Diebold Nixdorf

TestCon

TDD Demystified

Slavoj Písek
slavoj.pisek@dieboldnixdorf.com

15th October 2019

Agenda

10:00 – 10:30 Introduction

10:30 – 11:00 TDD quick intro

11:00 – 11:15 Break

11:15 – 12:45 Exercise

Agenda

12:45 – 13:45 Lunch

Agenda

13:45 – 14:15 Test first concept

14:15 – 15:15 TDD in real life

15:15 – 15:30 Break

15:30 – 16:30 TDD in real life continue

16:30 – 17:00 Refactoring

17:00 – 17:30 Conclusion

Agenda

- **Introduction**
- TDD quick intro
- Break
- Exercise

Slavoj Písek

- More than 15 years experience in SW development and testing
- Team Leader and Senior Developer at Diebold Nixdorf
- Author more than of two dozens of books on programming and IT
- Translator of some books about programming and photography for various publishing houses

You

TestCon

Please introduce yourself and try to answer the following questions:

- What is your name?
- Where are you from?
- How are you?
- What is your experience with TDD and ATDD?
- What is your expectation of this workshop?

Agenda

- Introduction
- **TDD quick intro**
- Break
- Exercise

TDD - Quick Introduction

What is TDD?

No DDT

No TBC

What is TDD?

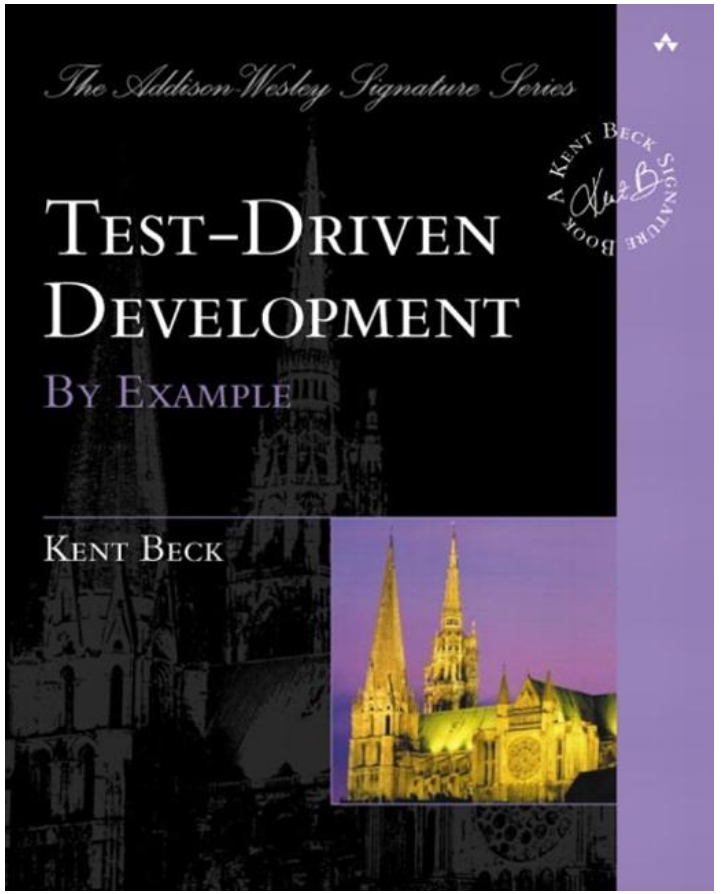
Test Driven Development

What is Test Driven Development?

