

CODE QUALITY JAVA

Done by Ravi Theja Kolluru

Email ID: ravi.kolluru@accolitedigital.com

Assignment

A) Please do install following code analyzers in your IDE and document the installation process along with a screenshot of a report of code analysis for each plugin

- 1) Sonarlint (<https://www.sonarlint.org/>)
- 2) PMD (we haven't discussed about it but it's a similar tool) (<https://pmd.github.io/>)
- 3) Checkstyle (<https://maven.apache.org/plugins/maven-checkstyle-plugin/usage.html>)

B) Install the sonarqube server in your local machine and run sonarqube analysis against one of your projects. Choose any project of your choice which has at least 4 different classes.

Document the process and add screenshot of sonarqube report.

<https://www.sonarqube.org/downloads/> <https://www.oracle.com/in/java/technologies/javase-jdk11-downloads.html> Path to config `/conf/wrapper.conf`

C) Document at least 5 points about following secure coding standards

- 1) CWE
- 2) OWASP Top 10
- 3) CERT
- 4) SANS 25 <https://www.perforce.com/blog/qac/secure-coding-standards>

ANSWER

A)

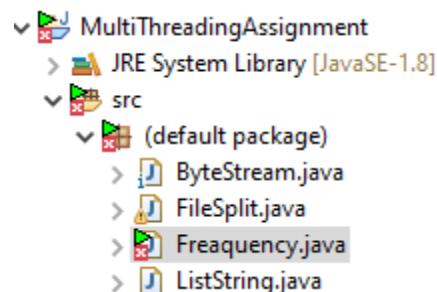
SONARLINT PLUGIN

1. Goto market-place in eclipse.
2. Type sonarlint in search box and install.
3. Restart Eclipse.
4. Right click on the file or folder, and analyze.

7 items		
Resource	Date	Description
Freaquency.java		Move this file to a named package.
Freaquency.java		Refactor your code to get this URI from a customizable parameter.
Freaquency.java		Replace this use of System.out or System.err by a logger.
Freaquency.java		Replace this use of System.out or System.err by a logger.
Freaquency.java		Replace this use of System.out or System.err by a logger.
Freaquency.java		Replace this use of System.out or System.err by a logger.
Freaquency.java		Use try-with-resources or close this "BufferedReader" in a "finally" clause.

PMD PLUGIN

1. Help -> Install New Software -> Add
2. Name it PMD and place the url of link to install.
3. To generate reports or to make any changes we can go to preferences and enable it.
4. Right click on the file or folder, PMD -> check code



The screenshot shows the Eclipse IDE with a Java file named `Freaquency.java`. The code is as follows:

```

2
3 import java.io.BufferedReader;
8
9 public class Freaquency {
10
11     public static void main(String[] args) throws IOException
12         long startTime = System.currentTimeMillis();
13
14         File file = new File("C:\\Users\\Ravi Theja Kolluru\\lec
15         FileInputStream fileStream = new FileInputStream(file);
16         InputStreamReader input = new InputStreamReader(fileStr
17         BufferedReader reader = new BufferedReader(input);
18
19         int tCount=1;
20         int countWord = 0;
21         String line;
22
23         while((line = reader.readLine()) != null) {
24             countWord += line.split(" ").length;
25

```

Below the code editor, the Checkstyle plugin's violations table is visible. It shows 23 violations for the `Freaquency.java` file. The table has the following columns: Element, # Violations, # Violations/K..., # Violations/M..., and Project.

Element	# Violations	# Violations/K...	# Violations/M...	Project
(default package)	23	126.4	2.56	MultiThreading...
Freaquency.java	23	851.9	23.00	MultiThreading...
LawOfDemeter	1	37.0	1.00	MultiThreading...
UseUtilityClass	1	37.0	1.00	MultiThreading...
LocalVariableCouldBeFinal	7	259.3	7.00	MultiThreading...
MethodArgumentCouldBeFinal	1	37.0	1.00	MultiThreading...
AssignmentInOperand	1	37.0	1.00	MultiThreading...
CommentRequired	2	74.1	2.00	MultiThreading...
AvoidFileStream	1	37.0	1.00	MultiThreading...
SystemPrintln	4	148.1	4.00	MultiThreading...
DataflowAnomalyAnalysis	1	37.0	1.00	MultiThreading...
CloseResource	3	111.1	3.00	MultiThreading...
NoPackage	1	37.0	1.00	MultiThreading...

