

## Aula 20

# Componentes GUI - Parte III

# JSlider

- O JSlider (ou controle deslizante) permite a um usuário selecionar um valor a partir de um intervalo de valores inteiros
- Os JSliders podem ser personalizados para exibir marcas de medidas principais e marcas de medidas secundárias
- Para se interagir com um JSlider pode-se utilizar
  - Mouse
  - Teclado
    - Setas para a Direita e Esquerda: incremento em uma unidade
    - Page Up e Page Down: incremento ou decremento de intervalos de valores

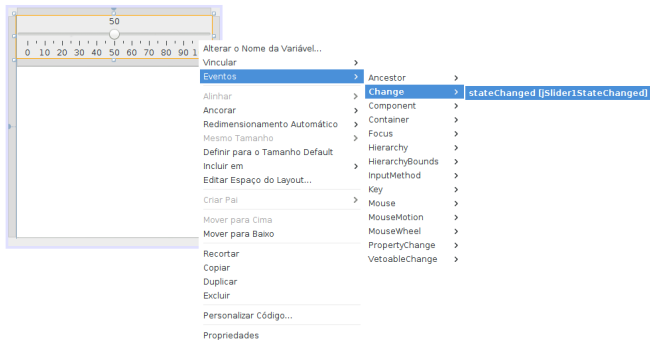
## Exemplo de criação e definição de propriedades de um JSlider

```
public Principal() {  
    initComponents();  
    JSlider slider = new JSlider();  
    slider.setSize(200, 200);  
    slider.setMajorTickSpacing(10);  
    slider.setMinorTickSpacing(5);  
    slider.setMaximum(100);  
    slider.setMinimum(0);  
    slider.setPaintTicks(true);  
    this.add(slider);  
}
```



## JSlider

- Podemos utilizar o `ChangeListener` para tratar o evento de alteração da posição do Marcador em um `JSlider`



# JSlider

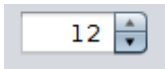
- Podemos utilizar o `ChangeListener` para tratar o evento de alteração da posição do Marcador em um `JSlider`

```
private void jSliderStateChanged(javax.swing.event.ChangeEvent evt) {  
    Font font = new Font("Serif", Font.PLAIN, jSlider.getValue());  
    jTextArea.setFont(font);  
}
```



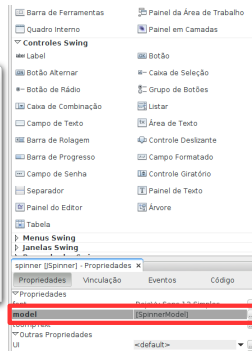
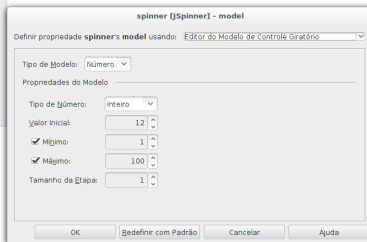
# JSpinner

- Nos JSpinners, o usuário pode selecionar valores ou digitando ou clicando nos botões com setas para cima e para baixo para aumentar ou diminuir os valores
- Pode-se especificar
  - O tipo de valores (ex: inteiro e decimal)
  - A quantidade de incremento dos valores ao acionar os botões do JSpinner
  - Valores mínimo e máximo



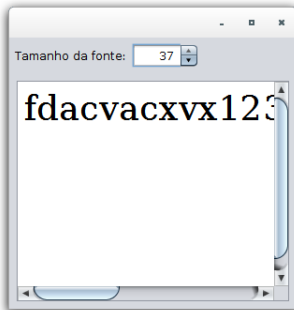
# JSpinner

- Podemos utilizar o `ChangeListener` para tratar o evento de alteração da posição do Marcador em um `JSlider`



# JSpinner

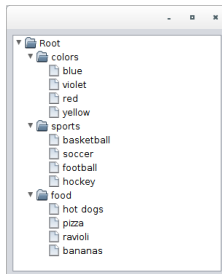
```
private void spinnerStateChanged(javax.swing.event.ChangeEvent evt) {  
    Font font = new Font("Serif", Font.PLAIN, (Integer)spinner.getValue());  
    JTextArea.setFont(font);  
}
```





