



Domain Driven Design Using Java

Prof. Gilberto Alexandre das Neves
profgilberto.neves@fiap.com.br

MVC – Model, View e Controller

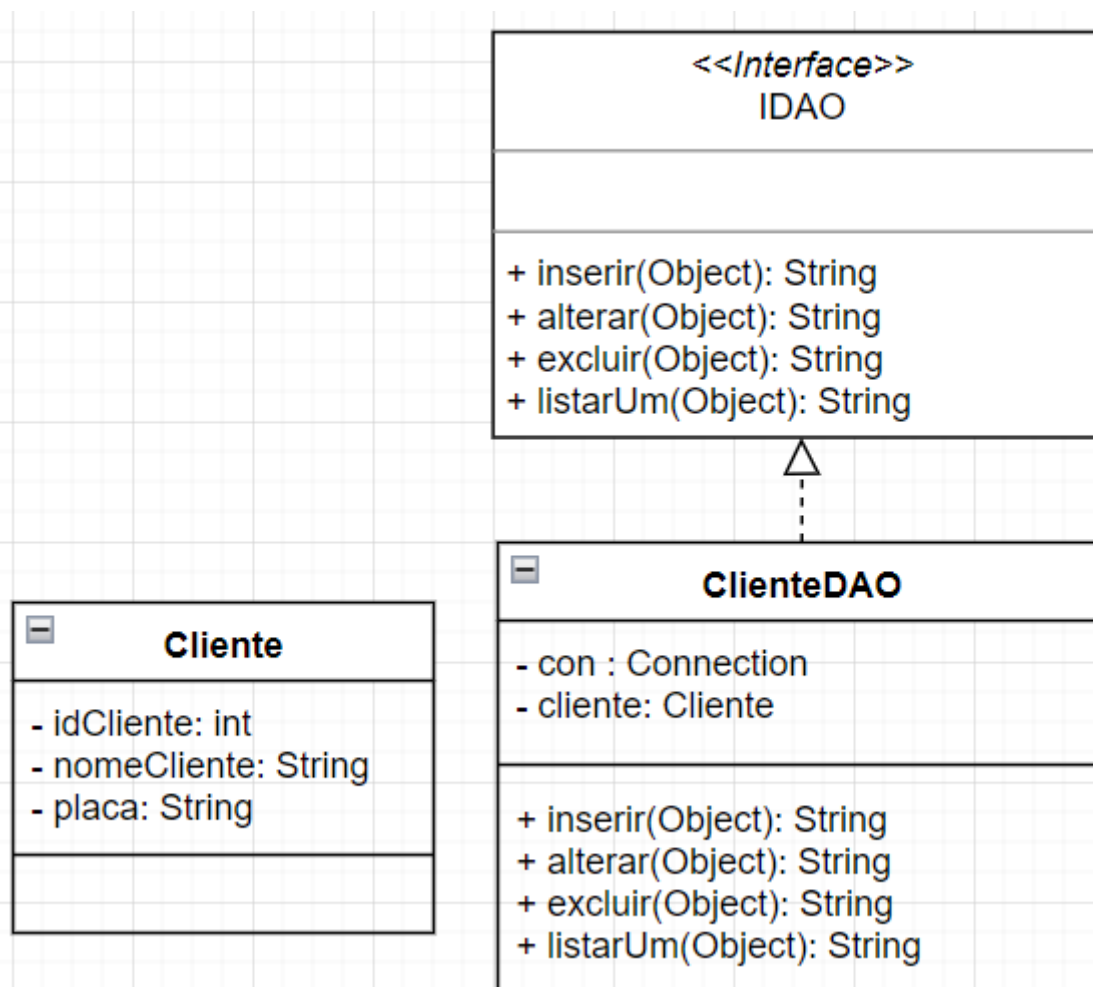
Praticando...

Vamos excluir nossa tabela **ddd_cliente**. E depois execute o script abaixo em seu banco no **Oracle SQL Developer**.

```
CREATE TABLE ddd_cliente(  
    id_cliente NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,  
    nome_cliente VARCHAR2(30),  
    placa VARCHAR2 (7) NOT NULL,  
    FOREIGN KEY (placa) REFERENCES ddd_carro(placa)  
);
```


Projeto Parking – Camada Model

Utilizando o projeto **parking**, implemente as **Classes** abaixo na camada **model** (pacotes **dto** e **dao**):



Projeto Parking – Camada **Controller**

Implemente a **Classe** abaixo no pacote **controller**:

 ClienteController
<ul style="list-style-type: none">+ inserirCliente(String nome, String placa): String+ alterarCliente(int id, String nome, String placa): String+ excluirCliente(int id): String+ listarUmCliente(int id): String

Implemente a classe **ClienteView** no pacote **view** com o método **main** (com JOptionPane). Esta classe deve realizar:

1. Exibir um menu perguntando ao usuário se deseja inserir, alterar, excluir ou listar um cliente.
2. Dependendo da escolha do usuário acima, solicite as informações necessárias para realizar a operação (id, nome, placa).
3. Perguntar ao usuário se deseja continuar (em caso afirmativo repetir os passos de 1 a 3 novamente).
4. Em caso negativo encerre o programa.


