Serviço Público Federal
Ministério da Educação
Fundação Universidade Federal de Mato Grosso do Sul

Giovanna Rodrigues Mendes
Roney Felipe de Oliveira Miranda

Trabalho de POO utilizando interface gráfica

Java Swing

Campo Grande, MS
2022

Giovanna Rodrigues Mendes
Roney Felipe de Oliveira Miranda

Trabalho de POO utilizando interface gráfica

Java Swing

Relatório apresentado à disciplina de Linguagem de Programação Orientada a Objeto de maneira avaliativa ao curso de Ciência da Computação.

Professor: Dr. Samuel Benjoino Ferraz.

Campo Grande, MS
2022

Sumário

1. Descrição geral

O sistema desenvolvido nesse trabalho consiste em uma aplicação para cadastro de artistas, solistas, grupos, shows e empresas no universo do K-Pop. Dessa forma, segue abaixo o modelo conceitual do sistema:

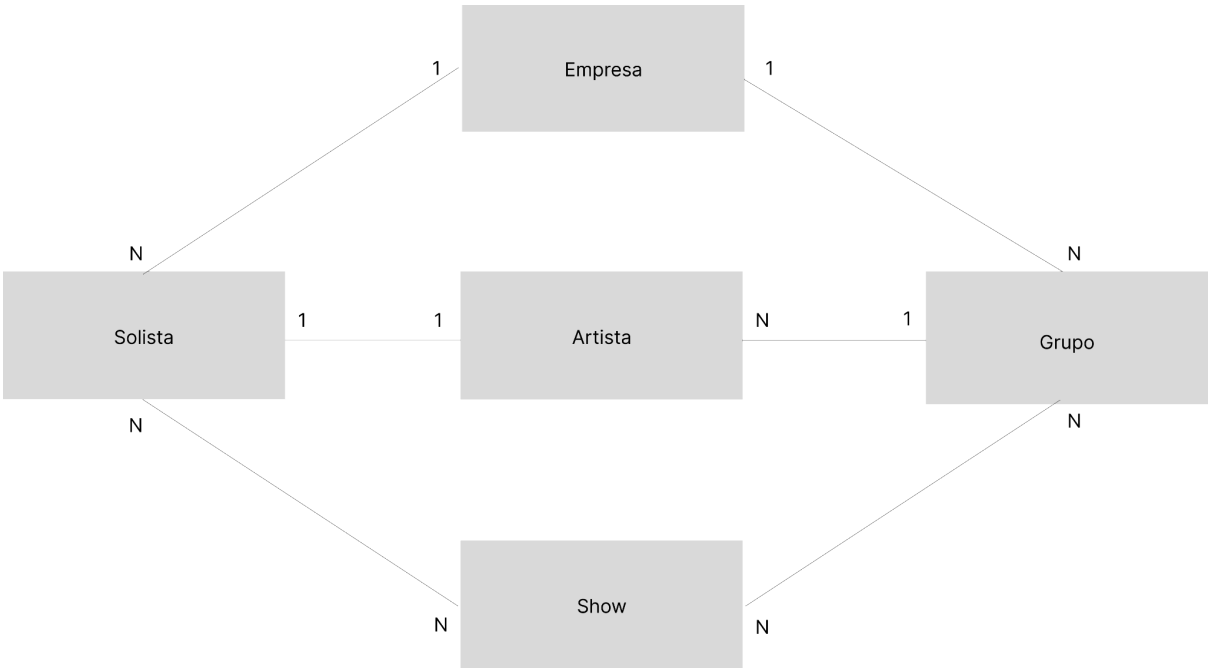

Figura 1: Modelo conceitual do sistema**.**

Fonte: Giovanna Mendes Rodrigues e Roney Felipe de Oliveira Miranda, 2022.

Diante disso, todas as entidades apresentadas na Figura 1 se tornaram classes concretas na codificação do trabalho, abaixo cada uma delas será descrita detalhadamente.

2. Classes

2.1 Classe Artista:

A classe Artista representa os artistas que podem ser inseridos no sistema.

