Iniciando o caminho pelo Java

Missão Prática | Nível 1 | mundo 3

POLO PRQ ARTUR ALVIM - SÃO PAULO – SP

Desenvolvimento Full Stack

Iniciando o caminho pelo Java

Turma 2025.1

Primeiro Semestre (2025)

Daniel Souza Pereira

Matrícula: 202401488135

Implementação de um cadastro de clientes em modo texto, com persistência em arquivos, baseado na tecnologia Java.

Objetivo da prática

Utilizar herança e polimorfismo na definição de entidades;

Criar uma hierarquia de classes (Pessoa, PessoaFisica, PessoaJuridica) para demonstrar herança e polimorfismo;

Utilizar persistência de objetos em arquivos binários;

Implementar métodos para salvar e recuperar dados em arquivos binários usando serialização.

Implementar uma interface cadastral em modo texto;

Desenvolver um menu interativo para interação com o usuário via linha de comando;

Utilizar o controle de exceções da plataforma Java;

Tratar possíveis erros durante operações como persistência e recuperação de dados.

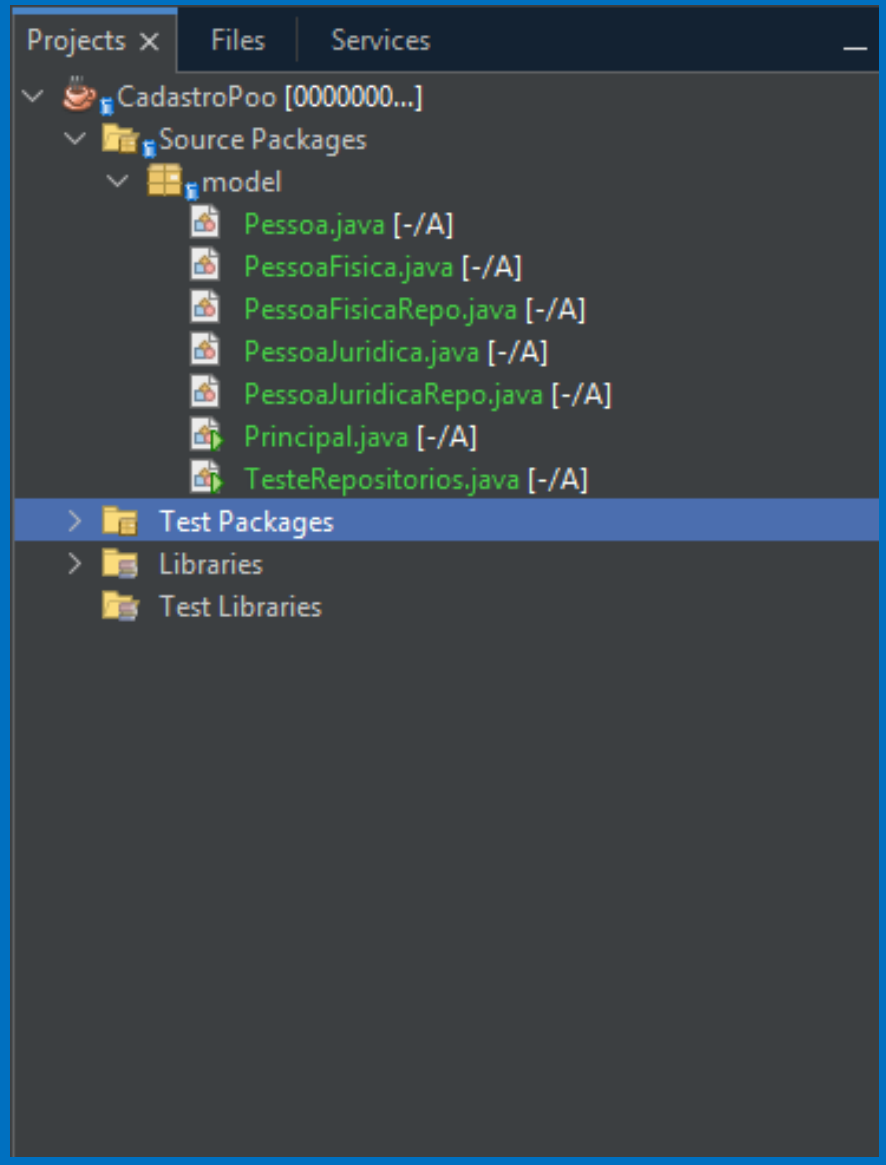
Segundo Procedimento:

