

Macoun

96% Testabdeckung

Maxim Zaks @iceX33

Ablauf

Ablauf

- Was ist Test Abdeckung und wie kann man es messen

Ablauf

- Was ist Test Abdeckung und wie kann man es messen
- Wann ist hohe Testabdeckung eine gute Strategie

Ablauf

- Was ist Test Abdeckung und wie kann man es messen
- Wann ist hohe Testabdeckung eine gute Strategie
- Taktiken wie man eine hohe Testabdeckung erreichen kann

Was ist Testabdeckung?

90.0% of 30 lines

/Users/mzaks/dev/Wooga/g10-ios/GameX/GameX/src/GXBattleMenuViewController.m

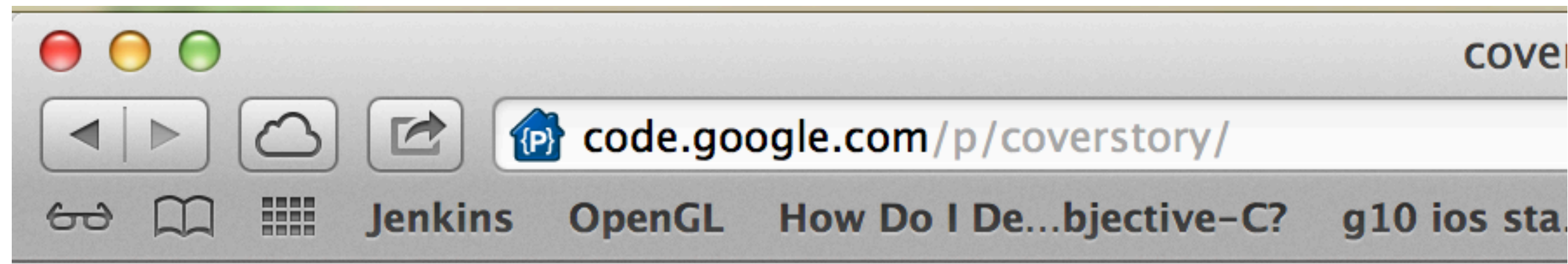
Open

Executed 90.0% of 30 lines (2 sources, 27 executed, 30 executable, 66 total lines)

29 lines (26 executed, 29 executable, 45 total lines)

...Controller.h 100

```
1  @implementation GXBattleMenuViewController {
2      GXContainerViewController *_containerViewController;
3  }
4
5  - (void) prepareForSegue:(UIStoryboardSegue *)segue sender:(id)sender {
6      [super prepareForSegue:segue sender:sender];
7      if([segue.destinationViewController isKindOfClass:[GXContainerViewController class]]
8          _containerViewController = segue.destinationViewController;
9      [self showTabIndex:lastSelectedTab];
10  }
11
12  - (IBAction) showEnemySelector{
13      [self showTabIndex:0];
14  }
15
16  - (IBAction) showAttackHistory {
17      [self showTabIndex:1];
18  }
19
20  - (IBAction) showNPCMap {
21      [self showTabIndex:2];
22  }
23
24  -(void)showTabIndex:(NSUInteger)index{
25      NSArray *ids = @[ENEMY_SELECTOR_ID, ATTACK_HISTORY_ID, NPC_MAP_ID];
26      for (GXTabButton *button in _tabButtons){
27          BOOL isSelected = (button.tag == index);
28          [button setSelected:isSelected];
29      }
30      lastSelectedTab = index;
31      [_containerViewController showSubviewControllerWithId:ids[index]];
32  }
33
34  -(void) resetSelectedTab {
35      lastSelectedTab = 0;
36  }
37
38  @end
```

coverstory

UI for analyzing files using gcov on the Mac

Project Home

[Downloads](#)

[Wiki](#)

[Issues](#)

[Source](#)

Summary

[People](#)