CHECKSTYLE

1. Goto market-place in eclipse.
2. Type checkstyle in search box and install checkstyle plugin.
3. Can also take plugins from maven-checkstyle-plugin and add it in pom.xml
4. Restart Eclipse.
5. Right click on the file or folder, and analyze.

The screenshot shows a Java IDE with a file named `Frequency.java`. The code is as follows:

```

import java.io.BufferedReader;

public class Frequency {

    public static void main(String[] args) throws IOException {
        long startTime = System.currentTimeMillis();

        File file = new File("C:\\Users\\Ravi Theja Kolluru\\eclipse-workspace\\Frequency\\src\\main\\java\\com\\example\\Frequency.txt");
        FileInputStream fileStream = new FileInputStream(file);
        InputStreamReader input = new InputStreamReader(fileStream);
        BufferedReader reader = new BufferedReader(input);

        int tCount=1;
        int countWord = 0;
        String line;

        while((line = reader.readLine()) != null) {
            countWord += line.split(" ").length;
        }
    }
}


```

Below the code editor, the 'Checkstyle v...' tab is active, showing an overview of 49 violations. The table below summarizes the violations shown in the screenshot:

Checkstyle violation type	Occurrences
Name 'X' must match pattern 'X'.	1
'X' is not preceded with whitespace.	5
'X' has incorrect indentation level X, expected level should be X.	4
'X' is not followed by whitespace.	6
Distance between variable 'X' declaration and its first usage is X, but allo...	1
Line contains a tab character.	16
Line is longer than X characters (found X).	1
Missing a Javadoc comment.	1
'X' child has incorrect indentation level X, expected level should be X.	14

B) SONAR QUBE

1. Helps in inspection of code quality and supports multiple languages, generates reports.
2. Download the java 11 and sonar cube.
3. After installing them copy the bin folder location of the java11 into `conf/wrapper.conf` location in solar-qube folder.
4. Go to `bin/windows-x86-64/` in sonar cube folder and open command prompt, and run the `StartSonar.bat` file in the folder.
5. After it starts running goto browser and specify the web address `http://localhost:9000/`
6. Login to sonar-qube using admin as username and password for the first time and create a new password.
7. Add a new project in dashboard by giving key, generate and specify type of project.
8. Execute the command generated.
9. GO to overview in browser and generate reports.

Perspective: Overall Status Sort by: Name  0 projects

Once you analyze some projects, they will show up here.

Here is how you can analyze new projects

[Create new project](#)

0 of 0 shown

Analyze your project

We initialized your project on SonarQube, now it's up to you to launch analyses!

1

Provide a token

Generate a token

[Generate](#)

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your [user account](#).

```
INFO] More about the report processing at http://localhost:9000/api/ce/task?id=AXd0
INFO] Analysis total time: 17.220 s
INFO] -----
INFO] BUILD SUCCESS
INFO] -----
INFO] Total time: 34.619 s
INFO] Finished at: 2021-01-29T18:41:38+05:30
INFO] -----
```

- hibernate.assignment
 - src/main/java
 - hibernate.assignment.demo
 - EmployeeDemo.java
 - hibernate.assignment.entity
 - Address.java
 - Courses.java
 - Employee.java
 - EmployeeDetails.java

QUALITY GATE STATUS

MEASURES

Passed
All conditions passed.

