



Universidade
Cruzeiro do Sul

Técnicas de Programação

Grade ou Tabela

Aula 10

Ms. Amilton Souza Martha

Grade



Identificação	Nome
1	Socrates
2	Platão
3	Aristóteles
4	Kant
5	Descartes

Registro selecionado: 1-Socrates

Grade – Características

- ✓ São componentes utilizados para a apresentação de estruturas bidimensionais de dados, dividida em linhas e colunas.
- ✓ Classe → `javax.swing.JTable`;
- ✓ As grades de dados são complexas por sua própria natureza, mas a classe `JTable` oculta boa parte dessa complexidade

JTable – Características

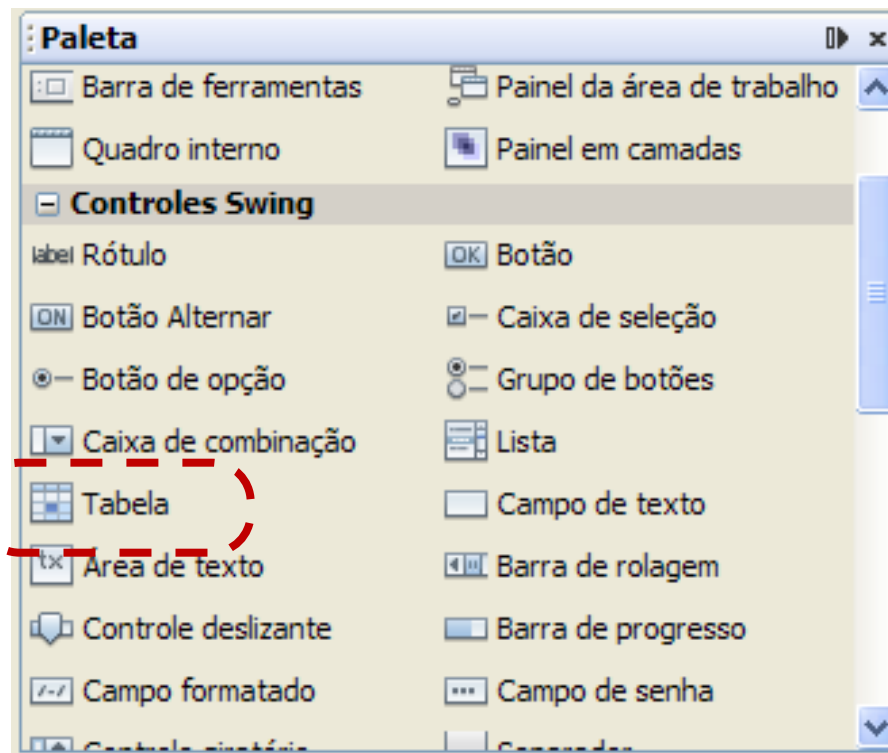
- ✓ A classe JTable não armazena os dados da grade que ela apresenta.
- ✓ Ela os obtém de um modelo representado pela classe *AbstractTableModel*.
- ✓ Um objeto da classe JTable simplesmente representa a aparência e o comportamento da grade, mas os dados que ela exibe são armazenados em um objeto da classe *AbstractTableModel*

JTable – Características

- ✓ Não é necessário criar um objeto do tipo `AbstractTableModel`, podemos criar uma grade utilizando:
 - ✓ **um vetor de textos:** com os títulos das colunas e
 - ✓ **uma matriz de objetos:** com os dados das linhas

Grade usando o NetBeans

- Crie um projeto e adicione um novo JFrame de nome Grade no pacote view
- A tabela se encontra na Paleta “Controles Swing”



Title 1	Title 2	Title 3	Title 4

Navegador**Inspetor**

- Formulário exemploGrade
 - + Outros componentes
 - [JFrame]
 - JScrollPane1 [JScrollPane]
 - jTable1 [JTable]



Use a propriedade do modelo para editar o conteúdo da JTable.

x

Title 1	Title 2

Conteúdo da tabela...

Alterar o nome da variável ...

Vincular



Eventos



Alinhar



Ancorar



Auto redimensionamento



Mesmo tamanho



Restaurar tamanho padrão

Espaço ao redor do componente...

