

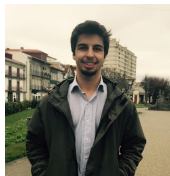
Universidade do Minho

Relatório - Programação Orientada a Objectos

MIEI - 2º ANO - 2º SEMESTRE
UNIVERSIDADE DO MINHO

IMOOBILIÁRIA

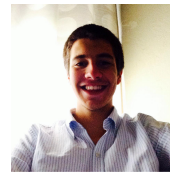
GRUPO 60



Dinis Peixoto
A75353



Ricardo Pereira
A74185



José Silva
A74576

19 de Maio de 2016

Conteúdo

1	Introdução	2
2	Descrição da aplicação	3
3	Arquitetura da aplicação	7
3.1	ImoobiliariaApp	7
3.1.1	Menu	8
3.2	Imoobiliaria	8
3.3	Utilizador	8
3.3.1	Comprador	9
3.3.2	Vendedor	9
3.4	Imóvel	9
3.4.1	Loja	10
3.4.2	LojaHabitável	10
3.4.3	Apartamento	11
3.4.4	Moradia	12
3.4.5	Terreno	12
3.4.6	Habitavel	13
3.5	Consulta	13
3.6	Leilao	13
3.6.1	Licitacao	14
4	Novos tipos de imóveis na aplicação	15
5	Conclusão	15

1. *Introdução*

Este projeto foi nos solicitado pelos docentes da UC *Programação Orientada a Objectos* e propunha a realização de uma aplicação capaz de fazer a gestão de imóveis, de maneira a que esta pudesse ser utilizada por uma agência imobiliária. Esta aplicação aceitaria dois tipos de atores: Vendedores e Compradores, sendo que os primeiros podiam controlar os imóveis à venda, assim como alterar o estado destes, aceder às consultas feitas aos seus imóveis e ter acesso a todos os imóveis à venda, os Compradores poderiam também fazer esta última, e se registados poderiam adicionar alguns destes aos seus favoritos para consultar no futuro.

A realização desta aplicação tinha como principal objectivo a aplicação de toda a matéria lecionada até ao momento na respectiva unidade curricular, com especial relevo para a *modularidade e encapsulamento de dados*, técnica de programação que não tínhamos abordado até então.

2. Descrição da aplicação

Esta é uma aplicação com uma interface para o utilizador muito simples, foi feita por nós de maneira que o utilizador pudesse tirar o maior proveito da mesma, com comandos simples, tendo em conta que todos os menus funcionam à base de opções por números.

Quando um utilizador executa a aplicação o primeiro menu a que está sujeito é o seguinte:

```
***** Menu *****
 1 - Registrar Utilizador
 2 - Iniciar sessão
 3 - Menu
 0 - Sair
*****
Opção: |
```

Neste menu inicial o utilizador pode registar-se na *Opção 1*, e caso já o tenha feito pode iniciar sessão na *Opção 2*. Caso o utilizador não pretenda registar-se pode aceder ao Menu na *Opção 3* onde terá acesso apenas a uma pequena parte das funcionalidades da aplicação. Se o utilizador decidir efetuar o registo será apresentado o seguinte menu, onde pode escolher se pretende registar-se como *Vendedor* ou *Comprador*:

```
***** Menu *****
 1 - Vendedor
 2 - Comprador
 0 - Sair
*****
Opção:
```

O utilizador necessitará de introduzir algumas informações suas para efetuar o registo, como no exemplo seguinte:

```
Nome: António José
Email: toninho@mail.com
Password: helloworld
Morada: Rua da Fonte
Data de nascimento: 26 de Janeiro de 1982
```

Posto isto, o utilizador pode agora iniciar sessão e desfrutar da aplicação que agora lhe apresentará mais funcionalidades do que quando não estava registado. Depois de efetuar o Login, isto é, iniciar sessão, este será o menu apresentado:

```
***** Menu *****
1 - Menu
2 - Fechar sessão
0 - Sair
*****
Opção:
```

Com a *Opção 2* o utilizador pode fechar sessão deixando assim de estar com a sua conta ligada.

