Campus: POLO SAGUAÇU - JOINVILLE - SC

Curso: DESENVOLVIMENTO FULL STACK

Disciplina: Iniciando o Caminho Pelo Java

Turma: 9001

Semestre: 1° Semestre (2024)

Aluno: Jederson Borges de Oliveira

Link: https://github.com/JedersonBorges/CadastroPOO

Iniciando o caminho pelo Java

Objetivos da prática

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.**

Códigos utilizados

Classe Pessoa:

```java
package model;

import java.io.Serializable;

public class Pessoa implements Serializable {
    private int id;
    private 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 + ", Nome: " + nome);
    }
}
```

Classe PessoaFisica:

```java
package model;

import java.io.Serializable;

public class PessoaFisica extends Pessoa implements Serializable {
    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() {
        System.out.println("ID: " + getId());
        System.out.println("Nome: " + getNome());
        System.out.println("CPF: " + cpf);
        System.out.println("Idade: " + idade);
        System.out.println();
    }
}
```

Classe PessoaJuridica:

```
package model;

import java.io.Serializable;

public class PessoaJuridica extends Pessoa implements Serializable {
    private String cnpj;

    public PessoaJuridica() {}

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

    public String getCnpj() {
        return cnpj;
    }

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

    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CNPJ: " + cnpj);
        System.out.println(); // Linha em branco para separar as
pessoas
    }
}
```

Classe PessoaFisicaRepo:

```
package model;

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

public class PessoaFisicaRepo {
    private List<PessoaFisica> pessoasFisicas;

    public PessoaFisicaRepo() {
        this.pessoasFisicas = new ArrayList<>();
    }

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

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

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

    public PessoaFisica obter(int id) {
        for (PessoaFisica p : pessoasFisicas) {
            if (p.getId() == id) {
                return p;
            }
        }
        return null;
    }

    public List<PessoaFisica> obterTodos() {
        return new ArrayList<>(pessoasFisicas);
    }

    public void persistir(String arquivo) throws IOException {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(arquivo))) {
            oos.writeObject(pessoasFisicas);
        }
    }

    public void recuperar(String arquivo) throws IOException,
ClassNotFoundException {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(arquivo))) {
            pessoasFisicas = (List<PessoaFisica>) ois.readObject();
        }
```

```
    }
}
```

Classe PessoaFisicaRepo:

```
package model;

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

public class PessoaJuridicaRepo {
    private List<PessoaJuridica> pessoasJuridicas;

    public PessoaJuridicaRepo() {
        this.pessoasJuridicas = new ArrayList<>();
    }

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

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

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

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

    public List<PessoaJuridica> obterTodos() {
        return new ArrayList<>(pessoasJuridicas);
    }

    public void persistir(String arquivo) throws IOException {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(arquivo))) {
            oos.writeObject(pessoasJuridicas);
        }
    }

    public void recuperar(String arquivo) throws IOException,
ClassNotFoundException {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(arquivo))) {
            pessoasJuridicas = (List<PessoaJuridica>)
```

```
ois.readObject();
        }
    }
}
```

