Code Quality Assurance
FH Technikum Wien, April 2013

Peter Kofler, ‘Code Cop’
@codecopkofler
www.code-cop.org

Copyright Peter Kofler, licensed under CC-BY.
Peter Kofler

- Ph.D. (Appl. Math.)
- Professional Software Developer for 14 years
- Lead Developer at IBM
- “fanatic about code quality”
The opinions expressed here are my own and do not necessarily represent those of current or past employers.
Tools of the Trade

- Mindset/Books
- Code Kata
- TDD/BDD
- Code Coverage
- CI
- Static Analysis
- Code Review
Zero-Defect Mindset (MSF)
Boy Scout Rule
Der Pragmatische Programmierer

Andrew Hunt
David Thomas

Übersetzt von Andreas Braig und Steffen Gemkow

Hансер
Craftsmanship

Engineering
“Craftsmanship Over Crap“
(Uncle Bob)
suddenly
"passion"
became
a buzzword!
Mindset and Passion
The Prime Factors Kata
What Exactly Will We Do?

- write code together
- using TDD
- see techniques and patterns
- discuss while doing
The Requirements.

• Write a class named “PrimeFactors” that has one static method: generate.
  • The generate method takes an integer argument and returns a List<Integer>.
  • That list contains the prime factors in numerical sequence.

http://butunclebob.com/ArticleS.UncleBob.ThePrimeFactorsKata
First Some Math.

- **Prime Number**: number > 1 that has no divisors other than 1 and itself.
  - e.g. 2, 3, 5, 61, 67, ..., 997, ..., $2^{43112609} - 1$

- **Prime Factors**: prime numbers that divide an integer without remainder.
  - e.g. $2 = 2$,
    - $4 = 2 \times 2$,
    - $24 = 2 \times 2 \times 2 \times 3$
    - $288 = 2^5 \times 3^2$
Requirements (for Mortals).

• code a file/module/class “PrimeFactors”
• function/method/routine “generate“
• accept an integer number as parameter
• return the prime factors of that number
Demo
(Code Kata)
Keep the bar green to keep the code clean
Unit Testing

- test individual units
- isolate each part
- show that the individual parts are correct
- regression testing
- sort of living documentation
- executed within a framework
Pragmatic Unit Testing

In Java with JUnit

The Pragmatic Starter Kit – Volume II

Andrew Hunt  David Thomas
Some xUnit Frameworks

- Smalltalk – SUnit
- Java - JUnit, TestNG
- .NET - NUnit, Typemock
- C++ - CppUTest
- Groovy, PHP, Python, Ruby, Scala, ...
“I write unit tests for one reason: so my coworkers don't f*** up my code.”
(David Angry)
Test-Driven Development

• add a test
• run all tests and see if the new one fails
• write some code
• run all tests and see them succeed
• refactor code **mercilessly**
• „Red Green Refactor“
A minute ago all their code worked
BDD is TDD “Done Right“
Scenarios and Stories

• focus on behaviour
(e.g. requirements are behaviour)
• full sentences
• expected behaviour should
• Use Arrange-Act-Assert in form of Given-When-Then
Testing is a mindset
You have to want it
TESTING
I FIND YOUR LACK OF TESTS DISTURBING.
Code Coverage

comprehensiveness of tests
Beware!
comprehensiveness ≠ quality!
100%
(What else?)
Some Code Coverage Tools

- Java - EMMA, Cobertura, Clover
- .NET - NCover, PartCover
- C++ - Covtool, gcov
- Ruby – rcov
- commercial products
Never measure developers by Code Coverage!
Demo
(Coverage)
Coverage helps.
Use it!
Really, Use it!
Demo
(Jenkins)
works on my machine
Daily Builds
Nightly Builds
Continuous Integration
Continuous Integration

• Maintain a code repository
• Automate the build
• Make the build self-testing
• Everyone commits every day
• Every commit should be built
• Keep the build fast
• Everyone can see the results of the build
Immediate Feedback and Early Warning
Some CI Servers

