Estácio

Campus: Dorival Caymmi

Nome: Filippe Markouizos Duarte - 202308599615

Curso: Desenvolvimento Full-Stack

Disciplina: Back-end sem Banco não tem

Turma: 2023.1 **3º Semestre**

Título:

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

Objetivo:

Esse trabalho tem como objetivo implementar um sistema de cadastro de clientes em modo texto utilizando a linguagem Java, com persistência dos dados em arquivos. O sistema permitirá o armazenamento, consulta, atualização e remoção de registros de clientes, proporcionando uma experiência prática no desenvolvimento de aplicações com manipulação de dados e persistência.

Análise e Conclusão

- **Diferenças entre a persistência em arquivo e a persistência em banco de dados**

  - A persistência em arquivo e a persistência em banco de dados têm abordagens distintas para o armazenamento de informações. A persistência em arquivo utiliza o sistema de arquivos para gravar dados em formatos como texto ou binário. É uma solução mais simples e direta, mas não possui suporte nativo para consultas complexas, controle de concorrência ou transações, tornando-a menos eficiente para sistemas com grandes volumes de dados ou operações frequentes de leitura e escrita.

Já a persistência em banco de dados envolve o uso de sistemas gerenciadores de banco de dados (SGBDs), que permitem a organização estruturada dos dados em tabelas, além de fornecer mecanismos para realizar consultas avançadas (SQL), integridade dos dados, transações e escalabilidade. O SGBD oferece suporte a múltiplos usuários simultâneos, além de permitir a recuperação e manipulação de dados de maneira mais eficiente e segura.

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

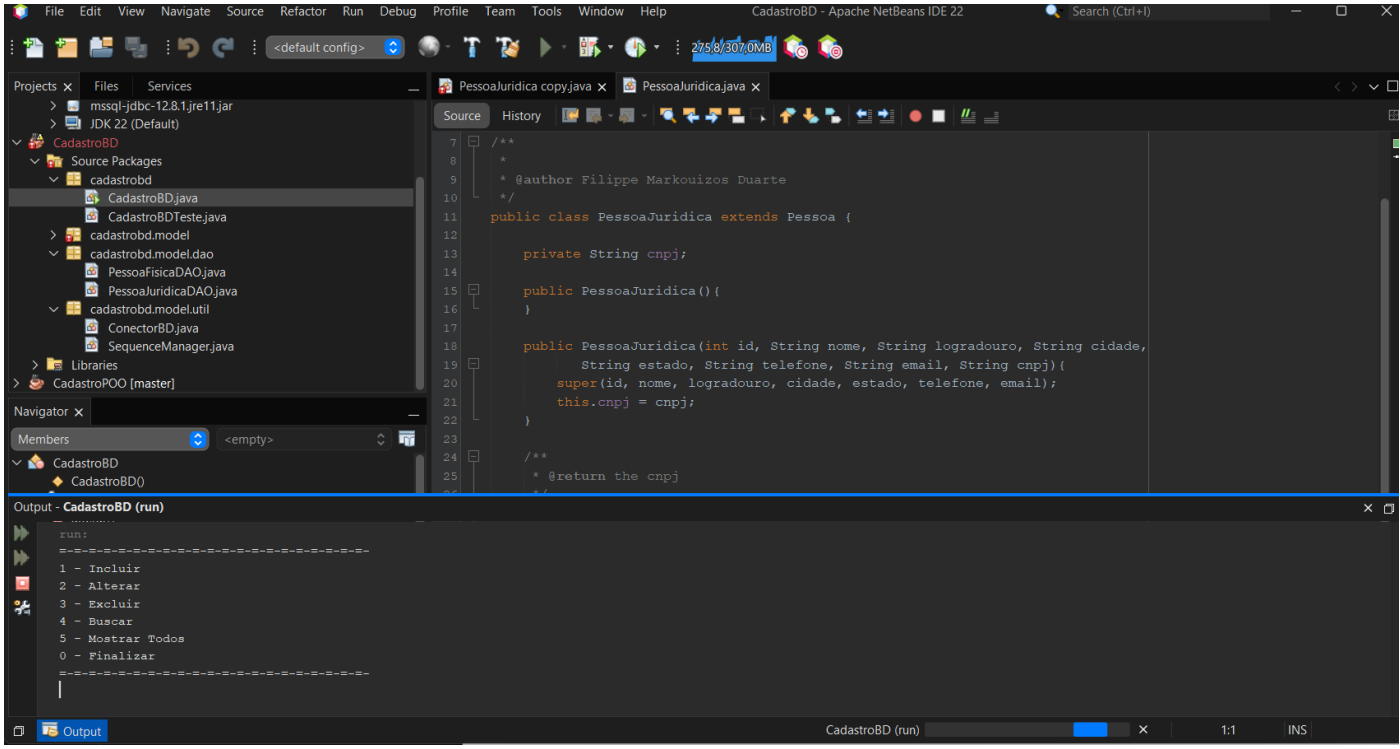
– Nas versões mais recentes do Java, o uso de expressões lambda simplificou bastante a manipulação e processamento de coleções de dados. No contexto da impressão de valores contidos nas entidades, o uso de lambdas permite uma sintaxe mais concisa e legível.

