UmCarroJá: Análise e Teste de Software

Henrique Faria A82200

Departamento de Informática, Universidade do Minho

Resumo Neste trabalho propusemo-nos a Analizar e testar o software feito no ambito da disciplina de *Programação Orientada a Objetos*.
Este relatório encontra-se estruturado em 4 secções: *Qualidade do Código Fonte, Refactoring da Aplicação, Teste da Aplicação e Análise de Desempenho da Aplicação*.
Foram também utilizadas as seguintes ferramentas para realizar o trabalho:

Palavras-chave: Code Smells · Technical debt

1 Glossário

Code smells não são bugs e também não estão tecnicamente incorretos. No entanto estes indicam fraquezas no design de uma aplicação que podem comprometer-la quer diminuindo o progresso do desenvolvimento da mesma quer provocando bugs ou falhas no futuro. Maus code smells podem provocar resultados adversos aos que se pretendem na aplicação conhecidos como technical debt.
Technical debt é um conceito de software que reflete o custo adicional implicito de modificação de código futuro como consequência da utilização de uma solução limitada mas facil de implementar em vez de implementar uma um pouco mais trabalhosa mas que não demande refazer ou reimplementar código no futuro.

2 Qualidade do Código Fonte

2.1 SonarQube

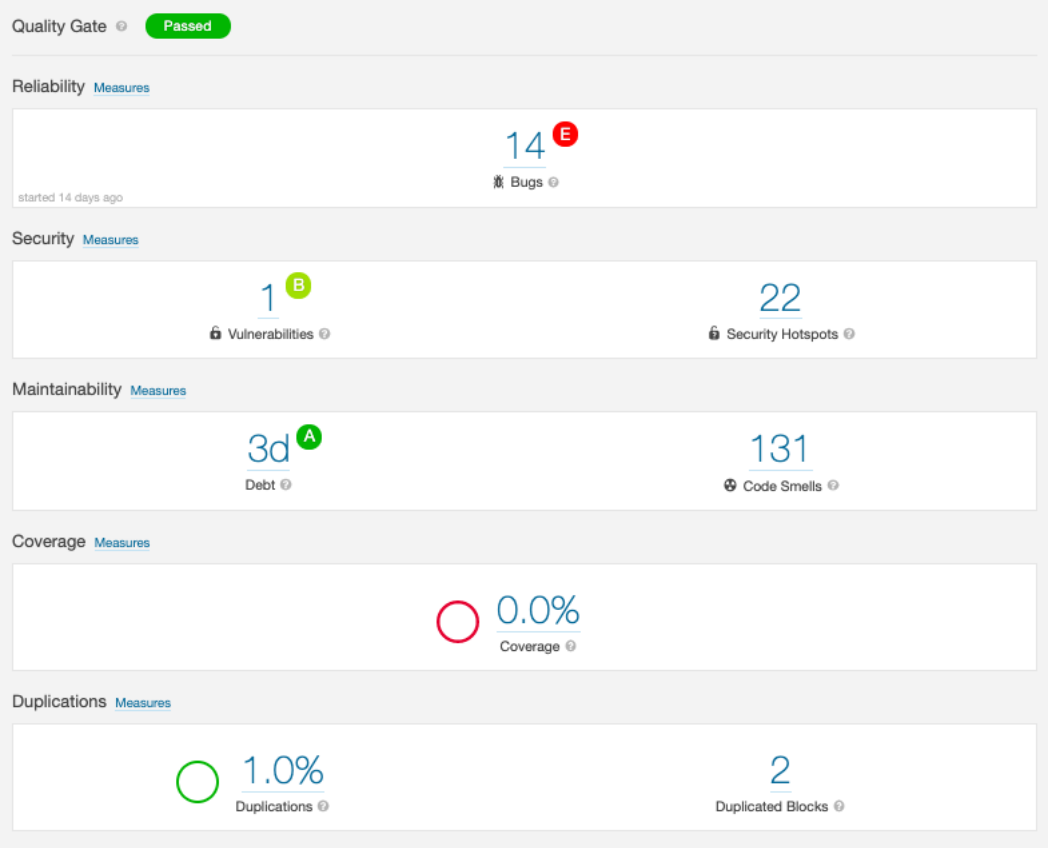

Figura 1. Menu geral de avaliação do SonarQube

Como se pode ver pela figura **??** este projeto possui alguns bugs, pelo menos 1 vulnerabilidade uma quantidade consideravel de code smells e 2 blocos duplicados.

De seguida apresenta-se um relatório detalhado dos tipos de erros e da sua gravidade:

2.2 Bugs

versão com bugs, sem bugs, com smells sem smells (por tipos de smells) -> descriminar o impacto dos smells

Blocker Bugs:

- Não usar blocos try/catch ao ler um ficheiro. resolução....
- Não usar blocos try/catch ao escrever em ficheiros.

Critical Bugs:

- Guardar e reutilizar variáveis random.

Major Bugs:

- Não obrigar a usar o método redifinido usando override.

  Para resolver este problema basta usar o segunite token: *@override* antes da função a qual estamos a redefinir.

Minor Bugs:

- Obrigar o override do equals e não do método hashCode().

2.3 Vulnerabilitys

- Utilizar printStackTrace() pode revelar informação sensivel.

2.4 CodeSmells

- Blocos de código repetidos.

3 Refactoring da Aplicação

4 Teste da Aplicação

5 Análise de Desempenho da Aplicação

Referências