Unit and Functional Testing for the iOS Platform

Christopher M. Judd

Judd Solutions

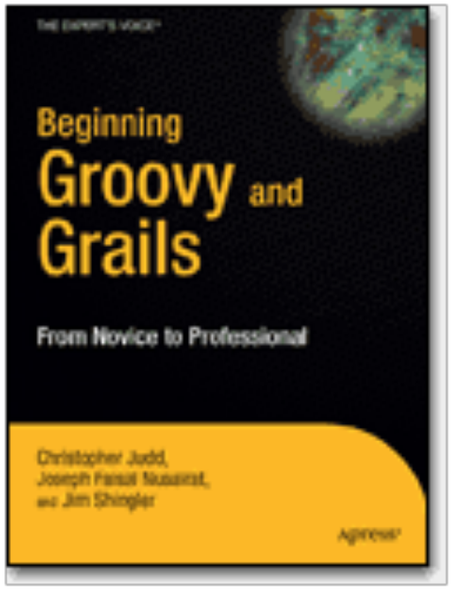
Christopher M. Judd

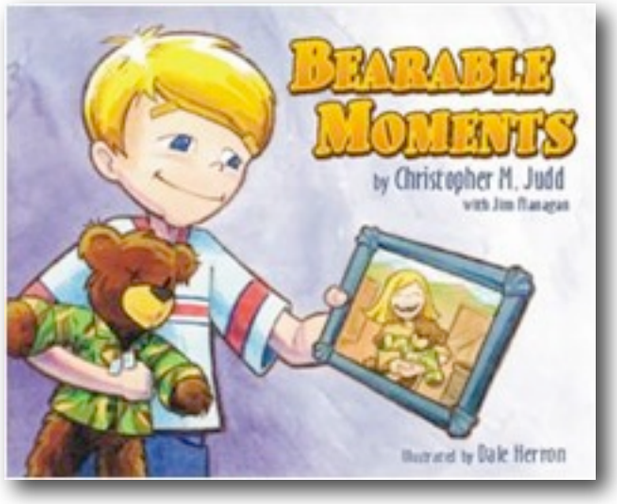
President/Consultant of **Judd Solutions**

Central Ohio Java Users Group leader

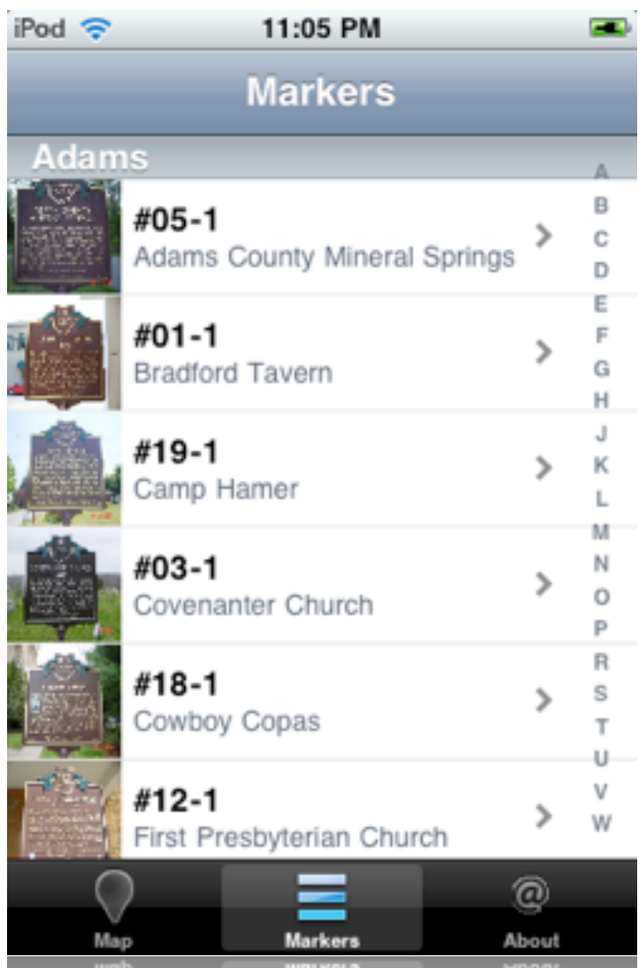
Columbus 

Developer User Group (CIDUG)





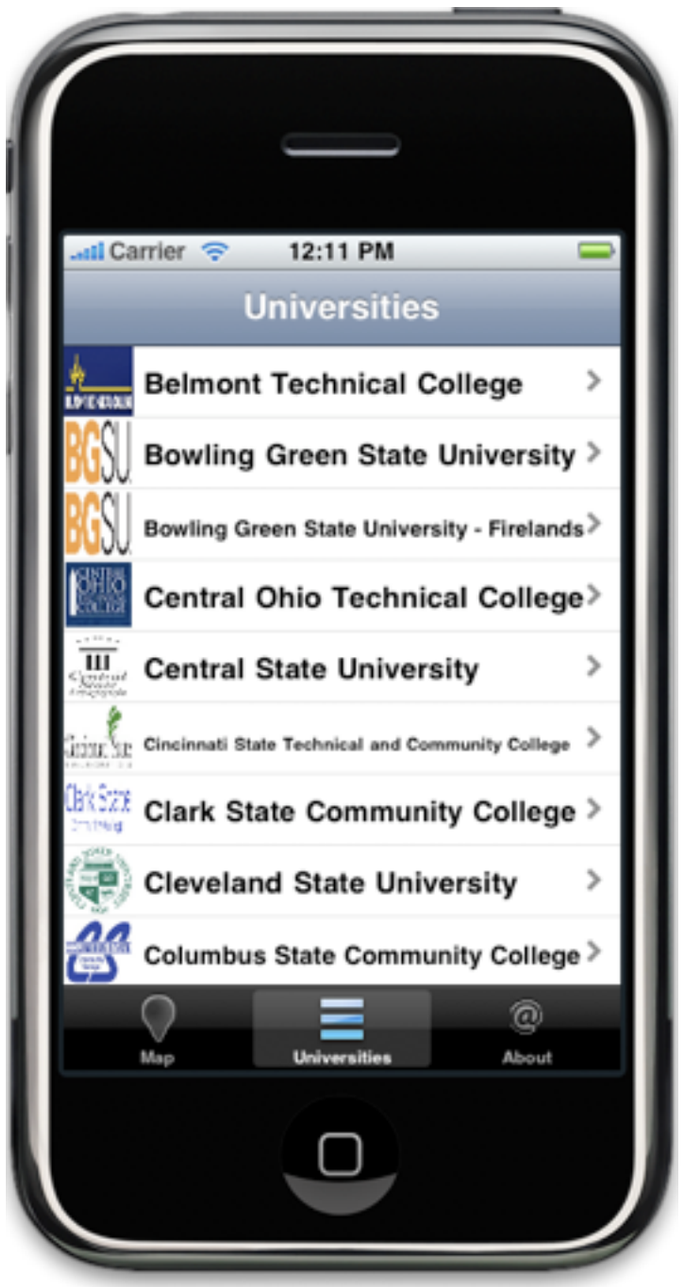
Remarkable Ohio

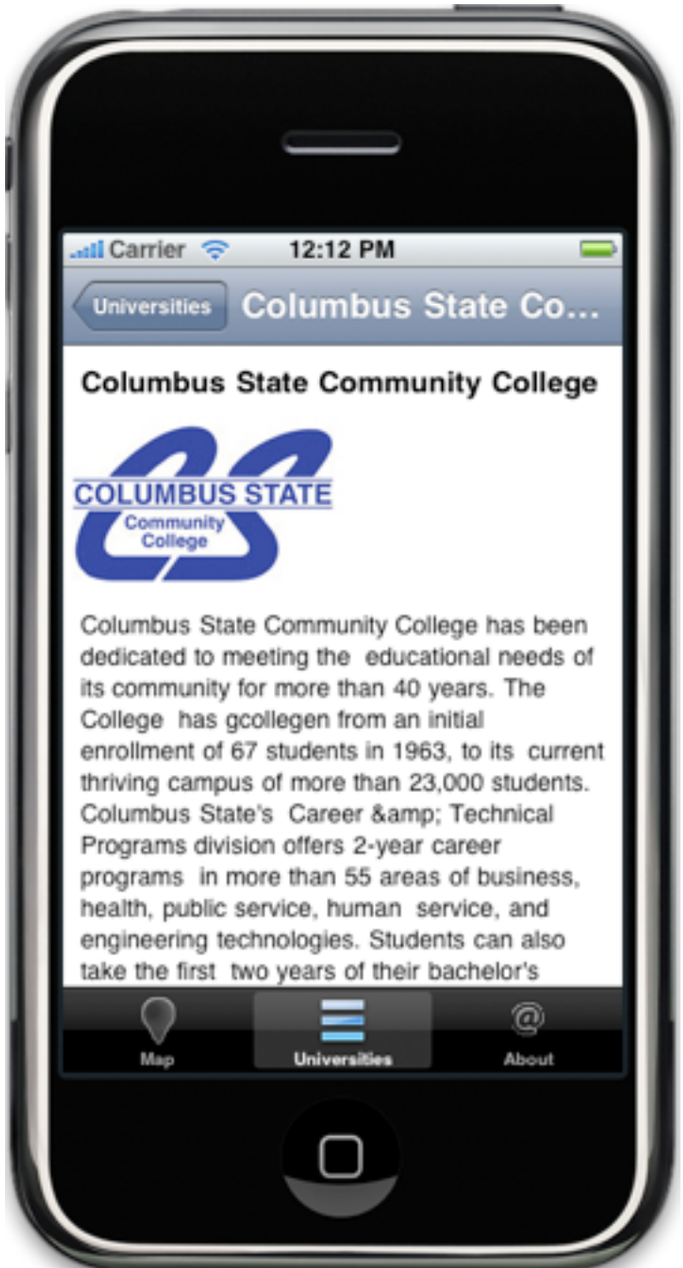


Free

Developed for eTech Ohio and Ohio Historical Center

University System Of Ohio





Free

Developed for eTech Ohio and University System Of Ohio

How many of you are currently or have developed applications for the iOS Platform?

How many of you have ever unit or functionally tested your iOS application?

How many of you have ever unit tested on another platform or language?

WHY AREN'T YOU TESTING YOUR IOS APPLICATIONS?

iOS unit and functional testing is neither straight forward or easy

iOS Unit Testing Challenges

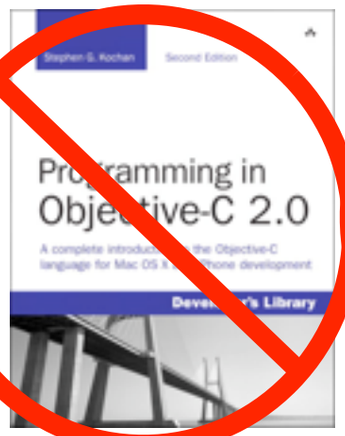
- Toooo many steps to get started
- No red bar/green bar
- Can't run individual tests
- Unit test template code contains to much
- Unit tests can not be debugged out of the box
- Little documentation
- Multiple targets
- Only includes very basic testing support
  - No mock framework included

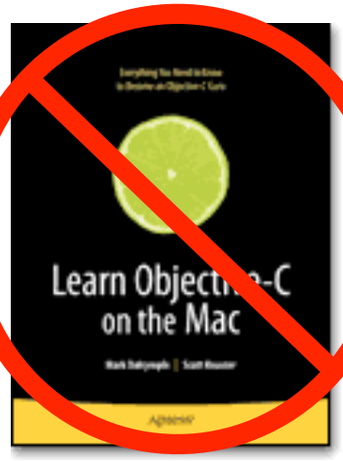
Good News

Xcode 4 is so much easier

iOS Functional Testing Challenges

Testing Coverage





The Objective-C Programming Language

STANFORD
UNIVERSITY

iPhone
Application
Development

WINTER 2010

Unit Testing

Unit Testing Basics

Why Unit Test?

