Wellington Cesar Fonseca

DESENVOLVIMENTO FULL STACK

MISSÃO PRÁTICA | NÍVEL 1 | MUNDO 3

RPG0014 - INICIANDO O CAMINHO PELO JAVA

OBJETIVO

1. Utilizar herança e polimorfismo na definição de entidades.
2. Utilizar persistência de objetos em arquivos binários.
3. Implementar uma interface cadastral em modo texto.
4. Utilizar o controle de exceções da plataforma Java.
5. No final do projeto, o aluno terá implementado um sistema cadastral em Java, utilizando os recursos da programação orientada a objetos e a persistência em arquivos binários.

2° PROCEDIMENTO | CRIAÇÃO DO CADASTRO EM MODO TEXTO

O que são elementos estáticos e qual o motivo para o método main adotar esse modificador?

- Elementos estáticos em Java são membros de uma classe que pertencem à própria classe, em vez de pertencerem a instâncias individuais dessa classe. Eles podem ser chamados diretamente na classe, sem a necessidade de criar um objeto dessa classe.
- Main é definido como estático para permitir sua execução direta pela JVM.

Para que serve a classe Scanner?

- A classe Scanner em Java é usada para ler e processar dados de entrada, geralmente provenientes do teclado (console).

Como o uso de classes de repositório impactou na organização do código?

- Positivamente, pois na hora de construir as funções, tive somente que chamar pela função que continha explicitamente o nome da ação que eu precisava.

CadastroPOO2.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this
template
 */
package cadastropoo;
import java.io.IOException;
import model.PessoaFisica;
import model.PessoaJuridica;
import model.PessoaFisicaRepo;
import model.PessoaJuridicaRepo;
import java.util.Scanner;
/**
 *
 * @author wellingtonfonseca
 */
public class CadastroPOO2 {

 /**
 * @param args the command line arguments
 */
 public static void main(String[] args) {

 try (Scanner scanner = new Scanner(System.in)) {
 PessoaFisicaRepo repoPessoaFisica = new PessoaFisicaRepo();
 PessoaJuridicaRepo repoPessoaJuridica = new PessoaJuridicaRepo();

 int tipo;
 do {
 System.out.println("Escolha o tipo:");
 System.out.println("1 - Pessoa Fisica");
 System.out.println("2 - Pessoa Juridica");
 System.out.println("0 - Finalizar execução");
 System.out.print("Escolha o tipo (1 ou 2) ou 0 e finalize: ");
 tipo = scanner.nextInt();
 scanner.nextLine();

 switch(tipo) {
 case 1 -> {
 System.out.println("Pessoa Fisica -> Selecione");
 System.out.println("1 - Incluir");
 System.out.println("2 - Alterar");
 System.out.println("3 - Excluir");
 System.out.println("4 - Buscar pelo ID");
 System.out.println("5 - Buscar todos");
 System.out.println("6 - Persistir dados");
 System.out.println("7 - Recuperar dados");
 System.out.println("0 - Voltar");
 int opcao = scanner.nextInt();

 switch (opcao) {
 case 1 -> incluirPessoaFisica(scanner, repoPessoaFisica);
 case 2 -> alterarPessoaFisica(scanner, repoPessoaFisica);
 case 3 -> excluirPessoaFisica(scanner, repoPessoaFisica);
 case 4 -> buscarPessoaFisica(scanner, repoPessoaFisica);
 case 5 -> buscarTodasPessoasFisicas(repoPessoaFisica);
 case 6 -> persistirPessoasFisicas(scanner, repoPessoaFisica);
 case 7 -> recuperarPessoasFisicas(scanner, repoPessoaFisica);
 case 0 -> System.out.println("Voltando...");
 default -> System.out.println("Tente novamente.");
 }

 }
 case 2 -> {
 System.out.println("Pessoa Juridica -> Selecione");
 System.out.println("1 - Incluir");
 System.out.println("2 - Alterar");
 System.out.println("3 - Excluir");
 System.out.println("4 - Buscar pelo ID");
 System.out.println("5 - Buscar todos");
 System.out.println("6 - Persistir dados");
 System.out.println("7 - Recuperar dados");
 System.out.println("0 - Voltar");
 int opcao = scanner.nextInt();

 switch (opcao) {
 case 1 -> incluirPessoaJuridica(scanner, repoPessoaJuridica);
 case 2 -> alterarPessoaJuridica(scanner, repoPessoaJuridica);
 case 3 -> excluirPessoaJuridica(scanner, repoPessoaJuridica);
 case 4 -> buscarPessoaJuridica(scanner, repoPessoaJuridica);
 case 5 -> buscarTodasPessoasJuridica(repoPessoaJuridica);
 case 6 -> persistirPessoasJuridicas(scanner, repoPessoaJuridica);
 case 7 -> recuperarPessoasJuridicas(scanner, repoPessoaJuridica);
 case 0 -> System.out.println("Voltando...");
 default -> System.out.println("Tente novamente.");
 }
 }
 case 0 -> System.out.println("Finalizando...");
 default -> System.out.println("Tente novamente.");
 }