- Atributos da classe:
  - **Nome do KIdol:** Tipo String. Inserido em input;
  - **Papel do KIdo:** Tipo String. Escolhido em um *JComboBox*, tendo as opções de Rapper, dançarino ou vocalista;
  - **Tem carreira:** Tipo *boolean*. Inicia em false, se tornando true quando o Artista segue carreira como solista ou quando integra algum grupo.

- Métodos da classe:
  - **Artista:** Construtor da classe, responsável por chamar o setNome, setPapel, setTemCarreira;
  - **setNome:** Seta o nome do artista;
  - **setPapel:** Seta o papel do artista;
  - **setTemCarreira:** Seta se o artista possui ou não carreira;
  - **getNome:** Retorna o nome do artista;
  - **getPapel:** Retorna o papel do artista;
  - **getTemCarreira**: Retorna se o artista possui ou não carreira.
  - **ToString:** Específica como o objeto será exibido para o usuário. Forma escolhida: **getNomeKidol()** + " seu estilo é " + **getPapel()** + "."

Todos os atributos são do tipo *private* e todos os métodos do tipo *public* pois todos os atributos são atualizáveis na tela de *update*, exceto o atributo **"TemCarreira"** porque ele só será setado quando o artista se tornar solista ou integrar um grupo de kpop.

2.2 Classe Solista:

A classe Solista representa os artistas do sistema que optaram em seguir carreira solo.

- Atributos da classe:
  - **Solista:** Tipo Artista. Escolhido em um *JComboBox* entre os artista já cadastrados no sistema;
  - **QuantidadeMiniAlbuns:** Tipo int. Representa a quantidade mini álbuns que um solista possui. Inserido via input;

- **QuantidadeDancarinos:** Tipo int. Representa a quantidade de dançarinos que um solista deve possuir em sua equipa, mínimo de 6 e máxima de 10.

- Métodos da classe:
  - **Solista:** Construtor da classe, responsável por chamar o setSolista, setQuantidadeMiniAlbuns e setQuantidadeDancarinos;
  - **setSolista:** Seta o artista que se tornou solista;
  - **setQuantidadeMiniAlbuns:** Seta a quantidade de mini álbuns de determinado solista;
  - **setQuantidadeDancarinos:** Seta a quantidade de dançarinos de determinado solista;
  - **getSolista:** Retorna o artista que se tornou solista;
  - **getQuantidadeMiniAlbuns:** Retorna a quantidade de mini álbuns de determinado solista;
  - **getQuantidadeDancarinos**: Retorna a quantidade de dançarinos de determinado solista;
  - **ToString:** Específica como o objeto será exibido para o usuário. Forma escolhida: **solista.getNomeKidol() + " possui " + getQuantidadeMiniAlbuns() + " mini-albuns e " + getQuantidadeDancarinos() + " dançarinos.";**

Todos os atributos são do tipo *private* e todos os métodos do tipo *public*, porém o atributo **solista** não pode ser alterado no *update*.

2.3 Classe Grupo:

A classe Grupo representa 4 artistas do sistema que optaram em seguir carreira juntos, formando um grupo.

- Atributos da classe:
  - **nomeGrupo:** Tipo String. Inserido via input;
  - **vetorArtistas:** Tipo Artistas[]. Representa os 4 artistas do grupo;
  - **shows:** Tipo ArrayList<Show>. Representa os shows que o grupo fará.

- Métodos da classe:
  - **Grupo:** Construtor da classe, responsável por chamar o setNomeGrupo e setVetorArtista.
  - **setNomeGrupo:** Seta o nome do grupo;
  - **setVetorArtista:** Seta os 4 artistas no grupo;
  - **getVetorArtistas:** Retorna o vetor contendo os 4 artistas;
  - **getShows:** Retorna os show que o grupo fará;

- **ToString:** Específica como o objeto será exibido para o usuário. Forma escolhida: **"O nome do grupo é " + nomeGrupo + " e seus integrantes são: " + vetorArtistas[0].getNomeKidol() + ", " + vetorArtistas[1].getNomeKidol() + ", " + vetorArtistas[2].getNomeKidol() + " e " + vetorArtistas[3].getNomeKidol() + ".";**

Todos os atributos são do tipo *private* e todos os métodos do tipo *public*, porém o atributo shows não pode ser alterado no *update* da classe Grupo.

