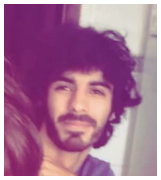

(a) André Gonçalves a80368



(b) Francisco Reinolds a82982

(c) Rafaela Rodrigues a80516

# Relatório do Projeto de Programação Orientada aos Objetos

Grupo 14

27 de Maio de 2018

# Conteúdo

# Capítulo 1

# Introdução

Este projeto, para a unidade curricular de Programação Orientada aos Objetos, consiste na criação de uma plataforma, idêntica ao Portal das Finanças, que permita a criação de contribuintes e empresas, a emissão de facturas, gestão de toda a plataforma e que permita guardar registro de todas as operações efetuadas, sendo possível, depois de sair da plataforma, as recuperar.
Neste programa é utilizada a linguagem Java.

# Capítulo 2

# Arquitetura de Classes

Neste projeto, foram implementadas várias classes, necessárias para o bom funcionamento da plataforma.
Iremos analisar cada uma delas em pormenor.

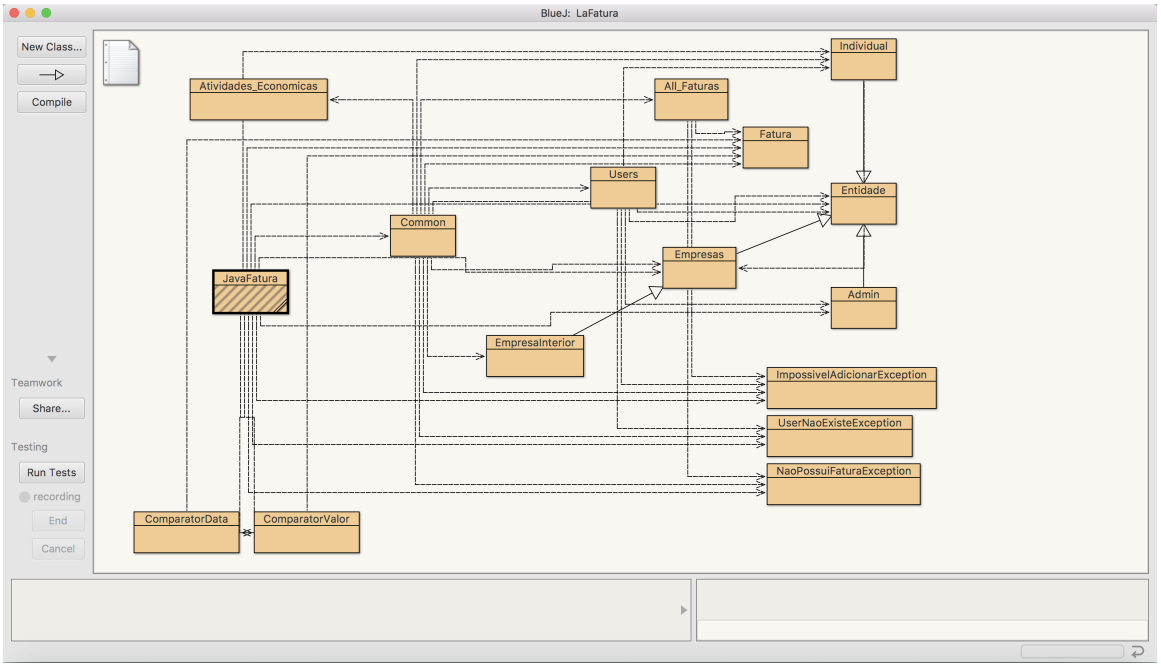



Figura 2.1: Classes

## 2.1 Classe Entidade

A classe Entidade é a super classe dos contribuintes.
Nesta classe são definidos todos os dados que os contribuintes individuais e os contribuintes empresariais possuem em comum.

### 2.1.1 Variáveis de instância

Nesta classe, apenas foram criadas as variáveis de instância obrigatórias.

