

Universidad de Sevilla

Escuela Técnica Superior de Ingeniería Informática


## **DP2-Reporte de Linting Student 4 D4**



Grado en Ingeniería Informática – Ingeniería del Software

Diseño y Pruebas II

Curso 2024 – 2025

|   |   |
|---|---|
| <b><u>Group:</u></b>  | C1.018  |
| <b><u>Repository:</u></b>   | <a href="https://github.com/alenicbra/acme-ans">https://github.com/alenicbra/acme-ans</a> |
| <b><u>Student #4</u></b>  |   |
|  |   |
| <b>UVUS:</b> josmirmar2   |   |
| <b>Name:</b> José Manuel Miret Martín   |   |
| <b>Email:</b> josmirmar2@alum.us.es   |   |
| <b><u>Date:</u></b>   | Sevilla mayo 26, 2024   |

## **Índice de contenido**

|                               |          |
|-------------------------------|----------|
| <b>1. Versiones</b>           | <b>2</b> |
| <b>2. Lista de bad smells</b> | <b>2</b> |
| <b>3. Conclusión</b>          | <b>2</b> |
| <b>5. Bibliografía</b>        | <b>2</b> |

## 1. Versiones

| Versión | Fecha      | Autor                    |
|---------|------------|--------------------------|
| 1.0     | 26/05/2025 | José Manuel Miret Martín |

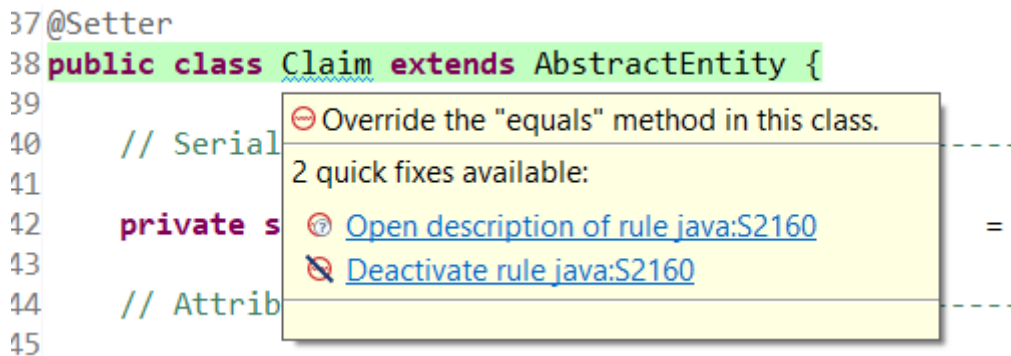
## 2. Lista de bad smells

### ● Eliminar asignaciones no utilizadas en este código.

Este mensaje aparece debido a la regla "Unused assignments should be removed".

Reconozco que se trata de un bad smell, ya que las asignaciones no utilizadas pueden generar confusión y afectar la mantenibilidad del código. Sin embargo, en este caso decido no eliminarla, ya que puede aportar claridad al flujo lógico y servir como referencia explícita de una intención del programador.

Por lo tanto, aunque es un bad smell, no considero necesario eliminar esta asignación en este contexto específico.

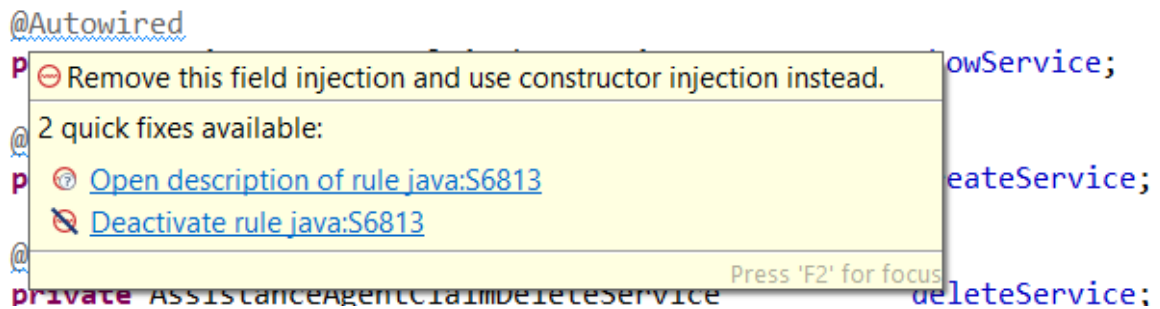


### ● Evitar la inyección por campos y utilizar inyección por constructor.

Este aviso corresponde a la regla "Remove this field injection and use constructor injection instead".

Aunque reconozco que la inyección por campos es un bad smell y que la inyección por constructor es una mejor práctica por razones de testabilidad y diseño limpio, en este caso decido no aplicar el cambio. La implementación actual es funcional, compatible con el framework y no representa un riesgo inmediato.

Por tanto, no considero necesario eliminar este bad smell por ahora, priorizando la estabilidad y simplicidad del sistema.

