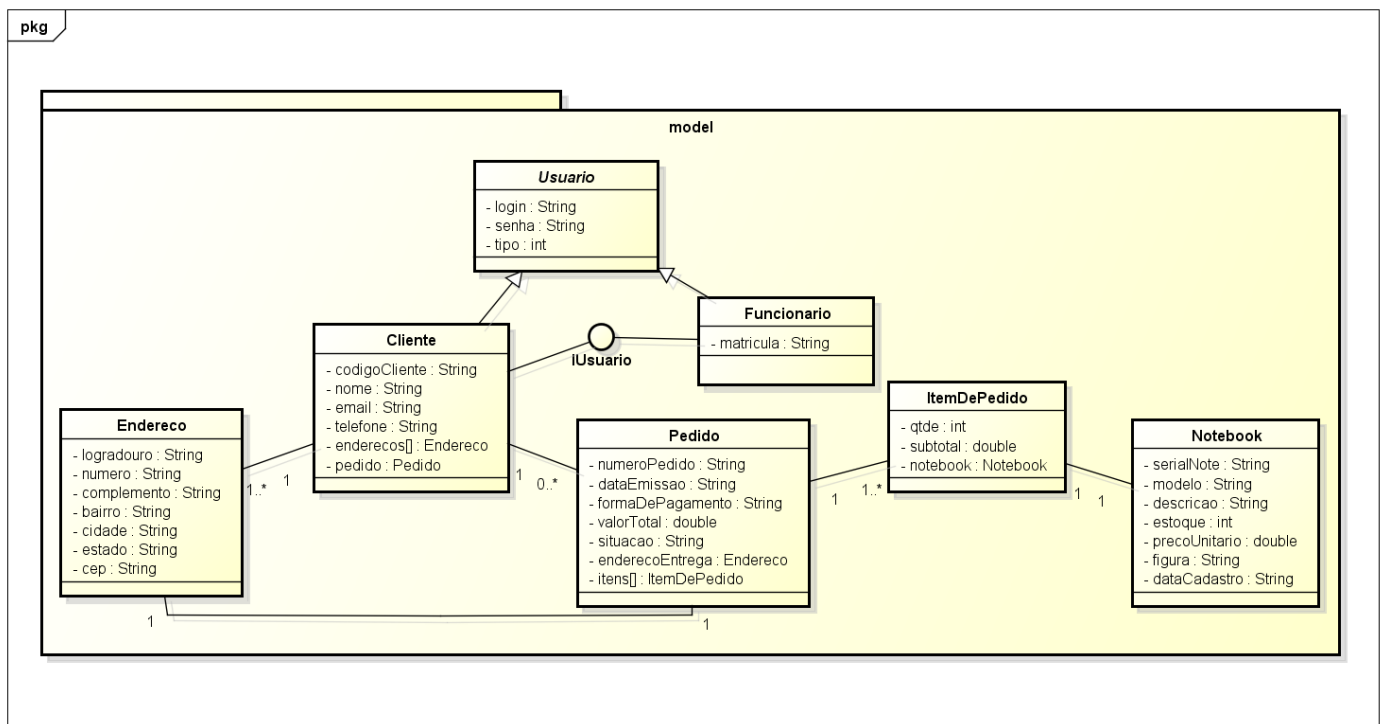


# Projeto *InfoNote\_14*



powered by astah®

## OBJETIVOS:

- 1) Substituir todos os vetores existentes por List
- 2) Finalizar o projeto InfoNote em ambiente console.

Execute este passo a passo para substituir todos os vetores por List:

1 – Copiar e colar o projeto **InfoNote\_13** e renomear para **InfoNote\_14**.

2 – Na Classe **NotebookDAO**, alterar o vetor notebook que está no método **buscarTodos()**:

```
public static List <Notebook> buscarTodos() {  
    List<Notebook> notebooks = null;
```

... continua ...

```
// vetor de objetos  
    notebooks = new ArrayList<Notebook>();
```

... continua ...

```
while (rs.next()) {  
    notebooks.add(new Notebook(  
        rs.getString("serialnote"),  
        rs.getString("modelo"),
```

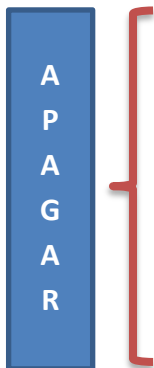
```
rs.getString("descricao"),
rs.getInt("estoque"),
rs.getDouble("precounitario"),
rs.getString("figura"),
rs.getString("datacadastro")));
```