#### NIF

O NIF, que representa o número fiscal, foi guardado como um int, sendo que apenas são representados por números positivos.

#### email

A variável email representa o email de contacto da entidade. É representada por uma String.

#### nome

O nome representa o nome da entidade. É representada por uma String.

#### morada

A morada representa a morada da entidade. É representada por uma String.

#### password

A password representa a password de acesso da entidade. Também é representada por uma String.

### 2.1.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets, sets, clone, toString (que imprime as variáveis de instância e o seu conteúdo).
Para além disso, foi criada um hash code, que apesar de não ser necessário para esta classe, será útil para outras classes, que possuem hash maps.
O hash code utiliza o valor das variáveis de instância para produzir um valor.

## 2.2 Classe Individual

A classe Individual é uma das sub classes da Entidade.
Nesta classe são definidos, para além dos que já foram definidos na Entidade, todos os dados que caracterizam um contribuinte individuais.

### 2.2.1 Variáveis de instância

Nesta classe, apenas foram criadas as variáveis de instância obrigatórias.

#### NIFagregado

O NIF agregado, são todos os Nifs de contribuintes individuais que pertencem ao agregado familiar do contribuinte em questão.
Os Nifs foram guardados num ArrayList (da interface List).
Para calcular o número de dependentes do agregado familiar, que era uma das informações obrigatórias, basta calcular o tamanho do array, permitindo não ter

de se criar mais uma variável de instância.

**coeficiente**

O coeficiente representa um fator multiplicativo que é associado a cada despesa elegível, sendo representado por um double.
O coeficiente deve ir de 0 a 1.

**codigos**

Os códigos representam todas as atividades económicas que os contribuintes individuais podem deduzir.
Os códigos são representados por uma String e são guardados num ArrayList(da interface List).

### 2.2.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets, sets, clone, toString (que imprime as variáveis de instância e o seu conteúdo).
Para além disso, foi criada um hash code, que complementa o hash code criado na super classe Entidade.
Este hash code também utiliza o valor das variáveis de instância para produzir um valor.

## 2.3 Classe Admin

A classe Empresas é uma das sub classes da Entidade.
Esta classe foi criada para gerir o sistema, sendo capaz de criar os contribuintes e ter acesso a dados do sistema.
Apenas o Admin pode criar novas Entidades.

### 2.3.1 Variáveis de instância

Nesta classe apenas foi adicionada uma variável de instância.

**ola**

Esta variável não representa nada. Apenas teve de ser criada pois como a classe Admin herdou toda a informação da classe Entidade, e não tinha sido adicionada mais nenhuma informação, não se conseguia distinguir se era Entidade ou uma Empresa.
A solução encontrada foi criar uma variável, que permitia distinguir.

### 2.3.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets, sets e clone.

## 2.4 Classe Empresas

A classe Empresas é uma das sub classes da Entidade.
Nesta classe são definidos, para além dos que já foram definidos na Entidade, todos os dados que caracterizam um contribuinte empresarial.

### 2.4.1 Variáveis de instância

Nesta classe, apenas foram criadas as variáveis de instância obrigatórias.

#### infoatividades

A infoatividades representa todas as atividades económicas em que determinada empresa actua.
Os códigos são representados por uma String e são guardados num ArrayList(da interface List).
Uma empresa pode ter mais de uma atividade económica, como por exemplo, um hospital universitário pode fornecer serviços de saúde e de educação.

#### infodeducaofiscal

O infodeducaofiscal representa um fator multiplicativo que é associado a cada despesa elegível, sendo representado por um double.
O coeficiente deve ir de 0 a 1.

### 2.4.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, desde gets, sets, clone, toString (que imprime as variáveis de instância e o seu conteúdo).
Para além disso, foi criada um hash code, que complementa o hash code criado na super classe Entidade.
Este hash code também utiliza o valor das variáveis de instância para produzir um valor.