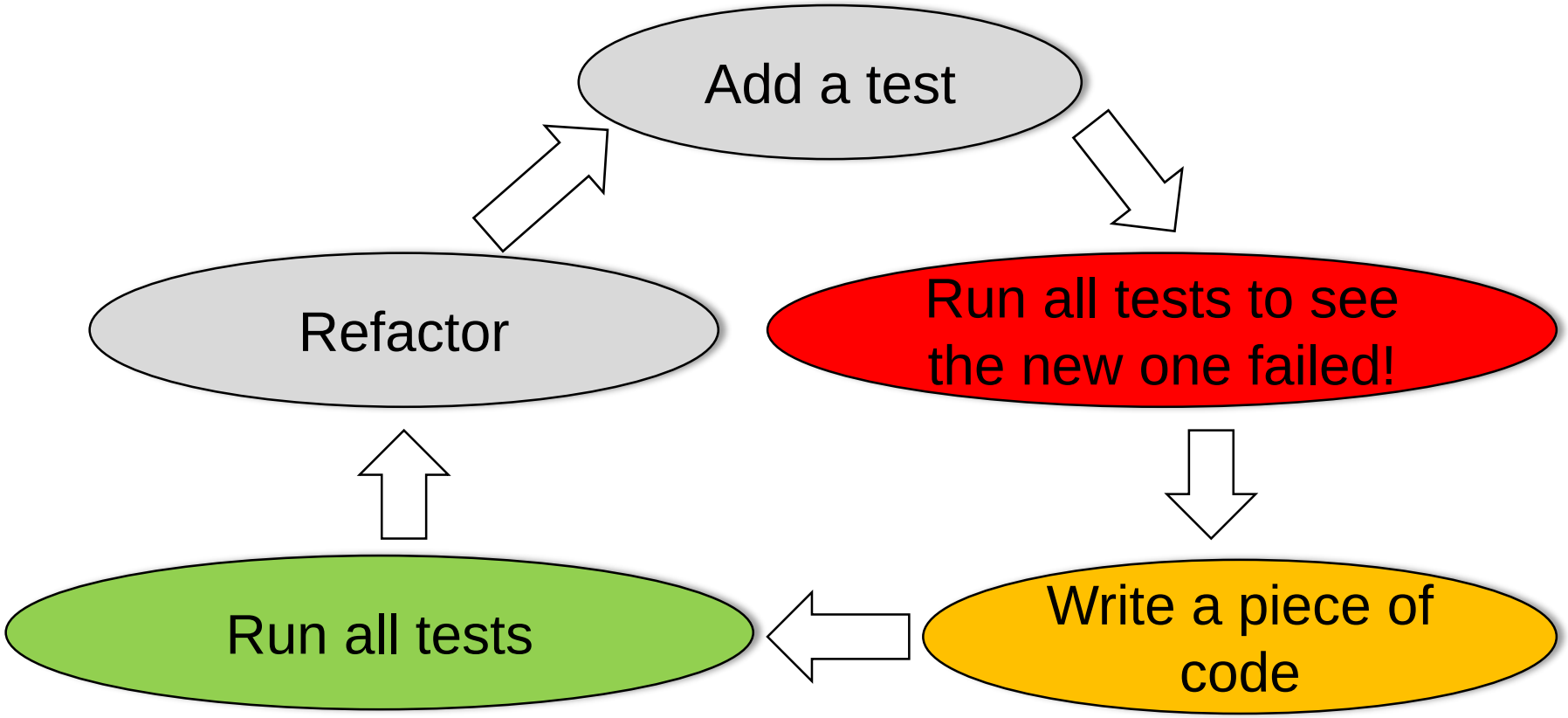
Wiki says…

Test-driven development (TDD) is a software development **process** that relies on the repetition of a very short development cycle…

Kent Beck's TDD by Example



Basic principle



What an Idea!

Advantages

TestCon

- Writing tests a developer thinks more about design.
- Less time spent with debugger.
- Tests serve as accurate, always up to date, low level documentation.
- Test-first concept forces coupling reduction in the code. Less coupling in the system the better.

Sounds good!

- Where to start?
- Theory is very simple
- Practice is not so obvious
- Let‘s start together with something simple…

Fibonacci sequence

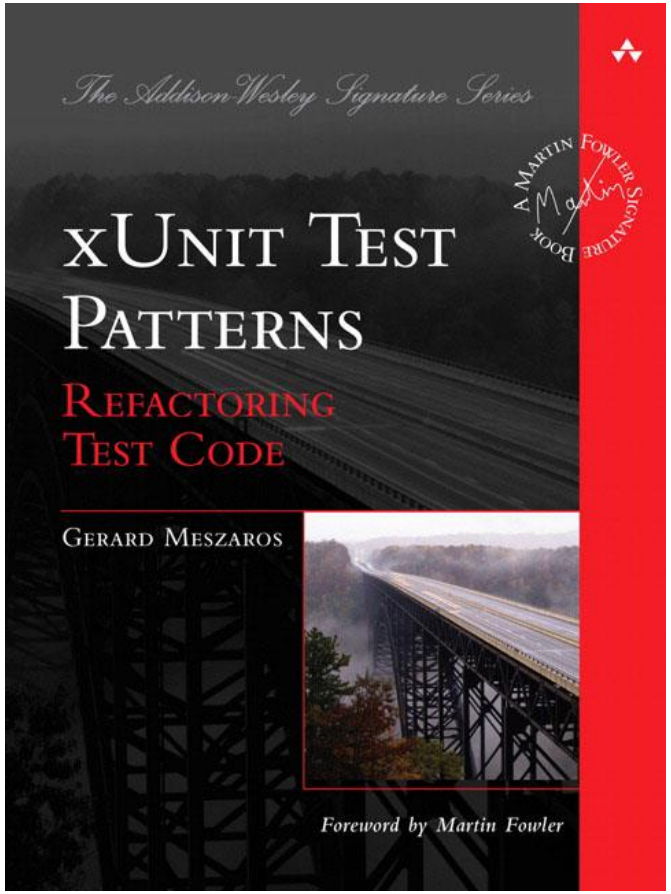
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …

