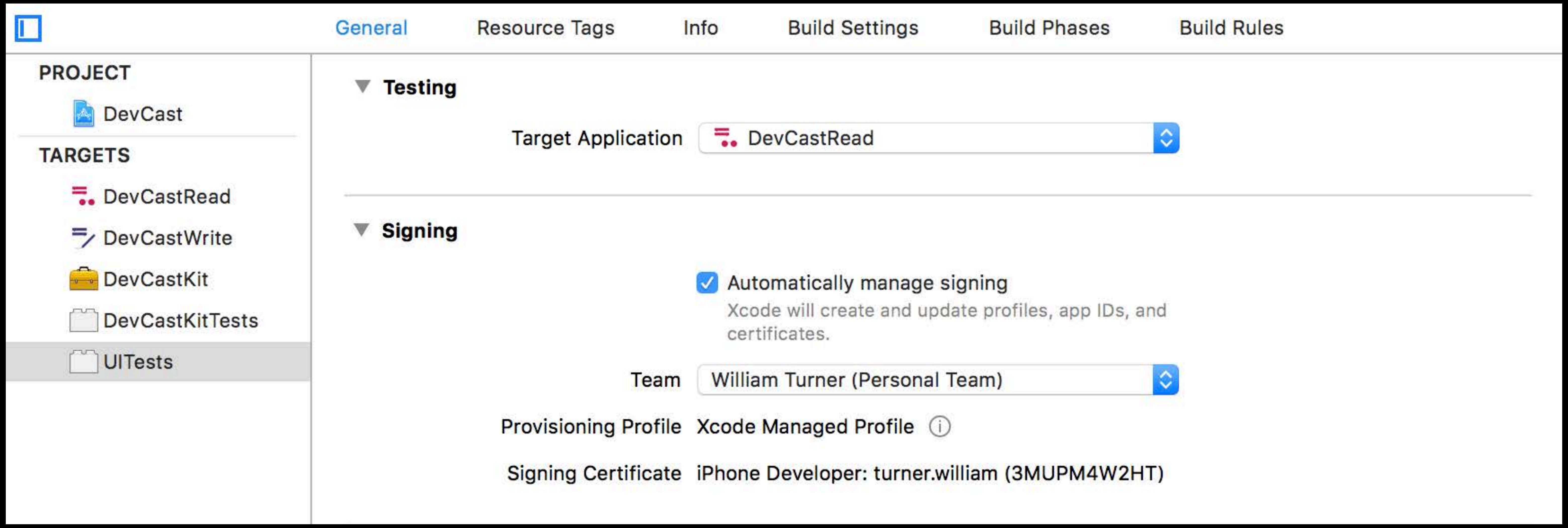
Terminate

Queries

Target Application

Target Application

Project configuration



General | Resource Tags | Info | Build Settings | Build Phases | Build Rules

PROJECT
- DevCast

TARGETS
- DevCastRead
- DevCastWrite
- DevCastKit
- DevCastKitTests
- UITests

Testing

Target Application: DevCastRead

Signing

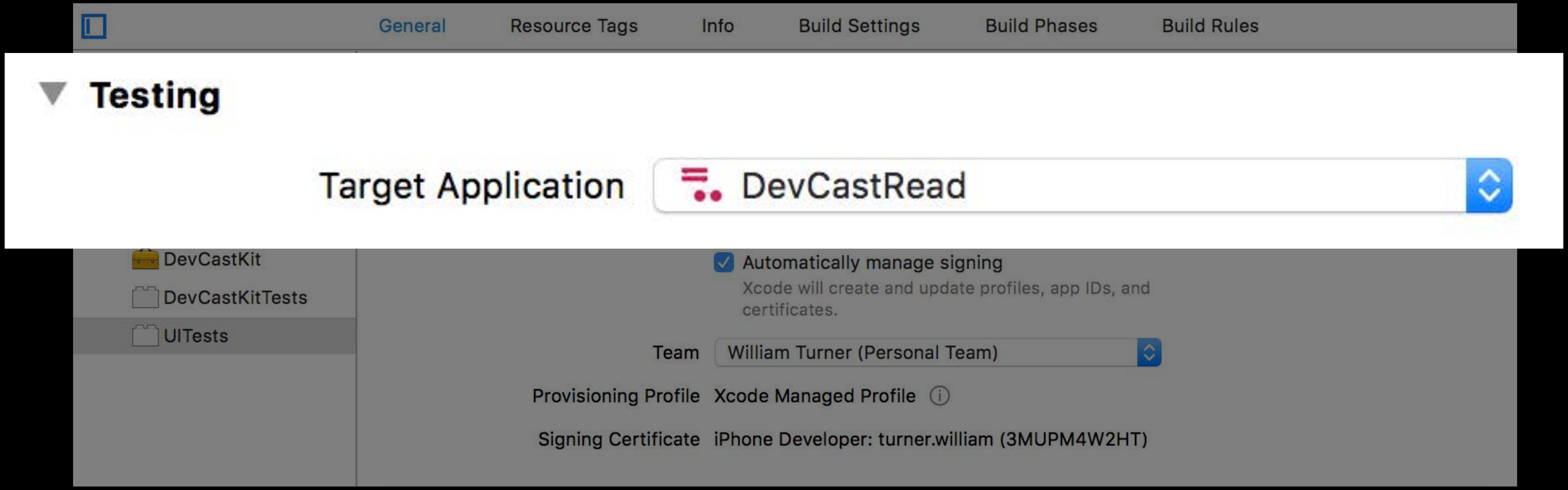
✓ Automatically manage signing

Xcode will create and update profiles, app IDs, and certificates.

Team	William Turner (Personal Team)
Provisioning Profile	Xcode Managed Profile
Signing Certificate	iPhone Developer: turner.william (3MUPM4W2HT)

Target Application

Project configuration



General | Resource Tags | Info | Build Settings | Build Phases | Build Rules

Testing

Target Application: DevCastRead

DevCastKit
DevCastKitTests
UITests

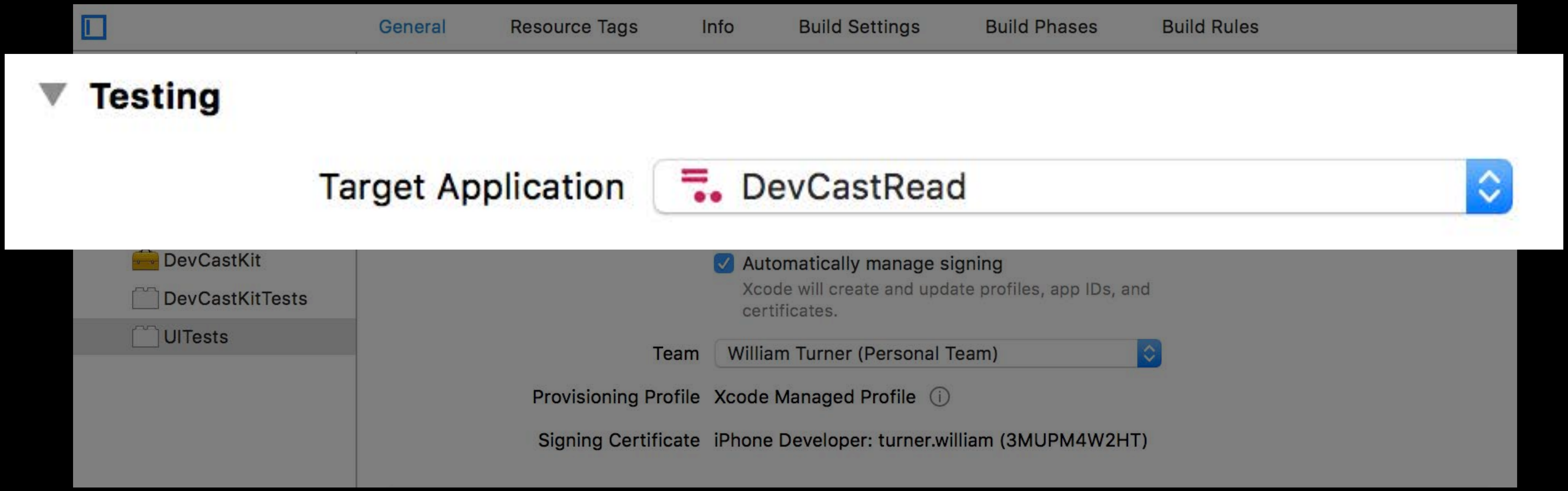
[✓] Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.

Team: William Turner (Personal Team)
Provisioning Profile: Xcode Managed Profile
Signing Certificate: iPhone Developer: turner.william (3MUPM4W2HT)

Target Application

Project configuration

Default initializer



General | Resource Tags | Info | Build Settings | Build Phases | Build Rules

Testing

Target Application: DevCastRead

DevCastKit
DevCastKitTests
UITests

[✓] Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.

Team: William Turner (Personal Team)
Provisioning Profile: Xcode Managed Profile
Signing Certificate: iPhone Developer: turner.william (3MUPM4W2HT)

```swift
let targetApp = XCUIApplication()
```

Multi-app Scenarios

Multi-app Scenarios

App groups

Multi-app Scenarios

App groups

Settings

Multi-app Scenarios

App groups

Settings

Extensions

Additions to XCUIApplication

NEW

Additions to XCUIApplication

NEW

New initializers

```
init(bundleIdentifier: String)
init(url: URL)
```

Additions to XCUIApplication

NEW

New initializers

```
init(bundleIdentifier: String)
init(url: URL)
```

Activate method

```
func activate()
```

Additions to XCUIApplication

NEW

New initializers

```
init(bundleIdentifier: String)
init(url: URL)
```

Activate method

```
func activate()
```

State property

```
var state: XCUIApplication.State { get }
```

```swift
let readerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Reader")
let writerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Writer")

readerApp.launch()
// interact with first app

writerApp.launch()
// interact with second app

readerApp.activate()
// return to first app without relaunching
```

```swift
let readerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Reader")
let writerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Writer")

readerApp.launch()
// interact with first app

writerApp.launch()
// interact with second app

readerApp.activate()
// return to first app without relaunching
```

```swift
let readerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Reader")
let writerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Writer")

readerApp.launch()
// interact with first app

writerApp.launch()
// interact with second app

readerApp.activate()
// return to first app without relaunching
```

```swift
let readerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Reader")
let writerApp = XCUIApplication(bundleIdentifier: "com.mycompany.Writer")

readerApp.launch()
// interact with first app

writerApp.launch()
// interact with second app

readerApp.activate()
// return to first app without relaunching
```

Demo

Multi-app UI testing

Warren Ma, Xcode Engineer

Enhancements

Async testing

Multi-app testing

UI testing performance

Activities, attachments, and screenshots

User Interface Elements

User Interface Elements

Buttons, labels, etc.

User Interface Elements

Buttons, labels, etc.

Queries are used to find elements

User Interface Elements

Buttons, labels, etc.