- Improves design
- Facility change and refactoring
- Simplifies integration
- Provides executable documentation

includes

OCUnit

also known as

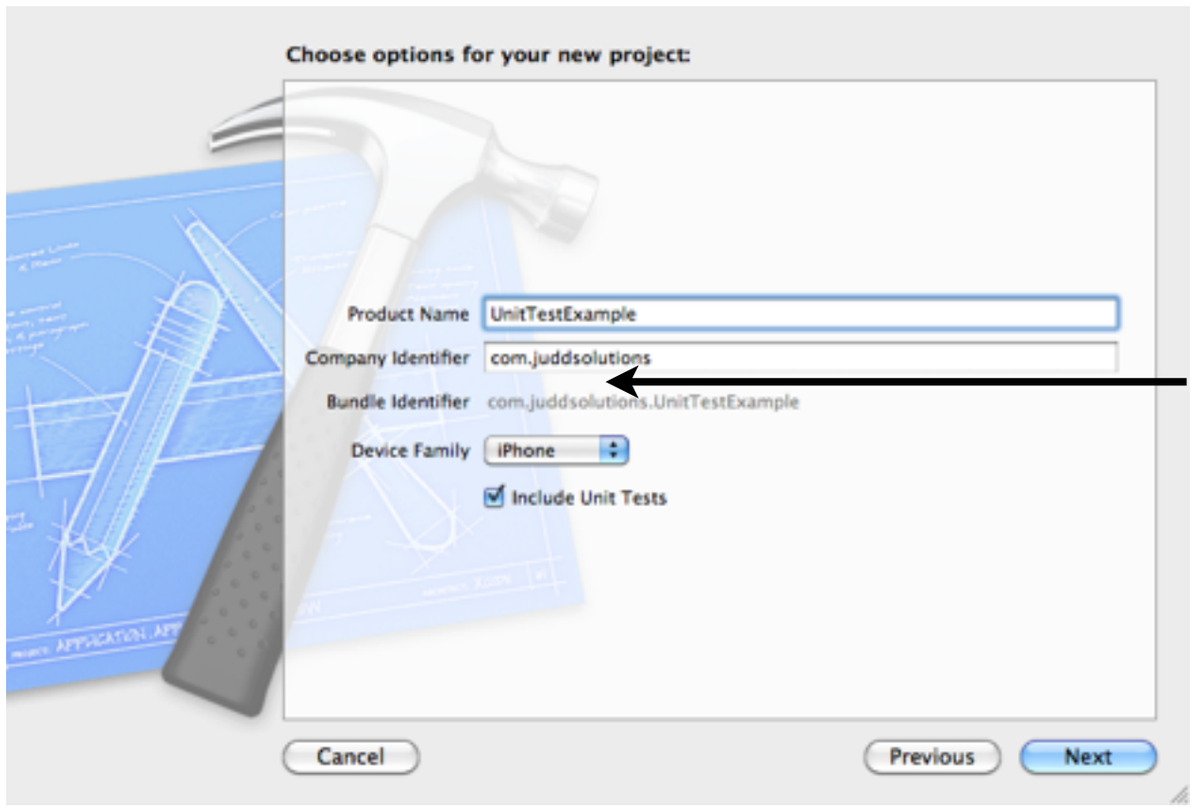
SenTestingKit

Logic Test == Unit Test

Application Test == More Functional Test

Getting Started

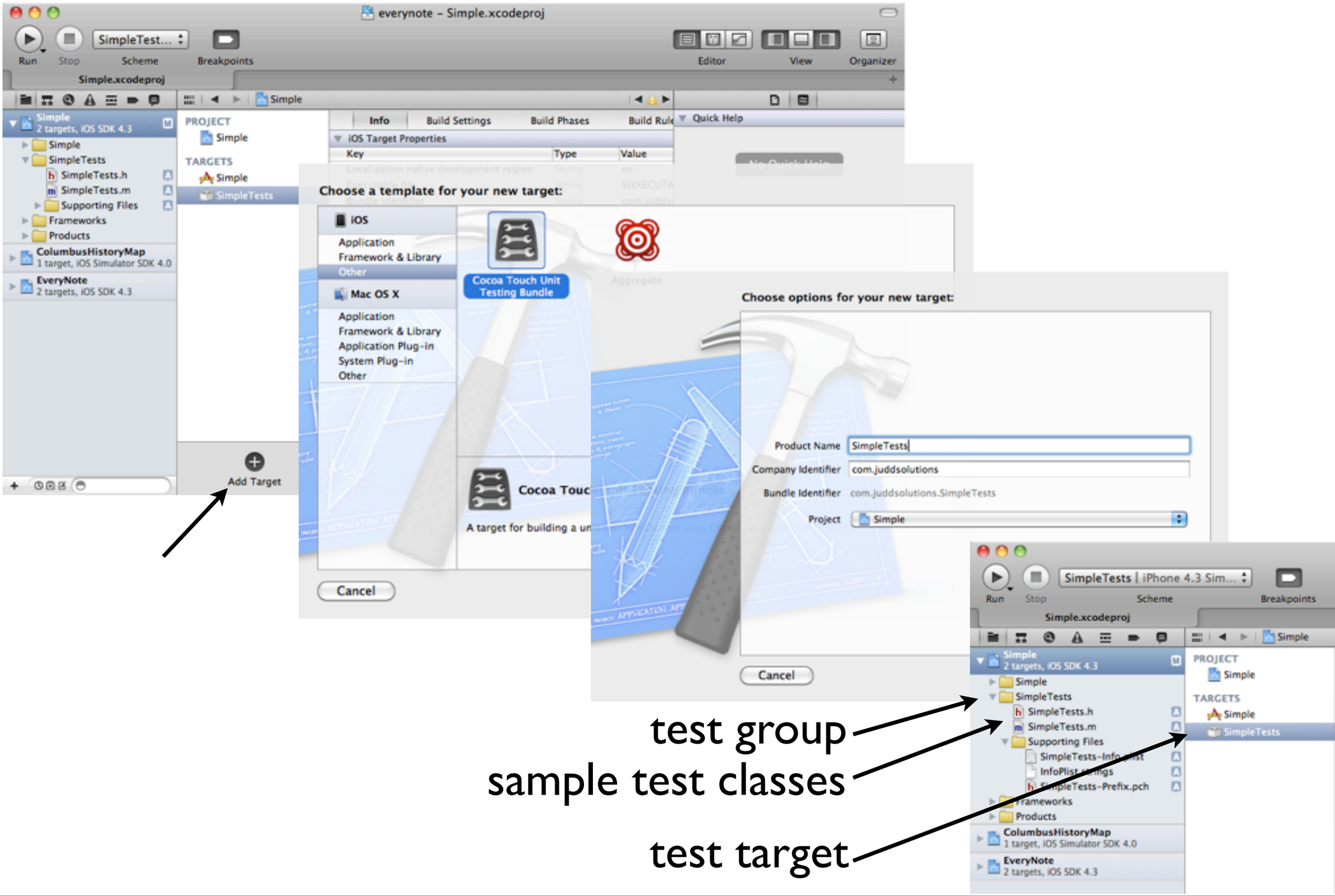
Unit Testing Setup Steps



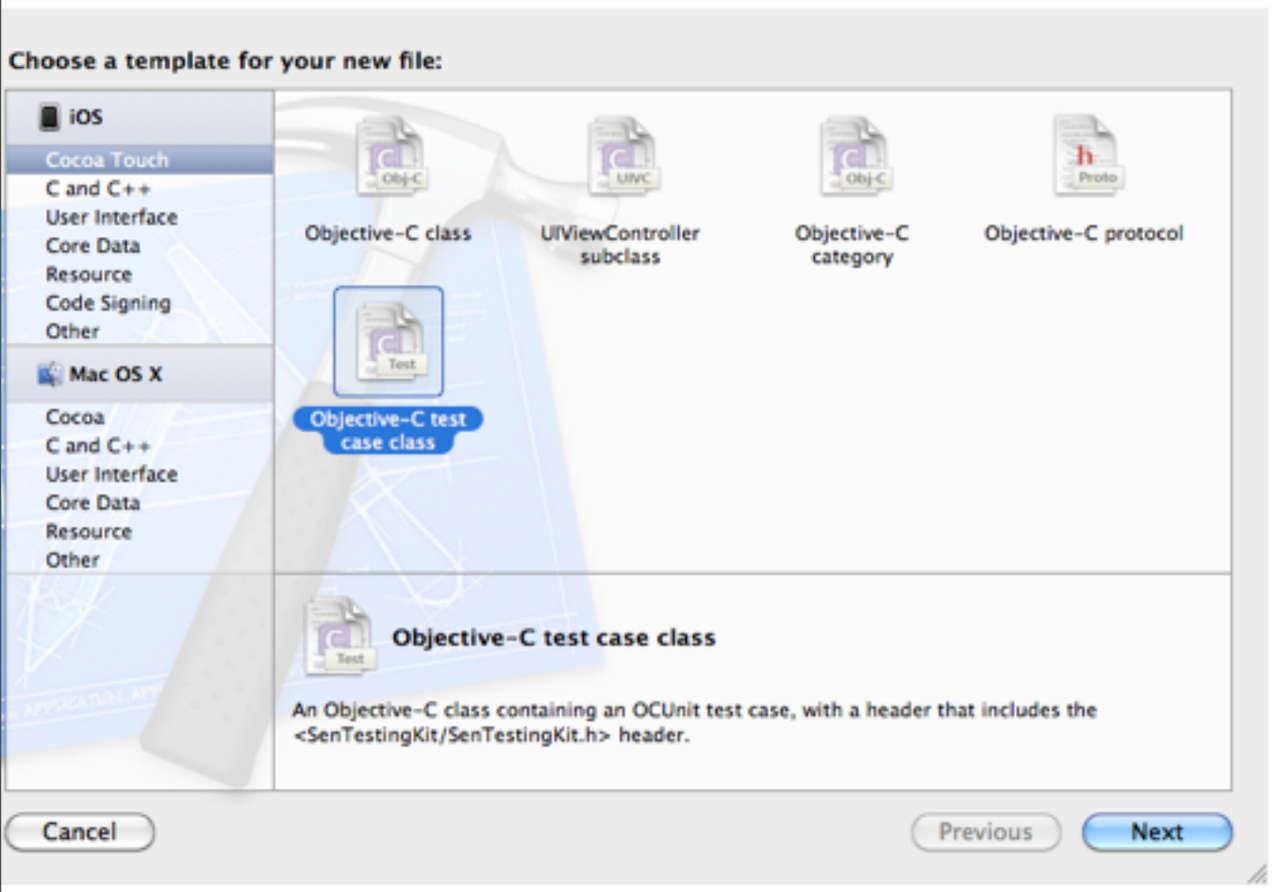
or

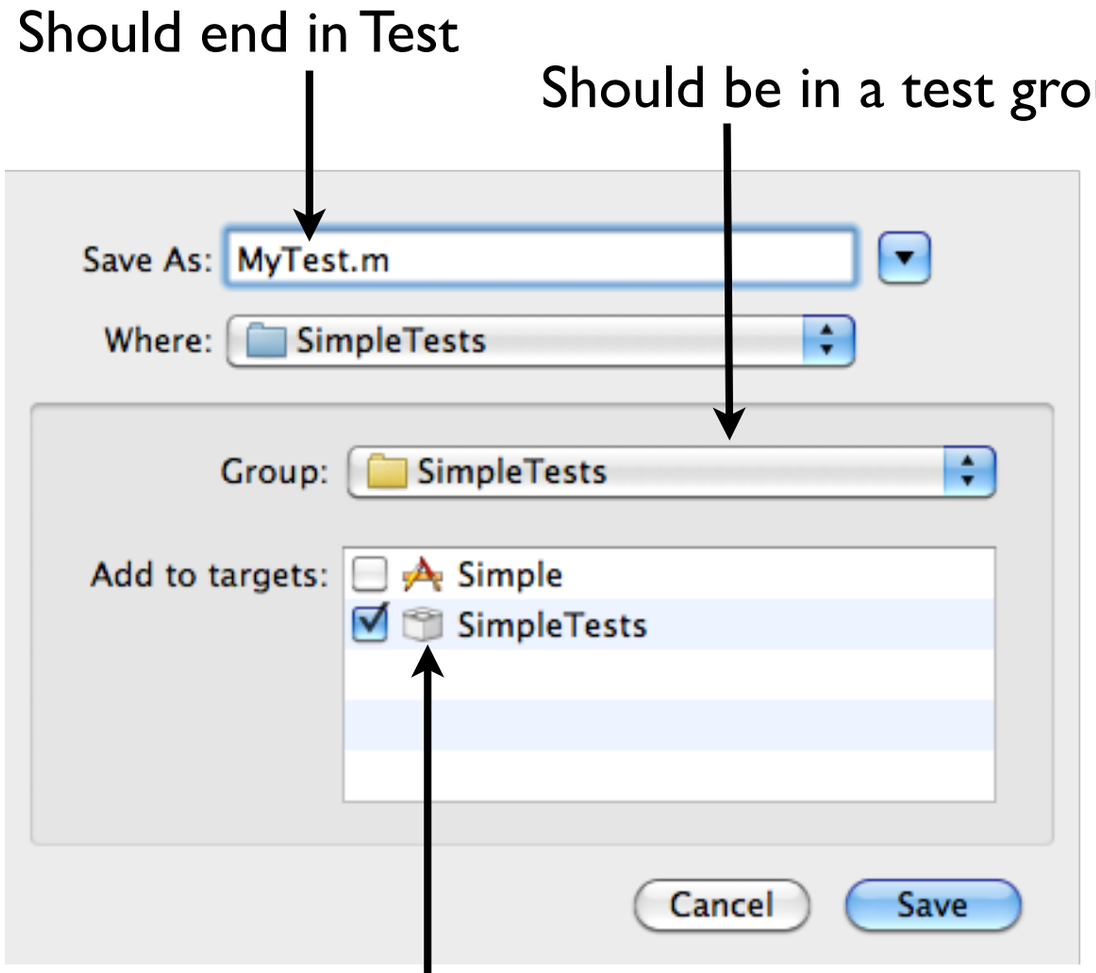
1. Add Target
2. Cocoa Touch Unit Testing Bundle

Add Unit Test Bundle Target



Add Unit Test Class





Generated Test Class

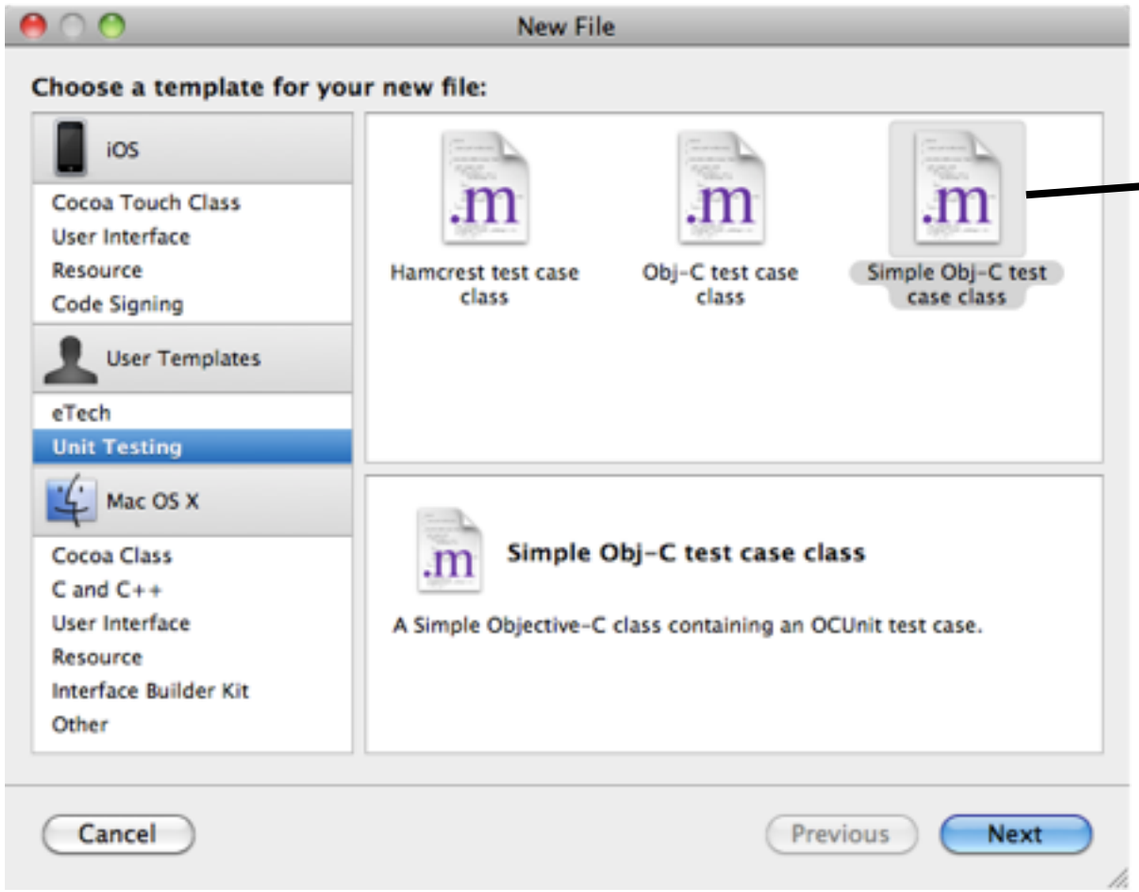
*Test.h

```
#define USE_APPLICATION_UNIT_TEST 1

#import <SenTestingKit/SenTestingKit.h>
#import <UIKit/UIKit.h>
//#import "application_headers" as required

@interface SimpleTest : SenTestCase {
}

#if USE_APPLICATION_UNIT_TEST
- (void) testAppDelegate; // simple test on application
#else