Criação do Cadastro em Modo Texto

Códigos utilizados:

Alterar o método main da classe principal do projeto, para implementação do cadastro em modo texto:

Selecionada a opção **incluir**, escolher o tipo (Física ou Jurídica);
Selecionada a opção **alterar**, escolher o tipo (Física ou Jurídica);
Selecionada a opção **excluir**, escolher o tipo (Física ou Jurídica);
Selecionada a opção **obter**, escolher o tipo (Física ou Jurídica);
Selecionada a opção **obterTodos**, escolher o tipo (Física ou Jurídica);
Selecionada a opção **salvar**, solicitar o prefixo dos arquivos e persistir os dados nos arquivos [prefixo].fisica.bin e [prefixo].juridica.bin.
Selecionada a opção recuperar, solicitar o prefixo dos arquivos e obter os dados a partir dos arquivos [prefixo].fisica.bin e [prefixo].juridica.bin.
Nas opções **salvar** e recuperar devem ser tratadas as exceções.
Selecionada a opção **sair**, finalizar a execução do sistema.



Classe Pessoa

Pessoa.java [-/A]

Source | History

```java
package model;

import java.io.Serializable;

public class Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;

    private int id;
    private String nome;

    public Pessoa() {
    }
    public Pessoa (int id, String nome) {
        this.id = id;
        this.nome = nome;
    }
    public void exibir() {
        System.out.println("ID:" + id);
        System.out.println("Nome:" + nome);
    }
    public int getId () {
        return id;
    }
    public void setId(int Id) {
        this.id = id;
    }
    public String getNome() {
        return nome;
    }
    public void setNome(String nome){
        this.nome = nome;
    }
}
```

Classe PessoaFisica

PessoaFisica.java [-/A]

Source | History

```java
package model;

import java.io.Serializable;

public class PessoaFisica extends Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;

    private String cpf;
    private int idade;

    public PessoaFisica(){
    }

    public PessoaFisica (int id, String nome, String cpf, int idade){
        super(id, nome);
        this.cpf = cpf;
        this.idade = idade;
    }

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

    public String getCpf(){
        return cpf;
    }
    public void setCpf (String cpf) {
        this.cpf = cpf;
    }
    public int getIdade(){
        return idade;
    }
    public void setIdade(int idade) {
        this.idade = idade;
    }
}

```

Classe PessoaJuridica

PessoaJuridica.java [-/A]

Source | History

```java
package model;

import java.io.Serializable;

public class PessoaJuridica extends Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;

    private String cnpj;

    public PessoaJuridica(){
    }

    public PessoaJuridica (int id, String nome, String cnpj) {
        super(id, nome);
        this.cnpj = cnpj;
    }

    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CNPJ:" + cnpj);
    }
    public String getCnpj(){
        return cnpj;
    }
    public void setCnpj(String cnpj){
        this.cnpj = cnpj;
    }
}
```

Classe PessoaFisicaRepo

PessoaFisicaRepo.java [-/A]

Source | History

```java
package model;

import java.io.*;
import java.util.ArrayList;

public class PessoaFisicaRepo {
    private ArrayList<PessoaFisica> pessoas = new ArrayList<>();

    public void inserir (PessoaFisica pessoa){
        pessoas.add(pessoa);
    }

    public void alterar (PessoaFisica pessoa){
        for (int i = 0; i < pessoas.size(); i++) {
            if (pessoas.get(i).getId() == pessoa.getId()){
                pessoas.set(i, pessoa);
                break;
            }
        }
    }

    public void excluir(int id){
        pessoas.removeIf(p -> p.getId() == id);
    }

    public PessoaFisica obter(int id) {
        for (PessoaFisica pessoa : pessoas){
            if (pessoa.getId() == id){
                return pessoa;
            }
        }
        return null;
    }
    public ArrayList<PessoaFisica> obterTodos(){
        return pessoas;
    }
    public void Persistir (String nomeArquivo) throws IOException {
        try (ObjectOutputStream outputStream = new ObjectOutputStream(new FileOutputStream(nomeArquivo))) {
            outputStream.writeObject(pessoas);
        }
    }
    public void recuperar (String nomeArquivo) throws IOException, ClassNotFoundException {
        try (ObjectInputStream inputStream = new ObjectInputStream(new FileInputStream(nomeArquivo))) {
            pessoas = (ArrayList<PessoaFisica>) inputStream.readObject();
        }
    }
}
```