Se o utilizador escolher a *Opção 1* o menu apresentado dependerá do tipo de utilizador que está a desfrutar da aplicação, se no caso for um *Comprador* será apresentado um menu com todas as funcionalidades da aplicação para um utilizador deste tipo:

```
***** Menu *****
1 - Lista de Imóveis de um dado tipo
2 - Lista de Imóveis habitáveis
3 - Todos os Imóveis e respectivos vendedores
4 - Marcar um Imóvel como favorito
5 - Consultar favoritos
0 - Sair
*****
Opção: |
```

- 1. Apresentar a lista de Imóveis de um dado tipo (dado pelo utilizador) até um determinado preço.
- 2. Apresentar a lista de todos Imóveis habitáveis até um determinado preço.
- 3. Apresentar a lista de todos os Imóveis e os seus respectivos vendedores.
- 4. Utilizando o ID de um Imóvel o utilizador pode marcá-lo como favorito, ficando assim guardado numa lista de Favoritos do utilizador.
- 5. Aprensetar a lista de todos os favoritos do utilizador em questão.
- 0. Sair do menu.

No entanto se o utilizador for do tipo *Vendedor* as funcionalidades apresentadas são um pouco diferentes:

```

***** Menu *****
1 - Colocar Imóvel à venda
2 - Consultas aos Imóveis
3 - Alterar estado de um Imóvel
4 - Imóveis mais consultados
5 - Lista de Imóveis de um dado tipo
6 - Lista de Imóveis habitáveis
7 - Todos os Imóveis e respectivos vendedores
8 - Leilões
0 - Sair
*****
Opção: |

```

- 1. Colocar um Imóvel à venda, introduzindo para tal todas as informações necessárias sobre o mesmo.
- 2. Apresentar a lista de todas as consultas aos imóveis dos quais este utilizador é responsável.
- 3. Alterar o estado de um dos imóveis existentes.
- 4. Tendo em conta as consultas feitas pelos utilizadores, verificar quais são os imóveis que têm mais que um determinado número de consultas.
- 5. Apresentar a lista de Imóveis de um dado tipo (dado pelo utilizador) até um determinado preço.
- 6. Apresentar a lista de todos Imóveis habitáveis até um determinado preço.
- 7. Apresentar a lista de todos os Imóveis e os seus respectivos vendedores.
- 8. Simular um leilão num determinado imóvel, durante um determinado tempo.
- 0. Sair do menu.

A nossa aplicação permite, apenas aos Vendedores, iniciar e simular um leilão num determinado imóvel já existente na Imobiliária e durante um determinado tempo (em segundos, unidade de tempo que nós achamos mais confortável para fazer a simulação).

Quando o leilão é iniciado começam as licitações por parte dos utilizadores, estas têm entre si um intervalo de tempo permitindo assim que o leilão seja o máximo organizado possível.

No momento em que o tempo de leilão se esgota são mostradas na aplicação as informações sobre o vencedor (exemplo na imagem seguinte), isto se o montante que se obteve ultrapassou o preço mínimo do Imóvel considerado.

```
Início do LEILÃO!
Licitador: utilizador_01 | Licitação: 300!
Licitador: utilizador_02 | Licitação: 500!
Licitador: utilizador_03 | Licitação: 600!
Licitador: utilizador_04 | Licitação: 1200!
Licitador: utilizador_05 | Licitação: 1300!
Licitador: utilizador_06 | Licitação: 1800!
Licitador: utilizador_07 | Licitação: 2000!
Licitador: utilizador_08 | Licitação: 2400!
Licitador: utilizador_09 | Licitação: 2700!
Licitador: utilizador_10 | Licitação: 2800!
Licitador: utilizador_04 | Licitação: 3400!
Licitador: utilizador_10 | Licitação: 3500!
Licitador: utilizador_02 | Licitação: 3700!
Licitador: utilizador_07 | Licitação: 3900!
Licitador: utilizador_04 | Licitação: 6400!
Licitador: utilizador_06 | Licitação: 6900!
Licitador: utilizador_10 | Licitação: 7000!
O vencedor do leilão é:
    Email: utilizador_10@mail.com
    Nome: utilizador_10
```

3. *Arquitetura da aplicação*

Neste capítulo falaremos ”esqueleto” da nossa aplicação, ao contrário do capítulo passado onde foi abordado o exterior da mesma. Falaremos sobre todas as classes presentes na aplicação, assim como os atributos e funcionamento de cada uma.