Queries are used to find elements

```
let button = app.navigationBars.buttons["Done"]
```

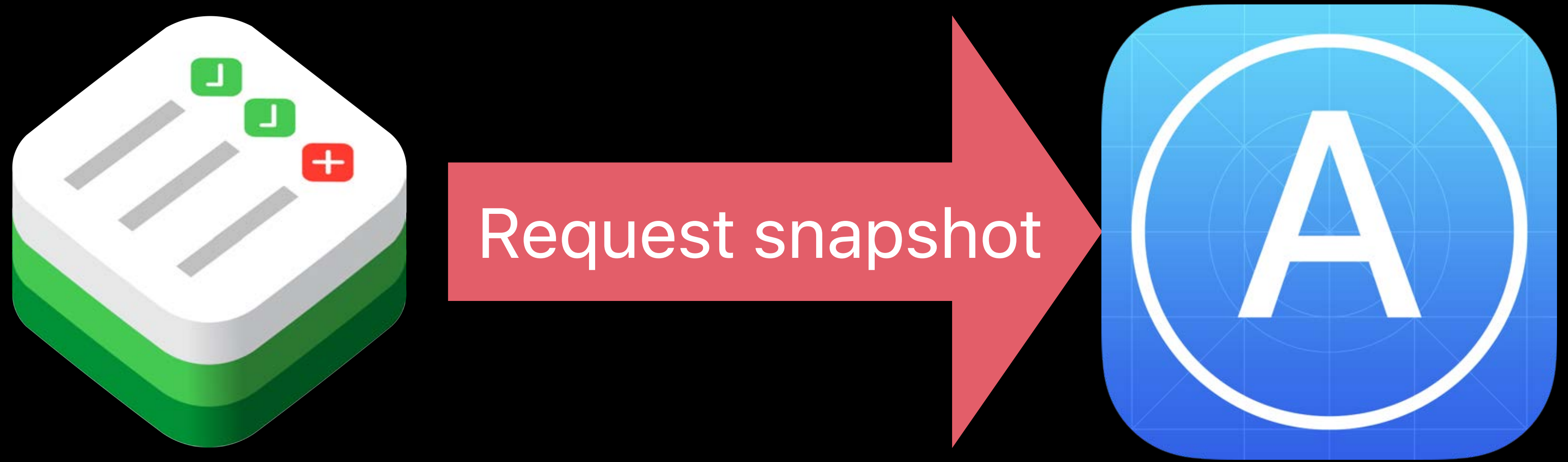
Queries Use Accessibility Data

Queries Use Accessibility Data

Test process fetches atomic “snapshot”

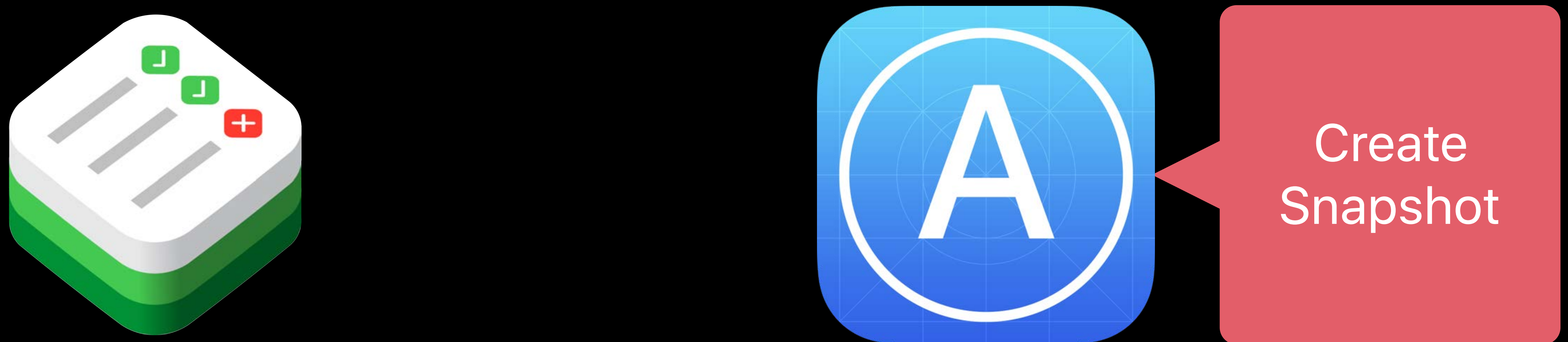
Queries Use Accessibility Data

Test process fetches atomic “snapshot”



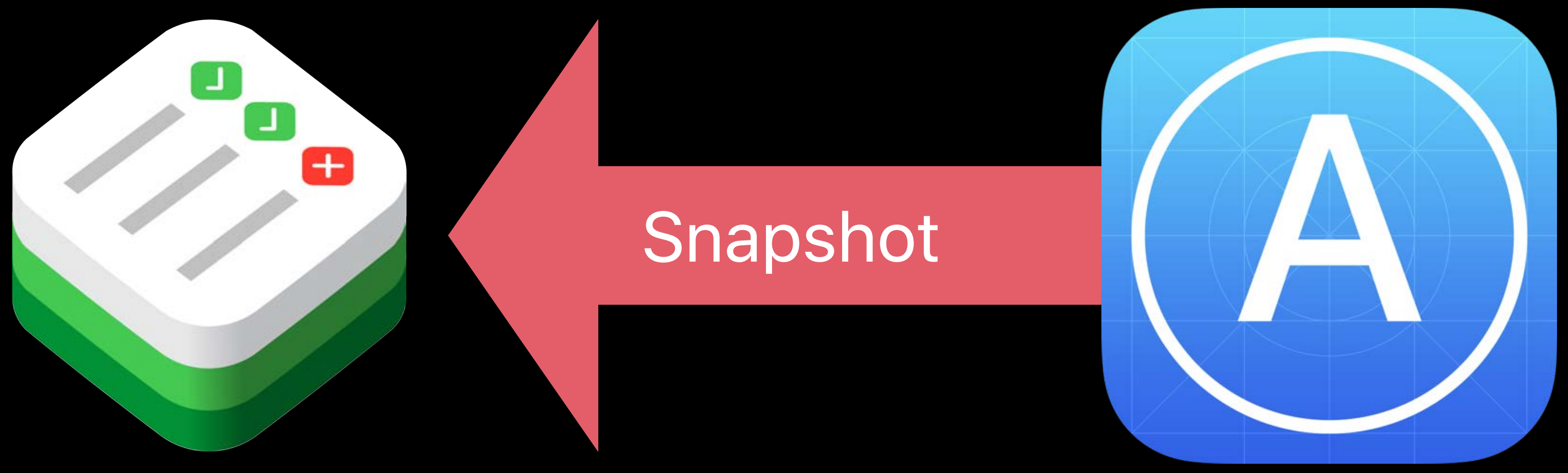
Queries Use Accessibility Data

Test process fetches atomic “snapshot”



Queries Use Accessibility Data

Test process fetches atomic “snapshot”



Queries Use Accessibility Data

Test process fetches atomic “snapshot”

Finds all matching elements

Evaluate Query

Performance Challenges

Performance Challenges

Time and memory

Performance Challenges

Time and memory

Timeouts

Performance Challenges

Time and memory

Timeouts

Low memory reports

How can we improve snapshot performance?

Optimization 1: Remote Queries

Reduce serialization and transport overhead

Optimization 1: Remote Queries

Reduce serialization and transport overhead

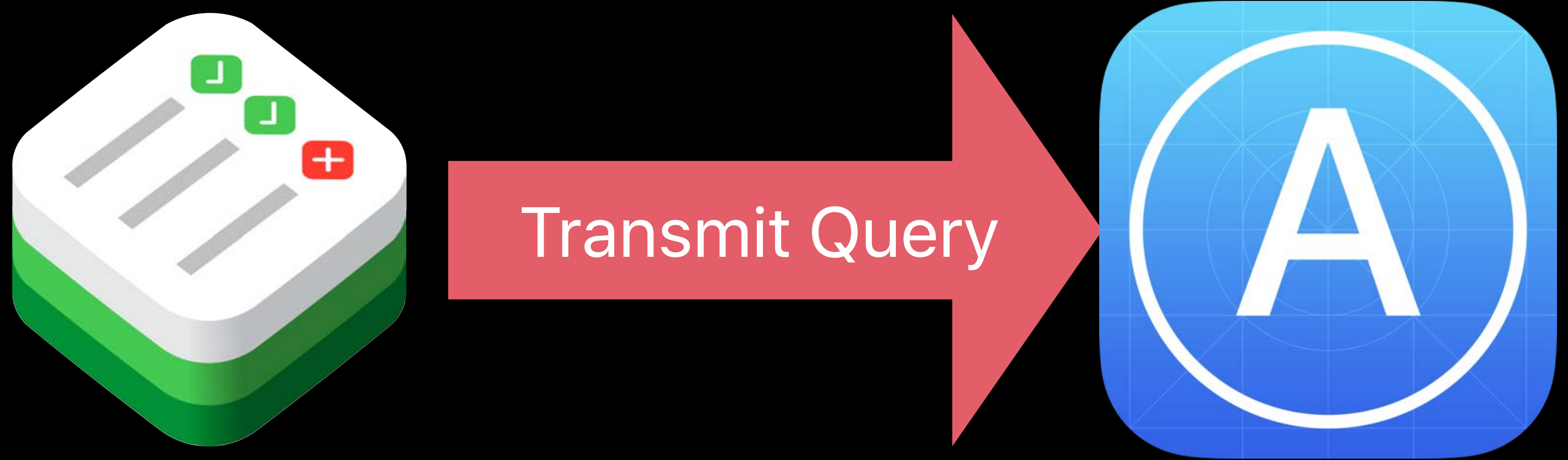
Don’t fetch the snapshot

Optimization 1: Remote Queries

Reduce serialization and transport overhead

Don’t fetch the snapshot

Transmit the query



Optimization 1: Remote Queries

Reduce serialization and transport overhead

Don’t fetch the snapshot

Transmit the query

Create
Snapshot

Optimization 1: Remote Queries

Reduce serialization and transport overhead

Don’t fetch the snapshot

Transmit the query

Evaluate remotely

Evaluate Query

Optimization 1: Remote Queries

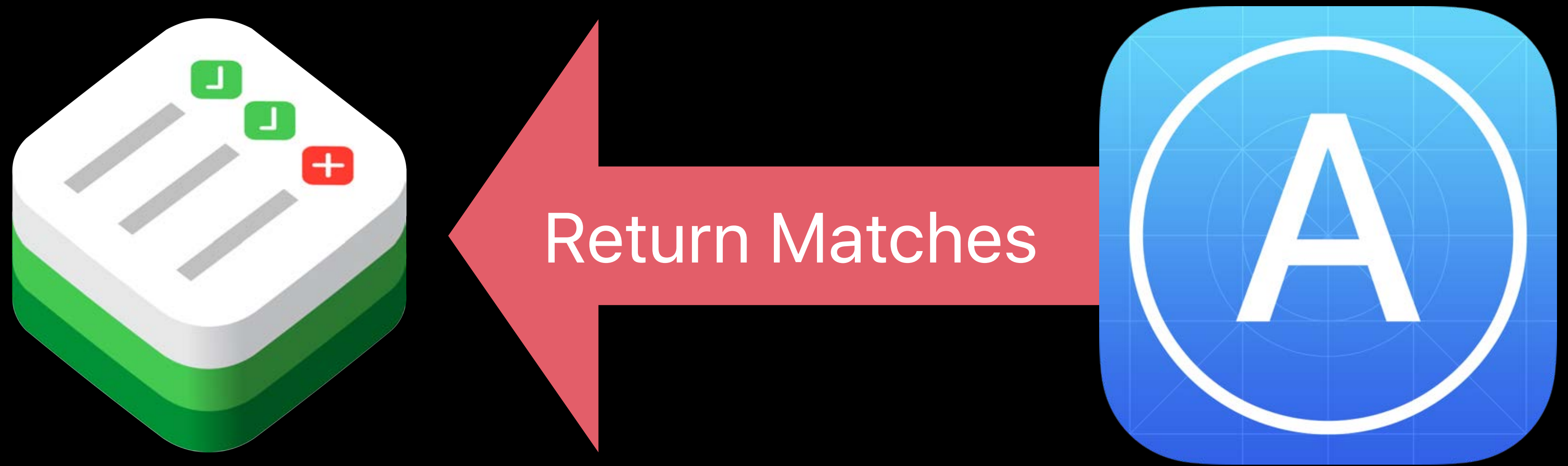
Reduce serialization and transport overhead

