



Estácio

Universidade Estácio de Sá

Curso Desenvolvimento Full Stack - Turma 9001
Disciplina: RPG0014 - Nível 1: Iniciando o Caminho Pelo
Java

Semestre Letivo: 2024.4

Repositorio Git: [TBD]

[SASHA CARDOSO]

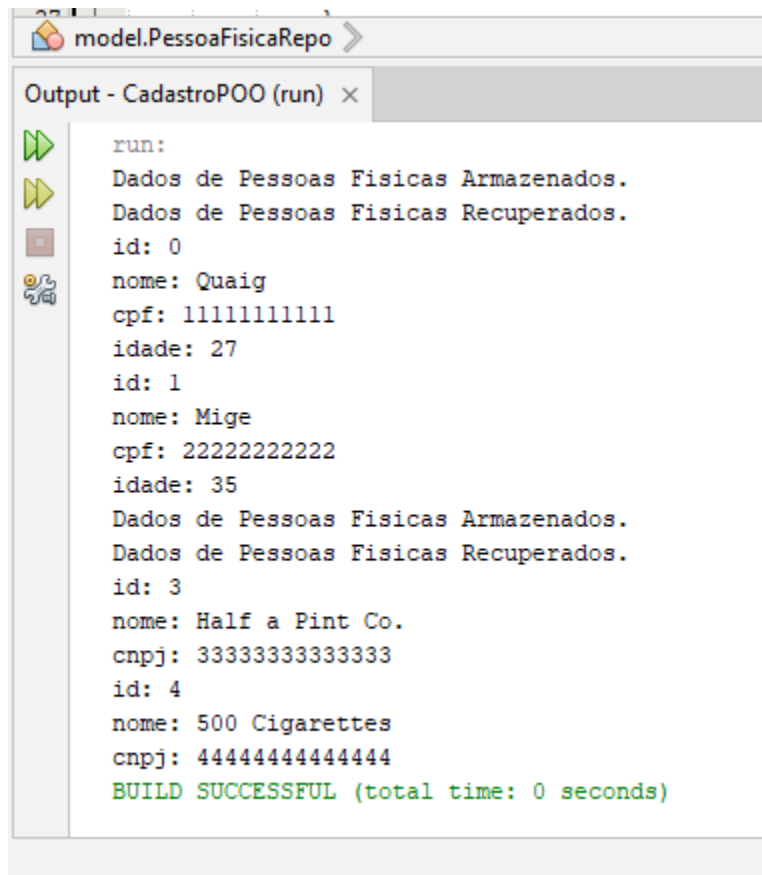
Missão Prática | Nível 1 | Mundo 3

Objetivo: Fazer um programa em Java que permite cadastro e armazenamento de dados.

1º Procedimento

Todos os códigos estão disponíveis no repositório Github linkado acima.

Resultados da execução:



```
run:
Dados de Pessoas Fisicas Armazenados.
Dados de Pessoas Fisicas Recuperados.
id: 0
nome: Quaig
cpf: 11111111111
idade: 27
id: 1
nome: Mige
cpf: 22222222222
idade: 35
Dados de Pessoas Fisicas Armazenados.
Dados de Pessoas Fisicas Recuperados.
id: 3
nome: Half a Pint Co.
cnpj: 33333333333333
id: 4
nome: 500 Cigarettes
cnpj: 44444444444444
BUILD SUCCESSFUL (total time: 0 seconds)
```

```

package model;

public class CadastroPOO {

    public static PessoaFisicaRepo repo1;
    public static PessoaFisicaRepo repo2;
    public static PessoaJuridicaRepo repo3;
    public static PessoaJuridicaRepo repo4;

    public static void main(String[] args) {
        repo1 = new PessoaFisicaRepo();
        repo1.inserir(new PessoaFisica(0, "Quaig", "1111111111", 27));
        repo1.inserir(new PessoaFisica(1, "Mige", "2222222222", 35));
        try {
            repo1.persistir("wawaFisica");
        } catch (Exception e) {
            System.out.println(e.toString());
        }

        repo2 = new PessoaFisicaRepo();
        try {
            repo2.recuperar("wawaFisica");
        } catch (Exception e) {
            System.out.println(e.toString());
        }
        for (PessoaFisica p : repo2.obterTodos()) {
            p.exibir();
        }

        repo3 = new PessoaJuridicaRepo();
        repo3.inserir(new PessoaJuridica(3, "Half a Pint Co.", "3333333333333333"));
        repo3.inserir(new PessoaJuridica(4, "500 Cigarettes", "4444444444444444"));
        try {
            repo3.persistir("wawaJuridica");
        } catch (Exception e) {
            System.out.println(e.toString());
        }

        repo4 = new PessoaJuridicaRepo();
        try {
            repo4.recuperar("wawaJuridica");
        } catch (Exception e) {
            System.out.println(e.toString());
        }
        for (PessoaJuridica p : repo4.obterTodos()) {
            p.exibir();
        }
    }
}

```

Análise e Conclusão

1. Quais as vantagens e desvantagens do uso de herança?

- Uma das vantagens de herança é que possibilita o design de funções mais genéricas. Essas funções podem ser usadas de forma mais ampla contanto que as classes compartilhem o mesmo parente. Além disso, herança permite reutilizar o mesmo código, ajudando na organização do código.

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

- Usar *Serializable* marca o objeto como *serializable* e é necessário para que você possa salvar um objeto em arquivo, pois permite que o arquivo seja convertido em *bytestream*. Sem implementar *Serializable*, isto não é possível e causa *java.io.NotSerializableException*.

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

- O paradigma funcional é usado de maneira em que funções podem ser passadas como parâmetro em métodos como *filter* e *mapToInt* para processar / iterar uma *Collection* de acordo com as especificações do programador.

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

- É adotado *Serialization* usando *ObjectOutputStream* e *ObjectInputStream*.