Estácio

Campus: Interlagos

Curso: Desenvolvimento Full Stack (2024.3)

Disciplina: RPG0014 - Iniciando o caminho pelo Java

Turma: EAD

Integrantes: 202303832338 - Rafael Rosendo Tagliaferro

github: https://github.com/minoscorpion/est006

codigo fonte Em : CadastroPOO - Procedimento 2.zip

1º Procedimento | Criação das Entidades e Sistema de Persistência

Implementação de um cadastro de clientes em modo texto, com persistência em

arquivos, baseado na tecnologia Java.

Objetivos da prática

- Utilizar herança e polimorfismo na definição de entidades.
- Utilizar persistência de objetos em arquivos binários.
- Implementar uma interface cadastral em modo texto.
- Utilizar o controle de exceções da plataforma Java.
- 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

Conclusão

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

Os elementos estaticos são objetos que são carregados em memoria e possuem valor fixo durante o carregamento do sistema, eles estão sempre endereçados para o mesmo lugar da memoria, tornando assim, muito mais facil de administra-los e seus respectivos valores. Podem ser facilmente carregados para dentro de outras classes, sem perder seus atributos e valores.

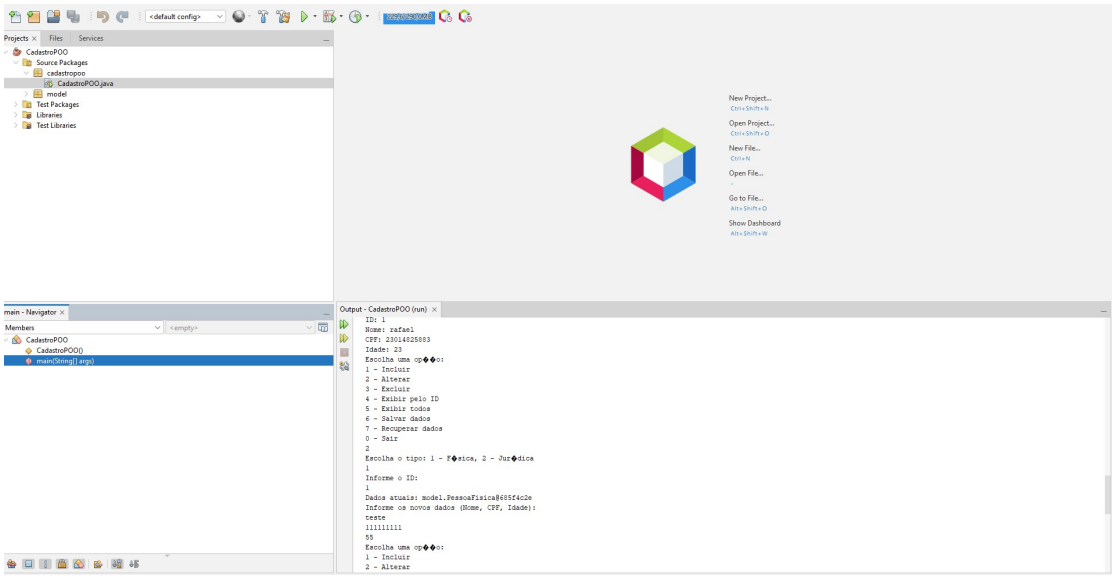
- Para que serve a classe Scanner?

Serve para fazer input de dados atraves do console, preenchendo variaveis inteiraras, textos etc.

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

Ter uma classe de repositorio ajudou a dar finalidade ao codigo, sem misturar as classes de definição (model), com as regras de negocio / gravação dos dados (repo), alem disso possibilitou ser muito mais rapido outras implementações, por ex, este procedimento 2.

RESULTADO DO CÓDIGO



Códigos do projeto:

