Testing in Xcode 6

Session 414

Brooke Callahan
Xcode Software Engineer

Wil Turner
Xcode Software Engineer

What We'll Cover

What We'll Cover

Benefits of testing

What We'll Cover

Benefits of testing

Getting started

What We'll Cover

Benefits of testing

Getting started

Asynchronous testing

What We'll Cover

Benefits of testing

Getting started

Asynchronous testing

Performance testing

Overview

Motivation

Why test?

Motivation

Why test?

Find bugs

Motivation

Why test?

Find bugs

Codify requirements

Workflow

Getting started

Workflow

Getting started

Add tests

Workflow

Getting started

Add tests

Verify that tests pass

Workflow

Getting started

Add tests

Verify that tests pass

Or

Workflow

Getting started

Add tests

Verify that tests pass

Or

Write tests

Workflow

Getting started

Add tests

Verify that tests pass

Or

Write tests

Write code that passes the tests

Workflow

Getting started

Add tests

Verify that tests pass

Or

Write tests

Write code that passes the tests

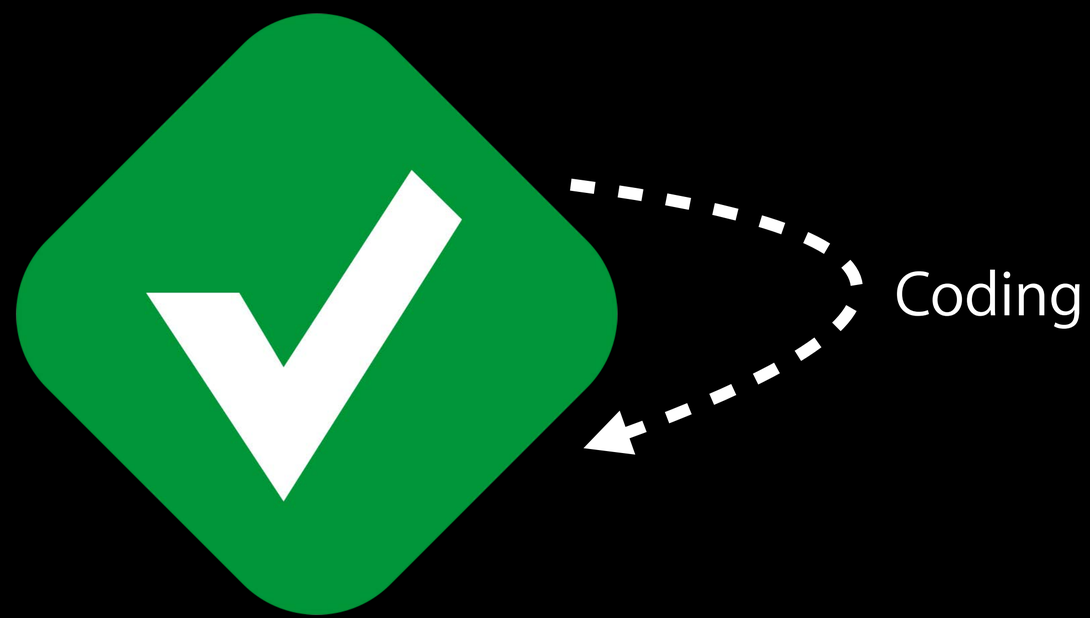
AKA “Test-Driven Development”

Workflow

Continuous Integration

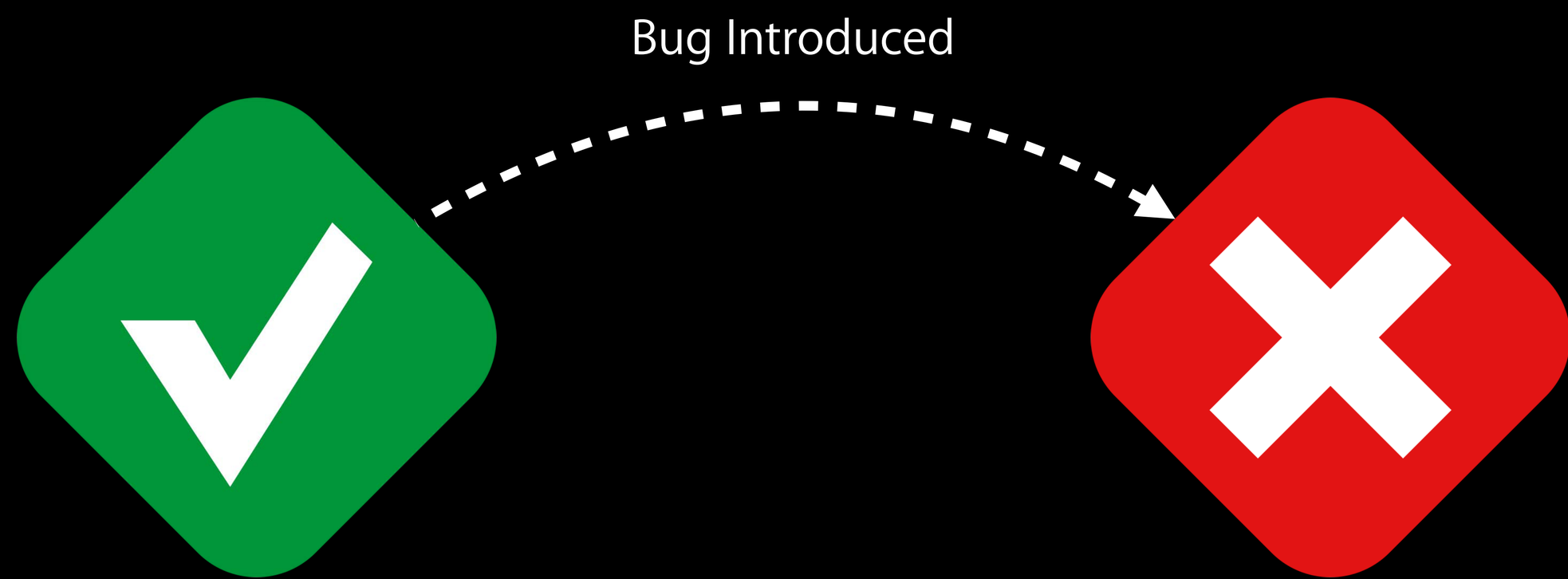
Workflow

Continuous Integration



Workflow

Continuous Integration



Workflow

Continuous Integration

Getting Started

XCTest

Xcode's framework for testing

XCTest

Xcode's framework for testing

Subclass XCTestCase

XCTest

Xcode's framework for testing

Subclass XCTestCase

Implement test methods

XCTest

Xcode's framework for testing

Subclass XCTestCase

Implement test methods

- `- (void)testThatMyFunctionWorks`

XCTest

Xcode's framework for testing

Subclass XCTestCase

Implement test methods

- `- (void)testThatMyFunctionWorks`

Use assertion APIs to report failures

XCTest

Xcode's framework for testing

Subclass XCTestCase

Implement test methods

- `- (void)testThatMyFunctionWorks`

Use assertion APIs to report failures

- `XCTAssertEqual(value, expectedValue);`

Test Targets

Test Targets

Tests targets build bundles

Test Targets

Tests targets build bundles

- Test code

Test Targets

Tests targets build bundles

- Test code
- Resources

Test Targets

Tests targets build bundles

- Test code
- Resources

Automatically included in new projects

Test Targets

Tests targets build bundles

- Test code
- Resources

Automatically included in new projects

Existing projects can add test targets

Test Hosting

How tests are run

Test Hosting

How tests are run

Test bundles are executed by a host process

Test Hosting

How tests are run

Test bundles are executed by a host process

- Injected into your app, or

Test Hosting

How tests are run

Test bundles are executed by a host process

- Injected into your app, or
- Hosting process provided by Xcode

Test Hosting

How tests are run

Test bundles are executed by a host process

- Injected into your app, or
- Hosting process provided by Xcode

Resources for tests are not in the main bundle

Test Hosting

How tests are run

Test bundles are executed by a host process

- Injected into your app, or
- Hosting process provided by Xcode

Resources for tests are not in the main bundle

- Don’t use +[NSBundle mainBundle]

Test Hosting

How tests are run

Test bundles are executed by a host process

- Injected into your app, or
- Hosting process provided by Xcode

Resources for tests are not in the main bundle

- Don’t use +[NSBundle mainBundle]
- Use +[NSBundle bundleForClass:[MyTest class]]