2.4 Classe EmpresaDeKpop:

A classe EmpresaDeKpop representa a corporação proprietária dos direitos autorais tanto de grupos quanto de solistas dentro do sistema.

- Atributos da classe:
  - **nomeDono:** Tipo String. Inserido via input, deve possuir no mínimo 3 letras;
  - **nomeEmpresa:** Tipo String. Inserido via input, deve possuir no mínimo 2 letras;
  - **vetorSolista:** Tipo ArrayList<Solista>. Representa os solistas onde a empresa é proprietária dos direitos autorais.
  - **vetorGrupos:** Tipo ArrayList<Grupo>. Representa os grupos onde a empresa é proprietária dos direitos autorais.

- Métodos da classe:
  - **EmpresaDeKpop:** Construtor da classe, responsável por chamar o setNomeDono, setNomeEmpresa, setVetorGrupo e setVetorSolista.
  - **setNomeDono:** Seta o nome do dono da empresa;
  - **setVetorSolista:** Seta o vetor de solistas pertencentes a empresa;
  - **setVetorGrupo:** Seta o vetor de grupos pertencentes a empresa;
  - **getNomeDono:** Retorna o nome do dono da empresa;
  - **getNomeEmpresa:** Retorna o nome da empresa;
  - **getSolistas:** Retorna o ArrayList contendo os solistas que pertencem a empresa;
  - **getGrupos:** Retorna o ArrayList contendo os grupos que pertencem a empresa.
  - **ToString:** Específica como o objeto será exibido para o usuário. Forma escolhida: **"A empresa " + this.nomeEmpresa + " é gerenciada por " + this.nomeDono + ", possui " + vetorGrupos.size() + " grupos e " + vetorSolista.size() + " solistas associados a ela.";**

Todos os atributos são do tipo *private* e todos os métodos do tipo *public*, porém o atributo VetorSolistas e VetorGrupos não pode ser alterado no *update*.

2.5 Classe Show:

A classe Show representa as apresentações artísticas que tanto solistas quanto os grupos podem realizar..

- Atributos da classe:
    - **local:** Tipo String. Inserido via input;
    - **duracao:** Tipo int. Inserido via input, deve possuir no mínimo 1 hora e no máximo 6 horas de duração ;
    - **vetorObj:** Tipo Object. Representa tanto os solistas quanto os grupos que irão se apresentar em determinado show. Um show só pode comportar 3 apresentações artísticas.

- Métodos da classe:
    - **Show:** Construtor da classe, responsável por chamar o setLocal, setDuracao e o setVetorObj.
    - **setLocal:** Seta o local onde o show ocorrerá;
    - **setDuracao:** Seta a duração do show;
    - **setVetorObj:** Seta o vetor que contém os solistas e/ou grupos que irão se apresentar em determinado show;
    - **getLocal:** Retorna o local onde o show ocorrerá;
    - **getDuracao:** Retorna a duração do show;
    - **getVetorObj:** Retorna o vetor contendo os solistas e/ou grupos que irão performar no show.
    - **ToString:** Específica como o objeto será exibido para o usuário. Nessa classe, o método ToString tem várias possíveis saídas por causa da limitação de 3 participações e pelo fato do vetorObj armazenar tanto solista quanto grupo.

Todos os atributos são do tipo *private* e todos os métodos do tipo *public*, porém o atributo VetorSolistas e VetorGrupos não pode ser alterado no *update*.

2.6 Exceções do sistema

2.6.1 Classe de exceção InvalidDancerNumberException:

A classe InvalidDancerNumberException representa uma exceção que se estende de RuntimeException.

Essa exceção virá quando o número inserido de dançarinos (atributo de Solista) não for de 6 até 10, sendo sua mensagem: “Você inseriu um valor inválido para dançarinos. Por favor, insira de 6 até 10 para dançarinos”.

2.6.2 Classe de exceção InvalidDurationException:

A classe InvalidDurationException representa uma exceção que estende de RuntimeException.

Essa exceção virá quando o número inserido da duração de um show (atributo de Show) não for de acordo com o especificado, sendo sua mensagem: “Você inseriu um valor inválido para duração. Por favor, insira de 1 até 6 para duração de um show”.

