Test-Driven Development

droga od podstaw do biegłości

w rozwijaniu dużych aplikacji

Krzysztof Jelski

PRAGMATISTS

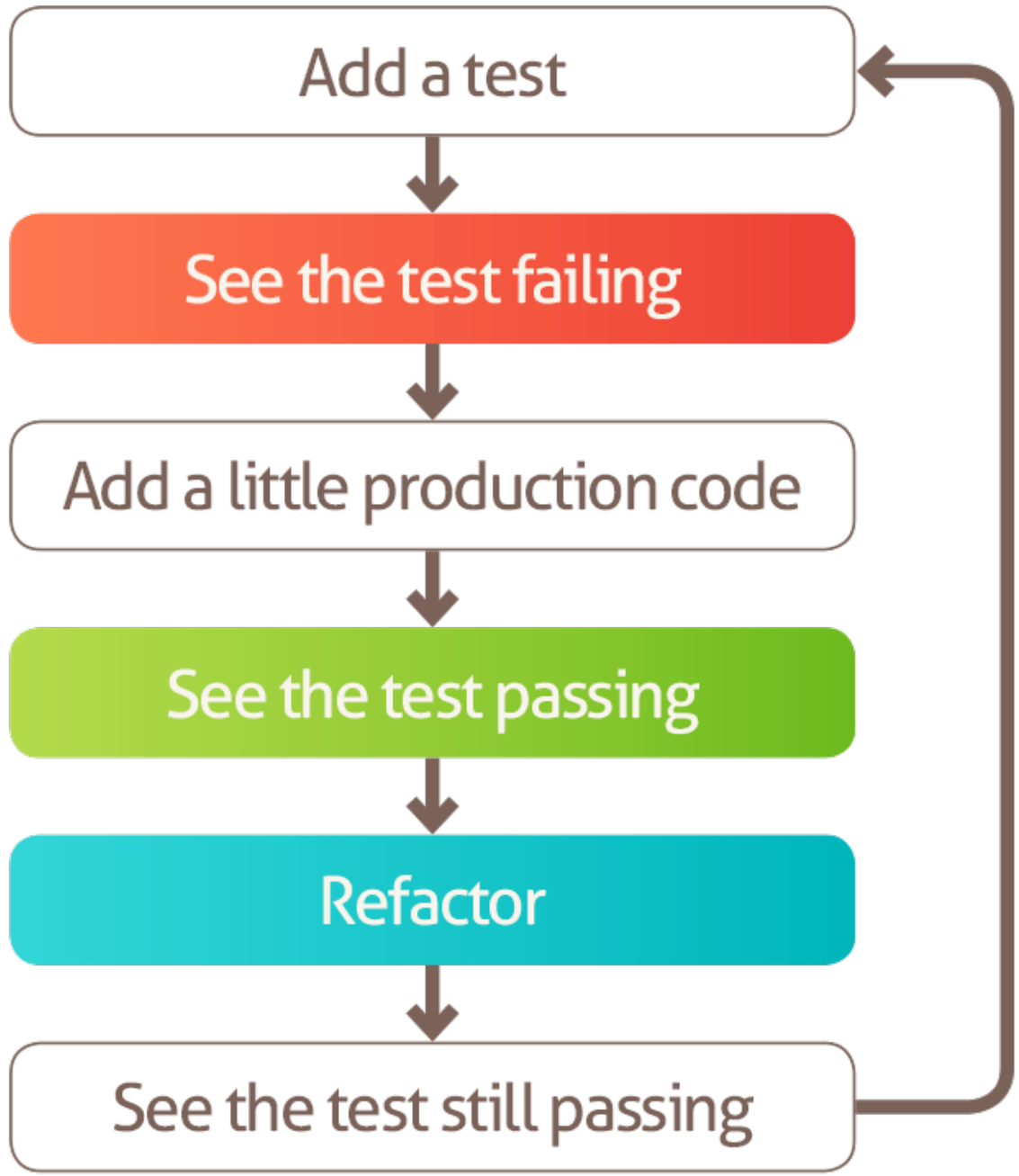
O mnie

Szkolenia

Software

PRAGMATISTS

TDD


Add a test
See the test failing
Add a little production code
See the test passing
Refactor
See the test still passing

Odcinek 1

TDD newbie


DELL
AXIM
Internet Explorer
13:41
http://pl.wikipedia.org/wiki/Pocket_
Z Wikipedii, wolnej encyklopedii.
Pocket PC jest przenośnym komputerem mieszczącym się w dłoni, działającym pod kontrolą systemu operacyjnego Microsoft Windows Mobile (opartego na Windows CE z .NET).
Na rynku istnieją wersje Pocket PC z wbudowanym telefonem komórkowym, odbiornikiem GPS i innymi urządzeniami.
View Tools
Pocket PC

TDD
C#
.NET CF


DELL
AXIM
Internet Explorer
13:41
http://pl.wikipedia.org/wiki/Pocket_
Z Wikipedii, wolnej encyklopedii.
Pocket PC jest przenośnym komputerem mieszczącym się w dłoni, działającym pod kontrolą systemu operacyjnego Microsoft Windows Mobile (opartego na Windows CE z .NET).
Na rynku istnieją wersje Pocket PC z wbudowanym telefonem komórkowym, odbiornikiem GPS i innymi urządzeniami.
View Tools
Pocket PC

Ciekawość

Oczekiwania

Nieznajomość C#

TDD
C#
.NET CF

Ciekawość

Oczekiwania

Nieznajomość C#

TDD
C#
.NET CF

Pewność

C# level up!