Xcode Integration

Running tests

Xcode Integration

Running tests

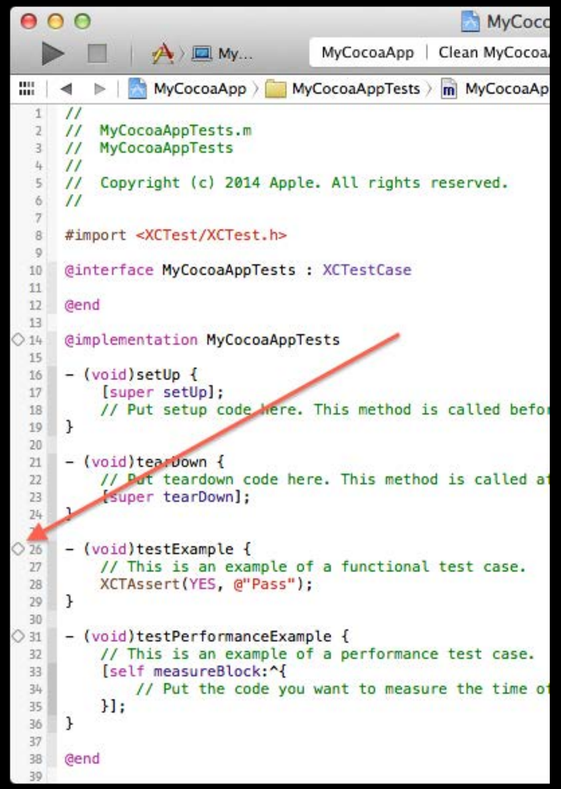
Command-U

Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar



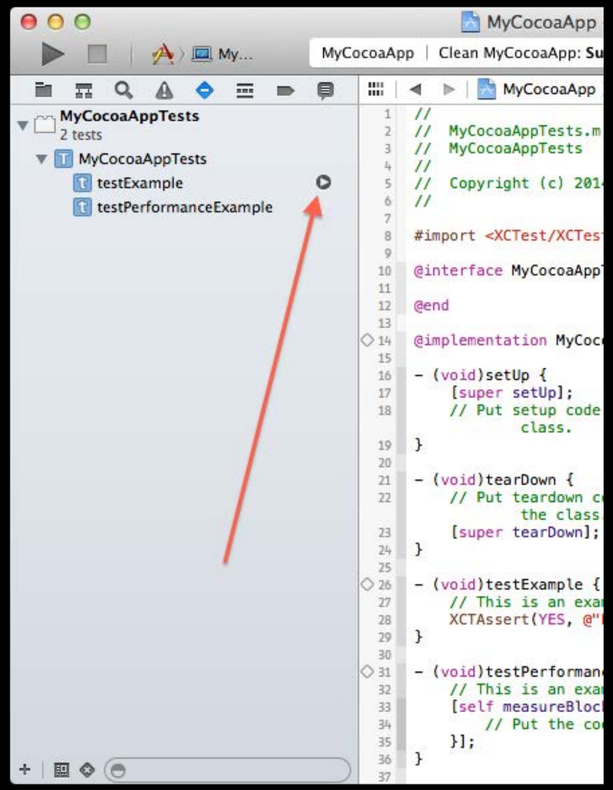
Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar

Buttons in the Test Navigator



Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar

Buttons in the Test Navigator

xcodebuild

Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar

Buttons in the Test Navigator

xcodebuild

```
xcodebuild test \
```

Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar

Buttons in the Test Navigator

xcodebuild

```
xcodebuild test \
   -project ~/Documents/MyApp.xcodeproj \
```

Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar

Buttons in the Test Navigator

xcodebuild

```
xcodebuild test \
    -project ~/Documents/MyApp.xcodeproj \
    -scheme MyApp \
```

Xcode Integration

Running tests

Command-U

Buttons in the Source Editor side bar

Buttons in the Test Navigator

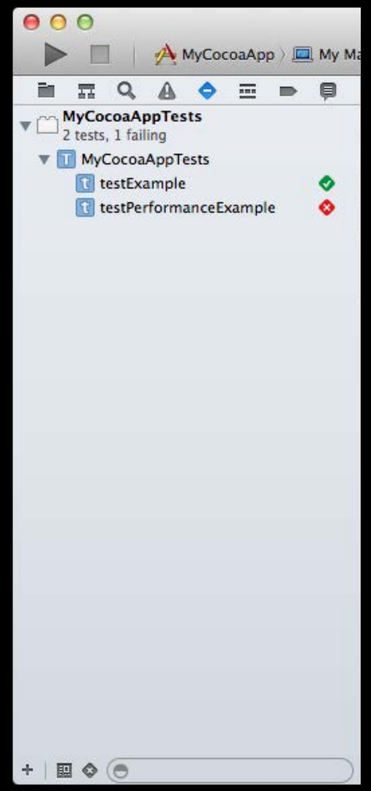
xcodebuild

```
xcodebuild test \
    -project ~/Documents/MyApp.xcodeproj \
    -scheme MyApp \
    -destination 'platform=iOS,name=iPhone'
```

