LINT REPORT

22 DE MAYO DE 2022

E2.07

https://github.com/gonmarfer2/Acme-Toolkits

Jaime Borrego Conde: jaiborcon@alum.us.es

Antonio Campos Gil: antcamgil@alum.us.es

Ana Conde Marrón: anaconmar@alum.us.es

Gonzalo Martínez Fernández: gonmarfer2@alum.us.es

Jaime Moscoso Bernal: jaimosber@alum.us.es

Enrique Muñoz Pérez: enrmunper@alum.us.es

Tabla de contenido

Resumen ejecutivo

El presente documento recopila los malos olores que el *plugin Lint* ha encontrado en la aplicación *Acme Toolkits*, con el fin de detectarlos y darles solución. En consecuencia, se logra mejorar requisitos no funcionales relacionadas con la calidad del producto, como la mantenibilidad o la legibilidad del sistema.

Tabla de revisión

Número de revisión	*Fecha*	*Descripción*
1	22/05/2022	Desarrollo del documento

Introducción

Este documento nace con la finalidad de aportar una descripción de los malos olores detectados en el proyecto. Un mal olor es un indicio de un problema que en el futuro podría conllevar consecuencias graves, normalmente debido a que el código no se ha programado manera ideal, puesto que ello llevaría mayor tiempo y esfuerzo. Por ende, no resolverlos genera lo que se denomina «deuda técnica». Cuanto más avance el tiempo, si no se mejora el código, el manejo del proyecto, la introducción de cambios o el entendimiento del código puede llegar a hacerse imposible.

Para obtener los malos olores que contiene el proyecto *Acme-Toolkits*, se procesó el proyecto a través de la herramienta *SonarLint*. Este plugin ofrece un análisis de estos basándose en siete principios o dimensiones: no seguir los estándares de código, defectos potenciales, distribución de complejidad pobre, duplicados, exceso de comentarios, falta de test y un mal diseño.

