



Jonathan Domingos Carneiro da Silva
matricula: 2023.04.12744-1
3274 POLO CENTRO - ITAITINGA – CE

Nível 3: Back-end Sem Banco Não Tem

– DESENVOLVIMENTO FULL STACK 2023.1 – 3º Semestre – 2024.2

Criação de aplicativo Java, com acesso ao banco de dados SQL Server através do middleware JDBC.

Todos os códigos solicitados neste roteiro de aula:

CadastroPOO

CadastroPOO.java

```
/*  
 * 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  
 */  
  
package cadastrapoo;  
  
import java.io.BufferedReader;  
import java.io.InputStreamReader;  
import model.PessoaFisica; import  
model.PessoaFisicaRepo; import  
model.PessoaJuridica;  
import model.PessoaJuridicaRepo;
```



```
/**
 *
 * @author Usuario
 */
public class CadastroPOO {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) throws Exception {

        BufferedReader reader = new BufferedReader(new
            InputStreamReader(System.in));

        String opcao = "";

        PessoaFisicaRepo repo_fisica = new PessoaFisicaRepo(); PessoaJuridicaRepo
        repo_juridica = new PessoaJuridicaRepo();

        while (!"8".equals(opcao)) { 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("8 - Finalizar Programa");
```



Estácio

```
System.out.println(".....");
```

```
// Reading data using readLine
```

```
opcao = reader.readLine();
```

```
// Printing the read line
```

```
System.out.println(opcao);
```

```
String pessoa = "";
```

```
switch (opcao) {
```

```
    case "1" -> {
```

```
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");pessoa  
        = reader.readLine();
```

```
        switch (pessoa) {
```

```
            case "F" -> {
```

```
                PessoaFisica p = new PessoaFisica();
```

```
                System.out.println("Digite o CPF do usuário");
```

```
                p.setCpf(reader.readLine());
```

```
                System.out.println("Digite a Idade do usuário");
```

```
                p.setIdade(Integer.parseInt(reader.readLine()));
```

```
                System.out.println("Digite o Id do usuário");
```

```
                p.setId(Integer.parseInt(reader.readLine()));
```

```
                System.out.println("Digite o Nome do usuário");
```

```
                p.setNome(reader.readLine());
```

```
                repo_fisica.inserir(p);
```

```
            }
```

```
        case "J" -> {
```

```
            PessoaJuridica j = new PessoaJuridica();
```



Estácio

```
System.out.println("Digite o CNPJ do usuário");
j.setCnpj(reader.readLine());

System.out.println("Digite o Id do usuário");
j.setId(Integer.parseInt(reader.readLine()));

System.out.println("Digite o Nome do usuário");
j.setNome(reader.readLine());

repo_juridica.inserir(j);
    }
}
}

case "2" -> {

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");pessoa
    = reader.readLine();

    switch (pessoa) {

        case "F" -> {

            System.out.println("Digite o Id que deseja alterar: "); int
            id = Integer.parseInt(reader.readLine()); PessoaFisica pf =
            repo_fisica.obter(id); System.out.println("CPF Antigo: " +
            pf.getCpf()); System.out.println("Digite o novo CPF do
            usuário"); pf.setCpf(reader.readLine());

            System.out.println("Idade Antigo: " + pf.getIdade());
            System.out.println("Digite a nova Idade do usuário");
            pf.setIdade(Integer.parseInt(reader.readLine()));

            System.out.println("Nome Antigo: " + pf.getNome());
            System.out.println("Digite o novo Nome do usuário");
            pf.setNome(reader.readLine()); repo_fisica.alterar(pf);
            repo_fisica.obter(pf.getId());
```



```
}  
case "J" -> {  
    System.out.println("Digite o Id que deseja alterar: "); int  
    id = Integer.parseInt(reader.readLine()); PessoaJuridica pj  
    = repo_juridica.obter(id); System.out.println("CNPJ  
    Antigo: " + pj.getCnpj()); System.out.println("Digite o  
    novo CNPJ do usuário");pj.setCnpj(reader.readLine());  
    System.out.println("Nome Antigo: " + pj.getNome());  
    System.out.println("Digite o novo Nome do usuário");  
    pj.setNome(reader.readLine()); repo_juridica.alterar(pj);  
    repo_juridica.obter(pj.getId());  
}  
}  
}  
  
case "3" -> {  
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");pessoa  
    = reader.readLine();  
    switch (pessoa) {  
        case "F" -> {  
            System.out.println("Digite o Id do usuário");int  
            id = Integer.parseInt(reader.readLine());  
            repo_fisica.excluir(id);  
        }  
        case "J" -> {  
            System.out.println("Digite o Id do usuário");
```