JUnit

TestCon

- JUnit is a simple framework to write repeatable tests.
- It is an instance of the xUnit architecture for unit testing frameworks.

xUnit test patterns



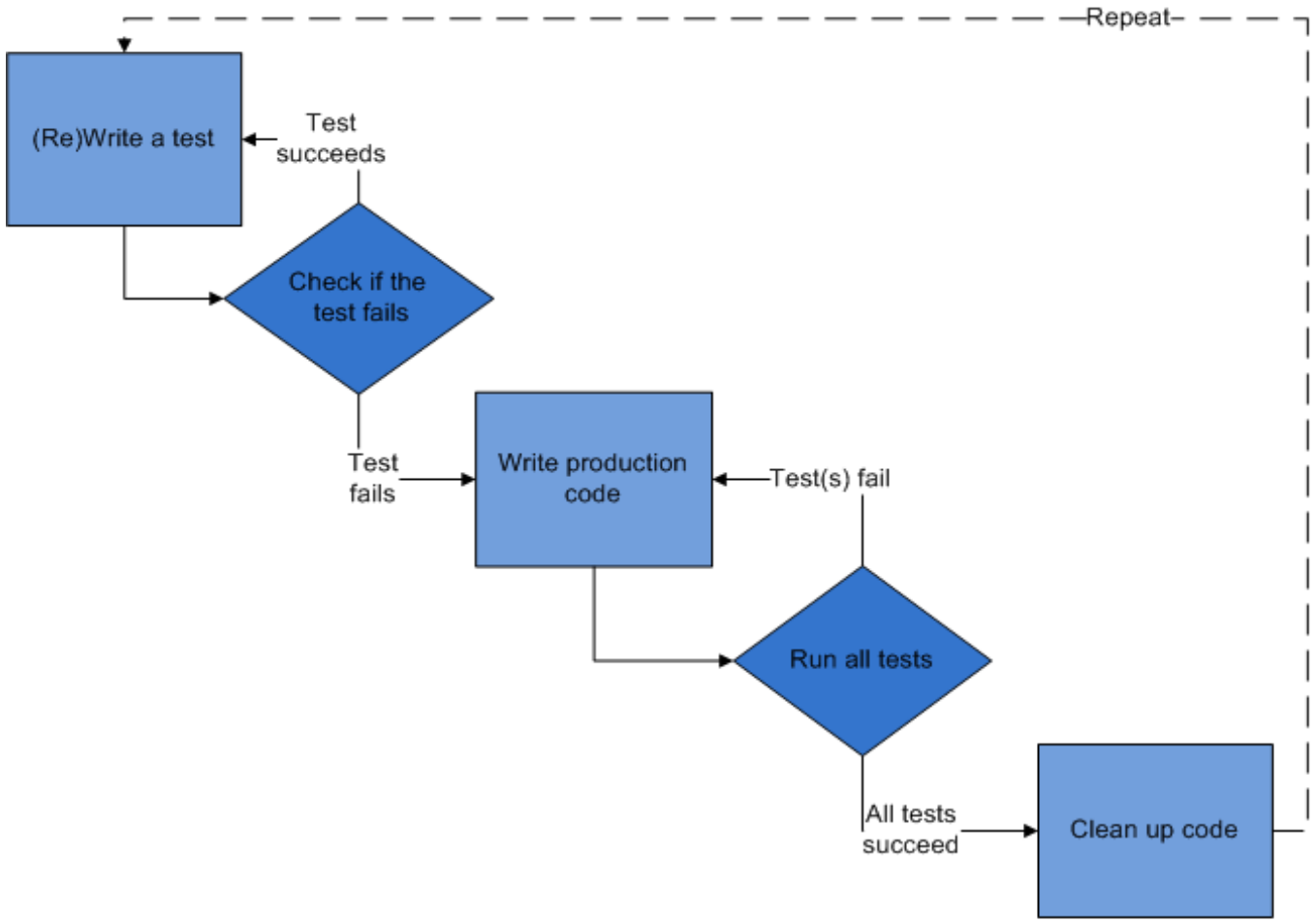
Practical Example

Retrospective

Reminder of TDD basic rules

- Add a test
- See it fail
- Add code to make all tests pass
- Refactor
- Do it again

Reminder of TDD basic rules



People often do this

- Think up design
- Write some tests that verify the design
- Write full implementation
- Test, debug, test, debug, test, debug, …
- Add TODO to refactor later

Exercise

Rules

TestCon

- Exactly one, simplest failing test
- Least code possible to pass failing test
- Add code only to test methods when duplication is spotted
- Extract non-test methods (extract method)
- New classes only as target for “move method”
- Refactor as required

Practice

Lunch

Part II

Agenda

- **Test first concept**
- TDD in real life
- Break
- TDD in real life continue
- Refactoring
- Conclusion