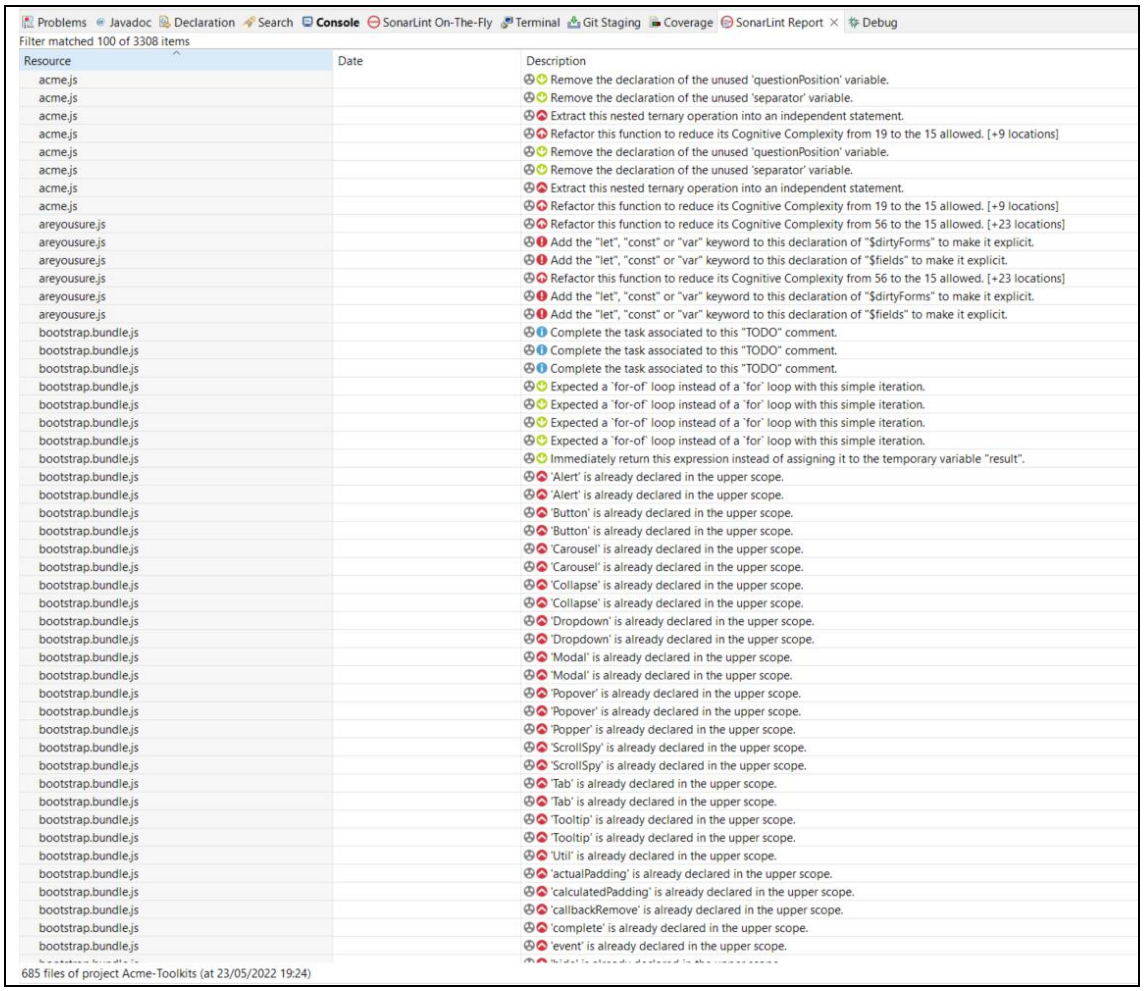
Por lo tanto, como se ha mencionado anteriormente, mantener en un proyecto malos olores es tremendamente perjudicial. Con el fin de detectarlos, corregirlos y mejorar los resultados, a continuación, se listarán los malos olores inherentes del proyecto, explicando asimismo sendos orígenes, el porqué de su severidad y, de ser posible, se propondrá una solución. Una vez se hayan documentado, se corregirán en el proyecto.

Finalmente, una vez se profundice en los malos olores encontrados, se presentará una conclusión que permitirá estudiar y reflexionar sobre todos los conceptos y conocimientos que se han trabajado a lo largo de la realización del documento.

Visión general

El equipo ha superado con creces las expectativas respecto a la redacción de código. A fecha de realización de este documento, la rama *master* no presenta ningún mal olor o bug introducido por el equipo de desarrollo.

A continuación, se muestran los resultados del análisis de SonarLint llevado a cabo previamente sobre dicha rama. Todos son problemas introducidos en ficheros JavaScript, por lo que no han sido causados por el equipo durante el desarrollo.



Conclusiones

El estudio y la realización de este documento ha permitido reafirmar el buen trabajo que ha realizado el equipo a lo largo del período de la cuarta entrega. En numerosas ocasiones ha bastado revisar el código para detectar malas prácticas antes de introducirlas en la versión final.

Sin embargo, no es posible que un programa esté exento de errores. Es muy probable que, indagando en el funcionamiento de la aplicación, se detecte el mal funcionamiento de una característica o una acción que no debería salir tal y como tendría que haberse obtenido a causa de una mala praxis en el código.

Ignorar la deuda técnica que aparece al no solucionar estos errores en su debido momento provoca un efecto «bola de nieve» que acaba volviendo el manejo de la misma insostenible. Debido a esto, el grupo realiza un gran esfuerzo por identificarlos y eliminarlos cuanto antes.

Por ello, aunque solo queda una entrega, el equipo ha de tener ojo avizor ante posibles problemas y desperfectos que ensucien el gran esfuerzo que se ha estado realizando durante el resto del curso académico. No obstante, según la evolución experimentada a lo largo de los últimos meses, es prácticamente seguro que no se hallarán trabas, malas prácticas o *bugs* en el código de la aplicación.

Bibliografía

Intencionalmente en blanco.