Incluir em



Pai do projeto



Mover para cima

Mover para baixo

Recortar

Ctrl+X

Copiar

Ctrl+C


Duplicar

Ctrl+D

Excluir

Delete

Personalizar código

 Caixa de diálogo Personalizador

Modelo de tabela Colunas Linhas

☒ Especificado pelo usuário

☐ Vinculado

☐ Valor de componente existente

☐ Personalizar código

Fechar

Caixa de diálogo Personalizador

Modelo de tabela **Colunas** Linhas

Título	Tipo	Redimensionável	Editável
Title 1	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Title 2	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Title 3	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Title 4	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Conta: 4

Inserir

Título: Title 4

Tipo: Object

Editor: <nenhum>

Renderizador: <nenhum>

Modelo de seleção: Não permitido

☒ Permitir reordenar colunas com o método arrastar e soltar

Caixa de diálogo Personalizador

Modelo de tabela **Colunas** Linhas

Título	Tipo	Redimensionável	Editável
Cidades	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Municípios	Object	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Conta: 2

Inserir

Excluir

Mover para cima

Mover para baixo

Título: Title 2

Tipo: Object

Editor: <nenhum>

Renderizador: <nenhum>

Modelo de seleção: Não permitido

☒ Redimensionável ☒ Editável

Largura pref: Padrão

Largura mín: Padrão

Largura máx: Padrão

☒ Permitir reordenar colunas com o método arrastar e soltar

Fechar

Caixa de diálogo Personalizador

Modelo de tabela | Colunas | Linhas

Cidades	Municipios
Rio de Janeiro	87
São Paulo	356
Minas Gerais	56

Conta: 3

Inserir

Excluir

Mover para cima

Mover para baixo

Fechar



Cidades



Cidades	Municipios
Rio de Janeiro	87
São Paulo	356
Minas Gerais	56

a) Para alterar o conteúdo da grade

```
String titulos[] = {"Identificação", "Nome"};  
Object dados[][] = {  
    {1, "Sócrates"},  
    {2, "Platão"},  
    {3, "Aristóteles"},  
    {4, "Kant"},  
    {5, "Descartes"} };
```

```
DefaultTableModel modelo =  
    new DefaultTableModel(dados, titulos);
```

```
grade.setModel(modelo);
```

b) Para adicionar conteúdo na grade

```
DefaultTableModel modelo = (DefaultTableModel)  
    tblGrade.getModel();
```

```
modelo.addRow(new String [] {"6","Locke"});
```

c) Adicionar à tabela dados do banco

```
ArrayList<Aluno> lista = new AlunoDAO().listaAlunos();  
  
DefaultTableModel modelo = (DefaultTableModel)  
tblGrade.getModel();  
  
for (int i = 0; i < lista.size(); i++) {  
    modelo.addRow(new String[]{lista.get(i).getRgm(),  
lista.get(i).getNome()});  
}
```

Buscando do Banco de Dados

Banco de dados: **unicsul**

Tabela: **alunos**

Workspace (Loc..		Table Editor [unicsul.alunos]				
	Name	Type	Size	Decimals	Null	Primary Key
▶	rgm	varchar	10	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	nome	varchar	50	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>
*		varchar	50	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Criando Classe Aluno

Pacote
classes

```
public class Aluno {  
  
    private String rgm;  
    private String nome;  
  
    /**  
     * @return the rgm  
     */  
    public String getRgm() {  
        return rgm;  
    }  
  
    /**  
     * @param rgm the rgm to set  
     */  
    public void setRgm(String rgm) {  
        this.rgm = rgm;  
    }  
  
    /**  
     * @return the nome  
     */  
    public String getNome() {  
        return nome;  
    }  
  
    /**  
     * @param nome the nome to set  
     */  
    public void setNome(String nome) {  
        this.nome = nome;  
    }  
  
}
```

