(Unit) Testing iOS Apps

Paweł Dudek

So a QA guys walks into your room…

That's a lot of wasted time.

Could it be saved?

YES!!!

One of the many reasons why you want to have tests.

What are the others?

Reasons for testing

- Saved time

- Better codebase

- Faster development cycles

- Being "confident" about your code

- More saved time

More saved time? What?

Common misconceptions

Common misconceptions

- "It will take longer to write code" or "Time spent writing/refactoring tests is time lost"
- "It will take more time to modify existing system"

Am I going to write poor software if I don't do tests?

Are unit tests an invaluable tool for writing great software? Heck yes.
Am I going to produce a poor product if I can't unit test? Hell no.

Jonathan Rasmusson

Now that we know
that writing tests is a
good idea...

How can we do it?

Warning

- You will feel confused

- You won't know how to start

- You will need help
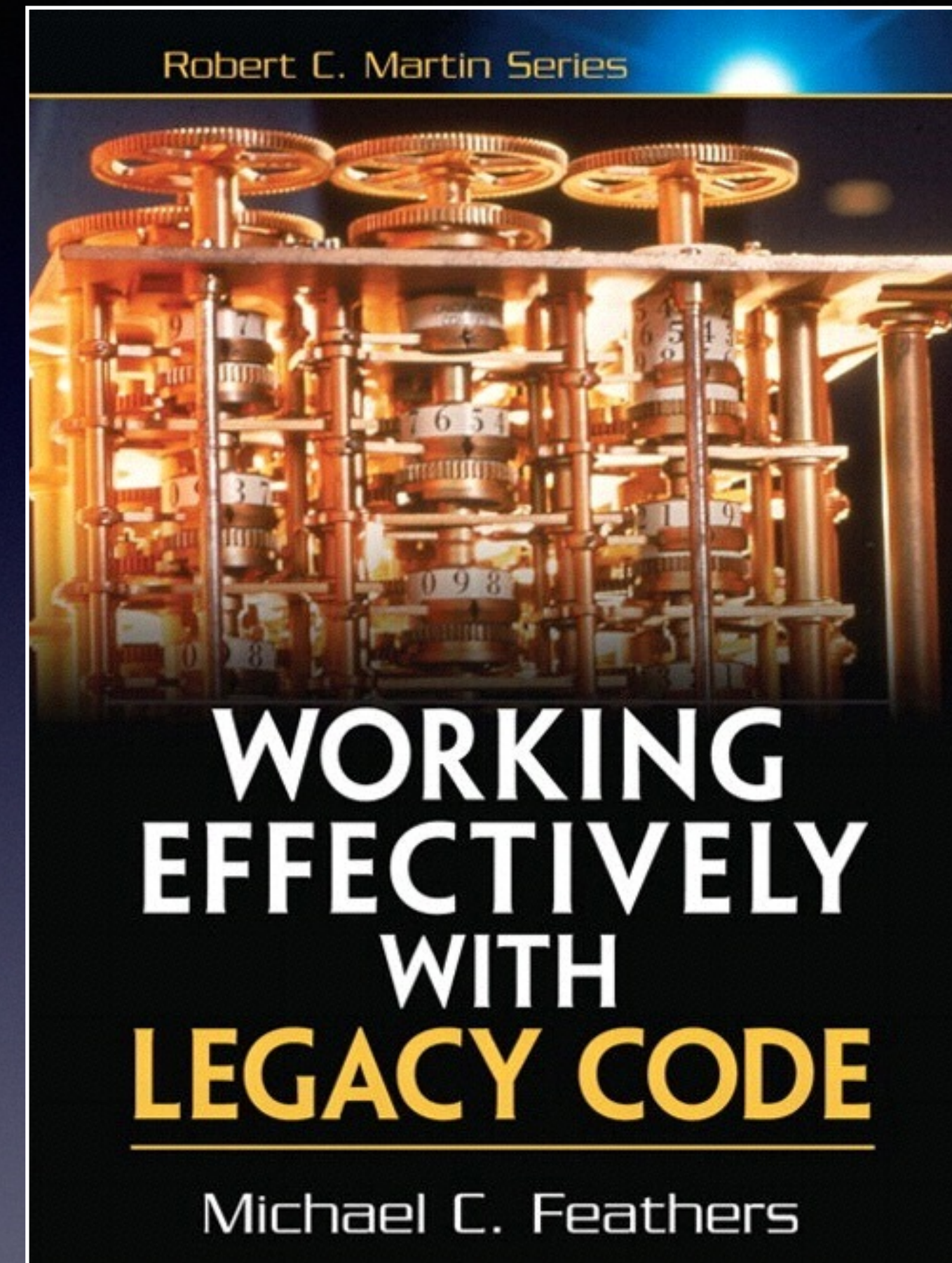
- Conclusion: it's not easy to start