Don't fetch the snapshot

Transmit the query

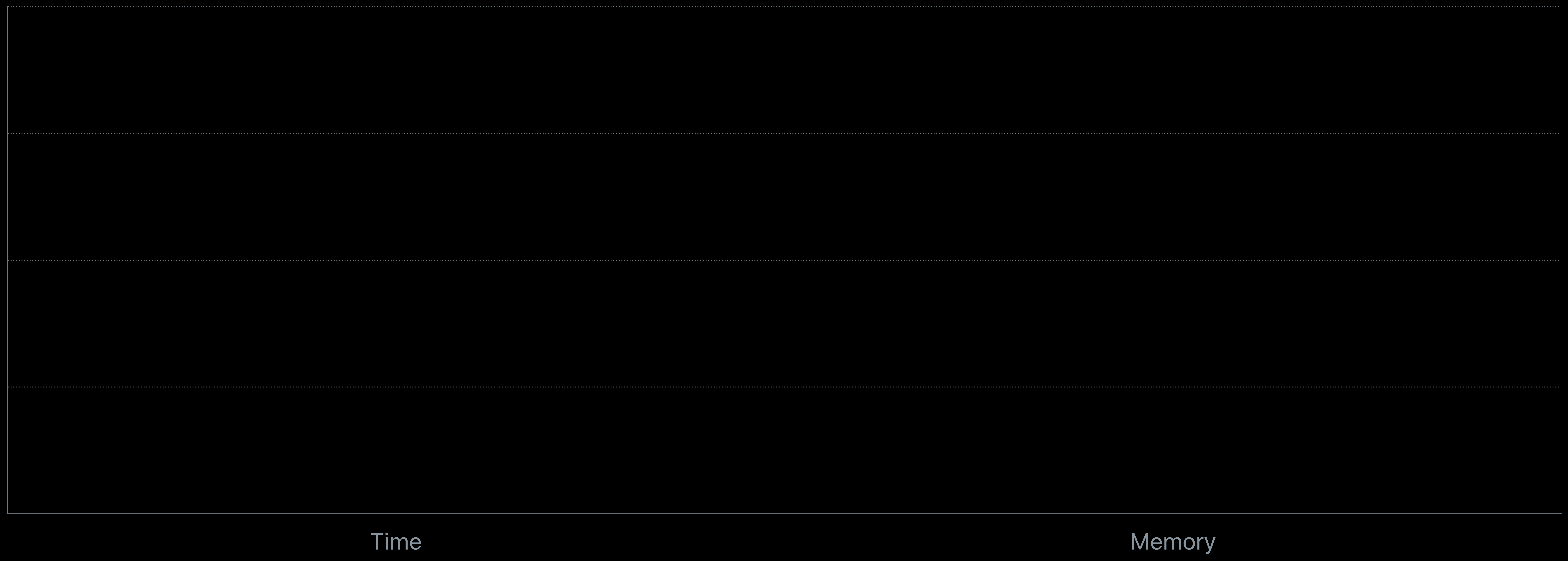
Evaluate remotely

Return results



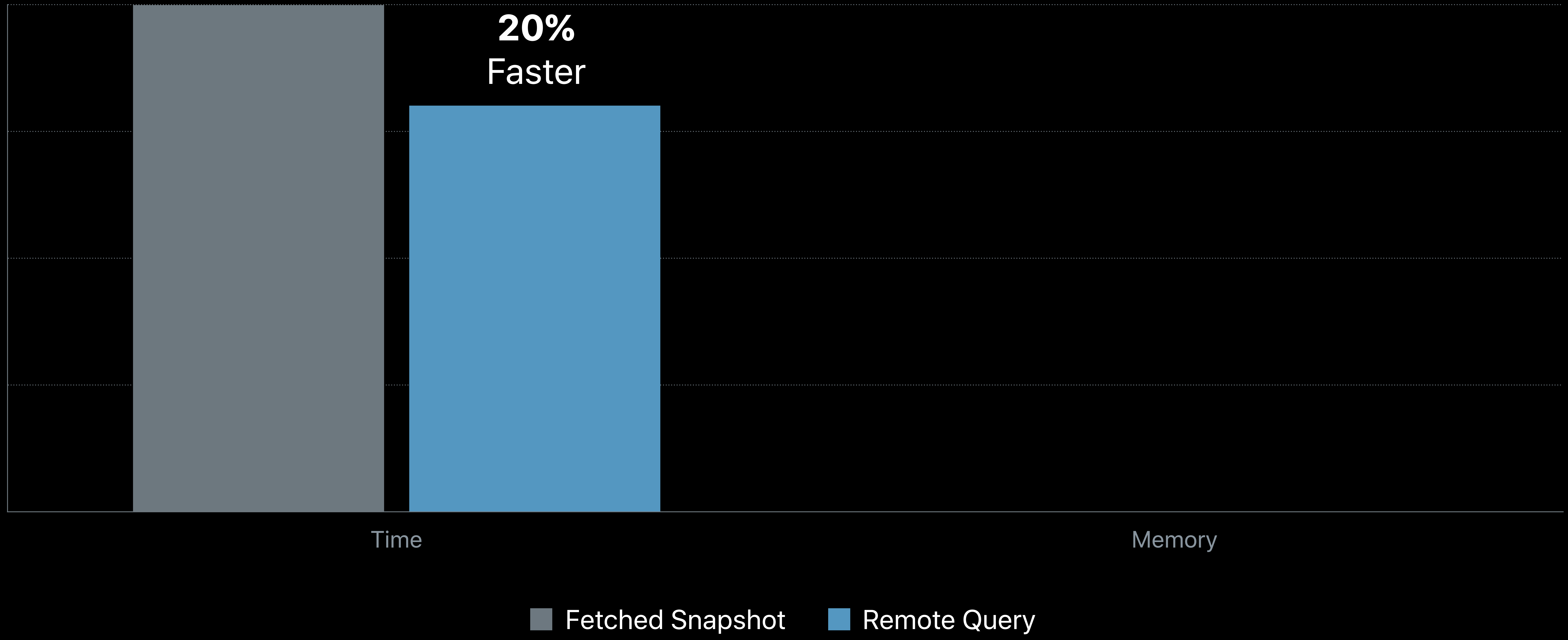
Remote Query Performance

Remote Query Performance

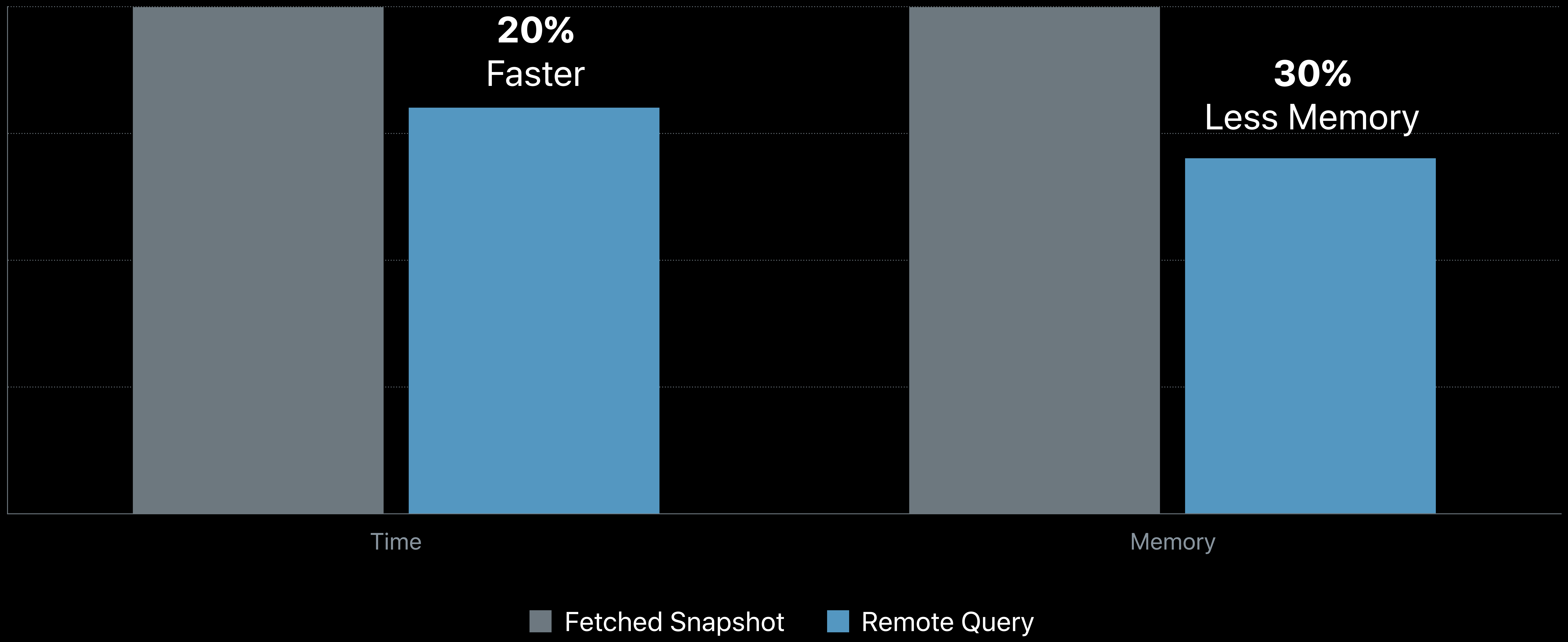


⬜ Fetched Snapshot 🟦 Remote Query

Remote Query Performance



Remote Query Performance



Optimization 2: Query Analysis

Reduce snapshot size

Optimization 2: Query Analysis

Reduce snapshot size

Minimal set of attributes

Optimization 2: Query Analysis

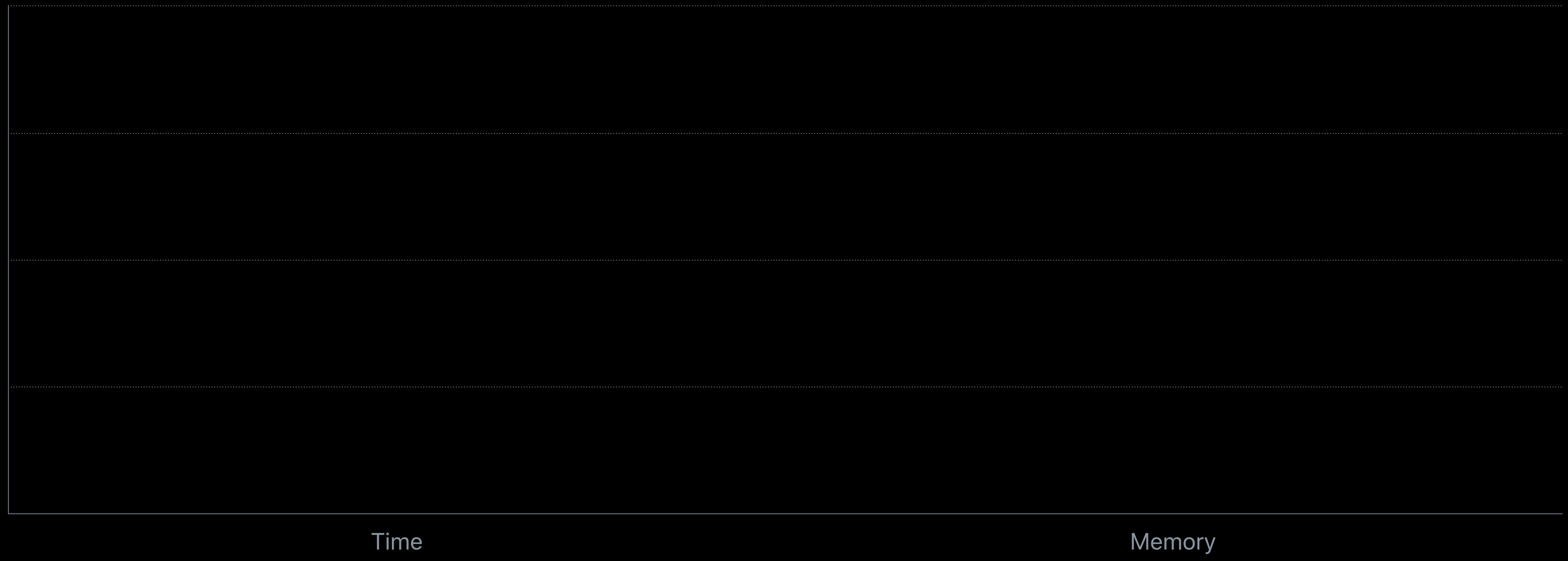
Reduce snapshot size

Minimal set of attributes

Fetch others on demand

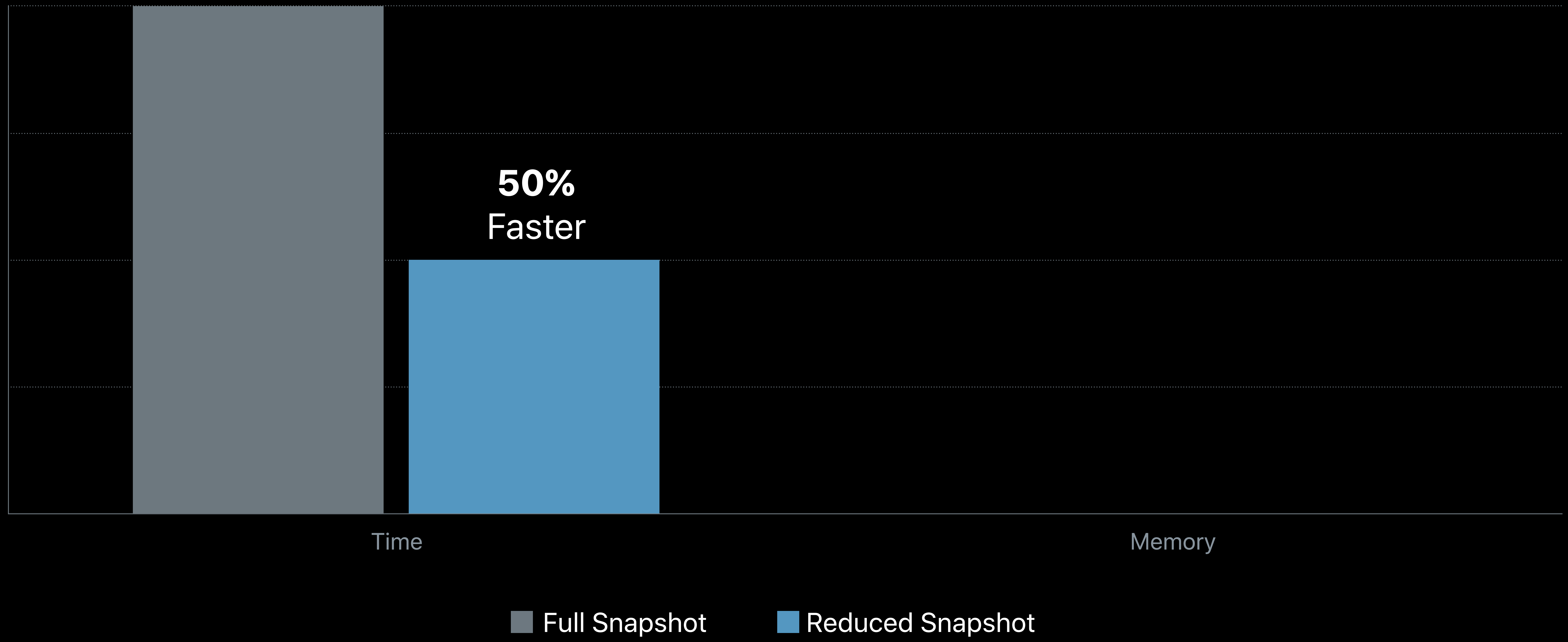
Query Analysis Performance

Query Analysis Performance

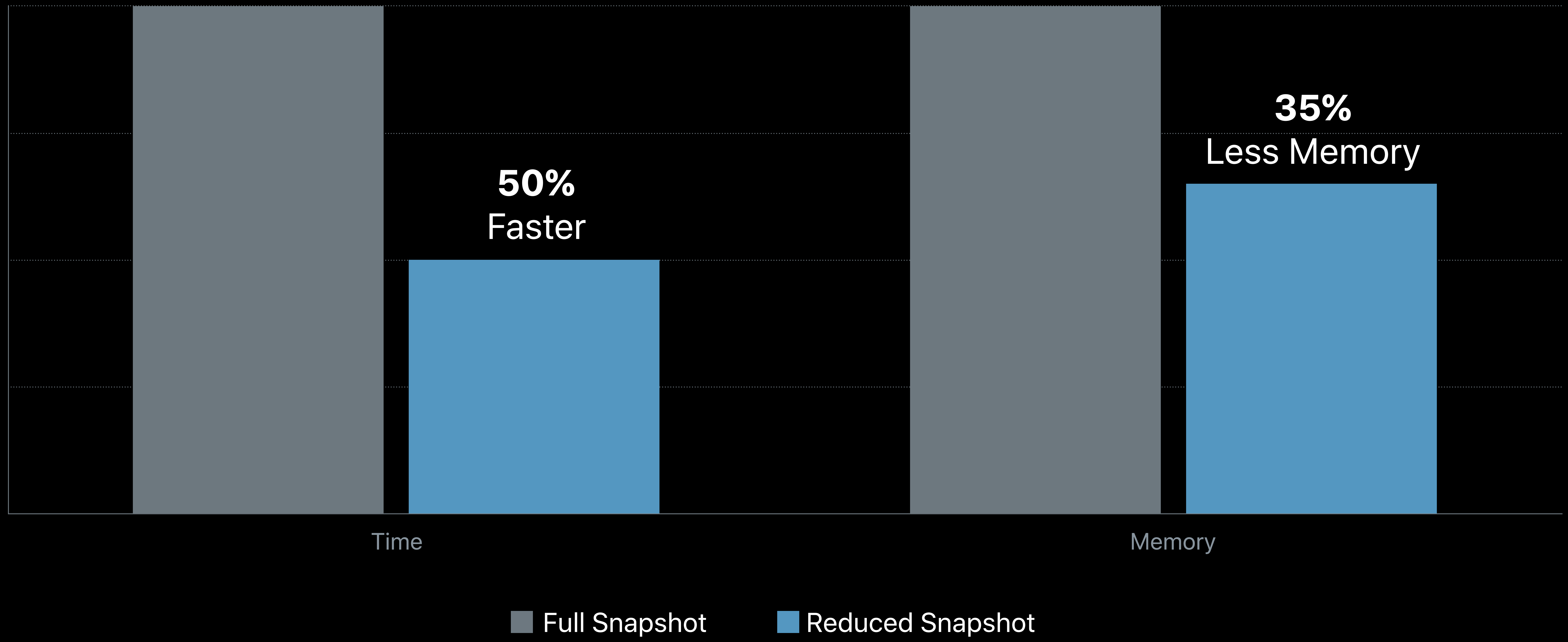


⬛ Full Snapshot   🟦 Reduced Snapshot

