Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Departamento de Engenharia Informática

Licenciatura Engenharia Informática

Programação Orientada aos Objetos 2022/2023

Projeto

StarThrive – Gestor de empresas

Marco Lucas Nº: 2021219146

Rui Coelho Nº: 2021235407

Índice

Introdução

Com o objetivo de planear e organizar a marca StarThrive criamos uma aplicação que permitisse a gestão das diversas empresas. Como tal, decidimos montar uma estrutura mais eficiente e que pudesse ser realmente útil. Inicialmente queríamos algo simples, como uma interface que tivesse algumas funcionalidades como adicionar empresas à nossa marca, listar os resultados das empresas ou algo tão simples como ter acesso às informações das empresas.

Organização de dados

De modo a facilitar a organização dos dados, organizamos a informação por arrayLists específicas para cada categoria, onde apenas existem informações de cada tipo. Isto dava-nos uma grande facilidade no acesso aos dados das empresas.

Vamos buscar informações sobre o ficheiro de objetos e atribuímos-lhe um arrayList. Caso o ficheiro de objetos não exista, ele vai buscar informações ao ficheiro de texto. Para um melhor acesso ao ficheiro de texto, a informação está dividida por ";", onde a primeira informação é a categoria.

Criamos dois packages, um que permite distinguir as classes que possuem os dados e as características de cada categoria de empresa, ao qual atribuímos o nome de classes e contém 10 classes e outro onde criamos e formatamos as interfaces gráficas, que chamamos de GUI e possui 9 classes. Fizemos esta divisão por uma questão de organização de classes.

Classes

Classe Abstrata Empresa

É caracterizada pela sua categoria (Restauração, Mercearia) (**categoria**), o seu nome (**nome**), o distrito (**distrito**) e por fim as coordenadas da sua localização (**coordenadas**).

- **Classe Abstrata Mercearia**

É uma classe que herda as características da Empresa ao qual adicionamos uma característica que é o custo de limpeza anual (**custoLimpezaAnual**).

- **Classe Mercado**

  Herda todas as características da classe Mercearia com três adições que são o tipo (Super, mini, etc…) (**tipo**), a àrea dos corredores (**areaCorredor**), e por fim a média de faturação anual por metro quadrado (**mediaFaturaçãoAnualM2**).

- **Classe Frutaria**

  Herda todas as características da classe Mercearia com duas adições que são o número de produtos (**numProdutos**) e a média de faturação anual por produto (**mediaFaturacaoAnualProduto**).

● Classe Abstrata Restauração

É uma classe que herda as características da Empresa ao qual adicionamos duas características que são o número de empregados (**numEmpregados**), o número médio de clientes por dia (**numMedioClientesDia**) e o seu salário médio anual (**salarioMedioAnual**).

- **Classe Abstrata Restaurante**

  Herda todas as características da classe Restauração com três adições que são o número de dias de funcionamento (**numDiasAberto**), o número de mesas interiores (**numMesasInterior**) e a média de Faturação diária por mesa (**mediaFaturacaoMesaDia**).

  - **Classe Local**

    Herda todas as características da classe Restauração com duas adições que são o número de mesas na esplanada (**numMesasEsplanada**) e o custo de licença da esplanada (**custoLicencaEsplanada**).

  - **Classe FastFood**

    Herda todas as características da classe Restauração com duas adições que são o número médio de clientes diários no Drive (**numMedioClientesDriveDia**) e a média da faturação diária no Drive (**mediaFaturacaoClientesDriveDia**).

- **Classe Café**

  Herda todas as características da classe Restauração com duas adições que são a média de cafés vendidos diariamente (**mediaVend**asCafesDia) e a média de faturação anual por café vendido por dia (**mediaFaturacaoAnualCafesDia**).

- **Classe Pastelaria**

Herda todas as características da classe Restauração com duas adições que são a média de bolos vendido por dia (**mediaVendasBolosDia**) e a média de faturação anual por bolo vendido por dia (**mediaFaturacaoAnualBolosDia**).

Criámos o nosso logo e aplicámos à aplicação.

Criamos uma classe chamada **StarThrive** que vai ser a nossa primeira frame que apresenta a lista de empresas que existem, juntamente com duas opções/Botões que podem **adicionar uma nova empresa** e **listar outros resultados**.