Project Information

 +11 Recommend this on Google

★ Starred by 138 users

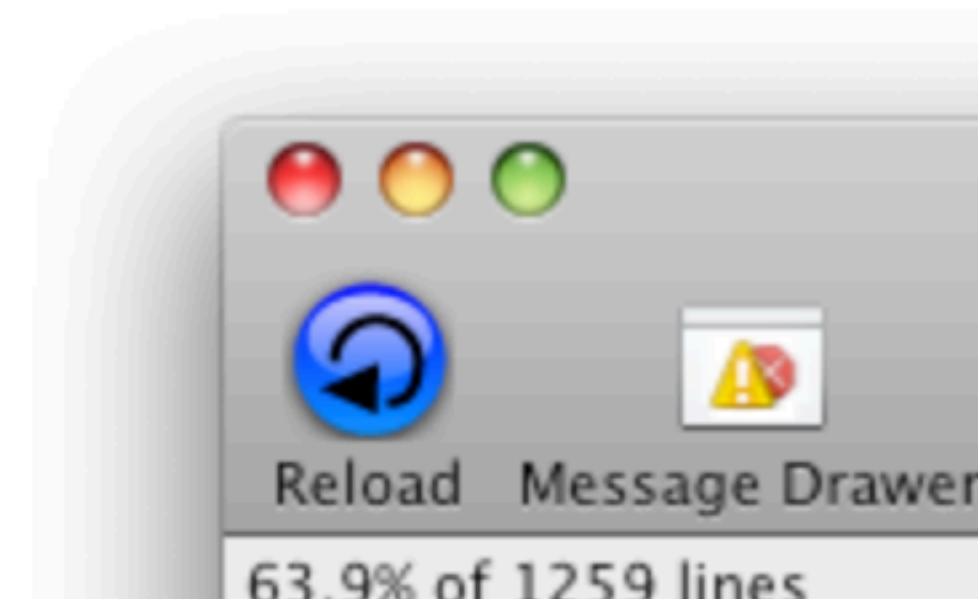
[Project feeds](#)

Code license

[Apache License 2.0](#)

. . .

CoverStory allows you to easily



Name	Klassen	Verzweigungen	Dateien	Zeilen
	94% <div><div>322/343</div></div>	68% <div><div>1737/2541</div></div>	94% <div><div>322/343</div></div>	91% <div><div>7991/8745</div></div>

Aufschlüsselung der Testabdeckung nach Datei

Name	Klassen	Verzweigungen	Zeilen
CCSprite+UIImage.m	0% <div><div>0/1</div></div>	-	0% <div><div>0/12</div></div>
ESEntities+Singleton.m	100% <div><div>1/1</div></div>	100% <div><div>2/2</div></div>	100% <div><div>9/9</div></div>
ESSystems+AddSystem.m	100% <div><div>1/1</div></div>	-	100% <div><div>12/12</div></div>
GXAcademyAllUnitsViewController.m	100% <div><div>1/1</div></div>	50% <div><div>2/4</div></div>	97% <div><div>29/30</div></div>
GXAcademyComponent.m	100% <div><div>1/1</div></div>	-	100% <div><div>3/3</div></div>
GXAcademyUnitCell.m	100% <div><div>1/1</div></div>	88% <div><div>7/8</div></div>	97% <div><div>38/39</div></div>
GXAcademyUnitDetailsViewController.m	100% <div><div>1/1</div></div>	50% <div><div>11/22</div></div>	98% <div><div>59/60</div></div>
GXAcademyUnitInTrainingViewController.m	100% <div><div>1/1</div></div>	50% <div><div>1/2</div></div>	95% <div><div>36/38</div></div>
GXAcademyViewController.m	100% <div><div>1/1</div></div>	100% <div><div>2/2</div></div>	100% <div><div>36/36</div></div>
GXActionLabelComponent.m	100% <div><div>1/1</div></div>	-	100% <div><div>12/12</div></div>
GXActionLabelFrameName.m	100% <div><div>1/1</div></div>	83% <div><div>5/6</div></div>	100% <div><div>11/11</div></div>
GXActionLabelSystem.m	100% <div><div>1/1</div></div>	81% <div><div>21/26</div></div>	96% <div><div>47/49</div></div>
GXAddGroundSystem.m	100% <div><div>1/1</div></div>	69% <div><div>11/16</div></div>	100% <div><div>31/31</div></div>
GXAddGuardRespawnTimerSystem.m	100% <div><div>1/1</div></div>	50% <div><div>3/6</div></div>	94% <div><div>16/17</div></div>
GXAddProgressBarSystem.m	100% <div><div>1/1</div></div>	50% <div><div>3/6</div></div>	100% <div><div>20/20</div></div>
GXAddResourcesPartiallyComponent.m	100% <div><div>1/1</div></div>	-	100% <div><div>15/15</div></div>

