iOS Client Testing

Integration tests using KIF Framework

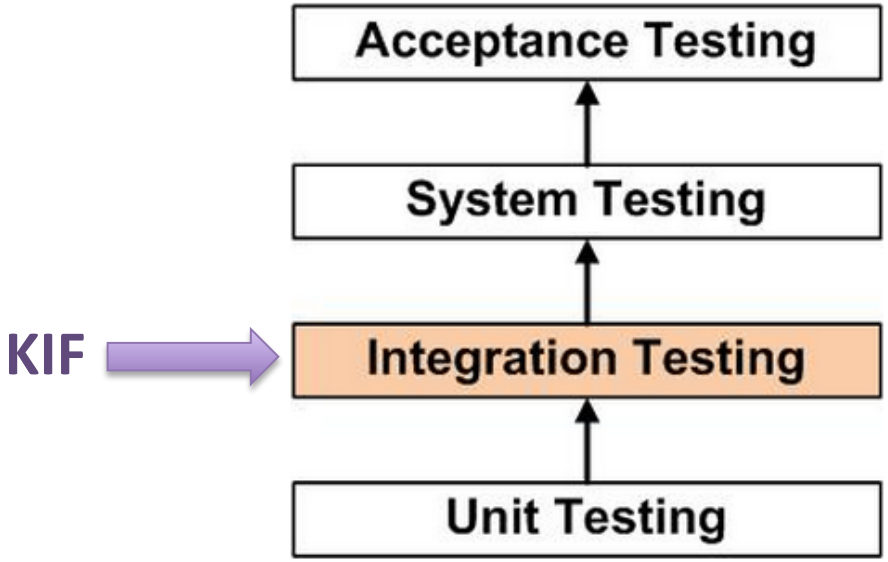
KIF: Keep It Functional

- KIF is an **Integration Test Framework** for **iOS**

Integration testing

From Wikipedia, the free encyclopedia

Integration testing (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.



UNIT TESTS

INTEGRATION TESTS

- Verify that a relatively small piece of code is doing what is intended to do
- Narrow in scope
- Easy to write and execute
- Intended for the use of the programmer
- Testers and users downstream should benefit from seeing less bugs
- Done in total isolation

- Demonstrate that different pieces of the system work together
- Can cover whole applications
- Require more effort to put together
- Intended for the use of non-programmers
- Their output is the integrated system ready for System Testing
- Done altogether

KIF: Keep It Functional

- Open source framework
- Black-box testing (it doesn't make it into production code)
- It allows for easy automation of iOS apps by leveraging ***accessibility attributes***
- Tests are run synchronously in the main thread (running the run loop to force the passage of time)
- KIF 2.0 is not compatible with KIF 1.0, it uses a different test execution mechanism

Features

Minimizes indirection

- All of the tests for KIF are written in Objective C.
- Maximum integration with your code.
- Minimizes the number of layers you have to build.

Easy configuration

- KIF integrates directly with Xcode.
- No need to run an additional web server or install any additional packages.

Wide OS coverage

- Works for iOS 5.1 and above.

Features

Test like a user

- KIF attempts to imitate actual user input.
- Automation is done using tap events wherever possible.

Automatic integration with Xcode 5 testing tools

- You can easily run a KIF test with the test navigator or kick off nightly acceptance tests with Bots.

Pros and Cons

> Pros

- Easy to integrate into your project.
- Easy to learn and implement.
- With just a few couple of methods you can simulate a lot of tests for your application.

> Cons

- Poorly documented.
- Not easy to set up accessibility labels for custom elements.
- Not easy to restore state between tests.

Classes in KIF

- There are just a couple of classes that you have to know from KIF 2.0 framework.

KIFTestCase is the class you have to subclass to write your tests. This class inherits from SenTestCase.

KIFUITestActor is a class you will make a category of in order to add some custom groups of steps which are specific to your application, such as "navigateToLoginPage", "returnToHomeScreen", etc.