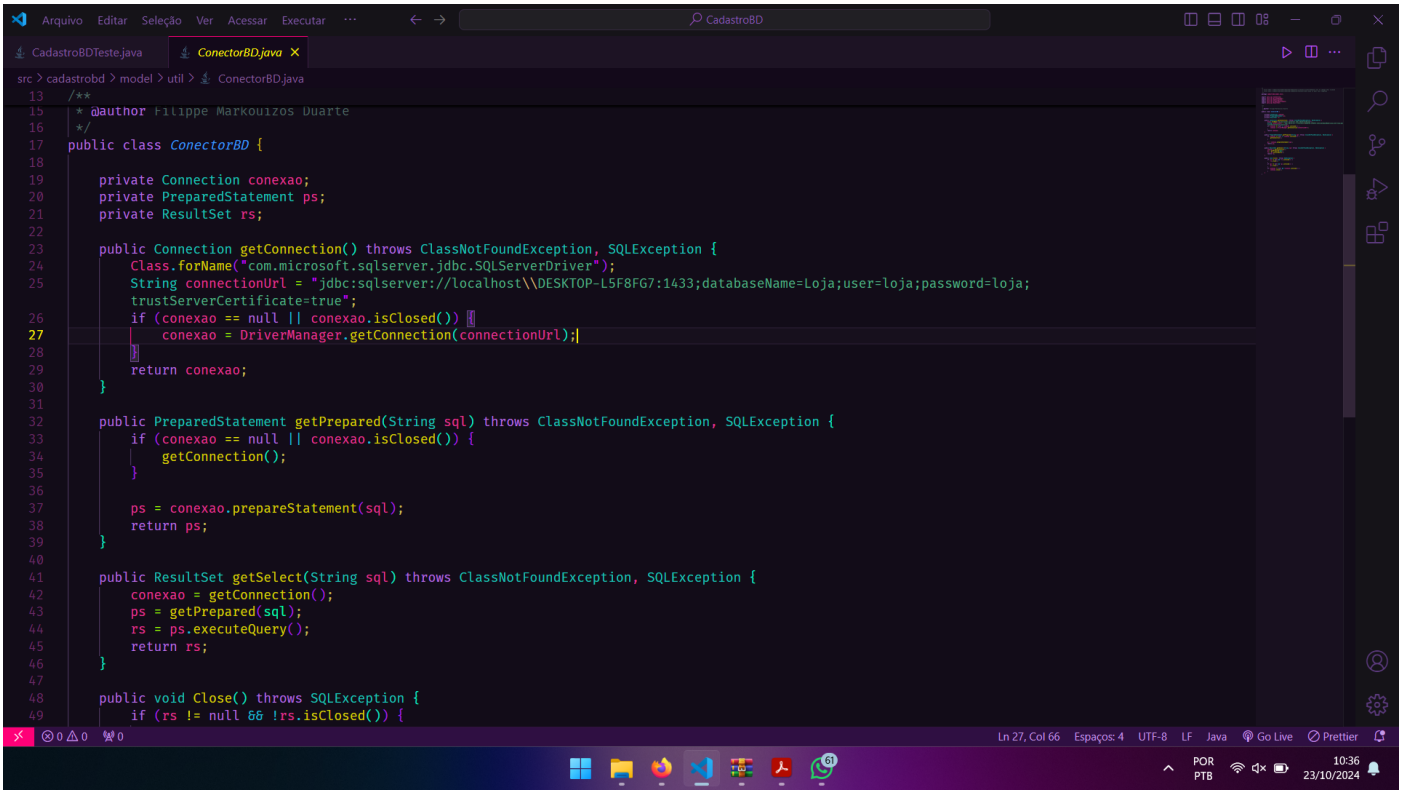
Por exemplo, ao invés de utilizar loops explícitos para iterar sobre uma coleção e imprimir os valores, uma expressão lambda pode ser passada diretamente para métodos como forEach() de uma List, reduzindo a quantidade de código boilerplate e facilitando o entendimento. Isso melhora a legibilidade do código e torna o processo mais declarativo.

