

Programação Avançada

Trabalho Prático – Aplicação de Apoio ao Processo de Gestão de Projetos e Estágios.

Daniel Fernandes (LEI-PL) - a2020116565@isec.pt

Hugo Jorge (LEI-PL) - a2020116988@isec.pt

Decisões Tomadas

No trabalho utilizamos três padrões de desenvolvimento. O finite state machine, o padrão command e o padrão Singleton. O finite state machine usado para a progressão da aplicação e mudança de estados, o command para a atribuição e remoção de propostas a alunos e para atribuição e remoção e alteração de orientadores a propostas. O Singleton foi usado para a criação de um Log em que são colocadas mensagens de erro.

Na importação e exportação de dados do CSV optados pela criação de uma classe com métodos estáticos para leitura e escrita nos ficheiros.

Para detenção de erros optamos na importação de CSV por lançar as nossas próprias exceções com informação mais detalhada sobre os erros gerados.

Também optamos por exceções para interrupção do processo de atribuição automática de propostas a alunos quando este processo gera conflito.

Para armazenar se os estados estão fechados optamos por ter um map nos dados em que a chave é a enumeração dos estados da finite state machine e o valor é um booleano que permite que indica se essa mesma fase está fechada ou não. Optamos por apenas guardar o estado quando o utilizador pretender terminar o programa. Mas é sempre perguntado ao utilizador se pretende guardar o estado da aplicação.

Classes

Data.java

Classe onde é armazenado todos os alunos, docentes, propostas e candidaturas. Tem alguns métodos de obtenção e manipulação dos seus dados .

Aluno.java

Classe que representa um Aluno. O aluno tem uma candidatura e uma proposta que podem estar a null caso ainda não as tenha. São diferenciados pelo número de aluno e email.

Docente.java

Classe que representa um docente. O docente tem um email e nome.

Pessoa.java

Classe que representa uma pessoa e que é base de aluno e docente. Possui um email e nome e métodos de acesso a estas propriedades.

Projeto.java

Classe que representa um projeto a ser realizado por um aluno. Deriva da classe Proposta. Armazena o email do docente proponente.

Estagio.java

Classe que representa um estágio a ser realizado por um aluno. Deriva da classe Proposta.

Projeto\_Estagio.java

Classe que representa um estágio ou projeto autoproposto por um aluno. Deriva da classe Proposta.

Proposta.java

Classe que representa uma proposta. Esta pode ser um projeto, estágio ou projeto/estagio autoproposto por um aluno. Armazena o docente orientador e proponente, o id que é o identificador da proposta, o seu tipo de proposta, os ramos às quais se aplica, o título, o número de aluno caso exista e um booleano para saber se está atribuída a um aluno.

Candidatura.java

Classe que representa uma candidatura e que armazena o número de aluno e os identificadores das propostas a que o aluno se candidata.

Log.java

Representa um log de mensagens, usando o padrão Singleton.

ApoioException.java

É a classe que representa uma exceção lançada por nós enquanto programadores da aplicação. Todas as exceções criadas por nós derivam desta.

CollectionBaseException.java

É uma classe derivada de ApoioException. É usado com um armazenador de exceções da classe BaseException.

BaseException.java

É a classe base de exceções mais especificas que contem métodos comuns às exceções derivadas.

