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
XCUIElement.Type.touchBar

Making Great SiriKit Experiences
XCTest in Xcode 9
XCCTest in Xcode 9

Swift 4
XCTest in Xcode 9

Swift 4

Block-based test teardown
UI Testing
UI Testing

XCUIClient.Type.statusItem
UI Testing

XCUIElement.Type.statusItem
XCUIElement.waitForExistence()
xcodebuild
xcodbuild

CoreSimulator
xcodebuild

CoreSimulator

Parallel testing
Localization
Localization
Xcode Server
Xcode Server
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

Server & Bots

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

Settings Permissions Mail

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

What's New in Signing in Xcode and Xcode Server

WWDC 2017
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”
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)
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)
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)
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

Extracted logic from XCTestCase
XCTWaiter

Extracted logic from XCTestCase

Explicit list of expectations
 XCTWaiter

Extracted logic from XCTestCase

Explicit list of expectations

Calls back to XCTWaiterDelegate
XCTWaiter

Extracted logic from XCTestCase

Explicit list of expectations

Calls back to XCTWaiterDelegate

Returns XCTWaiter.Result
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)
// Test case waits implicitly
waitForExpectations(timeout: 10)

// Test case waits \textit{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)
// 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

Public initializer
XCTestExpectation

Public initializer

• Decoupled from XCTestCase
XCTestExpectation

Public initializer
• Decoupled from XCTestCase

Multiple fulfillments
XCTestExpectation

Public initializer
• Decoupled from XCTestCase

Multiple fulfillments

Inverted behavior
XCTestExpectation

Public initializer

• Decoupled from XCTestCase

Multiple fulfillments

Inverted behavior

Ordering enforcement
Async Testing
Async Testing

XCTWaiter manages expectations
Async Testing

XCTWaiter manages expectations

XCTestExpectation has new features
Async Testing

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

![Project configuration screenshot](image_url)
Target Application

Project configuration

Default initializer

```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
Additions to XCUIApplication

New initializers

init(bundleIdentifier: String)
init(url: URL)
Additions to XCUIApplication

New initializers
- `init(bundleIdentifier: String)`
- `init(url: URL)`

Activate method
- `func activate()`
Additions to XCUIApplication

New initializers
- `init(bundleIdentifier: String)`
- `init(url: URL)`

Activate method
- `func activate()`

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

```swift
let button = app.navigationController.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
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
Optimization 1: Remote Queries
Reduce serialization and transport overhead

Don’t fetch the snapshot
Transmit the query
Evaluate remotely
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

Time

Memory

- Fetched Snapshot
- Remote Query
Remote Query Performance

- **20% Faster**

<table>
<thead>
<tr>
<th>Time</th>
<th>Fetched Snapshot</th>
<th>Remote Query</th>
</tr>
</thead>
</table>

- **Memory**

- **Legend**:
  - Grey: Fetched Snapshot
  - Blue: Remote Query
Remote Query Performance

- Time: 20% Faster
- Memory: 30% Less Memory

Fetched Snapshot | Remote Query
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

- Time
- Memory

- Full Snapshot
- Reduced Snapshot
Query Analysis Performance

- **Time**: 50% Faster
- **Memory**: Reduced Snapshot

Legend:
- Full Snapshot
- Reduced Snapshot
Query Analysis Performance

- **Time**:
  - Full Snapshot: 50% Faster

- **Memory**:
  - Reduced Snapshot: 35% Less Memory
Optimization 3: Eliminate Snapshots

“First match” API
Optimization 3: Eliminate Snapshots

“First match” API

Queries search entire tree
Optimization 3: Eliminate Snapshots

“First match” API

Queries search entire tree

First match halts early

```swift
var firstMatch: XCUIElement { get }
```
**Optimization 3: Eliminate Snapshots**

“First match” API

Queries search entire tree

First match halts early

```swift
var firstMatch: XCUIElement { get }
```
Optimization 3: Eliminate Snapshots