Classe Cliente (pacote dto)

```
1  package br.com.fiap.model.dto;
2  public class Cliente {
3      private int idCliente;
4      private String nomeCliente;
5      private String placa;
6  >  public Cliente() {}
7
8
9  >  public int getIdCliente() { return idCliente; }
10
11
12
13  >  public void setIdCliente(int idCliente) { this.idCliente =
14  >      idCliente; }
15
16
17  >  public String getNomeCliente() { return nomeCliente; }
18
19
20
21  >  public void setNomeCliente(String nomeCliente) { this.nomeCliente
22  >      = nomeCliente; }
23
24
25  >  public String getPlaca() { return placa; }
26
27
28
29  >  public void setPlaca(String placa) { this.placa = placa; }
30
31
32  }
```



Classe ClienteDAO (pacote dao)

```
11     private Connection con;  
12     private Cliente cliente;  
13  
14     > public ClienteDAO(Connection con) { this.con = con; }  
17  
18     > public Connection getCon() { return con; }  
21  
22     ↑ public String inserir(Object object) {  
23         cliente = (Cliente) object;  
24         String sql = "insert into ddd_cliente(nome_cliente,placa)  
25             values(?,?)";  
26         try (PreparedStatement ps = getCon().prepareStatement(sql)) {  
27             ps.setString(1, cliente.getNomeCliente());  
28             ps.setString(2, cliente.getPlaca());  
29             if (ps.executeUpdate() > 0) {  
30                 return "Inserido com sucesso.";  
31             } else {  
32                 return "Erro ao inserir";  
33             }  
34         } catch (SQLException e) {  
35             return "Erro de SQL: " + e.getMessage();  
36         }  
37     }
```


Classe ClienteDAO (pacote dao)

```
37
38  public String alterar(Object object) {
39     cliente = (Cliente) object;
40     String sql = "update ddd_cliente set nome_cliente=?,placa=?
41                 where id_cliente=?";
42     try (PreparedStatement ps = getCon().prepareStatement(sql)) {
43         ps.setString(1, cliente.getNomeCliente());
44         ps.setString(2, cliente.getPlaca());
45         ps.setInt(3, cliente.getIdCliente());
46         if (ps.executeUpdate() > 0) {
47             return "Alterado com sucesso.";
48         } else {
49             return "Erro ao alterar";
50         }
51     } catch (SQLException e) {
52         return "Erro de SQL: " + e.getMessage();
53     }
```

Classe ClienteDAO (pacote dao)

```
54
55  public String excluir(Object object) {
56     cliente = (Cliente) object;
57     String sql = "delete from ddd_cliente where id_cliente=?";
58     try (PreparedStatement ps = getCon().prepareStatement(sql)) {
59         ps.setInt(1, cliente.getIdCliente());
60         if (ps.executeUpdate() > 0) {
61             return "Excluído com sucesso.";
62         } else {
63             return "Erro ao excluir";
64         }
65     } catch (SQLException e) {
66         return "Erro de SQL: " + e.getMessage();
67     }
68 }
```