Przyjemność z pracy

Lesson learnt

#1 Nie trzeba dużo by zacząć

TDD
C#
.NET CF

TDD
C#
.NET CF

Moja ocena

TDD
C#
.NET CF

Moja ocena

Moja decyzja

TDD
C#
.NET CF

Moja ocena

Moja decyzja

Moja odpowiedzialność

Postawa | Technika | Droga

Lesson learnt

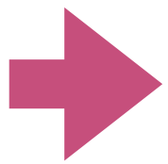
#2 TDD to odpowiedzialność developera

Odcinek 2

TDD zealot

Testy, którym mogę zaufać

GUI 

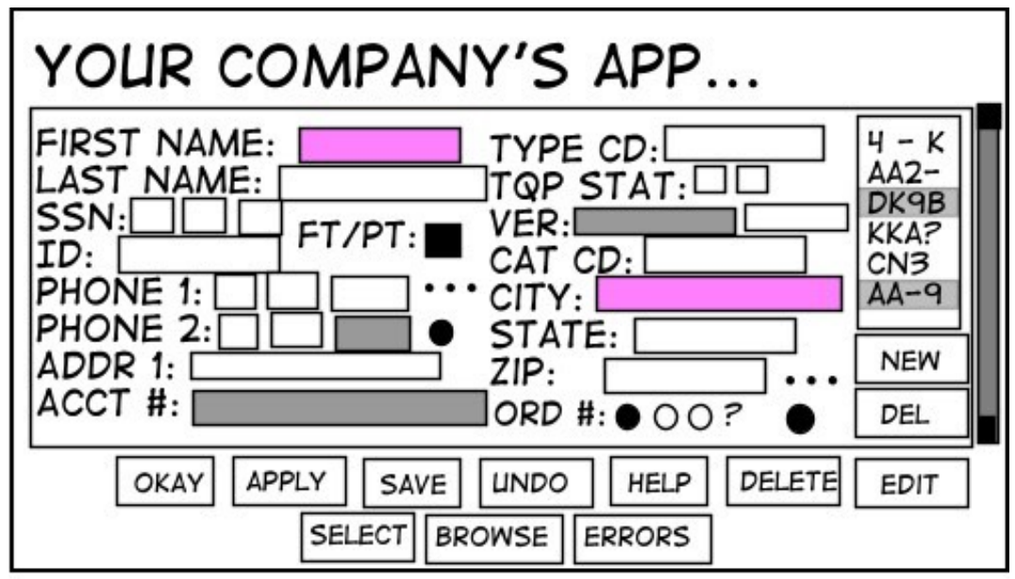


DB 

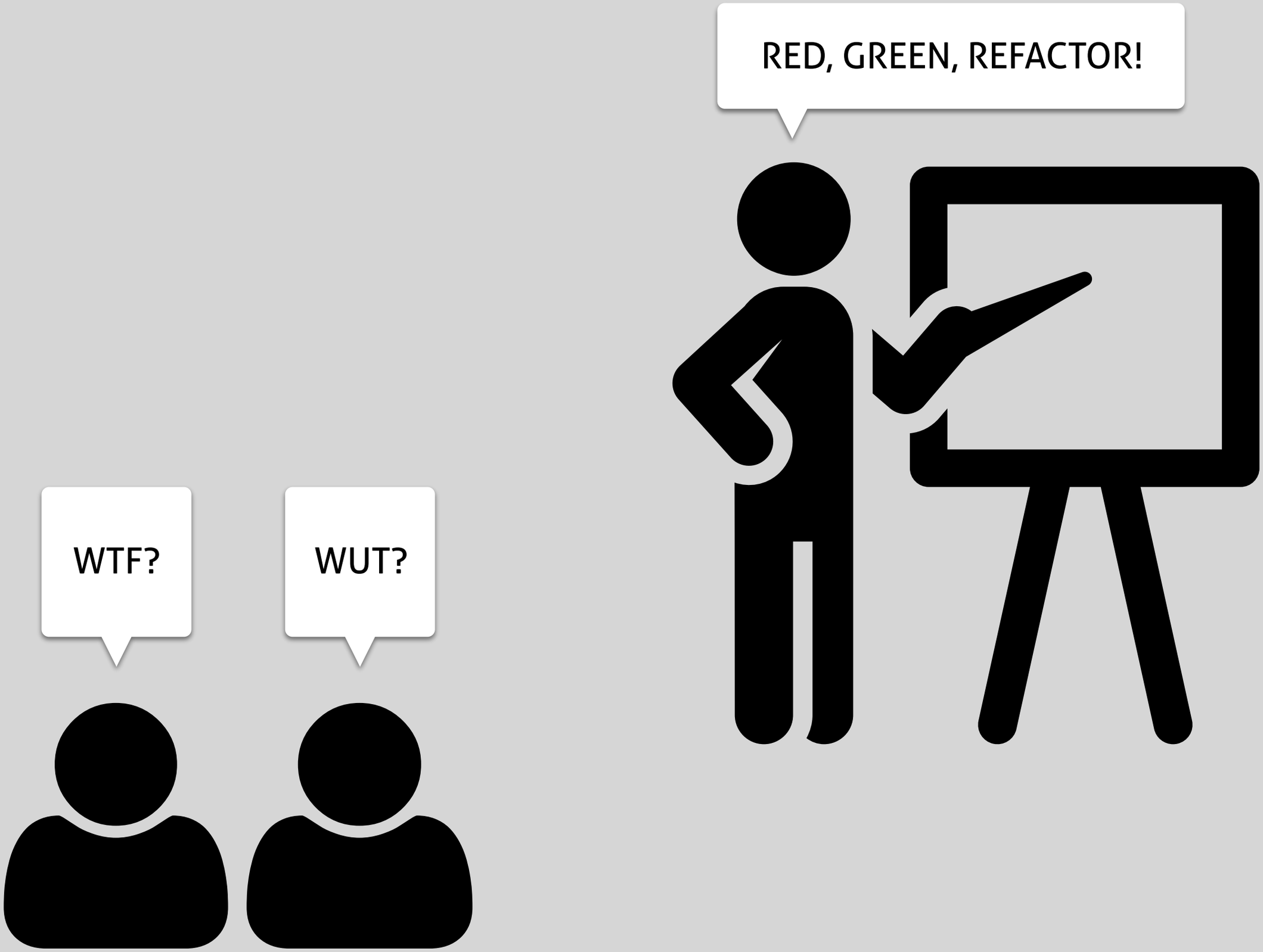
Testy, którym mogę zaufać

GUI

DB



Zdalny Pair Programming


RED, GREEN, REFACTOR!
WTF?
WUT?

