Ciências ULisboa | Informática

ANÁLISE E DESENHO DE SOFTWARE
PROJETO 2016/2017
PARTE II

O objetivo desta segunda parte do projeto é a implementação dos casos de uso **Inserir Empregado**, **Pedir Transferência** e **Processar Transferências** detalhados na primeira parte.

Para tal vão acrescentar o vosso código ao projeto disponibilizado. O código atual implementa os casos de uso **Processar Venda** e **Filtrar Vendas** realizados em iterações anteriores por uma outra equipa de desenvolvimento.

A implementação segue a estrutura típica por camadas:

- Na camada de domínio (pacote `business`) estão incluídas as classes que capturam o estado e comportamento dos conceitos analisados nos modelos de domínio e de desenho. Explorem os padrões lecionados na disciplina e usem aqueles que considerarem vantajosos para estruturar ou resolver certas tarefas no projeto. Justifiquem as vossas decisões no relatório.
- Na camada de persistência (pacote `dataaccess`) encontram classes que armazenam e fazem *queries* a tabelas de dados de uma base de dados relacional. No final da execução do programa, as mudanças efetuadas devem ser armazenadas nas respetivas tabelas. Usamos a biblioteca Java Derby que permite este tipo de funcionamento sem estarmos a instalar *software* de bases de dados. Devem ler as classes implementadas neste pacote para perceber como se executam comandos SQL. Leiam também os *scripts* SQL da pasta `data/scripts/`. Nestes scripts podem incluir informação para criar tabelas e inserir dados que precisam de existir mas para os quais ainda não temos casos de uso implementados (e.g., podem criar a tabela *Empregados* necessária para inserir empregados da empresa, bem como a tabela *Transferências*).
- Na camada dos casos de uso (pacote `use_cases`) encontram as classes que servem de *handlers*, i.e., representantes dos casos de uso que contêm as tarefas necessárias à execução desse caso de uso (cf. padrão GRASP *Controller*).
- No pacote `client` encontram um exemplo de execução. Devem criar uma ou mais classes que demonstrem as funcionalidades que implementaram.
- No pacote `dbutils` encontram classes que apagam o conteúdo e recriam as tabelas da base de dados com a informação original definida nos *scripts* SQL. Estas classes não precisam ser modificadas.

O código deve vir acompanhado por *javadocs,* contratos e comentários adequados.

Devem igualmente incluir no pacote `src/test/business/` uma classe JUnit que efetue testes para as classes relacionadas com o caso de uso **Inserir Empregado**.

A estrutura das classes e as suas relações devem estar de acordo com o modelo de desenho que tenham proposto. Se, porventura, durante a implementação houver a necessidade de alterar o modelo de desenho, ou outro artefacto, devem adicionar estes artefactos alterados no relatório justificando as alterações efetuadas.

Devem referir no relatório o conjunto de decisões mais importantes no que toca ao desenvolvimento e teste do vosso código.

Nota importante: quando importarem o projeto para o vosso Eclipse, façam-no via Maven para que tudo funcione corretamente! Quando o projeto estiver importado, a primeira coisa a fazer é executar a classe `CreateDatase` do pacote `dbutils`. A partir daí, a classe `SimpleClient` já pode ser executada.

Entrega

O relatório deve ter o mesmo tipo de formatação do relatório entregue na primeira parte. O projeto Eclipse deve ser compactado com a sua estrutura de pacotes intacta (não incluam os binários *.class*). Juntem toda a informação num único ficheiro zip com nome: `ads_grupo_XX.zip` (sendo `XX` o número do vosso grupo). Identifiquem-se no relatório e no código. O ficheiro zip deve ser entregue no moodle até às 23:00 do dia 18 de Maio de 2017.