3 – Ainda neste método, **APAGAR** a linha `int i = 0;`

4 – Na Classe **Cliente**, na declaração de atributos, alterar os vetores **enderecos** e **pedidos**:

```
private List<Endereco> enderecos = new ArrayList<Endereco>();
private List<Pedido> pedidos = new ArrayList<Pedido>();
```

5 – Ainda na Classe **Cliente**, apagar os métodos **get** e **set** que ficaram com erros após a alteração do item 4:



```
public Endereco getEnderecos() {
    return enderecos;
}
public void setEnderecos(Endereco[] enderecos) {
    this.enderecos = enderecos;
}
public Pedido[] getPedidos() {
    return pedidos;
}
public void setPedidos(Pedido[] pedidos) {
    this.pedidos = pedidos;
}
```

6 – Na Classe **Cliente**, gerar novamente os métodos **get** e **set** faltantes.

7 – Na Classe **Cliente**, observe que o método **inserirPedido()** agora acusou erro. Altere este método de forma que ele fique exatamente como está abaixo e analise o motivo desta alteração:

```
//inserir o pedido no vetor
public boolean inserirPedido(Pedido pedido) {
    return pedidos.add(pedido);
}
```

8 – Na Classe **Cliente**, fazer o mesmo no método **inserirEndereco()**:

```
//inserir o endereço no vetor
public boolean inserirEndereco (Endereco end) {
    return enderecos.add(end);
}
```

9 – Na Classe **Pedido**, alterar a declaração do vetor **ItemDePedido**:

```
private List<ItemDePedido> itens = new ArrayList<ItemDePedido>();
```

10 – Ainda na Classe **Pedido**, **apagar** o método **get Item** que deu erro e **refazer somente** este **get**.

11 – Ainda na Classe **Pedido**, alterar o método **inserirItem()**, para que fique exatamente assim:

```
public boolean inserirItem(ItemDePedido item) {
    return itens.add(item);
}
```

12 – Ainda na Classe **Pedido**, alterar o método **toString()**, na parte do **for**:

...continua...

```
for (ItemDePedido item : itens){
    retValue += item;
}
return retValue;
```

## - Finalizando o projeto no ambiente console:

13 – Na Classe **NotebookDAO**, criar o método **buscarPorModelo()**:

```
public static Notebook buscarPorModelo(String modelo){
    Notebook notebook = null;
    NotebookDAO noteDAO = new NotebookDAO();

    try {
        // Criação do select
        String sql = "Select * from notebook where modelo = ?";

        // Obter a conexão com o banco de dados
        Conexao conex = new Conexao (noteDAO.url, noteDAO.driver,
            noteDAO.login, noteDAO.senha);

        Connection con = conex.obterConexao();

        PreparedStatement comando = con.prepareStatement(sql);

        comando.setString(1, modelo);

        ResultSet rs = comando.executeQuery();

        if (rs.next()){
            notebook = new Notebook(rs.getString("serialnote"),
                rs.getString("modelo"),
                rs.getString("descricao"),
                rs.getInt("estoque"),
                rs.getDouble("precounitario"),
                rs.getString("figura"),
                rs.getString("datacadastro"));
        }

        rs.close();
        comando.close();
        con.close();
    } catch (Exception e){
        System.out.println(e.getMessage());
    }
    return notebook;
}
```

14 – Na Classe **InfoNote**, alterar a declaração do vetor **notebook**, que passa a ser um atributo comum:

```
Notebook notebook;
```