Main:

```
package model;

import java.io.IOException;

import java.util.Scanner;

public class Main {
    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("Menu:");
            System.out.println("1 - Incluir Pessoa");
            System.out.println("2 - Alterar Pessoa");
            System.out.println("3 - Excluir Pessoa");
            System.out.println("4 - Buscar pelo Id");
            System.out.println("5 - Exibir Todos");
            System.out.println("6 - Persistir Dados");
            System.out.println("7 - Recuperar Dados");
            System.out.println("0 - Finalizar Programa");
            System.out.print("Escolha uma opção: ");
            if(scanner.hasNextInt()){
                opcao = scanner.nextInt();
                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:
                        obter(scanner, repoFisica, repoJuridica);
                        break;
                    case 5:
                        obterTodos(scanner, repoFisica, repoJuridica);
                        break;
                    case 6:
                        salvar(scanner, repoFisica, repoJuridica);
                        break;
                    case 7:
                        recuperar(scanner, repoFisica, repoJuridica);
                        break;
                    case 0:
                        System.out.println("Finalizando execução.");
                        break;
                    default:
                        System.out.println("Opção inválida!");
                        break;
                }
```

```
            } else {
                System.out.println("Opção inválida. Por favor, escolha
uma opção válida.");
                scanner.next();
                opcao = -1;
            }
        } while (opcao != 0);
    }

    private static void incluir(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        String tipo = scanner.next().toUpperCase();
        scanner.nextLine();

        if (!tipo.equals("F") && !tipo.equals("J")) {
            System.out.println("Opção inválida!");
            return;
        }

        if (tipo.equals("F")) {
            System.out.print("Nome: ");
            String nome = scanner.nextLine();
            System.out.print("CPF: ");
            String cpf = scanner.nextLine();
            System.out.print("Idade: ");
            if (scanner.hasNextInt()) {
                int idade = scanner.nextInt();
                scanner.nextLine();
                PessoaFisica pessoa = new
PessoaFisica(repoFisica.obterTodos().size() + 1, nome, cpf, idade);
                repoFisica.inserir(pessoa);
            } else {
                System.out.println("Por favor, insira um número válido
para a idade.");
                scanner.nextLine();
            }
        } else if (tipo.equals("J")) {
            System.out.print("Nome: ");
            String nome = scanner.nextLine();
            System.out.print("CNPJ: ");
            String cnpj = scanner.nextLine();
            PessoaJuridica pessoa = new
PessoaJuridica(repoJuridica.obterTodos().size() + 1, nome, cnpj);
            repoJuridica.inserir(pessoa);
        }
    }



    private static void alterar(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        String tipo = scanner.next().toUpperCase();
        scanner.nextLine();

        System.out.print("ID da pessoa a ser alterada: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        if (tipo.equals("F")) {
```

```
            PessoaFisica pessoa = repoFisica.obter(id);
            if (pessoa != null) {
                System.out.print("Novo nome: ");
                String nome = scanner.nextLine();
                System.out.print("Novo CPF: ");
                String cpf = scanner.nextLine();
                System.out.print("Nova idade: ");
                if (scanner.hasNextInt()) {
                    int idade = scanner.nextInt();
                    scanner.nextLine();
                    pessoa.setNome(nome);
                    pessoa.setCpf(cpf);
                    pessoa.setIdade(idade);
                    repoFisica.alterar(pessoa);
                } else {
                    System.out.println("Por favor, insira um número
válido para a idade.");
                    scanner.nextLine();
                }
            } else {
                System.out.println("Pessoa física não encontrada com o
ID fornecido.");
            }
        } else if (tipo.equals("J")) {
            PessoaJuridica pessoa = repoJuridica.obter(id);
            if (pessoa != null) {
                System.out.print("Novo nome: ");
                String nome = scanner.nextLine();
                System.out.print("Novo CNPJ: ");
                String cnpj = scanner.nextLine();
                pessoa.setNome(nome);
                pessoa.setCnpj(cnpj);
                repoJuridica.alterar(pessoa);
            } else {
                System.out.println("Pessoa jurídica não encontrada com
o ID fornecido.");
            }
        } else {
            System.out.println("Opção inválida.");
        }
    }

    private static void excluir(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        String tipo = scanner.next().toUpperCase();
        scanner.nextLine();

        System.out.print("ID da pessoa a ser excluída: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        if (tipo.equals("F")) {
            if (repoFisica.obter(id) != null) {
                repoFisica.excluir(id);
                System.out.println("Pessoa física excluída com
sucesso.");
            } else {
                System.out.println("Pessoa física não encontrada com o
ID fornecido.");
            }
```

```
        } else if (tipo.equals("J")) {
            if (repoJuridica.obter(id) != null) {
                repoJuridica.excluir(id);
                System.out.println("Pessoa jurídica excluída com
sucesso.");
            } else {
                System.out.println("Pessoa jurídica não encontrada com
o ID fornecido.");
            }
        } else {
            System.out.println("Opção inválida.");
        }
    }


    private static void obter(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        String tipo = scanner.next().toUpperCase();
        scanner.nextLine();

        System.out.print("Digite o ID da pessoa: ");
        int id = scanner.nextInt();
        scanner.nextLine();

        if (tipo.equals("F")) {
            PessoaFisica pessoa = repoFisica.obter(id);
            if (pessoa != null) {
                pessoa.exibir();
            } else {
                System.out.println("Pessoa física não encontrada com o
ID fornecido.");
            }
        } else if (tipo.equals("J")) {
            PessoaJuridica pessoa = repoJuridica.obter(id);
            if (pessoa != null) {
                pessoa.exibir();
            } else {
                System.out.println("Pessoa jurídica não encontrada com
o ID fornecido.");
            }
        } else {
            System.out.println("Opção inválida.");
        }
    }


    private static void obterTodos(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        String tipo = scanner.next().toUpperCase();
        scanner.nextLine();

        if (tipo.equals("F")) {
            for (PessoaFisica pessoa : repoFisica.obterTodos()) {
                pessoa.exibir();
            }
        } else if (tipo.equals("J")) {
            for (PessoaJuridica pessoa : repoJuridica.obterTodos()) {
                pessoa.exibir();
            }
```

```
        } else {
            System.out.println("Opção inválida.");
        }
    }


    private static void salvar(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.print("Digite o prefixo para os arquivos: ");
        String prefixo = scanner.next();
        try {
            repoFisica.persistir(prefixo + ".fisica.bin");
            repoJuridica.persistir(prefixo + ".juridica.bin");
            System.out.println("Dados salvos com sucesso.");
        } catch (IOException e) {
            System.out.println("Erro ao salvar os dados: " +
e.getMessage());
        }
    }

    private static void recuperar(Scanner scanner, PessoaFisicaRepo
repoFisica, PessoaJuridicaRepo repoJuridica) {
        System.out.print("Digite o prefixo dos arquivos a serem
recuperados: ");
        String prefixo = scanner.next();
        try {
            repoFisica.recuperar(prefixo + ".fisica.bin");
            repoJuridica.recuperar(prefixo + ".juridica.bin");
            System.out.println("Dados recuperados com sucesso.");
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Erro ao recuperar os dados: " +
e.getMessage());
        }
    }
}
```