Lesson learnt

#3 Nie jest łatwo nauczyć TDD *

* - zwłaszcza, gdy ktoś nie chce

Java

Wicket

10 osób

2 z doświadczeniem
w TDD

Postawa | Technika | Droga

Lesson learnt

#4 Nie narzucaj TDD

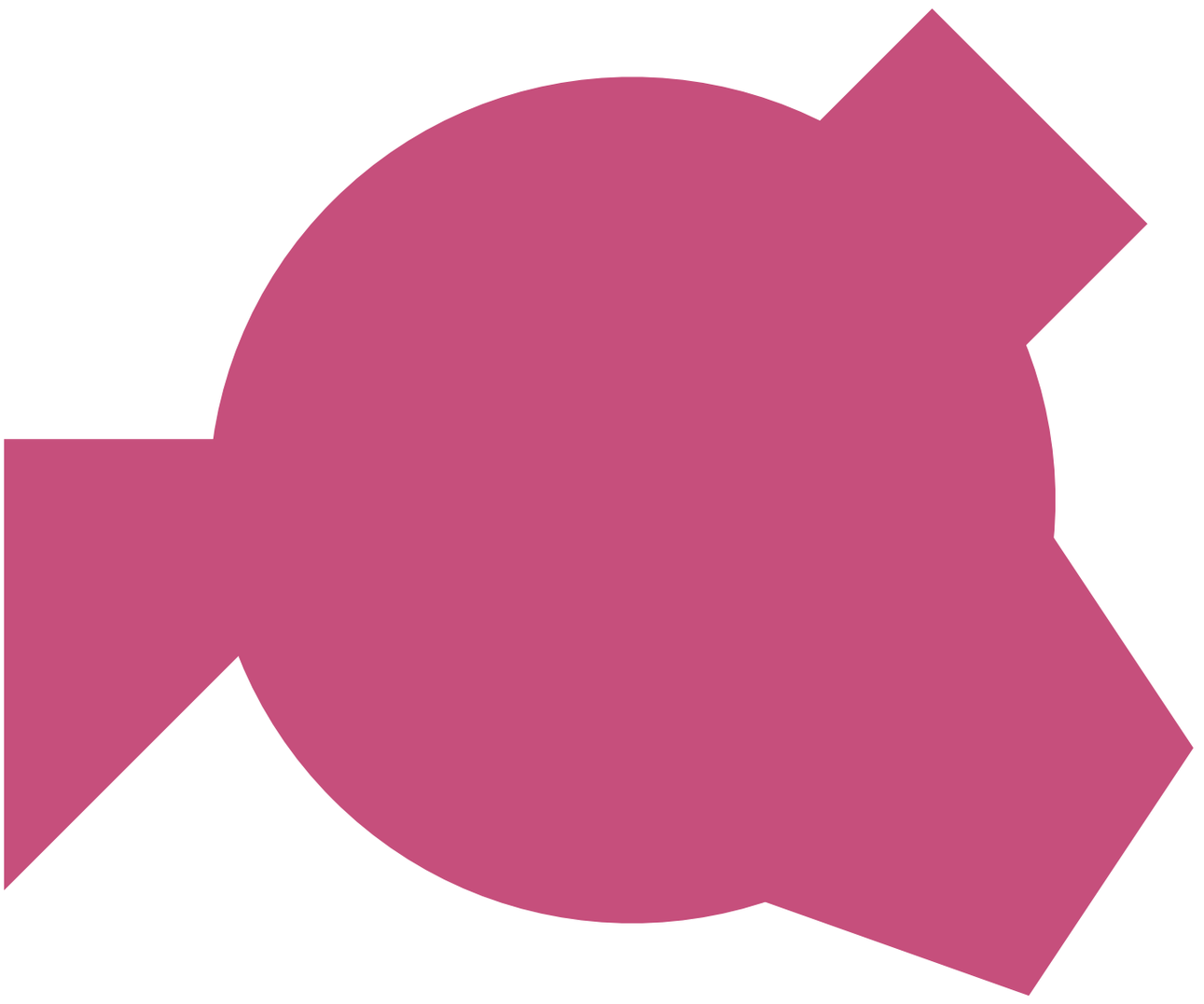
Duży test-suite potrafi boleć

Duży test-suite potrafi boleć

621 tests failed

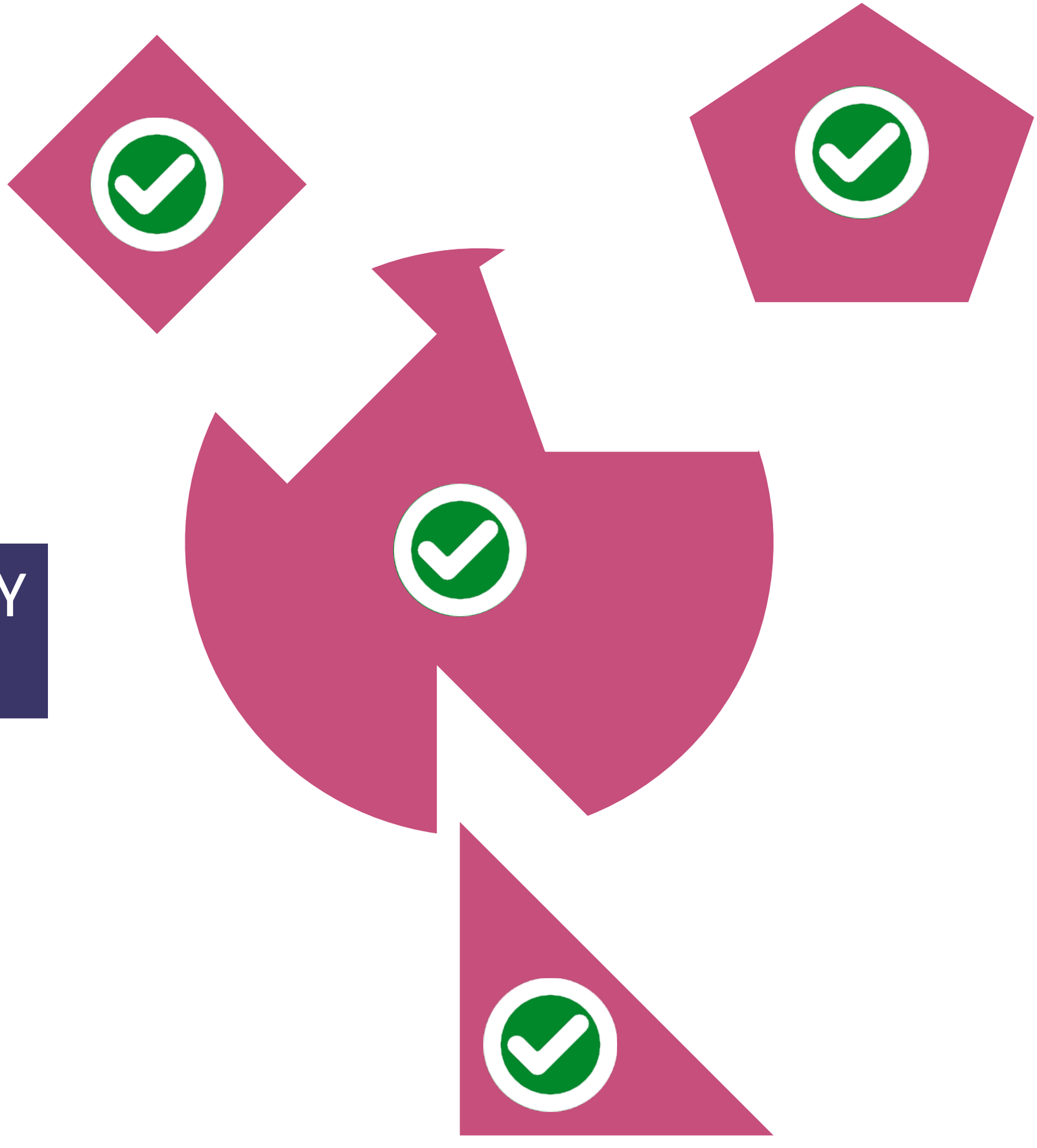
Lesson learnt

#5 Dbaj o kod testowy tak jak o produkcyjny



DEPENDENCY INJECTION

DEPENDENCY INJECTION



Lesson learnt

#6 Constructor Injection zwiększa testowalność

Odcinek 3

TDD practitioner

9,496 tests (+8)

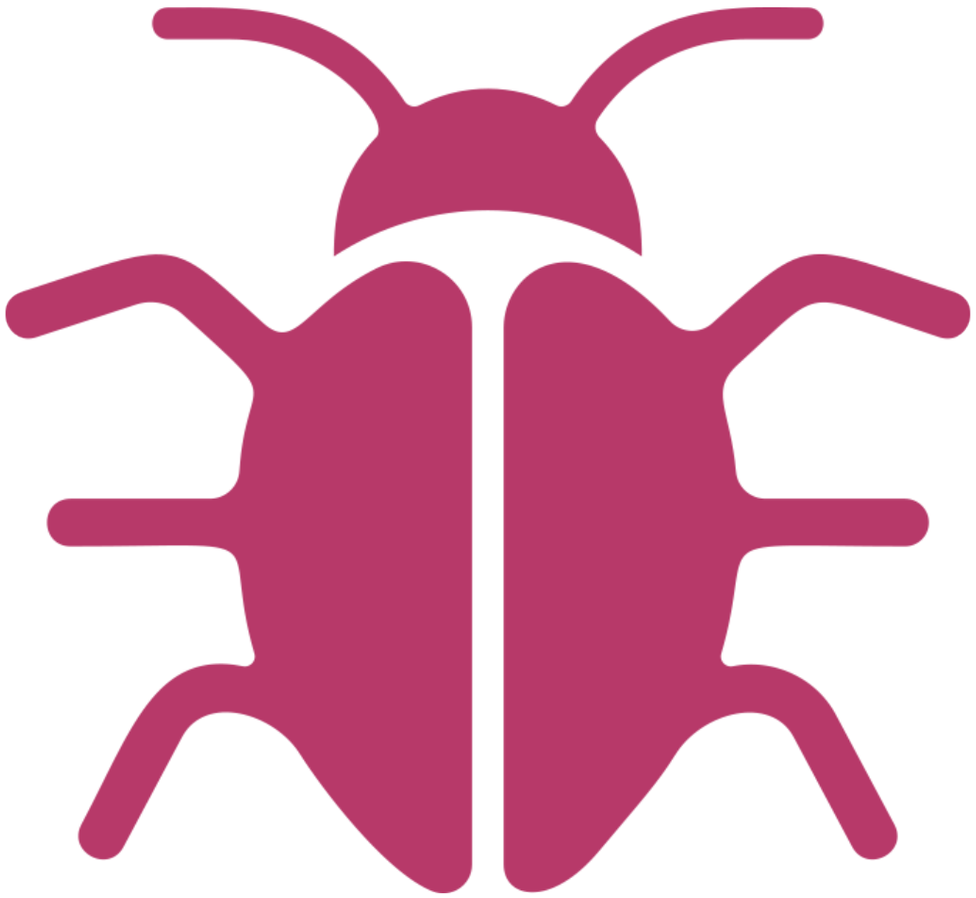
Took 3 min 4 sec.

Lesson learnt

#7 Dobrze utrzymany zestaw testów daje moc

9,496 tests

9,496 tests



Jakiego rodzaju testu zabrakło?

```
server = new Server(port);

System.setProperty("PORT", String.valueOf(port));

WebAppContext context = new WebAppContext();
context.setConfigurations(new Configuration[]{new WebXmlConfiguration()});
context.setResourceBase(calculateResourceBase());
server.setHandler(context);
```

```
server = new Server(port);

System.setProperty("PORT", String.valueOf(port));

WebAppContext context = new WebAppContext();
context.setConfigurations(new Configuration[]{new WebXmlConfiguration()});
context.setResourceBase(calculateResourceBase());
server.setHandler(context);
```

```
@Test
@Commit
public void ...
```

```
server = new Server(port);

System.setProperty("PORT", String.valueOf(port));

WebAppContext context = new WebAppContext();
context.setConfigurations(new Configuration[]{new WebXmlConfiguration()});
context.setResourceBase(calculateResourceBase());
server.setHandler(context);