Xcode Integration

Viewing results

Test Navigator

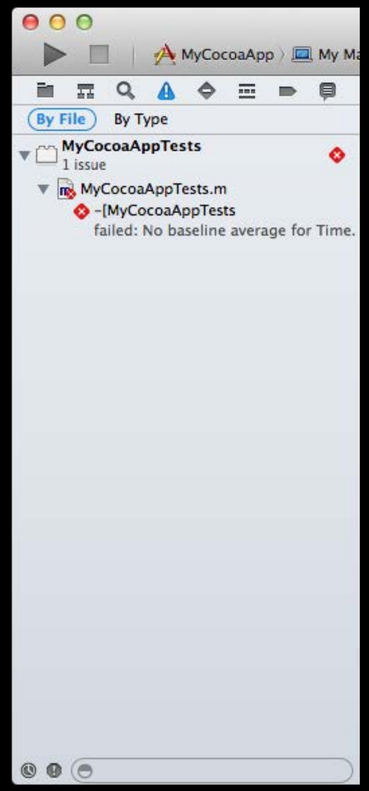


Xcode Integration

Viewing results

Test Navigator

Issue Navigator



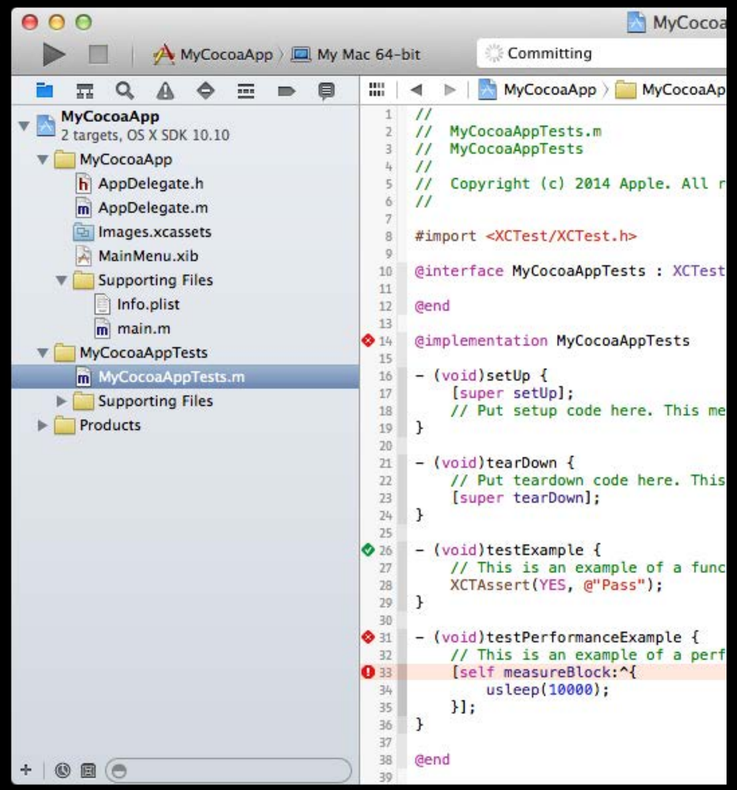
Xcode Integration

Viewing results

Test Navigator

Issue Navigator

Source Editor side bar



Xcode Integration

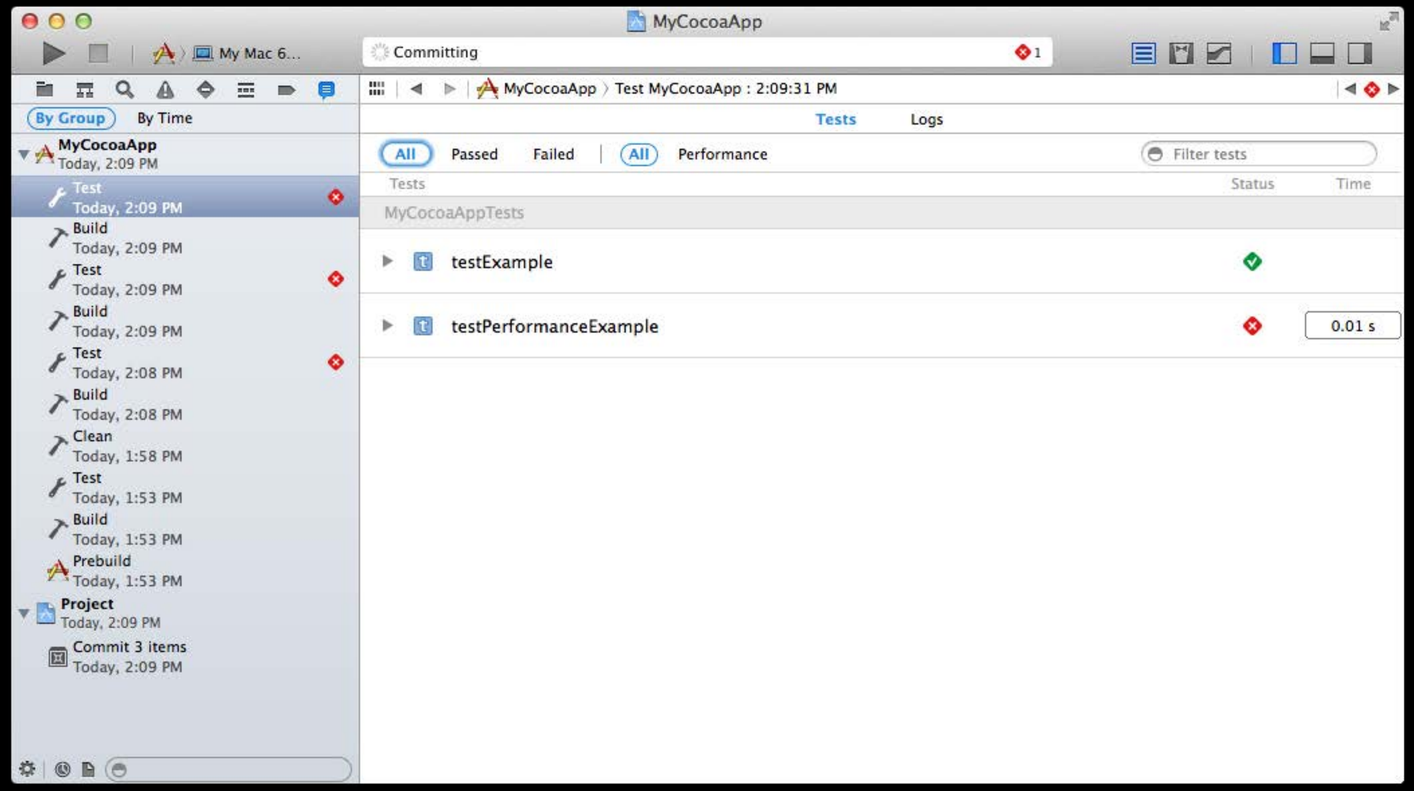
Viewing results

Test Navigator

Issue Navigator

Source Editor side bar

Test Reports



Demo

Adding tests to an existing project

What's New

APIs and tools

Testing with Xcode 6

Testing with Xcode 6

Compatibility improvements

Testing with Xcode 6

Compatibility improvements

Instruments integration

Testing with Xcode 6

Compatibility improvements

Instruments integration

New APIs

Compatibility Improvements

XCTest on older iOS versions

NEW

Compatibility Improvements

XCTest on older iOS versions

NEW

Originally part of iOS

Compatibility Improvements

XCTest on older iOS versions

NEW

Originally part of iOS

Now ships with Xcode

Compatibility Improvements

XCTest on older iOS versions

NEW

Originally part of iOS

Now ships with Xcode

New XCTest features work with older iOS versions

Compatibility Improvements

XCTest on older iOS versions

Originally part of iOS

Now ships with Xcode

New XCTest features work with older iOS versions

iOS 6 and later

OCUnit

OCUnit

Deprecated in Xcode 5.1

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

Use Migrator (recommended)

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

Use Migrator (recommended)

- Edit > Refactor > Convert to XCTest

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

Use Migrator (recommended)

- Edit > Refactor > Convert to XCTest
- Some settings not in UI

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

Use Migrator (recommended)

- Edit > Refactor > Convert to XCTest
- Some settings not in UI

Alternatively

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

Use Migrator (recommended)

- Edit > Refactor > Convert to XCTest
- Some settings not in UI

Alternatively

- Create a new test target

OCUnit

Deprecated in Xcode 5.1

New features are only in XCTest

Use Migrator (recommended)

- Edit > Refactor > Convert to XCTest
- Some settings not in UI

Alternatively

- Create a new test target
- Move tests manually

Asynchronous Testing

Asynchronous Testing

More and more APIs are asynchronous

Asynchronous Testing

More and more APIs are asynchronous

- Block invocations

Asynchronous Testing

More and more APIs are asynchronous

- Block invocations
- Delegate callbacks

Asynchronous Testing

More and more APIs are asynchronous

- Block invocations
- Delegate callbacks
- Network requests

Asynchronous Testing

More and more APIs are asynchronous

- Block invocations
- Delegate callbacks
- Network requests
- Background processing

Asynchronous Testing

More and more APIs are asynchronous

- Block invocations
- Delegate callbacks
- Network requests
- Background processing

Unit tests run synchronously

Asynchronous Testing

New APIs in XCTest

Asynchronous Testing

New APIs in XCTest

"Expectation" objects describe expected events

```
- (XCTestExpectation *)expectationWithDescription:(NSString *)description;
```

Asynchronous Testing

New APIs in XCTest

"Expectation" objects describe expected events

```
- (XCTestExpectation *)expectationWithDescription:(NSString *)description;
```

XCTestCase waits for expectations to "fulfill"

Asynchronous Testing

New APIs in XCTest

"Expectation" objects describe expected events

```
- (XCTestExpectation *)expectationWithDescription:(NSString *)description;
```

XCTestCase waits for expectations to "fulfill"

```
- (void)waitForExpectationsWithTimeout:(NSTimeInterval)timeout
                                handler:(XCWaitCompletionHandler)handlerOrNil;
```

Asynchronous Testing

Example

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
```

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
    XCTestExpectation *expectation = [self expectationWithDescription:@“open doc”];
```

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
    XCTestExpectation *expectation = [self expectationWithDescription:@“open doc”];

    UIDocument *doc = …;

    [doc openWithCompletionHandler:^(BOOL success) {
```

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
    XCTestExpectation *expectation = [self expectationWithDescription:@“open doc”];

    UIDocument *doc = …;

    [doc openWithCompletionHandler:^(BOOL success) {


    }];
```

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
    XCTestExpectation *expectation = [self expectationWithDescription:@“open doc”];

    UIDocument *doc = …;

    [doc openWithCompletionHandler:^(BOOL success) {


    }];

    [self waitForExpectationsWithTimeout:5.0 handler:nil];
}
```

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
    XCTestExpectation *expectation = [self expectationWithDescription:@“open doc”];

    UIDocument *doc = …;

    [doc openWithCompletionHandler:^(BOOL success) {
        XCTAssert(success);
        [expectation fulfill];
    }];

    [self waitForExpectationsWithTimeout:5.0 handler:nil];
}
```

Asynchronous Testing

Example

```
- (void)testDocumentOpening
{
    XCTestExpectation *expectation = [self expectationWithDescription:@“open doc”];

    UIDocument *doc = …;

    [doc openWithCompletionHandler:^(BOOL success) {
        XCTAssert(success);
        [expectation fulfill];
    }];

    [self waitForExpectationsWithTimeout:5.0 handler:nil];
}
```

Demo

Writing an asynchronous test

Performance Testing

Brooke Callahan
Xcode Software Engineer

Performance Testing

Code changes can introduce performance regressions

Performance Testing

Code changes can introduce performance regressions

Catching these regressions is difficult

Performance Testing

Code changes can introduce performance regressions

Catching these regressions is difficult

Performance testing automates this

Overview

New APIs to measure performance

Overview

New APIs to measure performance

New UI to interpret results

Overview

New APIs to measure performance

New UI to interpret results

Profiling tests with Instruments

Measuring Performance

New API in XCTestCase

```
- (void)measureBlock:(void (^)(void))block;
```

Takes a block of code and runs it 10 times

Measuring Performance

New API in XCTestCase

```
- (void)measureBlock:(void (^)(void))block;
```

Takes a block of code and runs it 10 times

Measures time

Measuring Performance

New API in XCTestCase

```
- (void)measureBlock:(void (^)(void))block;
```

Takes a block of code and runs it 10 times

Measures time

Results show up in Xcode

Measuring Performance

Example

```
- (void)testUseFileHandlePerformance
{



}
```

Measuring Performance

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureBlock:^{



    }];
}
```

Measuring Performance

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureBlock:^{

        NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
        XCTAssertNotNil(fileHandle);

        UseFileHandle(fileHandle);

        [fileHandle closeFile];
    }];
}
```

Profiling Tests

Instruments integration

NEW

Profile individual tests

- Source Editor context menu
- Test Navigator context menu

Profiling Tests

Instruments integration

Profile individual tests

- Source Editor context menu
- Test Navigator context menu

New Command in Product menu

- Perform Action -> Profile “My App Tests”

Profiling Tests

Building for Profiling

NEW

Profiling Tests

Building for Profiling

NEW

Uses settings from Scheme Profile Action

Profiling Tests

Building for Profiling

Uses settings from Scheme Profile Action

Behavior may be different when Profiling

- Test builds for Debug
- Profile builds for Release

Demo

Performance testing

Measuring Performance

Wrap-Up

Call `-measureBlock:` to detect performance regressions

Measuring Performance

Wrap-Up

Call `-measureBlock:` to detect performance regressions

View results in Source Editor and Test Report

Measuring Performance

Wrap-Up

Call `-measureBlock:` to detect performance regressions

View results in Source Editor and Test Report

Profile tests with Instruments

Performance Testing

Setting Baselines

Performance Testing

Setting Baselines

Standard Deviation

Performance Testing

Setting Baselines

Standard Deviation

Measuring precisely

Detecting Regressions

Baseline is the Average from a previous run

Detecting Regressions

Baseline is the Average from a previous run

Set Baseline Average to detect regressions

- Fail if >10% increase from Baseline Average
- Regressions less than 0.1 seconds are ignored

Detecting Regressions

Baseline is the Average from a previous run

Set Baseline Average to detect regressions

- Fail if >10% increase from Baseline Average
- Regressions less than 0.1 seconds are ignored

Baselines are stored in source

Detecting Regressions

Baseline is the Average from a previous run

Set Baseline Average to detect regressions

- Fail if >10% increase from Baseline Average
- Regressions less than 0.1 seconds are ignored