Query Analysis Performance



Query Analysis Performance



Optimization 3: Eliminate Snapshots

"First match" API

NEW

Optimization 3: Eliminate Snapshots

“First match” API

NEW

Queries search entire tree

Optimization 3: Eliminate Snapshots

“First match” API

NEW

Queries search entire tree

First match halts early

```
var firstMatch: XCUIElement { get }
```

Optimization 3: Eliminate Snapshots

“First match” API

NEW

Queries search entire tree

First match halts early

```
var firstMatch: XCUIElement { get }
```

Optimization 3: Eliminate Snapshots

“First match” API

NEW

Queries search entire tree

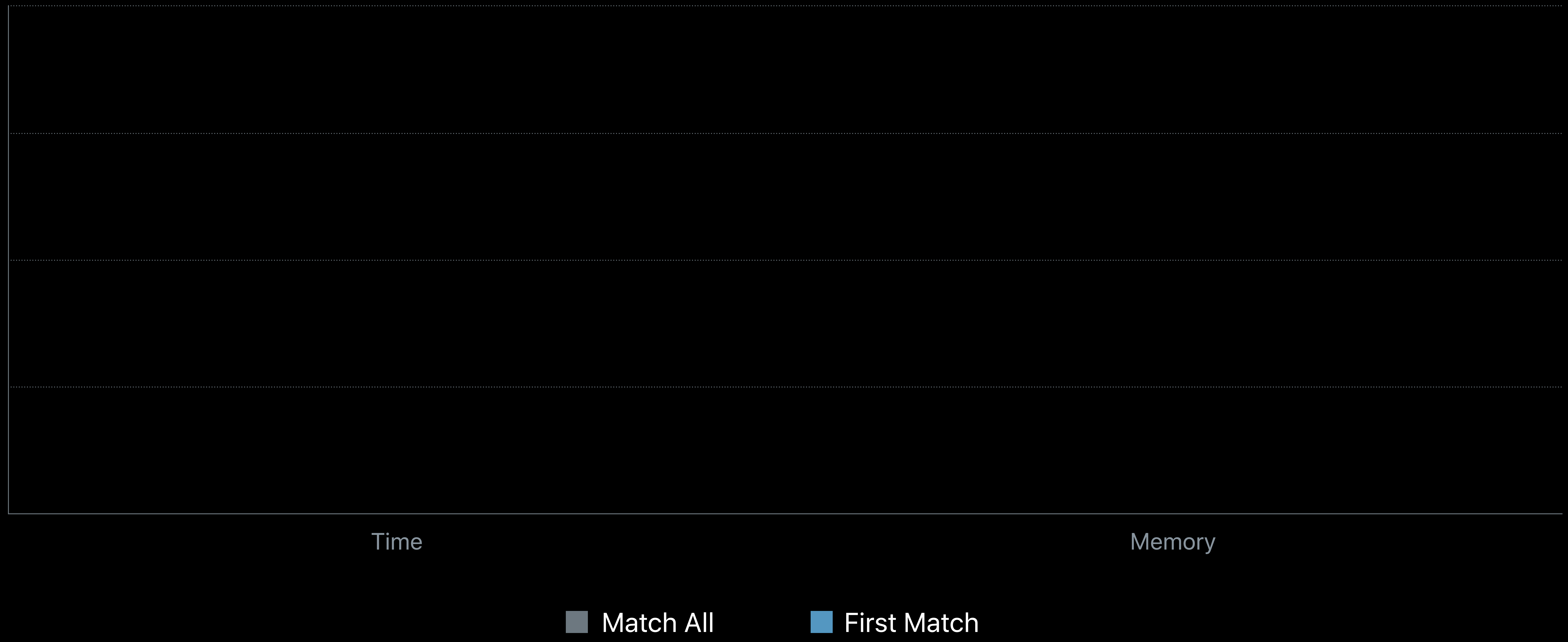
First match halts early

```
var firstMatch: XCUIElement { get }
```

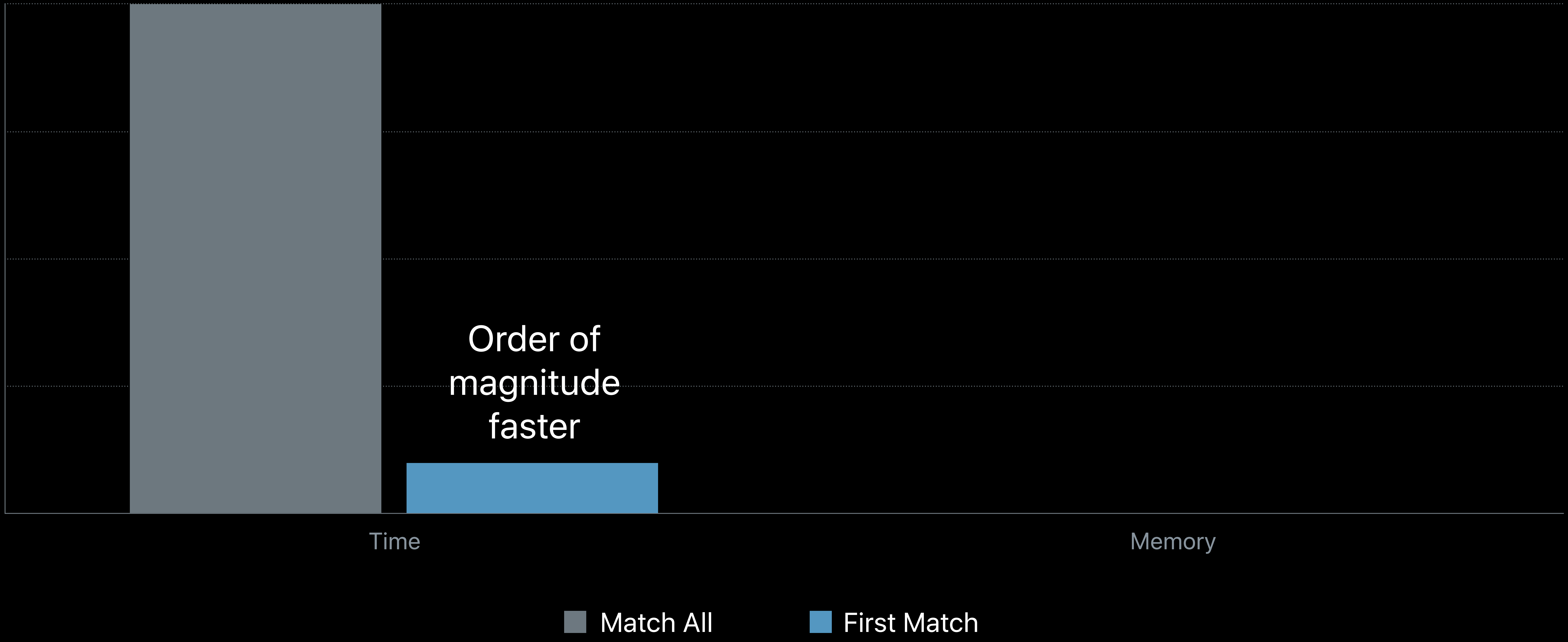
```
let button = app.navigationBars.buttons["Done"].firstMatch
```