- (void) testMath; // simple standalone test
#endif

@end
```

*Test.m

```
#import "SimpleTest.h"

@implementation SimpleTest

#if USE_APPLICATION_UNIT_TEST  // all code under test is in the iPhone Application

- (void) testAppDelegate {
    id yourApplicationDelegate = [[UIApplication sharedApplication] delegate];
    STAssertNotNil(yourApplicationDelegate,
                   @"UIApplication failed to find the AppDelegate");
}

#else      // all code under test must be linked into the Unit Test bundle

- (void) testMath {
    STAssertTrue((1+1)==2, @"Compiler isn't feeling well today :-(" );
}

#endif
@end
```

Create Custom User Templates

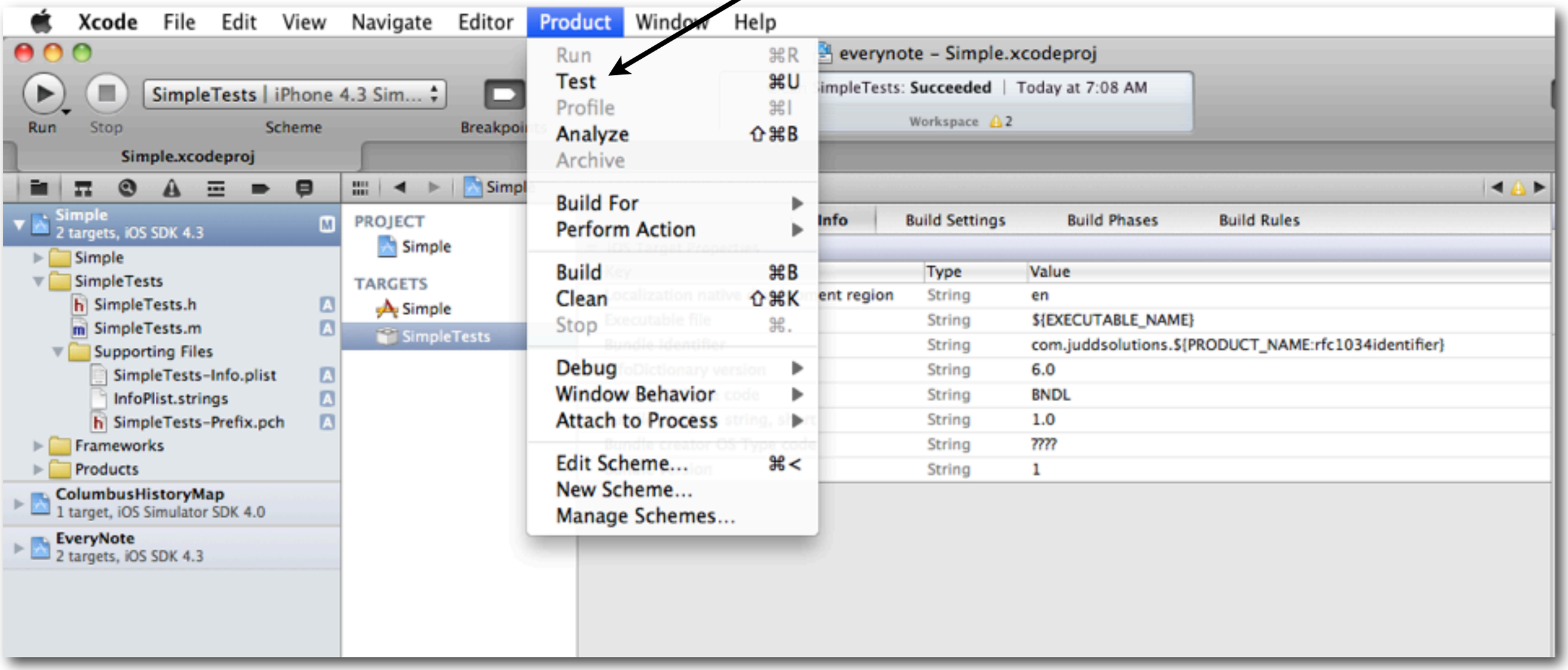


*Test.m

```
#import <SenTestingKit/SenTestingKit.h>

@interface SimpleTests : SenTestCase {

}

@end

@implementation SimpleTests

- (void) testMethod {

    STAssertTrue(true, @"message");

}

@end
```

