POLO OLARIA - RIO DE JANEIRO - RJ

DESENVOLVIMENTO FULL STACK

Nivel 3: Back-End Sem Banco Não Tem | Turma 9001

Erik Bastos de Moraes

Missão Prática | Nível 3 | Mundo 3

2º Procedimento | Alimentando a Base

Objetivos da prática

1- Implementar persistência com base no middleware JDBC.

2- Utilizar o padrão DAO (Data Access Object) no manuseio de dados.

3- Implementar o mapeamento objeto-relacional em sistemas Java.

4- Criar sistemas cadastrais com persistência em banco relacional.

5- No final do exercício, o aluno terá criado um aplicativo cadastral com uso do SQL Server na persistência de dados.

PessoaFisica.java

```
package cadastrobd.model;


public class PessoaFisica extends Pessoa {
    String cpf;


    public PessoaFisica() {}


    public PessoaFisica(int id, String nome, String logradouro, String cidade, String
estado, String telefone, String email, String cpf) {
        super(id, nome, logradouro, cidade, estado, telefone, email);
        this.cpf = cpf;
    }


    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CPF: " + cpf);
    }


    @Override
public String toString() {
    return """
            Pessoa F\u00edsica:
            ID: """ + id + "\n" +
            "Nome: " + nome + "\n" +
            "Logradouro: " + logradouro + "\n" +
            "Cidade: " + cidade + "\n" +
            "Estado: " + estado + "\n" +
```

```
        "Telefone: " + telefone + "\n" +
        "Email: " + email + "\n" +
        "CPF: " + cpf + "\n";
}


}
```

PessoaJuridica.java

```
package cadastrobd.model;


public class PessoaJuridica extends Pessoa {
    String cnpj;


    public PessoaJuridica() {}


    public PessoaJuridica(int id, String nome, String logradouro, String cidade, String
estado, String telefone, String email, String cnpj) {
        super(id, nome, logradouro, cidade, estado, telefone, email);
        this.cnpj = cnpj;
    }


    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CNPJ: " + cnpj);
    }


    @Override
public String toString() {
    return """
            Pessoa Jur\u00eddica:
            ID: """ + id + "\n" +
            "Nome: " + nome + "\n" +
            "Logradouro: " + logradouro + "\n" +
            "Cidade: " + cidade + "\n" +
            "Estado: " + estado + "\n" +
```

```
        "Telefone: " + telefone + "\n" +
        "Email: " + email + "\n" +
        "CNPJ: " + cnpj + "\n";
}
}
```