3.1 ImoobiliariaApp

Atributos

- Imoobiliaria imo
Imobiliária a correr na aplicação.
- Menu menu_principal
Menu principal.
- Menu menu_registo
Menu de registo.
- Menu menu_vendedor
Menu para vendedores.
- Menu menu_comprador
Menu para utilizadores não registados.
- Menu menu_comprador_registado
Menu para compradores registados.
- Menu menu_cria_imovel
Menu para criar um Imóvel.
- Menu menu_logado
Menu para utilizadores com sessão iniciada.
- Menu menu_leilao_vendedor
Menu para simulação de leilões.

Esta é a classe que trata de toda a interface apresentada a um utilizador, para que possa desfrutar ao máximo da nossa aplicação, como vimos no capítulo anterior a este. É também a classe responsável por gravar o estado da aplicação e ler esse mesmo de cada vez que a aplicação fechada e reiniciada, respectivamente.

3.1.1 Menu

Atributos

- List <String>opcoes
Lista de opções que o menu contém.
- int op
Operação escolhida pelo utilizador.

Esta classe é responsável por todos os menus criados e pelo funcionamento dos mesmos. Facilitando em muito o trabalho da classe principal, *ImoobiliariaApp*.

3.2 Imoobiliaria

Atributos

- Map<String,Imovel>imoveis
Lista dos imóveis disponíveis na Imobiliária.
- Map<String,Utilizador>utilizadores
Lista dos imóveis utilizadores na Imobiliária.
- Utilizador utilizador
Utilizador a utilizar a aplicação no momento (com sessão iniciada).
- Leilao leilao
Leilão a decorrer no momento.
- int id
#ID do próximo Imóvel a ser registado.

Esta é a classe responsável pelo funcionamento da Imobiliária, é nesta que estão desenvolvidas todas as funcionalidades da nossa aplicação, desde registar uma simples conta a simular um leilão com uma série de utilizadores.

3.3 Utilizador

Atributos

- String email
Email do utilizador.
- String nome
Nome do utilizador.

- String password
Password do utilizador.
- String morada
Morada do utilizador.
- String data_Nascimento
Data de nascimento do utilizador.

Classe abstrata para um utilizador, todos os diferentes tipos de utilizadores têm em comum o que se encontra desenvolvido nesta classe, tanto os atributos como os métodos.

3.3.1 Comprador

Atributos

- Map<String,Imovel>favoritos
Favoritos de um comprador.

Classe a que corresponde um Comprador, para além dos parâmetros herdados da classe anterior, os Compradores têm também uma Lista de Imóveis favoritos. Esta classe é constituída apenas por métodos básicos desde construtores, get's e set's ao comum toString.

3.3.2 Vendedor

Atributos

- Map<String,Imovel>portfolio
Portfólio dos Imóveis para venda do vendedor.
- Map<String,Imovel>vendidos
Portfólio dos Imóveis vendidos do vendedor.

Classe muito idêntica à Comprador, tal como esta herda os atributos da classe Utilizador aos quais são adicionadas duas listas, portfolio e vendidos, para o portfólio de Imóveis disponíveis e para o portfólio de Imóveis já vendidos, respectivamente. Tal como na Comprador os métodos são os mais comuns.

3.4 Imóvel

Atributos

- String id
ID do Imóvel.

- String rua
Rua onde se encontra o Imóvel.
- double preco
Preço do Imóvel.
- double preco_Minimo
Preço Mínimo do Imóvel.
- String estado
Estado do Imóvel ("em venda", "reservado" ou "vendido").
- List<Consulta>consultas
Lista de consultas ao Imóvel.

Classe abstrata, comum a todos os tipos de Imóveis admitidos pela aplicação. Nos métodos desta estão incluídos os métodos mais comuns: construtores, get's, set's, equals, toString e clone, sendo que este último é abstrato logo as classes que herdam esta têm o correspondente método diferenciando de classe para classe.

3.4.1 Loja

Atributos

- double area
Área da Loja.
- boolean wc
Existência ou não de WC na Loja.
- String tipo_Negocio
Tipo de Negócio a decorrer na Loja.
- int numero
Número da Loja.

Classe a que corresponde um Imóvel do tipo Loja, possui mais quatro atributos para além dos já herdados. Esta classe só tem os métodos mais comuns.

3.4.2 LojaHabitável

Atributos

- double area
Área da Loja.
- boolean wc
Existência ou não de WC na Loja.

- String tipo_Negocio
Tipo de Negócio a decorrer na Loja.
- int numero
Número da Loja.
- Apartamento apartamento
Apartamento correspondente.