Running Unit Tests

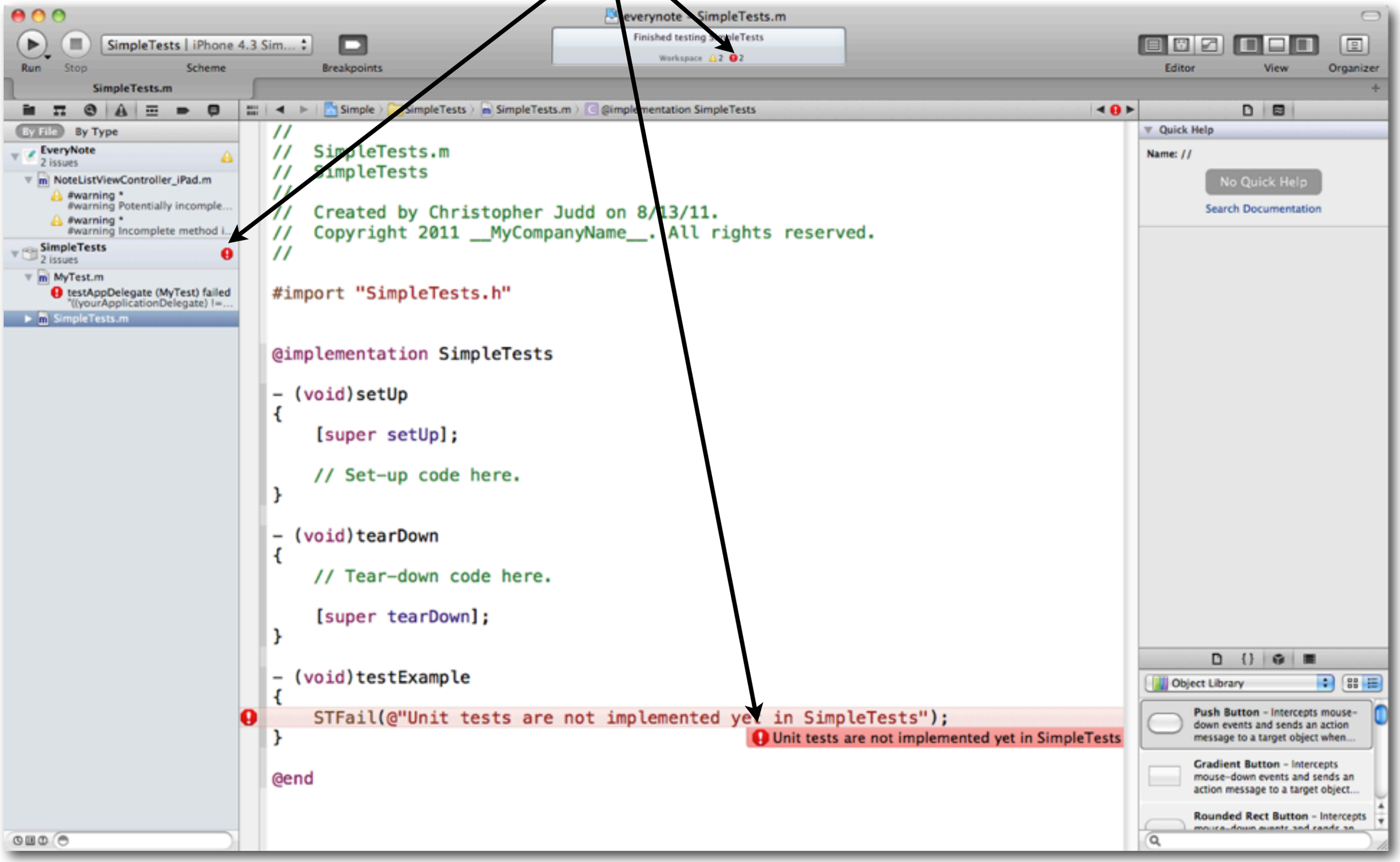
Running Tests



or

⌘U

Red Bar



Still no Green Bar :(

Writing Unit Tests

Basic Unit Test

*Test.m

```
#import <SenTestingKit/SenTestingKit.h>

@interface MyUnitTest : SenTestCase {

}

@end@implementation MyUnitTest

- (void) setUp {
    // initialize tests
}

- (void) testMethod {
    // Test code
}

- (void) tearDown {
    // clean up after test
}


@end
```

must extend SenTestCase

must return void and take no parameters

test methods must begin with test

Assert Macros

```
STAssertNil(a1, description, ...)
STAssertNotNil(a1, description, ...)
STAssertTrue(expression, description, ...)
STAssertFalse(expression, description, ...)
STAssertEqualObjects(a1, a2, description, ...)
STAssertEquals(a1, a2, description, ...)
STAssertEqualsWithAccuracy(left, right, accuracy, description, ...)
STAssertThrows(expression, description, ...)
STAssertThrowsSpecific(expression, specificException, description, ...)
STAssertThrowsSpecificNamed(expr, specificException, aName, description, ...)
STAssertNoThrow(expression, description, ...)
STAssertNoThrowSpecific(expression, specificException, description, ...)
STAssertNoThrowSpecificNamed(expr, specificException, aName, description, ...)
STFail(description, ...)
STAssertTrueNoThrow(expression, description, ...)
STAssertFalseNoThrow(expression, description, ...)
```

Basic Unit Test Example

*Test.m

```
#import <Foundation/Foundation.h>
#import <SenTestingKit/SenTestingKit.h>

@interface MyUnitTest : SenTestCase {
   NSMutableArray* _array;
}

@end

@implementation MyUnitTest

- (void) setUp {
   _array = [[NSMutableArray arrayWithObjects:@"1", @"2", @"3", nil] retain];
}

- (void) testSize {
   STAssertEquals(3, (int)[_array count], @"Size must be three.");
}

- (void) testIndex {
   STAssertEqualObjects(@"2", [_array objectAtIndex:1], @"Must retrieve object at index of 1.");
}

- (void) tearDown {
   [_array release];
}

@end
```

Additional Unit Testing Thoughts

OC Mock

http://www.mulle-kybernetik.com/software/OCMock/

```
- (void)testReturnsStubbedReturnValue
{
    id returnValue;

    [[[mock stub] andReturn:@"megamock"] lowercaseString];
    returnValue = [mock lowercaseString];

    STAssertEqualObjects(@"megamock", returnValue, @"Should have returned stubbed value.");
}
```

hamcrest http://code.google.com/p/hamcrest/

```
#import <SenTestingKit/SenTestingKit.h>

#define HC_SHORTHAND
#import <OCHamcrestIOS/OCHamcrestIOS.h>

@interface Hamcrest : SenTestCase {}
@end

@implementation Hamcrest

- (void) testMethod {
    assertThat(@"cool", equalTo(@"cool"));
}

@end
```

uispec

Behavior Driven Development for the iPhone

http://code.google.com/p/uispec/

```
-(void)itShouldHaveDefaultUsers {
        //Check that all default users are in list
        [[app.tableView.label text:@"Larry Stooge"] should].exist;
        [[app.tableView.label text:@"Curly Stooge"] should].exist;
        [[app.tableView.label text:@"Moe Stooge"] should].exist;
}
```

Main Page | Related Pages | Classes | Files

GHUnit

0.4.26

GHUnit is a test framework for Objective-C (Mac OS X 10.5 and above and iPhone 3.x and above). It can be used with SenTestingKit, GTM or all by itself.

For example, your test cases will be run if they subclass any of the following:

- GHTestCase
- SenTestCase
- GTMTestCase

Source: http://github.com/gabriel/gh-unit

View docs online: http://gabriel.github.com/gh-unit/

This manual is divided in the following sections:

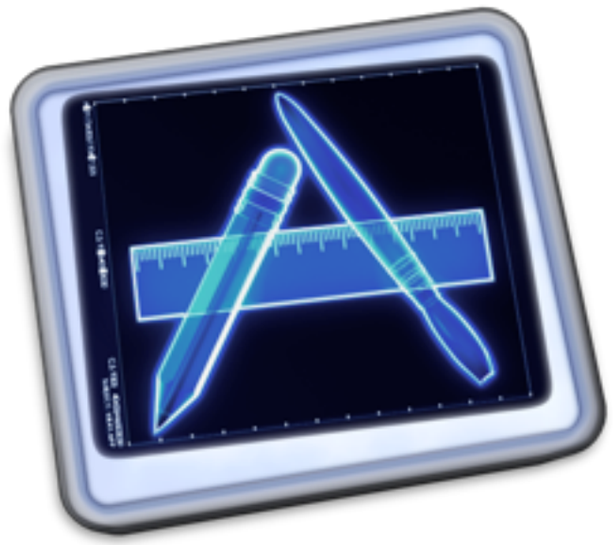
- Installing
- Examples
- Command Line & Makefiles
- Test Macros
- Building
- Environment Variables
- Customizing
- Hudson