## 2.5 Classe EmpresasInterior

A classe EmpresasInterior é uma das sub classes da Empresa.
Nesta classe são definidos, para além dos que já foram definidos na Empresa, todos os dados que caracterizam uma empresa do interior. As empresas do interior têm incentivos fiscais, dependendo do distrito.

### 2.5.1 Variáveis de instância

Nesta classe, apenas foi criada uma variável de instância.

#### distritos

Os distritos do interior são guardados num Map, onde a chave é o distrito em si (que está representado numa String) e o valor correspondente é o beneficio fiscal daquele distrito (que está representado num double).

### 2.5.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, desde gets, sets, clone, e toString (que imprime as variáveis de instância e o seu conteúdo).

## 2.6 Classe Atividades Economicas

Na classe Atividades Economicas são definidas, e guardadas, todas as atividades económicas disponíveis na plataforma.

### 2.6.1 Variáveis de instância

Nesta classe, apenas foi criada uma variável de instância.

#### atividades

As atividades são guardados num Map (mais propriamente um HashMap), onde a chave é a atividade em si (que está representado numa String) e o valor correspondente é a percentagem que permite deduzir (que está representado num double).

### 2.6.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, desde gets, sets, e toString (que imprime as variáveis de instância e o seu conteúdo).
Para além disso, foi criada um método private, que inicializa as atividades económicas pré-definifas, adicionando-as no HashMap.

## 2.7 Classe Fatura

A classe Fatura é a classe que permite a criação de faturas, com toda a informação necessária.
Neste projeto assumimos que apenas as empresas podem emitir faturas para os contribuintes individuais, e não para contribuintes empresariais.

### 2.7.1 Variáveis de instância

Nesta classe, foram criadas as variáveis de instância obrigatórias e mais duas complementar.

#### NIFemitente

Nif da empresa que está a emitir a fatura.
Como referido anteriormente, apenas as empresas podem emitir faturas.

**designacao**

A designacao é o nome da empresa.

**data**

A data é a data de criação da fatura.
Quando a empresa insere a data, esta tem de ser inserida no formato AAAA-MM-DD HH:MM .

**NIFcliente**

Nif do cliente, para o qual a fatura está a ser emitida.
Como referido anteriormente, apenas são emitidas faturas para contribuintes individuais.

**descricao**

Descrição da despesa.
É representada por uma String.

**atividade economica**

A atividade económica é a natureza da despesa, isto é, a atividade económica a que diz respeito.
Apenas pode ter uma atividade económica.
Como referido anteriormente, apenas as empresas podem emitir faturas.

**valor**

Valor da despesa em questão.
É representada por um double.

**catalogada**

Esta variável representa se a fatura está catalogada ou não. Uma fatura fica catalogada se a empresa que a emite apenas possui uma atividade económica onde atua. Caso possua mais de uma, esta fica pendente.

**historico**

Esta variável guarda todas as alterações que foram feitas nas atividades económicas, quando uma fatura está pendente.
As alterações são guardadas num ArrayList (da interface List).

### 2.7.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets, sets, clone, toString (que imprime as variáveis de instância e o seu conteúdo)e compareTo (que permite comparar faturas em relação a datas).
Para além disso, foi criada um hash code, que apesar de não ser necessário para esta classe, será útil para outras classes, que possuem hash maps e tree maps.

O hash code utiliza o valor das variáveis de instância para produzir um valor.
Possui também funções auxiliares, que se podem verificar no código.

## 2.8 Classe All Faturas

A classe All Fatura é a classe que permite criar uma estrutura para guardar todas as faturas .

### 2.8.1 Variáveis de instância

Nesta classe foram criadas duas variáveis de instância, uma para as empresas e outras para os individuais.
Dada uma fatura, essa fatura é guardada nas duas variáveis.

#### faturas clien

As faturas são guardados num Map (mais propriamente um HashMap), onde a chave é o nif do indivíduo para o qual a fatura foi emitida e o valor é um Set que possui todas as faturas relativas do cliente.