Estácio

```
int id = Integer.parseInt(reader.readLine());
repo_juridica.excluir(id);
    }
}
}

case "4" -> {
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");pessoa
    = reader.readLine();
    switch (pessoa) {
        case "F" -> {
            System.out.println("Digite o Id do usuário"); int
            id = Integer.parseInt(reader.readLine());
            PessoaFisica pf = repo_fisica.obter(id);
            System.out.println("Id: " + pf.getId());
            System.out.println("CPF: " + pf.getCpf());
            System.out.println("Idade: " + pf.getIdade());
            System.out.println("Nome: " + pf.getNome());

        }
        case "J" -> {
            System.out.println("Digite o Id do usuário"); int
            id = Integer.parseInt(reader.readLine());
            PessoaJuridica pj = repo_juridica.obter(id);
            System.out.println("Id: " + pj.getId());
            System.out.println("CNPJ: " + pj.getCnpj());
            System.out.println("Nome: " + pj.getNome());

        }
    }
}
```



```
case "5" -> {

    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");pessoa
    = reader.readLine();

    switch (pessoa) {

        case "F" -> {

            for (int i = 0; i < repo_fisica.obterTodos().size(); i++) { System.out.println("id:" +
                repo_fisica.obterTodos().get(i).getId()); System.out.println("Nome:" +
                repo_fisica.obterTodos().get(i).getNome());System.out.println("CPF:" +
                repo_fisica.obterTodos().get(i).getCpf());

            }

        }

        case "J" -> {

            for(int i = 0; i < repo_juridica.obterTodos().size(); i++){ System.out.println("id:" +
                repo_juridica.obterTodos().get(i).getId());

                System.out.println("Nome:" + repo_juridica.obterTodos().get(i).getNome());

                System.out.println("CNPJ:" + repo_juridica.obterTodos().get(i).getCnpj());

            }

        }

    }

}

case "6" -> {

    System.out.println("Qual o prefixo dos arquivos: ");String
    arquivoP = reader.readLine();

    repo_fisica.persistir(arquivoP);

    repo_juridica.persistir(arquivoP);

}

case "7" -> {
```



Estácio

```
System.out.println("Qual o prefixo dos arquivos: ");String
arquivoP = reader.readLine();
repo_fisica.recuperar(arquivoP);
repo_juridica.recuperar(arquivoP);

}

case "8" -> {
    System.out.println("Fechando o sistema");
}

default -> {
}

}
}
}

}
```

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 Usuario
 */
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;
    }

    /**
     * @return the id
     */
    public int getId() {
        return id;
    }

    /**
     * @param id the id to set
     */
    public void setId(int id) {
        this.id = id;
    }
}
```



```
}

/**
 * @return the nome
 */
public String getNome() {
    return nome;
}

/**
 * @param nome the nome to set
 */
public void setNome(String nome) {
    this.nome = nome;
}

public void exibir(){
    System.out.println("Classe pai");
}
}
```

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 Usuario
 */

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

    public PessoaFisica(){

    }

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

    /**
     * @return the cpf
     */
    public String getCpf() {
        return cpf;
    }

    /**
     * @param cpf the cpf to set
```



```
*/  
  
public void setCpf(String cpf) {  
    this.cpf = cpf;  
}  
  
/**  
 * @return the idade  
 */  
  
public int getIdade() {  
    return idade;  
}  
  
/**  
 * @param idade the idade to set  
 */  
  
public void setIdade(int idade) {  
    this.idade = idade;  
}  
  
@Override  
public void exibir() {  
    System.out.println("classe filha");  
}  
  
}
```



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.FileInputStream;  
import java.io.InputStream;  
import java.nio.charset.StandardCharsets;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.util.ArrayList;  
import java.util.List; import  
java.util.Scanner;  
  
/**  
 *  
 * @author Usuario  
 */  
  
public class PessoaFisicaRepo {  
  
  
    private final ArrayList<PessoaFisica> pessoaFi = new ArrayList<PessoaFisica>();  
  
  
    public void inserir(PessoaFisica pf) {  
        pessoaFi.add(pf);  
    }  
}
```



Estácio