Baselines are stored in source

Baselines are per-device configuration

- Includes device model, CPU, and OS

Setting Baselines

Source Editor Annotations

- No Baseline

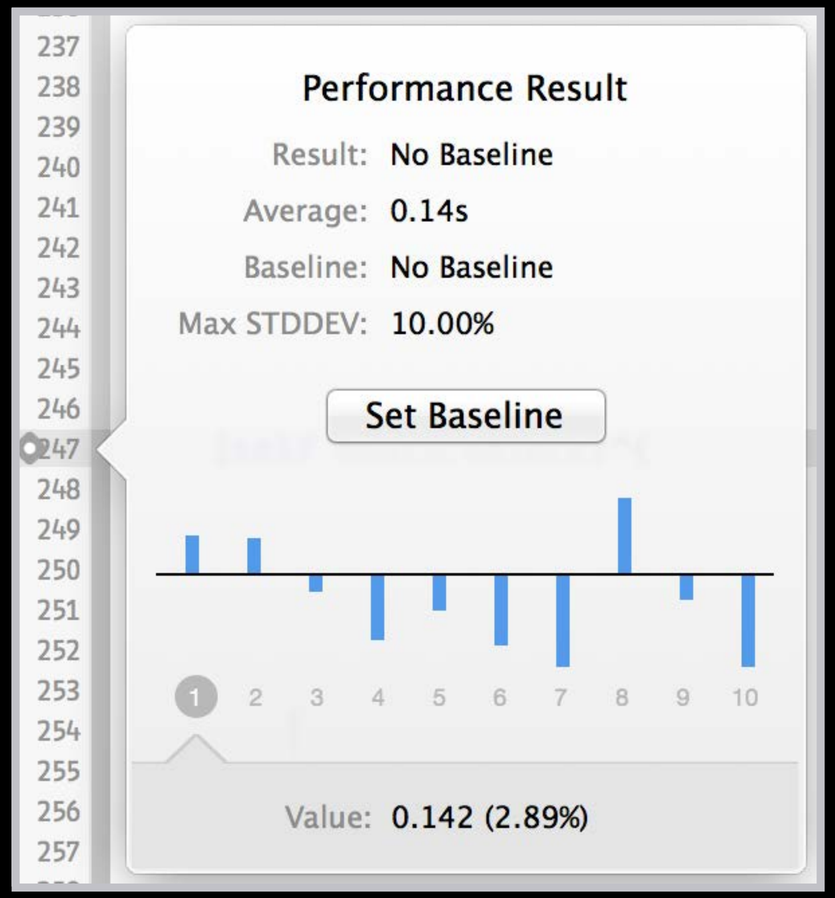
```
71
72      [self measureBlock:^{                                   Time: 0.242 sec (4% STDEV)
73                                                                No baseline average for Time.
74
```

Setting Baselines

Performance Result Popover

- Source Editor
- Test Report

Shows values offset from average

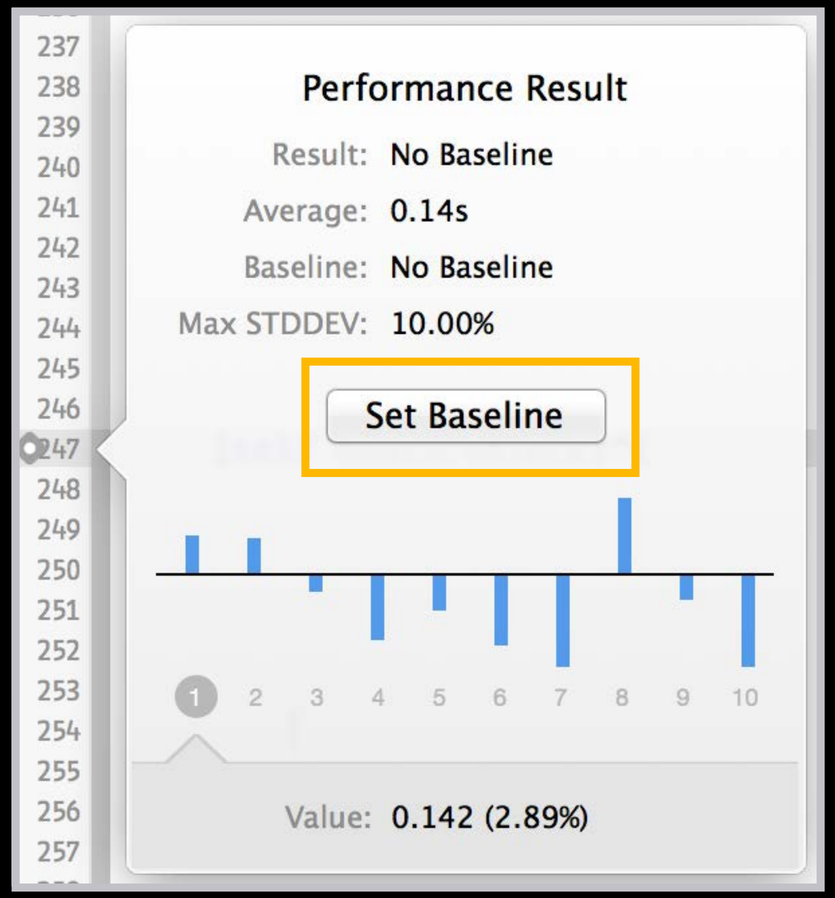


Setting Baselines

Performance Result Popover

- Source Editor
- Test Report

Set Baseline Average

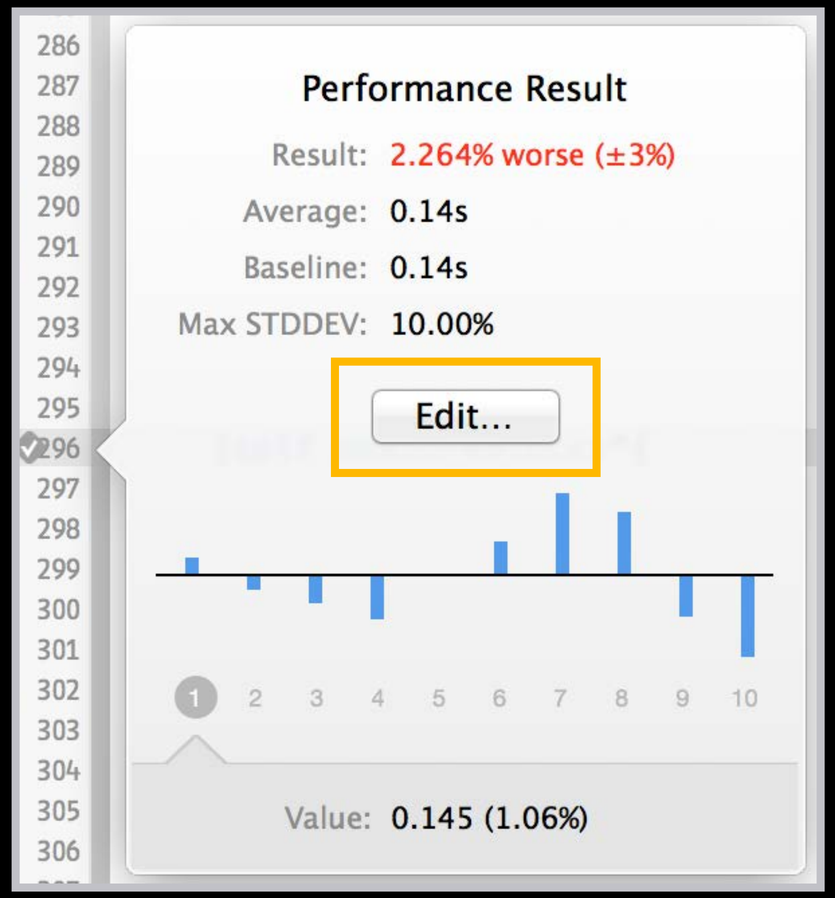


Setting Baselines

Performance Result Popover

- Source Editor
- Test Report

Edit Baseline and STDDEV

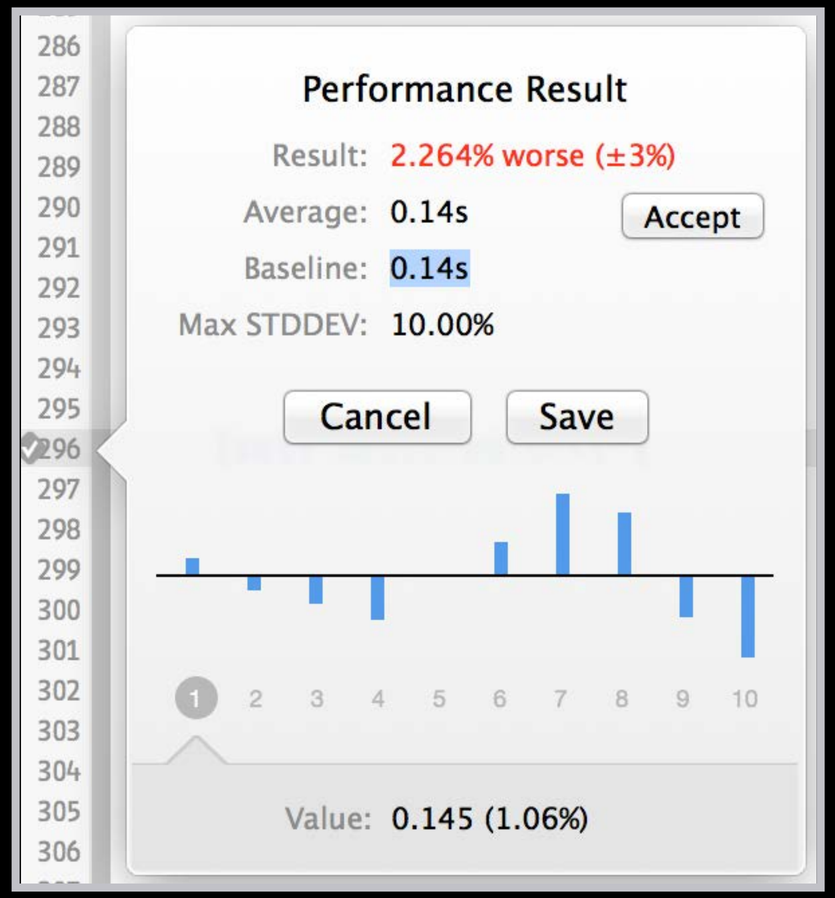


Setting Baselines

Performance Result Popover

- Source Editor
- Test Report

Edit Baseline and STDDEV



Using Baseline Average

Source Editor annotations

- Has Baseline: Passed

```
71
72      [self measureBlock:^{                                   Time: 0.249 sec (4% worse, 7% STDEV)
73
74
```