```

```
@Test
@Commit
public void ...
```

```
@Test
public void publishesErrorEventOn500() throws IOException {
    server.enqueue(
        new MockResponse().setStatus("HTTP/1.1 500 Internal Server Error"));
    OfficeGroupsFetcher officeGroupsFetcher = createOfficeGroupsFetcher();

    officeGroupsFetcher.fetch(ANY_ID);

    verify(bus).post(isA(RestServerError.class));
}
```

There is no such thing as untestable code.

Robert „Uncle Bob“ Martin

If you have code that you believe cannot be tested, then I suggest that you:

- Stop, and find a way to test it.
- Find a way to change it so that it's testable.

Robert „Uncle Bob" Martin

Postawa | Technika | Droga

Lesson learnt

#8 Odkrywaj nowe rodzaje testów

Testy są specyfikacją

```
@Test
public void perfect_game() {
    assertEquals(300, frames("XXXXXXXXXXXX"));
}
@Test
public void gutter_game() {
    assertEquals(0, frames("--------------------"));
}
@Test
public void bonus_for_spare_is_next_ball_score() {
    assertEquals(18, frames("5/4-"));
}
@Test
public void bonus_for_strike_is_next_frame_score() {
    assertEquals(10 + 9 + 9, frames("X36"));
}
```

```
@Test
public void detects_clash_on_single_day_in_longer_range() throws Exception
{
    Patient patient = createPatient();
    given(patient).hasPrescriptionFrom(2010, 3, 1)
        .forDays(31).forMedicine("Fluoxetine");
    given(patient).hasPrescriptionFrom(2010, 3, 15)
        .forDays(1).forMedicine("Codeine");

    Collection<LocalDate> clashDates = patient.clash(
        asList("Fluoxetine", "Codeine"));

    assertThat(clashDates).containsExactly(LocalDate.of(2010, 3, 15));
}
```

