UNIVERSIDADE FEDERAL DO MARANHÃO

Bacharelado Interdisciplinar em Ciência e Tecnologia

Israel Barbosa Silva

Explicação - Library

São Luís

2023

Sumário

1 Resumo

O programa se trata de um CRUD de uma livraria, onde é possível cadastrar, alterar, listar e deletar livros e autores.

2 Classes e Interfaces

2.1 Author

Classe responsável por conter os atributos referentes ao autor, como id, nome, data de nascimento, nacionalidade e site.

A classe contém 3 construtores sendo 1 sem parâmetros, e 2 com parâmetros. O construtor que recebe id é usado em outra classe para atualizar um registro do autor, e o construtor que não recebe id, é usado para cadastrar um novo autor, já que nesse construtor o id é gerado automaticamente pelo UUID.

O UUID é um Identificador Único Universal que gera um número de 128 bits, representado por 32 dígitos hexadecimais e exibidos em cinco grupos separados por hífens, sendo um total de 36 caracteres, 32 alfanuméricos e 4 hifens.

No momento da geração do UUID eu converto ele para string usando o método toString() e pego apenas os 8 primeiro dígitos usando o método substring(), e atribuo ao id sempre que o construtor específico é chamado. Dessa forma todo Autor tem um identificador único que será utilizado para alterações e exclusões.

A classe Author ainda tem o método getAuthorInfo() que exibe todas as informações do autor no console, fazendo isso por meio dos métodos getters, os métodos setters não foram definidos pois a atribuição é feita pelos construtores.

2.2 Book

Classe responsável por conter os atributos referentes ao livro, como

id, titulo, autor, editora, isbn, número de páginas, e ano de lançamento.

A classe também contém 3 construtores, pela mesma lógica que já foi citada na explicação da classe Author, com a mesma implementação do UUID.

A classe também contém o método getBookInfo() que exibe as informações do livro no console, utilizando os métodos getters.

2.3 Ebook

Classe derivada da classe book, ela é uma especialização de um ebook, na classe temos a adição de um novo atributo, digitalVersion, que se refere a versão digital do livro.

Além dos construtores herdados da classe book, temos um construtor implementado que chama o construtor da superclasse, que precisa do id preenchido. No construtor foi adicionado o parâmetro digitalVersion.

Temos também o método getBookInfo() sobrescrito para atender a necessidade da especificidade da classe, chamamos o método da super classe e em seguida adicionando o atributo digitalVersion para ser exibido no console, desta forma temos todos os atributos da classe Book e da classe Ebook sendo exibidos.

2.4 PhysicalBook

Classe derivada da classe book, ela é uma especialização de um livro físico, na classe temos a adição de um novo atributo, stock, que se refere ao estoque físico do livro.

Além dos construtores herdados da classe book, temos um construtor implementado que chama o construtor da superclasse, que precisa do id preenchido. No construtor foi adicionado o parâmetro stock.

Temos também o método getBookInfo() sobrescrito para atender a necessidade da especificidade da classe, chamamos o método da super classe e em seguida adicionando o atributo stock para ser exibido no console, desta forma temos todos os atributos da classe Book e da classe PhysicalBook sendo exibidos.

2.5 IController

Interface implementada pelos Controllers, que serão responsáveis por cadastrar, listar, atualizar e deletar os registros de livros e autores.

A interface conta com 4 métodos:

1. register() ⇒ Onde a lógica para cadastrar um registro será implementada na classe.
2. showAll() ⇒ Onde a lógica para listar os registros será implementada na classe.
3. delete() ⇒ Onde a lógica para deletar um registro será implementada na classe.
4. update() ⇒ Onde a lógica para atualizar um registro será implementada na classe.

2.6 MenuController

Classe responsável por exibir textos para o usuário escolher o que deseja fazer no programa.

A classe contém dois campos privados, sendo duas instâncias das classes BookController e AuthorController, que serão utilizadas para chamada dos seus respectivos métodos de acordo com as opções escolhidas pelo usuário.

A classe contém 3 métodos: ShowMenu(), listBooksIsEmpty() e listAuthorsIsEmpty().

ShowMenu() é responsável por mostrar o menu em si, será exibido no console mensagens com as opções de 0 a 8 indicando o que cada opção faz. Com base na escolha do usuário, por meio de um switch é chamado o método correspondente, por exemplo, se o usuário quiser cadastrar um autor, a opção é 2, então o método irá utilizar a instância de AuthorController instanciada na classe, para chamar o método de cadastrar um novo autor que está implementado na classe AuthorController.

No switch antes de fazer a chamada do método correspondente, os métodos listBooksIsEmpty() e listAuthorsIsEmpty() são utilizados para verificar se a lista de autores ou de livros está vazia. Por exemplo, se o método showAll da classe BookController for chamado, o método listBooksIsEmpty() verifica se a lista de livros contém algum livro, se ela não

tiver nenhum, uma mensagem será exibida no console informando que não há registros salvos, o mesmo funciona para as operações com autor.

Na opção de cadastrar um novo livro existe uma lógica diferente, ao escolher a opção, o método listAuthorsIsEmpty() é chamado, isso se dá porque para criar um objeto livro é necessário que um autor esteja previamente cadastrado, então caso tente cadastrar um livro sem que um autor esteja cadastrado, não será possível.