Using Baseline Average

Source Editor annotations

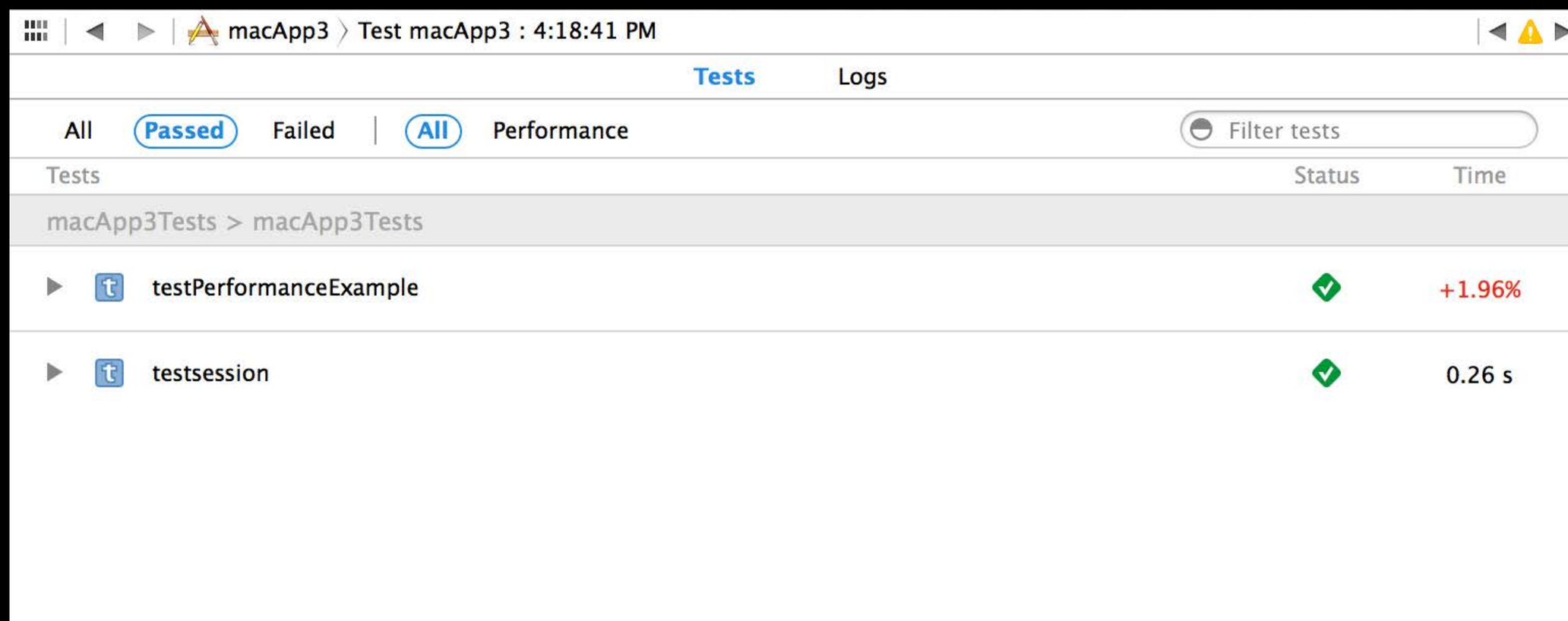
- Has Baseline: Failed

```
71
72      [self measureBlock:^{
73
74
```

Time average is 68% worse (max allowed: 10%).
Time: 0.402 sec (67% worse, 4% STDEV)

