Universidade do Minho
Escola de Engenharia
Mestrado Integrado em Engenharia Informática

Programação Orientada a Objectos

Ano Letivo de 2015/2016

ImOObiliária

A27748 - Gustavo José Afonso Andrez

A74634 - Rogério Gomes Lopes Moreira

A67664 - Samuel Gonçalves Ferreira

21 Maio 2016

Índice

1. Introdução

O objetivo deste trabalho é, recorrendo à linguagem Java e ao IDE BlueJ, desenvolver uma aplicação que simula algumas das funcionalidades de uma imobiliária. Deverão ser desenvolvidas por isso funcionalidades para serem utilizadas, tanto por vendedores como compradores, na gestão dos imóveis. Assim, o programa deverá permitir registo de utilizadores e imóveis, e a consulta e venda dos imóveis já introduzida.

2. Descrição da arquitetura das classes

Nesta parte do relatório são descritas tanto as classes como os atributos desenvolvidos no projeto.

2.1. Classes

Dentro da classe Imobiliária existe um Map de objectos da classe Utilizador e um Map de objetos da classe Imóvel.

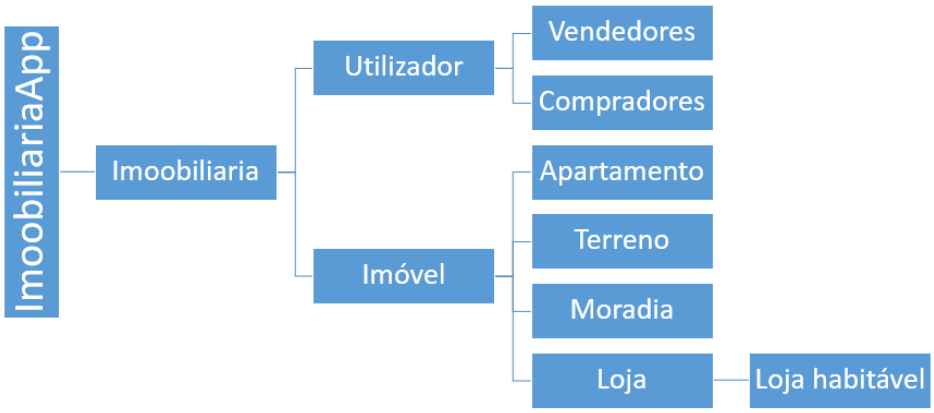
Além disso, existe um objecto da classe utilizador que tem por função indicar o utilizador que está logado (a usar aplicação) e um Boolean que indica a existência de um utilizador logado ou não.

A classe Imóvel tem 4 extensões que correspondem aos tipos de imóveis que se podem registar. Estas extensões possuem características próprias (diferentes de uma para as outras) e que, por isso estão definidas na classe correspondente. As características comuns estão definas na classe Imóvel. Para além disso, a classe loja tem por sua vez uma extensão que define uma loja com características habitacionais.

A classe Utilizador tem também duas extensões que correspondem aos dois tipos de utilizadores que a imobiliária pode ter: vendedores e compradores.

Temos ainda a classe ImoobiliariaApp onde está definido o objeto imobiliário e que é responsável por toda a interação com o terminal e pela gestão dos menus.

A imagem seguinte representa a relação descrita acima entre as diversas classes.



2.2. Atributos

Os atributos dos objetos da classe Imóvel são:

int consultas – número de vezes que o imóvel foi consultado;

String id – string única para cada imóvel criado e que o identifica de todos os outros;

String rua – morada do imóvel;

double preco – preço base do artigo;

double precoMinimo – valor abaixo do qual não é possível vender o imóvel;

Estado_Imovel estado - Estado atual do imóvel (tipo enum, ou seja, só pode ter um dos valores seguintes: para venda, vendido, reservado ou outro);

A classe **Apartamento** estende o Imóvel e implementa o habitável (ou seja um imóvel com características habitacionais). Assim o Apartamento possui os seguintes atributos próprios:

Tipo_Apartamento tipo – tipo de apartamento (tipo enum, só podendo ter 4 dos valores seguintes: Simples, Duplex, Triplex ou Outro);

double area – área habitável do apartamento;

int nQuartos – número de quartos;

int nWCs – número e quartos de banho;

boolean garagem – indicação se o imóvel possui garagem;

int nPorta – número da porta;

int andar – andar do imóvel;

A classe **Loja** estende o Imóvel mas não implementa o habitável por não possuir parte habitacional.

double area - área do imóvel;

boolean wc - indicação se o imóvel está dotado de quarto de banho;

String tipoNegocio – tipo de negócio para o qual a loja está preparada para implementar;

int nPorta – número da porta;

Contudo existe uma extensão da loja (loja com habitação) que, para além dos atributos de uma loja, é acrescido da parte habitacional equiparado a um apartamento.

A classe **Moradia** estende o Imóvel e implementa o habitável (ou seja um imóvel com características habitacionais). Assim a Moradia possui os seguintes atributos próprios:

Tipo_Moradia tipo - tipo de moradia (tipo enum, só podendo ter um dos seguintes valores: Isolada, Germinada, Banda, Gaveto ou Outro);

double areaImplantacao – área total do imóvel;

double areaCoberta – àrea coberta do imóvel;

double areaTerreno – área de terreno;

int nQuartos - número de quartos;

int nWCs - número e quartos de banho;

int nPorta - número da porta;

A classe **Terreno** estende o Imóvel mas não implementa o habitável por não possuir parte habitacional. Assim o Terreno possui os seguintes atributos próprios:

double areaConstrucao – indicação da área disponível para construção;

boolean habitação – indicação se o terreno tem licença para construção de habitação;

boolean armazem – indicação se o terreno tem licença para construção de armazém;

double diamCanalizacoes – indicação sobre o diâmetro das canalizações;

boolean eletricidade – indicação se existe ligação à rede elétrica;

double potenciaEletrica – indicação sobre a potência máxima suportada na rede;

boolean esgotos – indicação sobre existência de acesso à rede de esgotos;

Os atributos dos objetos da classe Utilizador são:

String email;

String nome;

String password;

String morada;

String data_nascimento;

A classe **Comprador** estende a classe Utilizador apresentando, para além das características do utilizador, um Map:

Map<String,Imovel> favorito - imóveis favoritos de cada comprador;

A classe **Vendedor** estende a classe Utilizador apresentando, para além das características do utilizador, dois sets:

Set<Imovel> portfolio – imóveis que o vendedor dispõe para venda;

Set<Imovel> historico – histórico dos imóveis já vendidos pelo vendedor;

3. Descrição da aplicação

A aplicação é toda controlada no ImoobiliariaApp. A navegação é realizada através de menus sendo o menu inicial o seguinte:

```
 *** MENU ***
1 - Login
2 - Registar Utilizador
0 - Sair
Opção:
```

Quando selecionado a opção 1 – Login – é solicitado ao utilizador que introduza o seu e-mail e password. Se as credencias não forem válidas é a apresentada a mensagem correspondente.

Se selecionada a opção 2 – Registar Utilizador – é apresentado novo menu:

```
 *** MENU ***
1 - Adicionar Vendedor
2 - Adicionar Comprador
0 - Sair
Opção:
```

Assim, o utilizador terá de se identificar como vendedor ou comprador, fornecendo os seguintes dados:

```
Email: xxx
Nome: xxx
Password: xxx
Morada: xxx
Data de Nascimento: xxx
```

Apenas é permitido aceder a funcionalidades a utilizadores logados.