Jenkins CI / Cobertura Plugin



<https://travis-ci.org/>

MattesGroeger / MGBenchmark66%

BRANCH: DEVELOP




GITHUB REPO

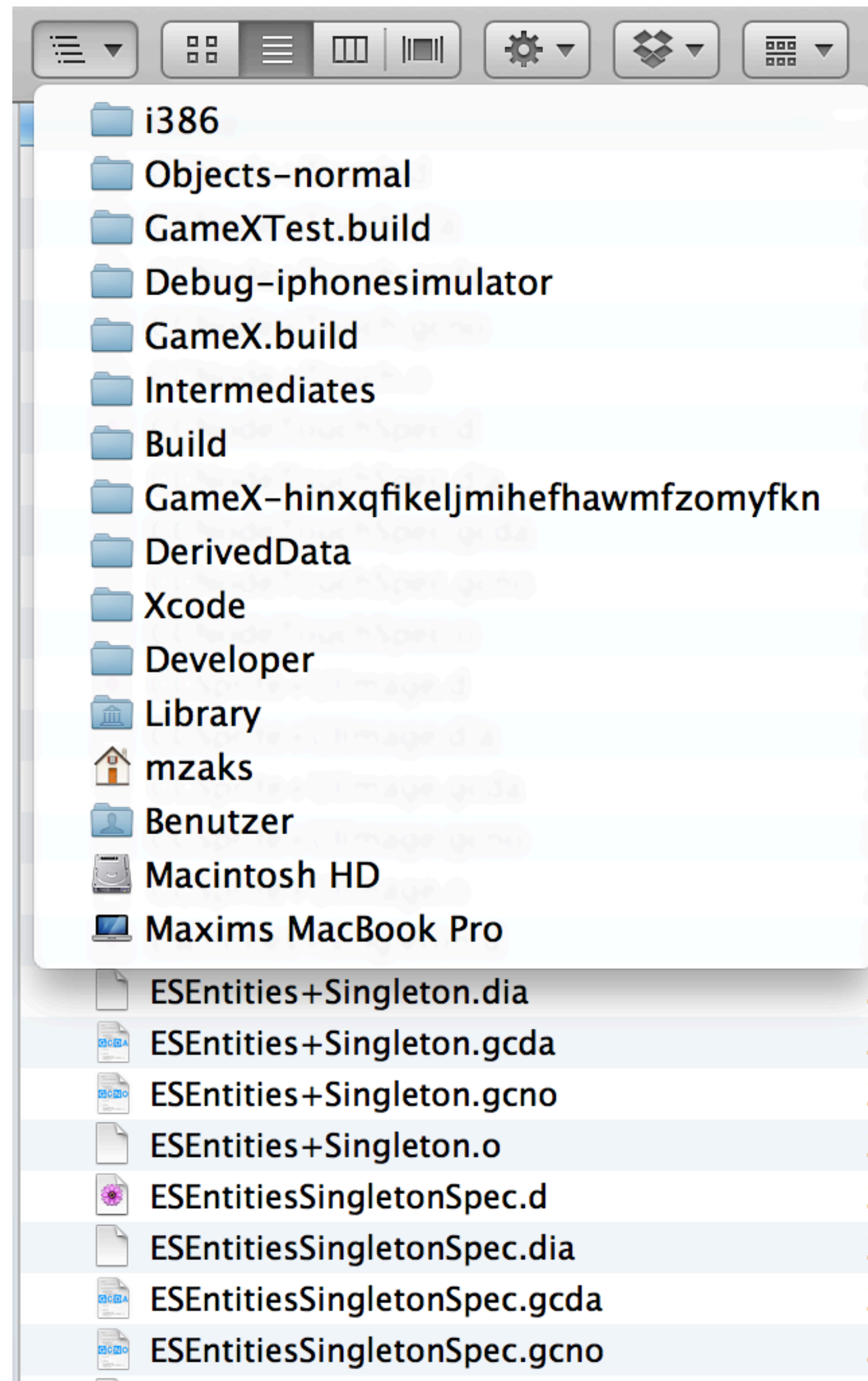
PRIVATE REPOS

BUILD	BRANCH	COVERAGE	COMMIT	COMMITTER	TYPE	TIME	VIA
#29	develop	66%	Make build fail when test or compilation fails	MattesGroeger	push	-	travis-ci
#28	develop	66%	Add setup rake task	MattesGroeger	push	-	travis-ci
#27	develop	66%	Removed unused coveralls shell script	MattesGroeger	push	-	travis-ci
#26	develop	66%	Exclude example folder for coverage	MattesGroeger	push	-	travis-ci
#25	develop	66%	Ignore Pods folder for code coverage	MattesGroeger	push	-	travis-ci
#24	develop	66%	Generate report files with rake script instead	MattesGroeger	push	-	travis-ci
#23	develop	100%	Using XCTool rather than xcodebuild	MattesGroeger	push	-	travis-ci

Wie funktioniert Testabdeckung?

