Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Laboratorio de Desarrollo y Herramientas:

Práctica 3: Herramientas de Calidad del Producto Software y Documentación.

(SonarQube, Maven y Doxygen)

David Hernández Lozano
(alu0101361093@ull.edu.es)

Índice:

1. Introducción.

En esta práctica se ha convertido el proyecto “ExpositoTOP” en un proyecto soportado por Maven, se ha documentado usando Doxygen, se han instalado algunos plugins de Maven y finalmente se ha integrado SonarQube dentro de Maven para analizar el proyecto.

El código fuente (modificado) del proyecto, el sitio del proyecto, la documentación y demás se pueden encontrar en el repositorio de GitHub: https://github.com/alu0101361093/LDHP3

2. Plugins instalados en Maven

Para este proyecto, se han instalado y configurado distintos plugins, aparte de los que ya vienen por defecto con Maven.

Todos los plugins que se comentan a continuación son de **tipo reporting**, por lo que se ejecutan al generar el sitio **(mvn site).**

Generan informes dentro de la sección **“Project Reports”** del sitio web.

2.1. Checkstyle

Genera un informe sobre el estilo de código usado y las buenas prácticas de programación.

El informe contiene “tags” de información, avisos o errores.

Para el proyecto ExpositoTOP, este es el resultado:

Summary

Files	Info	Warnings	Errors
11	0	0	918

Un ejemplo de error:

Severity	Category	Rule	Message	Line
Error	design	HideUtilityClassConstructor	Utility classes should not have a public or default constructor.	20

2.2. PMD

PMD es una herramienta de análisis de código estático que ya se ha mencionado previamente en la asignatura. Analiza el código y busca fallos o errores de acuerdo a buenas prácticas.

top/TOPTWGRASP.java

Rule	Violation
UnusedLocalVariable	Avoid unused local variables such as 'newDepot'.
CollapsibleIfStatements	These nested if statements could be combined

2.3. JXR

Genera referencias al código fuente que se pueden ver desde Javadoc, y viceversa.

Muestra secciones del código fuente en el informe.

2.4. Surefire-Report

Parsea los ficheros TEST-*.xml generados y muestra tanto los tests como los resultados de los tests en el sitio web.

2.5. JDepend

Genera métricas a partir del proyecto, como número de clases, dependencias cíclicas, la abstracción de clases y métodos, estabilidad, etc.

es.ull.esit.utilities

Afferent Couplings	Efferent Couplings	Abstractness	Instability	Distance
1	2	0.0%	67.0%	33.0%

Abstract Classes	Concrete Classes	Used by Packages	Uses Packages
None	es.ull.esit.utilities.BellmanFord es.ull.esit.utilities.PowerSet	main.java.top	java.lang java.util

2.6. Taglist

Informa sobre la cantidad de “tags”, como @TODO, que se encuentran en el código.

2.7. Javadoc

Javadoc es una de las herramientas de documentación de código más ampliamente utilizada en Java.

Su plugin de Maven genera la documentación en formato Javadoc dentro del sitio Maven.

Al juntarlo con el plugin JXR, se puede acceder directamente al código fuente desde la página de Javadoc, y viceversa.

2.8. OWASP

OWASP es una herramienta muy potente que contempla fallos de vulnerabilidad en el código y busca dependencias dentro del mismo.

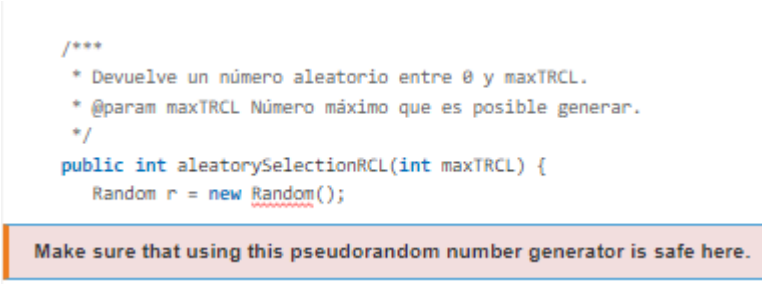
3. SonarQube y Maven

Se ha creado un proyecto SonarQube seleccionando como herramienta de automatización del código Maven.

Esto nos permite llamar a SonarQube con el comando “mvn clean verify …”, que genera la información en la página de SonarQube.

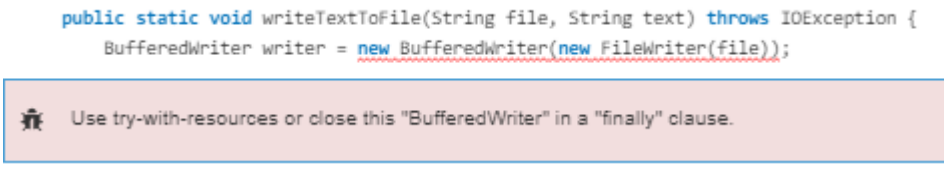
Al analizarlo con SonarQube, el proyecto tiene una vulnerabilidad de seguridad, relacionada con la generación de números aleatorios.

La solución es cambiar la librería “Random” por “SecureRandom”.




```
SecureRandom r = new SecureRandom();
```

El proyecto también tiene varios bugs, por ejemplo el que se muestra a continuación, al no usar una sentencia “try” para abrir un fichero.




```
try (BufferedWriter writer = new BufferedWriter(new FileWriter(file)))
```

El proyecto tiene otros 4 bugs, pero me parece más complicado arreglarlos debido a que no comprendo del todo el funcionamiento del código, y está el riesgo de que al intentar arreglarlo de la manera que muestra SonarQube, el programa principal no funcione.

5. Referencias.

1. https://maven.apache.org/plugins/
2. SonarQube