#### designacao

As faturas são guardados num Map (mais propriamente um HashMap), onde a chave é o nif da empresa que emitiu as faturas e o valor é um Set que possui todas as faturas relativas à empresa.

### 2.8.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets e sets.
Possui também funções auxiliares, que se podem verificar no código.

## 2.9 Classe Users

A classe Users a é a classe que permite criar uma estrutura para guardar todos os contribuintes, tanto individuais como empresas.

### 2.9.1 Variáveis de instância

Nesta classe foi criada uma variáveis de instância.

#### users

As faturas são guardados num Map (mais propriamente um HashMap), onde a chave é o nif do contribuinte (tanto individuais e empresas) e o valor é uma Entidade (que possui toda as informações do contribuinte).

### 2.9.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets e sets. Possui também funções auxiliares, que se podem verificar no código.

## 2.10 Classe de Exceptions

Foram definidas 3 classes que representam excepções necessárias para o bom funcionamento do programa.
Existem mais excepções no programa, mas não precisavam de ser definidas.

**ImpossivelAdicionarException**

Esta exception surge se tenta adicionar informações em Listas ou Maps e não se consegue.

**UserNaoExisteException**

Esta exception surge quando se tenta adicionar informações em Listas ou Maps e não se consegue.

**NaoPossuiFaturaException**

Esta exception surge quando se não existe nenhuma fatura associada a um certo nif.

## 2.11 Classe de Comparators

Foram definidas 2 classes de comparadores, que permitem ordenar estruturas ou determinar posições.

**ComparatorData**

Este comparador compara a data de duas faturas.
Retorna:

- negativo caso a data da primeira fatura seja menor que a segunda;
- zero se forem iguais;
- positivo se a data da primeira for maior que a da segunda

**ComparatorValor**

Este comparador compara o valor de duas faturas.
Retorna:

- -3 caso o valor da primeira fatura seja menor que a segunda;
- -1 se forem iguais;
- 3 se a data da primeira for maior que a da segunda

## 2.12 Classe Common

A classe Common é a classe que gere a relação entre o main e a implementação interna do sistema.

### 2.12.1 faturas

Esta variável guarda todas as faturas que são criadas no programa, de acordo com a implementação da classe All faturas.

#### entidade

Esta variável guarda a informação do contribuinte que está a utilizar o programa, de acordo com a implementação da classe Entidade.

#### atividade

Esta variável possui as atividades económicas e as suas informações, de acordo com a implementação da classe Atividades Economicas.

#### distritos

Esta variável possui os distritos e as suas informações, de acordo com a implementação da classe EmpresasInterior.

### 2.12.2 Métodos de instância

Nesta classe, foram criadas os métodos de instância básicos, como gets, sets, clone, toString (que imprime as variáveis de instância e o seu conteúdo) e todas as métodos de instância necessárias.
Porém, nesta classe também existem métodos de classe, que permitem trabalhar com toda a implementação interna, sem quebrar o encapsulamento.

## 2.13 Classe JavaFatura

A classe JavaFatura é a classe onde é implementado o main.
É através desta classe que se entra em contacto com o utilizador e se criam objetos.

# Capítulo 3

# Guia de Utilização

Iremos falar do guia de utilização da plantaforma.
Através da utilização de ficheiros binários, conseguimos guardar e posteriormente restaurar o estado do programa.

