CheckStyle Explanation

Stefan Boodt 4322258
Arthur Hovenesyan
Tomas
Daan Schipper
Jordy

May 28, 2015

Abstract

This document describes the use of our CheckStyle design decisions.

Contents

Chapter 1

Explanation of differences

This chapter describes the differences between the default CheckStyle configuration google style and the configuration our project uses.

1.1 General

This section describes the general differences. Which consists of differences in indentation, as well as any other general subject that is not covered elsewhere.

- The indentation level is set to 4 because 4 is the default indentation for Java.
- We disallow the use of whitespaces in generics. So for instance List< Integer > is disallowed.
- The usage of tabs is allowed for indentation. This is because editors often use them behind the scenes.

1.2 Code

This section describes the diffences in settings considering the code itself.

- CheckStyle should ignore inline conditionals as they are not bad in all situations. The team can decide for themselves if a inline conditional is readable enough.
- Every equals method should have an @Override. This is inspired by FindBugs. The reason behind this is that the code can then throws errors if the user accidentally overrides a not standard equals methods. Which means it can throw an error if you accidentally write equals(Snippet) instead of equals(Object).

- Default should be the last case in a switch statement because this makes the code more readable.
- It is a CheckStyle error to catch an Exception, RunTimeException and Throwable anywhere in the code, since these catch things that should not be catched, such as OutOfMemoryExceptions and RunTimeExceptions. A RuntimeException should not to be catched as they are exceptions that make notion that something went wrong after the code was compiled. All other exceptions should be catched since they do not appear at runtime.
- All integers are not considered by the magic numbers check. This is because of the high usage of these integers in the test case and the waste of memory that comes from initializing that many variables simply to use as the expected outcome. Reviews of pull requests should mention magic numbers in non test classes.
- No classes are allowed to be in the default package as that means that they cannot be imported. CheckStyle should throw errors on this.
- The override of the method finalize should include a super call because of what finalize does. CheckStyle is configured to do something about this.
- CheckStyle has to throw an error if the cyclomatic complexity of a method is more than 10.
- CheckStyle should throw just a warning for non final parameters in a method or constructor definition. The reason behind this is that the input cannot be changed in another function call and when the final is forgotten then people can start to think they can be altered. It also makes some methods work weird if you forget the final modifier and accidentally change the parameter value.
- Long literals should contain an uppercase L and not a lowercase one since the lowercase l resembles a 1. CheckStyle should enforce this.
- CheckStyle should ignore it's design for extension argument since this requires methods to be final. While we learned during Software Engineering Methods that final methods should be used less.
- Maximal line length is set to 100 as this is more useful for new IDEs, in contrast to the standard 80 because of the default command line width.

1.3 Javadoc

This section focusses on CheckStyle checks that are considering the style of the javadoc comments.

- Missing @Override are seen as errors when @InheritDoc is used because the method does not acknowledge
- Annotations should appear between the javadoc and the method name because of convention. CheckStyle should throw an error if an annotation does not. This check is removed from the maven version.
- The package annotation is set as well to disallow the javadoc documentation for packages outside the package-info.java file.
- The non empty @ clause is reported as a CheckStyle error because we find it bad practice to write nothing behind @return or @param. This check was removed from maven version.
- The javadoc paragraphs is set so that you need to write a white line and a <p> for every paragraph in the javadoc documentation. This keeps multi paragraph comments readable. This check was removed from maven version.

1.4 Import

This section describes the differences between the default configuration and our own that consider the imports of classes in the code.

- We added an import order check to check if the imports are alphabetical and there is a white line between the groups. Static imports are ignored for their order in the maven version.

1.5 Temporary Checking Commands

This section explains how you can manually alter CheckStyle output in the code. Usage of these commands should be explained.

- To stop the CheckStyle tests from running you can write CHECKSTYLE:ON in your comments.
- To restart CheckStyle tests you write CHECKSTYLE:OFF in a comment.
- Alternatively the @SuppressWarnings can be used to deactivate CheckStyle.

1.6 Two different CheckStyle files

As you might have noticed there are 2 different CheckStyle files. This is due to the fact that mavens current plugin for CheckStyle (version 2.14) is using CheckStyle version 5.8 while the current version of CheckStyle, where the CheckStyle.xml is based on, is version 6.6. The current version of the eclipse-cs plugin, which integrates CheckStyle with Eclipse is using version 6.5. Please note that all java 8 syntax will be reported by maven CheckStyle as java 8 support was added to CheckStyle in version 5.9. Due to these version differences different files are required as some tests have been added or removed since CheckStyle 6.0. We decided to remove all the tests from the CheckStyle for maven file that are not compatible with CheckStyle 5.8.

Since tests only have been removed from the maven version of the file, the maven results for CheckStyle are incomplete, but do not find any new errors. However there are only 4 checks that have been removed because of this so the view from the maven file is still largely complete.

Chapter 2

References

- eclipse-cs version retrieved from: http://eclipse-cs.sourceforge.net/#!/ on 20-05-2015
- maven-checkstyle version retrieved from: http://blog.soebes.de/blog/2015/02/05/apache-maven-checkstyle-plugin-version-2-dot-14-released/ on 20-05-2015
- checkstyle version differences retrieved from: http://checkstyle.sourceforge.net/releasenotes.htm on 20-05-2015