First Match Performance

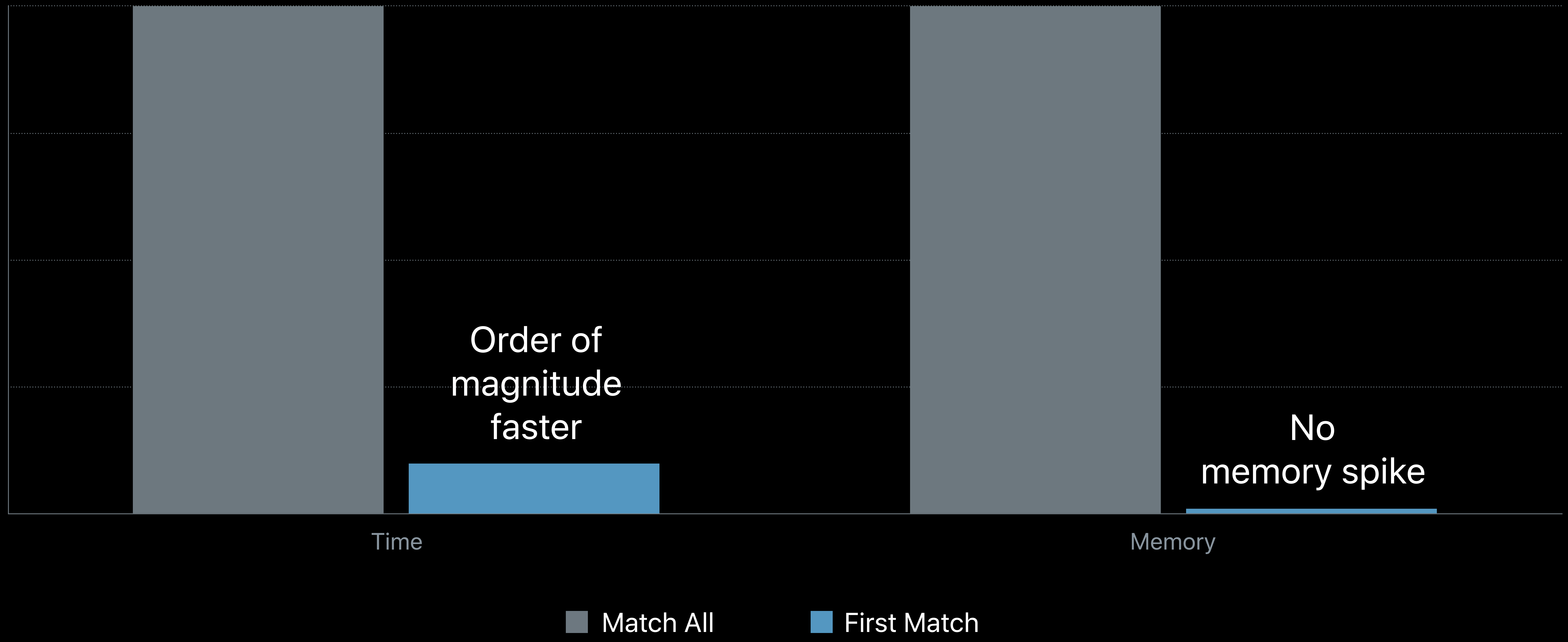
First Match Performance



First Match Performance



First Match Performance



First Match vs. Match All

First Match vs. Match All

Match all detects ambiguity

First Match vs. Match All

Match all detects ambiguity

First match requires precision

First Match vs. Match All

Match all detects ambiguity

First match requires precision

First Match vs. Match All

Match all detects ambiguity

First match requires precision

```
app.buttons.firstMatch // not a good idea!!
```

First Match vs. Match All

Match all detects ambiguity

First match requires precision

```
app.buttons.firstMatch // not a good idea!!
app.buttons["Done"].firstMatch // better
```

First Match vs. Match All

Match all detects ambiguity

First match requires precision

```
app.buttons.firstMatch // not a good idea!!
app.buttons["Done"].firstMatch // better
app.navigationBars.buttons["Done"].firstMatch // best
```

Block-based NSPredicate

Block-based NSPredicate

Prevents optimizations

Block-based NSPredicate

Prevents optimizations

No serialization

- ~~Remote query~~
- ~~First match~~

Block-based NSPredicate

Prevents optimizations

No serialization

- ~~Remote query~~
- ~~First match~~

No introspection

- ~~Reduced snapshot~~

Block-based NSPredicate

Block-based NSPredicate

Replace block predicates

- Format string
- NSExpression

Block-based NSPredicate

Replace block predicates

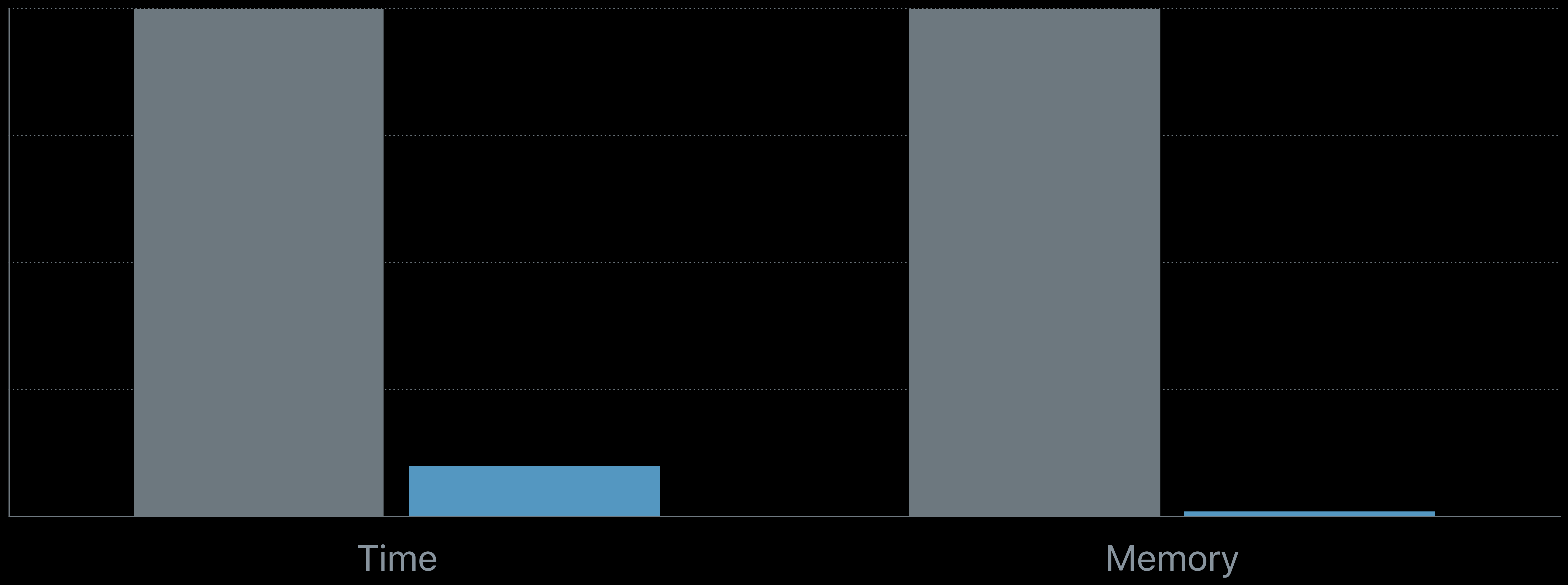
- Format string
- NSExpression

File enhancement requests!