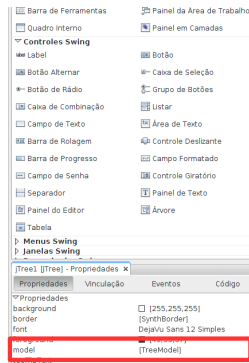
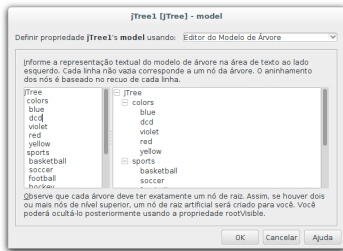
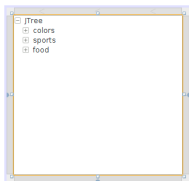
# JTree

- Um JTree é um componente que que exibe dados organizado hierarquicamente
- Ao selecionar um elemento de um JTree, pode-se exibir o caminho (pais e pais dos pais) desde a raiz até o nó selecionado



# JTree

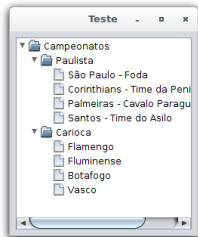
- Para editar um modelo na mão:



# JTree

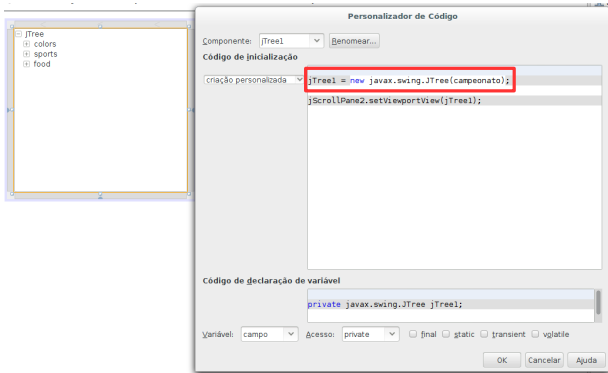
## Definindo o modelo via código:

```
14 public class Teste extends javax.swing.JFrame {
15
16     DefaultMutableTreeNode campeonato = new DefaultMutableTreeNode();
17
18     public Teste() {
19         campeonato = new DefaultMutableTreeNode("Campeonatos");
20
21         DefaultMutableTreeNode paulista = new DefaultMutableTreeNode("Paulista");
22         DefaultMutableTreeNode time = new DefaultMutableTreeNode("São Paulo - Foda");
23         paulista.add(time);
24         time = new DefaultMutableTreeNode("Corinthians - Time da Penitenciaría");
25         paulista.add(time);
26         time = new DefaultMutableTreeNode("Palmeiras - Cavalo Paraguaio");
27         paulista.add(time);
28         time = new DefaultMutableTreeNode("Santos - Time do Asilo");
29         paulista.add(time);
30
31         DefaultMutableTreeNode carioca = new DefaultMutableTreeNode("Carioca");
32         time = new DefaultMutableTreeNode("Flamengo");
33         carioca.add(time);
34         time = new DefaultMutableTreeNode("Fluminense");
35         carioca.add(time);
36         time = new DefaultMutableTreeNode("Botafogo");
37         carioca.add(time);
38         time = new DefaultMutableTreeNode("Vasco");
39         carioca.add(time);
40
41         campeonato.add(paulista);
42         campeonato.add(carioca);
43
44         initComponents();
45     }
46 }
```