Lesson learnt

#9 Inwestuj w testowe DSLe

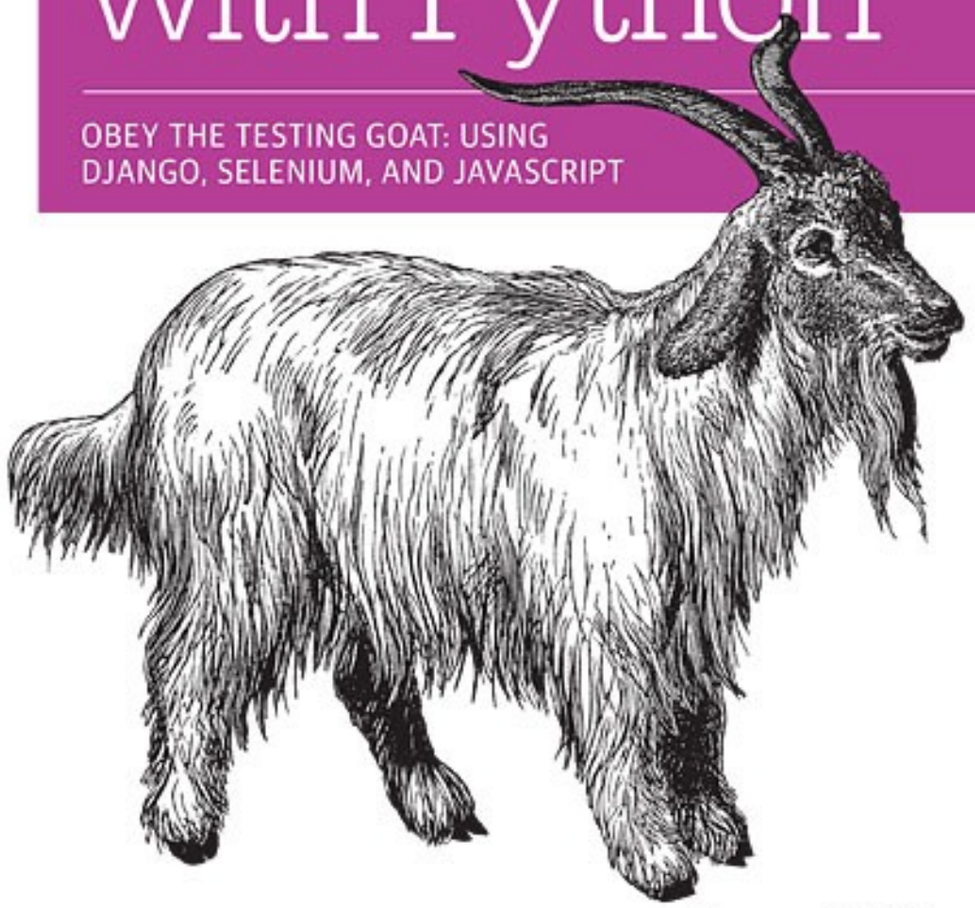
Odcinek 4

TDD mentor

O'REILLY®

Test-Driven Development with Python

OBEY THE TESTING GOAT: USING DJANGO, SELENIUM, AND JAVASCRIPT



Harry J.W. Percival

O'REILLY

Test-
Deve
with

OBEY THE TESTING
DJANGO, SELENIUM,

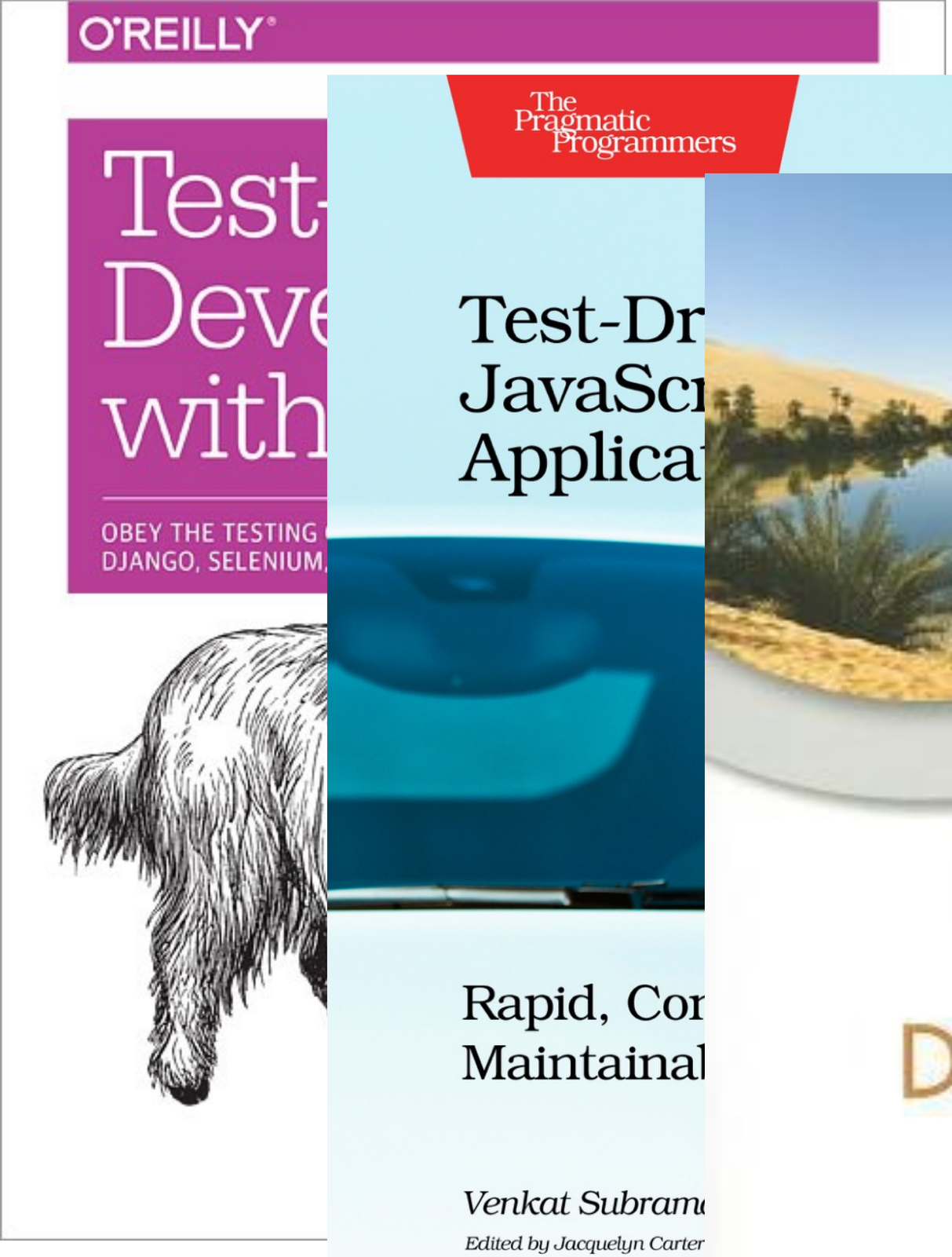
The Pragmatic Programmers

Test-Driving JavaScript Applications

Rapid, Confident, Maintainable Code

Venkat Subramaniam

Edited by Jacquelyn Carter


O'REILLY
Test-
Deve
with
OBEY THE TESTING
DJANGO, SELENIUM,

The Pragmatic Programmers
Test-Dr
JavaScr
Applicat
Rapid, Cor
Maintainal
Venkat Subrama
Edited by Jacquelyn Carter

TEST-DRIVEN
DATABASE
DEVELOPMENT
Unlocking Agility
MAX GUERNSEY, III
Foreword by SCOTT BAIN
NetObjectives
Lean-Agile Series