“First match” API

Queries search entire tree

First match halts early

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

- Time
- Memory

- Match All
- First Match
**First Match Performance**

- **Time**: Order of magnitude faster
- **Memory**: Match All vs First Match
First Match Performance

Order of magnitude faster

No memory spike
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

```javascript
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.navigationBar.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

Chart showing performance improvements in Xcode 9 for UI testing.
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
Activities

Create structure for long tests
Activities

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 message" TextView (15.20s)

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)
// 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()
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()
}
<table>
<thead>
<tr>
<th>Task</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open com.mycompany.Reader</td>
<td>0.09s</td>
</tr>
<tr>
<td>Tap Cell</td>
<td>7.04s</td>
</tr>
<tr>
<td>Find the &quot;Message content&quot; TextView</td>
<td>8.00s</td>
</tr>
<tr>
<td>Tap &quot;All Messages&quot; Button</td>
<td>8.03s</td>
</tr>
<tr>
<td>Open com.mycompany.Writer</td>
<td>8.82s</td>
</tr>
<tr>
<td>Tap &quot;Compose message&quot; TextView</td>
<td>15.10s</td>
</tr>
<tr>
<td>Type 'Any good coffee pl...' into &quot;Compose message&quot;</td>
<td></td>
</tr>
<tr>
<td>Tap &quot;return&quot; Button</td>
<td>16.80s</td>
</tr>
<tr>
<td>Tap &quot;send&quot; Button</td>
<td>17.67s</td>
</tr>
<tr>
<td>Tap &quot;Return to Reader&quot; Button</td>
<td>19.14s</td>
</tr>
</tbody>
</table>
Attachments
Attachments

XCTAttachment
Attachments

Data from tests

XCTAttachment
Attachments

Data from tests

Improved triage

XCTAttachment
Attachments

Data from tests
Improved triage
Post-processing

XCTAttachment
Attachments

Raw binary data

XCTAttachment
Attachments

Raw binary data

Strings

XCTAttachment
Attachments

Raw binary data
Strings
Property lists

XCTAttachment
Attachments

- Raw binary data
- Strings
- Property lists
- Codable objects

XCTAttachment
Attachments

- Raw binary data
- Strings
- Property lists
- Codable objects
- Files
Attachments

Raw binary data
Strings
Property lists
Codable objects
Files
Images

XCTAttachment
Screenshots!
Screenshots!

XCUIScreenshotProviding
Screenshots!

API for capturing on demand

XCUIScreenshotProviding
Screenshots!

API for capturing on demand

`XCUIElement.screenshot`

`XCUIScreenshotProviding`
Screenshots!

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
Attachment Lifetime Policies

Delete if test passes

Scheme option
Attachment Lifetime Policies

Delete if test passes

Scheme option

XCTAttachment API
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

# Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Venue</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering for Testability</td>
<td>Hall 3</td>
<td>Friday 1:50PM</td>
</tr>
<tr>
<td>Localizing with Xcode 9</td>
<td></td>
<td>WWDC 2017</td>
</tr>
<tr>
<td>What's New in Signing for Xcode and Xcode Server</td>
<td></td>
<td>WWDC 2017</td>
</tr>
<tr>
<td>What's New in Accessibility</td>
<td></td>
<td>WWDC 2017</td>
</tr>
<tr>
<td>Advanced Testing and Continuous Integration</td>
<td></td>
<td>WWDC 2016</td>
</tr>
<tr>
<td>UI Testing in Xcode</td>
<td></td>
<td>WWDC 2015</td>
</tr>
<tr>
<td>Testing in Xcode 6</td>
<td></td>
<td>WWDC 2014</td>
</tr>
<tr>
<td>Source Control, Simulator, Testing, and Continuous Integration with Xcode Lab</td>
<td>Technology Lab K</td>
<td>Thu 4:10PM–6:00PM</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Xcode Open Hours</td>
<td>Technology Lab K</td>
<td>Fri 1:50PM–4:00PM</td>
</tr>
</tbody>
</table>