

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática




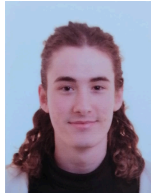
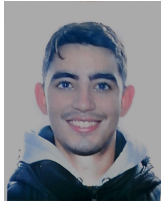
DP2-Reporte de Linting D3



Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas II

Curso 2023 – 2024

<u>Group:</u> C1.016	
<u>Repository:</u> https://github.com/luchersol/Acme-SF-D02-24.1	
<u>Student #1</u>  UVUS: luchersol Name: Herencia Solís, Lucas Manuel Email: luchersol@alum.us.es	<u>Student #2</u>  UVUS: marcallop7 Name: Calero López, Marina Email: marcallop7@alum.us.es
<u>Student #3</u>  UVUS: edurobrus Name: Robles Russo, Eduardo Email: edurobrus@alum.us.es	<u>Student #4</u>  UVUS: josmirmar2 Name: Miret Martín, José Manuel Email: josmirmar2@alum.us.es
<u>Student #5</u>  UVUS: dangalcan Name: Galván Cancio, Daniel Email: dangalcan@alum.us.es	
<u>Date:</u> Sevilla Abril 24, 2024	

Índice de contenido

1. Versiones	2
2. Lista de bad smells	2
3. Conclusión	2
5. Bibliografía	2

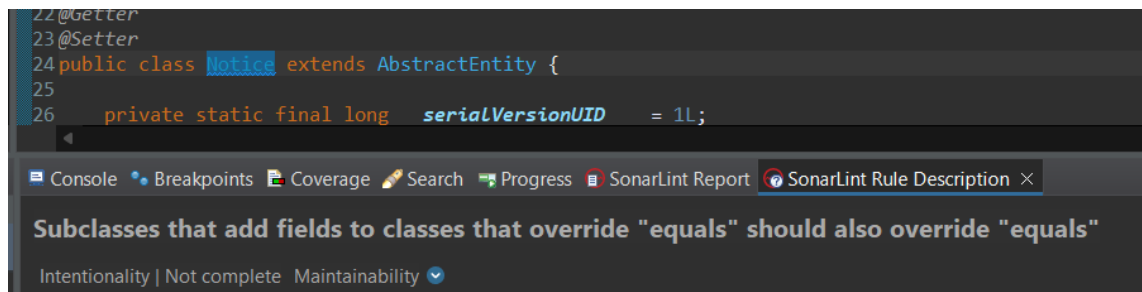
1. Versiones

Versión	Fecha	Autor
1.0	24/04/2024	José Manuel Miret

2. Lista de bad smells

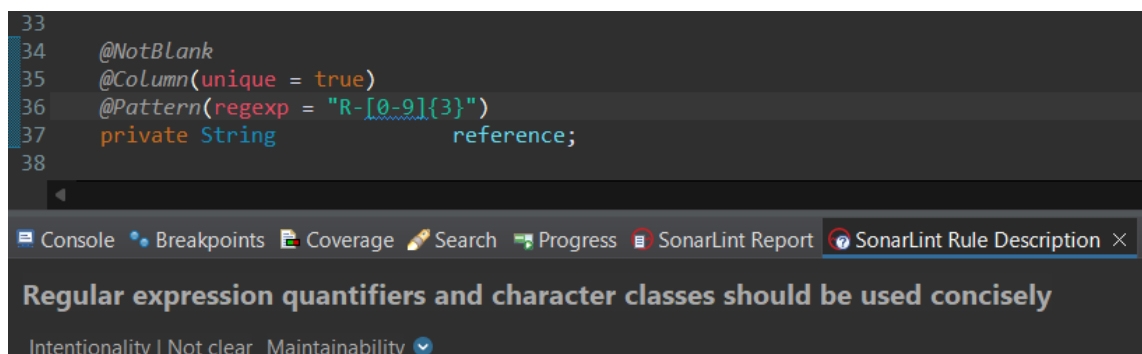
- Sobrecribir el método equals(): DONE

Su descripción es 'Subclasses that add fields to classes that override "equals" should also override "equals"'. Este bad smell me sugiere que sería beneficioso reescribir el método equals() en varias clases (como Notice o Risk) para establecer un método que determine cuándo dos instancias de esas clases son iguales. Sin embargo, dado que sabemos que serán consideradas iguales cuando tengan el mismo id, he optado por no seguir esta recomendación. Además, ninguna entidad tiene su método equals() modificado.