Resultado da execução dos códigos:

1 – Incluir Pessoa

(Física)

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 1
F - Pessoa Fisica | J - Pessoa Juridica
f
Nome: Max Verstappen
CPF: 123456
Idade: 26
```

(Juridica)

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 1
F - Pessoa Fisica | J - Pessoa Juridica
j
Nome: Red Bull
CNPJ: 123123
```

2 – Alterar pessoa

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 2
F - Pessoa Fisica | J - Pessoa Juridica
j
ID da pessoa a ser alterada: 1
Novo nome: Red Bull GmbH
Novo CNPJ: 111222
```

3 – Excluir pessoa

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 3
F - Pessoa Fisica | J - Pessoa Juridica
j
ID da pessoa a ser excluída: 1
Pessoa jurídica excluída com sucesso.
```

4 – Buscar pelo ID

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 4
F - Pessoa Fisica | J - Pessoa Juridica
f
Digite o ID da pessoa: 2
ID: 2
Nome: Sergio Perez
CPF: 123457
Idade: 33
```

5 – Exibir todos

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 5
F - Pessoa Fisica | J - Pessoa Juridica
f
ID: 1
Nome: Max Verstappen
CPF: 123456
Idade: 26

ID: 2
Nome: Sergio Perez
CPF: 123457
Idade: 33
```

6 – Persistir dados

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 6
Digite o prefixo para os arquivos: arquivo01
Dados salvos com sucesso.
```

7 – Recuperar dados

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 7
Digite o prefixo dos arquivos a serem recuperados: arquivo01
Dados recuperados com sucesso.
```

0 – Finalizar programa

```
Menu:
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
6 - Persistir Dados
7 - Recuperar Dados
0 - Finalizar Programa
Escolha uma opção: 0
Finalizando execução.

Process finished with exit code 0
```

Análise e Conclusão

Quais as vantagens e desvantagens do uso de herança?

A herança permite a reutilização de código, facilitando a manutenção do sistema, em contrapartida pode deixar o código mais complexo e menos flexível

Por que a interface Serializable é necessária ao efetuar persistência em arquivos binários?

A interface Serializable é necessária para permitir que os objetos sejam convertidos em formatos binários e reconstruídos posteriormente

Como o paradigma funcional é utilizado pela API stream no Java?

A API Stream no Java utiliza o paradigma funcional para manipular coleções de dados de forma mais eficiente e clara.

Quando trabalhamos com Java, qual padrão de desenvolvimento é adotado na persistência de dados em arquivos?

É adotado o uso de serialização, onde os objetos são convertidos em bytes e armazenados em arquivos binários