```
public void alterar(PessoaFisica pf) {
    for (int i = 0; i < pessoaFi.size() - 1; i++) { if
        (pf.getId() == pessoaFi.get(i).getId()) {
            pessoaFi.get(i).setNome(pf.getNome());pessoaFi.get(i).setIdade(pf.getIdade());
            pessoaFi.get(i).setCpf(pf.getCpf());
        }
    }
}

public void excluir(int id) {
    pessoaFi.remove(obter(id));
}

public PessoaFisica obter(int id) {
    PessoaFisica p = new PessoaFisica();for
    (int i = 0; i < pessoaFi.size(); i++) {
        if (pessoaFi.get(i).getId() == id) {p
            = pessoaFi.get(i);
        }
    }

    return p;
}

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



Estácio

```
public void recuperar(String arquivoModel) throws Exception { try {
```

```
    // Declaring and initializing the string with
```

```
    // custom path of a file
```

```
    String path = "D:\\dev-teo\\" + arquivoModel + ".fisica.bin";
```

```
    // Creating an instance of InputStream
```

```
    InputStream is = new FileInputStream(path);
```

```
    // Try block to check for exceptions
```

```
    Scanner sc = new Scanner(is, StandardCharsets.UTF_8.name());
```

```
    // It holds true till there is single element
```

```
    // left in the object with usage of hasNext()
```

```
    // method
```

```
    while (sc.hasNextLine()) {
```

```
        String[] linha = sc.nextLine().split(";");
```

```
        PessoaFisica pessoaFisica = new PessoaFisica();
```

```
        pessoaFisica.setId(Integer.parseInt(linha[0])); pessoaFisica.setNome(linha[1]);
```

```
        pessoaFisica.setIdade(Integer.parseInt(linha[2].trim()));
```

```
        pessoaFisica.setCpf(linha[3]);
```

```
        this.pessoaFi.add(pessoaFisica);
```

```
    }
```

```
} catch (Exception e) {
```



Estácio

```
        throw new Exception("Exception message");
    }
}

public void persistir(String arquivoModel) throws Exception {

    try {
        Path path = Paths.get("D:\\dev-teo\\" + arquivoModel + ".fisica.bin");

        String banco = "";

        for (int i = 0; i < obterTodos().size(); i++) {
            banco = banco + obterTodos().get(i).getId() + "; "; banco =
            banco + obterTodos().get(i).getNome() + "; "; banco =
            banco + obterTodos().get(i).getCpf() + "; "; banco = banco +
            obterTodos().get(i).getIdade() + "; \n";

        }
        // Custom string as an input
        Files.writeString(path, banco,
            StandardCharsets.UTF_8);
    } catch (Exception e) {
        throw new Exception("Exception message");
    }
}
}
```

PessoaJuridica.java

/*



Estácio

* 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 Usuario
```

```
*/
```

```
public class PessoaJuridica extends Pessoa implements Serializable {private
```

```
    static final long serialVersionUID = 1L;
```

```
    private String cnpj;
```

```
    public PessoaJuridica(){
```

```
    }
```

```
    public PessoaJuridica(String cnpj, int id, String nome) {
```

```
        super(id, nome);
```

```
        this.cnpj = cnpj;
```

```
    }
```

```
/**
```

```
 * @return the cnpj
```

```
*/
```

```
public String getCnpj() {
```

```
    return cnpj;
```



```
}

/**
 * @param cnpj the cnpj to set
 */
public void setCnpj(String cnpj) {
    this.cnpj = cnpj;
}

@Override
public void exibir() {

}

}
```

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.FileInputStream;
import java.io.InputStream;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
```



```
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Scanner;

/**
 *
 * @author Usuario
 */
public class PessoaJuridicaRepo {

    private final ArrayList<PessoaJuridica> pessoaJu = new ArrayList<PessoaJuridica>();

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

    public void alterar(PessoaJuridica pj) {
        for (int i = 0; i < pessoaJu.size() - 1; i++) { if
            (pj.getId() == pessoaJu.get(i).getId()) {
                pessoaJu.get(i).setNome(pj.getNome());
                pessoaJu.get(i).setCnpj(pj.getCnpj());
                pessoaJu.get(i).setId(pj.getId());
            }
        }
    }

    public void excluir(int id) {
        pessoaJu.remove(obter(id));
    }
}
```



```
public PessoaJuridica obter(int id) {
    PessoaJuridica j = new PessoaJuridica();
    for (int i = 0; i < pessoaJu.size(); i++) {
        if (pessoaJu.get(i).getId() == id) {
            j = pessoaJu.get(i);
        }
    }
    return j;
}

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

public void recuperar(String arquivoModel) throws Exception {
    try {
        // Declaring and initializing the string with
        // custom path of a file
        String path = "D:\\dev-teo\\" + arquivoModel + ".juridica.bin";

        // Creating an instance of InputStream
        InputStream is = new FileInputStream(path);

        // Try block to check for exceptions
        Scanner sc = new Scanner(is, StandardCharsets.UTF_8.name());

        // It holds true till there is single element
        // left in the object with usage of hasNext()
    }
}
```



```
// method
while (sc.hasNextLine()) {

    // Printing the content of file
    String[] linha = sc.nextLine().split(";");

    PessoaJuridica pessoaJuridica = new PessoaJuridica();
    pessoaJuridica.setId(Integer.parseInt(linha[0]));
    pessoaJuridica.setNome(linha[1]); pessoaJuridica.setCnpj(linha[3]);

    this.pessoaJu.add(pessoaJuridica);
}

} catch (Exception e) {
    throw new Exception("Exception message");
}

}

public void persistir(String arquivoModel) throws Exception {try {
    Path path = Paths.get("D:\\dev-teo\\" + arquivoModel + ".juridica.bin");

    String banco = "";

    for (int i = 0; i < obterTodos().size(); i++) {
        banco = banco + obterTodos().get(i).getId() + "; "; banco =
        banco + obterTodos().get(i).getNome() + "; "; banco = banco
        + obterTodos().get(i).getCnpj() + "; \n";
    }
}
```



```
}  
  
// Custom string as an input  
  
Files.writeString(path, banco,  
    StandardCharsets.UTF_8);  
} catch (Exception e) {  
    throw new Exception("Exception message");  
}  
}  
}
```