Classe PessoaJuridicaRepo

PessoaJuridicaRepo.java [-/A]

Source | History

C:\Users\rockn\OneDrive\Área de Trabalho\JavaProjects\Aulas\CadastroPoo\src\model\PessoaJuridicaRepo.java

```java
package model;

import java.io.*;
import java.util.ArrayList;

public class PessoaJuridicaRepo {
    private ArrayList<PessoaJuridica> pessoas = new ArrayList<>();

    public void inserir(PessoaJuridica pessoa) {
        pessoas.add(pessoa);
    }

    public void alterar (PessoaJuridica pessoa) {
        for (int i = 0; i < pessoas.size(); i++) {
            if (pessoas.get(i).getId() == pessoa.getId()) {
                pessoas.set(i, pessoa);
                break;
            }
        }
    }

    public void excluir(int id){
        pessoas.removeIf(p -> p.getId() == id);
    }

    public PessoaJuridica obter(int id) {
        for (PessoaJuridica pessoa : pessoas) {
            if (pessoa.getId() == id) {
                return pessoa;
            }
        }
        return null;
    }

    public ArrayList<PessoaJuridica> obterTodos() {
        return pessoas;
    }
    public void persistir(String nomeArquivo) throws IOException {
        try (ObjectOutputStream outputStream = new ObjectOutputStream(new FileOutputStream(nomeArquivo))) {
            outputStream.writeObject(pessoas);
        }
    }
    public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
        try (ObjectInputStream inputStream = new ObjectInputStream(new FileInputStream(nomeArquivo))) {
            pessoas = (ArrayList<PessoaJuridica>) inputStream.readObject();
        }
    }
}
```

Classe Principal

```java
package model;

import java.util.Scanner;
import java.io.IOException;

public class Principal {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        PessoaFisicaRepo repoFisica = new PessoaFisicaRepo();
        PessoaJuridicaRepo repoJuridica = new PessoaJuridicaRepo();

        int opcao;
        do {
            System.out.println("\n=== Menu ===");
            System.out.println("1 - Incluir");
            System.out.println("2 - Alterar");
            System.out.println("3 = Excluir");
            System.out.println("4 - Exibir pelo ID");
            System.out.println("5 - Exibir todos");
            System.out.println("6 - Salvar Dados");
            System.out.println("7 - recuperar Dados");
            System.out.println("0 - Sair");
            System.out.println("Escolha uma Opcão:");
            opcao = scanner.nextInt();
            scanner.nextLine();
```

```java
            try {
                switch (opcao){
                    case 1:
                        incluir(scanner, repoFisica, repoJuridica);
                        break;
                    case 2:
                        alterar(scanner, repoFisica, repoJuridica);
                        break;
                    case 3:
                        excluir(scanner, repoFisica, repoJuridica);
                        break;
                    case 4:
                        exibirPorId(scanner, repoFisica, repoJuridica);
                        break;
                    case 5:
                        exibirTodos(scanner, repoFisica, repoJuridica);
                        break;
                    case 6:
                        salvarDados(scanner, repoFisica, repoJuridica);
                        break;
                    case 7:
                        recuperarDados(scanner, repoFisica, repoJuridica);
                        break;
                    case 0:
                        System.out.println("Finalizando Sistema...");
                        break;
                    default:
                        System.out.println("Opção Invalida. Por favor tente novamente");
                }
            } catch (Exception e){
                System.out.println("Erro: " + e.getMessage());
            }
```