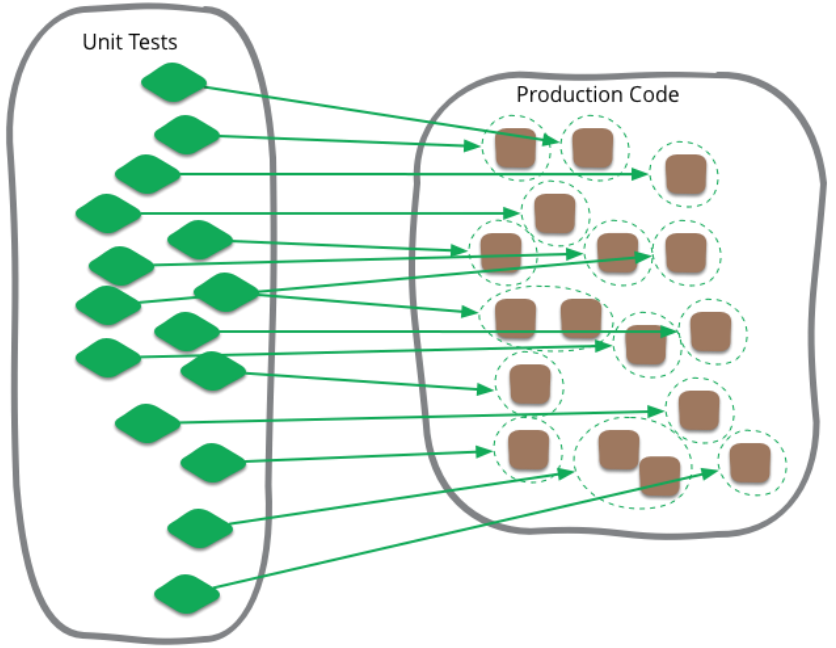
Practice

Agenda

- Test first concept
- **TDD in real life**
- Break
- TDD in real life continue
- Refactoring
- Conclusion

Unit testing

- Testing of system units
- Unit tests have to be independent
- Unit tests are automated
- There are frameworks for creating and maintaining Unit tests



What is an unit test, anyway

- Unit tests are supposed to be small.
- They have to be fast.
- They test a method or the interaction of a couple of methods.
- They are written by developers.

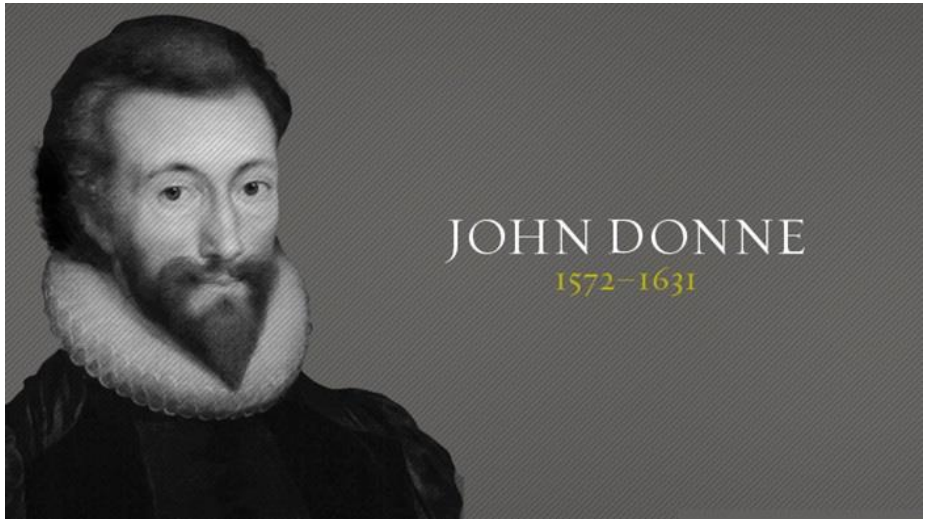
Test is not a unit test, when

- it talks to a database,
- it sends data over the network,
- it touches a file system,
- cannot be run concurrently with other unit tests,
- it needs some special configuration or initial steps.

TDD in real life

No man is an island, entire of itself...any man's death diminishes me, because I am involved in mankind; and therefore never send to know for whom the bell tolls; it tolls for thee.

