Static Code Analysis

Lasitha Dahanayake

What is Static Analysis

- Static analysis, also called static code analysis, is a method of computer program debugging that is done by examining the code without executing the program.

How Static Analysis can help Software Quality

- There are two ways of inspecting software quality
    - Examine the behavior during the run-time (Dynamic analysis)
    - Inspect source code / Code reviews (Static analysis)

- Inspecting and analyzing the source code of the program before it is tested, lower the cost of finding and fixing bugs in software in the early stage of the development cycle.

Static Analysis Tools

- Developers are human-beings, and everyone make mistakes. So it's extremely hard to guarantee things can be done correctly for the first time.
- Employing static code analysis tools is one of the best practices in software development.
- Some of the Static analysis tools available
  - .NET
    - CodeIt.Right, FxCop, StyleCop ….
  - Java
    - PMD, CheckStyle, FindBug ….
  - JavaScript
    - JSHint, JSLint …

Static Analysis platforms

- Static analysis platforms come with server component
- Static analysis platforms support multiple programming languages and produce various matrices for analysis. Even maintain historical data.
  - SonarQube, Moose, Kiuwan are some of the examples
- Selecting the right tool
  - There are language specific static analysis tools which are coming as IDE plug-in, so they are helpful for the purpose of catching issues while coding.
  - Static analysis platforms support multiple languages and can handle multiple projects, can run independently without development environment. Even suitable for organization level static code analysis, provide various views and dashboards.

SonarQube

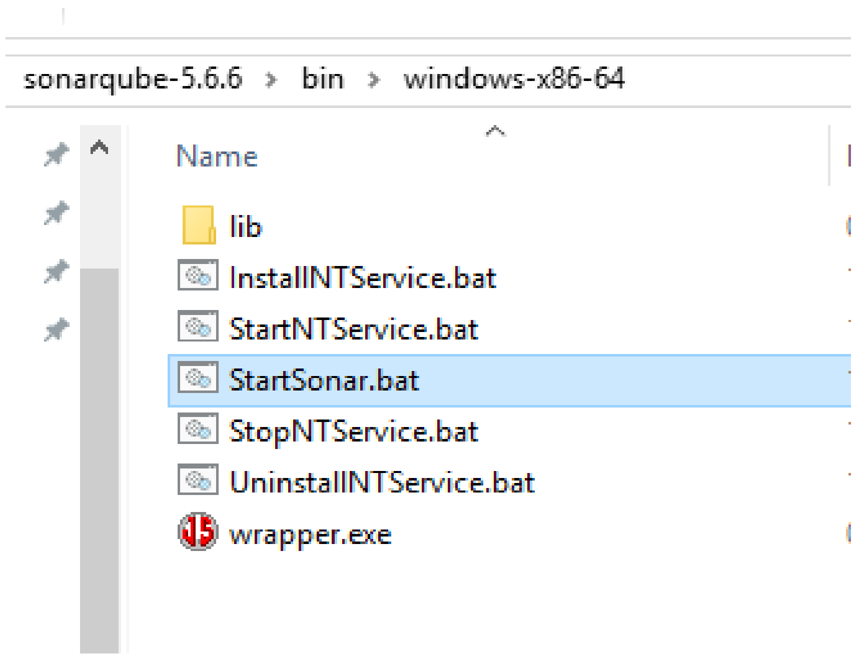
- Supports 20+ programming languages.
- More than 40 open-source and commercial plugins.
- Support integration with famous build tool such as Maven, Ant, MSBuild, Jenkins, Gradle
- Covers the 7 axis of code quality
  - Architecture & Design
  - Comments
  - Coding rules
  - Potential Bugs
  - Duplication
  - Unit Test
  - Complexity

SonarQube Server and Runner setup

- Download and install Java JDK if it's not available (Java 8).
- Download SonarCube from https://www.sonarqube.org/downloads/ and unzip to a desire location.
  - Make sure port 9000 is available for listening
- Download SonarCube runner from https://docs.sonarqube.org/display/SCAN/Analyzing+with+SonarQube+Scanner and unzip to a desire location.

Running SonarCube Server

- Start the SonarCube server using the startup script available in SONAR_HOME/bin/<Your Platform Folder>/<sonar.sh or StartSonar.bat>.