- Botão **adicionar nova empresa**

  Adicionar é também uma classe que é uma JFrame, que vai criar uma empresa nova, onde primeiramente precisamos de escolher a sua categoria numa "ComboBoxAction" (café, pastelaria, restaurante Fast Food ou restaurante Local) e mais a frente atribuímos as características da nossa nova empresa como o seu nome, a sua localização, etc…

Para obtermos mais detalhes das empresas basta darmos double-clique na empresa que desejamos na nossa lista inicial que levará a uma Jframe ao qual chamamos de **Details**:

- **createPanelInfo()** vai criar um painel com todas as informações destinadas à empresa, como o nome, a categoria, a despesa anual, a receita anual e o lucro (Sim/Não);
- Função **checkLucro(float lucro)**, verifica se houve lucro ou não;
- Função **settingButtons(),** que vai adicionar configurar e adicionar os botões “editar”, “apagar” e “voltar”:
  - O botão de **voltar** vai nos guiar à frame principal.
  - O botão **apagar** apaga a empresa da lista e deixa de existir voltando assim à primeira frame.
  - O botão **editar** permite modificar todas as atribuições da empresa como o nome, a localização, etc…

As **outras listagens** dão-nos informações como a listagem das empresas com maior receita anual, com a menor despesa anual, as empresas com maior lucro anual e por fim as empresas com maior capacidade de clientes. Para isso filtramos todos os resultados das nossas empresas que estão atribuídas na classe **Apresentar:**

- **getEmpresaMaiorLucroAnual(),** retorna um array que contém todas as empresas(uma de cada categoria) com maior lucro anual;

- **getArrayEmpresaMenorDespesaAnual(),** retorna um array que contém todas as empresas(uma de cada categoria) com menor despesa anual;
- **getArrayEmpresaMaiorReceitaAnual(),** retorna um array que contém todas as empresas(uma de cada categoria) com maior receita anual;
- **getArrayEmpresasMaiorCapacidadeClientes(),** retorna um array que contém todas as empresas(uma de cada categoria) com maior capacidade de clientes.

Posteriormente a função **Apresentar()** vai colocar num painel essa lista que vai ser apresentada consoante a opção que é feita pelo utilizador.

Na classe **Attributes** temos todos os métodos que permitem criar, filtrar e escolher os nossos painéis:

- **createPanelCafe()**, cria um painel que contém informação acerca de informação acerca da categoria cafés que é comum a todas as empresas (nome, categoria, distrito, coordenadas), média de cafés vendidos por dia e média anual de faturação de cafés vendidos por dia.
- **createPanelPastelaria()**, cria um painel que contém informação acerca de informação da categoria pastelaria que é comum a todas as empresas (nome, categoria, distrito, coordenadas), média de bolos vendidos por dia e média anual de faturação de bolos vendidos por dia.
- **createPanelLocal()**, cria um painel que contém informação acerca de informação da categoria restaurante local que é comum a todas as empresas (nome, categoria, distrito, coordenadas), número de mesas de esplanada e o custo de licença de esplanada.
- **createPanelFastFood()**, cria um painel que contém informação acerca de informação da categoria restaurante fast food que é comum a todas as empresas (nome, categoria, distrito, coordenadas), número médio de clientes Drive-Through e a média de faturação de clientes Drive- Through.
- **createPanelMercado()**, cria um painel que contém informação acerca de informação da categoria mercado que é comum a todas as empresas (nome, categoria, distrito, coordenadas), o tipo, área dos corredores e a faturação anual por metro quadrado.
- **createPanelFrutaria()**, cria um painel que contém informação acerca de informação da categoria frutaria que é comum a todas as empresas (nome, categoria, distrito, coordenadas), o número de produtos e a média de faturação anual por produto.

- **setAtribuites(String categoria)**, recebe uma string e compara com o tipo da categoria, se for igual atribui o seu respetivo painel.
- **settingEmpresaLabels()**, acrescenta toda a informação que é comum (nome, categoria, distrito e coordenadas).
- **settingRestaurauracaoLabels()**, **settingRestauranteLabels()** e **settingMercearíaLabels(),** acrescentam informações às respectivas categorias.
- **settingButtons(ArrayList<JButton> listaBotoes)**, recebe um array de botões que os adiciona ao painel.
- Possui ainda botões que permitem **voltar** e **registar**(no caso de uma nova empresa).

Ao utilizarmos estas funções temos um acesso mais facilitado para a criação dos painéis, o que torna o código mais versátil e compacto.

Conclusão

Com este projeto, conseguimos aprimorar as nossas técnicas de programação e evoluir um pouco naquilo que é a construção de código, a pesquisa e pensamento de como contornar e resolver problemas. A maior dificuldade pela qual passamos foi sem dúvida na gestão do tempo, dado que tivemos pouco tempo para aplicar aquilo que demos nas últimas semanas de aulas, nomeadamente a aplicação das interfaces.

Bibliografia

- ➔ https://docs.oracle.com/javase/7/docs/api/javax/swing/JFrame.html
- ➔ Slides de Apoio (dadas nas aulas práticas)