PS17 – Informe de Calidad: Historia de usuario: "Buscar parada por texto"

Resumen

El presente documento recoge el análisis de calidad asociado a la historia de usuario #241948–"Buscar parada por texto", y la correspondiente descripción del proceso de corrección realizado.

Document ID:	PS17/00/2017-QR003-US242580-BuscarParadaPorTexto
Departamento:	Calidad y Auditoría, dentro del proyecto integrado
Tipo:	Análisis de calidad
Privacidad:	CONFIDENCIAL
Estado:	ENTREGABLE
Versión:	1.0.3
Fecha:	15/11/2017
Autores:	Solar Iglesias, Fernando
Revisores:	Vila Martínez, Javier; Fernández Cerezo, Elsa

HISTORIAL DE CAMBIOS

Versión	Fecha	Cambio	Responsable
1.0.0	15/11/2017	Creación del documento.	Solar Iglesias, Fernando
1.0.1	15/11/2017	Redacción del documento.	Solar Iglesias, Fernando
1.0.2	15/11/2017	Revisión ortográfica.	Cerezo Fernández, Elsa
1.0.3	15/11/2017	Revisión estructural y gramatical.	Martínez Vila, Javier

1. Introducción

Este documento contiene el análisis de calidad llevado a cabo para la historia de usuario **#241948 – Buscar parada por texto**.

2. Análisis de calidad

2.1. Análisis inicial

Haciendo uso de la herramienta *SonarCloud*, se han obtenido inicialmente los siguientes resultados en el análisis:

- Encontrados seis bugs, una vulnerabilidad, treinta code smells y un porcentaje de duplicidad del código de un 4,6%.
- La deuda técnica se ve ascendida a seis horas.
- Entrando en detalle, los problemas se dividen en dos *issues* críticos, doce de severidad mayor, quince de severidad menor y ocho informativos.

A continuación, se exponen las acciones correctivas llevadas a cabo para los errores encontrados. Aunque se ha priorizado la resolución de los bugs y la vulnerabilidad frente a los code smells (sin olvidar la importancia de la severidad de cada uno), también se han solventado otros code smells de diversa severidad.

2.2. Acciones correctivas

	Bugs y Vulnerabilities
Clase Línea	El bug de severidad mayor, marcado por un *NullPointerException* sin tratar, se ha solucionado comprobando que el objeto afectado no fuese *null*.
	El método *equals*, debe sobrescribirse el método *hashCode*. Este bug de severidad menor se ha solucionado sobrescribiendo el método *hashCode*.
	La necesidad de comprobar el tipo de objeto pasado como parámetro ha sido solucionado verificando el tipo de objeto en el propio método.
Clase parada	El bug de severidad mayor *NullPointerException* sin tratar se ha solucionado verificando que el objeto afectado no fuese *null*.
	El bug de severidad menor encontrado muestra que si se sobrescribe el método *equals*, debe sobrescribirse el método *hashCode*. Se ha tratado sobrescribiendo el método *hashCode*.
	El bug de severidad menor en el que se debe comprobar el tipo de objeto pasado como parámetro se ha solucionado verificando el tipo de objeto en el propio método.
Clase ParadasFragment	La variable *paradas* "fácilmente accesible" por no ser estática o privada supone una vulnerabilidad que se ha solucionado haciendo dicha variable privada en lugar de pública.

	Code smells
Clase Database	Se ha encontrado que se duplica, en varias ocasiones, un *String* con el mismo contenido, suponiendo un code smell de severidad crítica. Se ha solucionado declarando una única constante en la clase con el valor de texto que se usaba.
Clase LoadDataAsync	Existen comentarios que contienen código Java y no documentación, suponiendo un code smell de severidad mayor. Como no eran de utilidad, se ha solventado eliminándolos.
	Se ha encontrado una constante no estática que podría calificarse como tal, suponiendo un code smell de severidad menor. Se ha solucionado declarándola como estática.
Clase ParserJSON	Se ha encontrado en varias ocasiones duplicidad de bloques de código idénticos al leer los *arrays* de paradas y líneas, siendo un code smell de severidad mayor. Para solucionarlo, se ha creado un método de lectura genérica que pueda ser usado en varias aplicaciones.
Clase ListLineasFragment	Existen comentarios que no contienen documentación sino código Java, suponiendo un code smell de severidad mayor. Como no era de utilidad, se ha solventado eliminándolos.
Clase ListLineasAdapter	Se ha encontrado un code smell de código duplicado de severidad mayor que no ha sido solucionado debido al enorme acoplamiento y a la gran cantidad de tiempo que suponía en comparación con otros code smell de severidad mayor.
Otros	Se han visualizado code smells de severidad menor en *ListLineasPresenter*, *ListParadasPresenter* y *LoadDataAsync*: debido a que existen variables que no se utilizan. Para eliminarlo, se ha prescindido de ellas o generado métodos observadores.
	Al existir *imports* que no son utilizados realmente se han detectado code smells de severidad menor que se han eliminado para ser solventados.
	También se han encontrado code smells simplemente informativos que se han corregido en función del beneficio respecto al esfuerzo.

2.3. Resultados

Una vez aplicadas las acciones correctivas descritas en el apartado anterior, se ha conseguido disminuir la deuda técnica de seis a cuatro horas y tan solo quedan once code smells (diez informativos y uno de severidad mayor).

Gracias a estas mejoras, se ha obtenido, finalmente, una calificación de A en las tres medidas. No obstante, se sigue visualizando una marca de *failed* debido a un error asociado a la cobertura de código que no ha podido ser solventado al tener que mantener las características pedidas.

3. Sumario

En este documento se ha recogido el proceso de análisis de calidad y acciones llevadas a cabo para mejorar la calidad del código asociado a la historia de usuario *"Buscar parada por texto"*. Además, los resultados obtenidos tras la corrección presentan un balance positivo, habiéndose establecido la calificación máxima (A) en todos los elementos evaluables.