



Missão Prática | Nível 3 | Mundo 3

João Rainier de Castro Carvalho (202208942661)

Estácio Distrito Federal (Polo Asa Sul)/DF – Desenvolvimento Full Stack – 2022.3 – Mundo 3

Link GitHub: [Faculdade-Estacio-3-semester/Nível 3 - Back-end Sem Banco Não Tem/CadastroBD at main · joaorainier/Faculdade-Estacio-3-semester \(github.com\)](https://github.com/Faculdade-Estacio-3-semester/Nivel-3-Back-end-Sem-Banco-Não-Tem/CadastroBD)

RPG0016 - BackEnd sem banco não tem
Criação de aplicativo Java, com acesso ao banco de dados SQL Server através do middleware JDBC

Objetivos da prática:

Implementar persistência com base no middleware JDBC.
Utilizar o padrão DAO (Data Access Object) no manuseio de dados.
Implementar o mapeamento objeto-relacional em sistemas Java.
Criar sistemas cadastrais com persistência em banco relacional.
No final do exercício, o aluno terá criado um aplicativo cadastral com uso do SQL Server na persistência de dados.

Materiais necessários para a prática:

SQL Server, com o banco de dados gerado em prática anterior (loja).

JDK e IDE NetBeans.

Navegador para Internet, como o Chrome.

Banco de dados SQL Server com o Management Studio.

Equipamentos:

- Computador com acesso à Internet.
- JDK e IDE NetBeans.
- Banco de dados SQL Server.
- Navegador de Internet instalado no computador.

1 – Qual a importância dos componentes de middleware, como o JDBC?

Desempenhar um papel crucial na construção de sistemas distribuídos, fornecendo abstração, portabilidade, segurança e otimização para o acesso a dados em diferentes ambientes e contextos de desenvolvimento. Eles simplificam a integração de aplicativos com bancos de dados e contribuem para a eficiência e robustez dos sistemas de software.

2 - Qual a diferença no uso de Statement ou PreparedStatement para a manipulação de dados?

Enquanto Statement é adequado para consultas simples e estáticas, o PreparedStatement é preferível em termos de segurança e desempenho, especialmente ao lidar com consultas parametrizadas ou consultas executadas várias vezes com diferentes valores. O uso apropriado depende dos requisitos específicos da aplicação.

3 - Como o padrão DAO melhora a manutenibilidade do software?

O padrão DAO (Data Access Object) é um padrão de projeto que visa encapsular o acesso a dados em uma camada separada, isolando o código de acesso ao banco de dados do restante da aplicação. Ao aplicar o padrão DAO, os desenvolvedores podem criar sistemas mais flexíveis, modulares e de fácil manutenção, reduzindo a complexidade e melhorando a escalabilidade da aplicação ao longo do tempo.

4 - Como a herança é refletida no banco de dados, quando lidamos com um modelo estritamente relacional?

A representação de herança em um modelo estritamente relacional pode ser realizada de diferentes maneiras, e a escolha depende dos requisitos específicos do sistema, da facilidade de modelagem e da eficiência desejada nas consultas.

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

A escolha entre persistência em arquivo e banco de dados depende dos requisitos específicos da aplicação, como escalabilidade, segurança, complexidade dos dados, necessidade de transações ACID, entre outros. Sistemas mais complexos ou que exigem recursos avançados geralmente optam por bancos de dados, enquanto aplicativos mais simples podem usar persistência em arquivo.

Ambas as abordagens têm vantagens e desvantagens, e a escolha depende das características únicas do projeto em questão.