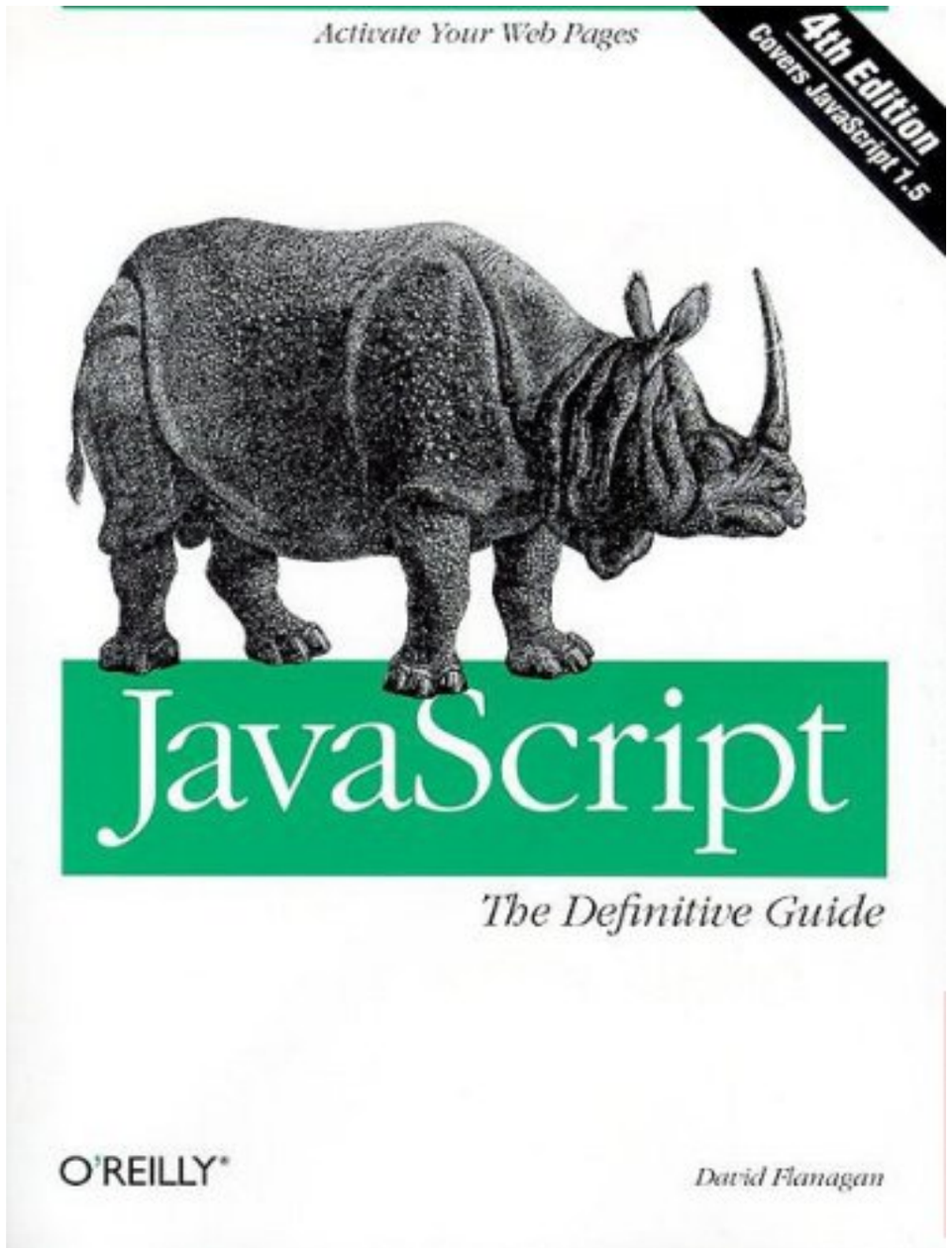
http://gabriel.github.com/gh-unit/

Functional Testing

UI Automation

Included in

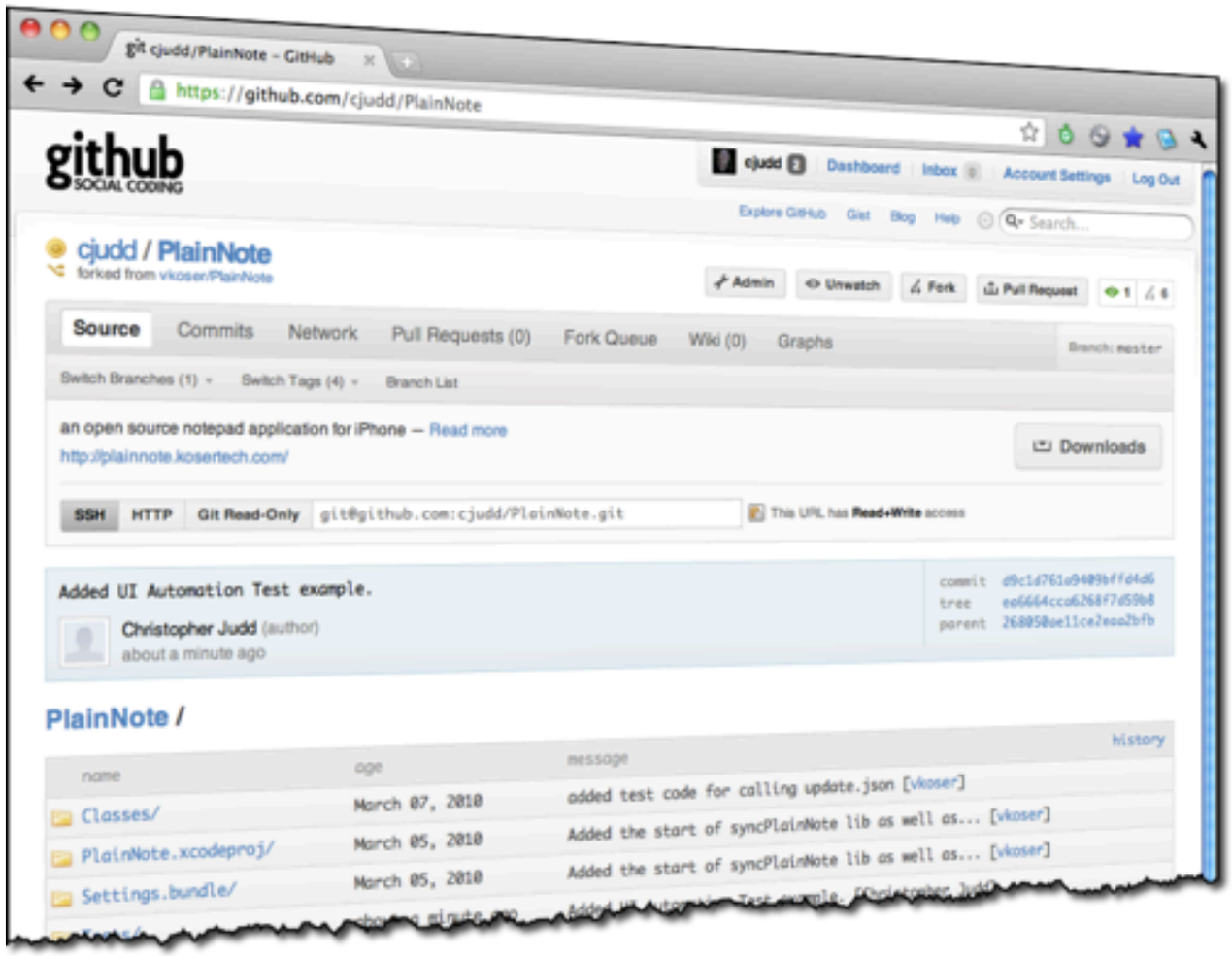


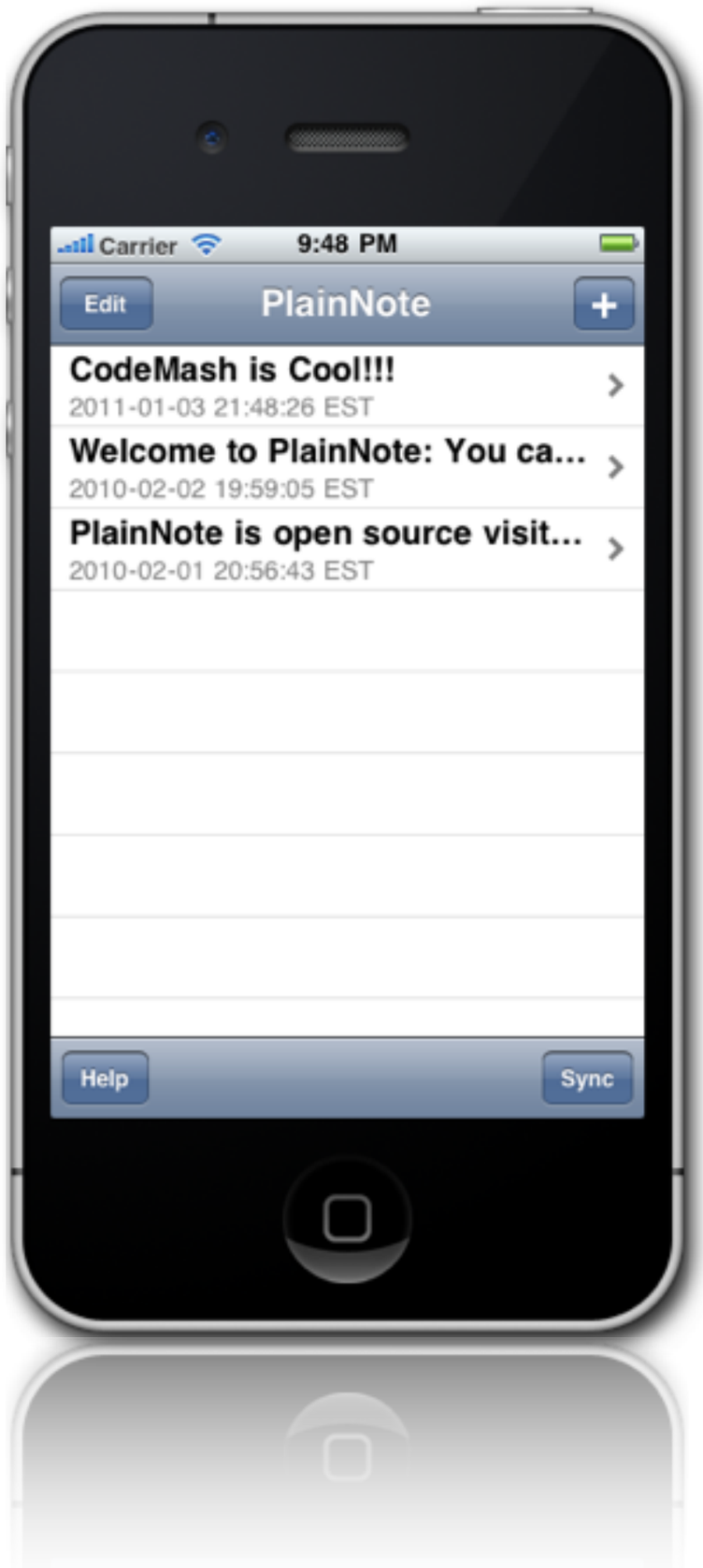

Activate Your Web Pages
4th Edition
Covers JavaScript 1.5
JavaScript
The Definitive Guide
O'REILLY
David Flanagan