▼ Apple LLVM 5.0 – Code Generation

Setting	 GameXTest
Accelerated Objective-C Dispatch	Yes ⬆️⬆️
Debug Information Level	Compiler default ⬆️⬆️
Enable Additional Vector Extensions	Platform default –
Enforce Strict Aliasing	Yes ⬆️⬆️
Generate Debug Symbols	Yes ⬆️⬆️
Generate Position-Dependent Code	No ⬆️⬆️
 Generate Test Coverage Files	Yes ⬆️⬆️
Inline Methods Hidden	Yes ⬆️⬆️
 Instrument Program Flow	Yes ⬆️⬆️



Einfach ... aber ... XCode5

__gcov_flush()

<http://stackoverflow.com/questions/18394655/xcode5-code-coverage-from-cmd-line-for-ci-builds/18977454#18977454>

Shell ausführen



Befehl

```
cd ${WORKSPACE}/GameX/build/GameXTest.build/Objects-normal/i386/  
  
python ${WORKSPACE}/GameX/Scripts/gcovr -r ${WORKSPACE}/GameX/GameX/src -e  
.*Spec.m\|.*Test.m\|.*.h --xml > ${WORKSPACE}/GameX/build/coverage.xml
```

[Liste der verfügbaren Umgebungsvariablen](#)

Löschen

<http://gcovr.com>

Wann ist hohe Testabdeckung
eine gute Strategie

Fakten aus unseren Projekt

Das Projekt

- Laufdauer - über ein Jahr
- 3 iOS Entwickler
- Agile Entwicklung
- Anzahl der Kompilierten Quelldateien für Production Target 465
- Anzahl der Kompilierten Quelldateien für Test Target 795

Codezeilen

- Gesamte Anzahl der Codezeilen 32615 davon 97% Abgedeckt
- Gesamte Anzahl der Codezeilen wenn Test und .h Dateien nicht erfasst werden 8745 davon 91% Abgedeckt
- Folgerung nur etwa 25% der Codezeilen befinden sich in Produktiven .m Dateien

Schlussfolgerung:
Tests machen Dich nicht
schneller

Weitere Fakten aus unseren Projekt

Leistung

- Robust gegen Refactoring und Änderung
- Schnelle Einarbeitung neuer Kollegen
- Sehr geringe Anzahl von Bugs

Schlussfolgerung:
Tests geben Dir mehr Leistung

Meine ACID Theorie

ACID

siehe <http://de.wikipedia.org/wiki/ACID>

- Atomic = Nichts ist abhängig von deinen Code
- Consistent = Es ist Eindeutig was, wie und warum es tut, was es tut
- Isolated = Es ist absolut unabhängig von Anderen
- Durable = Es wird nie verändert

Eine weniger strenge Theorie

Sprint vs. Marathon

Taktiken für hohe Testabdeckung

Falls ihr im Team arbeitet,
jeder muss mitziehen!

Pair Programming hilft!

Unit vs. Integration Test
Scharfschutze vs. Streubombe

Testabdeckung

- Mit Integration Test (Streubombe) erreicht man schnell hohe Testabdeckung (alte gute 80 / 20 Regel)
- Um eine Testabdeckung über 90% zu erreichen braucht man Scharfschützen an jeder Ecke

Unit Test Framework

Wir benutzen Kiwi

- Basiert auf OCUnit / XCTest
- Testbeschreibungssprache ist angenehm
 - describe what you expect it should do (RSpec style)
 - Wir ergänzen mit (given, expectations, when, then) Kommentaren
- Mocking und Matching ist integriert

Das Hauptrezept eines UnitTests
ist Isolierung

Reine Funktion

- Eingabe A resultiert immer in Ausgabe B
- Keine Seiteneffekte - es wird Nichts im Hintergrund gemacht

```
it(@"should convert snake case string to camel case string", ^{  
    // given  
    NSString *string = @"some_string_in_snake_case";  
  
    // when  
    NSString *convertedString = [string snakeCaseToCamelCase];  
  
    // then  
    [[convertedString should] equal:@"someStringInSnakeCase"];  
});
```



```
it(@"should handle empty string", ^{  
    // when  
    NSString *convertedString = [@" " snakeCaseToCamelCase];  
  
    // then  
    [[convertedString should] equal:@""];  
});
```

```
it(@"should still work even if converting a string several times (used to test
caching)", ^{
    // given
    NSString *string = @"some_string_in_snake_case";

    // when
    NSString *convertedString1 = [string snakeCaseToCamelCase];
    NSString *convertedString2 = [string snakeCaseToCamelCase];
    NSString *convertedString3 = [string snakeCaseToCamelCase];

    // then
    [[convertedString1 should] equal:@"someStringInSnakeCase"];
    [[convertedString2 should] equal:@"someStringInSnakeCase"];
    [[convertedString3 should] equal:@"someStringInSnakeCase"];
});
```