Main

```
package cadastropoo;

import model.PessoaFisica;

import model.PessoaJuridica;

import model.PessoaFisicaRepo;

import model.PessoaJuridicaRepo;

/*

 * 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

 */

import java.io.IOException;

import java.util.Scanner;

/**
```

```
 *
 * @author rafae
 */
public class CadastroPOO {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            PessoaFisicaRepo repoFisica = new PessoaFisicaRepo();
            PessoaJuridicaRepo repoJuridica = new PessoaJuridicaRepo();
            int opcao;
            do {
                System.out.println("Escolha uma opcao:");
                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");
                opcao = scanner.nextInt();
                scanner.nextLine(); // Limpa o buffer
                switch (opcao) {
                    case 1 -> {
```

```
    // Incluir
    System.out.println("Escolha o tipo: 1 - Física, 2 - Jurídica");
    int tipoInclusao = scanner.nextInt();
    scanner.nextLine(); // Limpa o buffer
    if (tipoInclusao == 1) {
        // Incluir Pessoa Física
        System.out.println("Informe o ID, Nome, CPF e Idade:");
        int id = scanner.nextInt();
        scanner.nextLine(); // Limpa o buffer
        String nome = scanner.nextLine();
        String cpf = scanner.nextLine();
        int idade = scanner.nextInt();
        repoFisica.inserir(new PessoaFisica(id, nome, cpf, idade));
    } else if (tipoInclusao == 2) {
        // Incluir Pessoa Jurídica
        System.out.println("Informe o ID, Nome e CNPJ:");
        int id = scanner.nextInt();
        scanner.nextLine(); // Limpa o buffer
        String nome = scanner.nextLine();
        String cnpj = scanner.nextLine();
        repoJuridica.inserir(new PessoaJuridica(id, nome, cnpj));
    }
}

case 2 -> {
    // Alterar
```

```
System.out.println("Escolha o tipo: 1 - Fisica, 2 - Juridica");
int tipoAlteracao = scanner.nextInt();
scanner.nextLine(); // Limpa o buffer
System.out.println("Informe o ID:");
int idAlterar = scanner.nextInt();
scanner.nextLine(); // Limpa o buffer

if (tipoAlteracao == 1) {
    // Alterar Pessoa Física
    PessoaFisica pessoaFisica = repoFisica.obter(idAlterar);
    if (pessoaFisica != null) {
        System.out.println("Dados atuais: " + pessoaFisica);
        System.out.println("Informe os novos dados (Nome, CPF, Idade):");
        String nome = scanner.nextLine();
        String cpf = scanner.nextLine();
        int idade = scanner.nextInt();
        pessoaFisica.setNome(nome);
        pessoaFisica.setCpf(cpf);
        pessoaFisica.setIdade(idade);
    }
} else if (tipoAlteracao == 2) {
    // Alterar Pessoa Jurídica
    PessoaJuridica pessoaJuridica = repoJuridica.obter(idAlterar);
    if (pessoaJuridica != null) {
        System.out.println("Dados atuais: " + pessoaJuridica);
        System.out.println("Informe os novos dados (Nome, CNPJ):");
```

```
            String nome = scanner.nextLine();
            String cnpj = scanner.nextLine();
            pessoaJuridica.setNome(nome);
            pessoaJuridica.setCnpj(cnpj);
        }
    }
}

case 3 -> {
    // Excluir
    System.out.println("Escolha o tipo: 1 - Física, 2 - Jurídica");
    int tipoExclusao = scanner.nextInt();
    scanner.nextLine(); // Limpa o buffer
    System.out.println("Informe o ID:");
    int idExcluir = scanner.nextInt();

    if (tipoExclusao == 1) {
        repoFisica.excluir(idExcluir);
    } else if (tipoExclusao == 2) {
        repoJuridica.excluir(idExcluir);
    }
}

case 4 -> {
    // Exibir pelo ID
    System.out.println("Escolha o tipo: 1 - Fisica, 2 - Juridica");
```

```
    int tipoExibicao = scanner.nextInt();
    scanner.nextLine(); // Limpa o buffer
    System.out.println("Informe o ID:");
    int idExibir = scanner.nextInt();

    if (tipoExibicao == 1) {
        PessoaFisica pessoa = repoFisica.obter(idExibir);
        if (pessoa != null) {
            pessoa.exibir();
        } else {
            System.out.println("Pessoa Fisica não encontrada.");
        }
    } else if (tipoExibicao == 2) {
        PessoaJuridica pessoa = repoJuridica.obter(idExibir);
        if (pessoa != null) {
            pessoa.exibir();
        } else {
            System.out.println("Pessoa Juridica não encontrada.");
        }
    }
}

case 5 -> {
    // Exibir todos
    System.out.println("Escolha o tipo: 1 - Fisica, 2 - Juridica");
    int tipoExibirTodos = scanner.nextInt();
```

```
    if (tipoExibirTodos == 1) {
        for (PessoaFisica p : repoFisica.obterTodos()) {
            p.exibir();
        }
    } else if (tipoExibirTodos == 2) {
        for (PessoaJuridica p : repoJuridica.obterTodos()) {
            p.exibir();
        }
    }
}

case 6 -> {
    // Salvar dados
    System.out.println("Informe o prefixo dos arquivos:");
    String prefixoSalvar = scanner.nextLine();
    try {
        repoFisica.persistir(prefixoSalvar + ".fisica.bin");
        repoJuridica.persistir(prefixoSalvar + ".juridica.bin");
        System.out.println("Dados salvos com sucesso.");
    } catch (IOException e) {
        System.out.println("Erro ao salvar os dados: " + e.getMessage());
    }
}

case 7 -> {
```

```
                // Recuperar dados
                System.out.println("Informe o prefixo dos arquivos:");
                String prefixoRecuperar = scanner.nextLine();
                try {
                    repoFisica.recuperar(prefixoRecuperar + ".fisica.bin");
                    repoJuridica.recuperar(prefixoRecuperar + ".juridica.bin");
                    System.out.println("Dados recuperados com sucesso.");
                } catch (IOException | ClassNotFoundException e) {
                    System.out.println("Erro ao recuperar os dados: " + e.getMessage());
                }
            }

            case 0 -> // Sair
                System.out.println("Saindo...");

            default -> System.out.println("Opção inválida!");
        }
    } while (opcao != 0);
  }
}
```

