Code Quality Assurance with PMD

Peter Kofler, ‘Code Cop’
Herold Business Data, 2004
Status Daily Build
Static Code Analysis
Rule Categorisation
Rules to fix/obey ;-)

PETER KOFLER, CODE-COP.ORG
FANATIC ABOUT CODE QUALITY
Daily Build

- Since three months
- Trunk is compiling (at least more often)
  - missing files ✓
  - shared subproject complexity ✓
- Build history
- JavaDoc
- XML validation
Next Steps

• Static code analysis
  – check the source

• JUnit integration
  – of few existing tests

• Initialise application
  – integration tests (that would be cool)
PMD Static Code Analyser

- http://pmd.sourceforge.net/
- Scans Java source
- JavaCC-generated parser
  - Abstract Syntax Tree (AST)
- Traverses AST
- Reports violations
Eclipse Plugin

• Install pmd-eclipse2-site-2.0RC3.zip

• Patch plugin with current PMD:
  – extract pmd.jar from pmd-bin-1.5.zip
  – rename to pmd-1.3.jar and replace in plugins\net.sourceforge.pmd.core_1.3.2\lib

• Customise rules:
  – delete all and import PMD20040427.xml
Rule

Categorisation
“Error”

- Really serious errors
  - e.g. bad practice
  - or correctness bugs
- Currently (only ;-) 125 in whole code base
“Warning”

- Other errors

- Currently 4023
“Info” (Information)

- Minor errors
  - e.g. performance problems

- Currently 4091
“Format”

• Formatting Problems
  – e.g. code style violations

• Currently 6834
“Fix” Process

• Activate not (yet) triggered rules
  – make sure there are no new violations
• Fix few “bad guys” first
• Activate their rules
• Choose next error from list, repeat
• In the end only low priority bugs left
Not Triggered Errors

- **BadComparison** if \( y == Double.NaN \)
- **EmptyFinalizer**
- **FinalizeOnlyCallsSuperFinalize**
- **FinalizeOverloaded**
- **ShortMethodName** \( (\leq 3) \)
- **SuspiciousHashCodeMethodName**

```java
public int hashCode() // <-> hashCode
```
• **EmptyWhileStmt**
  ```java
while (a == b) {
    // maybe here was some code before
}
```

• **NonStaticInitializer**
  ```java
// public void doSomething()
{
    // this block gets run before any
    // call to any constructor
    System.out.println("construct myself");
}
```
MethodWithSameName
AsEnclosingClass

public class MyClass
{
    // bad because it is a method
    public void MyClass() {}
    // OK because it is a constructor
    public MyClass() {}
}
• **JumbledIncrementer**
  ```java
  for (int i = 0; i < 10; i++) {
      for (int k = 0; k < 20; i++) {
          System.out.println("Hello");
      }
  }
  ```

• **NonCaseLabelInSwitchStatement**
  ```java
  switch (a) {
      case 1 : // do something
          break;
      mylabel : // legal but confusing
          break;
  }
  ```
Not Triggered Warnings

- DefaultLabelNotLastInSwitchStmt
- DontImportJavaLang
- EmptyStaticInitializer
- EmptySwitchStatements
- EmptySynchronizedBlock
- EmptyTryBlock & EmptyFinallyBlock
- ImportFromSamePackage
- ForLoopShouldBeWhileLoop
  ```java
  for (; true;) {
      // no init or update part
      // may as well be: while (true)
  }
  ```

- SuspiciousOctalEscape
  ```java
  public void foo() {
      System.out.println("suspicious: \128");
      // interpreted as octal 12,
      // followed by character '8'
  }
  ```
Epic Evil
10 Serious Errors (to fix)

- EmptyCatchBlock
- ExplicitCallToFinalize
- FinalizeDoesNotCallSuperFinalize
- FinalizeShouldBeProtected
- JUnitSpelling
  - framework methods are easy to misspell
- JUnitStaticSuite
  - suite() method needs to be public and static
DoubleCheckedLocking  (is broken)

```java
public Object getInstance() {
    if (inst == null) { // may be non-null
        synchronized (this) { // yet not fully created
            if (inst == null)
                inst = new Object();
        }
    }
    return inst;
}
```
OverrideBothEqualsAndHashcode

- Override both `boolean Object.equals(Object o)` and `int Object.hashCode()`, or override neither.