2.6.3 Classe de exceção InvalidMiniAlbumException:

A classe InvalidMiniAlbumException representa uma exceção que se estende de RuntimeException.

Essa exceção virá quando o número inserido de mini-álbuns (atributo de Solista) é inferior ao anterior já setado, isto é, quando é chamado a TelaAtualizarSolista o valor de min-álbuns alterado espera-se ser maior que o atribuído inicialmente. Sendo assim, a mensagem da tela é: “A quantidade de mini-álbuns é menor que a anterior. Isso não pode ocorrer. Por favor, coloque um valor maior”.

2.6.4 Classe de exceção InvalidNameException:

A classe InvalidNameException representa uma exceção que estende de RuntimeException.

Essa exceção virá quando os nomes de dono e de empresa (atributos de EmpresaDeKpop) não estiverem de acordo com suas especificações, que são, respectivamente, ter mais de 2 letras e mais de 1 letra. Além disso, ela será lançada se o local inserido (atributo de Show) for menor que ou igual a 1 letra.

Desse modo, a mensagem associada a esses equívocos é: “A quantidade de letras é imprópria para ser considerada um nome”.

2.6.5 Classe de exceção KIdolHaveCarreiraException:

A classe KIdolHaveCarreiraException representa uma exceção que se estende de RuntimeException.

Essa exceção será lançada em duas telas de deletar (TelaDeletarArtista e TelaDeletarSolista), porque não se pode excluir um artista ou solista se ele possui uma carreira, ou seja, faz parte de um grupo ou faz carreira solo. O mesmo ocorre nas telas TelaCriarSolista e TelaCriarGrupo, pois não pode haver um mesmo artista associado às duas classes (Solista e Grupo).

Dessa forma, a mensagem associada é: “O artista já faz parte de um grupo ou já possui carreira solo. Dessa forma, sua agenda está lotadíssima e não tem tempo para uma nova carreira”.

2.6.6 Classe de exceção MissingInformationException:

A classe MissingInformationException representa uma exceção que se estende de Exception.

Essa exceção será lançada em todas as telas que implementam a criação, a deletação, a listagem e a atualização e representa os campos vazios (seja nos campos textuais em que precisam de um texto, seja nos checkboxes do qual se seleciona uma opção).

Assim, a mensagem associada é: “Nem todos os campos estão preenchidos”.

2.6.7 Classe de exceção NotANameException:

A classe NotANameException representa uma exceção que se estende de RuntimeException.

Essa exceção é lançada quando há números em locais onde só poderia haver letras, como é o caso do nome do dono de uma empresa, do nome da própria empresa (atributos de EmpresaDeKpop), como também é o caso do local (atributo de Show).

Sendo assim, a mensagem associada é: “Campo textual recebeu números”.

2.6.8 Classe de exceção SameGroupOrSoloException.java:

A classe SameGroupOrSoloException representa uma exceção que se estende de RuntimeException.

Essa exceção será lançada quando houver a tentativa de salvar o mesmo grupo ou solista duas vezes no mesmo show, ou seja, como se um grupo ou solista fosse se apresentar duas vezes no mesmo evento artístico. A mensagem exibida será: “O mesmo grupo/solista foi selecionado mais de uma vez para o mesmo show.” Dessa maneira, o show não é salvo.

2.6.9 Classe de exceção SameNameException.java:

A classe SameNameException representa uma exceção que se estende de RuntimeException.

Essa exceção será lançada quando houver a tentativa de criar um novo artista ou grupo com um novo que já existe em nosso sistema, sendo assim lançando a mensagem: “O nome dos grupos/K-idols devem ser únicos, ou seja, você está criando um grupo/k-idol com um nome já existente. Por favor, insira um nome diferente”.

Dessa maneira, um novo artista ou um novo grupo não é criado pois para ser criado ele deve possuir um nome único.

2.6.10 Classe de exceção KIdolWorkingException.java:

A classe KIdolWorkingException representa uma exceção que se estende de RuntimeException.

