INFORME DE LINT

GRUPO: E3.07

URL: https://github.com/juacasalb/Acme-One

Juan Castro Albertos (juacasalb@alum.us.es)
Francisco Javier de la Prada Prados (fraprapra1@alum.us.es)
Miguel Gaviro Martínez (miggavmar@alum.us.es)
Álvaro Gómez Nieto (alvgomnie@alum.us.es)

Historial de versiones

Fecha	Versión	Descripción de los cambios	Sprint
25/04/2022	1.0	Creación de los documentos para el "Deliverable 3" y extensión del proyecto con las clases Java, integradas como funcionalidades para obtener datos a partir de los ficheros CSV, expresándose en vistas y comprobados con diversos *tests*.	3
23/05/2022	2.0	Creación de los documentos para el "Deliverable 4" y extensión del proyecto con las clases Java, integradas como funcionalidades para obtener, crear, modificar y eliminar datos dentro del sistema y de los ficheros CSV, expresándose cada funcionalidad mediante vistas y comprobados con diversos *tests*.	4

Índice

Introducción

En este documento comprobaremos los “bad smells” de nuestro proyecto informados por SonarLint. En los casos que Lint esté informando de un “bad smell” inocuo, se proporcionará una justificación clara.

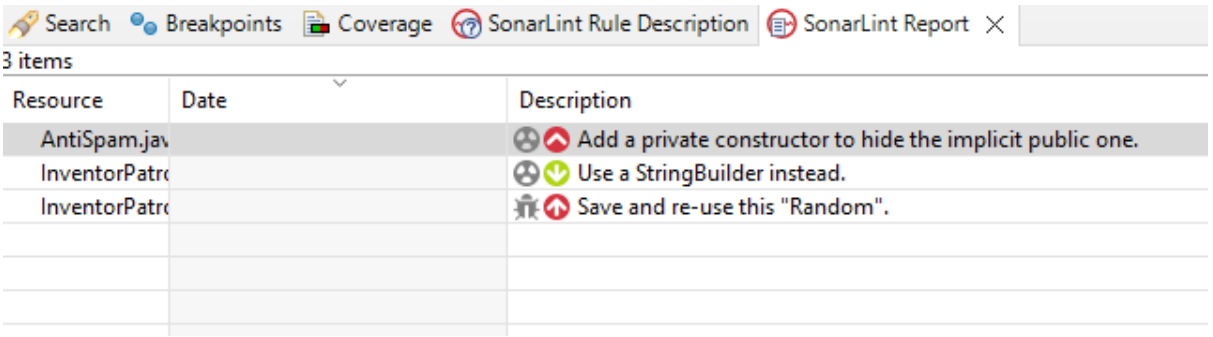
Resumen ejecutivo

La realización de este documento es importante, ya que así todo el equipo de trabajo puede conocer el análisis de la métrica de proyecto que se está llevando.

Este informe se ha realizado gracias a SonarLint, extensión en Eclipse IDE, el cual identifica y ayuda a solucionar los problemas de calidad y seguridad mientras escribes el código.

Contenido

Este sería el resultado del análisis de SonarLint:



Las dos primeras son recomendaciones y serían "bad smells" inocuos, ya que no consideramos que sean válidas para nuestro código.

Sin embargo, la última es "critical" y se refiere a esta parte del código:

```
public String generateAutomaticSeqNum() {

    int lastNum;
    String res = "";

    String finalSeq = ":";
    final Integer allPR = this.repository.findAllPat
    final Integer allPRSize = String.valueOf(allPR).

    final Random r = new Random();
    lastNum = r.nextInt(26);
    final char letter = (char)(lastNum+65);

    switch(allPRSize) {
        case 1: finalSeq += ("000"+allPR); break;
        case 2: finalSeq += ("00"+allPR); break;
        case 3: finalSeq += ("0"+allPR); break;
        default: finalSeq += (allPR); break;
    }

    for(int i=0; i<3; i++) {
        res+=letter;
    }
```

En la que nos recomiendan que creemos esa variable Random fuera para guardarla y reusarla más veces, ya que así sería más eficiente al dar un valor al azar, para que así sea siempre el mismo. También cuenta con una excepción SonarLint: si no se vuelve a usar esa variable en otro lado se puede ignorar esta "regla". Por lo que optamos por ignorarla como las anteriores porque la variable solo la usamos una vez.

Conclusión

En general, obviando lo comentado anteriormente, concluimos que seguimos unas buenas prácticas al no presentar “bad smells” graves. Y gracias a este informe el equipo puede saber que el código está bien realizado en su totalidad.

Bibliografía

Intencionadamente en blanco.