CadastroBDTeste.java

```
import cadastrobd.model.PessoaFisica;
import cadastrobd.model.PessoaJuridica;
import cadastrobd.model.PessoaFisicaDAO;
import cadastrobd.model.PessoaJuridicaDAO;

import java.util.List;
import java.util.Scanner;

public class CadastroBDTeste {

    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            PessoaFisicaDAO pfDAO = new PessoaFisicaDAO();
            PessoaJuridicaDAO pjDAO = new PessoaJuridicaDAO();
            int opcao;

            do {
                System.out.println("MENU");
                System.out.println("1 - Incluir");
                System.out.println("2 - Alterar");
                System.out.println("3 - Excluir");
                System.out.println("4 - Exibir por ID");
                System.out.println("5 - Exibir todos");
                System.out.println("0 - Sair");

                opcao = lerInt(scanner, "Escolha uma opção: ");
```

```
            switch (opcao) {
                case 1 -> incluir(scanner, pfDAO, pjDAO);
                case 2 -> alterar(scanner, pfDAO, pjDAO);
                case 3 -> excluir(scanner, pfDAO, pjDAO);
                case 4 -> exibirPorId(scanner, pfDAO, pjDAO);
                case 5 -> exibirTodos(scanner, pfDAO, pjDAO);
                case 0 -> System.out.println("Encerrando o programa...");
                default -> System.out.println("Opção inválida!");
            }
        } while (opcao != 0);
    }
}

private static int lerInt(Scanner scanner, String mensagem) {
    int valor;
    while (true) {
        System.out.print(mensagem);
        try {
            valor = Integer.parseInt(scanner.nextLine());
            return valor;
        } catch (NumberFormatException e) {
            System.out.println("Entrada inválida. Por favor, digite um número inteiro.");
        }
    }
}

private static void incluir(Scanner scanner, PessoaFisicaDAO pfDAO,
PessoaJuridicaDAO pjDAO) {
    int tipo = lerInt(scanner, "Tipo (1 - Física, 2 - Jurídica): ");
```

```
        System.out.print("Nome: ");
        String nome = scanner.nextLine();
        System.out.print("Logradouro: ");
        String logradouro = scanner.nextLine();
        System.out.print("Cidade: ");
        String cidade = scanner.nextLine();
        System.out.print("Estado: ");
        String estado = scanner.nextLine();
        System.out.print("Telefone: ");
        String telefone = scanner.nextLine();
        System.out.print("Email: ");
        String email = scanner.nextLine();

        switch (tipo) {
            case 1 -> {
                System.out.print("CPF: ");
                String cpf = scanner.nextLine();
                PessoaFisica pf = new PessoaFisica(0, nome, logradouro, cidade, estado,
telefone, email, cpf);
                pfDAO.incluir(pf);
                System.out.println("Pessoa Física incluída com sucesso!");
            }
            case 2 -> {
                System.out.print("CNPJ: ");
                String cnpj = scanner.nextLine();
                PessoaJuridica pj = new PessoaJuridica(0, nome, logradouro, cidade, estado,
telefone, email, cnpj);
                pjDAO.incluir(pj);
                System.out.println("Pessoa Jurídica incluída com sucesso!");
            }
            default -> System.out.println("Tipo inválido!");
```

```
        }
    }


    private static void alterar(Scanner scanner, PessoaFisicaDAO pfDAO,
PessoaJuridicaDAO pjDAO) {
        int tipo = lerInt(scanner, "Tipo (1 - Física, 2 - Jurídica): ");
        int id = lerInt(scanner, "ID: ");


        System.out.print("Nome: ");
        String nome = scanner.nextLine();
        System.out.print("Logradouro: ");
        String logradouro = scanner.nextLine();
        System.out.print("Cidade: ");
        String cidade = scanner.nextLine();
        System.out.print("Estado: ");
        String estado = scanner.nextLine();
        System.out.print("Telefone: ");
        String telefone = scanner.nextLine();
        System.out.print("Email: ");
        String email = scanner.nextLine();


        switch (tipo) {
            case 1 -> {
                System.out.print("CPF: ");
                String cpf = scanner.nextLine();
                PessoaFisica pf = new PessoaFisica(id, nome, logradouro, cidade, estado,
telefone, email, cpf);
                pfDAO.alterar(pf);
                System.out.println("Pessoa Física alterada com sucesso!");
            }
            case 2 -> {
```

```
            System.out.print("CNPJ: ");
            String cnpj = scanner.nextLine();
            PessoaJuridica pj = new PessoaJuridica(id, nome, logradouro, cidade, estado, telefone, email, cnpj);
            pjDAO.alterar(pj);
            System.out.println("Pessoa Jurídica alterada com sucesso!");
        }
        default -> System.out.println("Tipo inválido!");
    }
}

private static void excluir(Scanner scanner, PessoaFisicaDAO pfDAO, PessoaJuridicaDAO pjDAO) {
    int tipo = lerInt(scanner, "Tipo (1 - Física, 2 - Jurídica): ");
    int id = lerInt(scanner, "ID: ");

    switch (tipo) {
        case 1 -> {
            pfDAO.excluir(id);
            System.out.println("Pessoa Física excluída (se existia).");
        }
        case 2 -> {
            pjDAO.excluir(id);
            System.out.println("Pessoa Jurídica excluída (se existia).");
        }
        default -> System.out.println("Tipo inválido!");
    }
}

private static void exibirPorId(Scanner scanner, PessoaFisicaDAO pfDAO, PessoaJuridicaDAO pjDAO) {
```

```
        int tipo = lerInt(scanner, "Tipo (1 - Física, 2 - Jurídica): ");
        int id = lerInt(scanner, "ID: ");

        switch (tipo) {
            case 1 -> {
                PessoaFisica pf = pfDAO.getPessoa(id);
                System.out.println(pf != null ? pf : "Pessoa Física não encontrada!");
            }
            case 2 -> {
                PessoaJuridica pj = pjDAO.getPessoa(id);
                System.out.println(pj != null ? pj : "Pessoa Jurídica não encontrada!");
            }
            default -> System.out.println("Tipo inválido!");
        }
    }

    private static void exibirTodos(Scanner scanner, PessoaFisicaDAO pfDAO,
PessoaJuridicaDAO pjDAO) {
        int tipo = lerInt(scanner, "Tipo (1 - Física, 2 - Jurídica): ");

        switch (tipo) {
            case 1 ->               {
                    List<PessoaFisica> pessoas = pfDAO.getPessoas();
                    if (pessoas.isEmpty()) {
                        System.out.println("Nenhuma Pessoa Física cadastrada.");
                    } else {
                        pessoas.forEach(System.out::println);
                    }                   }
            case 2 ->               {
                    List<PessoaJuridica> pessoas = pjDAO.getPessoas();
```

```
                if (pessoas.isEmpty()) {
                    System.out.println("Nenhuma Pessoa Jurídica cadastrada.");
                } else {
                    pessoas.forEach(System.out::println);
                }                    }
            default -> System.out.println("Tipo inválido!");
        }
    }
}
```

Resultado

MENU

1 - Incluir

2 - Alterar

3 - Excluir

4 - Exibir por ID

5 - Exibir todos

0 - Sair

Escolha uma opção:

Análise e Conclusão

A. Quais as diferenças entre a persistência em arquivo e a persistência em banco de dados?

R: A persistência em banco de dados é mais eficiente manipulando um grande número de dados, além de recursos mais avançados, enquanto a em arquivo é simples e não requer software adicional, sendo mais ineficiente.

B. Como o uso de operador lambda simplificou a impressão dos valores contidos nas entidades, nas versões mais recentes do Java?

R: Operadores lambda simplificam a impressão dos valores contidos nas entidades em Java, proporcionando uma sintaxe mais concisa, resultando em um código mais limpo.

C. Por que métodos acionados diretamente pelo método main, sem o uso de um objeto, precisam ser marcados como static?

R: Pois o método main é o ponto de entrada, sendo executado em um conceito estático e apenas membros estáticos da mesma classe são acessíveis.

---

https://github.com/ErikBM2661/CadastroBD.git