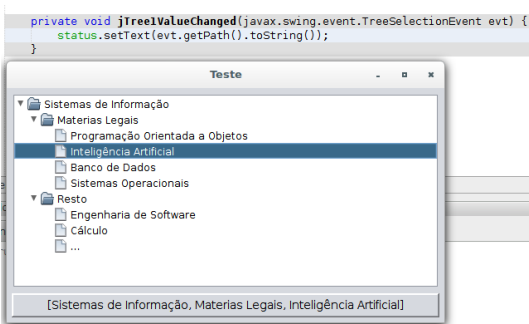
## JTree

- O a raiz do DefaultMutableTreeNode tem que ser passada no Construtor do JTree



## JTree

- Para realizar uma ação quando o usuário clica em um elemento da árvore, deve-se utilizar a interface `TreeSelectionListener`

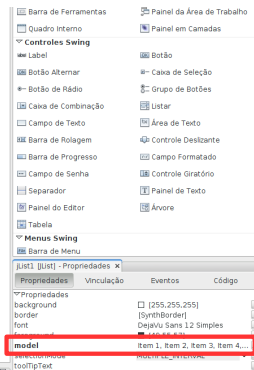
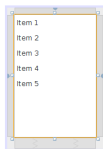


# JList

- Um JList apresenta ao usuário um grupo de itens
- A lista, geralmente, não aumenta nem diminui seu tamanho
- No caso de haver mais itens dos que os exibidos na tela, normalmente utiliza-se uma barra de rolagem para se mover entre todos os itens
- As listas permitem selecionar um único elemento ou múltiplos elementos

# JList

## ● Editando os elementos de uma lista via NetBeans

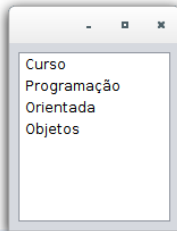


Resultados da Pesquisa Saída x

# JList

- Inserindo elementos na lista via código

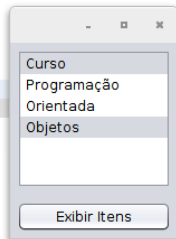
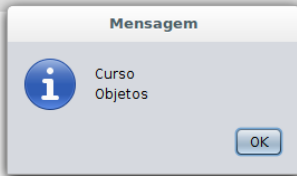
```
public Principal() {  
    initComponents();  
  
    DefaultListModel model = new DefaultListModel();  
    model.addElement("Curso");  
    model.addElement("Orientada");  
    model.addElement("Objetos");  
    model.add(1, "Programação");  
  
    jList1.setModel(model);  
}
```





- Obtendo os elementos selecionados de uma lista

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    Object[] itens = jList1.getSelectedValues();  
    String texto = "";  
    for(Object item : itens){  
        texto += item.toString() + "\n";  
    }  
    JOptionPane.showMessageDialog(null, texto);  
}
```



- Exemplo de código para remover um elemento selecionado em uma lista

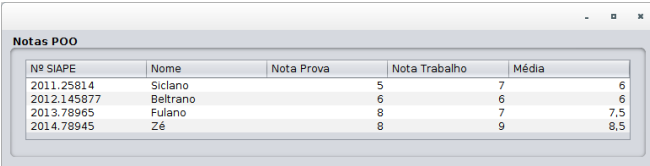
```
private void bRemoverActionPerformed(java.awt.event.ActionEvent evt) {  
    int id = lItens.getSelectedIndex();  
    DefaultListModel<String> model = (DefaultListModel) lItens.getModel();  
    model.removeElementAt(id);  
}
```

- Exemplo de código para adicionar um elemento dinamicamente à uma lista