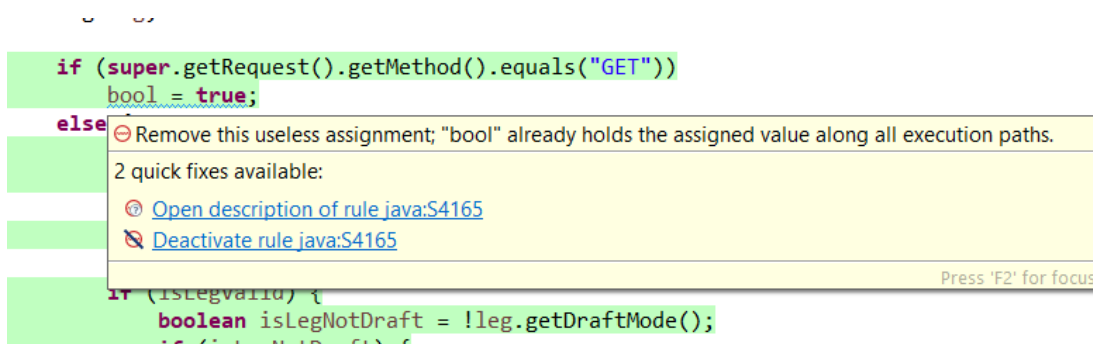


- **Eliminar asignación inútil: la variable ya contiene ese valor en todos los caminos de ejecución.**

Este mensaje responde a la regla "Remove this useless assignment; 'bool' already holds the assigned value along all execution paths".

Aunque la regla "Remove this useless assignment" (java:S4165) sugiere eliminar la asignación porque la variable ya contiene ese valor en todos los caminos de ejecución, decido mantenerla intencionalmente por razones de claridad semántica. Es un indicativo visual útil que resalta el propósito del if, especialmente cuando se trata de lógica de control relacionada con flujos de solicitudes o validaciones condicionales.

En ciertos contextos (como revisión de logs, debugging o trazabilidad mental del flujo), ser explícito vale más que ser minimalista. Por eso, prefiero mantener esta asignación aunque técnicamente sea innecesaria, para mejorar la comprensión del flujo por parte de futuros desarrolladores o del propio equipo.



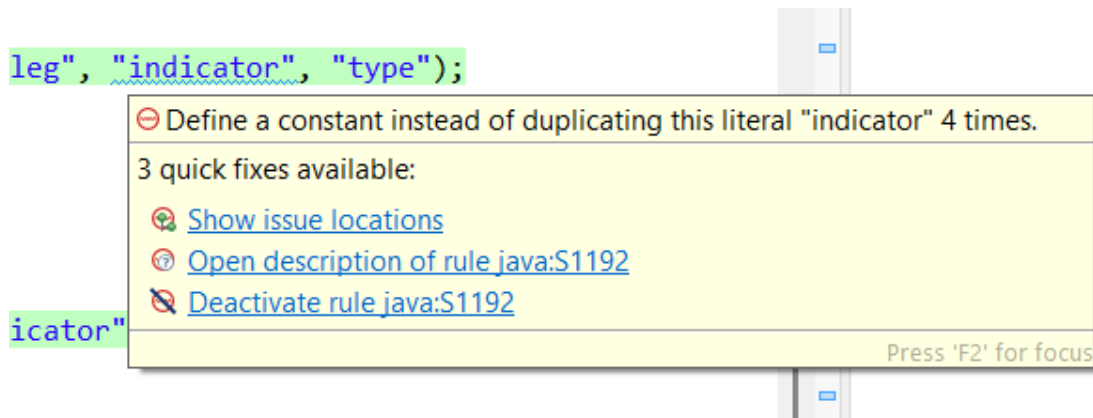
- **Define a constant instead of duplicating this literal.**

Este mensaje corresponde a la regla "Define a constant instead of duplicating this literal".

Aunque repetir literales como "indicator" varias veces puede considerarse un bad smell

por afectar la mantenibilidad, en este caso decido no extraerlo como constante, ya que se trata de una cadena simple y de uso claro, cuya duplicación no representa un riesgo ni complica el entendimiento del código.

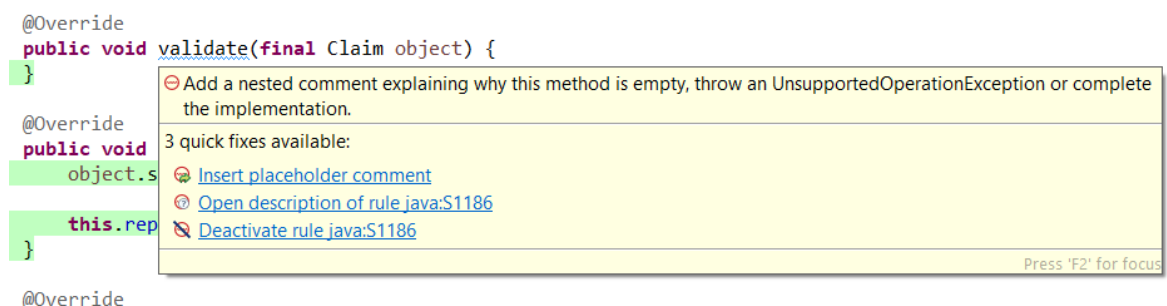
Por lo tanto, no considero necesario aplicar la recomendación en este contexto específico.



● **Add a nested comment explaining why this method is empty, throw an `UnsupportedOperationException` or complete the implementation.**

Este mensaje corresponde a la regla java: S1186, que indica que un método vacío puede generar confusión si no se justifica adecuadamente su estado. La ausencia de lógica en un método sobrescrito puede sugerir que la implementación está incompleta o fue olvidada accidentalmente.

En este caso particular, he decidido no lanzar una `UnsupportedOperationException` ni completar la implementación inmediatamente, ya que se trata de una sobrescritura obligatoria (por herencia o interfaz) que aún está en proceso de diseño o delegación futura.



### **3. Conclusión**

En resumen, aunque existan algunos "bad smells" en el código, podemos concluir que este código no es de baja calidad y funciona correctamente. Es fácil de entender y leer, lo que es fundamental para su mantenimiento y evolución.

### **4. Bibliografía**

No hay bibliografía presente para esta entrega.