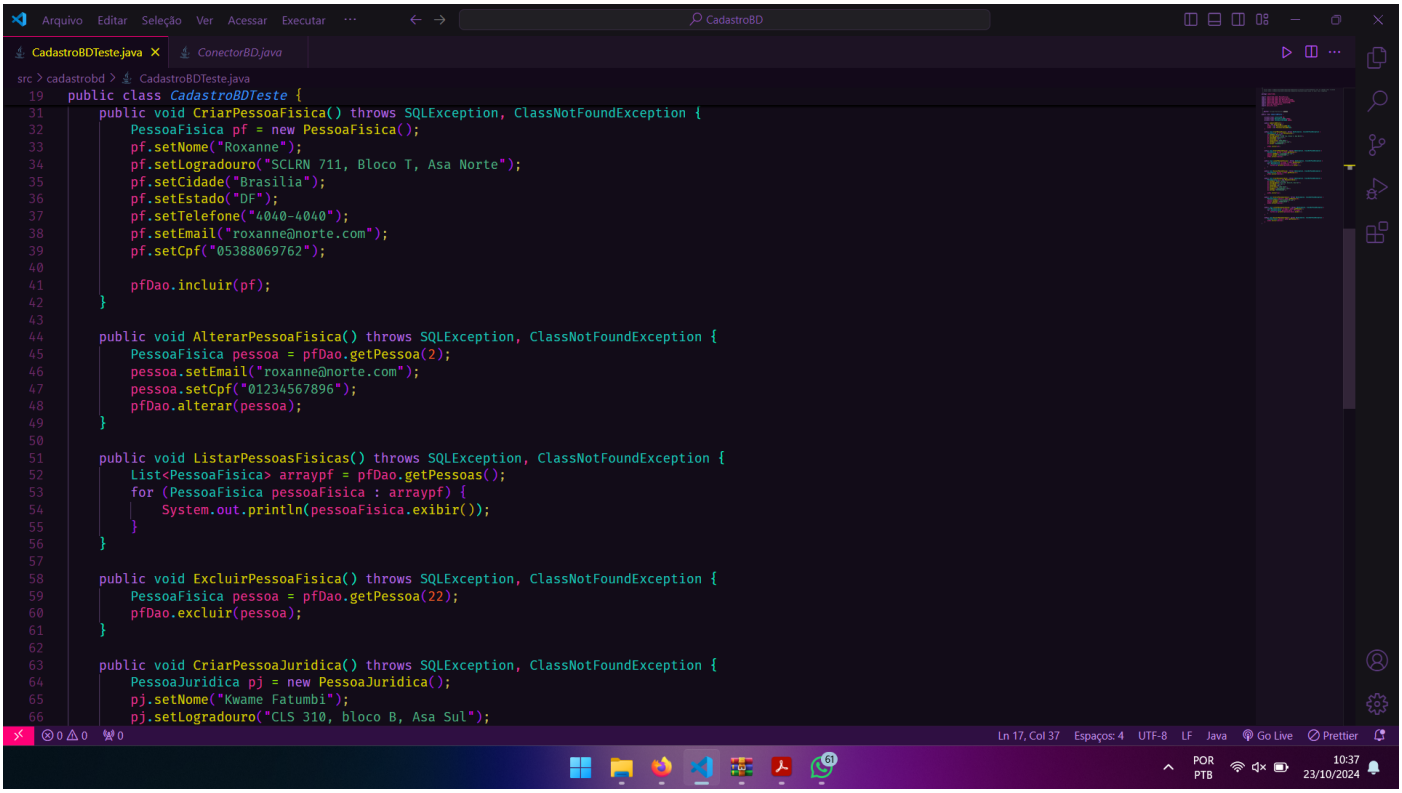
- **Por que métodos acionados diretamente pelo método main, sem o uso de um objeto, precisam ser marcados como static?**

– Em Java, o método main é o ponto de entrada da aplicação e é executado sem a criação de uma instância da classe onde ele está definido. Métodos que não pertencem a uma instância específica de uma classe precisam ser declarados como static para que possam ser chamados sem que um objeto seja instanciado.