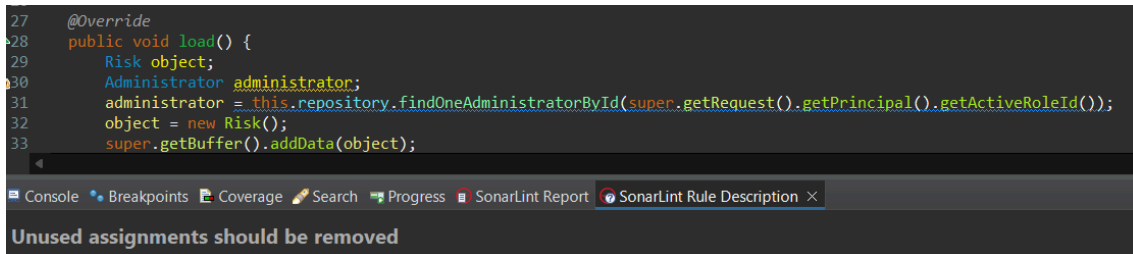
- Poner \d en lugar de [0-9] en expresiones regulares: DONE

Su descripción es 'Regular expression quantifiers and character classes should be used concisely'. En lugar de emplear la abreviatura \d para representar cualquier dígito numérico del 0 al 9, sería más conveniente. Sin embargo, por razones de claridad y debido a que esta fue la especificación requerida, he optado por mantenerla como está. Esta elección no tiene impacto en el funcionamiento del código y, además, ayuda a que la expresión sea más clara y comprensible.



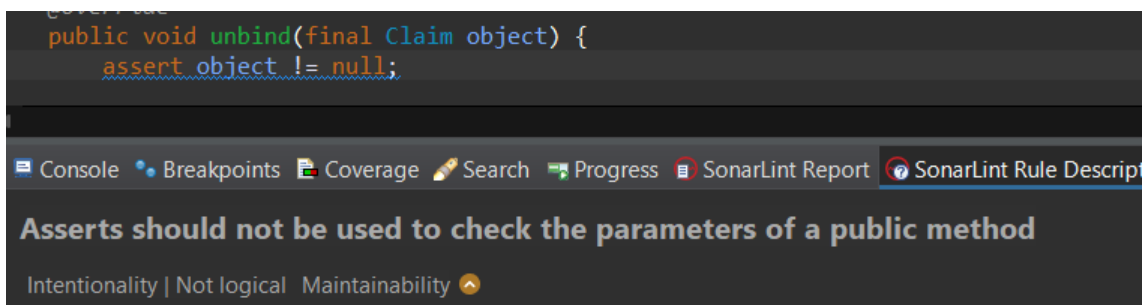
- Eliminar código inservible: DONE (revisar)

Aparece 'Unused assignments should be removed'. Considero que este es un bad smell, ya que si en el futuro se modifica alguna parte del código y ese código no utilizado genera errores, podría causar problemas. Por lo tanto, estoy considerando eliminar ese código para resolver este mal diseño.



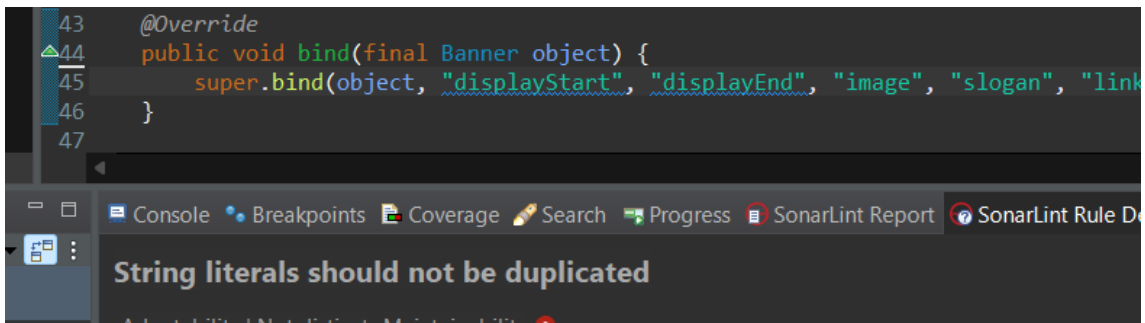
- Evitar usar assert: DONE

'Replace this assert with a proper check'. No considero que este problema de calidad de código sea crítico, dado que el código continúa garantizando que el objeto no sea nulo, lo cual cumple su propósito. Por esta razón, he optado por no abordarlo. Además, esta instrucción forma parte de las pautas recomendadas para implementar servicios utilizando el framework que estamos utilizando, por lo que no tengo planeado modificarla.



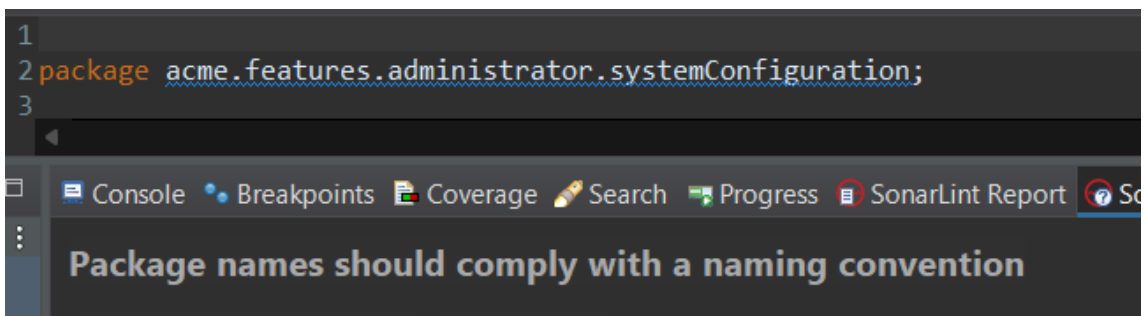
- Crear constantes genéricas en lugar de repetir la misma variable en varios sitios: DONE

Este code smell tiene la siguiente descripción: 'String literals should not be duplicated'. Este problema también se repite en varios archivos. Aunque este asunto de calidad de código no es crítico y sigue cumpliendo su propósito sin afectar la comprensión o el funcionamiento del código, es importante mencionar que estas estructuras fueron recomendadas durante la capacitación de servicios que recibimos. Por lo tanto, he seguido esta estructura en mi código y no tengo intención de cambiarla.