Classe Principal

```
    } while (opcao != 0);
    scanner.close();
}
private static void incluir (Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica) {
    System.out.println("Escolha o tipo (1 - fisica - 2 Juridica):");
    int tipo = scanner.nextInt();
    scanner.nextLine();

    if (tipo == 1) {
        System.out.println("ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();
        System.out.println("Nome: ");
        String nome = scanner.nextLine();
        System.out.println("CPF: ");
        String cpf = scanner.nextLine();
        System.out.println("Idade: ");
        int idade = scanner.nextInt();
        repoFisica.inserir(new PessoaFisica(id, nome, cpf, idade));
    } else if (tipo == 2){
        System.out.println("ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();
        System.out.println("Nome: ");
        String nome = scanner.nextLine();
        System.out.println("CNPJ: ");
        String cnpj = scanner.nextLine();
        repoJuridica.inserir(new PessoaJuridica(id, nome, cnpj));
    } else {
        System.out.println("Tipo Invalido!");
    }
}
public static void alterar (Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica) {
    System.out.println("Escolha o tipo (1 - fisica - 2 Juridica):");
    int tipo = scanner.nextInt();
    scanner.nextLine();
```

```
        System.out.println("ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        if (tipo == 1) {
            PessoaFisica pessoa = repoFisica.obter(id);
            if (pessoa != null){
                System.out.println("Dados Atuais:");
                pessoa.exibir();
                System.out.println("Novo Nome: ");
                String nome = scanner.nextLine();
                System.out.println("CPF: ");
                String cpf = scanner.nextLine();
                System.out.println("Nova Idade: ");
                int idade = scanner.nextInt();
                pessoa.setNome(nome);
                pessoa.setCpf(cpf);
                pessoa.setIdade(idade);
                repoFisica.alterar(pessoa);
                System.out.println("Pessoa Fisica alterada com sucesso!");
            } else {
                System.out.println("Pessoa Fisica Não encontrada!");
            }
        } else if (tipo == 2){
            PessoaJuridica pessoa = repoJuridica.obter(id);
            if (pessoa != null){
                System.out.println("Dados Atuais:");
                pessoa.exibir();
                System.out.println("Novo Nome: ");
                String nome = scanner.nextLine();
                System.out.println("Novo CNPJ: ");
                String cnpj = scanner.nextLine();
                pessoa.setNome(nome);
                pessoa.setCnpj(cnpj);
                repoJuridica.alterar(pessoa);
                System.out.println("Pessoa Juridica alterada com sucesso!");
            } else {
                System.out.println("Pessoa Juridica Não encontrada!");
            }
        }else {
            System.out.println("Tipo Inválido!");

        }
    }
```

```
private static void excluir(Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica){
    System.out.println("Escolha o tipo (1 - fisica - 2 Juridica):");
    int tipo = scanner.nextInt();
    scanner.nextLine();

    System.out.println("ID: ");
    int id = scanner.nextInt();

    if (tipo == 1){
        repoFisica.excluir(id);
        System.out.println("Pessoa Fisica excluida com sucesso!");
    }else if (tipo == 2){
        repoJuridica.excluir(id);
        System.out.println("Pessoa Juridica excluida com sucesso!");
    }else {
        System.out.println("Tipo Inválido!");
    }
}

private static void exibirPorId(Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica){
    System.out.println("Escolha o tipo (1 - fisica - 2 Juridica):");
    int tipo = scanner.nextInt();
    scanner.nextLine();

    System.out.println("ID: ");
    int id = scanner.nextInt();

    if (tipo == 1){
        PessoaFisica pessoa = repoFisica.obter(id);
        if (pessoa != null) {
            pessoa.exibir();
        }else {
            System.out.println("Pessoa Fisica não encontrada!");
        }
    }else if (tipo == 2){
        PessoaJuridica pessoa = repoJuridica.obter(id);
        if (pessoa != null){
            pessoa.exibir();
        }else {
            System.out.println("Pessoa Juridoca não encontrada!");
        }
    }else {
        System.out.println("Tipo Inválido!");
    }
}
```