-- John Donne



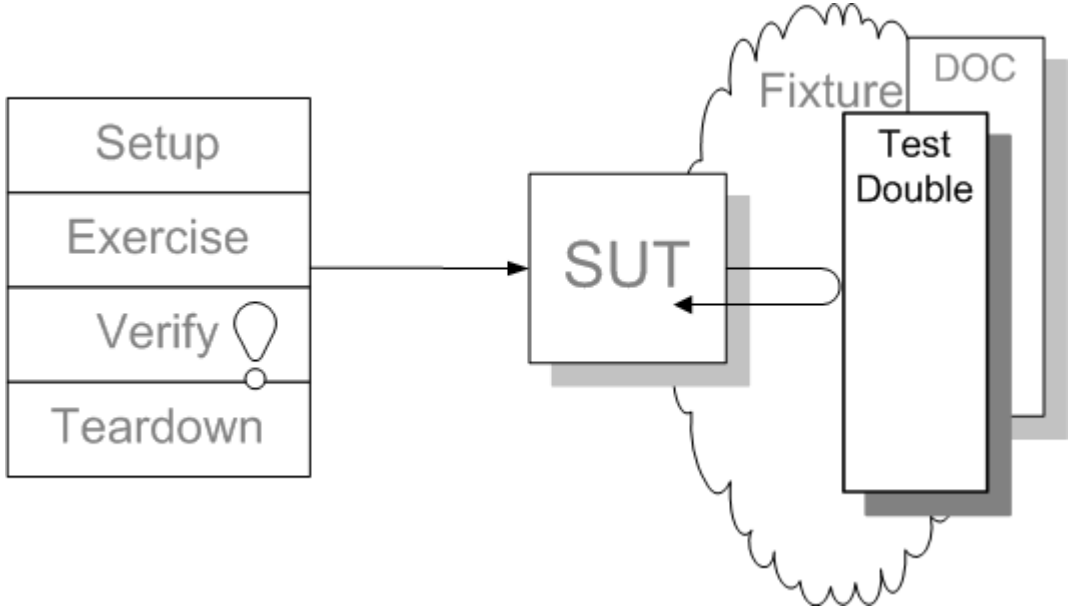
File Lottery Example

- Write an application that iterates through content of given folder and returns child folders in RANDOM order.
- When the application returns all available subfolders, it shall start over.
- If the argument is not a folder, but a file, its name shall be returned.

File Lottery test

In next 10 minutes write as much tests as possible

Objects double



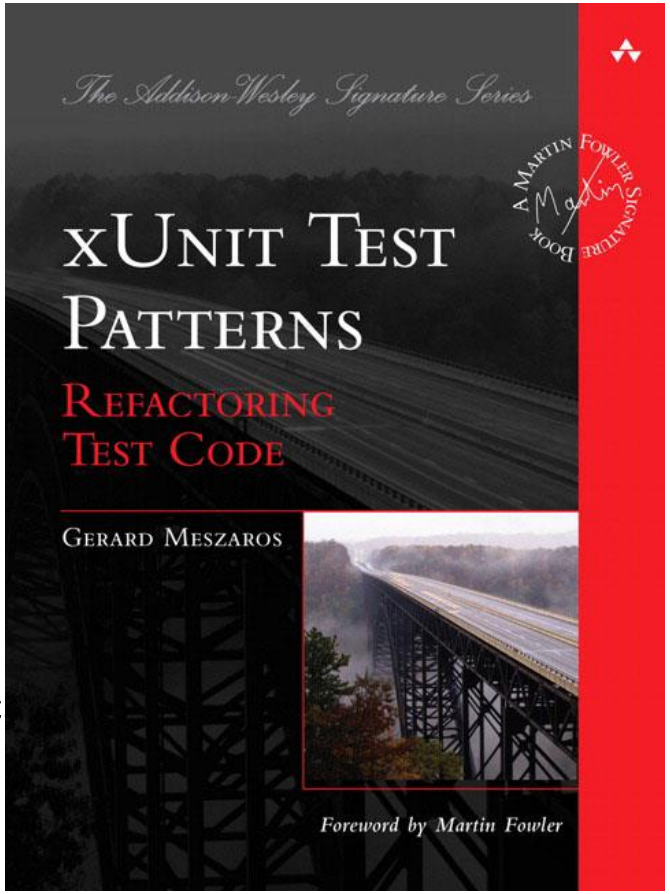
Dependency injection

- Control Inversion

- Instead of configuring dependency, class uses configured instance of dependency of outside.

Meszaros taxanomy

- **Dummy** objects are passed around but never actually used.
- **Fake** objects have working implementations.
- **Stubs** provide canned responses'.
- **Mocks** are pre-programmed with expectations which form a specific expected to receive.



Solving File Lottery tests

TestCon

- Create fake objects,
- inject them to the FileLottery class,
- configure fakes,
- perform tests.