Delegieren stat selber machen

- `@property` (Dependency Injection)
- Mit Erwartungen Arbeiten

```
__block GXRecruitmentService *service;
__block GXBackendInterface *backendInterface;

beforeEach(^{
    service = [GXRecruitmentService new];
    backendInterface = [GXBackendInterface mockWithName:@"bi"];

    service.backendInterface = backendInterface;
});
```

```
it(@"should call add_unit in the backend", ^{
    // given

    // expectation
    [[backendInterface should] receive:@selector(postMethod:parameters:)
withCount:1 arguments:ADD_UNIT_METHOD, @{
        @"position": @{
            @"x": @20,
            @"y": @21
        },
        @"type": @"unit1"
    }];

    // when
    [service addUnitOfType:@"unit1" toBarrackAtPosition:CGPointMake(20, 21)];
});
```

Geburtenkontrolle

- Stub alloc und init Methoden
- Stub Klassen Methoden
- Manchmal muss man sogar eine Extraschicht spendieren


```
__block GXBackendInterface *backendInterface;
__block AFHTTPClient *httpClient;

beforeEach(^{

    httpClient = [AFHTTPClient nullMockWithName:@"httpClient"];
    [AFHTTPClient stub:@selector(alloc) andReturn:httpClient];

});
```

```
it(@"should set the base URL on init", ^{
    // given
    NSURL *url = [[NSURL alloc] initWithString:BACKEND_SERVICE_URL];

    // expectations
    [[httpClient should] receive:@selector(initWithBaseURL:) withArguments:url];

    // when
    backendInterface = [GXBackendInterface new];

});
```


Was mache ich mit Callback blocks?

- Auf Parameter spionieren
- Callback selber aufrufen

```
- (void) postMethod:(NSString *)method parameters:(NSDictionary *)parameters
success:(void (^)(NSDictionary *))successBlock {
    ...

    [_httpClient postPath:nil
parameters:requestParams
success:^(AFHTTPRequestOperation *operation, id responseObject) {
    ...
}]
failure:^(AFHTTPRequestOperation *operation, NSError *error) {
    if (![operation isCancelled]) {

        UIAlertView *alertView = [[UIAlertView alloc] initWithTitle:@" "
                                                                    message:@" "
                                                                    delegate:self
                                                                    cancelButtonTitle:@"Reload"
                                                                    otherButtonTitles:nil];

        [alertView show];
    }
}];
```

```
__block void (^failureBlock)(AFHTTPRequestOperation *, id);

beforeEach(^{
    KWCaptureSpy *failureSpy = [(id) httpClient
captureArgument:@selector(postPath:parameters:success:failure:) atIndex:3];

    [backendInterface postMethod:@" " parameters:@{ } success:^(NSDictionary
*responseObject) {
        }];

    failureBlock = failureSpy.argument;

    operation = [AFHTTPRequestOperation nullMockWithName:@"operation"];
});
```

```
it(@"should show an alert", ^{

    // given
    [[operation stubAndReturntheValue(NO)] isCancelled];
    UIAlertView *alertView = [UIAlertView mockWithName:@"alertView"];
    [UIAlertView stub:@selector(alloc) andReturn:alertView];
    [alertView
stub:@selector initWithTitle:message:delegate: cancelButtonTitle:otherButtonTitles:) andReturn:alertView];

    // expectations
    [[alertView should] receive:@selector(show)];

    // when
    failureBlock(operation, nil);

});
```

Kann ich etwas nicht Testen?

- Aufruf von C Funktionen
- Wenn Code auf dem Bildschirm Zeichnet

Kleine Tips

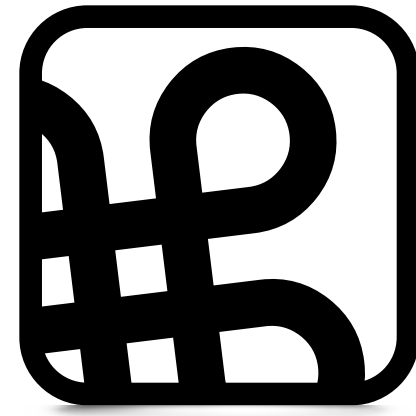
- Given, When, Then Kommentare helfen
- Versuche Wiederverwendung in Tests zu erzielen
- Erstelle Templates um Klassen direkt mit Specs zu erstellen

Fazit

- Hohe Testabdeckung erhöht die Leistung (keine Geschwindigkeit)
- Beantwortet folgende Fragen:
 - Muss ich ein Sprint oder ein Marathon laufen?
 - Kann ich mein Code isolieren?

Fragen?

Vielen Dank
@iceX33



Macoun