Classe Principal

```
    public static void exibirTodos(Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica){
        System.out.println("Escolha o tipo (1 - fisica - 2 Juridica):");
        int tipo = scanner.nextInt();
        scanner.nextLine();

        if (tipo == 1){
            for (PessoaFisica pessoa : repoFisica.obterTodos()) {
                pessoa.exibir();
                System.out.println("--------------------------");
            }
        }else if (tipo == 2) {
            for (PessoaJuridica pessoa : repoJuridica.obterTodos()) {
                pessoa.exibir();
                System.out.println("--------------------------");
            }
        }else {
            System.out.println("Tipo Inválido!");
        }
    }
    public static void salvarDados(Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica) throws IOException {
        System.out.println("Informe o prefixo dos arquivos: ");
        String prefixo = scanner.nextLine();

        repoFisica.Persistir(prefixo + ".fisica.bin");
        repoJuridica.persistir(prefixo + ".juridica.bin");
        System.out.println("Dados Salvos com sucesso!");
    }

    private static void recuperarDados (Scanner scanner,PessoaFisicaRepo repoFisica, PessoaJuridicaRepo repoJuridica) throws IOException, ClassNotFoundException {
        System.out.println("informe o prefixo dos arquivos ");
        String prefixo = scanner.nextLine();

        repoFisica.recuperar(prefixo + ".fisica.bin");
        repoJuridica.recuperar(prefixo + ".juridica.bin");
        System.out.println("Dados recuperados com Sucesso!");
    }
}
```

Saída de Execução

```
run:

=== Menu ===
1 - Incluir
2 - Alterar
3 = Excluir
4 - Exibir pelo ID
5 - Exibir todos
6 - Salvar Dados
7 - recuperar Dados
0 - Sair
Escolha uma Opc�o:
```

```
run:

=== Menu ===
1 - Incluir
2 - Alterar
3 = Excluir
4 - Exibir pelo ID
5 - Exibir todos
6 - Salvar Dados
7 - recuperar Dados
0 - Sair
Escolha uma Opc�o:
1
Escolha o tipo (1 - fisica - 2 Juric
2
ID:
3421
Nome:
Daniel
CNPJ:
52.168.778/0001-08
```

```
run:

=== Menu ===
1 - Incluir
2 - Alterar
3 = Excluir
4 - Exibir pelo ID
5 - Exibir todos
6 - Salvar Dados
7 - recuperar Dados
0 - Sair
Escolha uma Opc�o:
1
Escolha o tipo (1 - fisica - 2 Juridica):
2
ID:
3421
Nome:
Daniel
CNPJ:
52.168.778/0001-08
```

Análise e Conclusão:

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

Elementos estáticos são associados à classe e não às instâncias (objetos) da classe. Isso significa que eles pertencem à classe como um todo e podem ser acessados sem criar objetos dessa classe

O método main é o ponto de entrada de qualquer programa Java. Ele precisa ser estático porque o Java não cria automaticamente uma instância da classe principal ao iniciar o programa.

Com o modificador static, o método main pode ser chamado diretamente pela JVM (Java Virtual Machine) sem a necessidade de instanciar a classe

Análise e Conclusão:

2. Para que serve a classe Scanner?

A classe Scanner emJava é usada para ler dados de entrada do usuário ou de outras fontes, como arquivos. Ela é amplamente utilizada porque simplifica a tarefa de capturar e processar entradas em diferentes formatos (como números, strings, etc.).

Como funciona?
Um objeto Scanner é criado associado à entrada padrão (System.in), que é o teclado. Métodos como nextInt(), nextLine(), nextDouble() são usados para ler os dados inseridos pelo usuário.

Importância no projeto:
No menu interativo, a classe Scanner foi essencial para permitir que o usuário escolhesse opções, inserisse dados (como nome, CPF, CNPJ) e interagisse com o sistema.

Análise e Conclusão:

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

Impacto **na organização do código**: Separação de Responsabilidades:
As classes de repositório centralizam as operações de **CRUD** e persistência, enquanto a **classe Principal** foca na interação com o usuário.
Isso torna o código mais modular e fácil de manter.
Reutilização de Código: As classes de repositório podem ser reutilizadas em outros projetos ou partes do sistema sem a necessidade de reescrever a lógica de persistência.
Facilidade de Testes: Como as operações de persistência estão isoladas nas classes de repositório, é mais fácil testá-las independentemente do restante do sistema.
Escalabilidade: Caso seja necessário alterar a forma de persistência (ex.: migrar de arquivos binários para um banco de dados), basta modificar as classes de repositório, sem impactar o restante do código.

Link para o Repositório no GitHub:

DannPereira10/Iniciando-o-caminho-pelo-Java