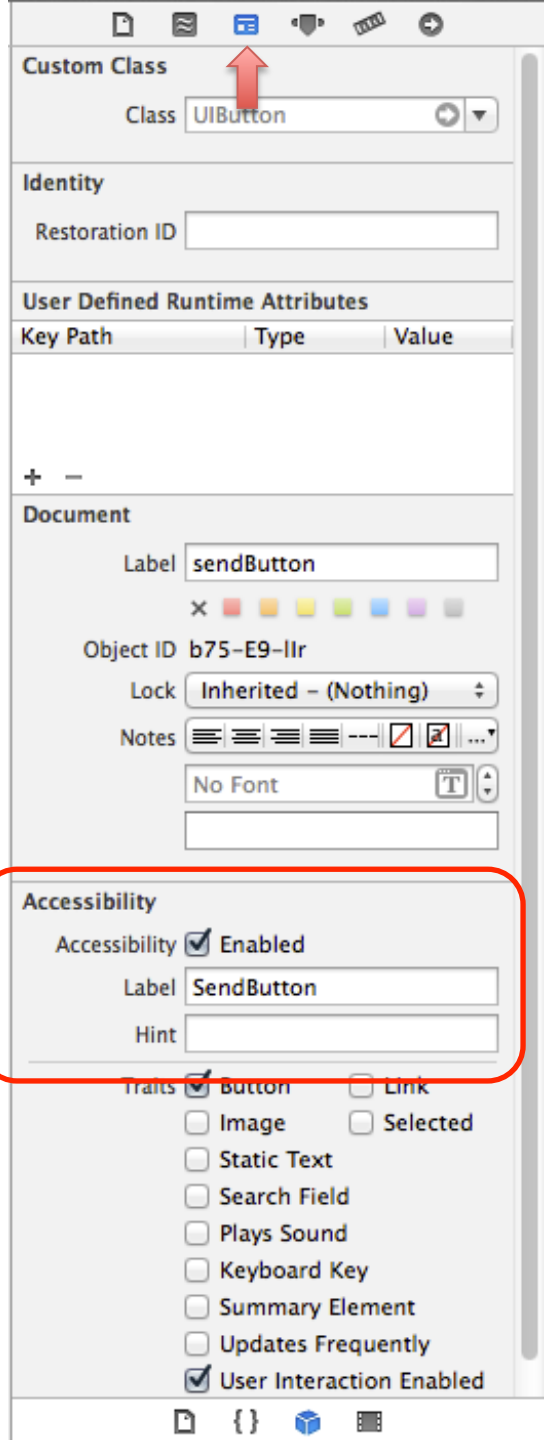
Accessibility Labels

- Every time you want to access an interface object from KIF, you do not access it like you usually do in your code.

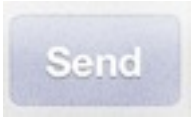
- Instead, you set up an specific accessibility label for that object and then you access it by using any "… `WithAccessibilityLabel: (NSString *)accessibilityLabel`" method from `KIFUITestActor` class.

- Example:

```
[tester tapViewWithAccessibilityLabel:@"SendButton"];
```



Commonly used methods




```
[tester waitForViewWithAccessibilityLabel:@"SendButton”];
```

```
[tester tapViewWithAccessibilityLabel:@"SendButton"];
```


```
[tester enterText:@”78"
intoViewWithAccessibilityLabel:@"ChatTextField"];
```

```
[tester swipeViewWithAccessibilityLabel:@"FeedScrollView"
inDirection:KIFSwipeDirectionDown];
```

```
[tester waitForTimeInterval:1.5];
```

```
[tester tapScreenAtPoint:CGPointMake(20., 20.)];
```