Essa exceção virá quando houver a tentativa de tornar um artista que já é solista ou que já faz parte de um grupo em um solista ou membro de outro grupo, sendo exibida a seguinte mensagem: “Esse KIdol já possui carreira, ou seja, faz parte de algum grupo ou é solista. Assim sendo, não vamos poder deletar seu artista por causa do hating que iríamos sofrer no Twitter! Para efetuar essa ação: Delete o solista ou grupo, antes de deletar o artista”.

Dessa forma, o artista não assume um nova função, permanecendo em seu estado inicial, além disso um novo solista não é criado assim como um novo grupo.

2.6.11 Classe de exceção KIdolNameEqualException.java:

A classe KIdolNameEqualException representa uma exceção que se estende de RuntimeException.

Essa exceção será lançada quando ao cadastrar um grupo o mesmo artista for inserido duas vezes, sendo exibida a mensagem: “Um grupo de Kpop deve possuir integrantes diferentes obrigatoriamente”. Desse modo, o grupo não é salvo pois para que

um grupo seja inserido no sistema ele precisa obrigatoriamente possuir 4 integrantes distintos, entre outros pontos.

2.6.12 Classe de exceção CorporationHaveKIdolsException.java:

A classe CorporationHaveKIdolsException representa uma exceção que se estende de RuntimeException.

Essa exceção será lançada quando houver a tentativa de deletar uma empresa que possui vínculo com dados solistas ou grupos, sendo exibida seguinte mensagem: “Essa empresa possui Grupos e/ou Solistas associados a ela e como não queremos deixá-los desempregados, você deve primeiro deletar o grupo e/ou os solistas associados a essa empresa”.

Dessa forma, a empresa não é removida do sistema pois para que um empresa seja deletada, tal não deve possuir vínculo com nenhum solista ou grupo.

2.6.13 Classe de exceção InvalidExclusionException.java:

A classe InvalidExclusionException representa uma exceção que se estende de RuntimeException.

Essa exceção será lançada quando houver a tentativa de apagar um solista/grupo que possui um vínculo com um show, sendo exibida a seguinte mensagem: “Essa exclusão não pode ser efetivada, pois esse grupo ou solista possui algum(ns) show(s) para fazer”.

3. Telas

Além das classes concretas Artista, Empresa, Solista, Grupo e Show, assim como as das exceções CorporationHaveKIdols, KIdolNameEqualException, KIdolWorkingException, SameNameException, SameGroupOrSoloException, NotANameException, MissingInformationException, KIdolHaveCarreiraException, InvalidNameException, InvalidMiniAlbumException, InvalidDurationException, InvalidDancerNumberException e InvalidExclusionException. Nessa aplicação, também existem as classes que representam as telas do sistema, onde ao todo somam 21 telas e implementam a interface janela, sendo elas:

Telas	Funções
TelaCriarArtista	Tela responsável por criar um novo artista.
TelaDeletarArista	Tela responsável por deletar um artista já cadastrado no sistema.
TelaAtualizarArtista	Tela responsável por atualizar um artista já cadastrado no sistema.
TelaListarArtista	Tela responsável por listar os artistas já cadastrados no sistema.
TelaCriarEmpresa	Tela responsável por criar uma nova Empresa.
TelaDeletarEmpresa	Tela responsável por deletar uma empresa já cadastrada no sistema.
TelaAtualizarEmpresa	Tela responsável por atualizar uma empresa já cadastrada no sistema.
TelaListarEmpresa	Tela responsável por listar as empresas já cadastradas no sistema.
TelaCriarGrupo	Tela responsável por criar um grupo artista.
TelaDeletarGrupo	Tela responsável por deletar um grupo já cadastrado no sistema.
TelaAtualizarGrupo	Tela responsável por atualizar um grupo já cadastrado no sistema.
TelaListarGrupo	Tela responsável por listar os grupos já cadastrados no sistema.
TelaCriarSolista	Tela responsável por criar um novo solista.
TelaDeletarSolista	Tela responsável por deletar um solista já

	cadastrado no sistema.
TelaAtualizarSolista	Tela responsável por atualizar um solista já cadastrado no sistema.
TelaListarSolista	Tela responsável por listar os solistas já cadastrados no sistema.
Tela Principal	Tela responsável por montar o status bar e o menu de navegação.