

Biblioteca Java Swing



Prof. Dr. João Paulo Lemos Escola
Copyright© 2022

Tópicos da aula

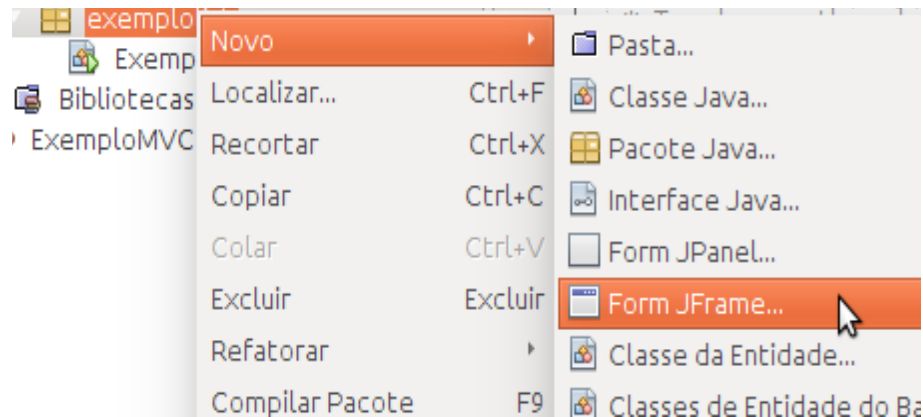
- Classes de componentes gráficos;
- Desenvolvimento de UI no NetBeans;
- Eventos.

Introdução

- Swing é a biblioteca gráfica padrão da linguagem Java para criação de aplicações gráficas Desktop;
- Nesta aula vamos aprender seus principais componentes e começar o desenvolvimento de aplicações para PC.

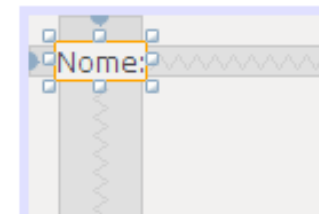
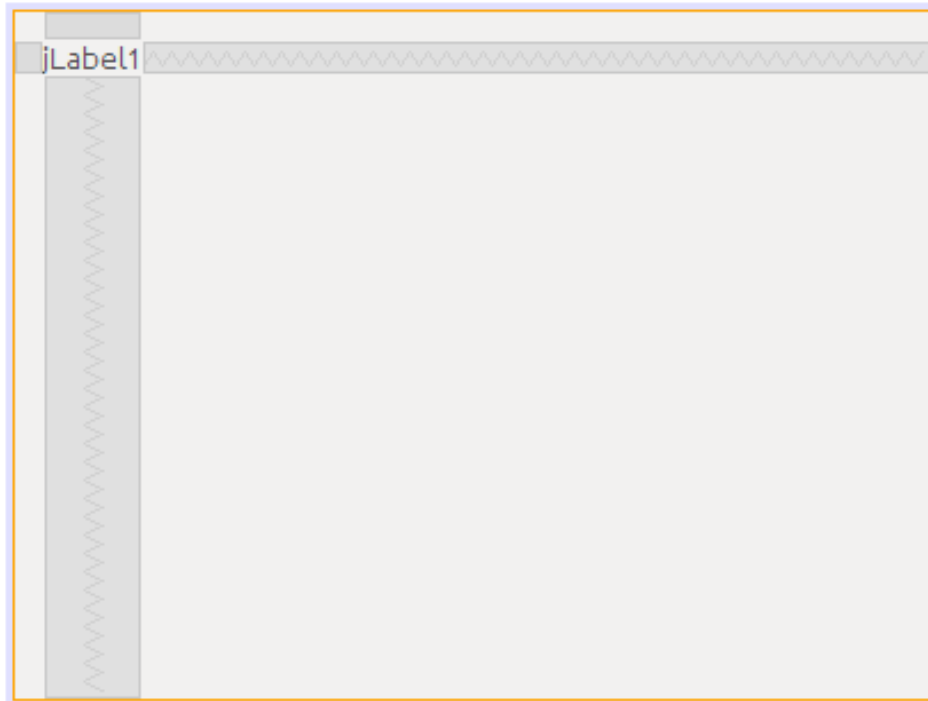
Classe JFrame