Classe ClienteDAO (pacote dao)

```
69
70  public String listarUm(Object object) {
71     cliente = (Cliente) object;
72     String sql = "select * from ddd_cliente where id_cliente=?";
73     try (PreparedStatement ps = getCon().prepareStatement(sql)) {
74         ps.setInt(1, cliente.getIdCliente());
75         ResultSet rs = ps.executeQuery();
76         if (rs.next()) {
77             return "Id: " + cliente.getIdCliente() + "\nNome: " +
78                 rs.getString("nome_cliente") + "\nPlaca: " +
79                 rs.getString("placa");
80         } else {
81             return "Registro não encontrado!";
82         }
83     } catch (SQLException e) {
84         return "Erro de SQL: " + e.getMessage();
85     }
86 }
```

```
10 public class ClienteController {
11     public String inserirCliente(String nome, String placa) throws
        ClassNotFoundException, SQLException {
12         String resultado;
13         Connection con = ConnectionFactory.abrirConexao();
14         Cliente cliente = new Cliente();
15         cliente.setNomeCliente(nome);
16         cliente.setPlaca(placa);
17         ClienteDAO clienteDAO = new ClienteDAO(con);
18         resultado = clienteDAO.inserir(cliente);
19         ConnectionFactory.fecharConexao(con);
20         return resultado;
21     }
```

```
22
23     public String alterarCliente(int id, String nome, String placa)
        throws ClassNotFoundException, SQLException {
24         String resultado;
25         Connection con = ConnectionFactory.abrirConexao();
26         Cliente cliente = new Cliente();
27         cliente.setIdCliente(id);
28         cliente.setNomeCliente(nome);
29         cliente.setPlaca(placa);
30         ClienteDAO clienteDAO = new ClienteDAO(con);
31         resultado = clienteDAO.alterar(cliente);
32         ConnectionFactory.fecharConexao(con);
33         return resultado;
34     }
```

```
35
36     public String excluirCliente(int id) throws
        ClassNotFoundException, SQLException {
37         String resultado;
38         Connection con = ConnectionFactory.abrirConexao();
39         Cliente cliente = new Cliente();
40         cliente.setIdCliente(id);
41         ClienteDAO clienteDAO = new ClienteDAO(con);
42         resultado = clienteDAO.excluir(cliente);
43         ConnectionFactory.fecharConexao(con);
44         return resultado;
45     }
```

```
46
47     public String listarUmCliente(int id) throws
        ClassNotFoundException, SQLException {
48         String resultado;
49         Connection con = ConnectionFactory.abrirConexao();
50         Cliente cliente = new Cliente();
51         cliente.setIdCliente(id);
52         ClienteDAO clienteDAO = new ClienteDAO(con);
53         resultado = clienteDAO.listarUm(cliente);
54         ConnectionFactory.fecharConexao(con);
55         return resultado;
56     }
57 }
```



```
7  ▶ public class ClienteView {
8  ▶     public static void main(String[] args) {
9      String nome, placa;
10     String[] escolha = {"Inserir", "Alterar", "Excluir", "Listar"};
11     int id, opcao;
12     ClienteController clienteController = new ClienteController();
13     do {
14         try {
15             opcao = JOptionPane.showOptionDialog(null, "Escolha uma das opções abaixo para
manipular um Cliente", "Escolha", JOptionPane.DEFAULT_OPTION, JOptionPane
.QUESTION_MESSAGE, null, escolha, escolha[0]);
16             switch (opcao) {
17                 case 0:
18                     nome = JOptionPane.showInputDialog("Digite o nome do cliente");
19                     placa = JOptionPane.showInputDialog("Digite a placa do carro");
20                     System.out.println(clienteController.inserirCliente(nome, placa));
21                     break;
22                 case 1:
23                     id = Integer.parseInt(JOptionPane.showInputDialog("Digite o Id do cliente"));
24                     nome = JOptionPane.showInputDialog("Digite o NOVO nome do cliente");
25                     placa = JOptionPane.showInputDialog("Digite a NOVA placa do carro");
26                     System.out.println(clienteController.alterarCliente(id, nome, placa));
27                     break;
```

```
28         case 2:
29             id = Integer.parseInt(JOptionPane.showInputDialog("Digite o Id do cliente"));
30             System.out.println(clienteController.excluirCliente(id));
31             break;
32         case 3:
33             id = Integer.parseInt(JOptionPane.showInputDialog("Digite o Id do cliente"));
34             JOptionPane.showMessageDialog(null, clienteController.listarUmCliente(id));
35             break;
36         default:
37             System.out.println("Opção inválida!");
38     }
39     } catch (Exception e) {
40         System.out.println("Erro: " + e.getMessage());
41     }
42 } while (JOptionPane.showConfirmDialog(null, "Deseja continuar?", "Atenção", JOptionPane
    .YES_NO_OPTION, JOptionPane.QUESTION_MESSAGE) == 0);
43 JOptionPane.showMessageDialog(null, "Fim de Programa!");
44 }
45 }
```

Praticando 2: O Retorno

I A classe ParkingView

Implemente a classe **ParkingView** no pacote **view** com o método **main** (com JOptionPane). Esta classe deve realizar:

1. Exiba um menu perguntando ao usuário quem ele quer manipular (carro ou cliente).
2. Se o usuário escolher carro: exiba um menu pedindo para o usuário escolher entre inserir, alterar, excluir ou listar (peça as informações necessárias e realize a ação escolhida).
3. Se o usuário escolher cliente: exiba um menu pedindo para o usuário escolher entre inserir, alterar, excluir ou listar (peça as informações necessárias e realize a ação escolhida).
4. Pergunte se o usuário deseja continuar. Em caso afirmativo repita os **passos de 1 a 4**. Em caso negativo, encerre o programa.



Java como programar. Paul Deitel e Harvey Deitel. Pearson, 2011.

Java 8 – Ensino Didático : Desenvolvimento e Implementação de Aplicações. Sérgio Furgeri. Editora Érica, 2015.

Até breve!