Depois de registado, um comprador pode efetuar o login, ficando disponíveis as seguintes funcionalidades:

```
 *** MENU ***
1 - Marcar um Imóvel como Favorito
2 - Consultar os meus Imóveis Favoritos
3 - Consultar Imóveis por Tipo
4 - Consultar Imóveis Habitáveis
5 - Consultar Mapeamento de Imóveis
0 - Sair
Opção:
```

- 1 - Marcar um Imóvel como Favorito - permite ao comprador assinalar um imóvel para ser consultado posteriormente;
- 2 - Consultar os meus Imóveis Favoritos – permite a consulta dos imóveis assinalados como favoritos.
- 3 - Consultar Imóveis por Tipo - é apresentada uma listagem dos imóveis depois de indicado o tipo de imóvel;

- 4 - Consultar Imóveis Habitáveis - é apresentada uma listagem dos imóveis do tipo Apartamento, Loja com habitação e Moradia;
- 5 - Consultar Mapeamento de Imóveis - é apresentada uma lista dos imóveis com a informação do vendedor associado;

Depois de registado, um vendedor pode efetuar o login, ficando disponíveis as seguintes funcionalidades:

```
 *** MENU ***
1 - Registar Imóvel
2 - Top de imóveis mais consultados
3 - Alterar Estado de um Imóvel
4 - Consultar Imóveis por Tipo
5 - Consultar Imóveis Habitáveis
6 - Consultar Mapeamento de Imóveis
0 - Sair
Opção:
```

Selecionando a opção 1 – registar imóvel – é solicitado ao vendedor a seguinte informação:

```
 *** MENU ***
1 - Moradia
2 - Apartamento
3 - Loja
4 - Terreno
0 - Sair
Opção:
```

O vendedor terá de selecionar uma opção e posteriormente é solicitado ao utilizador a introdução dos atributos do imóvel que selecionou.

As restantes opções do menu do vendedor apresentam as seguintes funcionalidades:

- 2 - Top de imóveis mais consultados - é solicitado ao vendedor o número mínimo de consultas e, após introdução do valor, é apresentada a listagem dos imóveis, do vendedor logado, por ordem decrescente de consultas;
- 3 - Alterar Estado de um Imóvel - o vendedor pode alterar o estado dos seu imóveis, alternando entre os estados de: para venda, vendido, reservado ou outro;
- 4 - Consultar Imóveis por Tipo - é apresentada uma listagem dos imóveis depois de indicado o tipo de imóvel;
- 5 - Consultar Imóveis Habitáveis - é apresentada uma listagem dos imóveis do tipo Apartamento, Loja com habitação e Moradia;
- 6 - Consultar Mapeamento de Imóveis - é apresentada uma lista dos imóveis com a informação do vendedor associado;

4. Comentários

Para gerar objetos foi desenvolvido um gerador automático (diretoria gerador). Com recurso a este gerador foram criados 2000 utilizadores e 5000 imóveis. A aplicação, após criação dos objetos, continuou a comportar-se adequadamente.

A aplicação salvaguarda os dados (estado atual) no ficheiro estado.im sempre que o utilizador fecha a aplicação. Os dados do ficheiro são carregados quando o programa é inicializado.

Os id's dos imóves são gerados a partir de um hashCode e operações com bits dos dados gerais a todos os imóveis.

Para gerar este id são seguidos os passos:

- É gerado um hashCode a partir da instância rua (string);
- É feito um shift sobre o binário do preço (double);
- É feito um shift sobre o binário do preço mínimo (double);
- O id resulta da soma dos três valores obtidos nos passos anteriores.

O método getConsultas não foi terminado. Assim, o código está presente no projeto em comentário, não estando esta funcionalidade disponível.

Os testes automáticos disponibilizados com o enunciado não foram finalizados. No entanto o programa funciona adequadamente, sem apresentar erros quando exploradas as suas funcionalidades.

Para incluir novos tipos de imóveis na aplicação é necessário criar uma nova classe que estende os imóveis, ou seja, herda as suas variáveis, mas que tem as suas características próprias. Caso este novo tipo de imóvel seja habitável, vai também implementar a interface habitável.

Como conclusão, consideramos que o programa desenvolvido cumpre com quase a totalidade dos objetivos iniciais, funcionando corretamente.