- Classe que implementa uma janela gráfica em Java:



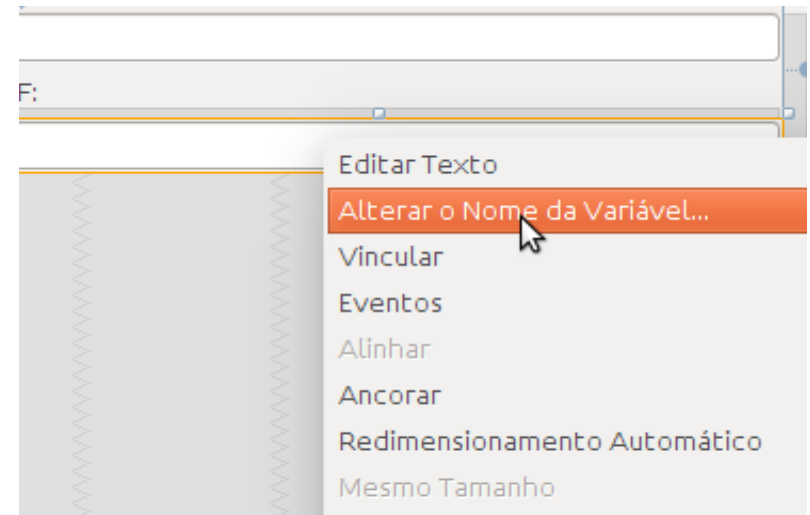
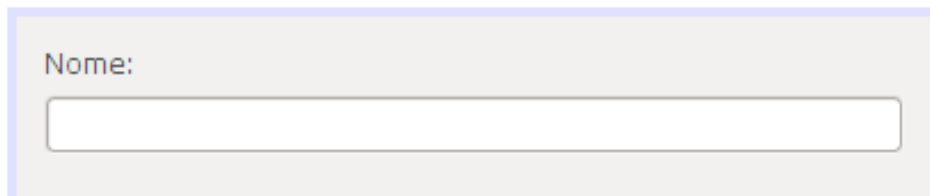
Componente JLabel

- Label é um rótulo em um JFrame;
- Pressione F2 para renomeá-lo e Enter pra finalizar;



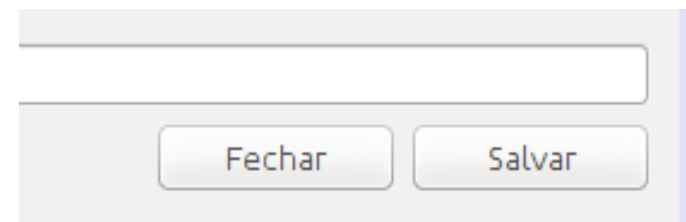
Componente JTextField

- Campo de texto;
- Com a tecla F2 altere o conteúdo ou deixe sem texto;
- Ajuste sua largura com a janela;
- Clicando com o botão direito no campo de texto, podemos alterar o nome da variável que representa o componente no código fonte.



Componente JButton

- Componente que representa um botão;
- É possível ajustar sua largura, nome da variável e o texto do botão da mesma forma que os outros componentes.