Using Baseline Average

Test Report



Using Baseline Average

Test Log

macApp3 〉 Test macApp3 : 4:18:41 PM

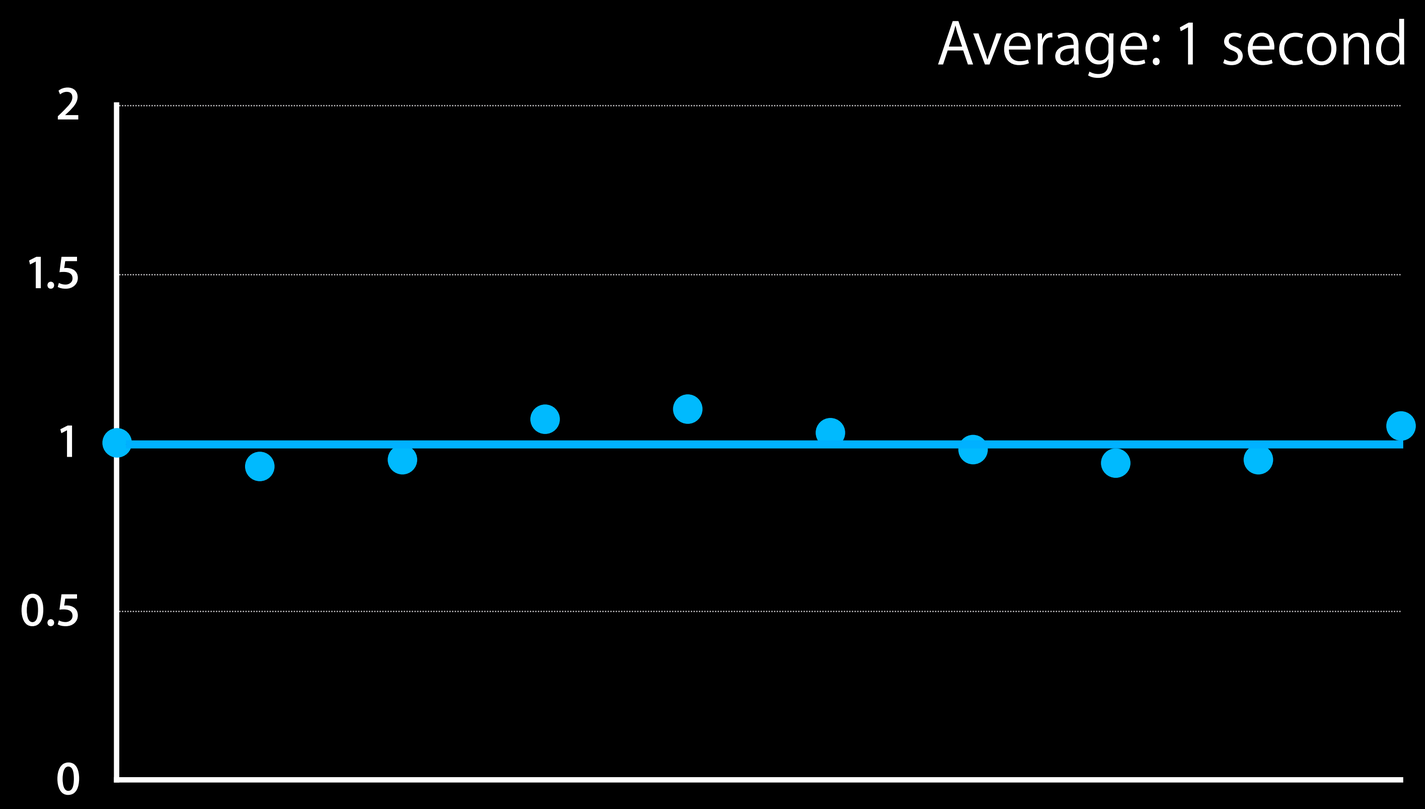
Tests Logs

All Recent | All Messages All Issues Errors Only

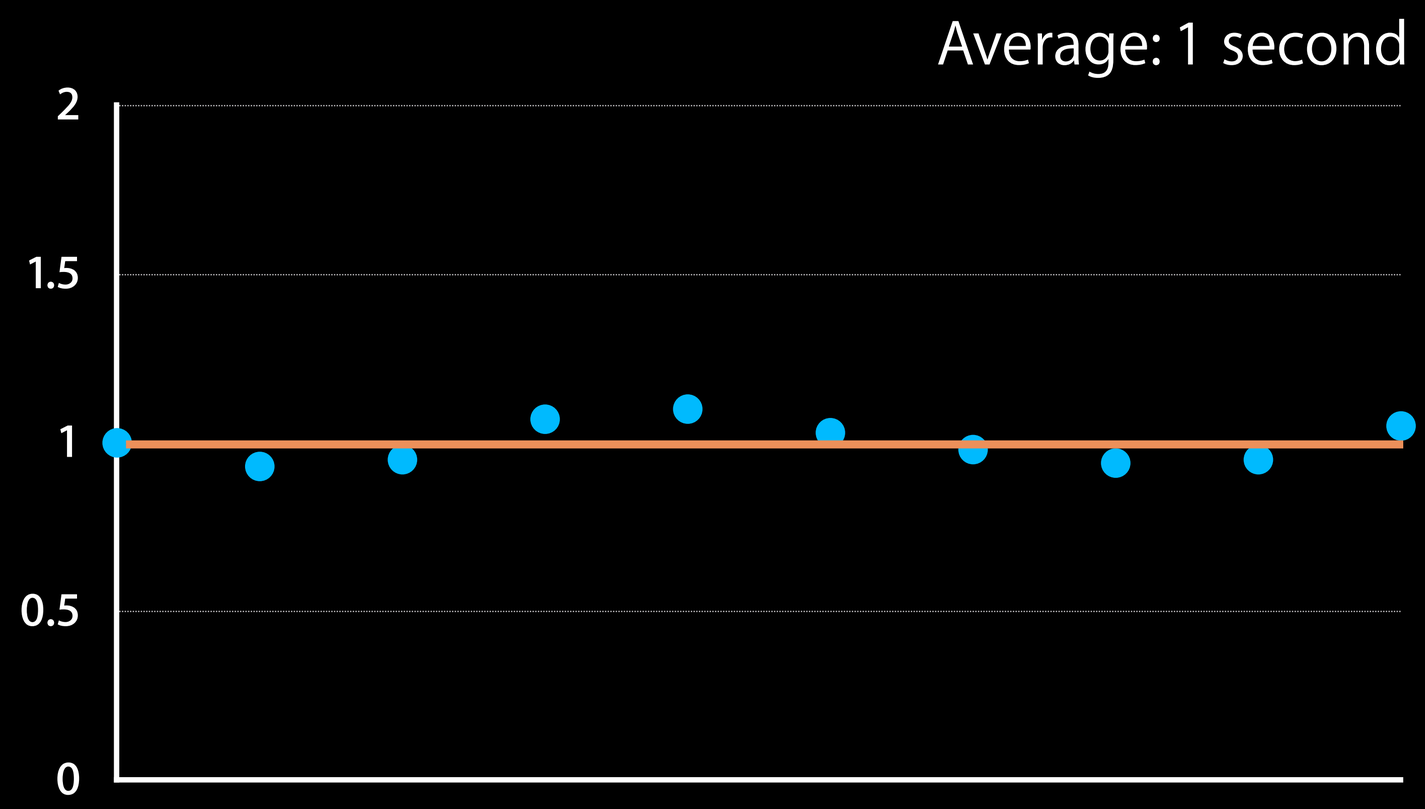
Test target macApp3Tests
Scheme macApp3 | Destination My Mac 64-bit
Run test suite All tests 2 out of 2 tests passed, 4.241 seconds
Run test suite macApp3Tests.xctest 2 out of 2 tests passed, 4.241 seconds
Run test suite macApp3Tests 2 out of 2 tests passed, 4.241 seconds
Run test case testPerformanceExample 1.357 seconds – [Time] average: 0.102 seconds, %RSD: 0.4%

```
Test Case '-[macApp3Tests testPerformanceExample]' started.
Test Case '-[macApp3Tests testPerformanceExample]' started.
/Users/bcallaha/Desktop/macApp3/macApp3Tests/macApp3Tests.m:57: Test Case '-[macApp3Tests
  testPerformanceExample]' measured [Time, seconds] average: 0.102, relative standard deviation:
  0.396%, values: [0.101267, 0.101385, 0.102166, 0.101393, 0.102161, 0.102251, 0.102259, 0.102260,
  0.102227, 0.102247], performanceMetricID:com.apple.XCTPerformanceMetric_WallClockTime,
  baselineName: "Local Baseline", baselineAverage: 0.100, maxPercentRegression: 10.000%,
  maxPercentRelativeStandardDeviation: 10.000%, maxRegression: 0.100, maxStandardDeviation: 0.100
```