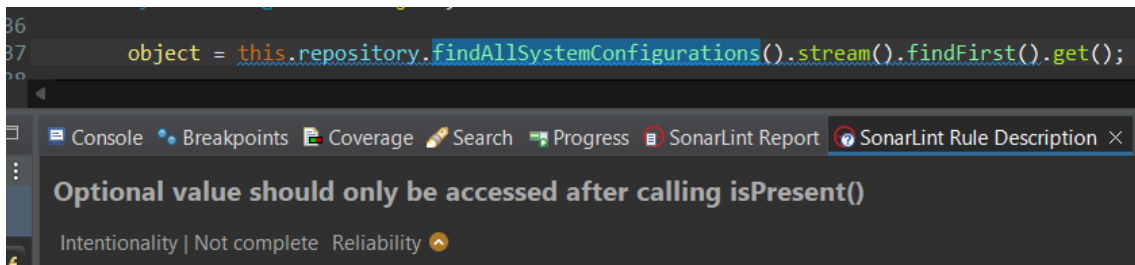
- Renombrar archivos: DONE

En los paquetes, aparece muchas veces 'Package names should comply with a naming convention'. Considero que esto no es simplemente un bad smell, sino más bien un incumplimiento de las prácticas comunes y estándares de nombrado de paquetes. Aunque este problema no afecta la comprensión del código ni su funcionamiento, es importante mantener estándares consistentes en el código base. Sin embargo, debido a que no tiene un impacto crítico en la funcionalidad actual y no afecta negativamente la comprensión del código, he decidido no modificarlo en este momento.



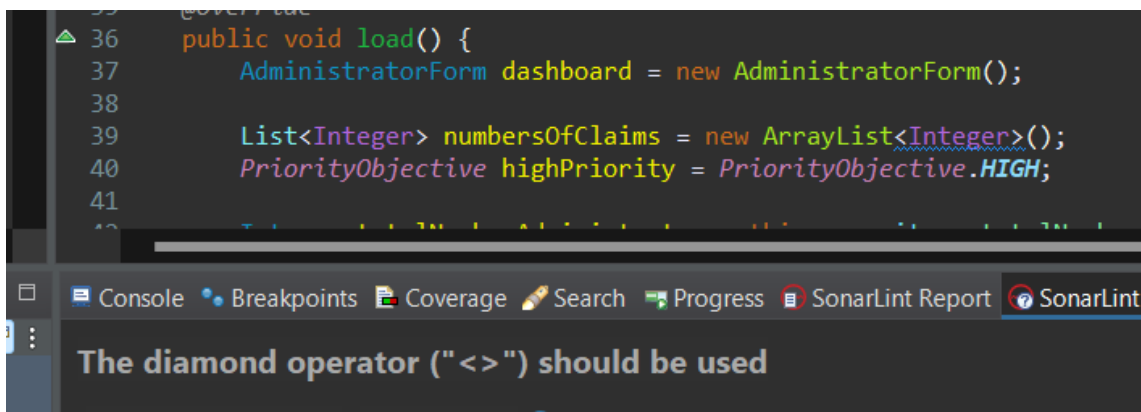
- Valores opcionales utilizar una función para ver si está presente: DONE (revisar con equipo)

Aparece 'Optional value should only be accessed after calling isPresent()'. Este problema podría considerarse un "bad smell" debido al uso de la clase Optional en Java para representar un valor que podría estar ausente, lo cual requiere verificar si el valor está presente antes de acceder a él. Aunque, dado que en la base de datos siempre hay exactamente un systemConfiguration disponible y obtener todos los systemConfigurations de la base de datos, veo buena idea elaborar un cambio en el código para mejorarlo. Por lo tanto, considero que este aspecto debo modificarlo.



- Redundancia al crear un ArrayList: DONE

Aparece 'The diamond operator ("<>") should be used'. Este problema surge al crear un ArrayList donde se especifica el tipo de elementos que contendrá. Al definir el tipo de elementos al crear la variable, repetir este tipo al definir el ArrayList resulta redundante. Por esta razón, hemos decidido abordar este asunto y realizar modificaciones correspondientes.



3. Conclusión

En resumen, a pesar de que puedan existir algunos aspectos que podrían considerarse "bad smells" en el código, podemos concluir que en general el código no es de baja calidad y está funcionando correctamente. Es fácil de entender y leer, lo cual es fundamental para su mantenimiento y futuras mejoras.

4. Bibliografía

No hay bibliografía presente para esta entrega.