Test example

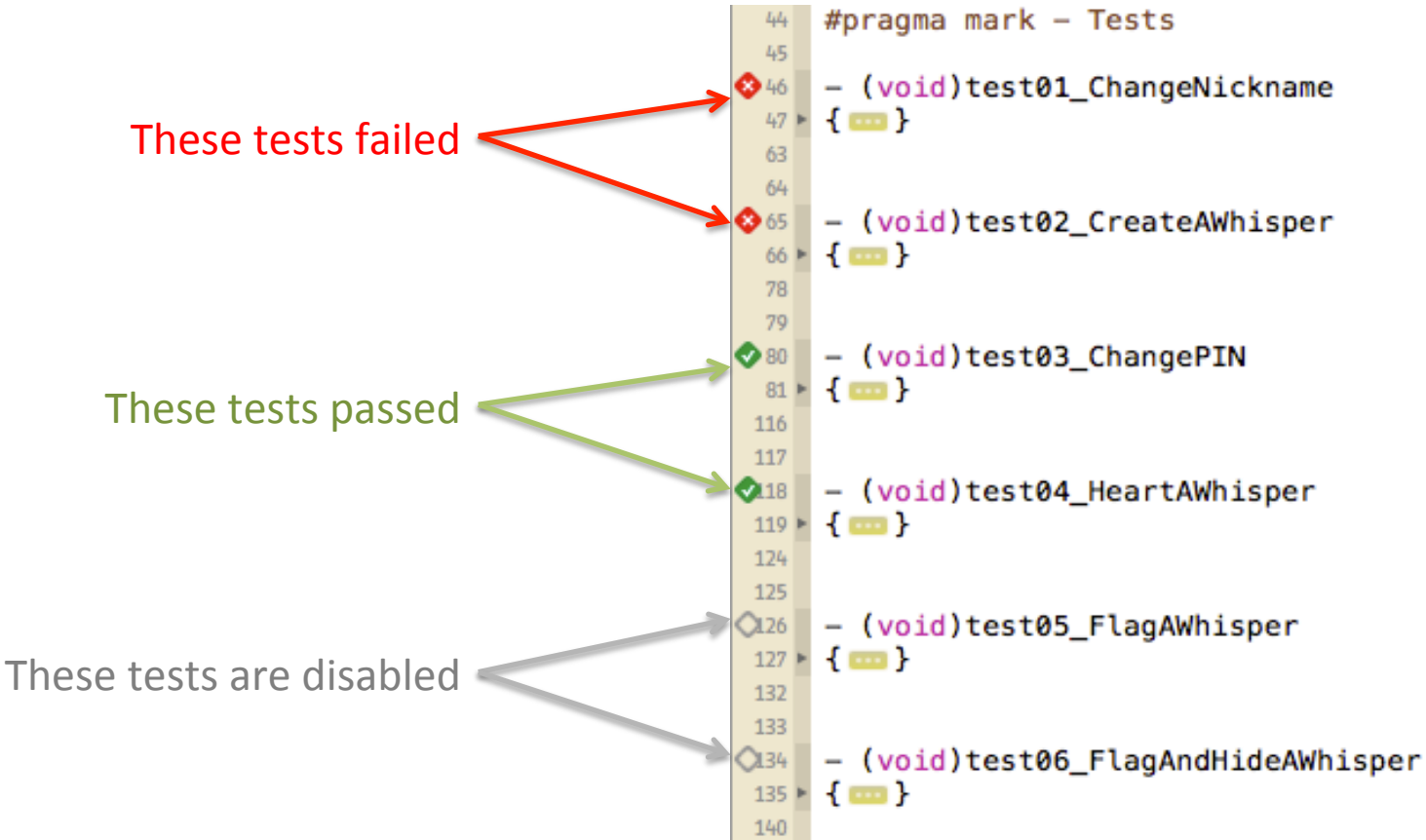
```
- (void)testChangeNickname:(NSString *)newName
{
    /* Go to settings */
    [tester navigateToSettings];

    /* Enter new text */
    [tester waitForTappableViewWithAccessibilityLabel:@"NicknameTextField"];
    [tester clearTextFromAndThenEnterText:newName
intoViewWithAccessibilityLabel:@"NicknameTextField"];
    [tester waitForTimeInterval:1.];
    [tester waitAndTapTappableViewWithAccessibilityLabel:@"done"];

    /* Go back to initial state */
    [tester navigateHome];
}
```

Running tests

- To run **the whole suite of tests**, you just press CMD+U.
- To run **an individual test** you can press the diamond button at the left of the test method signature.



Running tests

- There are 4 special test methods, which are:

```
- (void)beforeAll
{ ... }
```

Executed **only once *before*** running all your tests.

```
- (void)afterAll
{ ... }
```

Executed **only once *after*** running all your tests.

```
- (void)beforeEach
{ ... }
```

Executed ***before*** running **each** test

```
- (void)afterEach
{ ... }
```

Executed ***after*** running **each** test

Source

https://github.com/kif-framework/KIF (almost the unique site where you'll find any KIF documentation).

http://cocoadocs.org/docsets/KIF/2.0.0/ (an 'apple-documentation like' site for KIF framework).

http://en.wikipedia.org/wiki/Integration_testing