# Testing in Xcode 5

你没有看错！！！！！！！ 我们先从爷爷的爷爷的爷爷开始

Session 409

**Mike Swingler**
Xcode Engineer

# Why Test?

- Catch crashes 获取崩溃
- Catch logic errors 获取逻辑错误
- Help you write your code 帮助你写代码，TDD
- Catch regressions 获取回归测试
- Cover more configurations 覆盖更多的配置
- Cover everyone, all the time 无时无刻与他人产生关联

# Overview

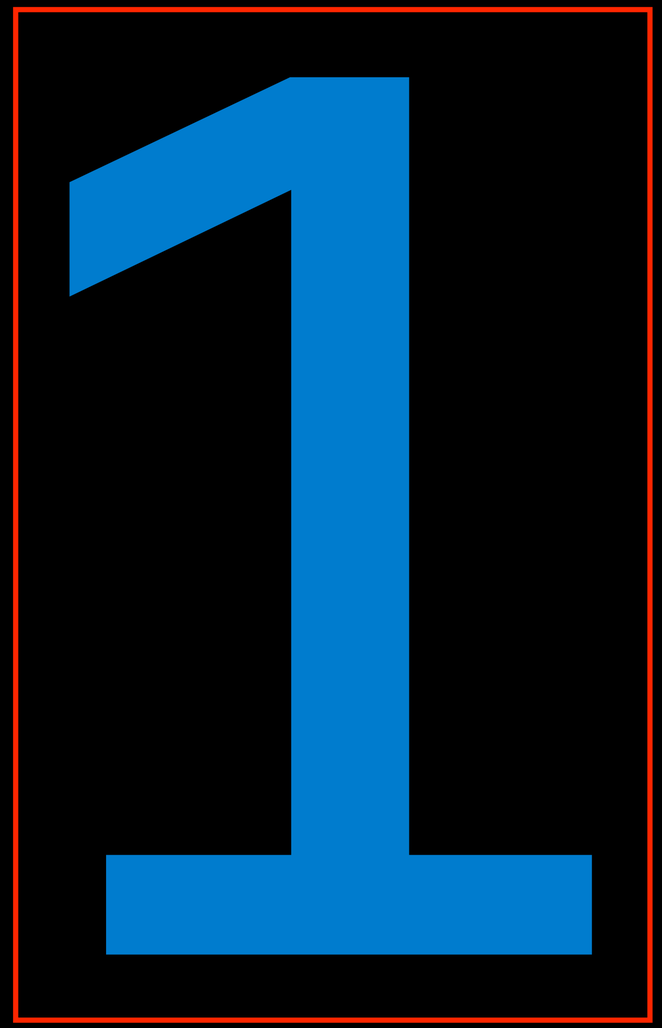
- What is a unit test?
- Introducing XCTest 介绍XCTest测试框架
- Writing, running and debugging tests 写、运行、调试测试
- Continuous integration 持续集成
- Advanced setups 高级配置

# Unit Testing

# What is a Unit Test?

- Tests **ONE** thing 测试一件事情
  - Single “unit” of functionality 1.单一函数单元
  - Pass/Fail 2.测试是通过还是失败
  - Small, fast, isolated 3.小、快、独立
- Unit tests don’t cover
  - Performance
  - UI interaction
  - Whole system integration

未覆盖



# Your App

## Where to start testing

从哪里开始测试

Database

View

Server

Controller

Model

# Introducing XCTest

# XCTest.framework


- iOS and OS X
- Requires Xcode 5
- Derived from OCUnit 源自于OCUnit
  - Modernized
  - Migration tool
- Builds .xctest bundles 编译，xctest bundles
- Test runner injected in your app 测试运行注入到App
- Test rig loads libraries and test bundles

测试暗箱加载库与测试bundles

体现在不同Bundle下可以直接导入投文件我

# Test Code 测试代码

- Subclass XCTestCase
- test = method
- Prefixed “test”
- No arguments, returns void
  - Makes assertions
  - Built into test target

```
@interface ExampleTests : XCTestCase
@end

@implementation ExampleTests

- (void) testExample {
    XCTAssertTrue(2 + 2 == 4);
}

@end
```