Credits

TestCon

- Previous example is based on programming kata by Zsolt Fabók

- http://zsoltfabok.com/blog/2010/07/file-lottery-kata/

Agenda

- Test first concept
- TDD in real life
- **Break**
- TDD in real life continue
- Refactoring
- Conclusion

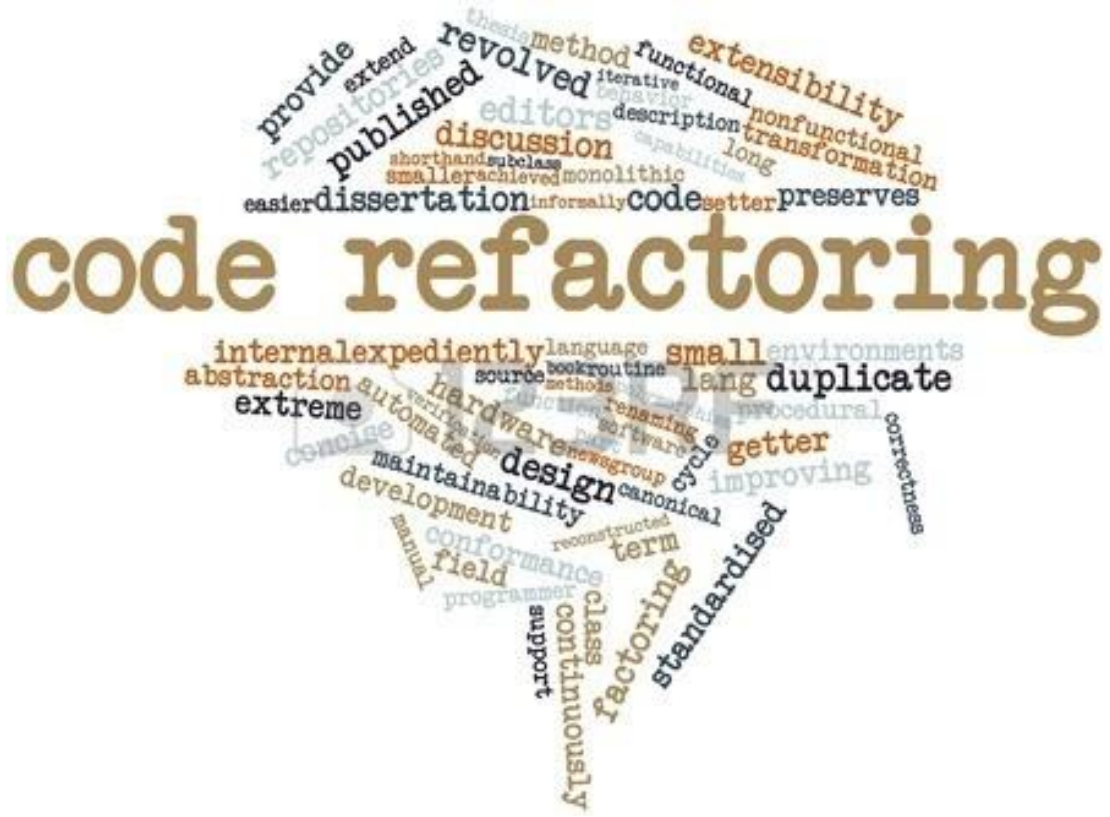
Agenda

- Test first concept
- TDD in real life
- Break
- **TDD in real life continue**
- Refactoring
- Conclusion

Agenda

- Test first concept
- TDD in real life
- Break
- TDD in real life continue
- **Refactoring**
- Conclusion

