Code Quality Assurance

Peter Kofler, ‘Code Cop’
FH Technikum Wien, February 2010
Peter Kofler

- Ph.D. (Appl. Math.)
- Professional Software Developer for 11 years
- Lead Developer at System One
- fanatic about code quality
Zero-Defect Mindset (MSF)
Der Pragmatische Programmierer

Andrew Hunt
David Thomas

Übersetzt von Andreas Braig und Steffen Genskow

HANSER
Robert C. Martin
("Uncle Bob")
Software
Craftsmanship
Craftsmanship
Over Crap
The Prime Factors Kata
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
Demo
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

http://en.wikipedia.org/wiki/Unit_testing
xUnit Frameworks

- e.g.
- Smalltalk - SUnit
- Java - JUnit, TestNG
- .NET - NUnit, Typemock
- C++ - CppUTest
- Groovy, PHP, Python, Ruby, Scala, etc.

http://en.wikipedia.org/wiki/List_of_unit_testing_frameworks
TESTING
I FIND YOUR LACK OF TESTS DISTURBING.
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
• „Red Green Refactor“

http://en.wikipedia.org/wiki/Test_Driven_Development
The Three Laws of TDD

• You are not allowed to write any production code unless to make a failing unit test pass.
• You are not allowed to write any more of a unit test than is sufficient to fail; and compilation failures are failures.
• You are not allowed to write any more production code than is sufficient to pass the one failing unit test.

http://butunclebob.com/ArticleS.UncleBob.TheThreeRulesOfTdd
“A minute ago all their code worked!”
Test-Driven Development
By Example
Kent Beck
Testing is a mindset.
You have to want it.
Code Coverage

comprehensiveness of tests
Beware!

comprehensiveness ≠ quality!
Some Code Coverage Tools

• Java - EMMA, Cobertura, Clover
• .NET - NCover, PartCover
• C++ - Covtool, gcov
• Ruby - rcov
• commercial products
Demo
Demo
Works on my machine
Daily/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

http://en.wikipedia.org/wiki/Continuous_integration
Immediate Feedback & Early Warning
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
Some CI Servers

- Java - CruiseControl, Hudson
- .NET - CruiseControl.NET, Hudson (NAnt)
- C++ - Hudson (Shell/Make)
- Ruby - CruiseControl.rb
- commercial products - TeamCity, ALA
Book
“Continuous Integration” (Paul Duvall)
Good Enough?
Demo
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
Good Enough?
Demo
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.
Thank You
Links #1

Mindset

• http://codebetter.com/blogs/darrell.norton/archive/2003/12/03/4222.aspx (Zero-Defect Mindset)
• http://en.wikipedia.org/wiki/Fixing_Broken_Windows
• http://pragprog.com/the-pragmatic-programmer
• http://en.wikipedia.org/wiki/Robert_Cecil_Martin
• http://www.clean-code-developer.de/
• http://en.wikipedia.org/wiki/Software_craftsmanship

Test

• http://butunclebob.com/ArticleS.UncleBob.ThePrimeFactorsKata
• http://en.wikipedia.org/wiki/Kata_(programming)
Links #2

Test
• http://en.wikipedia.org/wiki/Unit_testing
• http://en.wikipedia.org/wiki/XUnit
• http://en.wikipedia.org/wiki/List_of_unit_testing_frameworks
• http://en.wikipedia.org/wiki/Test_Driven_Development
• http://butunclebob.com/Articles/UncleBob.TheThreeRulesOfTdd

Code Coverage
• http://en.wikipedia.org/wiki/Code_coverage
• http://www.ibm.com/developerworks/java/library/j-cq01316/

Continuous Integration
• http://hudson-ci.org/
• http://www.codinghorror.com/blog/archives/000818.html
Links #3

Continuous Integration
• http://www.stevemcconnell.com/ieeesoftware/bp04.htm
• http://www.joelonsoftware.com/articles/fog000000023.html
• http://en.wikipedia.org/wiki/Continuous_integration

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
Images

- spray face: http://www.flickr.com/photos/iangallagher/4115047191/
- broken egg: http://www.flickr.com/photos/nickwheeleroz/2475011402/
- windows: http://www.flickr.com/photos/sketchglass/4281424410/
- red-green: http://www.flickr.com/photos/30830597@N08/3630649274/
- Darth Vader: http://rubystammtisch.at/
- Model http://www.flickr.com/photos/cabfhablab/3855825945/
- covered car: http://www.flickr.com/photos/paulk/3166328163/
- works: http://www.codinghorror.com/blog/archives/000818.html
- Agnes: http://www.youbrokethebuild.com/
- code review: http://www.flickr.com/photos/seandreilinger/2326448445/
- questions: http://www.flickr.com/photos/seandreilinger/2326448445/