测试的基本结构：Given-> When -> Then　测试3段式

1.given: 初始条件的设置

2.When：被测试的一些方法

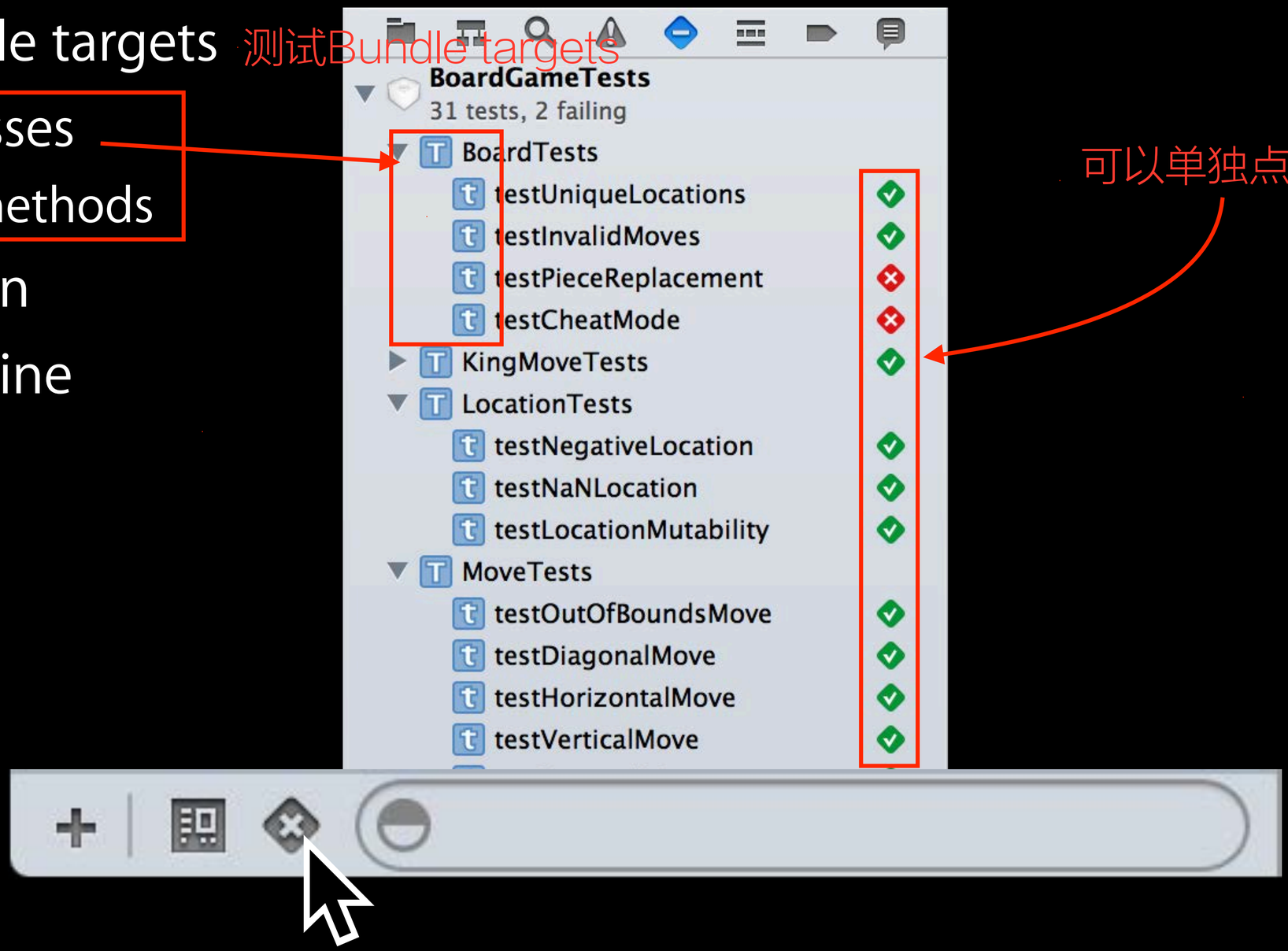
3.测试的一些结果

# Test Navigator


- Test bundle targets
  - Test classes
    - Test methods
- Click to run
- Results inline