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

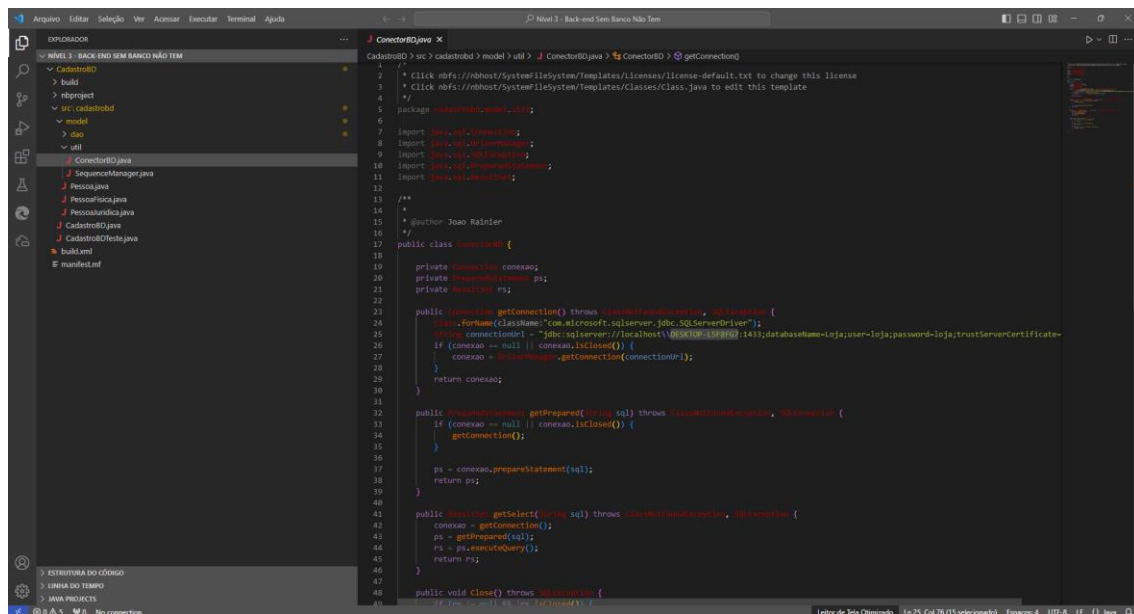
O uso de operadores lambda e Streams no Java permite escrever código mais funcional, reduzindo a quantidade de código boilerplate e facilitando a expressão de operações em coleções. Além disso, a sintaxe mais limpa e declarativa torna o código mais legível e manutenível. Isso é particularmente útil ao trabalhar com operações como filtragem, mapeamento e redução em coleções de dados.

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

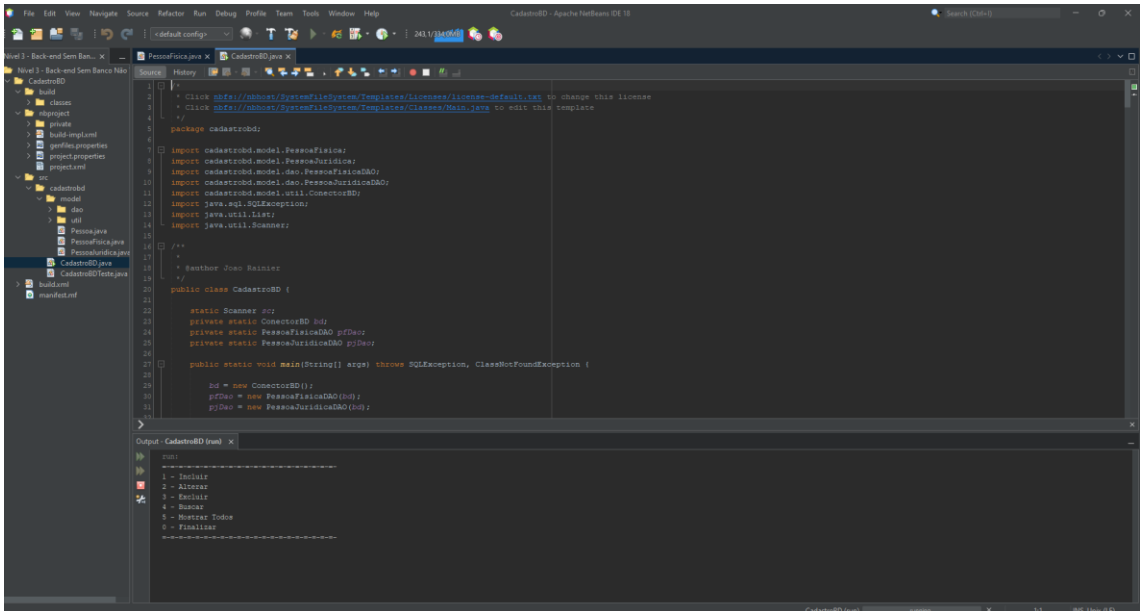
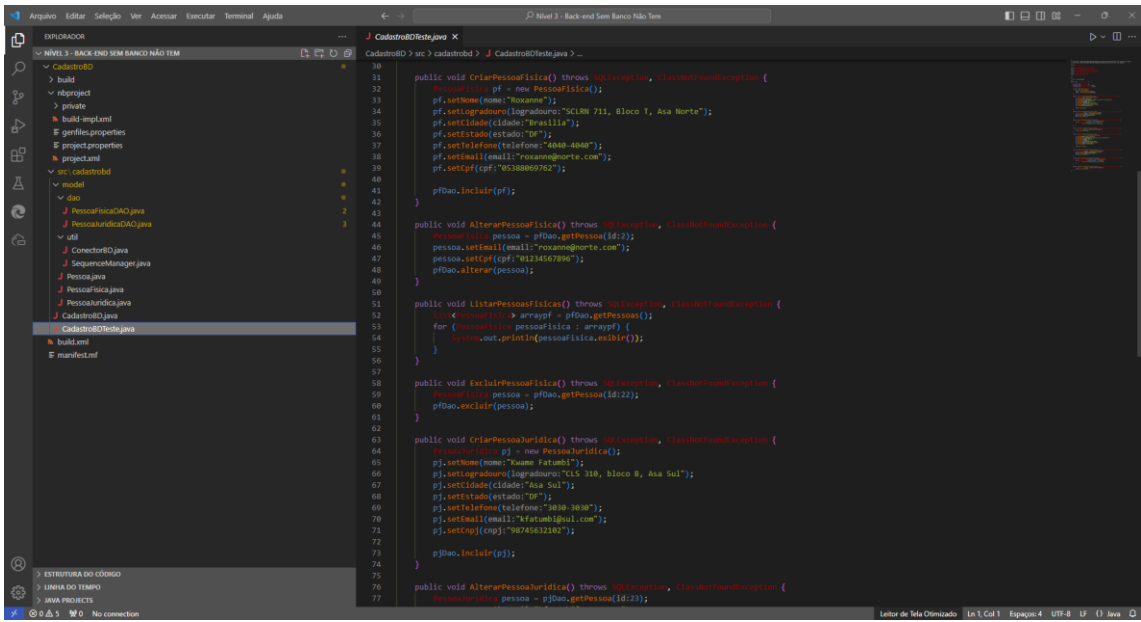
No Java, o método main é o ponto de entrada para a execução de um programa. Ele é chamado pelo Java Virtual Machine (JVM) ao iniciar a aplicação. Para que o método main seja acionado diretamente, sem a necessidade de criar uma instância da classe que o contém, ele deve ser marcado como static.