Device

Simulator

Example Application



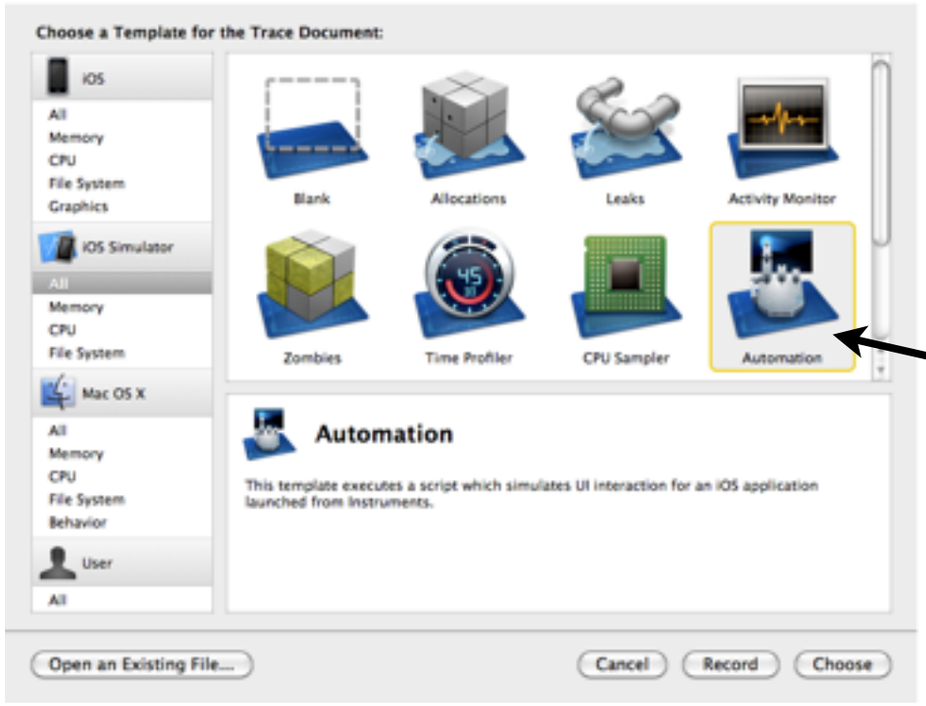


https://github.com/cjudd/PlainNote

derived from

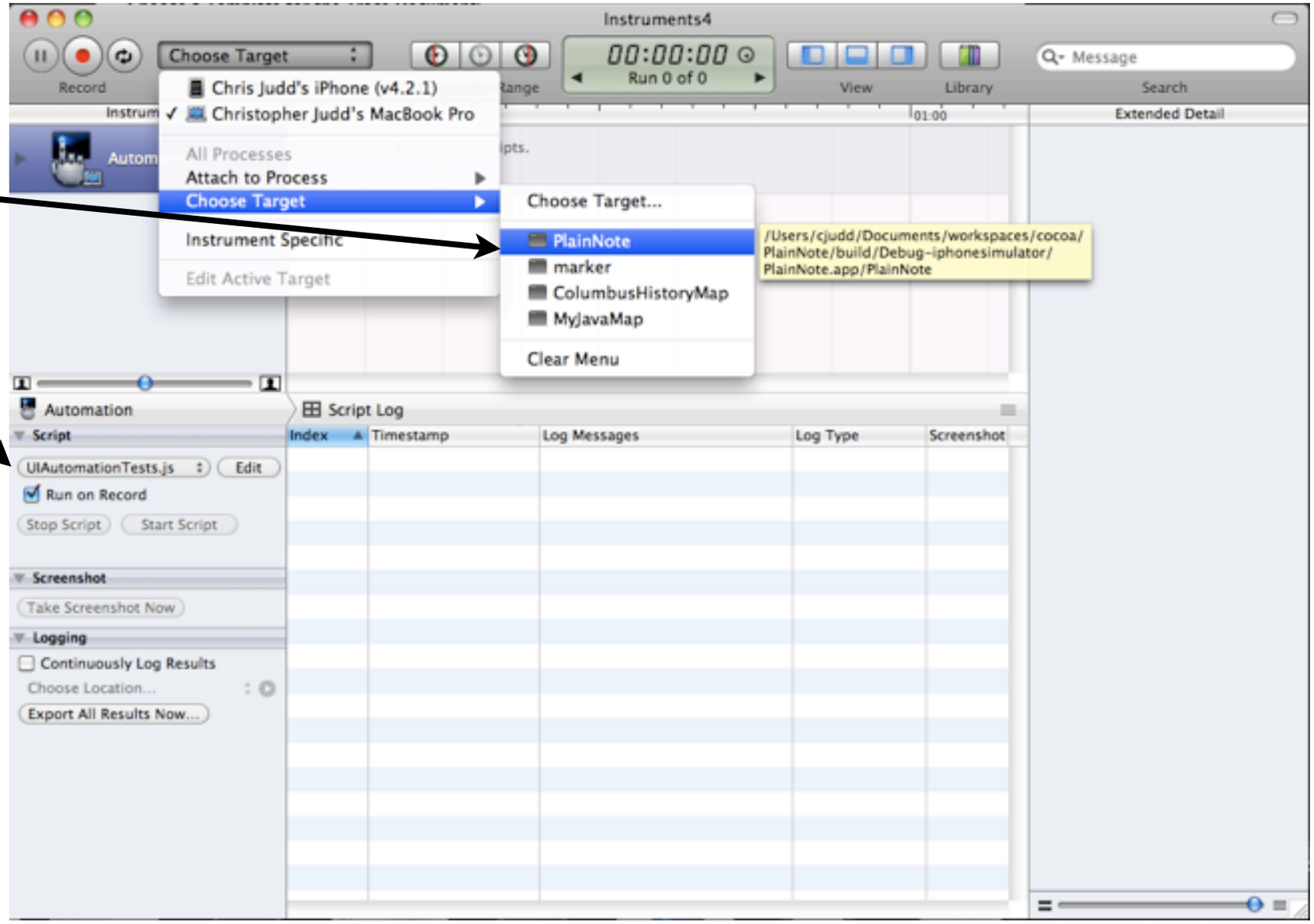
https://github.com/vkoser/PlainNote

Running UI Automation Scripts



1. Start Instruments
2. Select Automation Instrument

3. Select Target App
4. Select Script
5. Record (Play)



Test Results

Instruments1

Record | Target: PlainNote | Inspection Range | 00:00:19, Run 1 of 1 | View | Library | Search: Message

Instruments

Automation

00:00 — 01:00

Extended Detail

Log Data

Error

1/6/11 11:01:32 PM EST

Cannot perform action on invalid element: UIAElementNil from target.frontMostApp().mainWindow().navigationBar().buttons()["Add"]

Screenshot

Automation

Script

UIAutomationTests.js | Edit

Run on Record

Stop Script | Start Script

Script is stopped.

Screenshot

Take Screenshot Now

Logging

Continuously Log Results

Choose Location...

Export All Results Now...

Script Log

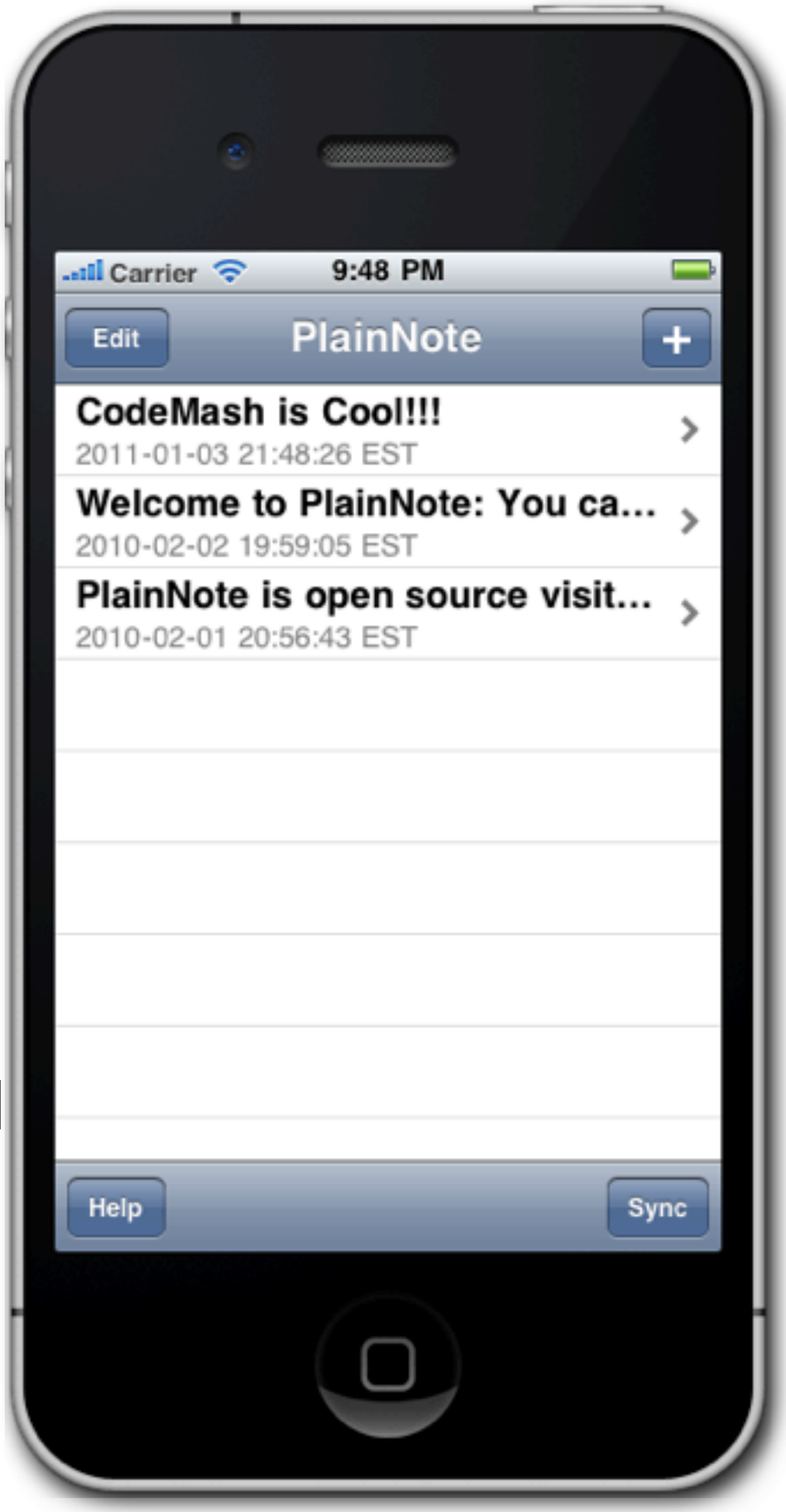
Index	Timestamp	Log Messages	Log Type	Screenshot
0	11:01:24 PM EST	▶ Open Help Test	Pass	
2	11:01:25 PM EST	▼ Add Note Test	Fail	
3	11:01:31 PM EST	Cannot perform action on invalid eleme...	Debug	
4	11:01:32 PM EST	Cannot perform action on invalid eleme...	Error	
5	11:01:32 PM EST	1) UIATarget [name:Chris Judd's iPhone ...	Debug	
6	11:01:32 PM EST	2) UIAApplication [name:PlainNote value...	Debug	
7	11:01:32 PM EST	3) UIAWindow [name:(null) value:(null) r...	Debug	
8	11:01:32 PM EST	4) UIANavigationBar [name:PlainNote val...	Debug	
9	11:01:32 PM EST	5) UIAButton [name:Edit value:(null) rect:...	Debug	
10	11:01:32 PM EST	5) UIAStaticText [name:PlainNote value:(...	Debug	
11	11:01:32 PM EST	5) UIAButton [name:(null) value:(null) rec...	Debug	
12	11:01:32 PM EST	4) UIATableView [name:Empty list value:...	Debug	
13	11:01:32 PM EST	5) UIATableCell [name:Welcome to Plain...	Debug	
14	11:01:32 PM EST	6) UIAElement [name:Welcome to PlainN...	Debug	
15	11:01:32 PM EST	5) UIATableCell [name:PlainNote is open...	Debug	
16	11:01:32 PM EST	6) UIAElement [name:PlainNote is open s...	Debug	
17	11:01:32 PM EST	4) UIAToolbar [name:(null) value:(null) re...	Debug	
18	11:01:32 PM EST	5) UIAButton [name:Help value:(null) rec...	Debug	
19	11:01:32 PM EST	5) UIAButton [name:Sync value:(null) rec...	Debug	
20	11:01:33 PM EST	Add Note Test	Fail	
21	11:01:33 PM EST	Script completed.	Default	