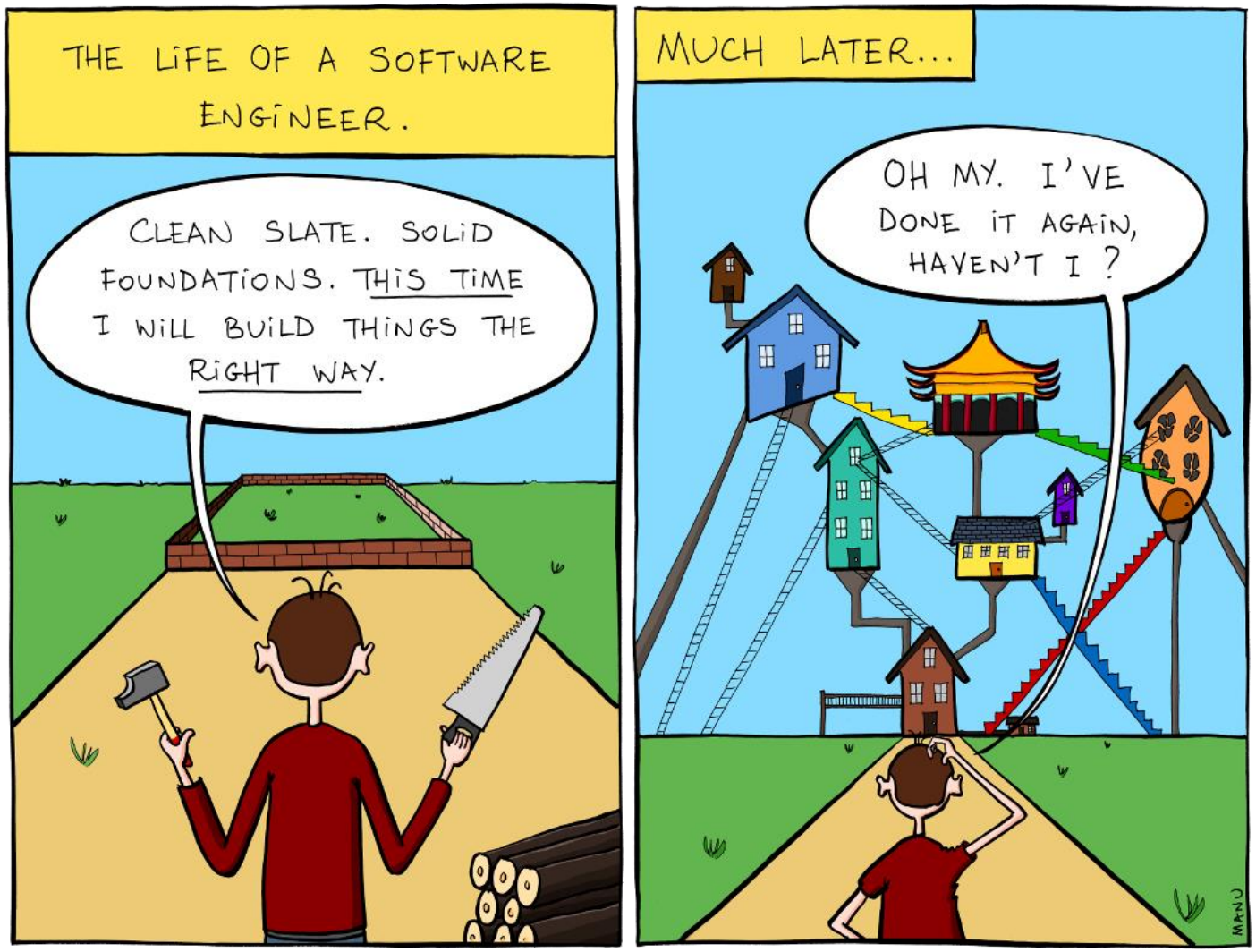
Code Refactoring



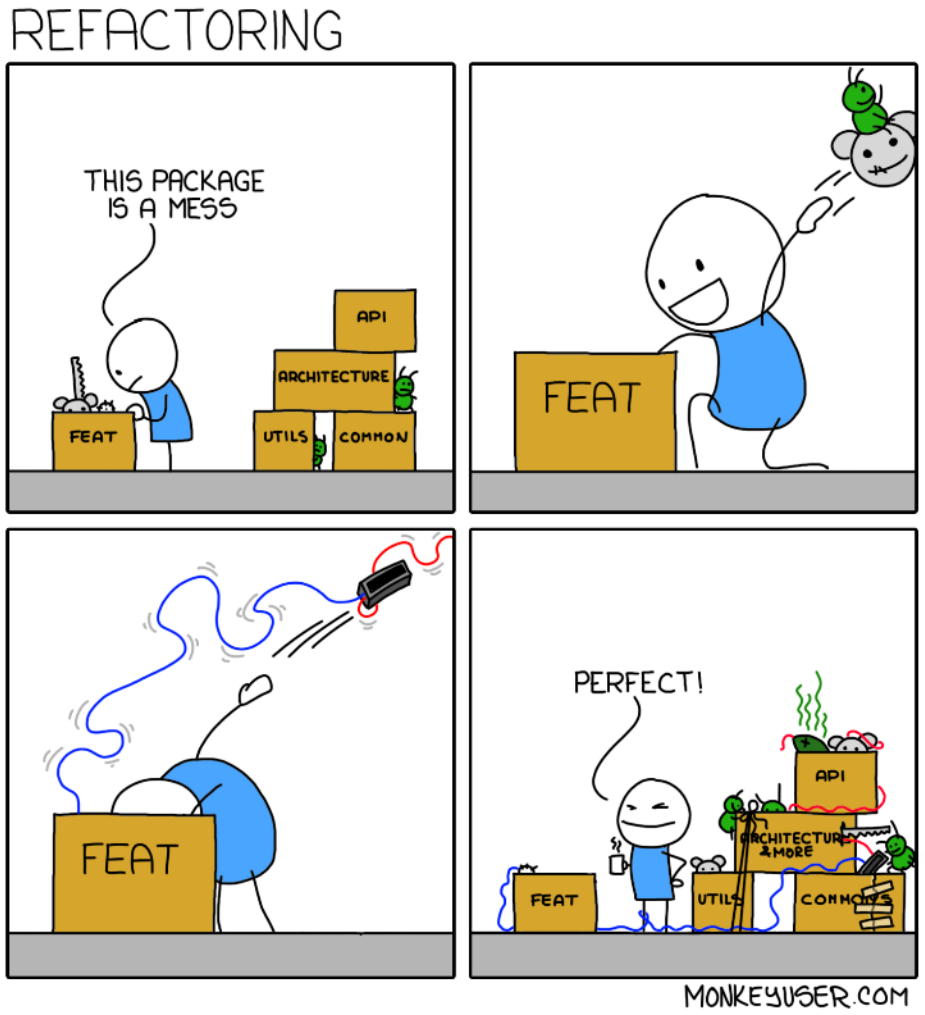
Refactoring

Code refactoring is the process of restructuring existing computer code **without changing its external behavior**.