A palavra-chave static indica que o método ou atributo pertence à classe em vez de instâncias individuais da classe. Quando um método é marcado como static, ele pode ser chamado diretamente na classe, sem a necessidade de criar um objeto daquela classe.

É importante observar que, mesmo sendo uma convenção padrão, a declaração do método main como static é obrigatória para que a JVM encontre o ponto de entrada do programa. Se você tentar executar um programa Java sem um método main static, a JVM não será capaz de iniciar o programa corretamente.



```
1 package cadastro.util;
2
3 import java.sql.*;
4 import javax.swing.JOptionPane;
5 import java.util.*;
6
7 public class ConectorBD {
8     private Connection conexao;
9     private PreparedStatement ps;
10    private ResultSet rs;
11
12    /**
13     *
14     * @author João Rainier
15     */
16    public Connection getConnection() throws ClassNotFoundException, SQLException {
17        try {
18            Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
19            conexao = DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databaseName=loja;user=loja;password=loja;trustServerCertificate=true");
20            if (conexao == null || !conexao.isClosed()) {
21                conexao = ConectorManager.getConnection(conexao);
22            }
23            return conexao;
24        } catch (Exception e) {
25            JOptionPane.showMessageDialog(null, e.getMessage());
26        }
27    }
28
29    public PreparedStatement getPreparedStatement(String sql) throws ClassNotFoundException, SQLException {
30        if (conexao == null || !conexao.isClosed()) {
31            getConnection();
32        }
33        ps = conexao.prepareStatement(sql);
34        return ps;
35    }
36
37    public ResultSet getSelectQuery(String sql) throws ClassNotFoundException, SQLException {
38        conexao = getConnection();
39        ps = getPreparedStatement(sql);
40        rs = ps.executeQuery();
41        return rs;
42    }
43
44    public void close() throws SQLException {
45        if (conexao != null) {
46            conexao.close();
47        }
48        if (ps != null) {
49            ps.close();
50        }
51        if (rs != null) {
52            rs.close();
53        }
54    }
55 }
```



 Conectar ao Servidor

SQL Server

Tipo de servidor:

Mecanismo de Banco de Dados

Nome do servidor:

localhost\SQLEXPRESS01

Autenticação:

Autenticação do Windows

Nome de usuário:

DESKTOP-L5F8FG7\Joao Rainier

Senha:

☐ Lembrar senha

Conectar

Cancelar

Ajuda

Opções >>