## 3.1 Inicializar

Ao inicializar-se o programa, é necessário escolher se pretende carregar um ficheiro já existente ou não.
Caso deseje carregar um ficheiro e este existir, é restaurado o estado do programa que ficou guardado nesse ficheiro.
Caso deseje criar um novo ficheiro, ou tentar carregar um ficheiro que não exista, o programa avança diretamente para o login.
Para carregar o ficheiro já populado, carregue o ficheiro estado

## 3.2 Login

No menu login, o utilizador tem a oportunidade de iniciar sessão ou de sair do programa.
Se sair do programa, o programa fecha e pede ao cliente para inserir o nome do ficheiro onde pretende guardar o estado da aplicação. Se desejar fazer login, é pedido o seu NIF e a sua password.
Apenas é possível iniciar sessão se o user existir e o NIF e password inseridas forem compatíveis com o registrado em memória.
Para fazer login como administrador, que cria entidade (entre outras coisas), o seu NIF é 1 e a sua password é admin.

Se o login for bem sucedido, e dependendo da entidade do utilizador que fez login, existem 3 opções:

- ir para o menu do Admin;
- ir para o menu das Empresas;
- ir para o menu dos Individuais;

Sempre que um utilizador termina sessão dentro do respetivo menu, é redirecionado para este menu, permitindo assim não ter de sair do programa para mudar de utilizador.

## 3.3 Menu do Admin

Caso o utilizador que tenha feito login seja o administrador, este é redirecionado para este menu.
Neste menu, existem várias opções:

1. criar uma nova entidade;
2. consultar tops;
3. terminar sessão;

### 3.3.1 Criar uma nova entidade

Caso a opção escolhida seja uma nova entidade, o admin pode criar um individual ou uma empresa.
Para a criação do **contribuinte individual** é necessário:

- NIF;
- nome;
- email;
- morada;
- password;
- coeficiente, que é necessário ser um número entre 0 e 1 (ex: 0.3);
- escolha da(s) atividade(s) economica(s) que pode deduzir (não existe número limite);
- inserção, ou não,de NIFs de agregados familiares. Apenas vão ser inseridos os NIFs que existam na plataforma. No momento em que se insere os NIFs do agregado, também é inserido o NIF da entidade que se está a criar na lista dos seus agregados.

Para a criação do **contribuinte empresarial** é necessário:

- NIF;
- nome;
- email;
- morada;
- password;
- a dedução fiscal, que é necessário ser um número entre 0 e 1 (ex: 0.3);

- escolha da(s) atividade(s) economica(s) em que atua (não existe número limite);
- escolha do distrito onde a empresa reside. Caso a empresa for do interior, terá uma bonificação na sua dedução fiscal.

Em ambos os casos, se for possível a criação, o utilizador volta para o menu do admin.
Caso contrário, pode optar por tentar novamente ou voltar para o menu.

### 3.3.2 Consultar tops

Escolhendo esta opção, o admin vai ter à sua disposição vários tops que permitem ter conhecimento dos dados da plataforma.

#### Top contribuintes que mais gastam

Este top apresenta, tendo em conta o valor gasto em todas as faturas existentes no programa, quais os NIFs que gastam mais em todo o sistema.
O top é apresentado em ordem crescente, sendo que o que gasta mais é o primeiro.

#### Top empresas que mais faturam

Este top apresenta, tendo em conta o valor gasto em todas as faturas existentes no programa, quais os NIFs das empresas que mais faturaram em todo o sistema.
É necessário indicar o tamanho do top que pretende ver. Caso o número de empresas seja menor, apresenta as que existem.
O top é apresentado em ordem crescente, sendo que o que gasta mais é o primeiro.

#### Top empresas que mais deduzem

Este top apresenta, tendo em conta o valor gasto em todas as faturas existentes no programa, quais os NIFs das empresas que mais deduzem em todo o sistema.
É necessário indicar o tamanho do top que pretende ver. Caso o número de empresas seja menor, apresenta as que existem.
O top é apresentado em ordem crescente, sendo que o que gasta mais é o primeiro.

#### Voltar para menu anterior

Volta para o menu do Admin.

### 3.3.3 Terminar sessão

Termina a sessão do utilizador e retorna para o menu do login.