UI Testing Performance

UI Testing Performance

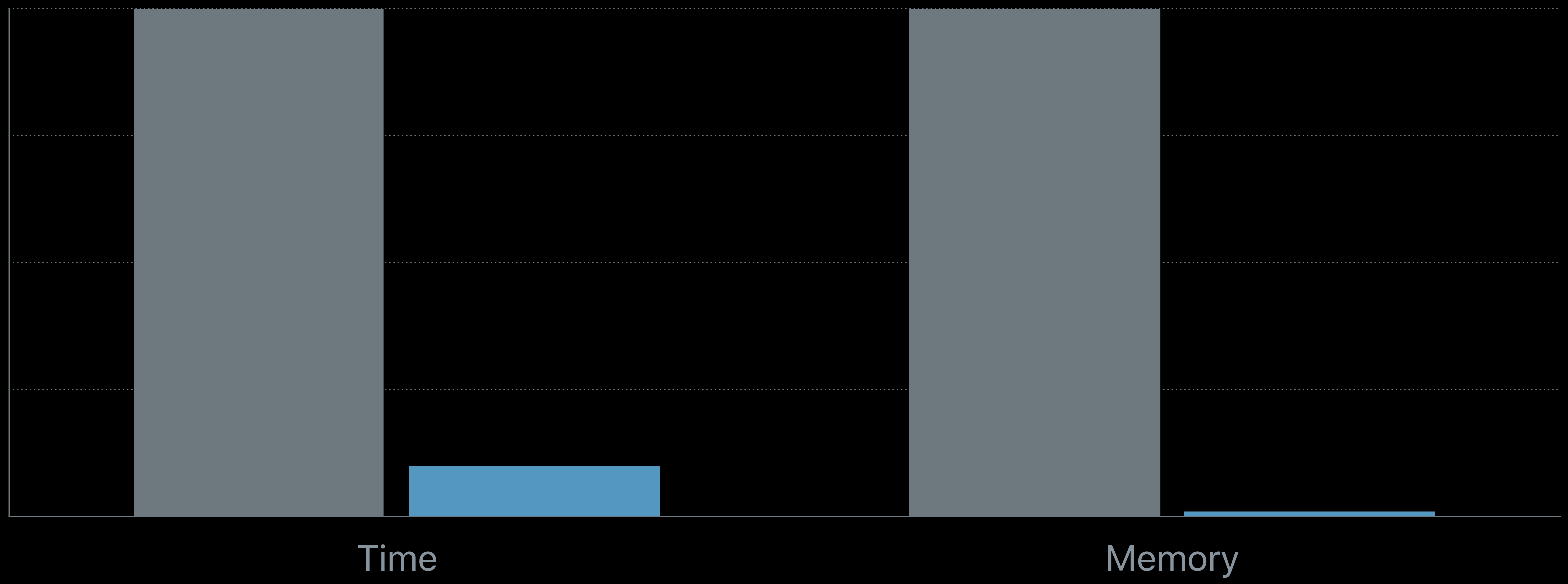
Faster in Xcode 9



UI Testing Performance

Faster in Xcode 9

Best on newest OS



Enhancements

Async testing

Multi-app testing

UI testing performance

Activities, attachments, and screenshots

Activities

NEW

Activities

NEW

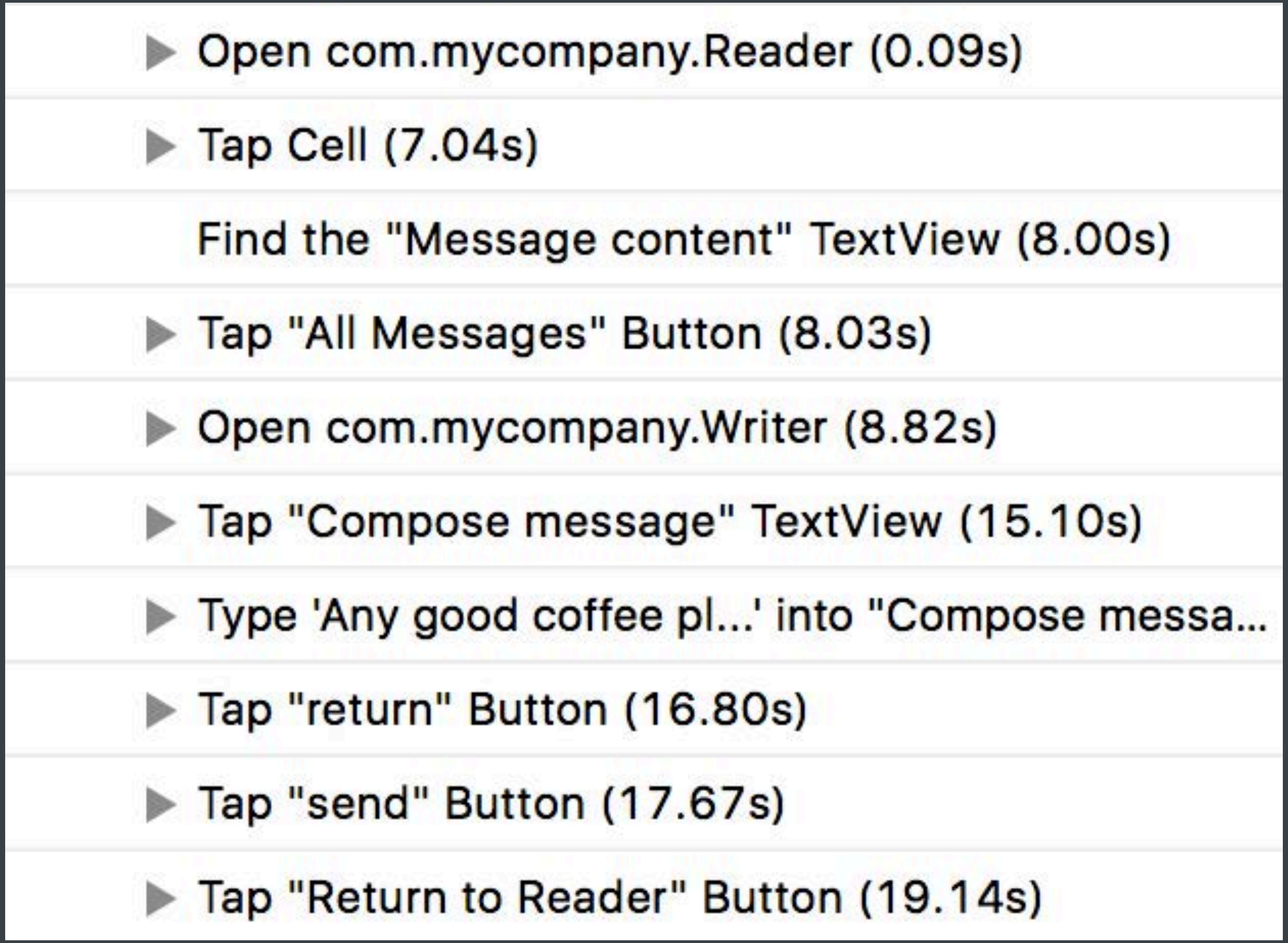
Create structure for long tests

Activities

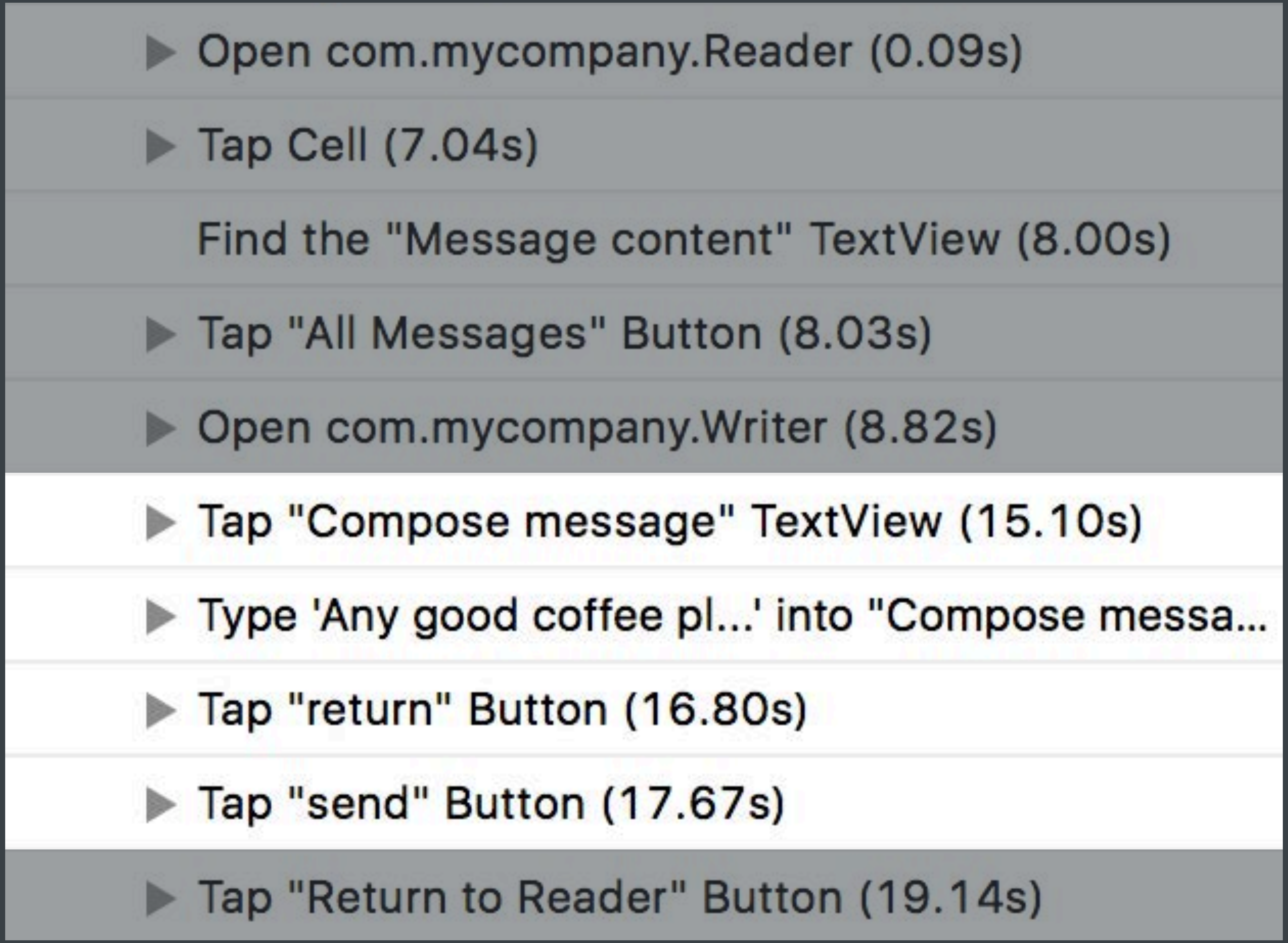
NEW

Create structure for long tests

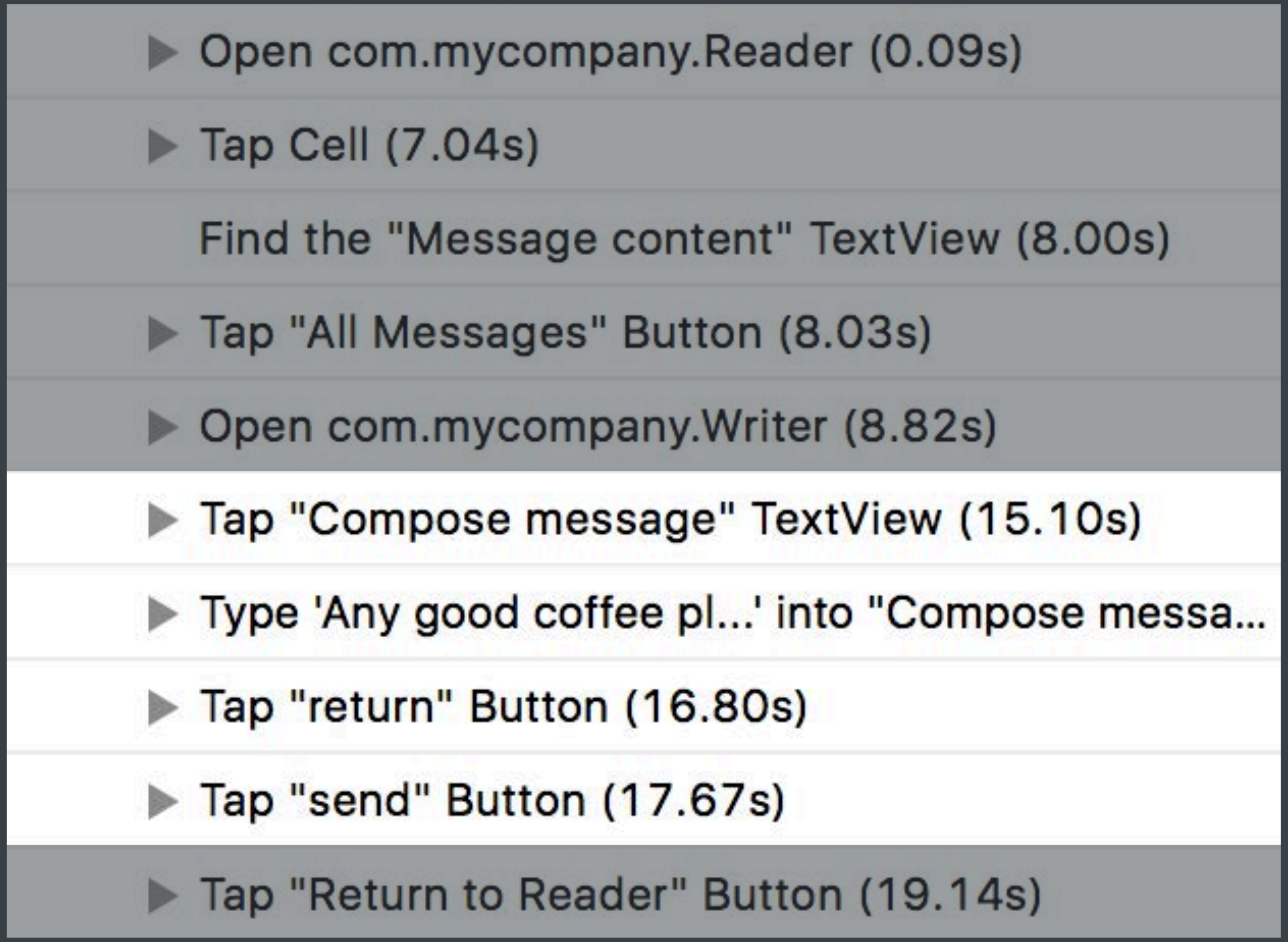
```
XCTContext.runActivity(named name: String, block: (XCTActivity))
```



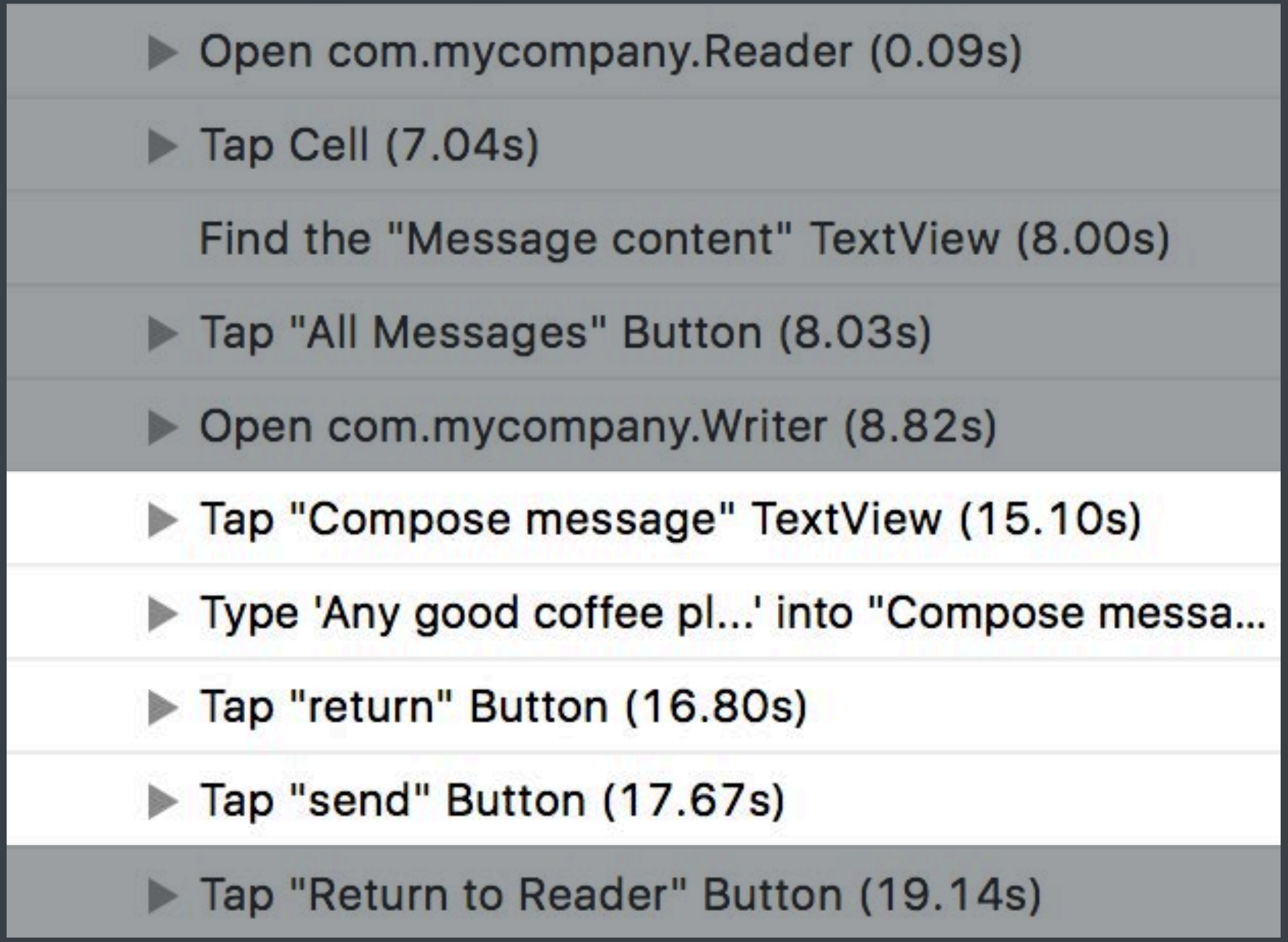
```
▶ Open com.mycompany.Reader (0.09s)
▶ Tap Cell (7.04s)
  Find the "Message content" TextView (8.00s)
▶ Tap "All Messages" Button (8.03s)
▶ Open com.mycompany.Writer (8.82s)
▶ Tap "Compose message" TextView (15.10s)
▶ Type 'Any good coffee pl...' into "Compose messa...
▶ Tap "return" Button (16.80s)
▶ Tap "send" Button (17.67s)
▶ Tap "Return to Reader" Button (19.14s)
```



```
▶ Open com.mycompany.Reader (0.09s)
▶ Tap Cell (7.04s)
  Find the "Message content" TextView (8.00s)
▶ Tap "All Messages" Button (8.03s)
▶ Open com.mycompany.Writer (8.82s)
▶ Tap "Compose message" TextView (15.10s)
▶ Type 'Any good coffee pl...' into "Compose messa...
▶ Tap "return" Button (16.80s)
▶ Tap "send" Button (17.67s)
▶ Tap "Return to Reader" Button (19.14s)
```