测试Bundle targets

可以单独点击



# Editor Test Indicators


Given

When

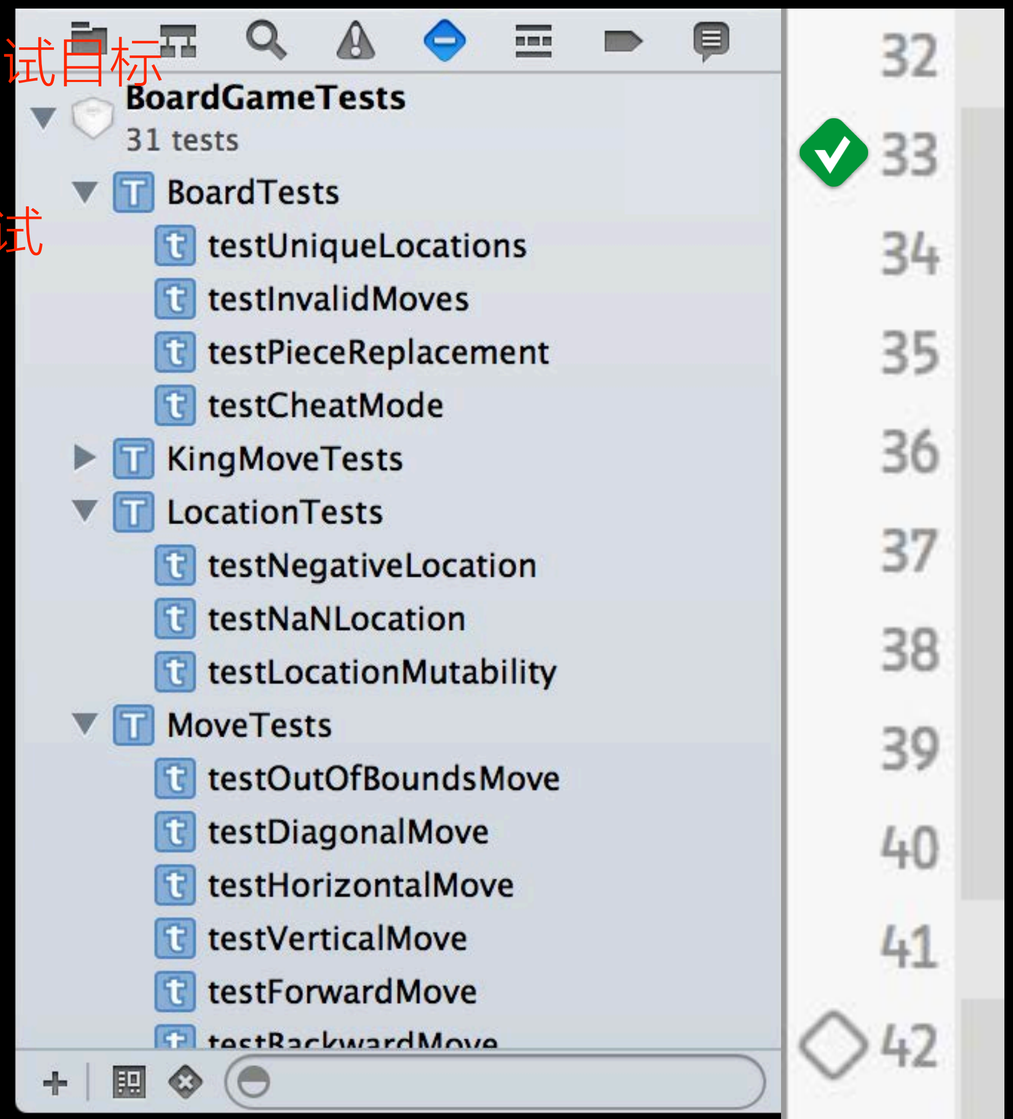
Then

```
32
33  - (void)testVerticalMove {
34      Location *invalidLocation = [[Location alloc] initWithX:0 andY:0];
35      XCTAssertFalse([self.ruleChecker validateLocation:invalidLocation forPiece:self.piece]);
36
37      Location *location = [[Location alloc] initWithX:0 andY:1];
38      BOOL verticalMoveWasValid = [_ruleChecker validateLocation:location forPiece:_piece];
39      XCTAssertFalse(verticalMoveWasValid, @"Vertical move succeeded!");
40  }
41
42  - (void)testOutOfBoundsMove {
43      // negative location
44      Location *location = [[Location alloc] initWithX:-1 andY:-1];
45      BOOL negativeMoveWasValid = [_ruleChecker validateLocation:location forPiece:_piece];
46      XCTAssertFalse(negativeMoveWasValid, @"Negative bounds move succeeded");
47
48      // too big x
49      location = [[Location alloc] initWithX:9 andY:0];
```