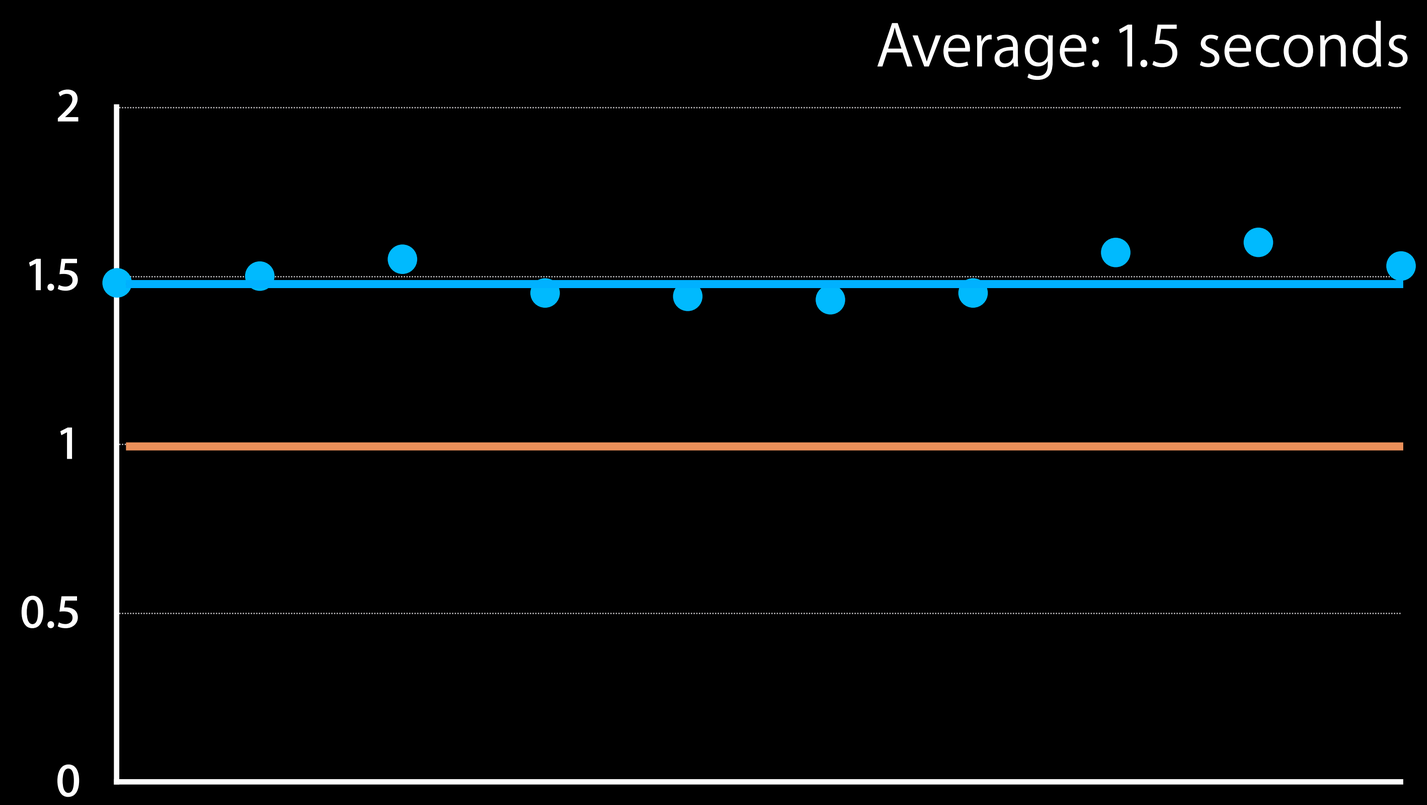
Using Baseline Average



Using Baseline Average



Using Baseline Average

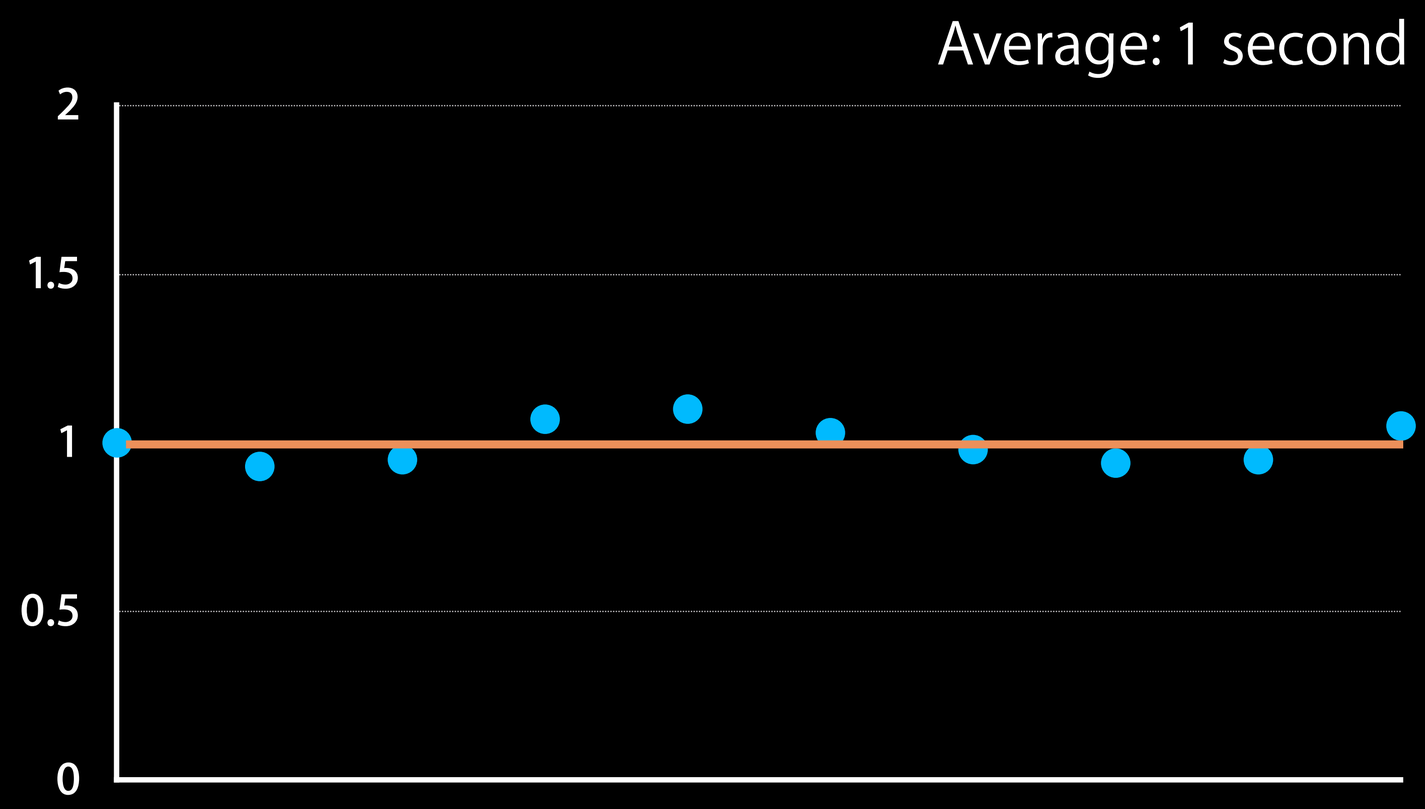


Using Baseline Average

Fail if (Average–Baseline Average) is more than 10% of Baseline Average

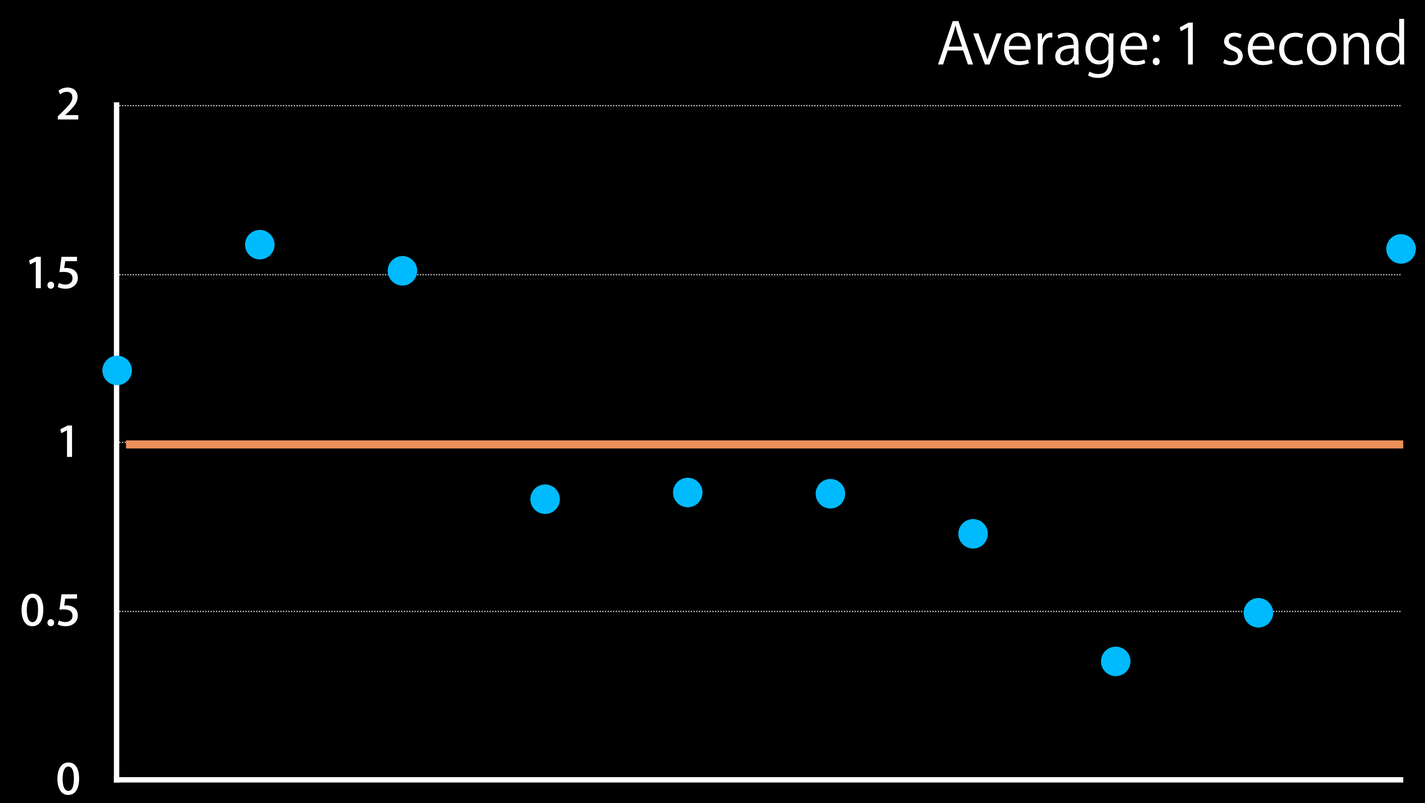
Ignore if (Average–Baseline Average) less than 0.1 seconds

Is Average Enough?



Is Average Enough?

Problem?