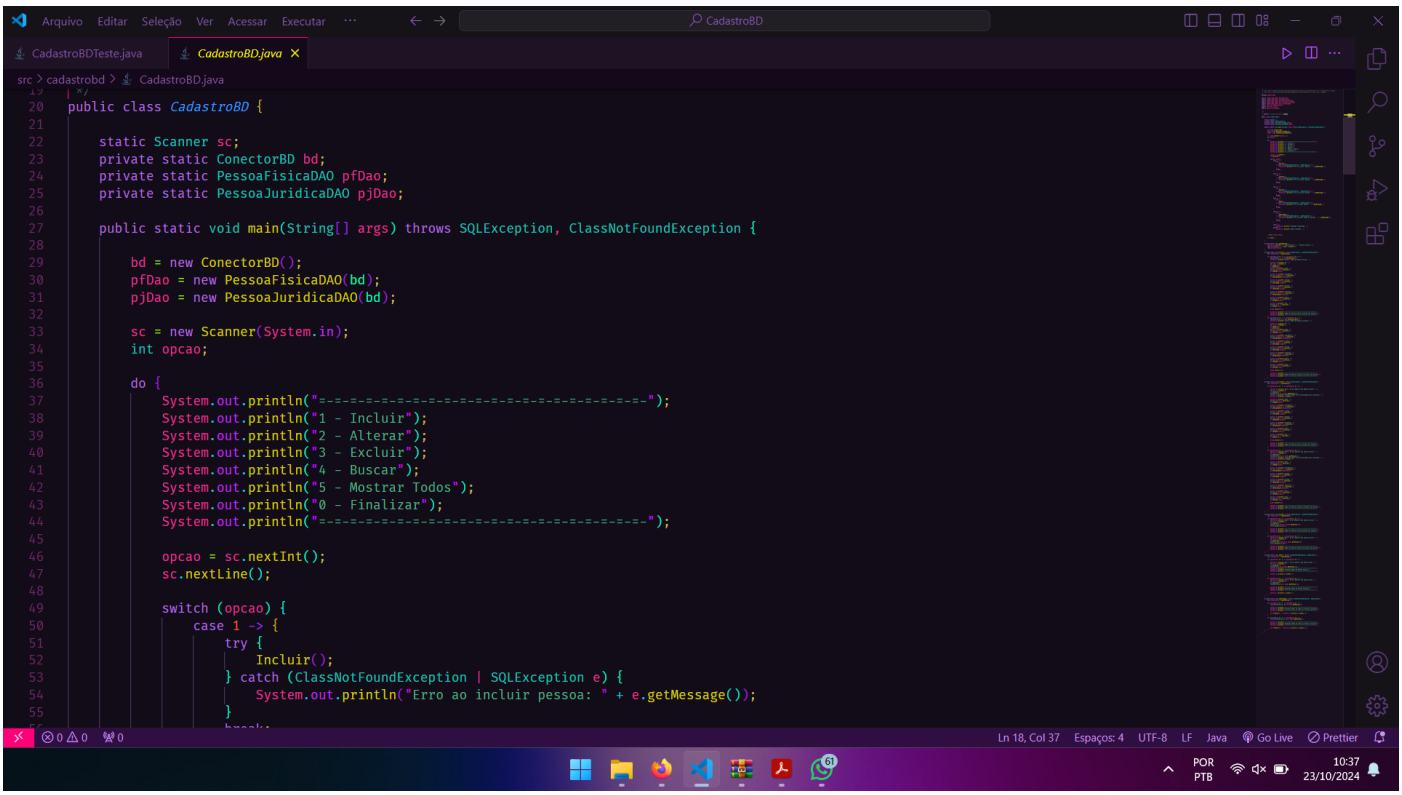
Ao marcar um método como static, estamos dizendo que ele pertence à própria classe e não a uma instância da classe. Como o método main é estático, ele só pode invocar outros métodos estáticos diretamente, pois esses também não dependem de uma instância de objeto. Caso contrário, seria necessário criar uma instância da classe antes de chamar o método.


File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
CadastroBD - Apache NetBeans IDE 22
Search (Ctrl+I)
<default config>
275,8/307,0MB
Projects
Files
Services
mssql-jdbc-12.8.1.jre11.jar
JDK 22 (Default)
CadastroBD
Source Packages
cadastrobd
CadastroBD.java
CadastroBDTeste.java
cadastrobd.model
cadastrobd.model.dao
PessoaFisicaDAO.java
PessoaJuridicaDAO.java
cadastrobd.model.util
ConectorBD.java
SequenceManager.java
Libraries
CadastroPOO [master]
Navigator
Members
<empty>
CadastroBD
CadastroBD()
PessoaJuridica copy.java
PessoaJuridica.java
Source
History
/**
*
* @author Filippe Markouizos Duarte
*/
public class PessoaJuridica extends Pessoa {
private String cnpj;
public PessoaJuridica(){
}
public PessoaJuridica(int id, String nome, String logradouro, String cidade,
String estado, String telefone, String email, String cnpj){
super(id, nome, logradouro, cidade, estado, telefone, email);
this.cnpj = cnpj;
}
/**
* @return the cnpj
Output - CadastroBD (run)
run:
1 - Incluir
2 - Alterar
3 - Excluir
4 - Buscar
5 - Mostrar Todos
0 - Finalizar
Output
CadastroBD (run)
1:1
INS