Codificando evento no botão

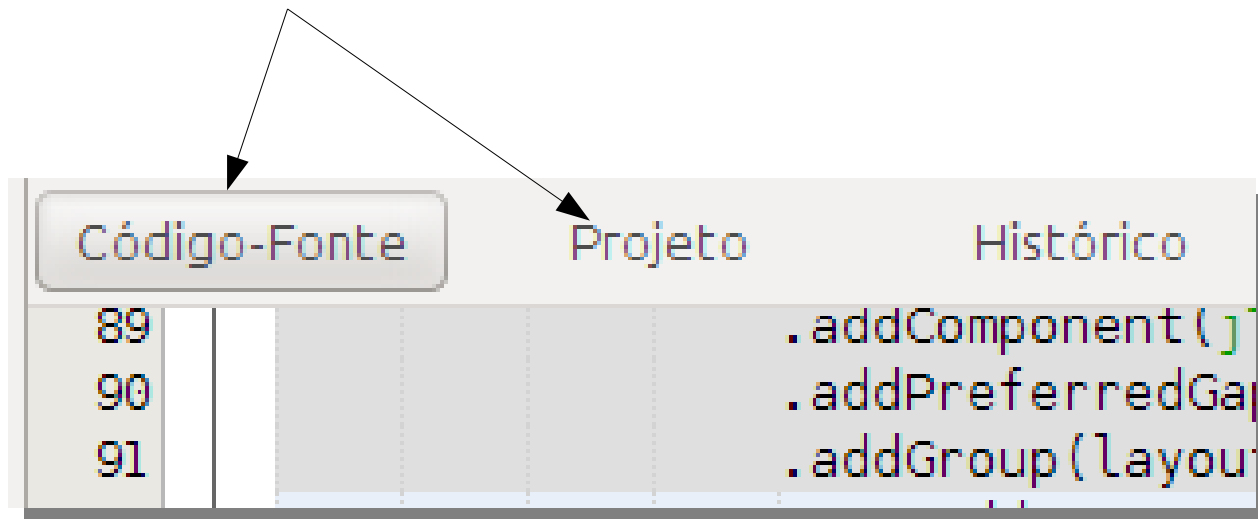
- Ajuste o nome da variável do botão Fechar para btnFechar;
- Para codificar o evento do botão, dê um duplo clique no botão e será direcionado ao código fonte:

```
private void btnFecharActionPerformed(java.awt.event.ActionEvent evt) {  
    System.exit(0); // fecha o programa  
}
```

- As linhas em cinza só podem ser alteradas pelo NetBeans;
- Execute o projeto e clique em Fechar para testar a funcionalidade.

Alternando entre o código e o design

- Os botões abaixo possibilitam alternar do código para o desenho da janela e vice-versa:



Botão Salvar

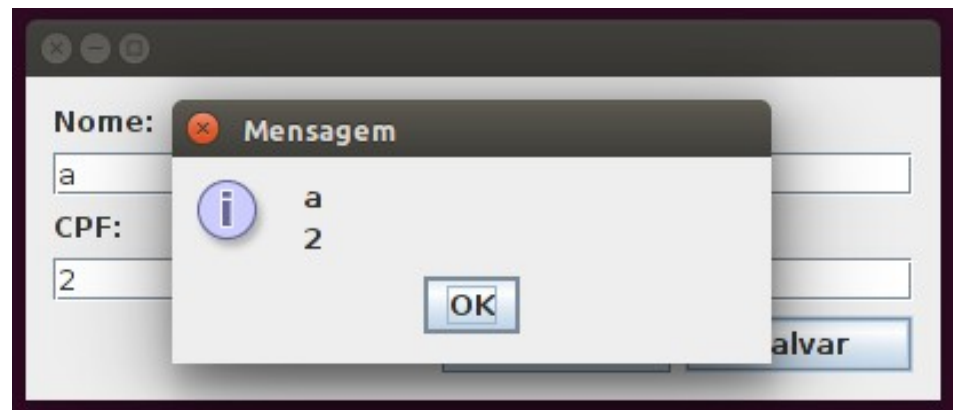
- Altere os nomes dos campos de texto para txtNome e txtCpf;
- Adicione o código do botão btnSalvar:

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    String nome = txtNome.getText();  
    String cpf = txtCpf.getText();  
  
    JOptionPane.showMessageDialog(rootPane, nome+"\n"+cpf);  
}
```

Execute o projeto

- Se o seu JFrame não aparecer na tela, falta configurar o método *main* para exibi-lo:

```
public static void main(String[] args) {  
    // instancia a janela e a exibe  
    new PessoaCadastro().setVisible(true);  
}
```



A13ex01.jar

- Crie um programa com os campos nome e cpf;
- Adicione o botão “Salvar”;
- Mostre os dados digitados quando o usuário clicar no botão “Salvar”.

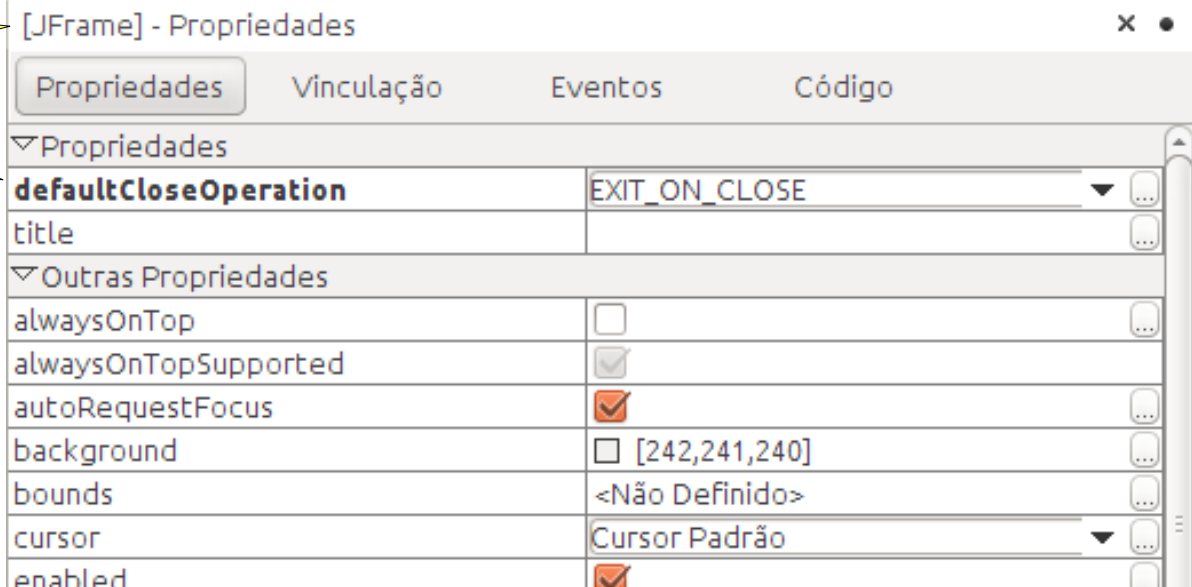
Propriedades do componente

- Clique no JFrame para selecioná-lo;
- No NetBeans, podemos ajustar as propriedades do componente (que está selecionado) pelas opções que aparecem do lado direito da janela;