Detecting Variance

Standard deviation

Average doesn’t tell the whole story

Detecting Variance

Standard deviation

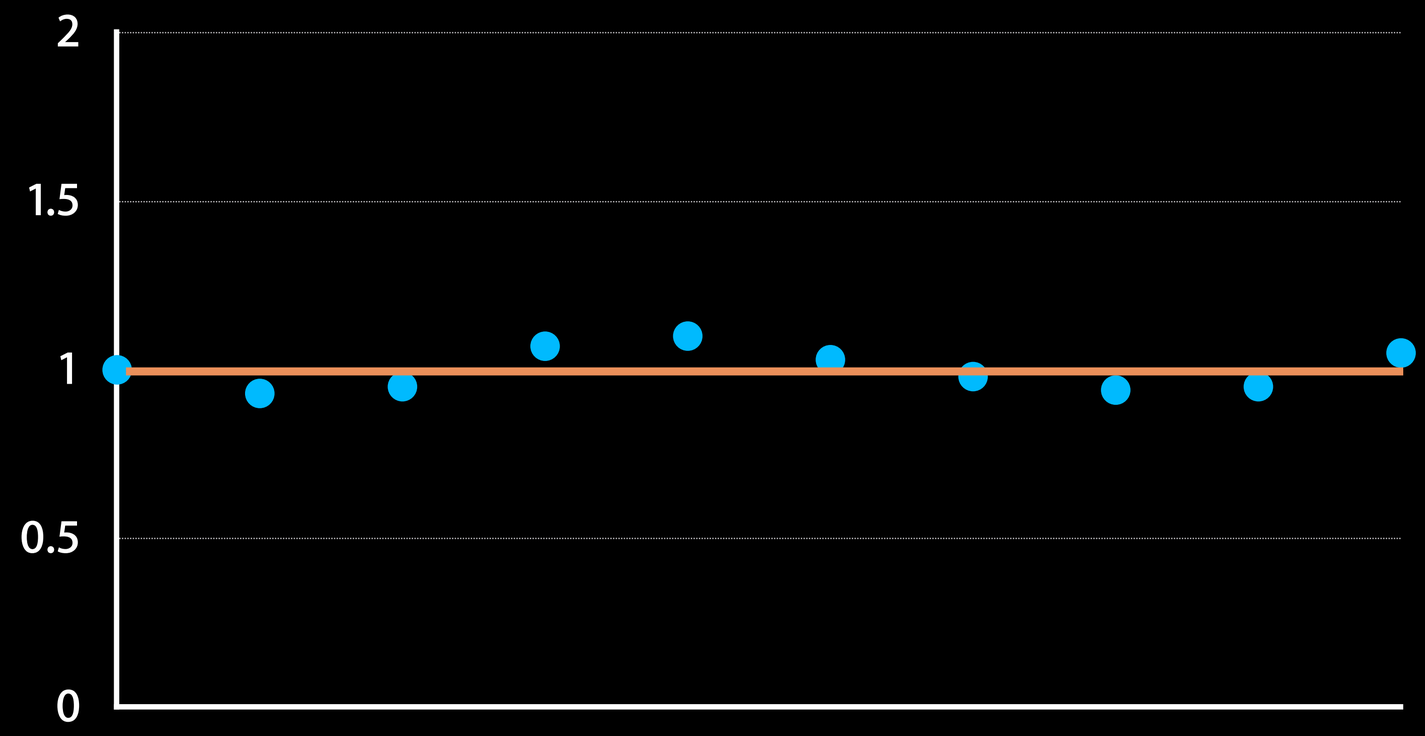
Average doesn’t tell the whole story

Standard deviation indicates spread of measurements

Detecting Variance

Average: 1 second

Standard Deviation: 6%

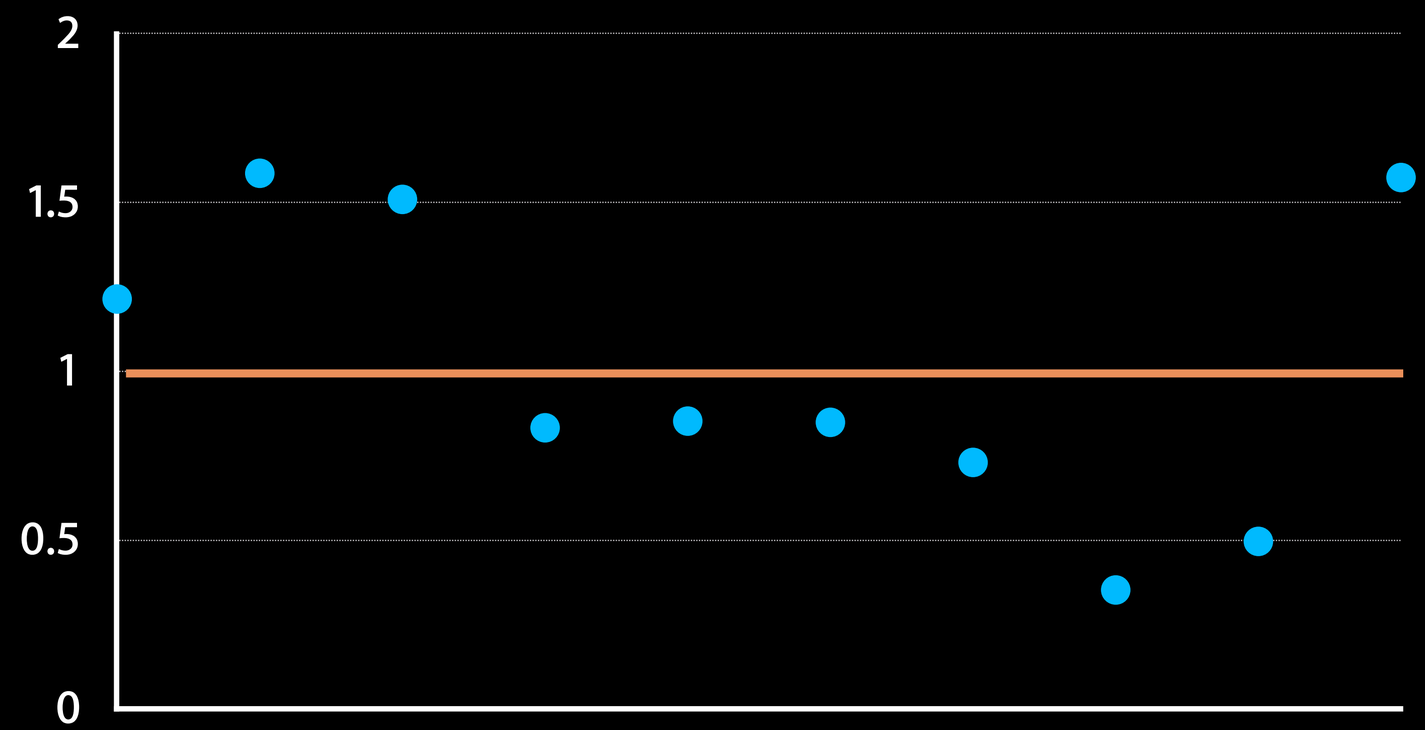


Detecting Variance

Problem?

Average: 1 second

Standard Deviation: 40%



Using Standard Deviation (STDDEV)

Fail if STDDEV is more than 10% of Average (adjustable)

Using Standard Deviation (STDDEV)

Fail if STDDEV is more than 10% of Average (adjustable)

Ignore if STDDEV is less than 0.1 seconds

Excessive STDDEV

Block being measured

Excessive STDDEV

Block being measured

- Does file I/O or network I/O

Excessive STDDEV

Block being measured

- Does file I/O or network I/O
- Doesn’t do the same work each time it’s called

Excessive STDDEV

Block being measured

- Does file I/O or network I/O
- Doesn’t do the same work each time it’s called

System is busy with other processes

Detecting Regressions

1. If test has no Baseline Average, done

Detecting Regressions

1. If test has no Baseline Average, done
2. If STDDEV >0.1 seconds and >10%, fail

Detecting Regressions

1. If test has no Baseline Average, done
2. If STDDEV >0.1 seconds and >10%, fail
3. If (Average–Baseline Average) >0.1 seconds and >10%, fail

Detecting Regressions

1. If test has no Baseline Average, done
2. If STDDEV >0.1 seconds and >10%, fail
3. If (Average–Baseline Average) >0.1 seconds and >10%, fail
4. Else, pass

Measuring Precisely

Only measure code you think that's important to you

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureBlock:^{

        NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
        XCTAssertNotNil(fileHandle);

        UseFileHandle(fileHandle);

        [fileHandle closeFile];
    }];
}
```

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
    XCTAssertNotNil(fileHandle);

    [self measureBlock:^{

        UseFileHandle(fileHandle);

        [fileHandle closeFile];
    }];
}
```

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
    XCTAssertNotNil(fileHandle);

    [self measureBlock:^{

        UseFileHandle(fileHandle);
    }];

   [fileHandle closeFile];
}
```

Measuring Precisely

More XCTestCase APIs

```
- (void)measureMetrics:(NSArray *)metrics automaticallyStartMeasuring:
(BOOL)automaticallyStartMeasuring withBlock:(void (^)(void))block;
```

Use this to measure part of the block

Measuring Precisely

More XCTestCase APIs

```
- (void)measureMetrics:(NSArray *)metrics automaticallyStartMeasuring:
(BOOL)automaticallyStartMeasuring withBlock:(void (^)(void))block;
```

Use this to measure part of the block

Measures passed in metrics

Currently supports one metric: `XCTPerformanceMetric_WallClockTime`

Measuring Precisely

More XCTestCase APIs

```
- (void)startMeasuring;
- (void)stopMeasuring;
```

Isolate part of the block to measure

Measuring Precisely

More XCTestCase APIs

```
- (void)startMeasuring;
- (void)stopMeasuring;
```

Isolate part of the block to measure

May be called once per block invocation

Measuring Precisely

More XCTestCase APIs

```
- (void)startMeasuring;
- (void)stopMeasuring;
```

Isolate part of the block to measure

May be called once per block invocation

`-startMeasuring` requires `automaticallyStartMeasuring:NO`

Measuring Precisely

More XCTestCase APIs

```
- (void)startMeasuring;
- (void)stopMeasuring;
```

Isolate part of the block to measure

May be called once per block invocation

`-startMeasuring` requires `automaticallyStartMeasuring:NO`

`-stopMeasuring` called automatically after block

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureBlock:^{

        NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
        XCTAssertNotNil(fileHandle);

        UseFileHandle(fileHandle);

        [fileHandle closeFile];
    }];
}
```

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureMetrics:@[XCTPerformanceMetric_WallClockTime]
          automaticallyStartMeasuring:NO forBlock:^{

        NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
        XCTAssertNotNil(fileHandle);

        UseFileHandle(fileHandle);

        [fileHandle closeFile];
    }];
}
```

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureMetrics:@[XCTPerformanceMetric_WallClockTime]
          automaticallyStartMeasuring:NO forBlock:^{

        NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
        XCTAssertNotNil(fileHandle);

        [self startMeasuring];
        UseFileHandle(fileHandle);
        [self stopMeasuring];

        [fileHandle closeFile];
    }];
}
```

Measuring Precisely

Example

```
- (void)testUseFileHandlePerformance
{
    [self measureMetrics:@[XCTPerformanceMetric_WallClockTime]
          automaticallyStartMeasuring:NO forBlock:^{

          NSFileHandle *fileHandle = [NSFileHandle fileHandleForReadingAtPath:PATH];
          XCTAssertNotNil(fileHandle);

          [self startMeasuring];
          UseFileHandle(fileHandle);
          [self stopMeasuring];

          [fileHandle closeFile];
    }];
}
```

Demo

Adding performance tests

Performance Testing

Wrap-Up

Use new APIs to measure performance

Performance Testing

Wrap-Up

Use new APIs to measure performance

Set Baseline to detect regressions

Performance Testing

Wrap-Up

Use new APIs to measure performance

Set Baseline to detect regressions

Use Standard Deviation to show spread of measurements

Performance Testing

Wrap-Up

Use new APIs to measure performance

Set Baseline to detect regressions

Use Standard Deviation to show spread of measurements

Use Instruments to profile tests

Testing in Xcode 6

Testing in Xcode 6

Why test?

Testing in Xcode 6

Why test?

Adding tests

Testing in Xcode 6

Why test?

Adding tests

Asynchronous testing

Testing in Xcode 6

Why test?

Adding tests

Asynchronous testing

Performance testing

More Information

Dave DeLong
Developer Tools Evangelist
delong@apple.com

Related Sessions

• Continuous Integration with Xcode 6	Marina	Thursday 2:00PM

Labs

• Continuous Integration Lab	Tools Lab C	Thursday 2:00PM

WWDC14