COMP0104 Software Development Practice: Checking Style

Jens Krinke

Centre for Research on Evolution, Search & Testing
Software Systems Engineering Group
Department of Computer Science
University College London

Programming guidelines

- Programming in a team means
  - many authors
  - working on unknown code
- If the programming style is different,
  new programmers will find it difficult
  to read and understand source code quickly.
- It is sensible for teams to create *programming guidelines*,
  to which the team members adhere.

Java Code Conventions

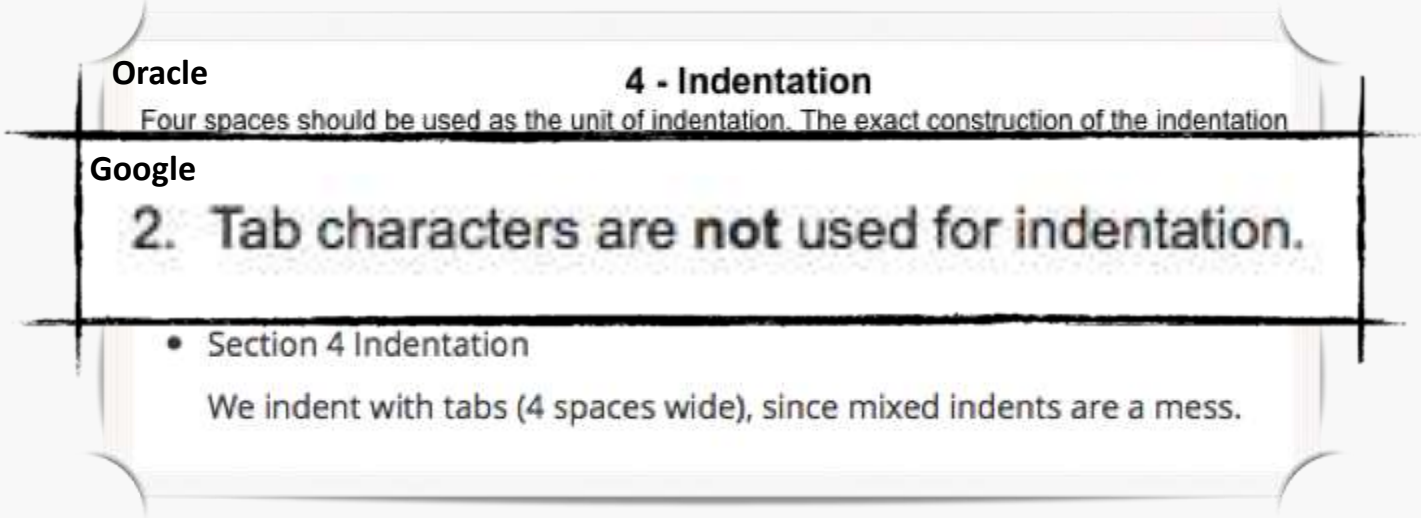
Sun Microsystems has established programming guidelines for the JAVA programming language, the "*Java Code Conventions*".

This set of guidelines provides direction for questions such as:

- How should files, classes, variables etc. be named?
- Where should comments be placed, and where not?
- How should source text be indented and wrapped?
- Where is white space required and where prohibited?

Deviations cause pain!

Problem: The default layout in Eclipse
does not adhere to the Java Code Conventions



Checking of guidelines

- The adherence to such guidelines is difficult as violations easily creep in.
- A good policy is to examine existing source texts against existing guidelines.
- Often enough, such checks require expensive reviews or audits, or need to be enforced during code review.

Conventions in Practice.

- Research reported that though developers recognized the importance of code conventions to code quality, they did not follow them in practice when meeting deadlines.
- Research also showed that less readable code usually has more convention violations than more readable code.

Automatic checking of guidelines

- A substantial part of the guidelines
  can be checked automatically.
- For example, it is possible to check
  the source texts with a scanner
  for required or prohibited white space.
- Automatic checking can be integrated
  into the development process and can be done continuously.

CHECKSTYLE

- CHECKSTYLE is highly configurable
- Comes with configurations conforming to
  - Java Code Conventions
  - Google Java Style

Example
(Sun did not follow its own guidelines)

```
$ java -jar checkstyle-all.jar \
-c sun_checks.xml Notepad.java
Starting audit...
Notepad.java:0: error: Missing package-info.java file.
Notepad.java:42: error: Using the '.*' form of import
should be avoided - java.awt.*.
Notepad.java:43: error: Using the '.*' form of import
should be avoided - java.awt.event.*.
Notepad.java:44: error: Using the '.*' form of import
should be avoided - java.beans.*.
...
```

173 errors are printed!

Notepad.java from Java SE Development Kit 8u66 Demos and Samples Downloads

Configuring CHECKSTYLE

- A lot of the warnings relate to missing Javadoc comments.
- If we create a copy of `sun_checks.xml` in `our_checks.xml`, we can configure which checks are performed.
- If the Javadoc lines are removed, 30% of the warnings disappear.

Example

- The following lines are removed:

```
<module name="JavadocMethod"/>
<module name="JavadocType"/>
<module name="JavadocVariable"/>
<module name="“JavadocStyle"/>
```

```
java -jar checkstyle-all.jar \
-c our_checks.xml Notepad.java
...
```

Other checks

The programmer violated a series of rules:

- `` `for' `` should always be followed by white space
- `` `{' `` should be preceded with whitespace.
- lines should not be longer than 80 characters
- some parameters should be `final`
- variables should be `private`
- …

Code Smells

- Many rules also search for **Code Smells**, problem indicators in the source code.
- Two typical examples are:
  - If a class overrides `equals`, then it must overrides `hashCode` too.
  - Local variables or parameters should not cover fields.

CHECKSTYLE checks

- CHECKSTYLE uses ANTLR to transform the source code into an abstract syntax tree.
- The syntax tree is traversed and every node is checked against the configured guidelines.
- The traversal uses the **Visitor** pattern.
- Own checks can be realised by instantiating the `TreeWalker`-Visitor.

Define your own check

```
import com.puppycrawl.tools.checkstyle.api.*;

public class MethodLimitCheck extends Check {

  private int max = 30;

  public int[] getDefaultTokens() {
    return new int[] {TokenTypes.CLASS_DEF,
                      TokenTypes.INTERFACE_DEF};
  }
```

```
public void visitToken(DetailAST ast) {
  DetailAST objBlock = ast.findFirstToken(
    TokenTypes.OBJBLOCK);

  int methodDefs = objBlock.getChildCount(
    TokenTypes.METHOD_DEF);

  if (methodDefs > max) {
    log(ast.getLineNo(),
        "too many methods, only "
        + max + " are allowed");
  }
}
```

Use the new check

- To integrate this check it is sufficient to add one line to the configuration file:

```
<module name="MethodLimitCheck"/>
```

- The new check will now be loaded and executed:

```
java -cp .:checkstyle-all.jar \
com.puppycrawl.tools.checkstyle.Main \
-c our_checks.xml …
```

Concepts

- A consistent programming style eases the comprehension of unfamiliar code.
- SUN has provided a set of guidelines for JAVA, the **JAVA Code Conventions**.
- CHECKSTYLE checks a set of accepted guidelines.
- **Code smells** are typical programming problems.
- CHECKSTYLE can be extended by own checks.