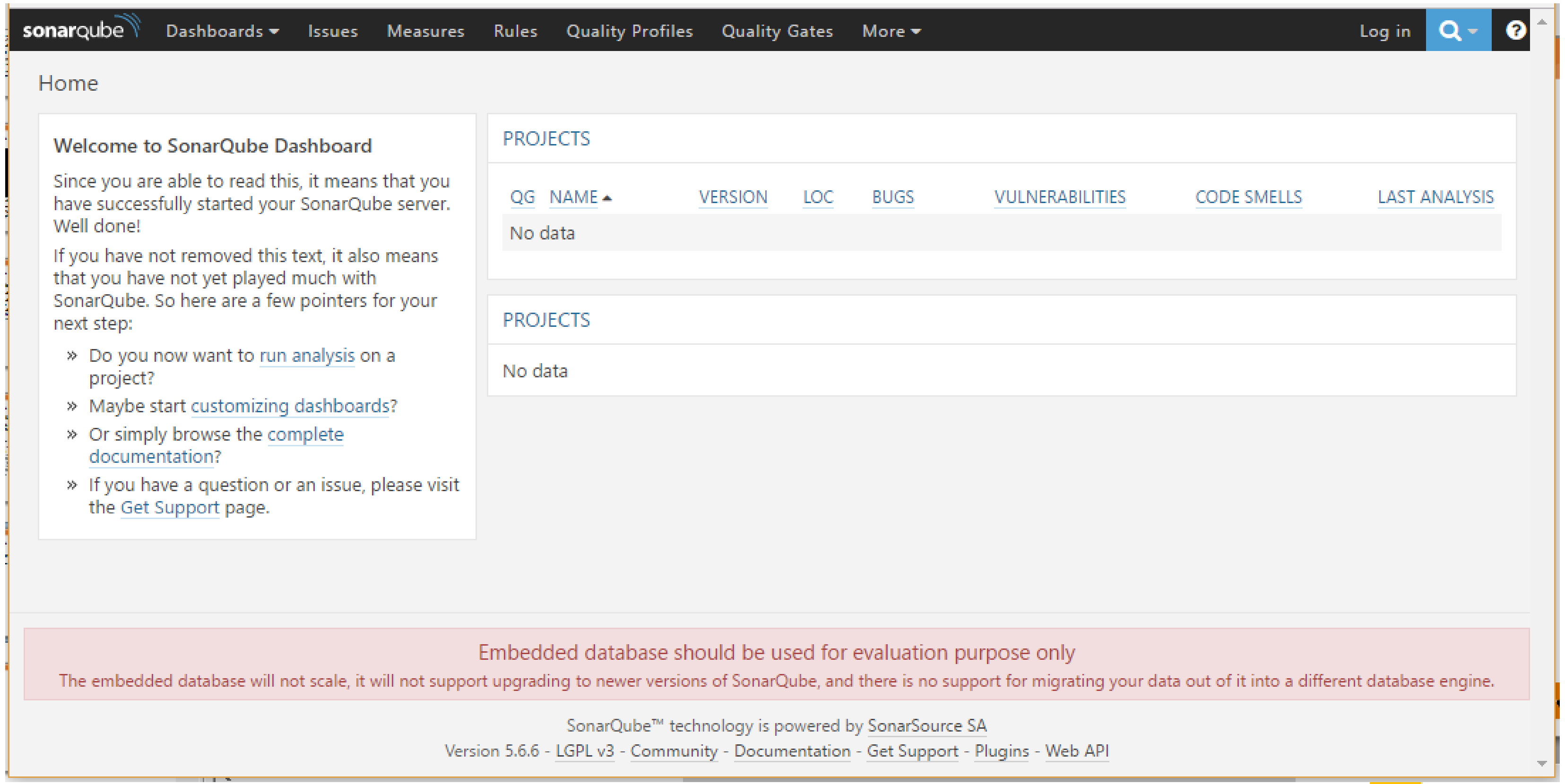
sonarqube-5.6.6 › bin › windows-x86-64

Name

- lib
- InstallNTService.bat
- StartNTService.bat
- StartSonar.bat
- StopNTService.bat
- UninstallNTService.bat
- wrapper.exe

Access SonarQube Server

- After SonarCube start, access the web interface of the SonarCube at http://localhost:9000.



Home

Welcome to SonarQube Dashboard

Since you are able to read this, it means that you have successfully started your SonarQube server. Well done!

If you have not removed this text, it also means that you have not yet played much with SonarQube. So here are a few pointers for your next step:

» Do you now want to run analysis on a project?
» Maybe start customizing dashboards?
» Or simply browse the complete documentation?
» If you have a question or an issue, please visit the Get Support page.