2.7 BookController

Classe responsável por gerenciar a criação, alteração, listagem e exclusão dos livros, a classe implementa a interface IController e portanto além dos métodos da interface, contém os métodos createBook(), createEbook(String id), createPhysicalBook(String id) e getIndex(String id).

Começando pelo register(), é o método responsável por cadastrar um livro. O método pergunta ao usuário se o livro que será cadastrado será físico ou ebook e assim por meio de um switch são chamados os métodos createEbook(String id) ou createPhysicalBook(String id). Ambos os métodos usam o método createBook() que pede as informações comuns para um livro para o usuário, e em seguida dentro dos métodos createEbook(String id) ou createPhysicalBook(String id) é solicitado a informação específica, no caso do ebook a versão do livro, e no livro físico a sua quantidade em estoque e em seguida retorna o objeto montado para uma variável no método register() que em seguida adiciona o objeto a collection estática da classe.

Nos métodos createEbook(String id), createPhysicalBook(String id) temos os parâmetros de id, caso seja informado null no parâmetro, o método irá retornar um objeto com um id gerado automaticamente pelo UUID, este objeto será usado para inserir um novo registro, caso seja informado o id, o método retorna um objeto com o mesmo id informado com as alterações informadas pelo usuário, este objeto será usado para atualizar um registro já existente.

Esse foi o meio que encontrei para que quando o usuário alterasse as informações de um registro, o id permanecesse o mesmo.

O método showAll() é apenas um foreach que percorre a collection

de livros e utiliza o método getBooksInfo() para listar todos os livros que já foram cadastrados.

O método update() solicita o id do livro que será alterado, em seguida uso o método getIndex(String id) para capturar o índice do objeto na lista de livros e poder substituir pelo que será criado posteriormente.

Em seguida, percorro a lista de livros até encontrar um objeto que contenha o mesmo id que foi informado, caso ele não encontre uma mensagem será informada que não foi encontrado nenhum registro.

Caso seja encontrado é verificado se o objeto encontrado é uma instância de Ebook ou PhysicalBook utilizando o instanceof, com base no retorno é chamado o método createEbook(String id) ou createPhysicalBook(String id) com o id preenchido e a variável localBook recebe o retorno do método. Logo em seguida o método set() da collection é utilizado, passando o índice e o objeto como parâmetro para assim o objeto com os novos dados informados pelo usuário ser adicionado no lugar do objeto antigo de mesmo id.

O método delete() funciona parecido, com a diferença de que não é necessário identificar se o objeto é um Ebook ou um PhysicalBook, apenas o foreach procurando o objeto é executado e caso encontre o índice, o método remove() da collection é chamado, sendo passado como parâmetro, o índice do objeto a ser removido, caso não encontre, uma mensagem será exibida no console informando que não foi encontrado nenhum registro.

2.8 AuthorController

Classe responsável por gerenciar a criação, alteração, listagem e exclusão dos autores, a classe implementa a interface IController e portanto além dos métodos da interface, contém os métodos createAuthor(String id), getIdAndName() e getIndex(String id). O funcionamento da classe é parecido com a classe BookController.

Começando no método register() onde o mesmo chama o método createAuthor(String id), seguindo a mesma lógica de que quando chamado passando null no parâmetro, o método irá pedir as informações para o usuário e retornará um objeto com os dados informados pelo usuário e com id gerado automaticamente pelo UUID, e se for informado um id ele irá

retornar um objeto com o id informado.

O método update() funciona da mesma forma da classe BookController, o usuário irá informar um id e com base nesse id o método getIndex(String id) irá capturar o index do registro na collection de autor, irá chamar o método createAuthor(String id) passando o id informado pelo usuário e de acordo com as novas informações inseridas pelo usuário, fazer a substituição do novo objeto na collection.

Seguindo basicamente a mesma lógica, o método delete() solicita ao usuário um id para excluir, esse id é passado ao método getIndex(String id) que percorre a collection de autores para encontrar o objeto com base no id informado pelo usuário, caso encontre, o método remove() da collection é chamado, passando o índice do objeto para ser removido, caso não encontre o registro, uma mensagem é exibida no console informando.

O método showAll() como já explicado anteriormente percorre a collection usando o método getAuthorInfo() para exibir os dados de todos os autores cadastrados.

O método getIdAndName() faz o mesmo que o showAll() porém ao invés de exibir todos os dados, exibe apenas o nome e id, esse método é utilizado no momento do cadastro de um livro, para exibir ao usuário o id e nome dos autores, já que é preciso informar o id de um autor ao cadastrar um livro.

E por último o método getIndex(String) que utiliza um for percorrendo a collection até encontrar o índice do objeto com base no id passado como parâmetro, não seja encontrado nenhum registro o método retorna -1, e isso é utilizado para validação de que não foi encontrado nenhum registro com base no id informado.

2.9 Main

Por último a classe main, a classe responsável por dar início ao programa, contém apenas uma instância da classe MenuController, que por sua vez chama o método ShowMenu() que é responsável pelo gerenciamento do programa.