Unit-Testing mit Xcode

Dennis Lewandowski



Überblick

1. Unit-Testing
2. SenTestingKit
3. Demo
4. Debugging

Unit-Testing

- Funktionalität von Code sicherstellen
- Vorteile
  - “Saubere Umgebung”
  - Methoden unabhängig testen
  - automatisches Testen
- Nachteile
  - zusätzlicher Programmieraufwand
  - Einarbeitung in Testumgebung nötig

SenTestingKit

- Open source testing framework
- Wird mit Xcode installiert
- Seit Xcode4 integrierter Bestandteil der IDE



SenTesting Aufbau

```
    //  ExampleTest.h

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

@interface ExampleTest : SenTestCase {
}


//  ExampleTest.m

@implementation ExampleTest

-(void)setUp
{
//  wird vor jedem Test aufgerufen
    [super setUp];
}

-(void)tearDown
{
//  wird nach jedem Test aufgerufen
    [super tearDown];
}

-(void)testExistanceOfObjects
{
     //  Beispiel für einen Test
     STAssertTrue(YES, @"Logic is inverted");
}
```

Assertions

- STAssertTrue/STAssertFalse
  - Hat das Argument den Wert true/false?
- STFail
  - Lässt einen Test automatisch scheitern
- STAssertEquals
  - Prüft auf Gleichheit
- STAssertEqualObjects
- STAssertNoThrow
- STAssertNotNil

SenTestingKit in XCode3

- Project > New Target
  - Cocoa > Unit Test Bundle Target
- File > New File
  - Cocoa > Objective C test case class

DEMO

Debugging (Xcode3)

- Tests werden beim Build-Durchlauf ausgeführt
- Zum Debuggen muss getrickst werden!
  - Mac OS: http://chanson.livejournal.com/120740.html
  - iOS: http://hiediutley.com/2011/03/08/xcode3-debugging-ios-unit-tests/

Vielen Dank.