PROJECTS

QG	NAME ▲	VERSION	LOC	BUGS	VULNERABILITIES	CODE SMELLS	LAST ANALYSIS
No data							

PROJECTS

No data

Embedded database should be used for evaluation purpose only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA
Version 5.6.6 - LGPL v3 - Community - Documentation - Get Support - Plugins - Web API

SonarQube Runner (standalone)

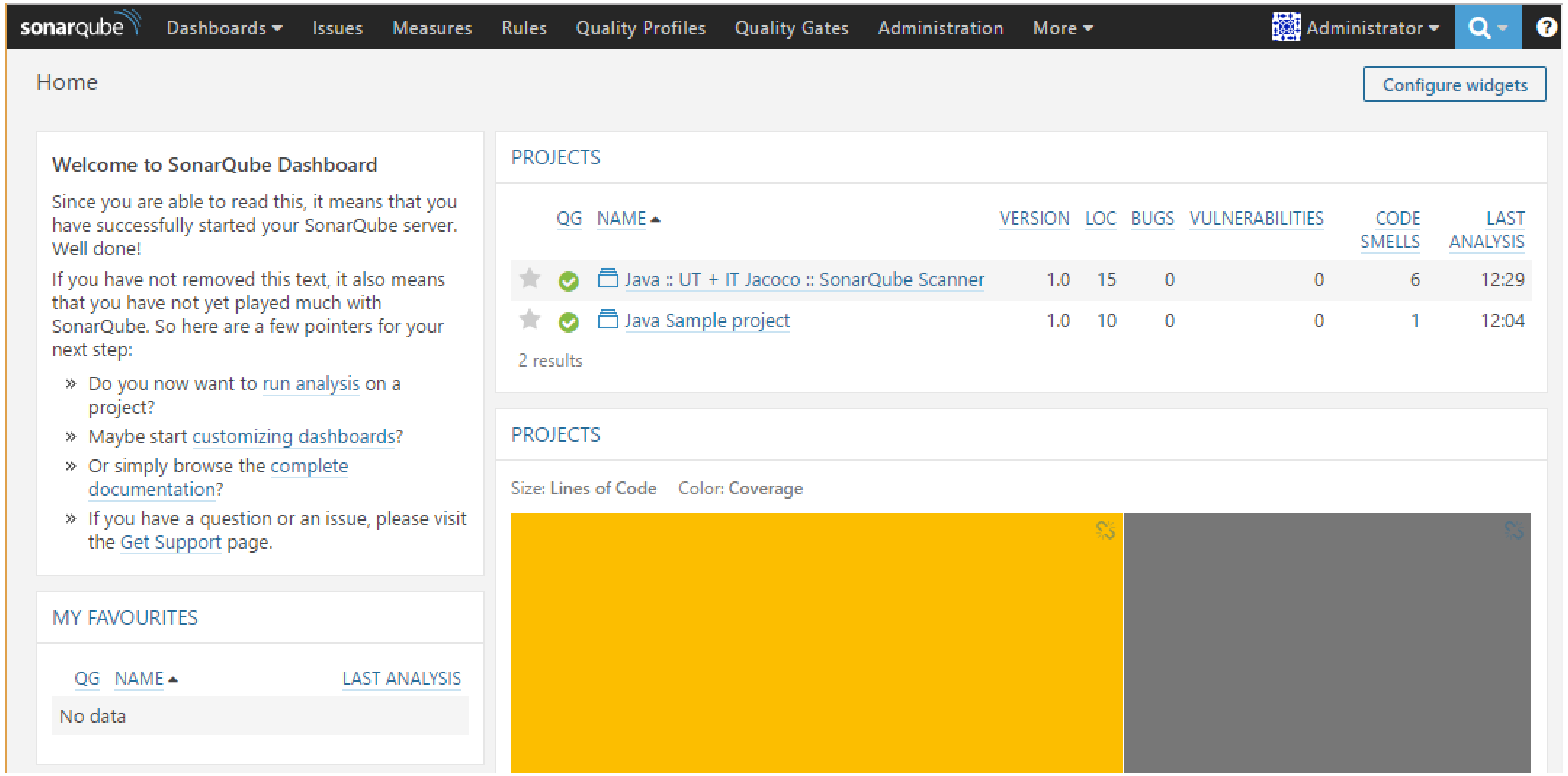
- Set the environment variable SONAR_RUNNER_HOME and set the value to the path you extracted SonarCube runner zip file (e.g C:\sonar-scanner-3.0.1.733-windows).
- Append Sonar runner bin folder to the path environment variable.
- Update Sonar runner setting in SONAR_RUNNER_HOME/conf/sonar-scanner.properties file (specify correct URL, e.g http://localhost:9000).
- In your project home folder, create a file call 'sonar-project.properties' and enter following lines as content (change values as you needed).

```
sonar.projectKey=mysample:project
sonar.projectName=Java Sample project
sonar.projectVersion=1.0
sonar.sources=src\\main\\java
```

- Run 'sonar-scanner' command from your project home (make sure project has no compile errors)

View SonarQube analysis results

- Access SonarCube server at http://localhost:9000



Welcome to SonarQube Dashboard

Since you are able to read this, it means that you have successfully started your SonarQube server. Well done!