- Java - Jenkins (Hudson), CruiseControl
- .NET - TFS, Jenkins, CruiseControl.NET
- C++ - Jenkins (Shell/Make)
- Ruby – CruiseControl.rb
- commercial products - TeamCity, ALA
Continuous Integration: Deli Delivery, Improving Software Quality and Reducing Risk

Jez Humble
Paul M. Duvall
Steve Matyas
Andrew Glover

Forewords by Martin Fowler and Paul Julius

The Addison-Wesley Signature Book A Martin Fowler Signature Book

The Addison-Wesley Signature Book A Martin Fowler Signature Book
YOU BROKE THE BUILD!

This is Agnes and she's none too pleased with you, Bub. You broke the build. You should have known that you were making breaking changes, but you checked them in anyway.

If you start using continuous integration, Agnes won't have to come back.

www.YouBrokeTheBuild.com
Good Enough?
Demo

(Warnings)
Check What?

• **Lexical analysis**
  – naming, coding conventions, design idioms

• **Flow/path analysis**
  – null-pointer, dead code (conditional)

• **Dependency analysis**
  – architectural/design flaws

• **Verification**
  – mathematical proof of correctness
Some Static Analysis Tools

- Java - Checkstyle, Findbugs, PMD
- .NET - FxCop, (ReSharper, NDepend)
- C++ - (Sp)Lint
- Ruby - Roodi, Dust, Flog, Saikuro
- commercial products
Many, Many Tools
Good Enough?
Demo

(Review)
Some Code Review Tools

- Your peer (Pair Programming)
- Your version control system
- Your IDE (e.g. ReviewClipse)
- Commercial products
The only valid measurement of code quality: WTFs/minute

Good code.

Bad code.
Lessons Learned

(!) Boy Scout Rule

(!) Do TDD/BDD

(!) Use the Tools
  (Coverage, CI, Static Analysis)
Thank You
Peter Kofler
@codecopkofler
www.code-cop.org
Links #1

Mindset

- http://pragprog.com/the-pragmatic-programmer

Software Craftsmanship

- http://clean-code-developer.de/

Kata

- https://bitbucket.org/pkofler/primefactors.java/ (Kata Source Code)
Links #2

Unit Test

TDD

BDD
- [http://dannorth.net/introducing-bdd/](http://dannorth.net/introducing-bdd/)

Code Coverage
Links #3

Continuous Integration
• http://en.wikipedia.org/wiki/Continuous_integration
• http://www.codinghorror.com/blog/archives/000818.html
• http://www.stevemcconnell.com/ieeesoftware/bp04.htm
• http://www.joelonsoftware.com/articles/fog0000000023.html
• http://jenkins-ci.org/

Static Code Analysis
• http://en.wikipedia.org/wiki/Static_code_analysis
• http://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis

Code Review
• http://en.wikipedia.org/wiki/Code_review
• http://en.wikipedia.org/wiki/Pair_programming
CC Images

- spray face: http://www.flickr.com/photos/iangallagher/4115047191/
- Judge Dredd: http://www.flickr.com/photos/eldave/6169431454/
- list: http://www.flickr.com/photos/kylesteeddesign/3724074594/
- eggs: http://www.flickr.com/photos/nickwheeleroz/2475011402/
- windows: http://www.flickr.com/photos/sketchglass/4281424410/
- shoe maker: http://www.flickr.com/photos/tbatty/1450209613/ (by 2010, now dead link)
- factory: http://www.flickr.com/photos/94693506@N00/4643248587/
- questions: http://www.flickr.com/photos/oberazzi/318947873/
CC Images

- fence: http://www.flickr.com/photos/30830597@N08/3630649274/
- Darth Vader: http://rubystammtisch.at/
- covered car: http://www.flickr.com/photos/paulk/3166328163/
- works: http://www.codinghorror.com/blog/archives/000818.html
- Agnes: http://www.youbrokethebuild.com/
- wtf: http://www.flickr.com/photos/smitty/2245445147/
- questions: http://www.flickr.com/photos/seandreilinger/2326448445/