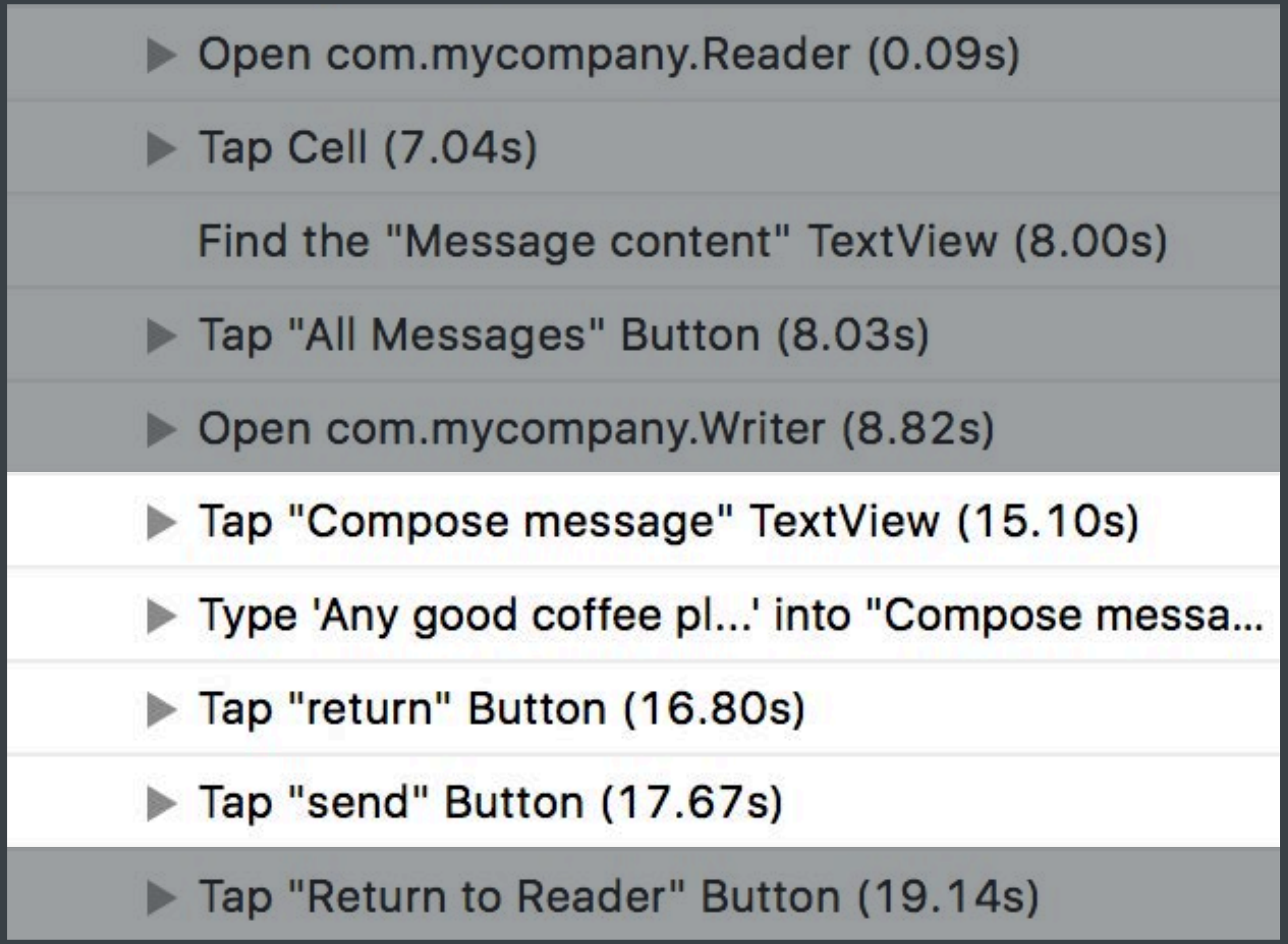

▶ Open com.mycompany.Reader (0.09s)
▶ Tap Cell (7.04s)
Find the "Message content" TextView (8.00s)
▶ Tap "All Messages" Button (8.03s)
▶ Open com.mycompany.Writer (8.82s)
▶ Tap "Compose message" TextView (15.10s)
▶ Type 'Any good coffee pl...' into "Compose messa...
▶ Tap "return" Button (16.80s)
▶ Tap "send" Button (17.67s)
▶ Tap "Return to Reader" Button (19.14s)

```swift
// Compose and send a new message
let composeView = writerApp.textViews["Compose message"]
composeView.tap()
composeView.typeText("Any good coffee places around McEnery? ☕ /cc @jane")
writerApp.buttons["return"].tap()
writerApp.buttons["send"].tap()
```

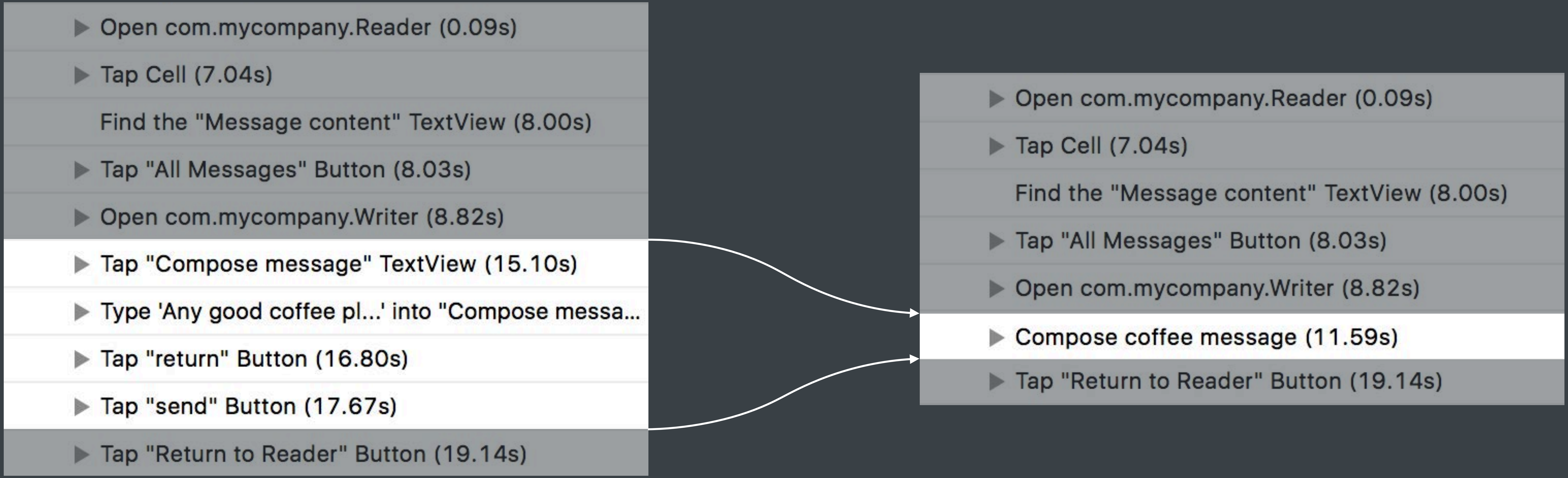


▶ Open com.mycompany.Reader (0.09s)
▶ Tap Cell (7.04s)
Find the "Message content" TextView (8.00s)
▶ Tap "All Messages" Button (8.03s)
▶ Open com.mycompany.Writer (8.82s)
▶ Tap "Compose message" TextView (15.10s)
▶ Type 'Any good coffee pl...' into "Compose messa...
▶ Tap "return" Button (16.80s)
▶ Tap "send" Button (17.67s)
▶ Tap "Return to Reader" Button (19.14s)

```
XCTContext.runActivity(named: "Compose coffee message") { _ in
    // Compose and send a new message
    let composeView = writerApp.textViews["Compose message"]
    composeView.tap()
    composeView.typeText("Any good coffee places around McEnery? ☕ /cc @jane")
    writerApp.buttons["return"].tap()
    writerApp.buttons["send"].tap()
}
```



```
▶ Open com.mycompany.Reader (0.09s)
▶ Tap Cell (7.04s)
  Find the "Message content" TextView (8.00s)
▶ Tap "All Messages" Button (8.03s)
▶ Open com.mycompany.Writer (8.82s)
▶ Tap "Compose message" TextView (15.10s)
▶ Type 'Any good coffee pl...' into "Compose messa...
▶ Tap "return" Button (16.80s)
▶ Tap "send" Button (17.67s)
▶ Tap "Return to Reader" Button (19.14s)
```



Open com.mycompany.Reader (0.09s)
Tap Cell (7.04s)
Find the "Message content" TextView (8.00s)
Tap "All Messages" Button (8.03s)
Open com.mycompany.Writer (8.82s)
Tap "Compose message" TextView (15.10s)
Type 'Any good coffee pl...' into "Compose messa...
Tap "return" Button (16.80s)
Tap "send" Button (17.67s)
Tap "Return to Reader" Button (19.14s)

Open com.mycompany.Reader (0.09s)
Tap Cell (7.04s)
Find the "Message content" TextView (8.00s)
Tap "All Messages" Button (8.03s)
Open com.mycompany.Writer (8.82s)
Compose coffee message (11.59s)
Tap "Return to Reader" Button (19.14s)

Attachments

NEW

Attachments

NEW

XCTAttachment

Attachments

NEW

Data from tests

`XCTAttachment`

Attachments

NEW

Data from tests

Improved triage

XCTAttachment

Attachments

NEW

Data from tests

Improved triage

Post-processing

XCTAttachment

Attachments

NEW

XCTAttachment

Attachments

NEW

Raw binary data

XCTAttachment

Attachments

NEW

Raw binary data

Strings

`XCTAttachment`

Attachments

NEW

Raw binary data

Strings

Property lists

XCTAttachment

Attachments

NEW

Raw binary data

Strings

Property lists

Codable objects

`XCTAttachment`

Attachments

NEW

Raw binary data

Strings

Property lists

Codable objects

Files

XCTAttachment

Attachments

NEW

Raw binary data

Strings

Property lists

Codable objects

Files

Images

`XCTAttachment`

Screenshots!

NEW

Screenshots!

NEW

`XCUIScreenshotProviding`

Screenshots!

NEW

API for capturing on demand

`XCUIScreenshotProviding`

Screenshots!

NEW

API for capturing on demand

`XCUIElement.screenshot`

`XCUIScreenshotProviding`

Screenshots!