InvalidCSVField.java

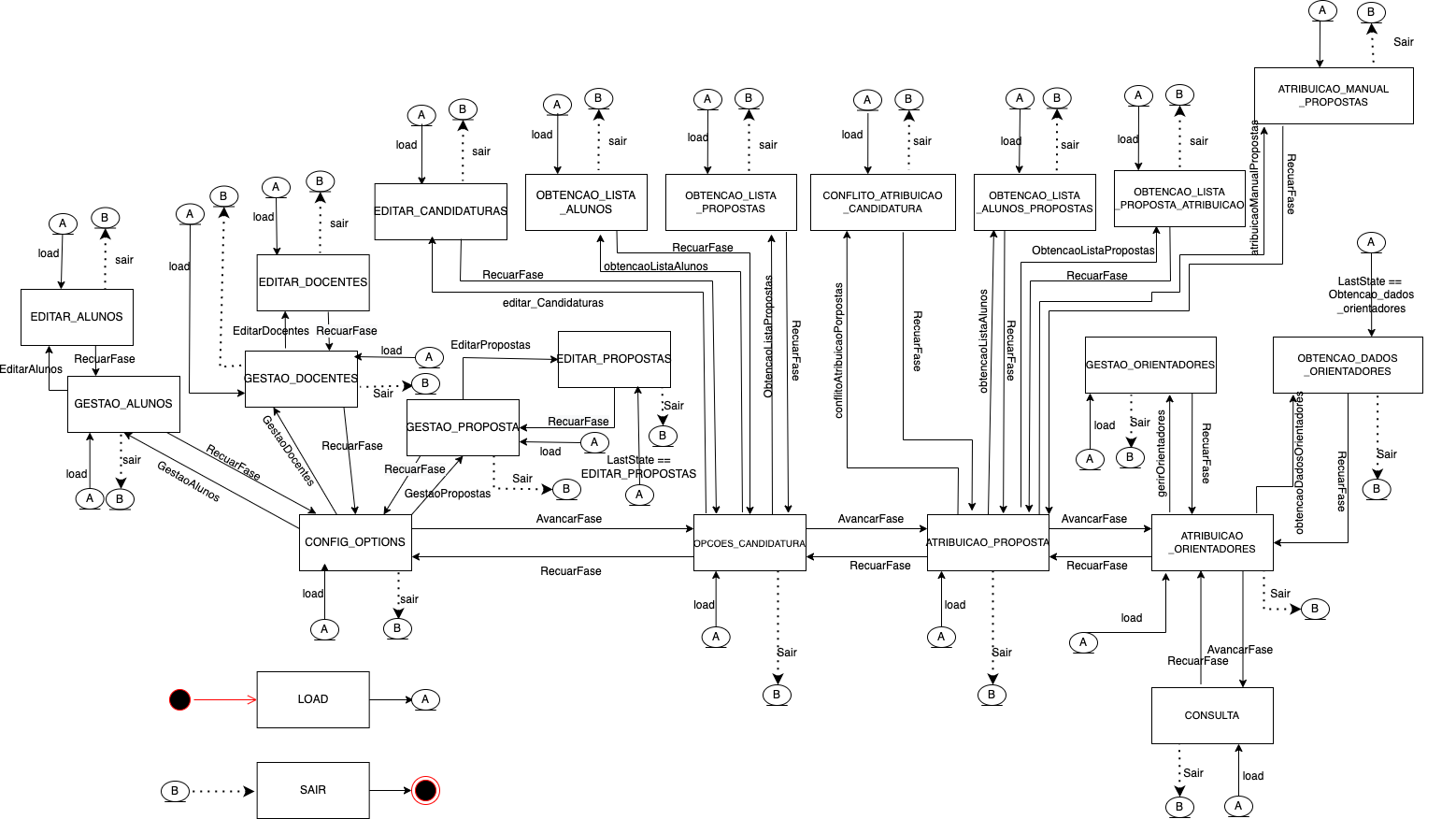
É uma classe derivada de BaseException que é lançada quando na importação de CSV existe um campo invalido.

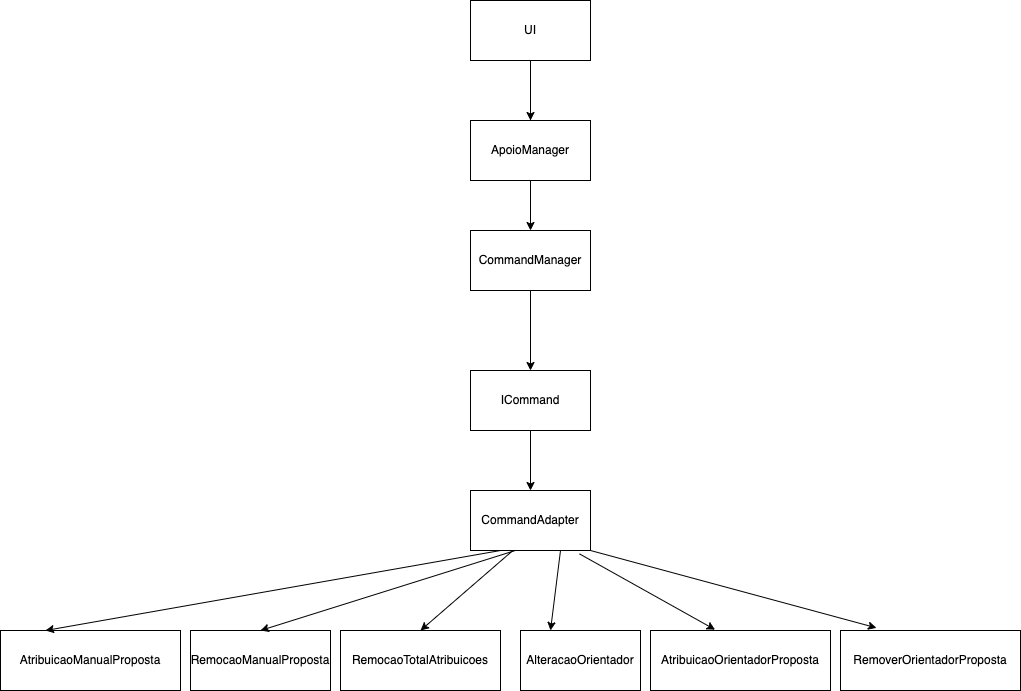
IncompleteCSVLine.java

É derivada de BaseException lançada quando uma linha de CSV está incompleta.

ConflitoAtribuicaoAutomatica.java

É derivada de ApoioException e é lançada quando existe um conflito na atribuição de proposta a alunos.





Uma imagem com texto

Descrição gerada automaticamente