If you have not removed this text, it also means that you have not yet played much with SonarQube. So here are a few pointers for your next step:

- Do you now want to run analysis on a project?
- Maybe start customizing dashboards?
- Or simply browse the complete documentation?
- If you have a question or an issue, please visit the Get Support page.

MY FAVOURITES

QG	NAME	LAST ANALYSIS
No data		

PROJECTS

QG	NAME	VERSION	LOC	BUGS	VULNERABILITIES	CODE SMELLS	LAST ANALYSIS
✓	Java :: UT + IT Jacoco :: SonarQube Scanner	1.0	15	0	0	6	12:29
✓	Java Sample project	1.0	10	0	0	1	12:04

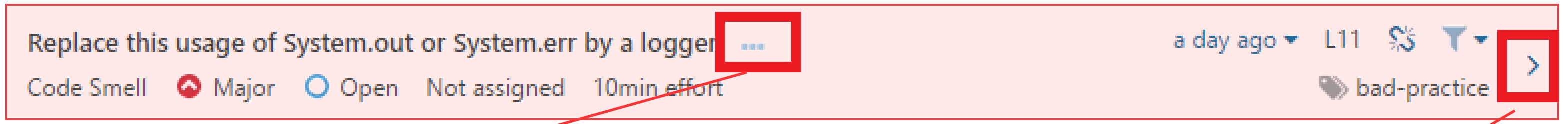
2 results

PROJECTS

Size: Lines of Code Color: Coverage

View SonarQube analysis results

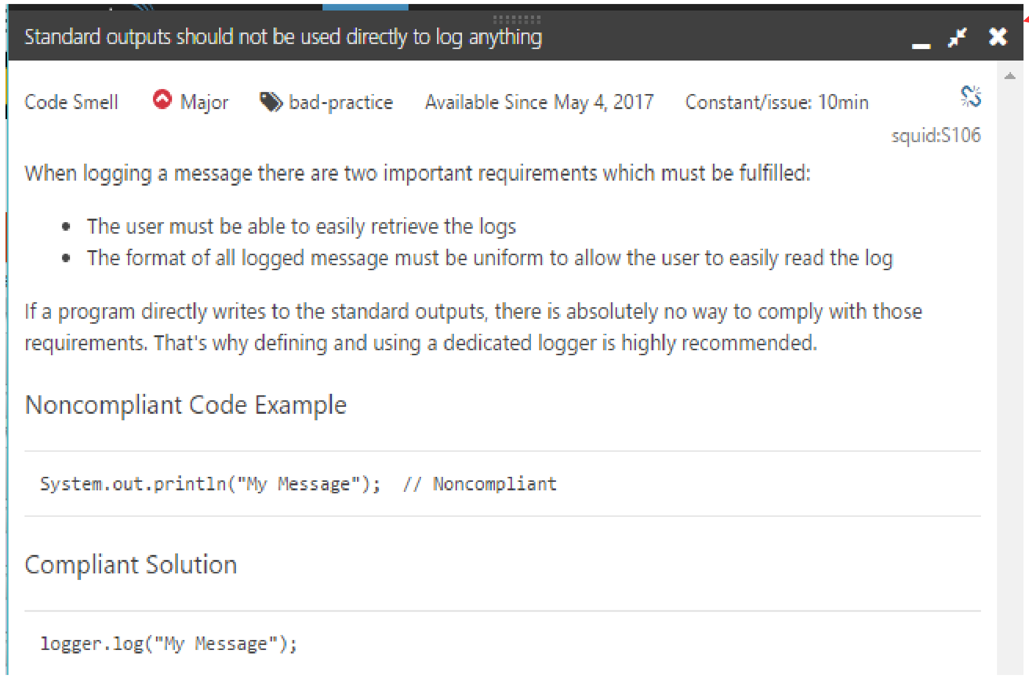
- Issue details



Replace this usage of System.out or System.err by a logger ...