NEW

API for capturing on demand

`XCUIElement.screenshot`

`XCUIScreen.screenshot`

`XCUIScreenshotProviding`

Attachment Lifetime Policies

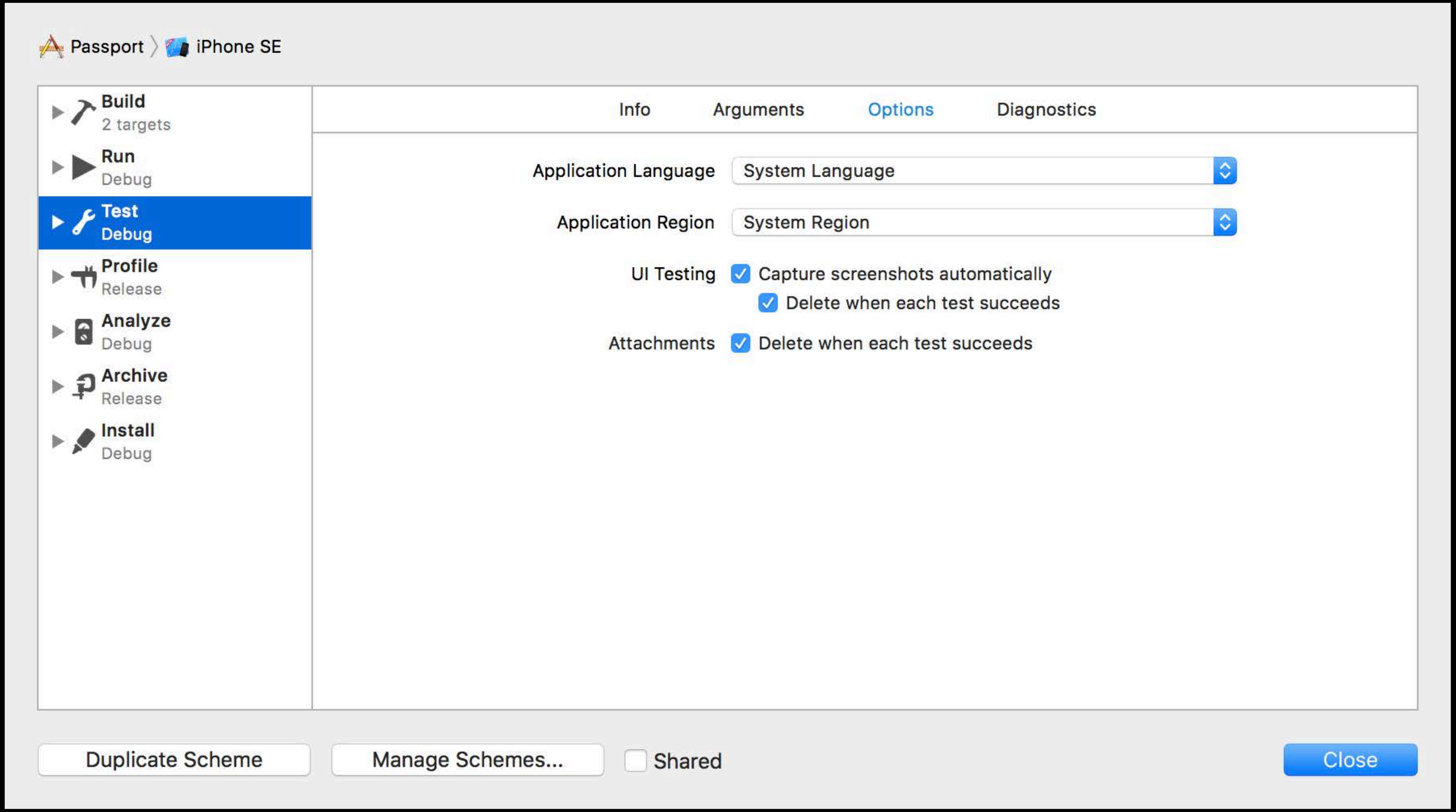
Attachment Lifetime Policies

Delete if test passes

Attachment Lifetime Policies

Delete if test passes

Scheme option


Passport 〉 iPhone SE

Application Language: System Language

Application Region: System Region

UI Testing: ✓ Capture screenshots automatically
✓ Delete when each test succeeds

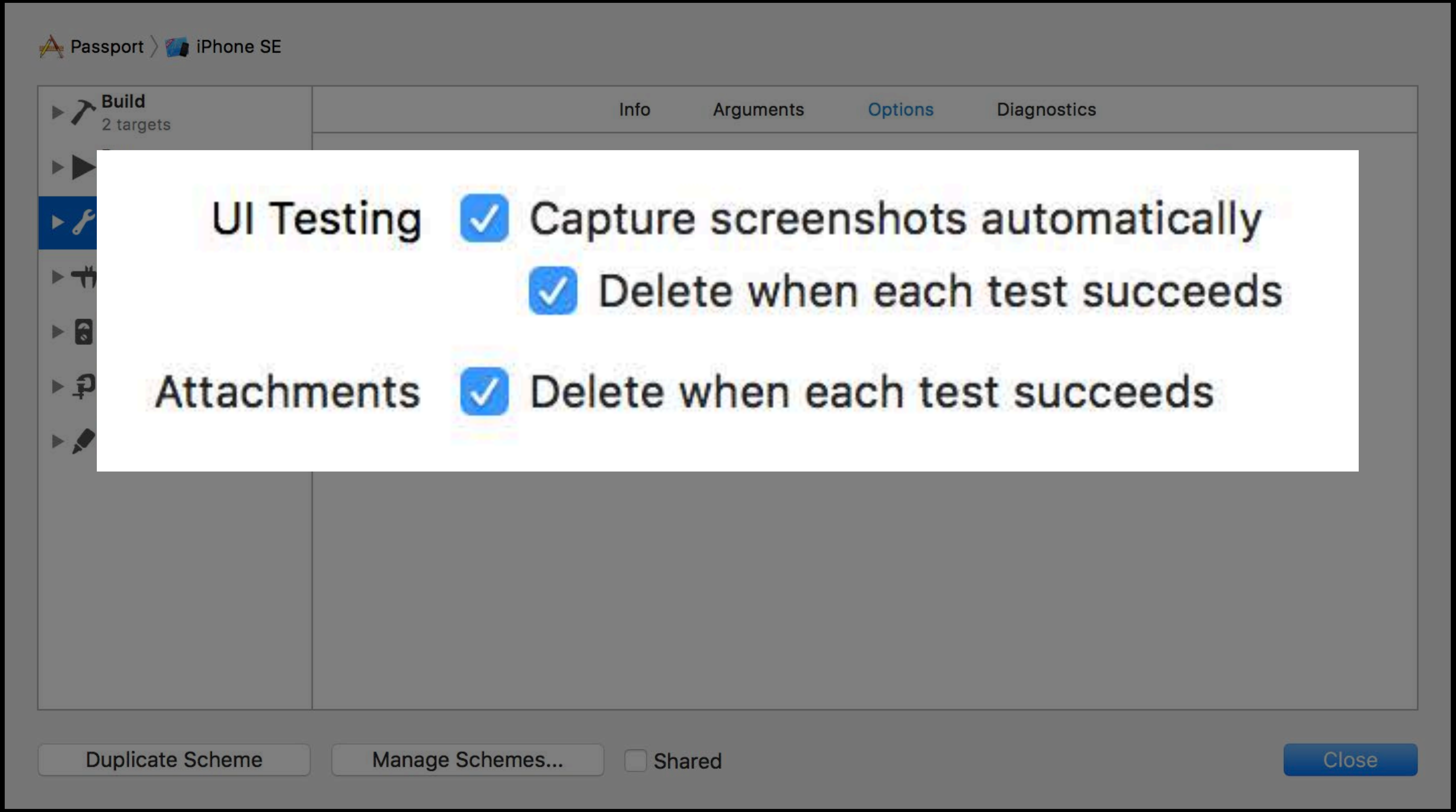
Attachments: ✓ Delete when each test succeeds

☐ Shared

Attachment Lifetime Policies

Delete if test passes

Scheme option



UI Testing [✓] Capture screenshots automatically

[✓] Delete when each test succeeds

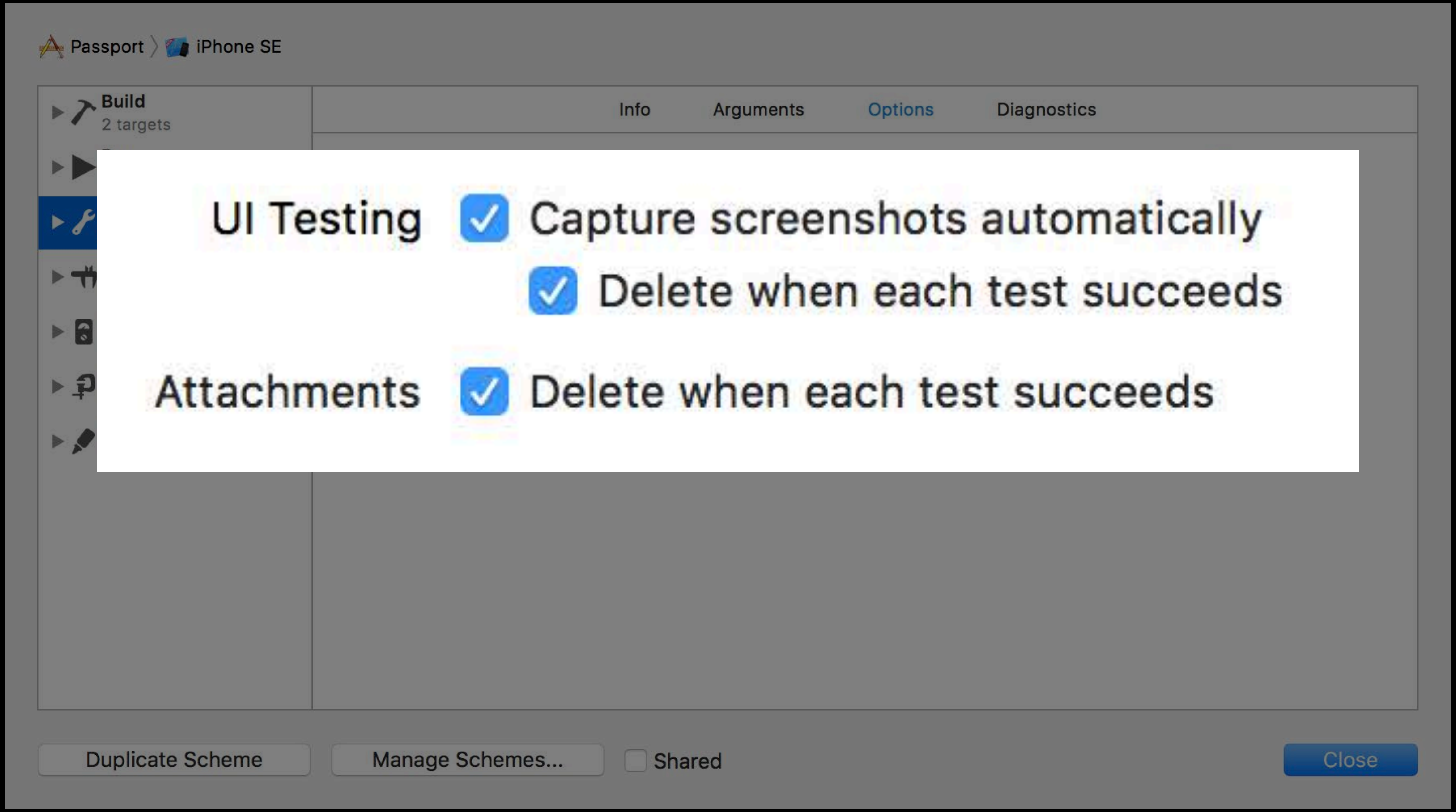
Attachments [✓] Delete when each test succeeds

Attachment Lifetime Policies

Delete if test passes

Scheme option

XCTAttachment API



UI Testing ✓ Capture screenshots automatically
✓ Delete when each test succeeds

Attachments ✓ Delete when each test succeeds

Demo

Activities, attachments, and screenshots

Honza Dvorsky, Xcode Engineer

What’s New in Testing?

What’s New in Testing?

Many new APIs!

What’s New in Testing?

Many new APIs!

Workflow and CI features

What’s New in Testing?

Many new APIs!

Workflow and CI features

Performance improvements

More Information

https://developer.apple.com/wwdc17/409

Related Sessions

Engineering for Testability	Hall 3	Friday 1:50PM
Localizing with Xcode 9		WWDC 2017
What’s New in Signing for Xcode and Xcode Server		WWDC 2017
What’s New in Accessibility		WWDC 2017
Advanced Testing and Continuous Integration		WWDC 2016
UI Testing in Xcode		WWDC 2015
Testing in Xcode 6		WWDC 2014

Labs

Source Control, Simulator, Testing, and Continuous Integration with Xcode Lab	Technology Lab K	Thu 4:10PM–6:00PM
Xcode Open Hours	Technology Lab K	Fri 1:50PM–4:00PM

WWDC17