```
private void bAdicionarActionPerformed(java.awt.event.ActionEvent evt) {  
    DefaultListModel<String> model = (DefaultListModel)lItens.getModel();  
    String novoItem = JOptionPane.showInputDialog("Novo item:");  
    model.addElement(novoItem);  
}
```

# JTable

- Um JTable é utilizada para exibir dados em uma maneira bidimensional → linhas e colunas
- Cada dado corresponde a uma célula
- Pode-se apenas exibir o conteúdo ou permitir que o usuário edite o conteúdo das dados

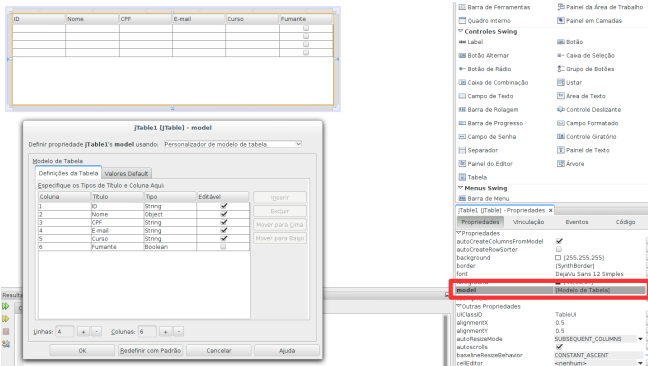


The screenshot shows a Java Swing window titled "Notas POO". Inside the window is a JTable with 5 columns: "Nº SIAPE", "Nome", "Nota Prova", "Nota Trabalho", and "Média". The table contains 4 rows of data representing student records.

Nº SIAPE	Nome	Nota Prova	Nota Trabalho	Média
2011.25814	Siclano	5	7	6
2012.145877	Beltrano	6	6	6
2013.78965	Fulano	8	7	7,5
2014.78945	Zé	8	9	8,5

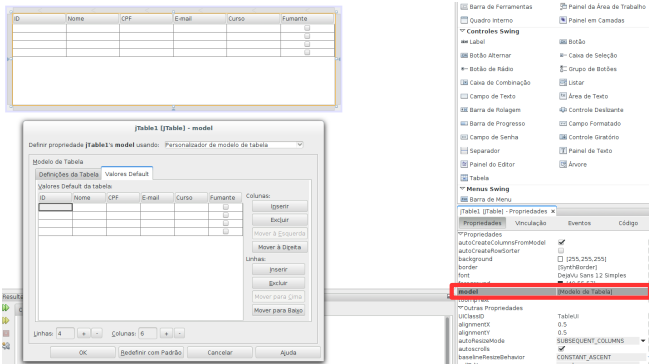
# JTable

- Editando os campos e o conteúdo de um JTable via NetBeans



# JTable

- Editando os campos e o conteúdo de um JTable via NetBeans



# JTable

## ● Criando o JTable, colunas e conteúdo via código

```
private JTable tabela;  
  
public Principal() {  
    initComponents();  
  
    String[] colunas = {"ID","Nome","CPF","E-mail","Curso", "Fumante"};  
  
    Object[][] dados = {  
        {new Integer(1), "Rafael Geraldeli Rossi", "346.048.787-98", "rafael.g.rossi@ufms.br", "Sistemas", new Boolean(false)},  
        {new Integer(2), "Ricardo Marcacini", "581.584.978-98", "ricardo.marcacini@ufms.br", "Cosmetologia", new Boolean(true)},  
    };  
  
    tabela = new JTable(dados, colunas);  
    tabela.setSize(new Dimension(700,300));  
    tabela.getColumnModel().getColumn(0).setPreferredWidth(30);  
    tabela.getColumnModel().getColumn(1).setPreferredWidth(300);  
    tabela.getColumnModel().getColumn(2).setPreferredWidth(250);  
    tabela.getColumnModel().getColumn(3).setPreferredWidth(400);  
    tabela.getColumnModel().getColumn(4).setPreferredWidth(200);  
    tabela.getColumnModel().getColumn(5).setPreferredWidth(100);  
    this.add(tabela);  
}
```

# JTable

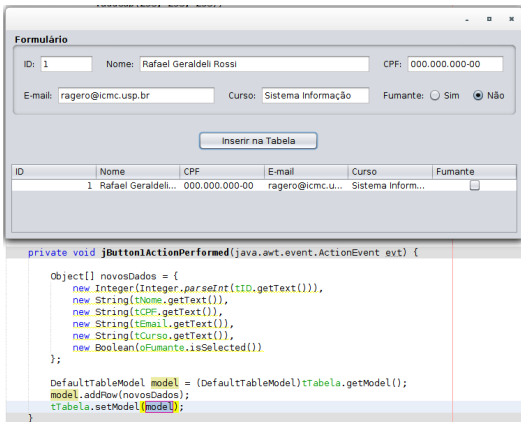
- Forma alternativa é criar um modelo de tabelas (DefaultTableModel)

