HOGESCHOOL VAN AMSTERDAM

Comparison Quality tools

Project Game Technology

Jasper Meier, 500703267

February 2017

Glenn Hoff, Wouter van de Hauw, Michael Fuents Rodriguez, Rosa Corstjens, Guus van Gend, Kevin de Leeuw, Frederick van der Meulen & Jasper Meier

Project Game Technology
Team 5

Onder begeleiding van
Alexander Mulder, Product Owner
Anders Bouwer, Scrum coach
Eric Klok, Technisch consultant

Comparison Quality tools

Project Game Technology

Jasper Meier, 500703267
Game Technology
Hogeschool van Amsterdam
February 2017

ABSTRACT
The goal of this rapport is to find the usefulness of the SonarQube and ReSharper code-quality tools and compare the both to determine which one has the most potential for Project Game Technology or if it's better to use both.

1. INTRODUCTION
In order to improve the code quality of Project Game Technology, both an automated refactoring and code quality tool are needed. For this reason ReSharper and SonarQube have been researched for their capabilities.

At first the function of both programs seemed to represent each other, however after taking a closer look, you can see ReSharper focuses more on direct code improvements (while typing) and error prevention, whereas SonarQube has its own dashboard, and focuses more on the quality and code-conventions of GIT commits. In a project environment SonarQube would ensure the quality of different branches and the upholding of certain code-conventions and ReSharper would help the developers uphold this quality and give the access to tools, advice and hints.

ReSharper (https://www.jetbrains.com/resharper/) gives feedback and help in the following categories, this will also be the structure of the ReSharper chapter:

- Quality issues;
- Solution-wide inspection;
- Tracking;
- Customization;
- Navigation;
- Code-cleanup.

SonarQube (https://sonarqube.com), formally Sonar, is an open source platform for continuous inspection of code quality. The feedback categories and chapter structure for sonar is as follows:

- Clean code;
- Rules;
- Quality gates.

2. RESHARPER
This chapter will talk about the functions of ReSharper, a code-quality tool developed by Jetbrains. ReSharper watches and analyses your solution and offers help in the form of hints and tools which simplify the programming process.

2.1. Quality issues
ReSharper lets you find code quality issues, redundancies language usage opportunities and code improvement suggestions from a single window called *inspection results*, see Figure 1.
Within the Visual Studio environment two types of green underlining have been added:

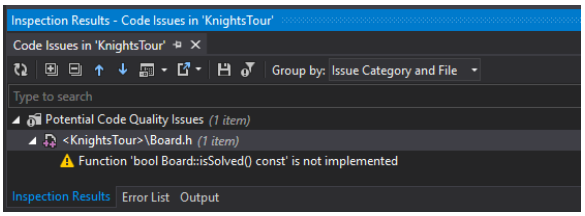

Figure1, inspection results

- A curly underlining: indicating a suggestion. These things aren't necessarily wrong, but they are probably useful to know and might result in cleaner code.
- A dashed underlining: indicates a hint. Hints do not influence the color of the *marker bar*, highlighted in yellow in Figure 2. A hint brings

your attention to a certain code detail and recommends a way of improvement.

Furthermore, ReSharper also supports quick fixes in

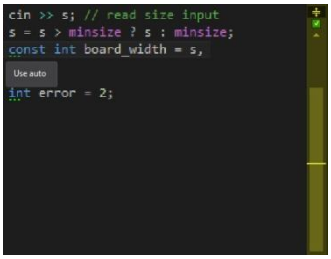

Figure 2, marker bar

code. There are more than 1000 quick fixes included with ReSharper. Examples include, declaring classes, variables and structs or including libraries. Applying a quick fix is done with the ctrl+enter command.

The "Inspect this" function, (ctrl + shift + alt + a), is a single shortcut to several ReSharper features that combine code analysis and navigation, including call tracking, see paragraph 2.3. tracking, value tracking, type hierarchy and a type dependency diagram.

2.2. Solution-wide inspection

Solution-wide warnings help you detect unused non private members. These inspections only work if you let ReSharper analyze your whole solution. Look for issues highlighted in code and the markerbar and you might find:

- Unused private declarations;
- Unused return values of non-private methods;
- Unaccessed non-private fields;
- Unused parameters in non-private members;
- Abstract or virtual events that are never invoked;
- Unassigned fields;
- Members and types what can be made internal instead of public.

2.3. Tracking

ReSharper can visualize the whole call sequence in one window. This is called *call tracking*, and it enables you to view and navigate through the call chain in your code.