## 3.4 Menu das Empresas

Caso a utilizador que tenha feito login seja uma empresa, esta é redirecionado para este menu.
Neste menu, existem várias opções:

1. criar faturas;
2. total faturado por data;
3. faturas emitidas;
4. faturas por cliente;
5. terminar sessão;

### 3.4.1 Criar faturas

Para a criação de uma **fatura** é necessário:

- Data de emissão da fatura, sendo que a data deve obrigatóriamente ter o formato yyyy-MM-dd HH:mm ;
- NIF do cliente;
- escolha da atividade económica a que a fatura se refere;
- descrição da compra;
- valor do gasto;

Se for possível a criação, o utilizador volta para o menu do admin.
Caso contrário, pode optar por tentar novamente ou voltar para o menu.

### 3.4.2 Total faturado por data

Este opção apresenta o total faturado pela empresa, entre duas datas.
É necessário o formato das datas ser yyyy-MM-dd HH:mm .

### 3.4.3 Faturas emitidas

Este opção apresenta o todas as faturas emitidas pela empresa.
As faturas podem ser visualizadas tanto ordenada por **valor** como por **data de criação**.
As faturas são apresentadas no valor por ordem decrescente, sendo que a que tem o maior valor aparece primeiro; nas datas, aparece a mais recente primeiro.

### 3.4.4 Faturas por cliente

Este opção apresenta o todas as faturas emitidas pela empresa para uma determinada entidade individual .
As faturas podem ser visualizadas tanto ordenada por **valor** como por um **intervalo de tempo**.
Se for num intervalo de tempo, é necessário o formato das datas ser yyyy-MM-dd HH:mm .

### 3.4.5 5-Terminar sessão

Termina a sessão do utilizador e retorna para o menu do login.

## 3.5 Menu dos Individuais

Caso a utilizador que tenha feito login seja uma individual, esta é redirecionado para este menu.
Neste menu, existem várias opções:

1. consultar perfil;
2. consultar faturas;
3. confirmar faturas;
4. dedução fiscal;
5. terminar sessão;

### 3.5.1 Consultar perfil

Esta opção permite ao utilizador visualizar o seu perfil, com todas as suas informações.
Esta opção é apenas de visualização.

### 3.5.2 Consultar faturas

Esta opção permite ao utilizador consultar todas as faturas passadas em seu nome.

### 3.5.3 Confirmar faturas

Esta opção permite ao utilizador aprovar todas as faturas que estão pendentes. As faturas pendentes são faturas que são emitidas por empresas que atuem em mais de uma atividade económica.
Se o utilizador pretender confirmar a fatura, passa para a fatura seguinte; caso contrário, o utilizador escolhe a atividade que considere correta e passa para a fatura seguinte.
É sempre guardado o histórico das alterações.

### 3.5.4 Dedução fiscal

Esta opção permite ao utilizador ter conhecimento de toda a dedução fiscal feita até ao momento por si e pelo seu agregado familiar.

### 3.5.5 Terminar sessão

Termina a sessão do utilizador e retorna para o menu do login.

# Capítulo 4

# Algoritmo de cálculo de dedução fiscais

O nosso algoritmo para a dedução escolhido foi

$$valor * (percentagem + info + bonus)$$

em que o **valor** é o valor da fatura, a **percentagem** é o valor de dedução da atividade económica da fatura em questão, o **info** é o valor da dedução da empresa e o **bónus** é, caso seja uma familia númerosa, ou seja, mais de 3 pessoas no agregado familiar, é adicionado 0.05 por cada pessoa no agregado.

# Capítulo 5

# Conclusão

Em conclusão, este projeto permitiu-nos consolidar os nossos conhecimentos em relação à linguagem Java e perceber as aplicações da linguagem.
A nível do projeto, sentimos que deveríamos melhorar na legibilidade do código, poderíamos ter impresso as deduções pelas atividades económicas a que dizem respeito, poderíamos ter usado outras estrutura de dados que fossem mais adequadas às necessidades.
Em geral, foi um bom projeto.