Relatório Acadêmico

Universidade - Estácio de Sá

Campus Ribeirao Preto

Curso: Desenvolvimento Full Stack

Disciplina: Iniciando o caminho pelo java

Integrantes:

- Luan Guilherme Zufi de Lima

Título da Prática

Cadastro com Programação Orientada a Objetos e Persistência de Dados em Arquivos

Objetivo da Prática

Desenvolver um projeto Java utilizando **Programação Orientada a Objetos** com persistência de dados em arquivos. O sistema simula um cadastro de pessoas físicas e jurídicas, com repositórios específicos para cada tipo e com utilização de serialização para salvar e recuperar os dados.

Códigos Utilizados

Main.java

```java
package Main;

import cadastro.*;

public class Main {
    public static void main(String[] args) {
        try {
            PessoaFisicaRepo repo1 = new PessoaFisicaRepo();
            repo1.inserir(new PessoaFisica(1, "Ana", "11111111111", 25));
            repo1.inserir(new PessoaFisica(2, "Carlos", "22222222222", 52));
            repo1.persistir("pessoasFisicas.dat");

            PessoaFisicaRepo repo2 = new PessoaFisicaRepo();
            repo2.recuperar("pessoasFisicas.dat");

            System.out.println("Dados de Pessoa Física Recuperados:");
            for (PessoaFisica pf : repo2.obterTodos()) {
                pf.exibir();
            }

            PessoaJuridicaRepo repo3 = new PessoaJuridicaRepo();
            repo3.inserir(new PessoaJuridica(3, "XPTO S/A", "33333333333333"));
            repo3.inserir(new PessoaJuridica(4, "XPTO Soluções", "44444444444444
            repo3.persistir("pessoasJuridicas.dat");

            PessoaJuridicaRepo repo4 = new PessoaJuridicaRepo();
            repo4.recuperar("pessoasJuridicas.dat");

            System.out.println("\nDados de Pessoa Jurídica Recuperados:");
            for (PessoaJuridica pj : repo4.obterTodos()) {
                pj.exibir();
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Pessoa.java

```java
package cadastro;

import java.io.Serializable;

public abstract class Pessoa implements Serializable {
    protected int id;
    protected String nome;

    public Pessoa(int id, String nome) {
        this.id = id;
        this.nome = nome;
    }

    public abstract void exibir();
}
```

Pessoafisica.java

```java
package cadastro;

public class PessoaFisica extends Pessoa {
    private String cpf;
    private int idade;

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

    @Override
    public void exibir() {
        System.out.println("Pessoa Fisica -> ID: " + id + ", Nome: " + nome + ", CPF: " + cpf + ", Idade: " + idade);
    }
}
```

PessoaFisicaRepo.java

```java
package cadastro;

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

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

    public void inserir(PessoaFisica pf) {
        lista.add(pf);
    }

    public void persistir(String filename) throws IOException {
        FileOutputStream fos = new FileOutputStream(filename);
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(lista);
        oos.close();
    }

    public void recuperar(String filename) throws IOException, ClassNotFoundException {
        FileInputStream fis = new FileInputStream(filename);
        ObjectInputStream ois = new ObjectInputStream(fis);
        lista = (ArrayList<PessoaFisica>) ois.readObject();
        ois.close();
    }

    public ArrayList<PessoaFisica> obterTodos() {
        return lista;
    }
}
```

PessoaJuridica.java

```java
package cadastro;

public class PessoaJuridica extends Pessoa {
    private String cnpj;

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

    @Override
    public void exibir() {
        System.out.println("Pessoa Jurídica -> ID: " + id + ", Nome: " + nome + ", CNPJ: " + cnpj);
    }
}
```

PessoaJuridicaRepo.java

```java
public class PessoaJuridicaRepo {
    private ArrayList<PessoaJuridica> lista = new ArrayList<>();

    public void inserir(PessoaJuridica pj) {
        lista.add(pj);
    }

    public void persistir(String filename) throws IOException {
        FileOutputStream fos = new FileOutputStream(filename);
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(lista);
        oos.close();
    }

    public void recuperar(String filename) throws IOException, ClassNotFoundException {
        FileInputStream fis = new FileInputStream(filename);
        ObjectInputStream ois = new ObjectInputStream(fis);
        lista = (ArrayList<PessoaJuridica>) ois.readObject();
        ois.close();
    }

    public ArrayList<PessoaJuridica> obterTodos() {
        return lista;
    }
}
```

Resultado da Execução

```
Dados de Pessoa Física Recuperados:
Pessoa Física -> ID: 1, Nome: Ana, CPF: 11111111111, Idade: 25
Pessoa Física -> ID: 2, Nome: Carlos, CPF: 22222222222, Idade: 52

Dados de Pessoa Jurídica Recuperados:
Pessoa Jurídica -> ID: 3, Nome: XPTO S/A, CNPJ: 33333333333333
Pessoa Jurídica -> ID: 4, Nome: XPTO Soluções, CNPJ: 44444444444444
```

Análise e Conclusão

O que são elementos estáticos e por que o método main é static?

Elementos estáticos pertencem à classe e não aos objetos instanciados. O método main é *static* porque é o ponto de entrada da aplicação e precisa ser acessado pela JVM sem criar uma instância da classe.

Para que serve a classe Scanner?

A classe Scanner é utilizada para capturar entrada de dados do usuário, seja via teclado, arquivos ou outras fontes. É útil para tornar o programa interativo.

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

As classes de repositório ajudaram a organizar o código separando a lógica de armazenamento da lógica de negócio. Isso facilitou a manutenção, a leitura e a reutilização do código, além de seguir boas práticas de programação orientada a objetos.

Repositório no GitHub

LuanGZ/**Missao_P_1**

missão prática do 1º nível de conhecimento.

1 Contributor

0 Issues

0 Stars

0 Forks