

Manual de Qualidade e Entrega Contínua

Projeto: **Note Keeper**
Preparado por: Gonçalo Pinto – 65716
Data: 05-07-2017

Conteúdo do Relatório

Gestão e qualidade do código.....	2
Guia para os colaboradores (<i>coding style</i>).....	2
SCM <i>workflow</i>	2
Revisão de código	2
Análise estática.....	3

Gestão e qualidade do código

Guia para os colaboradores (*coding style*)

Foram usados 2 tipos de Coding, estes foram:

- Code Conventions for the Java Programming language (Oracle).
- Coding Style (Google).

SCM *workflow*

Para um maior controlo e segurança (backup) do trabalho de recurso, usei o GitHub (<https://github.com/goncalomix14/tqsRecurso.git>), o projecto é público, para ter conta privada é necessário pagar (ou registar com email da universidade, porém na altura que fiz o registo a opção para estudantes estava indisponível).

Para aceder ao projecto criado, deve-se usar o comando:

Git clone <https://github.com/goncalomix14/tqsRecurso.git>

Username: goncalomix14 ou goncalopinto@ua.pt

Password: acpa2014

Revisão de código

A revisão de código passou pelas diversas fases que foram aprendidas nas aulas.

Voltei a consultar os .pdf das aulas para entender melhor como realizar esta parte.

Como estou a trabalhar sozinho, não foi precisa tanta atenção ao controlo que era feito sobre o repositório (sei aquilo que faço push).

É de referir que na época normal utilizei o BitBucket, agora o GitHub e senti um conforto “diferente” com este último, porém tanto um como o outro utilizam-se bastante bem.

Análise estática

Foi usada a ferramenta SonarQube, foi instalada localmente no porto por defeito localhost:9000 e também foi integrada na ferramenta de Integração Contínua (Jenkins).

No Jenkins configurei Builds automáticos de 1 em 1 minuto, isto se houverem alterações no repositório apenas (senão teria uma infinidade de Builds).

Configurei o mesmo no porto localhost:8081, pois haviam processos em execução no porto por defeito (8080). Devo referir também que utilizei o Payara Server, este sofreu do mesmo problema e tive de alterar o porto para localhost:8083.

The screenshot shows the Jenkins web interface in a browser window. The address bar indicates the URL is localhost:8081. The Jenkins logo and navigation menu are visible on the left. The main content area displays a table of builds for the job 'tqsRecurso'. The table has columns for status (S, W), name, last success, last failure, and duration. A tooltip is shown for the 'W' (Warning) status, indicating that 0 tests failed out of 15, and build stability is good. The bottom of the page shows the footer with the date and time (00:28 07/07/2017) and the Jenkins version (2.60.1).

S	W	Name ↓	Último Sucesso	Última falha	Duração da Última
		tqsRecurso	4 min 23 seg - #29	4 h 10 min - #19	17 seg

Ícone: S M L

W Descrição %

- Test Result: 0 tests failing out of a total of 15 tests. 100
- Build stability: No recent builds failed. 100

Legenda: RSS para todos RSS só para falhas RSS só para últimas builds

Página gerada em: 7/jul/2017 0:28:31 BST REST API Jenkins ver. 2.60.1

00:28 07/07/2017

Com o SonarQube integrado no Jenkins foi-me possível nestes poucos dias meter o programa a passar, com tudo positivo.

Inicialmente havia um incómodo maior, apesar de haveram cerca de 60 Code Smells, na secção de Bugs apenas tinha 1, ou seja, foi a minha prioridade de resolver. Consegui resolver com sucesso este Bug.

O SonarQube indicou-me que o trabalho está bom, os últimos 5 builds foram todos com sucesso, o que permite assim ao SonarQube indicar-me o símbolo “sol” (as coisas vão no bom caminho).