Tipo do componente

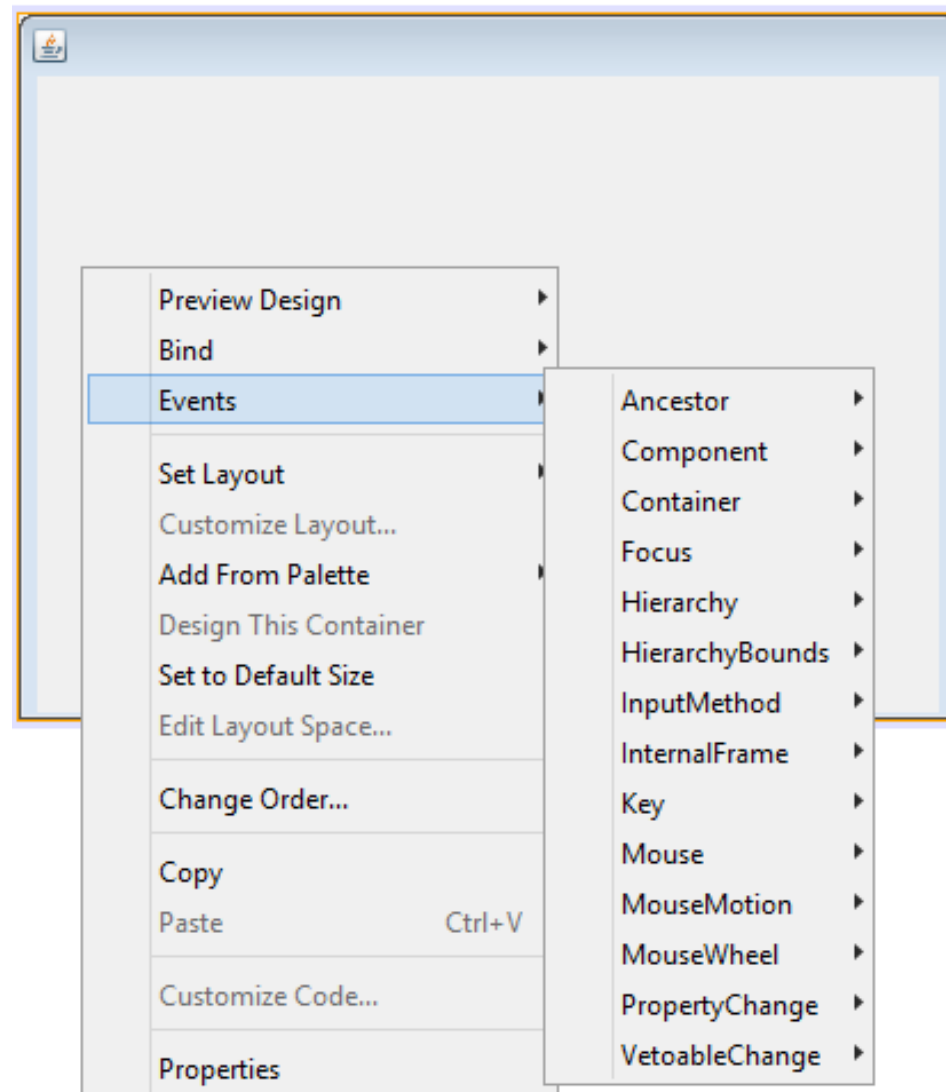
Operação padrão quando fechar a janela

Título da janela



Eventos

- Podemos configurar todo tipo de evento em qualquer tipo de componente da biblioteca Swing;
- Exemplo: clique no botão, foco na caixa de texto, mouse passando sobre o botão, fechamento, minimização etc;
- Clique com o botão direito sobre o componente e selecione a opção “Evento”.



Eventos MouseEntered e MouseExited

- Vamos adicionar um evento a um JLabel;
- Quando o mouse passa sobre o rótulo, ele muda o texto apresentado.