Tips

- Never think of tests as tests

  - Think of a scenario, behavior, example

- Grab a mature project from github with tests included

- Find someone experienced and ask questions

- Program in pairs!

Get on with it!

How can we test?

Working effectively with legacy code

Michael C. Feathers

TDD

- Test Driven Development
- Red, Green, Refactor
- Write failing test first
- Fix it
- Refactor

BDD

Behavior Driven Development

How does BDD differ from TDD?

BDD builds upon TDD by formalising the good habits of the best TDD practitioners.

Matt Wynne,
XP Evangelist

http://blog.mattwynne.net/2012/11/20/tdd-vs-bdd/

Good habits

- Work outside-in

- Use examples to clarify requirements

- Use ubiquitous language

Thanks Matt!!

Unit Tests

OCUnit

- Oldest Mac testing framework - officially supported by Apple since 2005

- Integrated with XCode

OCUnit Syntax

- All test classes inherit from SenTestCase

- All tests begin with test

- Setup and teardown method

- Everything else is ignored by testing framework

  - Means you can use as additional setup methods!

OCUnit

```
-(void)testFullName {
    Person *person = [Person person];
    person.firstName = @"Mariusz";
    person.secondName = @"Testowniczek";
    NSString *fullName = [person fullName];
    NSString *expectedName = @"Mariusz Testowniczek";
    STAssertTrue([fullName isEqualToString:expectedName], @"");
}
```

OCUnit vs XCTest

OCUnit vs XCTest

Behavior Tests

Kiwi and Cedar

- BDD-style frameworks

- Optimized for readability

- Both have nearly identical syntax

Kiwi and Cedar Syntax

```objectivec
SPEC_BEGIN(PersonSpec)

describe(@"Person", ^{
    __block Person *person;

    beforeEach(^{
        person = [[Person alloc] init];
        person.firstName = @"Mariusz";
        person.lastName = @"Fixture Last Name";
    });


    describe(@"full name", ^{

        __block NSString *fullName;

        beforeEach(^{
            fullName = [person fullName];
        });

        it(@"should return the full name", ^{
            expect(fullName).to(equal(@"Mariusz Testowniczek"));
        });
    });
});

SPEC_END
```

iOS Testing Tips

Testing UI Layout

- Hard to maintain
- Gives little value
- Usually takes time to set up

System Singletons

```
[UIDevice currentDevice]
[UIScreen mainScreen]
```

• Makes hard to test if accessed directly

• Nice candidate for injecting as a dependency

• Using singletons - generally discouraged

Common caveats

- Don't set mocks on `[UIViewController view]`

- Avoid using categories to override system properties or existing behaviour

- Keychain and most of system objects are unavailable when tests are run from command line w/o simulator

Example

More advanced topics

- Application Tests - Frank / KIF

- Dependency injection - Objection, Typhoon

- Mocks/Stubs: OCMock, OCMockito

- Syntax Sugars: Expecta

- Matchers: OCHamcrest

Summary

Summary

- Testing is a great way to help developers

- Better codebase, faster iterations

- Invaluable for larger projects

Resources & Contact

Code Examples
github.com/paweldudek

Contact
@eldudi
pawel@dudek.mobi