O'REILLY

Test-
Deve
with

OBEY THE TESTING
DJANGO, SELENIUM

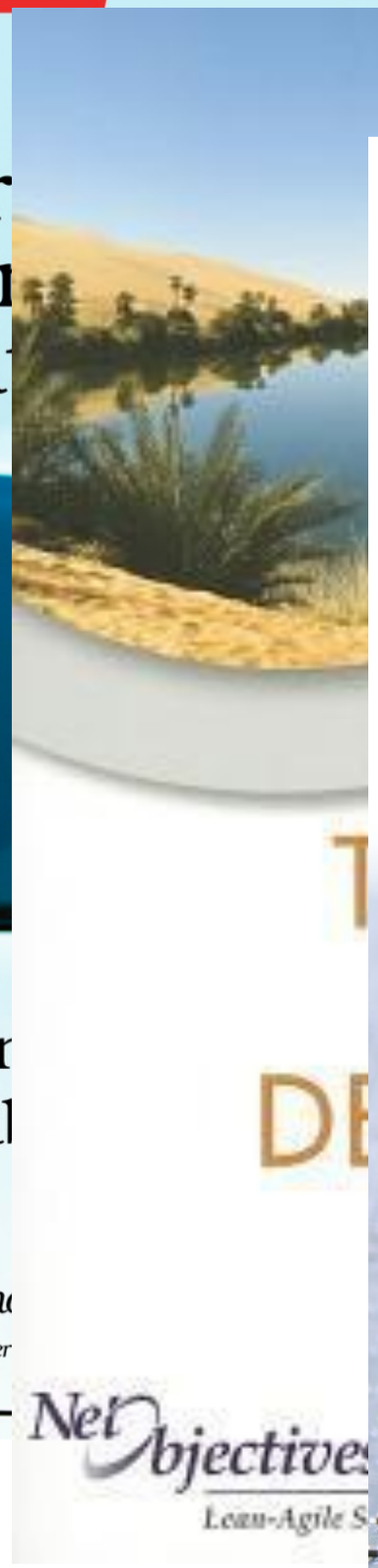
The Pragmatic Programmers

Test-Dr
JavaScr
Applicat

Rapid, Cor
Maintainal

Venkat Subrama

Edited by Jacquelyn Carter



T

DE

NetObjectives

Lean-Agile S

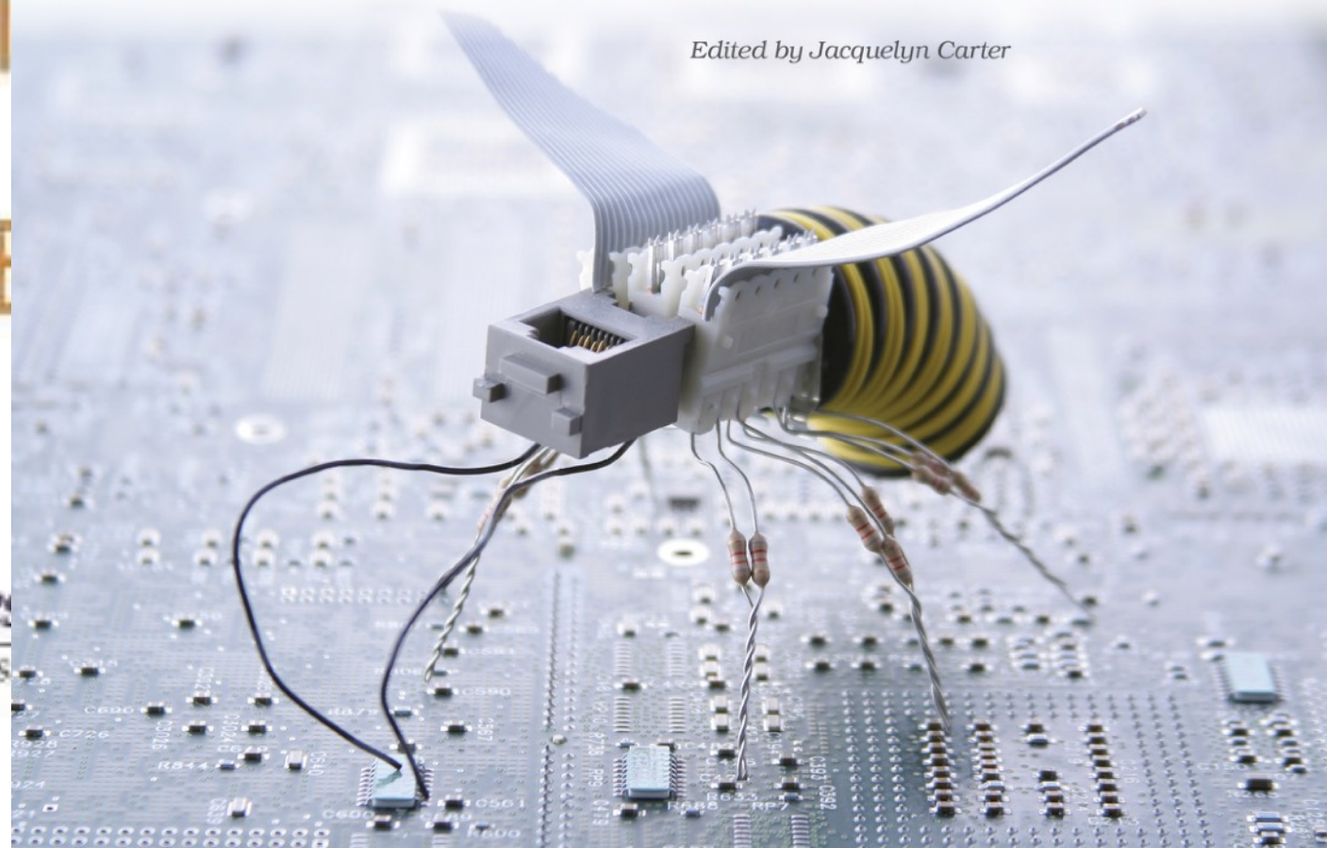
The Pragmatic Programmers

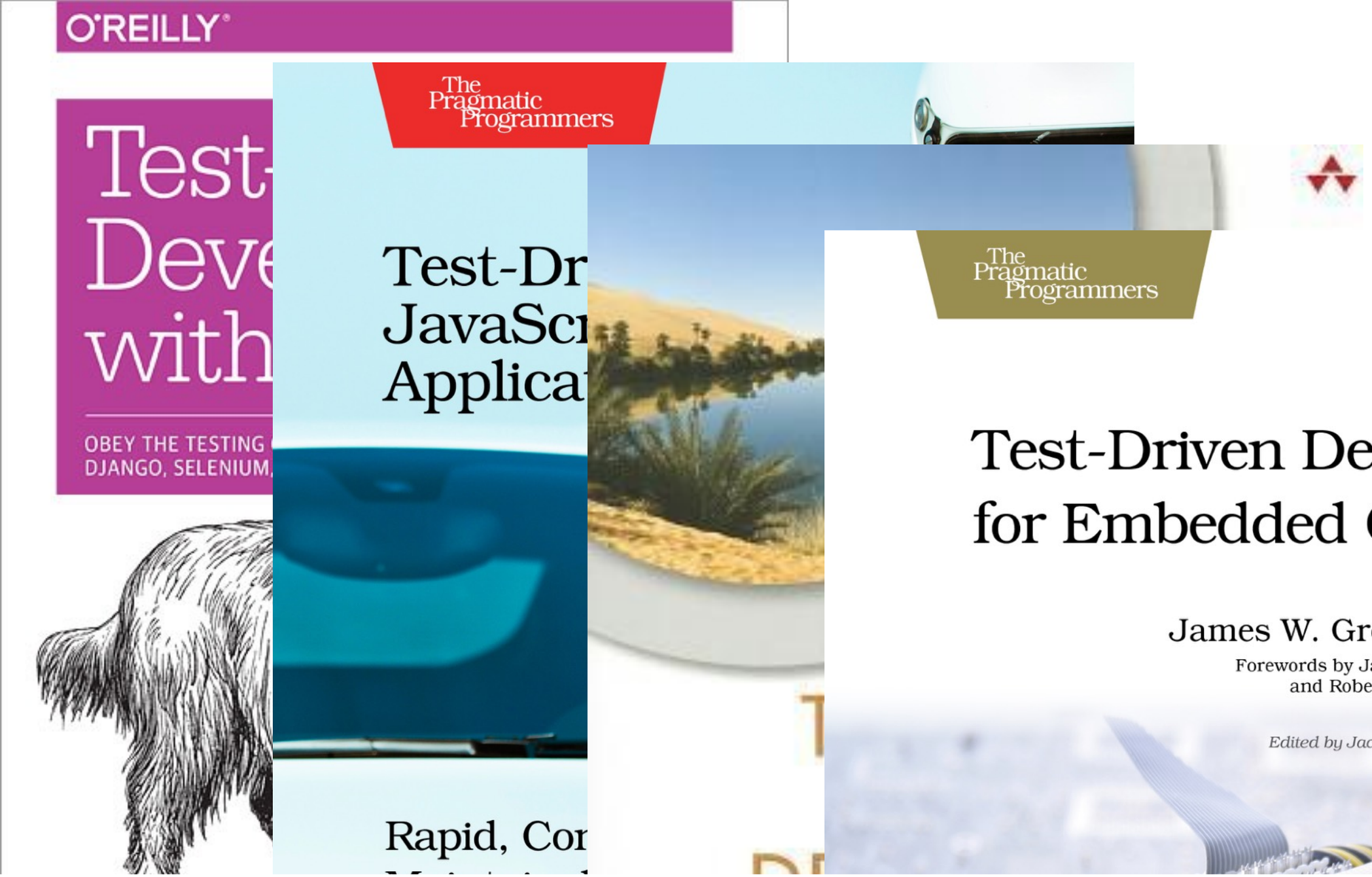
Test-Driven Development for Embedded C

James W. Grenning

Forewords by Jack Ganssle
and Robert C. Martin

Edited by Jacquelyn Carter





APPROACHING OUTSIDE-IN TDD ON ANDROID I

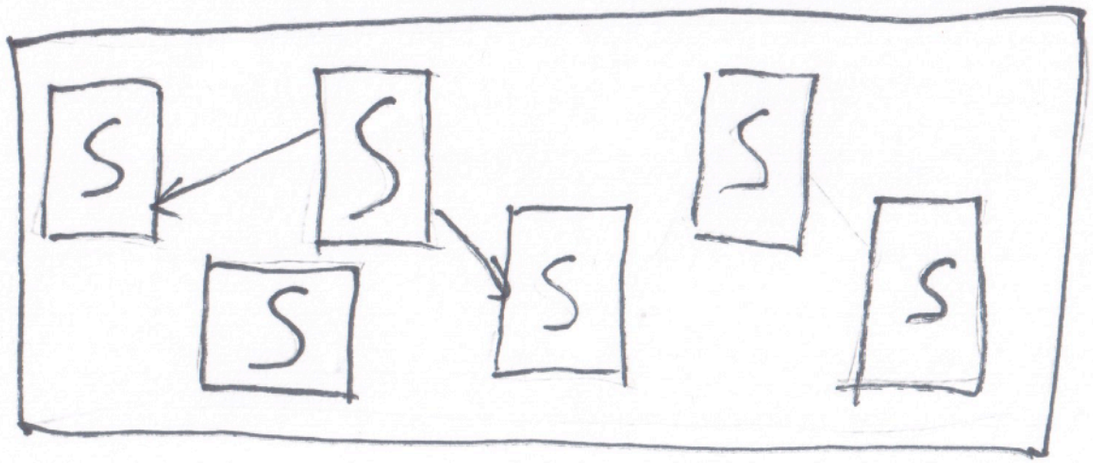
By Carlos Morera de la Chica / Posted 28 Sep 2016 / android tdd design

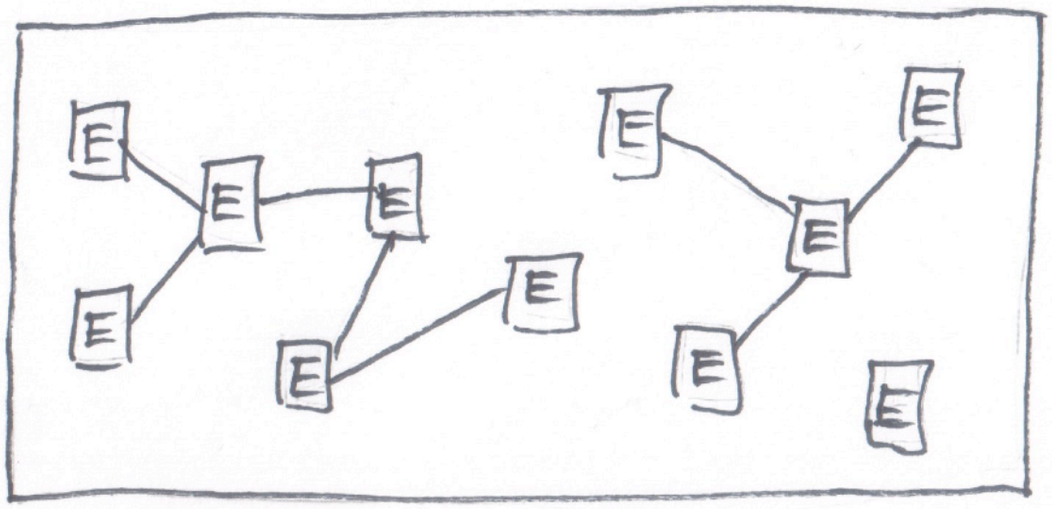
Outside-in Test-Driven Development (TDD) can be a challenge to implement. In this 3-part post series, Christian and I would like to share our experiences applying it to Android development and offer some practical tips for doing so yourself. In this first post of the series we will introduce the necessary concepts and present our broad approach to the problem.