```
public class Principal extends javax.swing.JFrame {  
  
    private JTable tabela;  
  
    public Principal() {  
  
        initComponents();  
  
        String [] colunas = {"ID", "Nome", "CPF", "E-mail", "Curso", "Fumante"};  
        Object [][] dados = {  
            { new Integer(1), "Rafael G. Rossi", "346.789.789-10", "rafael.g.rossi@gmail.com", "Sistema", new Boolean(false)},  
            { new Integer(2), "Ricardo M. Marcacini", "471.587.157-30", "ricardo.marcacini@ufms.br", "Manicure", new Boolean(true)},  
            { new Integer(3), "Zé Ruela", "358.789.547-10", "ruela@aems.com.br", "Migué", new Boolean(true)}  
        };  
  
        DefaultTableModel model = new DefaultTableModel(dados,colunas);  
  
        tabela = new JTable();  
        tabela.setSize(600, 300);  
        tabela.setModel(model);  
  
        this.add(tabela);  
  
    }  
}
```



# JTable

- Adicionando conteúdo em um JTable via código



Formulário

ID: 1 Nome: Rafael Geraldelli Rossi CPF: 000.000.000-00

E-mail: ragero@icmc.usp.br Curso: Sistema Informação Fumante: ☐ Sim ☒ Não

Inserir na Tabela

ID	Nome	CPF	E-mail	Curso	Fumante
1	Rafael Geraldelli...	000.000.000-00	ragero@icmc.u...	Sistema Inform...	<input type="checkbox"/>

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    Object[] novosDados = {  
        new Integer(Integer.parseInt(tID.getText())),  
        new String(tNome.getText()),  
        new String(tCPF.getText()),  
        new String(tEmail.getText()),  
        new String(tCurso.getText()),  
        new Boolean(tFumante.isSelected())  
    };  
    DefaultTableModel model = (DefaultTableModel)tTabela.getModel();  
    model.addRow(novosDados);  
    tTabela.setModel(model);  
}
```

# JTable

- Obtendo os valores de uma tabela

ID	Nome	CPF	E-mail	Curso	Fumante
1	Rafael G. Rossi	346.789.789-10	rafael.g.rossi@gmail...	Sistema	<input type="checkbox"/>
2	Ricardo M. Marcacini	471.587.157-30	ricardo.marcacini@u...	Manicure	<input checked="" type="checkbox"/>
3	Zé Ruela	358.789.547-10	ruela@aems.com.br	Migüé	<input checked="" type="checkbox"/>

Imprimir Valores na Tela

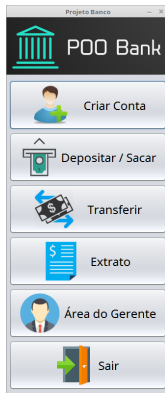
```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  
    TableModel model = jTable1.getModel();  
    int coluna = model.getColumnCount();  
    int linhas = model.getRowCount();  
  
    for(int lin=0;lin<linhas;lin++){  
        for(int col=0;col<coluna;col++){  
            System.out.print(model.getValueAt(lin, col) + "\\t");  
        }  
        System.out.println();  
    }  
}
```

Depurador: Teste (run)

```
run:  
1 Rafael G. Rossi 346.789.789-10 rafael.g.rossi@gmail.com Sistema null  
2 Ricardo M. Marcacini 471.587.157-30 ricardo.marcacini@u... Manicure true  
3 Zé Ruela 358.789.547-10 ruela@aems.com.br Migüé true
```

## Exercício

- Modificar o Projeto Banco fazendo a interação com o usuário por meio de interface gráfica
- O menu inicial deverá ficar da seguinte forma:



## Exercício

### Interface Gráfica para adicionar uma conta



- Acrescentar os campos Estado, Cidade e Bairro na classe Proprietário
- Fazer a programação dos botões Cadastrar e Cancelar

## Exercício

### Interface Gráfica para sacar e depositar

The image shows a graphical user interface window titled "Sacar/Depositar". The window is divided into two main sections. The left section, titled "Tipo de Operação", contains a list box with two options: "Sacar" (selected) and "Depositar". The right section, titled "Dados da Conta", contains two text input fields: "Nº da Conta:" and "Valor:". Below these sections are two buttons: "Processar" and "Cancelar".