# Overview

## Making your first unit test in Xcode 5


- Creating a test target is easy 轻松创建一个测试目标
- Add a new test by adding a method
  通过添加方法来添加一个测试
- Easy to run tests
  - Test navigator
  - Editor



# Assert the Expected 断言的预料

## XCTestAssertions.h

XCTAssertThrowsSpecific

XCTAssertTrueNoThrow

XCTAssertNil

XCTAssertNoThrowSpecific

XCTAssertNotNil

XCTAssertEqualObjects

XCTAssertEqualsWithAccuracy

XCTFail

XCTAssertNoThrow

XCTAssertTrue

XCTAssertThrowsSpecificNamed

XCTAssertFalse

XCTAssertThrows

XCTAssertNoThrowSpecificNamed

XCTAssertEquals

XCTAssertFalseNoThrow

# Expect the Unexpected

预估的结果

- Expected success 符合预期
  - Can come first
- Regressions 回归
- Expected failure 预估失败
  - Overflow, ∞, NaN 益处、无穷，不是一个值
  - Nil 对象为空
  - Empty collections 空集合
  - Unexpected types in collections 集合类型不符合预期
  - NSError 错误

# Set Up a Test

## `-setUp`

- Runs before every test method 所有测试方法运行前会运行
- Create “shim” objects 创建“垫片”对象
- Load data from .xctest bundle 从测试bundle中加载数据
  - `[NSBundle bundleForClass:[MyTestClass class]]`
- …anything you need to setup “the world”
- Use `-tearDown` to perform any cleanup 使用 tearDown对象方法执行清除

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ } 只调用一次

+ (void) tearDown { /* Class tear-down. */ } 只调用一次

- (void) setUp { /* Test set-up. */ } 每个测试方法执行前调用

- (void) tearDown { /* Test tear-down. */ } 每个测试方法执行后调用

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# How XCTest Works

## Test class and methods

```
@interface ExampleTests : XCTestCase
@implementation ExampleTests

+ (void) setUp { /* Class set-up. */ }

+ (void) tearDown { /* Class tear-down. */ }

- (void) setUp { /* Test set-up. */ }

- (void) tearDown { /* Test tear-down. */ }

- (void) testExamplePassing {
    XCTAssertTrue(YES);
}

- (void) testExampleFailing {
    XCTAssertTrue(NO);
}
```

# Debugging Tests

# Testing Using OS X Server


Xcode 服务机器人，执行集成

- Xcode service bots perform integrations
  - Every time your commit code
  - On the hour

