



INFORMA. FORMA. TRANSFORMA.



Tabela

INFORMA. FORMA. TRANSFORMA.

Projeto *GUI05_Tabela*



Vamos apresentar o uso de tabelas adaptando o projeto 18_ConexaoBD que já temos em nossa workspace.

Aplicaremos o recurso de tabelas apenas para o recurso “Mostrar Mensagens” deste projeto.

Por gentileza, copie o projeto *18_ConexaoBD* e cole renomeando-o para *GUI05_Tabela*.

Certifique-se de possuir o banco de dados chamado 18_conexaobd no MySQL.

4



 INFORMA. FORMA. TRANSFORMA.


Primeiro passo:

Transformar os Vetores existentes em Listas Dinâmicas:

Na Classe *ControleDeContatos*, substitua o método *buscarMensagens()* por este:

```

public void buscarMensagens() {
    List<Contato> contatos = ContatoDAO.buscarTodos();
    new JanelaExibeMensagens(contatos);
}
  
```


 5

Na Classe *ContatoDAO*, a declaração do método *buscarTodos()* deve ficar assim:

```

public static List<Contato> buscarTodos() {
  
```

A declaração do vetor *contatos* deve ser substituída por

```

List<Contato> contatos = new LinkedList<Contato>();
  
```

Apagar as linhas:


```

contatos = new Contato[10];
int i = 0;
  
```

O laço com *while* deve ficar assim:

```

while (rs.next()) {
    Contato c = new Contato();
    c.setId(rs.getInt("id"));
    c.setNome(rs.getString("nome"));
    c.setEmail(rs.getString("email"));
    c.setMensagem(rs.getString("mensagem"));
    contatos.add(c);
}
  
```


 6

Segundo passo:



Dentro do pacote *util*, criar a Classe ***ContatoTableModel***.

```
public class ContatoTableModel extends AbstractTableModel{
    private final List<Contato> contatos;

    public ContatoTableModel(List<Contato> contatos){
        this.contatos=contatos;
    }

    @Override
    public int getColumnCount() {
        // TODO Auto-generated method stub
        return 4;
    }

    @Override
    public int getRowCount() {
        // TODO Auto-generated method stub
        return contatos.size();
    }
}
```



7

Continuando a Classe ***ContatoTableModel***.

```
@Override
public Object getValueAt(int linha, int coluna) {
    // TODO Auto-generated method stub
    Contato c = contatos.get(linha);

    switch (coluna) {
        case 0:
            return c.getId();
        case 1:
            return c.getNome();
        case 2:
            return c.getEmail();
        case 3:
            return c.getMensagem();
        default:
            break;
    }

    return null;
}
```

8

Finalizando a Classe ***ContatoTableModel***.

```
@Override
public String getColumnName(int column){
    switch (column) {
        case 0:
            return "ID";
        case 1:
            return "Nome";
        case 2:
            return "E-mail";
        case 3:
            return "Mensagem";
        default:
            return "";
    }
}
```



9

Terceiro passo:

Criar o pacote *view* e a Classe ***JanelaExibeMensagens***

```
public class JanelaExibeMensagens extends JFrame{
    public JanelaExibeMensagens(List<Contato> contatos){
        super("Lista de Mensagens:");

        JTable tabela = new JTable();
        ContatoTableModel tableModel=new ContatoTableModel(contatos);
        tabela.setModel(tableModel);


        JScrollPane scroll = new JScrollPane();
        scroll.getViewPort().add(tabela);
        scroll.setSize(100, 200);

        Container c = getContentPane();
        c.add(scroll);


        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setSize(400, 300);
        setVisible(true);
        setAlwaysOnTop(true);
    }
}
```




10




Dúvidas?




11




Bibliografia



Java Como Programar 8ª Edição
Paul Deitel e Harvey Deitel
Ed. Pearson



Java 8 Programação de Computadores
José Augusto N. G. Manzano & Roberto Affonso da Costa Junior
Ed. Érica | Saraiva



Fundamentos de Computação e Orientação a Objetos Usando Java
Francisco A. C. Pinheiro
Ed. LTC

12