\*Fazer a programação dos botões Processar e Cancelar

# Exercício

## Interface Gráfica para transferir

The screenshot shows a window titled "Transferir" with a standard macOS-style title bar (minimize, maximize, close buttons). The window contains a form titled "Informações da Transferência". The form is organized into three main sections: "Conta de Origem", "Conta de Destino", and "Valor da Transferência". Each section has a label and a text input field. The "Conta de Origem" section has two input fields: "Nº da Conta:" and "Proprietário:". The "Conta de Destino" section also has two input fields: "Nº da Conta:" and "Proprietário:". The "Valor da Transferência" section has a single input field. At the bottom right of the window, there are two buttons: "Transferir" and "Cancelar".

\*Fazer a programação dos botões Transferir e Cancelar

# Exercício

## Interface Gráfica para o Extrato

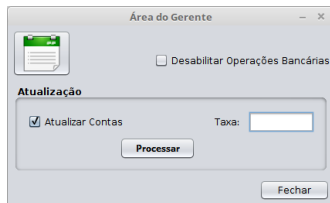
The screenshot shows a window titled "Extrato" with a standard Mac OS-style title bar (minimize, maximize, close buttons). The window is divided into two main sections. The top section, labeled "Conta", contains a text field labeled "Nº da Conta:" and a button labeled "Procurar". The bottom section, labeled "Extrato", contains a label "Proprietário:" followed by a large rectangular area. This area contains a table with three columns: "Data", "Descrição", and "Valor". Below the table is a label "Saldo:". At the bottom right of the window is a button labeled "Fechar".

Data	Descrição	Valor
------	-----------	-------

\*Fazer a programação do botão Fechar

## Exercício

### Interface Gráfica para a área do gerente



- Fazer a programação dos botões Processar e Cancelar
- O ícone no canto superior esquerdo é para lista os proprietários e saldos dados contas do banco
- A caixa de texto e o botão processar só devem ser habilitados quando o usuário selecionar a opção "Atualizar Contas"



# Material Complementar

- How to Use Sliders

https:

[//docs.oracle.com/javase/tutorial/uiswing/components/slider.html](https://docs.oracle.com/javase/tutorial/uiswing/components/slider.html)

- How to Use Spinners https:

[//docs.oracle.com/javase/tutorial/uiswing/components/spinner.html](https://docs.oracle.com/javase/tutorial/uiswing/components/spinner.html)

- How to Use Trees

https:

[//docs.oracle.com/javase/tutorial/uiswing/components/tree.html](https://docs.oracle.com/javase/tutorial/uiswing/components/tree.html)

# Material Complementar

- How to Use Lists

https:

[//docs.oracle.com/javase/tutorial/uiswing/components/list.html](https://docs.oracle.com/javase/tutorial/uiswing/components/list.html)

- How to Use Combo Boxes

https:

[//docs.oracle.com/javase/tutorial/uiswing/components/combobox.html](https://docs.oracle.com/javase/tutorial/uiswing/components/combobox.html)

- How to Use Tables

https:

[//docs.oracle.com/javase/tutorial/uiswing/components/table.html](https://docs.oracle.com/javase/tutorial/uiswing/components/table.html)

# Programação Orientada a Objetos

<http://lives.ufms.br/moodle/>

Rafael Geraldeli Rossi  
rafael.g.rossi@ufms.br

Slides baseados em [Deitel and Deitel, 2010]

## Referências Bibliográficas I



Deitel, P. and Deitel, H. (2010).

*Java: How to Program.*

How to program series. Pearson Prentice Hall, 8th edition.