Java code defect analysis

Jiří Kiml,
11/11/2010, ZCU

Agenda

- Motivation
- Eclipse compiler
- Checkstyle
- FindBugs
- Best Practices
- General problems
- Summary

Motivation

Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live.
~Martin Golding

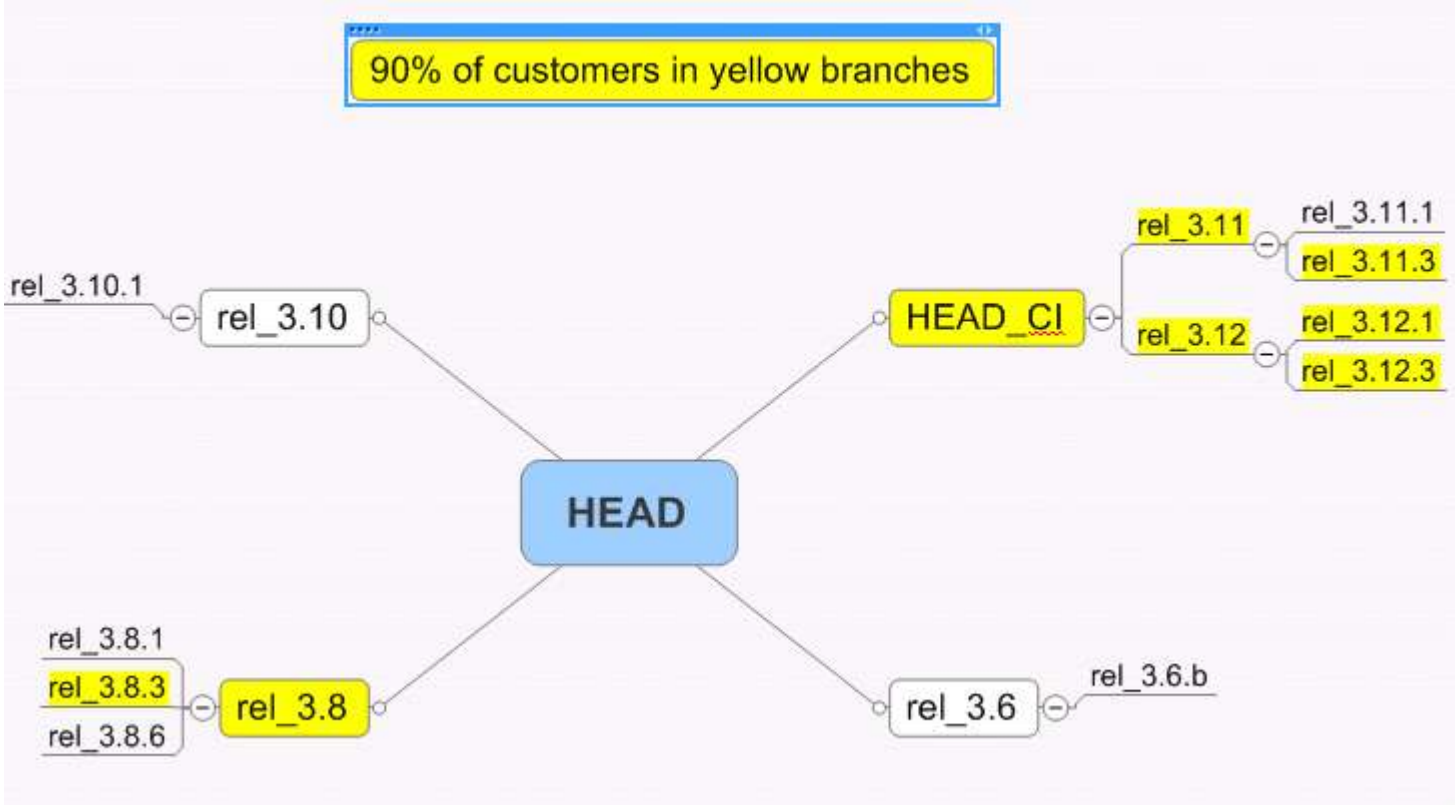
- **What is the cost of code maintenance?**
- **Can we prevent bugs?**

MPA – medical process assistant

- 16 active branches
- 34.000 classes
- 250 eclipse projects
- About 50 programmers
- 12 years old (jdk 1.1)
- More than 100 hospitals
- Austria/Germany/Czech Republic/Serbia
- http://www.systema.info/en/solutions/mpa-medical-process-assistant/

Programming is like sex. One mistake
and you have to support it for the rest of your life.
~Michael Sinz

MPA branches



Any fool can write code that a computer can understand. Good programmers write code that humans can understand.
~Martin Fowler

ThreadSleepExample

Eclipse compiler

- **Preferences|Java|Compiler|Error/Warnings**
  - Code style (Non-static access to a static member, ...)
  - Potential programming problems
    (class overrides equals() but NOT hashCode(), ...)
  - Name shadowing and conflicts
  - Deprecated and restricted API
  - Unnecessary code (Parameter is never read, ...)
  - Generic types (Usage of a raw type, ...)
  - Annotations (Missing '@Override' annotation, ...)
- **Preferenes|Java|Compiler|Javadoc**