New Code | **Overall Code**

0 Bugs Reliability **A**

0 Vulnerabilities Security **A**

0 Security Hotspots — Reviewed Security Review **A**

39min Debt 15 Code Smells Maintainability **A**

0.0% Coverage on 105 Lines to cover Unit Tests

0.0% Duplications on 329 Lines Duplicated Blocks 0

ACTIVITY

Assignment master

Last analysis had 1 warning January 29, 2021, 6:41 PM Version 0.0.1-SNAPSHOT

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

My Issues All Bulk Change to select issues to navigate 1 / 15 issues 39min effort

Filters Clear All Filters

Type CODE SMELL Clear

Bug 0

Vulnerability 0

Code Smell 15

Ctrl + click to add to selection

Severity

Blocker 0 Minor 12

Critical 1 Info 0

Major 2

Scope

src/_/java/hibernate/assignment/demo/EmployeeDemo.java

This block of commented-out lines of code should be removed. Why is this an issue? 6 minutes ago L76 unused

Code Smell Major Open Not assigned 5min effort Comment

src/_/hibernate/assignment/entity/Address.java

Rename this field "AddressId" to match the regular expression "[a-z][a-zA-Z0-9]*". Why is this an issue? 6 minutes ago L26 convention

Code Smell Minor Open Not assigned 2min effort Comment

src/_/hibernate/assignment/entity/Courses.java

Rename this field "CourseId" to match the regular expression "[a-z][a-zA-Z0-9]*". Why is this an issue? 6 minutes ago L30 convention

Code Smell Minor Open Not assigned 2min effort Comment

Quality Profiles Quality Gates Administration Search for projects... **A**

Perspective: Overall Status Sort by: Name Search by project name or key 1 projects

Assignment **Passed** Last analysis: 51 minutes ago

Bugs **A** Vulnerabilities **A** Hotspots Reviewed **A** Code Smells **A** Coverage 0.0% Duplications 0.0% Lines 329 **XS** Java, XML

1 of 1 shown

CODING - STANDARDS

CWE

- The CWE (Common Weakness Enumeration) is a list of errors that can be present in code and could lead to security issues.
- It acts as a measuring stick for security tools, and a standard for weakness identification, mitigation, and prevention efforts.
- It identifies weaknesses around concepts that are frequently used or encountered in software development.
- This includes all aspects of the software development lifecycle including both architecture and implementation.
- Henceforth it can be widely used by architects, developers, educators, and assessment vendors.
- It provides a variety of categories that are intended to simplify navigation, browsing, and mapping.

OWASP TOP 10

- The OWASP Top 10 is a standard awareness document for developers and web application security.
- It represents a broad consensus about the most critical security risks to web applications.
- Injection: It talks about injection flaws such as SQL, NoSQL, OS and LDAP injection.
- Broken Authentication: It points about allowing attackers to compromise passwords, keys, or session tokens.
- Sensitive Data Exposure: Related to financial, healthcare, and PII.
- XML External Entities (XXE): References within XML documents.
- Broken Access Control: View sensitive files, modify other users' data, change access rights.
- Security Misconfiguration: Misconfigured HTTP headers.
- Cross-Site Scripting (XSS): Browser API that can create HTML or JavaScript.
- Insecure Deserialization: Attacks, injection attacks, and privilege escalation attacks.
- Using Components with Known Vulnerabilities Components such as libraries, frameworks.
- Insufficient Logging & Monitoring: Tamper, extract, or destroy data.

CERT

- CERT secure coding standards include guidelines for avoiding coding and implementation errors as well as low-level design errors.
- It is secure coding standards for programming languages such as C, C++, and Java for developers.
- Used as a metric to evaluate source code (using manual or automated processes) for compliance with the standard.
- Risk assessment is classified into Severity, Likelihood, and Remediation Cost.
- Along with detecting security risks it also provides suggestions that can improve code quality.
- It is vital for reviewing code at all stages.
- Well-documented and enforceable coding standards are essential to secure software development.
- Few of the CERT secure coding guidelines are mapped directly with weakness in CWE.
- Some are not because some coding errors do not relate to any given weakness.

SANS 25

- SANS Top 25 is a list of highly critical Common Weakness Enumeration's (CWE) errors.
- These are errors can cause attackers to steal data, control and manipulate applications.
- Few of the errors include
 1. Improper Restriction of Operations within the Bounds of a Memory Buffer
 2. Improper Input Validation
 3. Out-of-bounds Read
 4. Information Exposure
 5. Integer Overflow or Wraparound