- HashMap uses first `hashCode()` method and then `equals()` method inside the hash-bucket...
Contract of `hashCode`

- Same objects must return same integer
- If two objects are `equals()`, the `hashCode()` method on each of them must produce the same integer.
- It is not required that if two objects are unequal according to the `equals()`, the two objects must produce distinct results.
hashCode Implementation

private String member;

public int hashCode() {
    // calculate hashCode from int-values
    // of all members and sum them up
    int result = 17;
    result = 37*result+member.hashCode();
    return result;
}
ProperCloneImplementation

• should be implemented with `super.clone()`

```java
public Object clone() {
    return new MyClass();  // wrong
}
```
clone Implementation

```java
public Object clone() {
    try {
        return super.clone();
    } catch (CloneNotSupportedException e) {
        // should not happen, we are Cloneable
        throw new InternalError("error in clone");
    }
}
```
public String bugga() {
    try {
        throw new Exception("My Exception");
    } catch (Exception e) {
        throw e;
    }
    finally {
        return "O.K.";   // bad
    }
}
Avoid!
More Warnings (to avoid)

• AvoidCatchingThrowable
  – there will be some exceptions to this rule...

• ExcessiveClass- & -MethodLength

• ExcessiveParameterList

• FinalizeOverloaded

• SignatureDeclareThrowsException

• SwitchStmtsShouldHaveDefault
ExceptionTypeChecking

try {
    returnString = sdf.format(value);
} catch (Exception e) {
    /* BAD STUFF */
    if (e instanceof NumberFormatException)
        System.out.println("NumberFormat!!!");
    if (ex instanceof IllegalArgumentException)
        System.out.println("illegal argument...!!!");
}
ConstructorCalls
OverridableMethodRule

```java
public class Senior {
    public Senior() {
        toString(); // may throw a NPE if overridden
    }

    public String toString() {
        return "IAmSenior";
    }
}
```
ConstructorCalls
OverridableMethodRule #2

```java
public class Junior extends Senior {
    private String name;
    public Junior() {
        super(); // inserted by compiler -> NPE
        name = "JuniorClass";
    }
    public String toString() {
        return name.toUpperCase();
    }
}
```
Update

DRINK

BOOGIE

REPEAT
Second Iteration (November)

• Improve quality ✔

• 1st iteration removed all 125 errors ✔
  – 100 seems to be a good work package ✔

• In the meantime ...
  – PMD has been updated
  – New rules are available (and in use ;-)  

• Now let’s fix another 8 rules
  – With approx. 100 violations
5 Errors (to fix)

- **AvoidCatchingNPE**
- **AvoidThrowingCertainExceptionTypes**
- **EmptyStatementNotInLoop**
  ```java
  if (false);
  {
   // will be executed
  }
  ```
- **ForLoopsMustUseBraces & WhileLoopsMustUseBraces**
3 Warnings (to fix)

- **BooleanInstantiation**
  
  `new Boolean(*) → Boolean.valueOf(*)`

- **UnnecessaryReturn**

- **UnusedImports** – Eclipse warning
New and Not Triggered #1

- **DontImportSun** (not in HP’s JDK)
- **EqualsNull**
  ```java
  if (x.equals(null)) { // never true
  ```
- **BrokenNullCheck**
  ```java
  if (s!=null || !s.equals("")) { // use && !
  ```

- **JUnitTestsShouldIncludeAssert**
- **TestClassWithoutTestCases**
New and Not Triggered #2

• **SimplifyStartsWith**
  
  `startsWith("a") \rightarrow charAt(0)=='a'`

• **AppendCharacterWithChar**
  
  `b.append("a") \rightarrow b.append('a')`

• **UseIndexOfChar**
  
  `s.indexOf("a") \rightarrow s.indexOf('a')`

• **AvoidProtectedFieldInFinalClass**
Questions
www.code-cop.org
CC Images

- agenda: http://www.flickr.com/photos/24293932@N00/2752221871/
- micro: http://www.flickr.com/photos/wessexarchaeology/183177852/
- trash: http://www.flickr.com/photos/togr/244902037/
- building: http://www.flickr.com/photos/stephen_rees/440201126/
- evil: http://www.flickr.com/photos/legofenris/4476593087/
- avoid: http://www.flickr.com/photos/howardlake/4850758742/
- repeat: http://www.flickr.com/photos/cyanocorax/288232991/
- questions: http://www.flickr.com/photos/seandreilinger/2326448445/