Classe a que corresponde um Imóvel do tipo Loja Habitável, possui mais cinco atributos para além dos já herdados. Esta classe só tem os métodos mais comuns.

3.4.3 Apartamento

Atributos

- String tipo
Tipo do Apartamento.
- double area
Área do Apartamento.
- int quartos
Número de quartos do Apartamento.
- int casas_Banho
Número de casas de banho do Apartamento.
- int numero
Número do Apartamento.
- int andar
Andar do Apartamento.
- boolean garagem
Existência de garagem ou não no Apartamento.

Classe a que corresponde um Imóvel do tipo Apartamento, possui mais sete atributos para além dos já herdados. Esta classe só tem os métodos mais comuns.

3.4.4 Moradia

Atributos

- String tipo
Tipo de Moradia.
- double area
Área da Moradia.
- double area_Coberta
Área coberta da Moradia.
- double area_Terreno
Área do terreno da Moradia.
- int quartos
Número de quartos da Moradia.
- int casas_Banho
Número de casas de banho da Moradia.
- int numero
Número da Moradia.

Classe a que corresponde um Imóvel do tipo Moradia, possui mais sete atributos para além dos já herdados. Esta classe só tem os métodos mais comuns.

3.4.5 Terreno

Atributos

- int area
Área do Terreno.
- String tipo
Tipo de Terreno.
- float diametro_canalizacoes
Diâmetro das canalizações.
- float carga_eletrica
Capacidade de carga elétrica.
- boolean saneamento
Existência de saneamento.

Classe a que corresponde um Imóvel do tipo Terreno, possui mais cinco atributos para além dos já herdados. Esta classe só tem os métodos mais comuns.

3.4.6 Habitavel

Esta classe é uma interface sem qualquer variáveis de instância ou métodos.

3.5 Consulta

Atributos

- String email
Email do utilizador que fez a consulta.
- GregorianCalendar data
Data da consulta.

Classe a que corresponde uma consulta feita por um comprador a um determinado Imóvel disponível. Sempre que uma consulta é feita, esta é armazenada numa lista de consultas presente em cada Imóvel, com informação sobre o utilizador e a data e hora a que a mesma foi realizada. Esta classe só tem os métodos mais comuns.

3.6 Leilao

Atributos

- ArrayList<Licitacao>licitadores
Lista de licitaçõeslicitadores.
- Licitacao vencedora
Licitação vencedora até ao momento.
- Imovel imovel
Imóvel em causa no leilão.
- int horas
Número de horas que o leilão demora.
- int montante
Montante máximo leilado até ao momento.

Classe que controla um leilão, esta classe é responsável por controlar toda a simulação do leilão, desde iniciá-lo até ao encerramento do mesmo revelando, se existir, um vencedor. Esta classe é imutável e como tal não pode ter qualquer Set.

3.6.1 Licitacao

Atributos

- double minutos
Tempo entre cada licitação.
- double limite
Montante máximo que possível licitar.
- double incremento
Incremento a fazer entre cada licitação.
- long tempo
Tempo da última licitação.
- String licitador
Email do licitador.

Classe a que corresponde cada uma licitações durante um leilão. Esta classe só tem os métodos mais comuns.

4. *Novos tipos de imóveis na aplicação*

Toda esta aplicação foi desenvolvida por nós de maneira a que para qualquer eventual alteração desejada por parte da agência imobiliária fosse fácil de a aplicar, ainda que não sendo uma alteração insignificante. Como tal, sempre que a agência imobiliária desejar adicionar um novo tipo de Imóvel, só será necessário criar uma classe com todas as informações e respectivos métodos do Imóvel e adicioná-lo aos Menus a que os Vendedores têm acesso, para ser possível criar imóveis deste tipo.

5. *Conclusão*

Ao longo da realização deste trabalho fomos confrontados com algumas dificuldades, sendo que a maior foi interpretar o enunciado do projecto na parte relativa aos Leilões, no qual a informação não era suficiente, a nosso ver.

Tivemos também de ultrapassar outras barreiras para além desta, o que achamos perfeitamente normal tendo em conta ser a primeira vez em que todos os elementos do grupo utilizaram uma linguagem de programação para objectos para fazer um trabalho minimamente completo.

Todas estas dificuldades foram, no entanto, ultrapassadas tendo por fim um trabalho completo e para o qual podemos olhar com orgulho.