Writing UI Automation Scripts

ELEMENT TREE

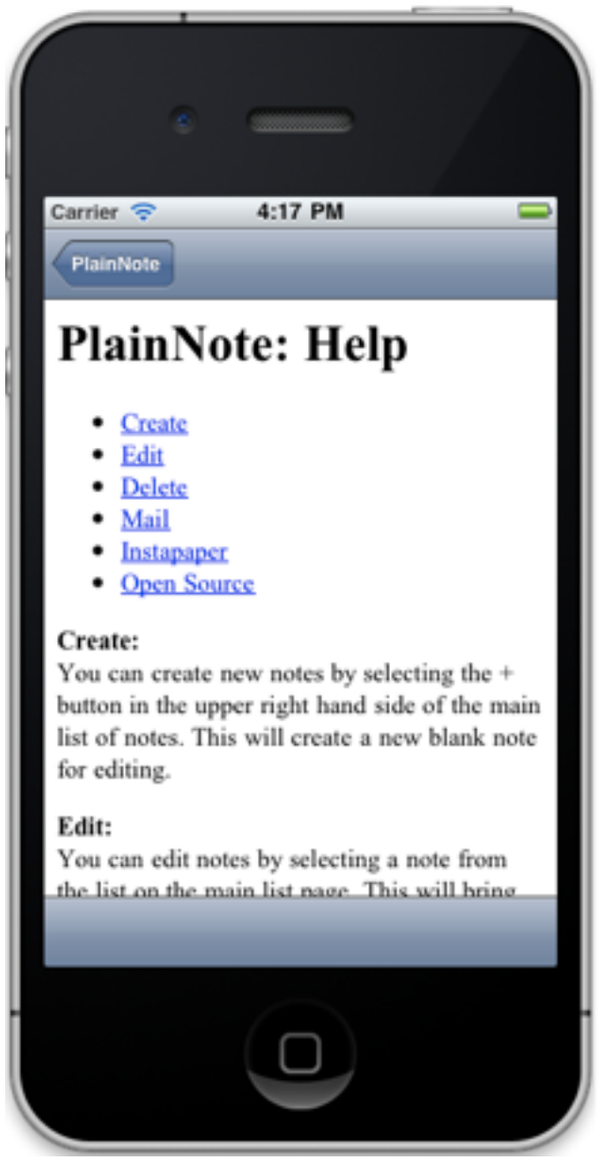
UIATarget
UIAApplication
UIANavigationBar
UIAWindow
UIATableView
UIATableCell

```
UIATarget.localTarget().frontMostApp()
  .mainWindow().tableViews()[0].cells()[0]
```



Technique to visualize element tree

```
UIATarget.localTarget()
  .logElementTree();
```



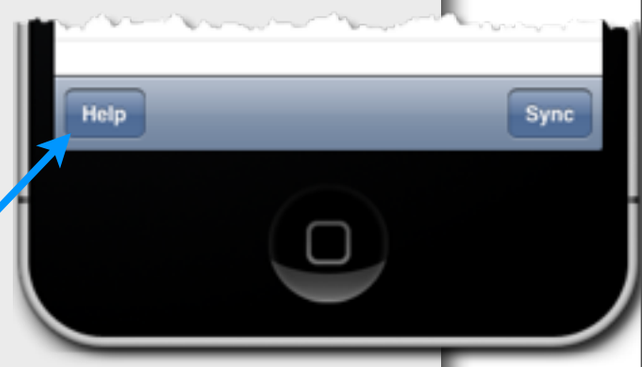

```
1) UIATarget [name:(null) value:(null) NSRect: {{2.7520829e-39, 2.0667855e-36}, {-1.9984865, 1.7892661e-37}}]
2) UIAApplication [name:PlainNote value:(null) NSRect: {{0, 20}, {320, 460}}]
3) UIAWindow [name:(null) value:(null) NSRect: {{0, 0}, {320, 480}}]
4) UIAScrollView [name:(null) value:(null) NSRect: {{320, 64}, {320, 372}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {54, 54}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {54, 54}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {54, 54}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {54, 54}}]
5) UIAImage [name:(null) value:(null) NSRect: {{305.5, 78.5}, {30, 1}}]
5) UIAImage [name:(null) value:(null) NSRect: {{305.5, 78.5}, {30, 1}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {1, 30}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {1, 30}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 470}, {320, 30}}]
5) UIAImage [name:(null) value:(null) NSRect: {{320, 64}, {320, 30}}]
5) UIAWebView [name:(null) value:(null) NSRect: {{320, 64}, {320, 1019}}]
6) UIAStaticText [name:1 value:(null) NSRect: {{328, 73}, {220, 36}}]
6) UIAElement [name:• value:(null) NSRect: {{350, 131}, {7, 19}}]
6) UIALink [name:Create value:(null) NSRect: {{368, 131}, {42, 19}}]
7) UIALink [name:Create value:(null) NSRect: {{368, 131}, {42, 19}}]
6) UIAElement [name:• value:(null) NSRect: {{350, 151}, {7, 19}}]
6) UIALink [name:Edit value:(null) NSRect: {{368, 151}, {27, 19}}]
7) UIALink [name:Edit value:(null) NSRect: {{368, 151}, {27, 19}}]
6) UIAElement [name:• value:(null) NSRect: {{350, 171}, {7, 19}}]
6) UIALink [name:Delete value:(null) NSRect: {{368, 171}, {42, 19}}]
7) UIALink [name:Delete value:(null) NSRect: {{368, 171}, {42, 19}}]
6) UIAElement [name:• value:(null) NSRect: {{350, 191}, {7, 19}}]
6) UIALink [name:Mail value:(null) NSRect: {{368, 191}, {31, 19}}]
7) UIALink [name:Mail value:(null) NSRect: {{368, 191}, {31, 19}}]
6) UIAElement [name:• value:(null) NSRect: {{350, 211}, {7, 19}}]
6) UIALink [name:Instapaper value:(null) NSRect: {{368, 211}, {67, 19}}]
7) UIALink [name:Instapaper value:(null) NSRect: {{368, 211}, {67, 19}}]
6) UIAElement [name:• value:(null) NSRect: {{350, 231}, {7, 19}}]
6) UIALink [name:Open Source value:(null) NSRect: {{368, 231}, {84, 19}}]
7) UIALink [name:Open Source value:(null) NSRect: {{368, 231}, {84, 19}}]
6) UIAStaticText [name:Create: value:(null) NSRect: {{328, 267}, {52, 19}}]
```

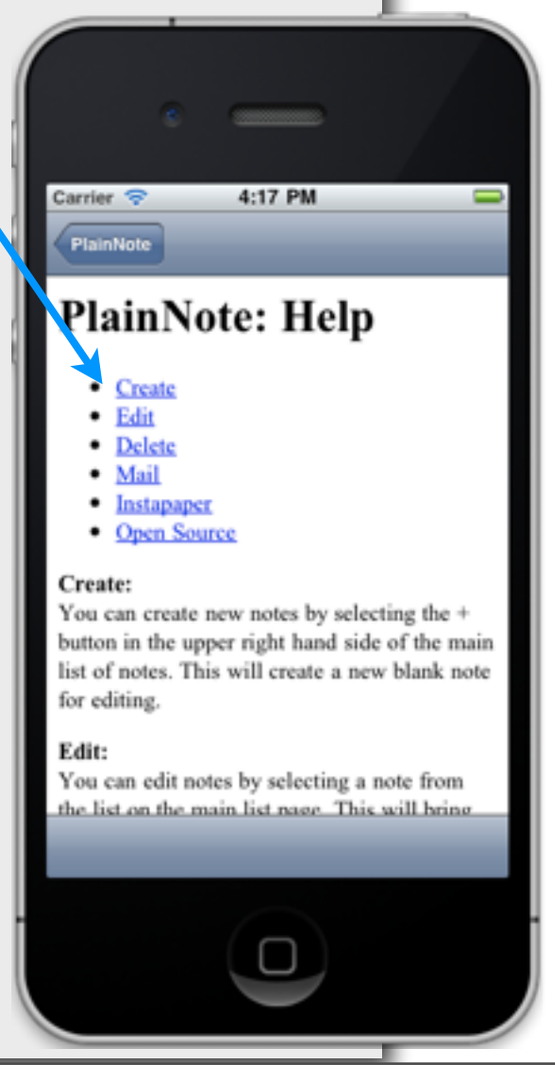
EXAMPLE TEST SCRIPT

Start test →

End test →

```
var target = UIATarget.localTarget();
var app = target.frontMostApp();
var window = app.mainWindow();

function testDisplayingHelp(testName) {
   UIALogger.logStart(testName);
   try {
       htmlIsDisplayed = false;

       app.toolbar().buttons()["Help"].tap();

       // without the delay the element was not on the screen
       target.delay(1);

       // No good way to get to the URL or contents of a WebView
       webView = window.scrollViews()[0].webViews()[0];
       if("Create" == webView.links()[0].name()) {
           htmlIsDisplayed = true;
       }

       window.navigationBar().leftButton().tap();

       if(htmlIsDisplayed) {
           UIALogger.logPass(testName);
       } else {
           UIALogger.logFail(testName);
       }

   } catch (e) {
       UIALogger.logError(e);
       UIATarget.localTarget().logElementTree();
       UIALogger.logFail(testName);
   }
}

testDisplayingHelp("Open Help Test");
```