Arquivo Editar Seleção Ver Acessar Executar
CadastroBD
CadastroBDTeste.java
ConectorBD.java
src > cadastrobd > model > util > ConectorBD.java
/**
* @author Filippe Markouizos Duarte
*/
public class ConectorBD {
private Connection conexao;
private PreparedStatement ps;
private ResultSet rs;
public Connection getConnection() throws ClassNotFoundException, SQLException {
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
String connectionUrl = "jdbc:sqlserver://localhost\\DESKTOP-L5F8FG7:1433;databaseName=Loja;user=loja;password=loja;
trustServerCertificate=true";
if (conexao == null || conexao.isClosed()) {
conexao = DriverManager.getConnection(connectionUrl);
}
return conexao;
}
public PreparedStatement getPrepared(String sql) throws ClassNotFoundException, SQLException {
if (conexao == null || conexao.isClosed()) {
getConnection();
}
ps = conexao.prepareStatement(sql);
return ps;
}
public ResultSet getSelect(String sql) throws ClassNotFoundException, SQLException {
conexao = getConnection();
ps = getPrepared(sql);
rs = ps.executeQuery();
return rs;
}
public void Close() throws SQLException {
if (rs != null && !rs.isClosed()) {
Ln 27, Col 66
Espaços: 4
UTF-8
LF
Java
Go Live
Prettier
POR
PTB
10:36
23/10/2024



```java
public class CadastroBDTeste {
    public void CriarPessoaFisica() throws SQLException, ClassNotFoundException {
        PessoaFisica pf = new PessoaFisica();
        pf.setNome("Roxanne");
        pf.setLogradouro("SCLRN 711, Bloco T, Asa Norte");
        pf.setCidade("Brasilia");
        pf.setEstado("DF");
        pf.setTelefone("4040-4040");
        pf.setEmail("roxanne@norte.com");
        pf.setCpf("05388069762");

        pfDao.incluir(pf);
    }

    public void AlterarPessoaFisica() throws SQLException, ClassNotFoundException {
        PessoaFisica pessoa = pfDao.getPessoa(2);
        pessoa.setEmail("roxanne@norte.com");
        pessoa.setCpf("01234567896");
        pfDao.alterar(pessoa);
    }

    public void ListarPessoasFisicas() throws SQLException, ClassNotFoundException {
        List<PessoaFisica> arraypf = pfDao.getPessoas();
        for (PessoaFisica pessoaFisica : arraypf) {
            System.out.println(pessoaFisica.exibir());
        }
    }

    public void ExcluirPessoaFisica() throws SQLException, ClassNotFoundException {
        PessoaFisica pessoa = pfDao.getPessoa(22);
        pfDao.excluir(pessoa);
    }

    public void CriarPessoaJuridica() throws SQLException, ClassNotFoundException {
        PessoaJuridica pj = new PessoaJuridica();
        pj.setNome("Kwame Fatumbi");
        pj.setLogradouro("CLS 310, bloco B, Asa Sul");
```



```java
public class CadastroBD {

    static Scanner sc;
    private static ConectorBD bd;
    private static PessoaFisicaDAO pfDao;
    private static PessoaJuridicaDAO pjDao;

    public static void main(String[] args) throws SQLException, ClassNotFoundException {

        bd = new ConectorBD();
        pfDao = new PessoaFisicaDAO(bd);
        pjDao = new PessoaJuridicaDAO(bd);

        sc = new Scanner(System.in);
        int opcao;

        do {
            System.out.println("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-");
            System.out.println("1 - Incluir");
            System.out.println("2 - Alterar");
            System.out.println("3 - Excluir");
            System.out.println("4 - Buscar");
            System.out.println("5 - Mostrar Todos");
            System.out.println("0 - Finalizar");
            System.out.println("=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-");

            opcao = sc.nextInt();
            sc.nextLine();

            switch (opcao) {
                case 1 -> {
                    try {
                        Incluir();
                    } catch (ClassNotFoundException | SQLException e) {
                        System.out.println("Erro ao incluir pessoa: " + e.getMessage());
                    }
```