Why do you need refactoring



Refactoring is a good servant…



Refactor techniques

TestCon

Techniques to add more abstraction:

- Encapsulate Field
- Generalize Type
- Replace conditional with polymorphism

Refactor techniques

TestCon

Techniques for breaking code apart into more logical pieces

- Componentization
- Extract Class
- Extract Method

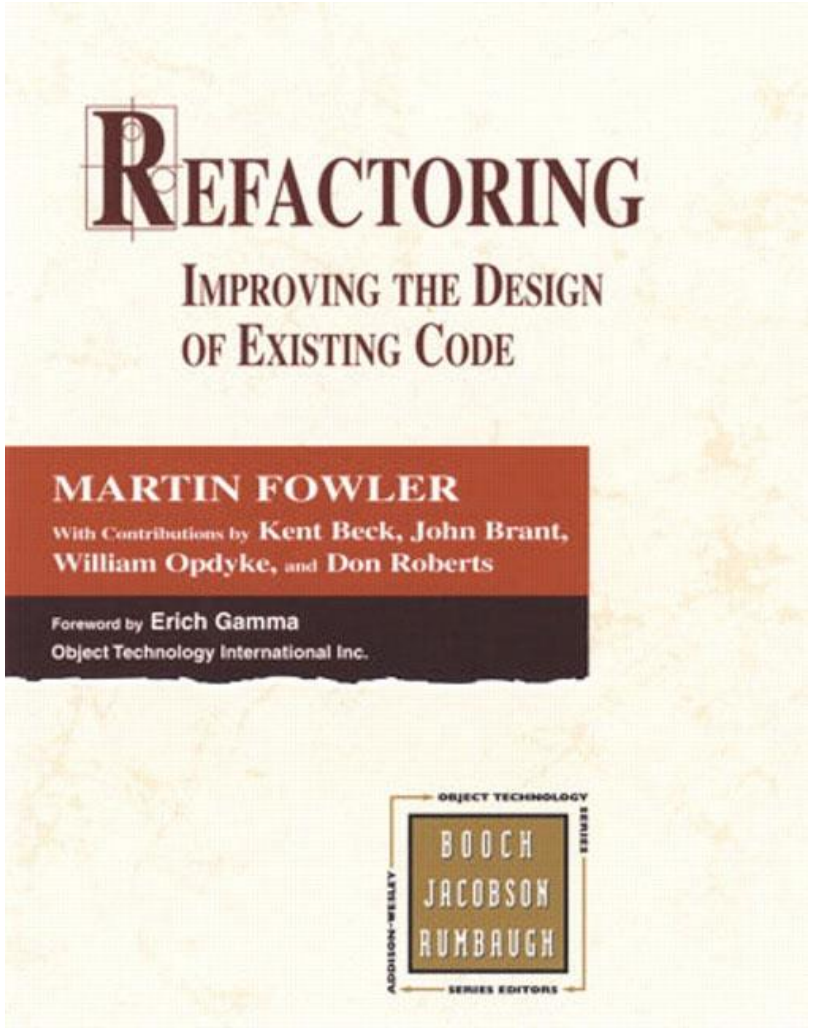
Refactor techniques

TestCon

Techniques for improving names and location of code

- Move Method or Move Field
- Rename Method or Rename Field
- Pull Up
- Pull Down

Refactoring by Martin Fowler



Signs of smell in the code

TestCon

- Lack of tests
- Name not expressing intent and not from domain
- Unnecessary if and else
- Duplication of constant
- A Method does more than one thing
- Primitive obsession
- Too long methods (> 6 lines)
- Too many parameters (> 3)

Signs of smell in the test

TestCon

- Not unitary
- Setup too complex
- Unclear Act
- More than one assert
- No assert
- Too many paths

Practice

Agenda

- Test first concept
- TDD in real life
- Break
- TDD in real life continue
- Refactoring
- **Conclusion**

Retrospective

- What did you learn today?
- What surprised you today?
- What will you do differently in the future?

Feedback


TestCon
WORKSHOP FEEDBACK
Take 5 minutes to help us
make our next event better.
http://bit.ly/testconworkshop

Questions