15 – Ainda na Classe **InfoNote**, preste atenção na alteração que foi feita na inserção de notebooks, que agora passa a trabalhar diretamente com o banco de dados:

```
/* OBS IMPORTANTE: A partir de agora, cada vez que esta Classe for
 * executada, tentará gravar no Banco de Dados os Notebooks abaixo.
 *
 * A dica é: Executar uma única vez para gravar no banco, e depois voltar
 * aqui na Classe e comentar todos estes notebooks abaixo, para que não
 * ocorra erro de banco de dados por inserção duplicada de chave primária.
 */
NotebookDAO.inserir("1", "Negativo N22BR",
    "CPU Intel Core 2 Duo, Memória 2 GB, HD 250 GB", 6, 1200.00,
    "img\\n22br.jpg", "19/05/2011");

NotebookDAO.inserir("2", "Bell B55BR",
    "CPU Intel I3, Memória 4 GB, HD 500 GB", 3, 1800.00,
    "img\\b55br.jpg", "20/05/2011");

NotebookDAO.inserir("3", "Pompaq P41BR",
    "CPU Intel I3, Memória 3 GB, HD 320 GB", 1, 1600.00,
    "img\\p41br.jpg", "21/05/2011");

NotebookDAO.inserir("4", "CCF CR71CH",
    "CPU Intel Dual Core, Memória 2 GB, HD 160 GB", 5, 1100.00,
    "img\\cr71ch.jpg", "10/06/2011");

NotebookDAO.inserir("5", "BradescoTech BD22BR",
    "CPU AMD Phenon II, Memória 4 GB, HD 500 GB", 2, 1900.00,
    "img\\bd22br.jpg", "10/06/2011");
```

16 – Na Classe **InfoNote**, alterar o método **buscarNotebook()**, para que fique exatamente assim:

```
public void buscarNotebook() {
    List<Notebook> notebooks = NotebookDAO.buscarTodos();

    for (Notebook notebook: notebooks){
        System.out.println(notebook);
    }
}
```

17 – Na Classe **InfoNote**, alterar dentro o método **inserirNotebook()** a forma de busca do notebook:

```
// Busca notebook selecionado no banco
Notebook aux = NotebookDAO.buscarPorModelo(modelo);
```

17.1 – Substituir, ainda no método **inserirNotebook()**, a variável *serialNote* por *modelo*:

```
String modelo = Teclado.lerTexto("Informe o modelo do notebook"
    + " para compra: ");
```

18 – Na Classe **InfoNote**, criar o método **manterCarrinho()**, pois este até agora estava exibindo a frase em Construção:

```
public void manterCarrinho(int operacao) {
    // Insere notebook no carrinho
    if (operacao == 4){
        inserirNotebook();
        // Visualiza o carrinho
    } else if (operacao == 6){
        verCarrinho();
    }
}
```

19 – Na Classe **InfoNote**, desenvolver o método **verCarrinho()**:

```
public void verCarrinho() {  
    if (pedido == null) {  
        System.out.println("Carrinho vazio!");  
    } else {  
        System.out.println(pedido);  
    }  
}
```

20 – Na Classe **InfoNote**, no método **info.mostrarMenu()**, alterar os cases que acionam o método **manterCarrinho()**, deixando-os exatamente assim:

```
case INSERIR_NOTEBOOK:  
    info.manterCarrinho(opcao);  
    break;  
  
case REMOVER_NOTEBOOK:  
    info.manterCarrinho(opcao);  
    break;  
  
case VER_CARRINHO:  
    info.manterCarrinho(opcao);  
    break;
```

21 – Na Classe **ItemDePedido**, alterar o método **toString()** para que este exiba o modelo do notebook:

```
@Override  
public String toString() {  
    final String ENTER = "\n";  
    String retValue = "";  
    retValue = "Informações sobre Item de Pedido: " + ENTER +  
        "Modelo: " + notebook.getModelo() + ENTER +  
        "Quantidade: " + qtde + ENTER +  
        "Subtotal: " + subtotal;  
  
    return retValue;  
}
```

22 – Na Classe **Pedido**, criar o método **removerItem()**:

```
public boolean removerItem(ItemDePedido item) {  
    return itens.remove(item);  
}
```

22 – Na Classe **InfoNote**, criar o método **removerCarrinho()**:

```
public void removerCarrinho() {  
    // Lê o notebook escolhido do teclado  
    String modelo = Teclado.lerTexto("Informe o modelo do notebook"  
        + " para remover do carrinho: ");  
  
    // Busca notebook selecionado no banco  
    Notebook aux = NotebookDAO.buscarPorModelo(modelo);  
  
    // Se não existir, interrompe  
    if (aux == null) {  
        return;  
    }  
  
    // Cria item a ser removido  
    ItemDePedido item = new ItemDePedido(1, aux.getPrecoUnitario(), aux);  
  
    pedido.removerItem(item);  
}
```

OBS: A função **Efetuar Compra** será feita futuramente no ambiente Web.