 } while(tipo != 0);

 scanner.close();
 }

 }
```

```
 private static void incluirPessoaFisica(Scanner scanner, PessoaFisicaRepo
pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Incluir");

 System.out.print("informe id: (numero) ");
 int id = scanner.nextInt();
 scanner.nextLine();

 System.out.print("informe nome: (texto) ");
 String nome = scanner.nextLine();

 System.out.print("informe cpf: (texto) ");
 String cpf = scanner.nextLine();

 System.out.print("informe idade: (numero) ");
 int idade = scanner.nextInt();
 scanner.nextLine();

 PessoaFisica pessoaFisica = new PessoaFisica(id, nome, cpf, idade);
 pessoaFisicaRepo.inserir(pessoaFisica);

 System.out.println("Pessoa Física incluída com sucesso!");
 }

 private static void alterarPessoaFisica(Scanner scanner, PessoaFisicaRepo
pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Alterar");

 System.out.print("informe o id para localizarmos os dados: ");
 int id = scanner.nextInt();
 scanner.nextLine();

 PessoaFisica pessoaFisica = pessoaFisicaRepo.obter(id);
 if (pessoaFisica != null) {
 System.out.println("Dados atuais:");

 pessoaFisica.exibir();

 System.out.print("informe nome: (texto) ou [enter] para manter ");
 String nome = scanner.nextLine();
 pessoaFisica.setNome(nome);

 System.out.print("informe cpf: (texto) ");
 String cpf = scanner.nextLine();
 pessoaFisica.setCpf(cpf);

 System.out.print("informe idade: (numero) ");
 int idade = scanner.nextInt();
 scanner.nextLine();
 pessoaFisica.setIdade(idade);

 pessoaFisicaRepo.alterar(pessoaFisica);
 System.out.println("Pessoa Física alterada com sucesso!");
 } else {
 System.out.println("Pessoa Física não encontrada.");
 }
 }

 private static void excluirPessoaFisica(Scanner scanner, PessoaFisicaRepo
pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Excluir");

 System.out.print("id a ser excluido: ");
 int id = scanner.nextInt();
 scanner.nextLine();

 try {
 pessoaFisicaRepo.excluir(id);
 System.out.println("Pessoa Física excluída com sucesso!");

 } catch (IllegalArgumentException e) {
 System.out.println("Erro ao excluir: " + e.getMessage());
 }
 }

 private static void buscarPessoaFisica(Scanner scanner, PessoaFisicaRepo
pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Buscar");

 System.out.print("informe o id: ");
 int id = scanner.nextInt();
 scanner.nextLine();

 PessoaFisica pessoaFisica = pessoaFisicaRepo.obter(id);
 if (pessoaFisica != null) {
 System.out.println("Dados da Pessoa Física:");
 pessoaFisica.exibir();
 } else {
 System.out.println("Pessoa Fisica não encontrada.");
 }
 }

 private static void buscarTodasPessoasFisicas(PessoaFisicaRepo pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Buscar todos");

 for (PessoaFisica pessoa : pessoaFisicaRepo.obterTodos()) {
 pessoa.exibir();
 }
 }
```

```
 private static void persistirPessoasFisicas(Scanner scanner, PessoaFisicaRepo
pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Persistir Dados");

 System.out.print("informe um prefixo para o arquivo: ");
 String prefixoArquivo = scanner.next();

 try {
 pessoaFisicaRepo.persistir(prefixoArquivo + ".fisica.bin");
 System.out.println("Dados salvos com sucesso!");
 } catch (IOException ex) {
 System.err.println("Erro ao manipular arquivos: " + ex.getMessage());
 }
 }

 private static void recuperarPessoasFisicas(Scanner scanner, PessoaFisicaRepo
pessoaFisicaRepo) {
 System.out.println("Pessoa Fisica -> Recuperar Dados");