This function is also available for values. This is called *value tracking*.At any location in your program you can point to a variable, field or parameter and see how it flows through your program. This can help you determine how a certain incorrect value might have been passed to a given point in your program.

2.4. Customization

At any time you may tell ReSharper to treat a particular analysis item. For example, make ReSharper display a particular inspection/notification as a warning, error, hint or suggestion according to your individual coding style.

ReSharper provides Structural Search and Replace to find all code that matches to a certain pattern, and optionally replace it with code matching another pattern. In combination with solution-wide analysis this tool can highlight code that matches the patterns, and provide quick-fixes to replace the code according to your replace patterns. This essentially means you can replace ReSharper's own set of inspections with your own custom inspections. For example when you're switching from an old API to a newer version, you can create search patterns to find usages of the old API and replace them with the new one.

2.5. Navigation

Each error, warning or suggestion is represented as a mark on the marker bar. Clicking the stripe navigates to the line of code that contains the error, warning or suggestion. You can also press alt+page down or alt+page up to navigate between errors, warnings and suggestions. If you want to skip the warnings and suggestions, you should press alt + shift + page up or alt + shift + page down. This just navigates between errors.

With ctrl+shift+v you open the clipboard history, which gives you the option of pasting a piece of text you copied earlier (even if you already copied something else in the meantime). ReSharper starts copying items to its clipboard history as soon as you start.

2.6. Code-cleanup

The Code-cleanup feature helps you instantly eliminate code style violations in one or more files, all with a single shortcut Ctrl+E, C (press c after releasing Ctrl + E). There are two profiles:

- Full cleanup, which applies all code style settings, except naming style
- Reformat code, which only applies your formatting rules.

ReSharper automatically prefers implicit datatypes (var i = 1 or auto i = 1), However, you can change your preferences for implicit/explicit typing.
For every kind of symbol, you can configure one of five casting options, prefixes, and suffixes, variations

for different access rights, abbreviations to preserve and other options. All violations of naming style are highlighted in the editor and can easily be fixed. In addition you can browse through and quick fix all naming style violations in your whole solution in the inspection results window (using the Find Code Issues feature).

3. SONARQUBE

This chapter will talk about the functions of SonarQube. This online dashboard and refactoring tool helps you protect and maintain a predefined project standards.

3.1. Clean Code

SonarQube has a main dashboard/ homepage where you have an overview of your current project(s) and the bugs, vulnerabilities and code smells in them. The dashboard page gives a nice overview of the project and supports access to the different bugs, vulnerabilities and code smells. This page gives you an immediate sense about the status of your project.

Once a bug or leak has been found, the right part of the screen shows a yellow box. Once these bugs and/or leaks have been fixed, the quality will improve. This method helps you evaluate your code before submission/merging them. This method also helps maintain the quality gate, however this will be discussed in paragraph 3.2.

When a push or pull request is made, SonarQube analyses the feature branches without being pushed to SonarQube, giving you the opportunity to fix your errors before they ever reach SonarQube.

SonarQube provides an easy overview of the main issues in the program and where they are located. You can see and sort the list to match your likings. For example, in the “Measures” page you can browse your project files in different ways, to highlight files that need your attention. More generally SonarQube provides a bubble chart that correlates different metrics to highlight other potential hot spots.

3.2. Quality Gates

To prevent files which do not meet your quality standards, a quality gate can be created. A quality gate is a set of requirements that tells whether or not a new version of a project can go into production. The default gate checks what happened on the leak period and fails if your new code got worse in this period.

3.3. Rules

The rules which SonarQube applies can all be defined by the user. SonarQube provides a standard set, however all these rules can be enables, disabled and changed to match your likings.

A rule is defined as a bug, vulnerability or a code smell. This can depend on the impact certain things (for example moving certain project files) can have a major impact on your project. Not only can the rules be defined by the type, but also how important the bug is among other bugs. This is shown as a minor, major or critical importance.

The standard set contains the following rulesets:103 rules for spotting bugs; 203 rules to spot code smells and 2 rules for finding vulnerabilities (http://dist.sonarsource.com/reports/coverage/misra_c++_2008.html).

CONCLUSION

Both ReSharper and SonarQube can be used to improve the quality of your project. When the two are combined ReSharper will give feedback while the developer is within the Visual Studio environment and SonarQube will give feedback when the developer is pushing or pulling code from the GIT repository.

While ReSharper lays its focus on improving code and giving hints about the usage of certain methods and libraries, SonarQube focuses more on semantic rules and GIT changes which could influence the progress of the program.

By predetermining and assigning the project code-conventions SonarQube will control and even stop developers from pushing, until the code meets the quality standard and code conventions.