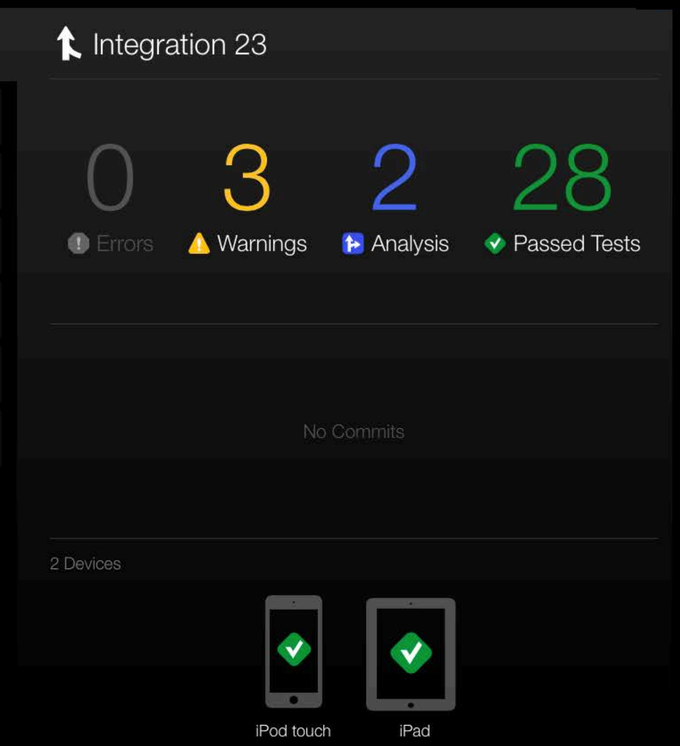
每次提交Commit点

- Create a Scheme 创建一个Scheme
  - Bots run shared schemes

机器人运行Shared Scheme配置进行打包

# Testing Using OS X Server


- Xcode service bots perform integrations Xcode服务机器人执行集成
  - Every time your commit code 任何时间的提交的Commit点
  - On the hour 间隔一个小时
- Create a Scheme 创建一个Scheme
  - Bots run shared schemes 机器人运行共享的schemes
- Brings results to you
  返回结果给你

# Many Configurations

NEW

- iOS devices

# Many Configurations


- iOS devices
- iOS simulator
- Different OS versions

# Overview

## Continuous integration with OS X Server

在server上持续集成

- Setup shared schemes for bot 1.配置shared Schemes 为机器人
- Covers multiple configurations 2. 覆盖种设备的配置
  - Devices, OS versions, simulators
- Bot and test results summary 3.机器人测试结果总结

# Command-line Testing


```
> xcodebuild test
```

# Command-line Testing


```
> xcodebuild test
-scheme MyLibrary
```

# Command-line Testing


```
> xcodebuild test
-scheme MyLibrary
-destination 'platform=OS X,arch=x86_64'
-destination 'platform=iOS,name=My Development iPod Touch’
-destination 'platform=iOS Simulator,name=iPhone,OS=6.1'
```

# Wrap Up 总结

- What is a unit test? 什么是单元测试?
- Introducing XCTest 介绍XCTest框架
- Writing, running and debugging tests 写、运行、调试一个测试
  - Test Navigator 测试导航器
  - Editor indicators 编辑指示器
  - Test failure breakpoint 测试失败断点
  - Assistant categories
  - Test Again command
- Continuous integration, advanced setups 持续集成与高级设置
- Testing helps you write better, high quality apps 测试帮助你写更高质量的APP

# More Information

**Dave DeLong**
Developer Tools Evangelist
delong@apple.com

**Xcode Documentation**
http://developer.apple.com/

**Apple Developer Forums**
http://devforums.apple.com

# Related Sessions

| | | |
|---|---|---|
| Continuous Integration with Xcode 5 | Presidio<br>Tuesday 3:15PM | |

# Related Labs

| | |
|---|---|
| Xcode and Continuous Integration Lab | Tools Lab A<br>Thursday 9:00AM |
| Tools Lab | Tools Lab<br>Ongoing |

WWDC2013