 System.out.print("informe um prefixo para o arquivo: ");
 String prefixoArquivo = scanner.next();

 try {
 pessoaFisicaRepo.recuperar(prefixoArquivo + ".fisica.bin");
 System.out.println("Dados recuperados com sucesso!");
 } catch (IOException | ClassNotFoundException ex) {
 System.err.println("Erro ao recuperar objeto: Classe não encontrada.");
 }
 }

 private static void incluirPessoaJuridica(Scanner scanner, PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.println("Pessoa Juridica -> Incluir");

 System.out.print("informe id: (numero) ");
 int id = scanner.nextInt();

 scanner.nextLine();

 System.out.print("informe nome: (texto) ");
 String nome = scanner.nextLine();

 System.out.print("informe cnpj: (texto) ");
 String cnpj = scanner.nextLine();

 PessoaJuridica pessoaJuridica = new PessoaJuridica(id, nome, cnpj);
 pessoaJuridicaRepo.inserir(pessoaJuridica);

 System.out.println("Pessoa Juridica incluída com sucesso!");
 }

 private static void alterarPessoaJuridica(Scanner scanner, PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.print("Pessoa Juridica -> Alterar");

 System.out.print("informe o id para localizarmos os dados: ");
 int id = scanner.nextInt();

 scanner.nextLine();

 PessoaJuridica pessoaJuridica = pessoaJuridicaRepo.obter(id);
 if (pessoaJuridica != null) {
 System.out.println("Dados atuais:");

 pessoaJuridica.exibir();


 System.out.print("informe nome: (texto) ");
 String nome = scanner.nextLine();
 pessoaJuridica.setNome(nome);

 System.out.print("informe cnpj: (texto) ");
 String cnpj = scanner.nextLine();
 pessoaJuridica.setCnpj(cnpj);

 pessoaJuridicaRepo.alterar(pessoaJuridica);
 System.out.println("Pessoa Juridica alterada com sucesso!");
 } else {
 System.out.println("Pessoa Juridica não encontrada.");
 }
 }

 private static void excluirPessoaJuridica(Scanner scanner, PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.println("Pessoa Juridica -> Excluir");

 System.out.print("id a ser excluido: ");
 int id = scanner.nextInt();
 scanner.nextLine();

 try {
 pessoaJuridicaRepo.excluir(id);
 System.out.println("Pessoa Jurisica excluída com sucesso!");

 } catch (IllegalArgumentException e) {
 System.out.println("Erro ao excluir: " + e.getMessage());
 }
 }
```

```
 private static void buscarPessoaJuridica(Scanner scanner, PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.println("Pessoa Juridica -> Buscar");

 System.out.print("informe o id: ");
 int id = scanner.nextInt();
 scanner.nextLine();

 PessoaJuridica pessoaJuridica = pessoaJuridicaRepo.obter(id);
 if (pessoaJuridica != null) {
 System.out.println("Dados da Pessoa Juridica:");
 pessoaJuridica.exibir();
 } else {
 System.out.println("Pessoa Juridica não encontrada.");
 }
 }

 private static void buscarTodasPessoasJuridica(PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.println("Pessoa Juridica -> Buscar todos");

 for (PessoaJuridica pessoa : pessoaJuridicaRepo.obterTodos()) {
 pessoa.exibir();
 }
 }

 private static void persistirPessoasJuridicas(Scanner scanner, PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.println("Pessoa Juridica -> Persistir Dados");

 System.out.print("informe um prefixo para o arquivo: ");
 String prefixoArquivo = scanner.next();

 try {
 pessoaJuridicaRepo.persistir(prefixoArquivo + ".juiridica.bin");
 System.out.println("Dados salvos com sucesso!");
 } catch (IOException ex) {
 System.err.println("Erro ao manipular arquivos: " + ex.getMessage());
 }
 }

 private static void recuperarPessoasJuridicas(Scanner scanner, PessoaJuridicaRepo
pessoaJuridicaRepo) {
 System.out.println("Pessoa Juridica -> Recuperar Dados");

 System.out.print("informe um prefixo para o arquivo: ");
 String prefixoArquivo = scanner.next();

 try {
 pessoaJuridicaRepo.recuperar(prefixoArquivo + ".juiridica.bin");
 System.out.println("Dados recuperados com sucesso!");
 } catch (IOException | ClassNotFoundException ex) {
 System.err.println("Erro ao recuperar objeto: Classe não encontrada.");
 }
 }
}
```