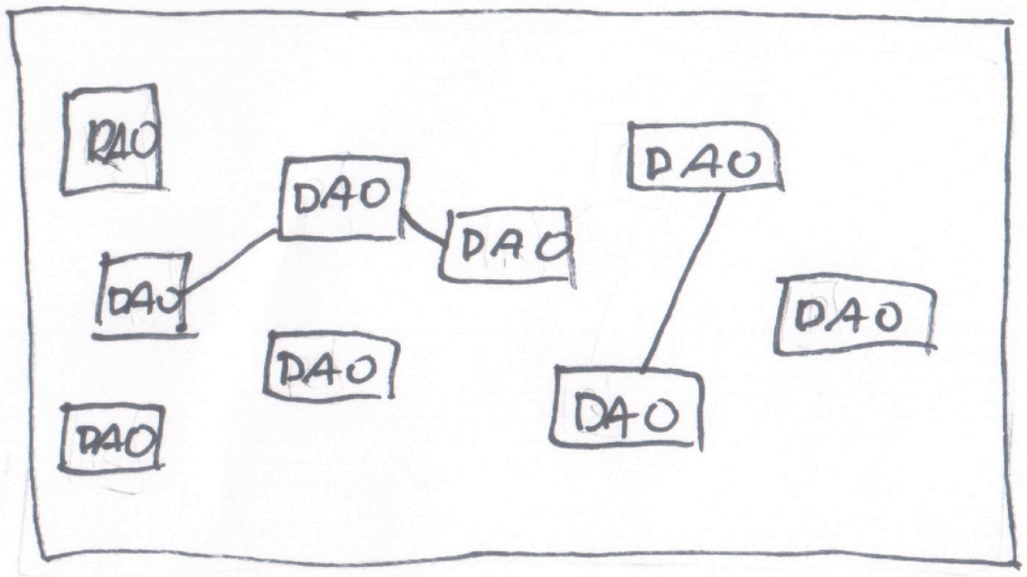
Lesson learnt

#10 TDD jest możliwe w każdej technologii

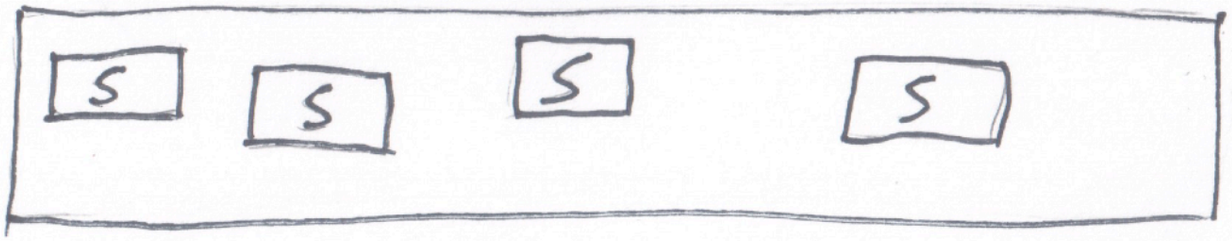
Anemic Domain Model

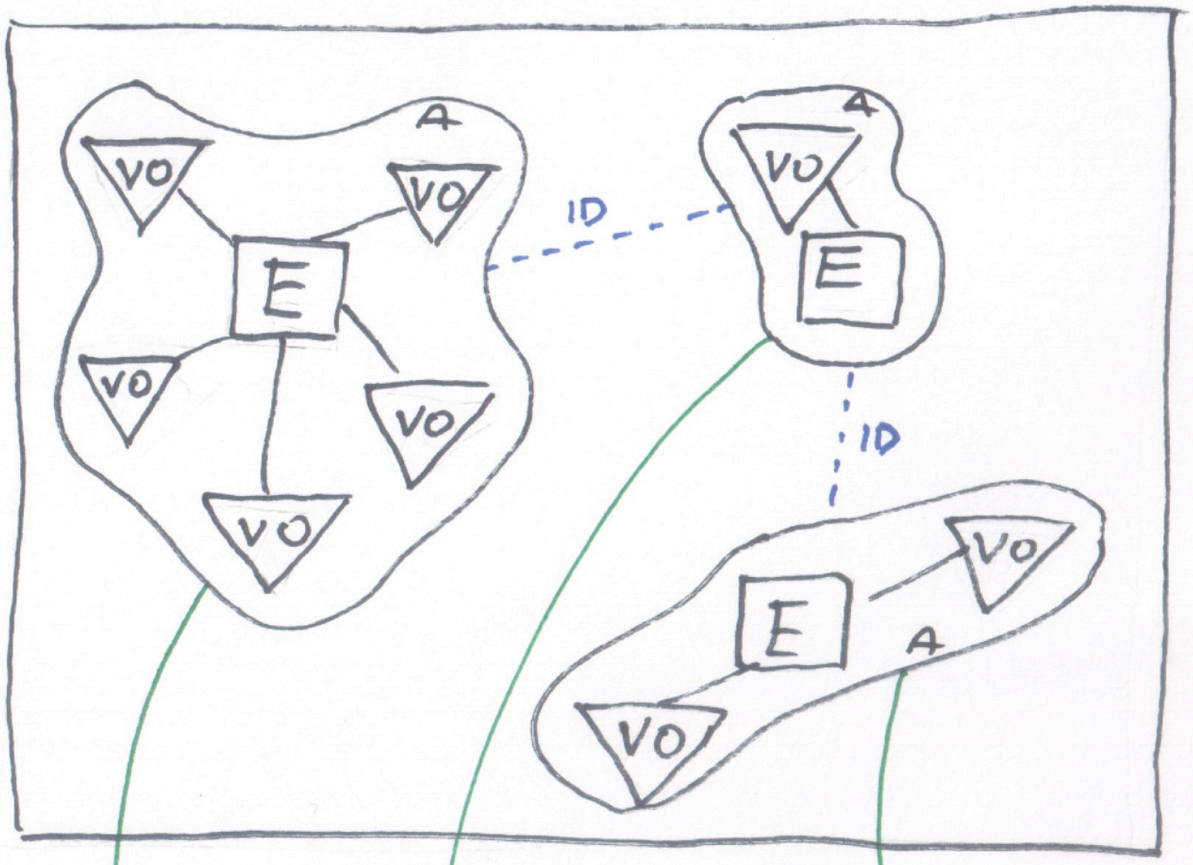


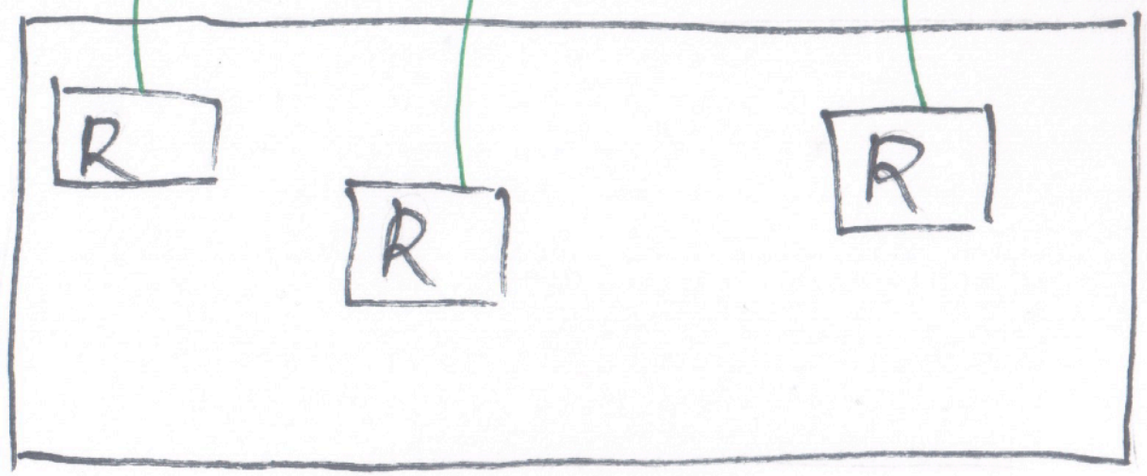




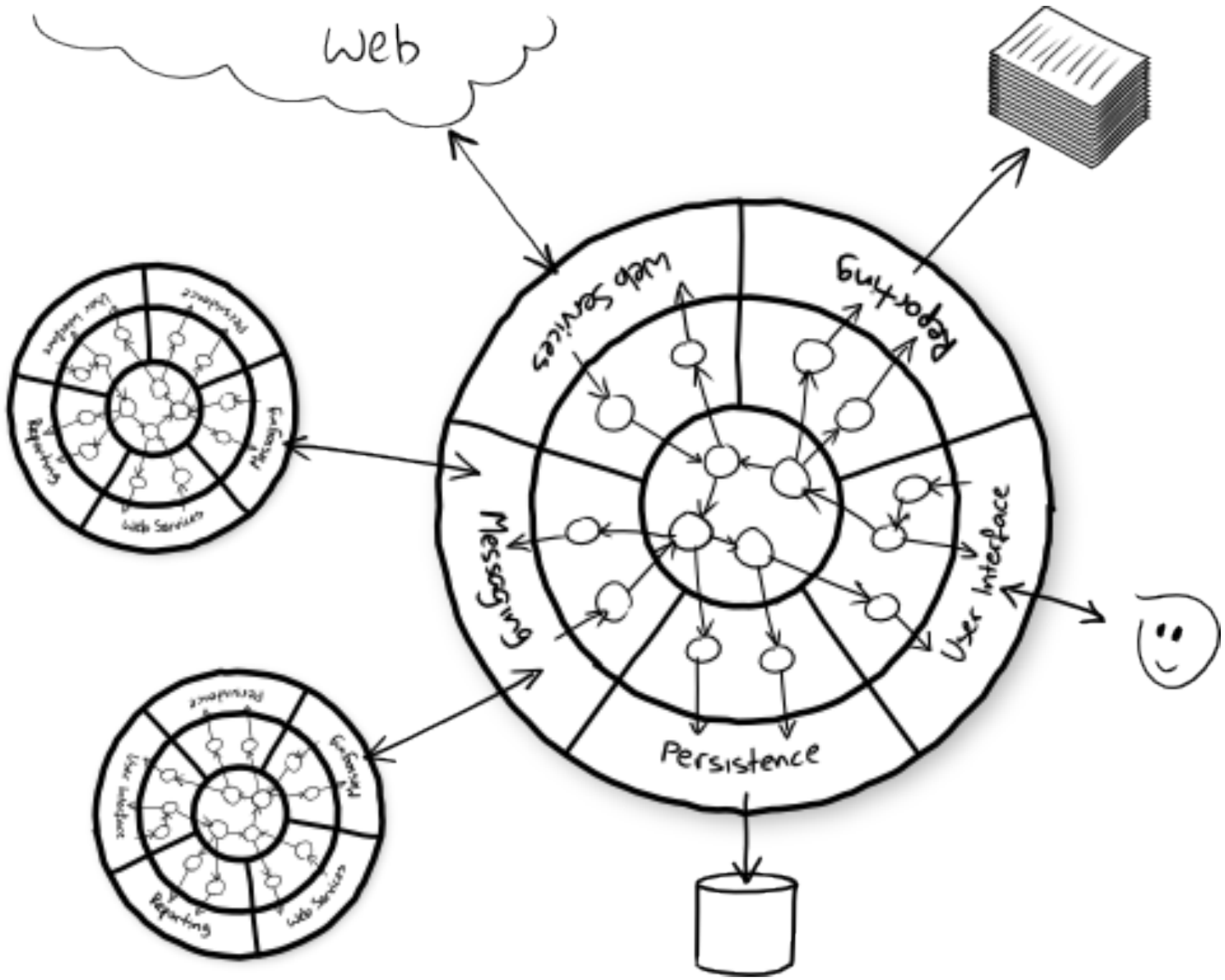
DDD







Ports and adapters



http://www.natpryce.com/articles/000772.html

Postawa | Technika | Droga

Lesson learnt

#11 Stosuj DDD oraz Ports & Adapters

Legacy Code

Tworzenie obiektów

Zależności

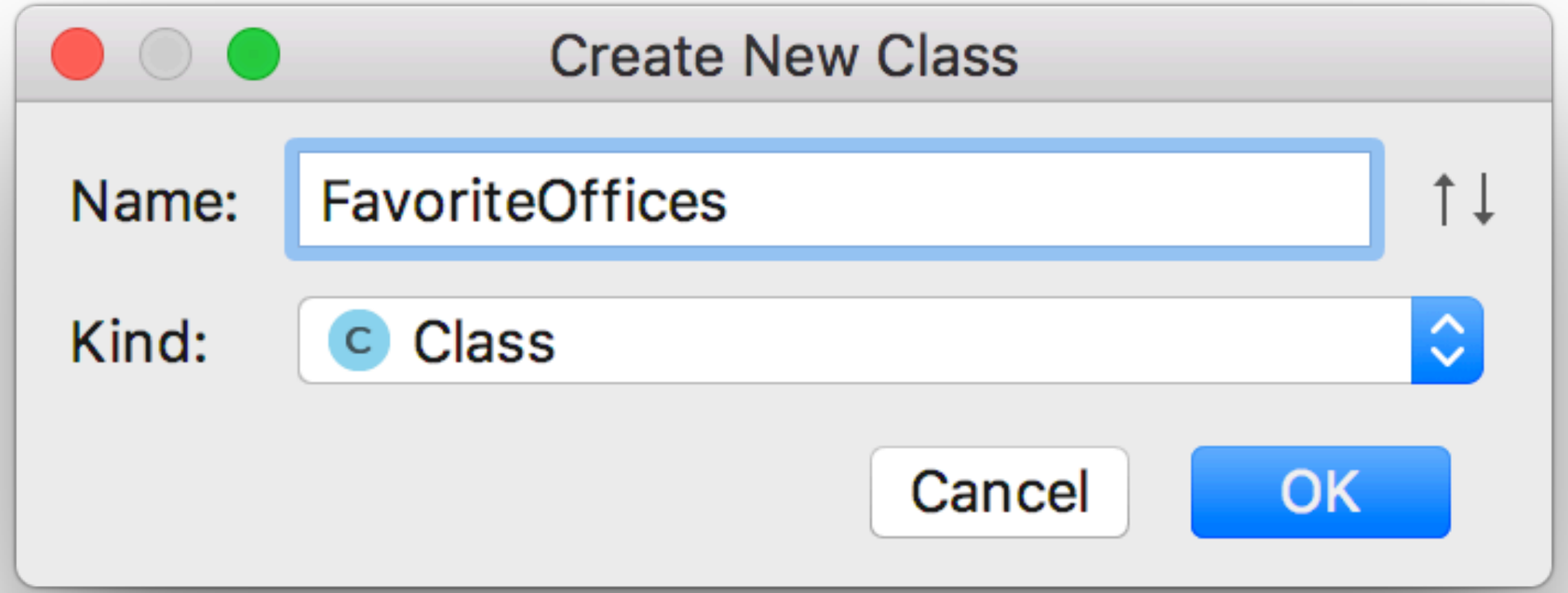
Dekompozycja

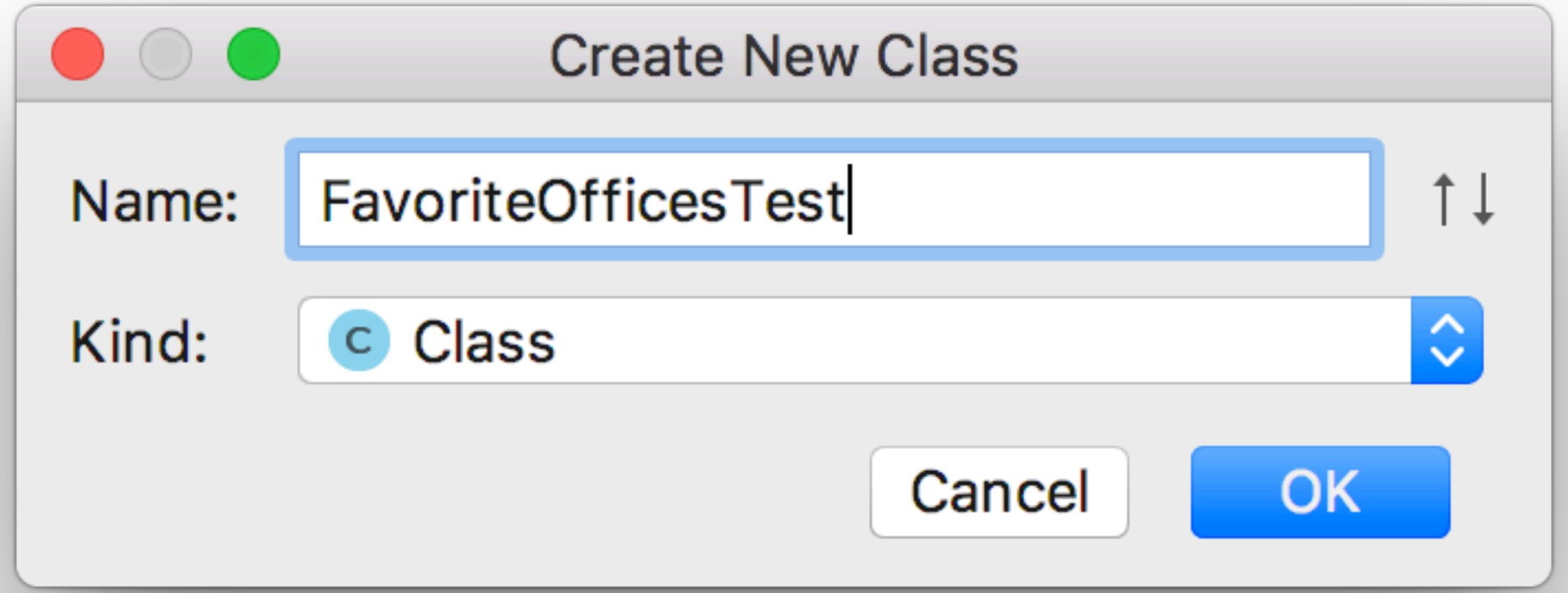
Niezależne fragmenty

Lesson learnt

#12 Nie zniechęcaj się trudno testowalnym kodem

Rola nawyków


Create New Class
Name:
FavoriteOffices
Kind:
Class
Cancel
OK


Create New Class
Name:
FavoriteOfficesTest
Kind:
Class
Cancel
OK

http://blog.gdinwiddie.com/2012/12/26/tdd-hat/

Lesson learnt

#13 Jeśli mam napisać kod, zaczynam od testu

Lesson learnt

#14 Merciless refactoring

Postawa

Technika

Droga

Postawa | Technika | Droga

#1 Nie trzeba dużo by zacząć

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		**#1 Nie trzeba dużo by zacząć**

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		**#1 Nie trzeba dużo by zacząć** **#3 Nie jest łatwo nauczyć TDD**

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		**#1 Nie trzeba dużo by zacząć** **#3 Nie jest łatwo nauczyć TDD** **#4 Nie narzucaj TDD**

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny		#3 Nie jest łatwo nauczyć TDD
		#4 Nie narzucaj TDD

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
		#4 Nie narzucaj TDD

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
		#4 Nie narzucaj TDD
		#7 Dobrze utrzymany zestaw testów daje moc

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera		#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
		#4 Nie narzucaj TDD
		#7 Dobrze utrzymany zestaw testów daje moc
		#8 Odkrywaj nowe rodzaje testów

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera	#9 Inwestuj w testowe DSLe	#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
		#4 Nie narzucaj TDD
		#7 Dobrze utrzymany zestaw testów daje moc