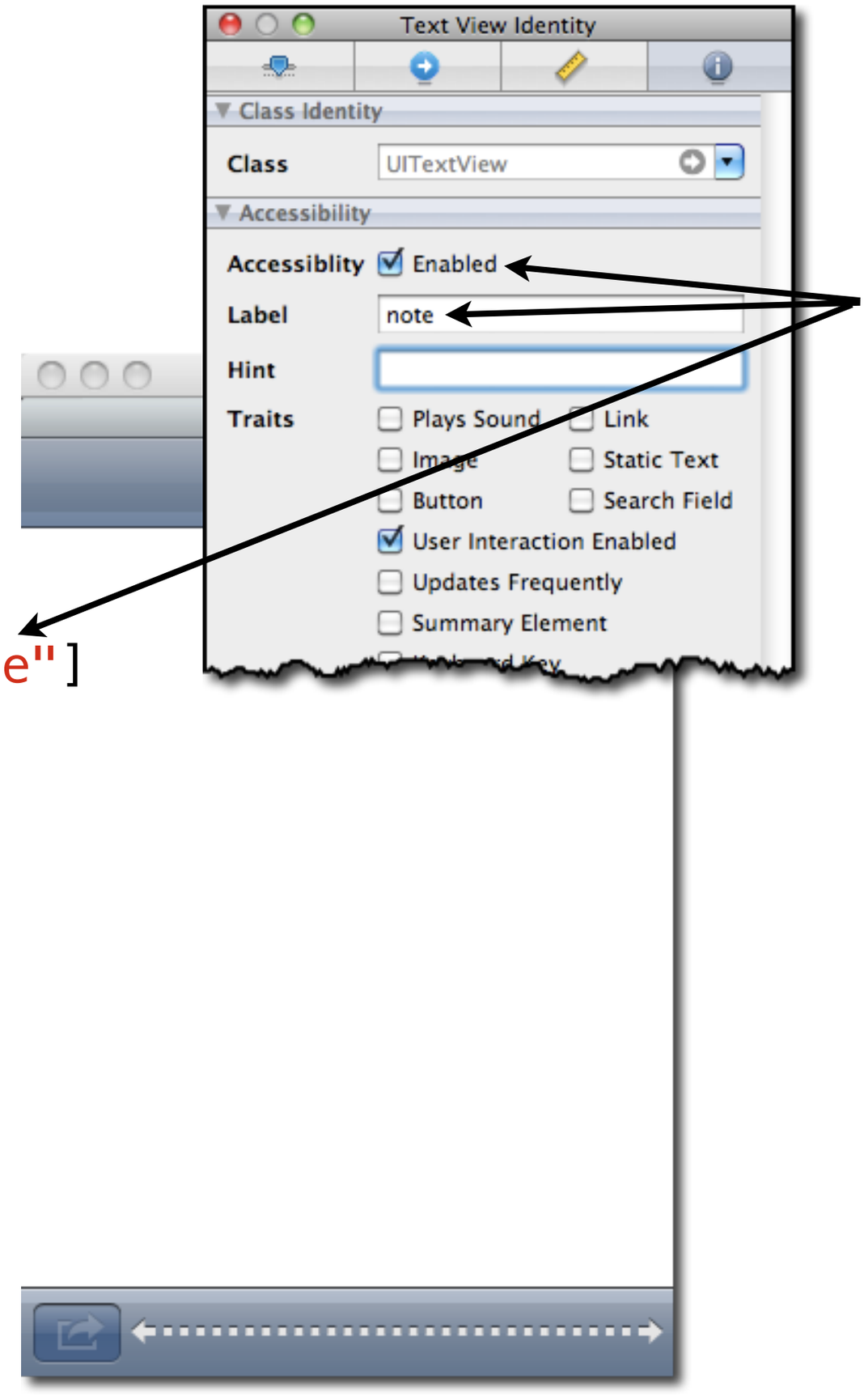


Instead of

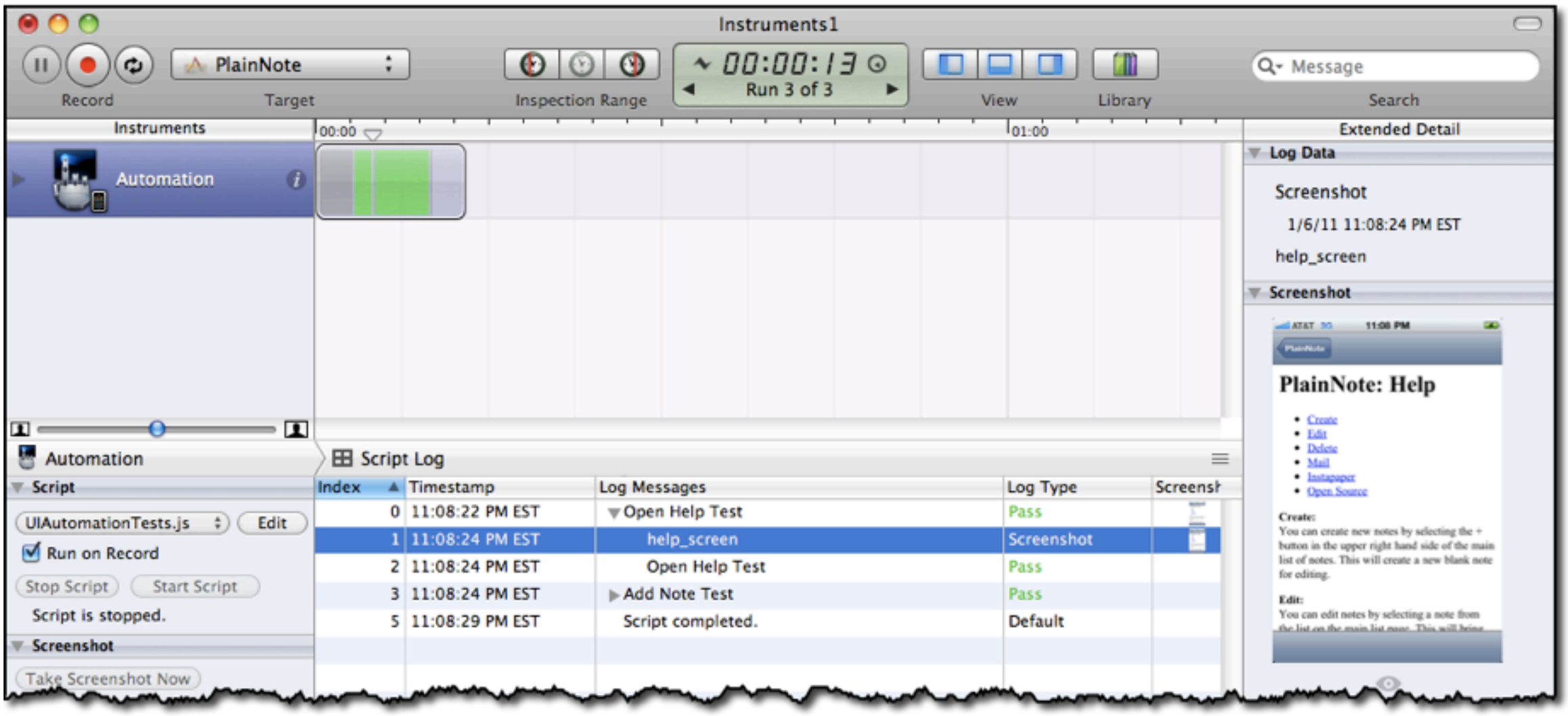
```
window.scrollViews()[0].textViews()[0]
.setValue("CodeMash is cool!!!");
```

use

```
window.scrollViews()[0].textViews()["note"]
.setValue("CodeMash is cool!!!");
```



```
UITarget.localTarget().captureScreenWithName("help_screen");
```



Does not work in simulator :(

When things dont work

add

```
UIATarget.delay(1);
```

Make it easier with **tuneup**

provides templates and asserts

```
#import "tuneup/tuneup.js"

test("Open Help Test", function(target, app) {

    app.toolbar().buttons()["Help"].tap();

    // without the delay the element was not on the screen target.delay(1);
    target.captureScreenWithName("help_screen");

    // No good way to get to the URL or contents of a WebView
    webView = app.mainWindow().scrollViews()[0].webViews()[0];

    assertEquals("Create", webView.links()[0].name(), "HTML must be displayed");

    app.mainWindow().navigationBar().leftButton().tap();

});
```

https://github.com/alexvollmer/tuneup_js

Page Object

CheezyWorld

extremely cheezy

Home Contact

← Slides from Columbus Code Camp

UI Tests – Part Two →

UI Tests – How do we keep them from being brittle?

Posted on November 9, 2010 by cheezy

There has been a lot of talk about the value of tests that drive the user interface. Some people in the industry that I have a lot of respect for have gone as far as to say that you should not create these type of tests. You should instead create tests that access the application code directly under the UI. Part of their reasoning is that tests that hit the UI are brittle and therefore high maintenance.

And yet UI tests are very valuable. When a user describes the behavior of a system they usually do so in terms of the user interface. Also, acceptance tests that provide examples of behavior for a story almost always present this behavior from the end users point of view.

This post is the first in a series that will describe the techniques I use to make my UI tests agile. These techniques have been developed while coaching teams in very diverse technologies and platforms and as a result are "field tested". The tools I will use for all of the examples are ruby and cucumber.

The application under test

We'll be writing scripts against the depot application. It is a simple rails application that

Search

Archives

- January 2011
- December 2010
- November 2010
- October 2010
- September 2010

Categories

- Agile
- C++
- Cheezy Stuff
- Cucumber
- Lean
- Pair Programming
- Product Owner
- Ruby
- Test Driven Development
- Testing
- Watir

I am a @LeanDog

http://www.cheezyworld.com/2010/11/09/ui-tests-not-brittle/

- works with other instruments
- can run in the simulator or the device

- no command-line automation
- no fixtures (setup, teardown)
- difficult to debug (element tree & delays)
- little documentation
- test run must be manually stopped

UI Automation Resources

UI Automation Reference Collection

http://developer.apple.com/library/ios/#documentation/DeveloperTools/Reference/UIAutomationRef/_index.html

Session 306 - Automation User Interface Testing with Instruments

iCuke

https://github.com/unboxed/icuke

Cuke Script

```
Feature: Adding a note
  In order to remember an important item
  As a common person
  I want to add a note to my collection of notes

  Background:
    Given "PlainNote.xcodeproj" is loaded in the iphone simulator

Scenario: Adding a note
  When I tap "Add"
  And I type "CodeMash is Cool!!!"
  And I tap "Save"
  Then I should see "CodeMash is Cool!!!"
```

- can be automated
- human readable
- easy to write

- can only run on simulator
- little documentation
- can not run in instruments

Resources

The Pragmatic Bookshelf

Books | Video |

BDD on iPhone: iCuke

iCuke Gives iPhone the Cucumber Treatment

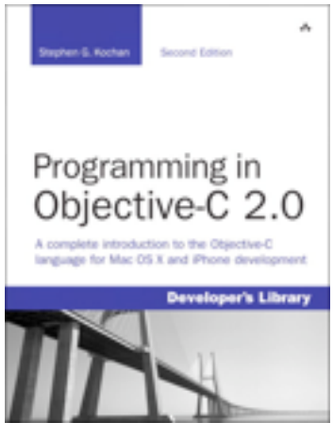
by Rob Holland

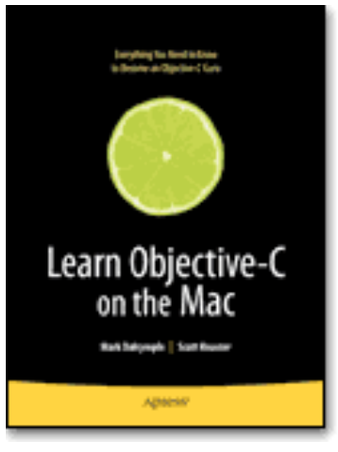
Almost a year ago in these pages, Ian Dees showed how to use Cucumber to test your iPhone apps. Now iCuke makes it even easier.

Aslak Hellesoy's Cucumber has seen huge success helping developers and project owners clearly communicate desired behavior in the Ruby on Rails arena.

http://pragprog.com/magazines/2010-07/bdd-on-iphone-icuke

Great iOS Resources





The Objective-C Programming Language

Christopher M. Judd

Judd Solutions

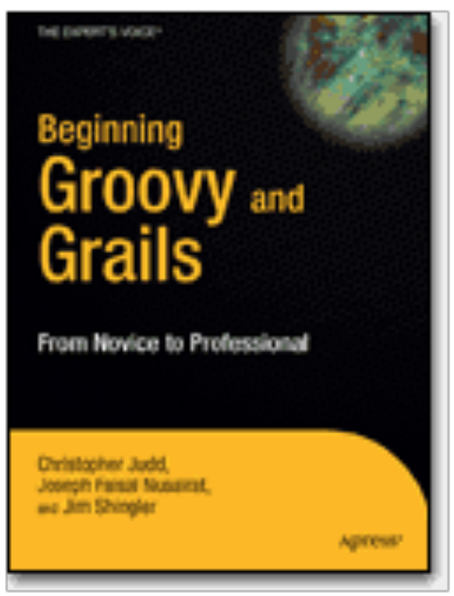
President/Consultant/Author

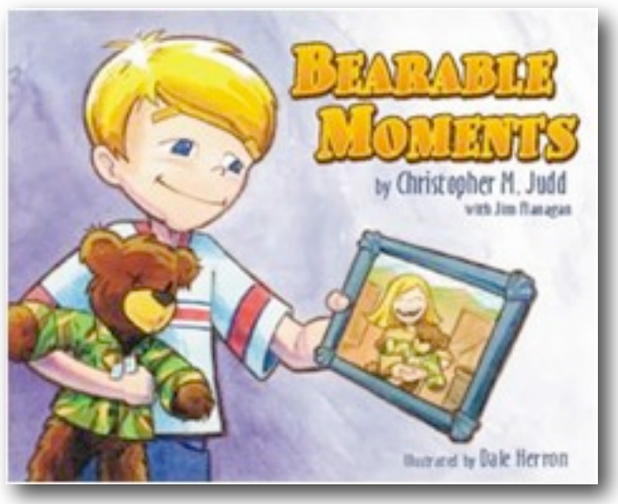
email: cjudd@juddsolutions.com

web: www.juddsolutions.com

blog: juddsolutions.blogspot.com

twitter: javajudd





Attributions

http://www.organicdesign.co.nz/File:Warning.svg

http://www.flickr.com/photos/heliotrop3/4310957752/