Resultado do Código:

run:

1 - Incluir pessoa 2

- Alterar Pessoa3 -

Excluir Pessoa4 -

Buscar pelo ID5 -

Exibir Todos

6 - Persistir Dados

7 - Recuperar Dados 8 -

Finalizar Programa



Análise e Conclusão:

Quais as diferenças entre a persistência em arquivo e a persistência em banco de dados?

a escolha entre persistência em arquivo e persistência em banco de dados depende dos requisitos da sua aplicação, do volume de dados, das necessidades de consulta, da concorrência e de outros fatores. Em muitos casos, os bancos de dados são preferidos para aplicações empresariais e de grande escala, enquanto a persistência em arquivo pode ser mais adequada para tarefas simples ou aplicações menores.

Como o uso de operador lambda simplificou a impressão dos valores contidos nas entidades, nas versões mais recentes do Java?

O uso de operadores lambda no Java simplificou a impressão de valores em entidades nas versões mais recentes do Java, tornando o código mais conciso e legível. Agora, você pode usar expressões lambda com métodos de alto nível, como `forEach`, para realizar a ação desejada em cada elemento de uma coleção, eliminando a necessidade de escrever loops explícitos. Isso torna o código mais expressivo e moderno.

Por que métodos acionados diretamente pelo método `main`, sem o uso de um objeto, precisam ser marcados como `static`?

os métodos `main` são declarados como `static` para permitir que sejam chamados diretamente pela JVM no contexto da classe, sem a necessidade de criar uma instância da classe. Isso é fundamental para o funcionamento do ponto de entrada do programa Java.

Essa análise aborda três conceitos importantes em desenvolvimento de software: persistência de dados, uso de operadores lambda em Java e a necessidade de métodos `main` serem estáticos. Vou resumir cada ponto:

1. Persistência em arquivo versus persistência em banco de dados:

- A persistência em arquivo envolve armazenar dados diretamente em arquivos no sistema de arquivos do computador.
- Já a persistência em banco de dados envolve armazenar dados em um sistema gerenciado de armazenamento, como um banco de dados relacional ou não relacional.
- A escolha entre eles depende dos requisitos da aplicação, volume de dados, necessidades de consulta, concorrência, entre outros fatores.
- Bancos de dados são preferidos para aplicações empresariais e de grande escala, enquanto a persistência em arquivo pode ser mais adequada para tarefas simples ou aplicações menores.

2. Uso de operadores lambda em Java:

- Os operadores lambda foram introduzidos no Java para simplificar a escrita de código utilizando expressões funcionais.



Estácio

- Eles simplificam a impressão de valores em entidades, tornando o código mais conciso e legível.
- Com o uso de expressões lambda, métodos de alto nível como `forEach` podem ser utilizados para realizar ações em coleções, eliminando a necessidade de escrever loops explícitos.

3. Métodos main marcados como estáticos:

- Os métodos main são o ponto de entrada de um programa Java, e são invocados diretamente pela JVM.
- Eles precisam ser marcados como estáticos para que possam ser chamados sem a necessidade de instanciar a classe.
- Isso permite que a JVM execute o programa sem criar uma instância da classe que contém o método main.

Cada um desses conceitos é fundamental para entender aspectos importantes do desenvolvimento de software e do funcionamento interno da linguagem Java.