Universidade Federal do Espírito santo – UFES

CENTRO UNIVERSITÁRIO NORTE DO ESPÍRITO SANTO - CEUNES

DEPARTAMENTO DE COMPUTAÇÃO E ELETRÔNICA

Fabiano Amaral Freitas

Elias de Castro Amorim Júnior

**documentação do sistema acadêmico da UFLN (universidade federal de lugar nenhum).**

São Mateus

2015

**INTRODUÇÃO**

Esse trabalho foi realizado pelos alunos Fabiano Amaral e Elias Junior na disciplina de Programação 3, o que essa documentação apresenta é uma linha de raciocínio adotada pelos membros do grupo para uma melhor solução do problema proposto pelo professor em sala. Foram usados conceitos de herança, composição, polimorfismo, interfaces, uso de arquivos txt, JSON, Design Pattern.

O Objetivo geral do trabalho é simular um sistema acadêmico universitário, onde existem várias entidades, como alunos, disciplinas, turmas, professores, atividades, dentre outras coisas modeladas dentro do POJO (**P**lain **O**ld **J**ava **O**bject) a seguir será abordado mais detalhadamente como cada pacote trabalha e sua organização interna.

**Model.pojo**

Esse pacote contém as seguintes classes: Aluno, Professor, Pessoa, Atividade, Nota, Falta, Disciplina, Turma. Dentro desse pacote nós temos classes que modelam entidades do sistema, com seus atributos e alguns poucos métodos relacionados a ela, como por exemplo o método *getNome()* dentro de aluno, que serve simplesmente para obter informações privadas da classe.

Inicialmente nós tínhamos atributos dentro da classe Aluno como por exemplo um ArrayList<Turma> que na hora de desenvolver o DAO nos gerou alguns problemas, por que quando o arquivo era carregado do TXT para construir um aluno eu precisava das turmas, só que cada turma tinha dentro dela uma lista de alunos, então eu precisava dos alunos já construir pra construir alunos, o que nos gerou um grande dilema para resolver esse problema, então nós decidimos não usar nenhum objeto dentro do outro no nosso projeto, substituímos os Array’s de objetos por Array’s de ID’s que fazem referência a algum objeto do tipo anterior. Isso será discutido melhor no próximo tópico que falará do DAO.

**Model.dao**

Um pacote somente com interfaces, possuindo as declarações de métodos que devem ser obrigatórios na implementação para que possa migrar de banco de dados sem que necessite fazer a modificação da view. A maneira que o método é implementado não possui nenhuma relação com essas interfaces. Só é necessário que possuam os mesmos argumentos e os mesmos retornos.

**Model.dao.txt**

Este pacote é a implementação das interfaces acima, havendo uma outra classe chamada DaoTxtUtils que foi desenvolvida para facilitar a interação com os arquivos, aproveitando-se do formato JSON para salvar as informações dos objetos no código.

Para essa finalidade, essa classe DaoTxtUtils recebe objetos e converte em JSON e cria um objeto à partir do JSON. Desta maneira não é necessário implementar em cada DAO, é somente utilizar, e como são métodos estáticos, não é necessário instanciá-los para a utilização.

Basicamente nos DAOs implementados é feito uma leitura inicial do arquivo, obtém cada linha e adiciona em um ArrayList, já com o objeto criado, pronto para a utilização em seus métodos, que assim que modificados utilizam um dos métodos que salva essa ArrayList em um arquivo novamente, convertendo todo objeto em JSON e colocando em cada linha desse arquivo.