Pessoa.Java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 * to change this license Click
 * nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
 * template
```

```
 */
package model;
import java.io.Serializable;
/**
 *
 * @author rafaee
 */
public class Pessoa implements Serializable {
  private static final long serialVersionUID = 1L;
  private int id;
  public String nome;
  public Pessoa() {}
  public Pessoa(int id, String nome) {
    this.id = id;
    this.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;
  }
  public void exibir() {
    System.out.println("ID: " + id);
    System.out.println("Nome: " + nome);
  }
}
```

PessoaFisica.Java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;
import java.io.Serializable;

/**
 *
 * @author rafaee
 */

// Classe PessoaFisica
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;
}

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;
}
```

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

PessoaFisicaRepo.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;
import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

/**
 *
 * @author rafae
 */
// Classe PessoaFisicaRepo
public class PessoaFisicaRepo {
    private List<PessoaFisica> lista = new ArrayList<>();
```

```
// Método para inserir uma nova pessoa
public void inserir(PessoaFisica pessoa) {
    lista.add(pessoa);
}

// Método para alterar uma pessoa existente
public void alterar(PessoaFisica pessoa) {
    Optional<PessoaFisica> existente = lista.stream()
            .filter(p -> p.getId() == pessoa.getId())
            .findFirst();
    existente.ifPresent(p -> p.setNome(pessoa.getNome()));
}

// Método para excluir uma pessoa por id
public void excluir(int id) {
    lista.removeIf(p -> p.getId() == id);
}

// Método para obter uma pessoa por id
public PessoaFisica obter(int id) {
    return lista.stream()
            .filter(p -> p.getId() == id)
            .findFirst()
            .orElse(null);
}
```

```
    // Método para obter todas as pessoas
    public List<PessoaFisica> obterTodos() {
        return lista;
    }

    // Método para persistir os dados em arquivo
    public void persistir(String nomeArquivo) throws IOException {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(nomeArquivo))) {
            oos.writeObject(lista);
        }
    }

    // Método para recuperar os dados de arquivo
    public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(nomeArquivo))) {
            lista = (List<PessoaFisica>) ois.readObject();
        }
    }
}
```

PessoaJuridica.Java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change
this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
```

```
 */
package model;
import java.io.Serializable;

/**
 *
 * @author rafae
 */
public class PessoaJuridica extends Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;
    private String cnpj;

    // Construtor padrão
    public PessoaJuridica() {
    }

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

    // Getters e Setters
    public String getCnpj() {
        return cnpj;
    }
```

```
    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

    // Método exibir polimórfico
    @Override
    public void exibir() {
        super.exibir(); // Chama o método da classe pai
        System.out.println("CNPJ: " + cnpj);
    }
}
```

PessoaJuridicaRepo.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package model;
import java.io.*;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

/**
 *
```

```
 * @author rafae
 */
// Classe PessoaJuridicaRepo
public class PessoaJuridicaRepo {
    private List<PessoaJuridica> lista = new ArrayList<>();

    // Método para inserir uma nova pessoa jurídica
    public void inserir(PessoaJuridica pessoa) {
        lista.add(pessoa);
    }

    // Método para alterar uma pessoa jurídica existente
    public void alterar(PessoaJuridica pessoa) {
        Optional<PessoaJuridica> existente = lista.stream()
                .filter(p -> p.getId() == pessoa.getId())
                .findFirst();
        existente.ifPresent(p -> p.setNome(pessoa.getNome()));
    }

    // Método para excluir uma pessoa jurídica por id
    public void excluir(int id) {
        lista.removeIf(p -> p.getId() == id);
    }

    // Método para obter uma pessoa jurídica por id
    public PessoaJuridica obter(int id) {
```

```
        return lista.stream()
                .filter(p -> p.getId() == id)
                .findFirst()
                .orElse(null);
    }

    // Método para obter todas as pessoas jurídicas
    public List<PessoaJuridica> obterTodos() {
        return lista;
    }

    // Método para persistir os dados em arquivo
    public void persistir(String nomeArquivo) throws IOException {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(nomeArquivo))) {
            oos.writeObject(lista);
        }
    }

    // Método para recuperar os dados de arquivo
    public void recuperar(String nomeArquivo) throws IOException, ClassNotFoundException {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(nomeArquivo))) {
            lista = (ArrayList<PessoaJuridica>) ois.readObject();
        }
    }
}
```