AlunoDAO

Pacote
dao

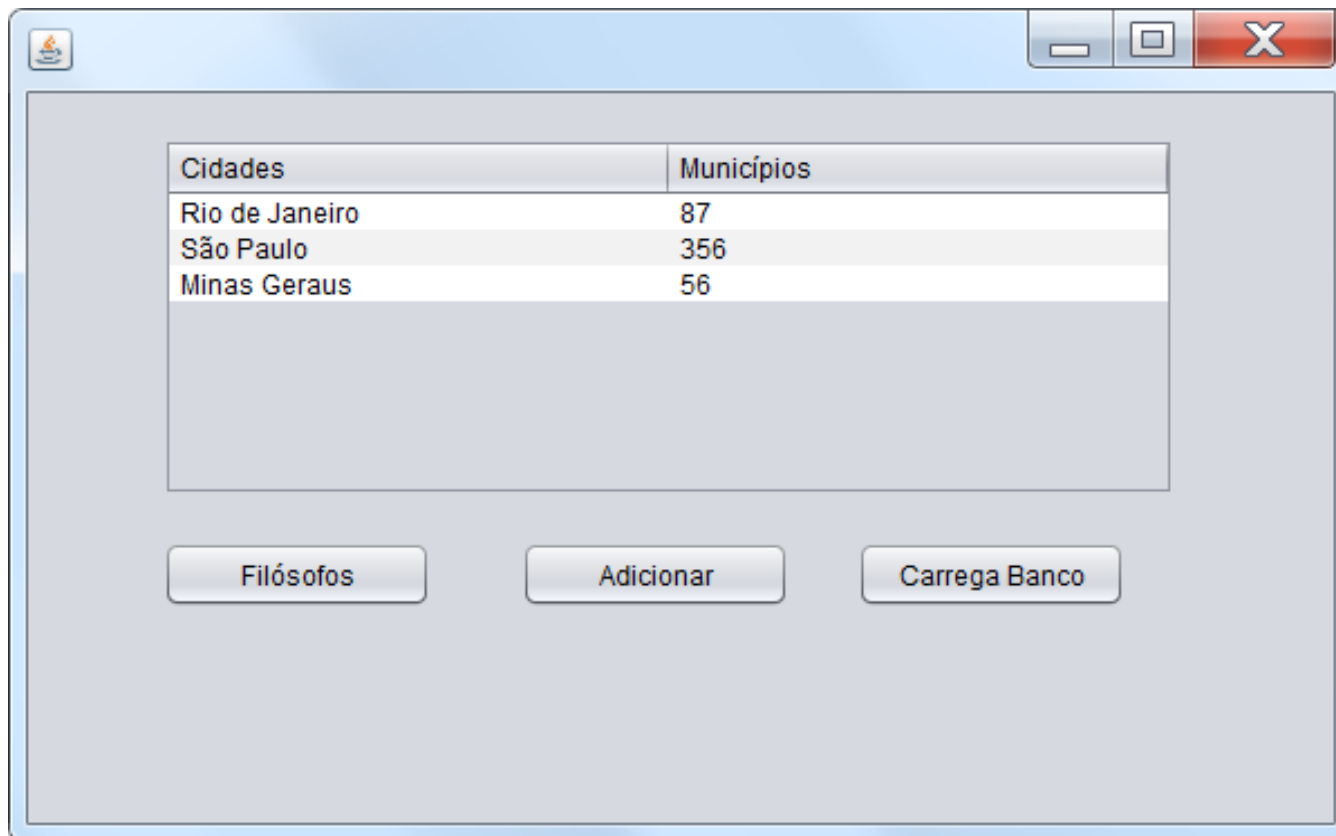
```
/**
 * Devolve a lista de alunos cadastrados
 *
 * @return
 */
public ArrayList<Aluno> listaAlunos() {
    ArrayList<Aluno> lista = new ArrayList<Aluno>();

    try {
        Connection con = Conecta.getConexao();
        String sql = "SELECT * FROM alunos ORDER BY rgm";
        PreparedStatement ps = con.prepareStatement(sql);
        ResultSet rs = ps.executeQuery();
        while (rs.next()) {
            Aluno aluno = new Aluno();
            aluno.setRgm(rs.getString("rgm"));
            aluno.setNome(rs.getString("nome"));

            lista.add(aluno);
        }
        rs.close();
        ps.close();
        con.close();
    } catch (Exception e) {
        System.out.println("ERRO: " + e.toString());
    }
    return lista;
}
```

b) Para saber qual está selecionado

```
private void tblGradeMouseReleased(java.awt.event.MouseEvent evt) {  
  
    if (tblGrade.getSelectionModel().isSelectionEmpty()) {  
        JOptionPane.showMessageDialog(null, "Nada selecionado");  
    } else {  
        int it = tblGrade.getSelectionModel().getMinSelectionIndex();  
        String escolhido = tblGrade.getModel().getValueAt(it, 0) + ", " +  
tblGrade.getModel().getValueAt(it, 1);  
        JOptionPane.showMessageDialog(null, escolhido);  
    }  
}
```



A screenshot of a web application window. The window has a title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main content area contains a table with two columns: 'Cidades' and 'Municípios'. The table lists three cities: Rio de Janeiro (87), São Paulo (356), and Minas Geraus (56). Below the table, there are three buttons: 'Filósofos', 'Adicionar', and 'Carrega Banco'.

Cidades	Municípios
Rio de Janeiro	87
São Paulo	356
Minas Geraus	56

Filósofos Adicionar Carrega Banco

Exercício

- Criar uma tabela no MySQL de nome agenda, com os campos para nome (varchar 50) e telefone (varchar 11)
- Criar um formulário contendo os campos para nome e telefone e os botões Incluir, Excluir e Alterar
- Incluir uma Table que imprima todos os registros do bando de dados em ordem alfabética
- Ao clicar em um item da Tabela, os dados preenchem o formulário nos respectivos campos
- A cada registro alterado, incluído ou excluído, a tabela é atualizada

Obrigado!



www.cruzeirosul.edu.br