		#8 Odkrywaj nowe rodzaje testów

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera	#9 Inwestuj w testowe DSLe	#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
#10 TDD jest możliwe w każdej technologii		#4 Nie narzucaj TDD
		#7 Dobrze utrzymany zestaw testów daje moc
		#8 Odkrywaj nowe rodzaje testów

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera	#9 Inwestuj w testowe DSLe	#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
#10 TDD jest możliwe w każdej technologii	#11 Stosuj DDD oraz Ports & Adapters	#4 Nie narzucaj TDD
		#7 Dobrze utrzymany zestaw testów daje moc
		#8 Odkrywaj nowe rodzaje testów

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera	#9 Inwestuj w testowe DSLe	#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
#10 TDD jest możliwe w każdej technologii	#11 Stosuj DDD oraz Ports & Adapters	#4 Nie narzucaj TDD
#12 Nie zniechęcaj się trudno testowalnym kodem		#7 Dobrze utrzymany zestaw testów daje moc
		#8 Odkrywaj nowe rodzaje testów

Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera	#9 Inwestuj w testowe DSLe	#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
#10 TDD jest możliwe w każdej technologii	#11 Stosuj DDD oraz Ports & Adapters	#4 Nie narzucaj TDD
#12 Nie zniechęcaj się trudno testowalnym kodem		#7 Dobrze utrzymany zestaw testów daje moc
#13 Jeśli mam napisać kod, zaczynam od testu		#8 Odkrywaj nowe rodzaje testów

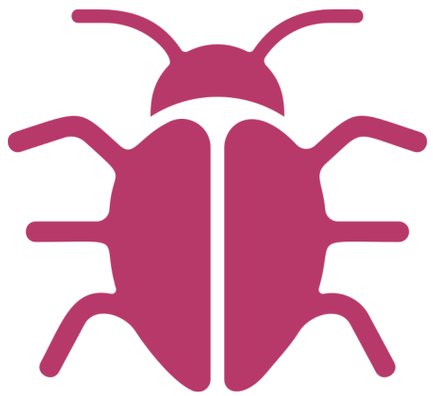
Postawa	Technika	Droga
#2 TDD to odpowiedzialność developera	#9 Inwestuj w testowe DSLe	#1 Nie trzeba dużo by zacząć
#5 Dbaj o kod testowy tak jak o produkcyjny	#6 Constructor Injection zwiększa testowalność	#3 Nie jest łatwo nauczyć TDD
#10 TDD jest możliwe w każdej technologii	#11 Stosuj DDD oraz Ports & Adapters	#4 Nie narzucaj TDD
#12 Nie zniechęcaj się trudno testowalnym kodem		#7 Dobrze utrzymany zestaw testów daje moc
#13 Jeśli mam napisać kod, zaczynam od testu		#8 Odkrywaj nowe rodzaje testów
#14 Merciless refactoring		

Korzyści z TDD

Bezpieczeństwo

Mniej defektów

Lepszy design



Mniej debugowania

Mniej „czy to zadziała?”

Dziękuję!

krzysztof.jelski@pragmatists.pl

@krzysztofjelski

PRAGMATISTS