Code Smell   Major   Open   Not assigned   10min effort

a day ago   L11   bad-practice



Standard outputs should not be used directly to log anything

Code Smell   Major   bad-practice   Available Since May 4, 2017   Constant/issue: 10min

squid:S106

When logging a message there are two important requirements which must be fulfilled:

- The user must be able to easily retrieve the logs
- The format of all logged message must be uniform to allow the user to easily read the log

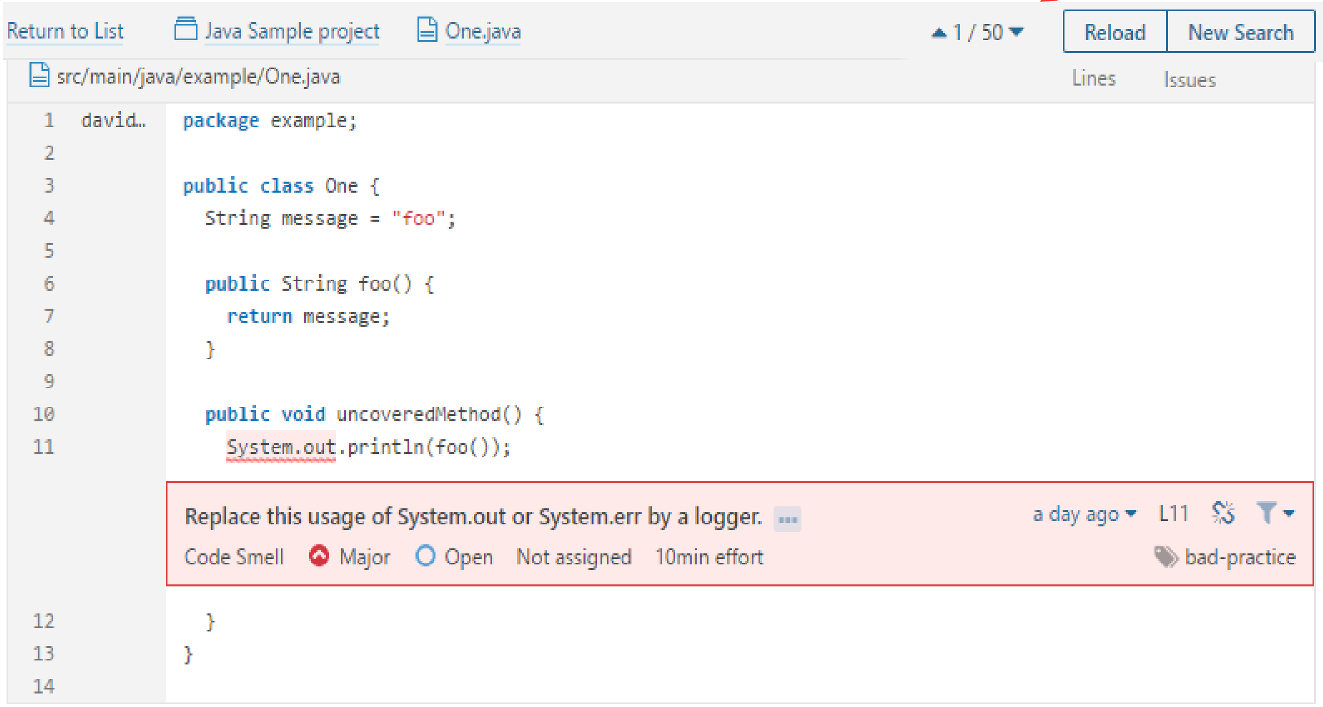
If a program directly writes to the standard outputs, there is absolutely no way to comply with those requirements. That's why defining and using a dedicated logger is highly recommended.

Noncompliant Code Example

```
System.out.println("My Message");  // Noncompliant
```

Compliant Solution

```
logger.log("My Message");
```



```
package example;

public class One {
  String message = "foo";

  public String foo() {
    return message;
  }

  public void uncoveredMethod() {
    System.out.println(foo());
```

Replace this usage of System.out or System.err by a logger. ...

Code Smell   Major   Open   Not assigned   10min effort

a day ago   L11   bad-practice

```
  }
}
```

Conclusion

- Please note that clearing all static analysis issues doesn't mean your code is in good quality.
- Static analysis tools only catch mistake that has a common pattern.
- So use wisely for your benefit.

Thank You

Lasitha Dahanayake

iTelasoft

Making IT Simple

ICTA

ideas actioned

QUALITY CODER 2017v1.0