Good code is its own best documentation.
As you're about to add a comment, ask yourself,
"How can I improve the code so that this comment isn't needed?"
~Steve McConnell

Eclipse compiler

+ IDE integration (save actions and cleanup)
+ Speed
+ Quick fix

- Small rule set
- @SuppressWarnings limitations
- no custom detectors

Checkstyle

- a development tool to help programmers write Java code that adheres to a coding standard.
- http://checkstyle.sourceforge.net/

Metrics example (MainFrame, RepoEmbedableCustomizer)

Checkstyle (2)

- **Javadoc Comments**
- **Naming Conventions** (Constant names, parameter names, ...)
- **Imports** (Illegal imports, redundant imports, ...)
- **Size Violations** (method length, file length, number of params, ...)
- **Whitespace** (tabs, whitespace before/after, ...)
- **Modifiers** (order, redundant, ...)
- **Blocks** (empty block, curly brace, ...)

Checkstyle (3)

- **Codding Problems** (return count, modified control variable, ...)
- **Class Design** (throws count, fields visiblity, ...)
- **Duplicates**
- **Metrics** (Expression complexity, ...)
- **Miscellaneous** (final parameters, Upper Ell, ...)
- **J2EE** (final static, ...)

Checkstyle

+ Speed
+ Size violations/Metrics
+ Excludes from checking

- Quick fix
- No eclipse perspective

It's not a bug - it's an undocumented feature.
~Author Unknown

MyClassExample

FindBugs

- a Java static analysis tool that scans compiled java code for potential defects and bad programming practices.
- http://findbugs.sourceforge.net/
- 369 bug patterns

We should forget about small efficiencies,
say about 97% of the time:
premature optimization is the root of all evil.
~C.A.R. Hoare, quoted by Donald Knuth

GetSetSynchronizationExample

FindBugs (2)

- **Malicious code** (FI_PUBLIC_SHOULD_BE_PROTECTED, ...)
- **Style/Dodgy** (DLS_DEAD_LOCAL_STORE, REC_CATCH_EXCEPTION, ...)
- **Bad practice** (ES_COMPARING_PARAMETER_STRING_WITH_EQ, ...)
- **Correctness** (DMI_BAD_MONTH, RE_POSSIBLE_UNINTENDED_PATTERN, ...)
- **Internationalization**
- **Performance** (DM_GC, SBSC_USE_STRINGBUFFER_CONCATENATION)
- **Security** (sqli, xss, ...)
- **Multithreaded correctness** (SC_START_IN_CTOR)

When debugging, novices insert corrective code; experts remove defective code.
~Richard Pattis

NNExample

JSR 305: Annotations for Software Defect Detection

- http://jcp.org/en/jsr/detail?id=305
- @CheckForNull, @CheckReturnValue
- @NonNull, @OverrideMustInvoke
- @GuardedBy, @Immutable
- @NotThreadSafe, @ThreadSafe
- .....

FindBugs

+++ Rule set
+ JSR 305 annotations
+

- Quick fix
- false warnings
--- **SPEED**

You cannot teach beginners top-down programming, because they don't know which end is up.
~C.A.R. Hoare

SubClassWithProtected

PMD and others

- http://pmd.sourceforge.net/
- Copy/Paste Detector

- DoctorJ, AppPerfect
- ESC/Java, Lint4J
- Hammurapi, Jamit
- many others ...

Best Practices

There are two ways to write error-free programs; only the third one works.
~Alan J. Perlis

Best Practices

- Choose the rules that are right for you
  - from zero to hero
- Rules are not set in stone
  - justification
- Use IDE integration (CI is too late)
- Never commit with warnings
- Use the jsr 305 Annotations

General problems

Should array indices start at 0 or 1?
My compromise of 0.5 was rejected without,
I thought, proper consideration.
~Stan Kelly-Bootle

Calendar.getInstance().set(2000, **1**, 1) → 1. **2**. 2000
Calendar.getInstance().set(**2000**, **12**, 31) → 31. **1**. **2001**

General problems

- **Custom rule sets** (different opinions, changes)
- **Performance and memory requirements**
- **Version control systems** (code cleanup in separate changelists, different branches)
- **Long projects**
- **Third party / generated code**
- **How to convince developers to use it?**

Summary

- **What is the cost of code maintenance?**
- **Can we prevent bugs?**

The only way for errors to occur in a program is by being put there by the author. No other mechanisms are known. Programs can't acquire bugs by sitting around with other buggy programs.
~Harlan Mills

Links

- http://checkstyle.sourceforge.net/
- http://findbugs.sourceforge.net/
- http://jcp.org/en/jsr/detail?id=305
- http://pmd.sourceforge.net/
- Coding Without Comments - http://www.codinghorror.com/blog/archives/001150.html
- Common Excuses Used To Comment Code - http://www.codeodor.com/index.cfm/2008/6/18/Common-Excuses-Used-To-Comment-Code-and-What-To-Do-About-Them/2293

Questions ...

… and maybe answers

Thank you

Jiří Kiml,
11/11/2010, ZCU