```
private void lblTextoMouseEntered(java.awt.event.MouseEvent evt) {  
    lblTexto.setText("mouse sobre o rótulo");  
}
```

```
private void lblTextoMouseExited(java.awt.event.MouseEvent evt) {  
    lblTexto.setText("mouse fora do rótulo");  
}
```


A13ex02.jar

- Crie um programa com um rótulo;
- Ao passar o mouse sobre o rótulo, ele deve mudar o texto do rótulo.

A13ex03.jar

- Crie um programa que mostre uma mensagem no prompt quando os eventos seguintes forem executados:
 - Janela ativa;
 - Janela minimizada;
 - Janela restaurada;
 - Janela fechada.

A13ex04.jar

- Crie um programa com um campo de texto que apresente a mensagem “Dados gravados com sucesso” quando o usuário pressionar a tecla F12.

A13ex05.jar

- Crie um programa que altere o título da janela com o texto digitado em um campo de texto.

Componente JList

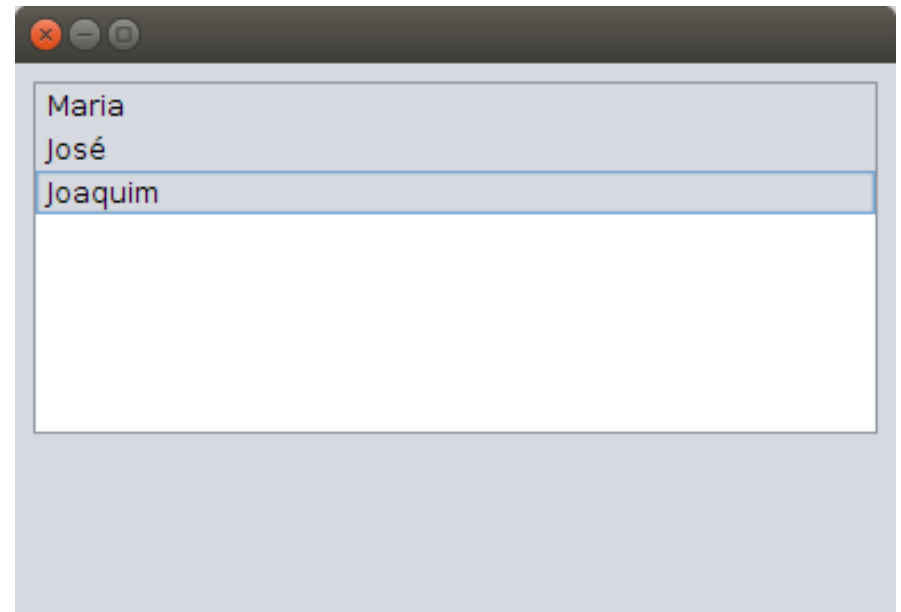
- Permite apresentar uma lista de dados:



Componente JList (cont.)

- Renomeie o JList para “lstPessoas”;
- Vamos preencher o JList utilizando um vetor:

```
public JList() {  
    initComponents();  
  
    // vetor  
    String[] dados = {"Maria", "José", "Joaquim"};  
  
    // limpa o JList  
    lstPessoas.removeAll();  
  
    // adiciona os dados  
    lstPessoas.setListData(dados);  
}
```



Classe DefaultListModel


- Classe que representa um “modelo de lista” que pode ser utilizado em componentes como JList e JTable (veremos à frente);
- Representa um modelo de lista vazia, que deve ser utilizado para preencher o componente gráfico (JList ou JTable);
- Não podemos substituir os dados de listagens gráficas diretamente, somente por meio de modelos de lista.

JList preenchimento dinâmico

- Adicionando um campo de texto, vamos incluir o valor digitado no JList em tempo de execução:

```
private void txtValorFocusGained(java.awt.event.FocusEvent evt) {  
    txtValor.setText("");  
}  
  
private void txtValorFocusLost(java.awt.event.FocusEvent evt) {  
    if (txtValor.getText().equals(""))  
        txtValor.setText("Digite um valor e clique em Adicionar");  
}
```

Código do botão
Adicionar



```
// pega a lista atual  
ListModel listaAtual = lstPessoas.getModel();  
  
// cria uma lista nova  
DefaultListModel listaNova = new DefaultListModel();  
  
// percorre a lista atual e copia para a lista nova  
for (int i=0; i<listaAtual.getSize(); i++)  
    listaNova.addElement(listaAtual.getElementAt(i));  
  
// adiciona o novo elemento na lista nova  
listaNova.addElement(txtValor.getText());  
  
// substitui a lista da janela pela lista nova  
lstPessoas.setModel(listaNova);
```


A13ex06.jar

- Crie um programa que apresente um JList de nomes de pessoas;
- Adicione um campo de texto para permitir incluir novos nomes na lista.

Componente JTable

- Permite criar uma tabela na janela gráfica;
- Adicione a tabela pelo botão



Title 1	Title 2	Title 3	Title 4

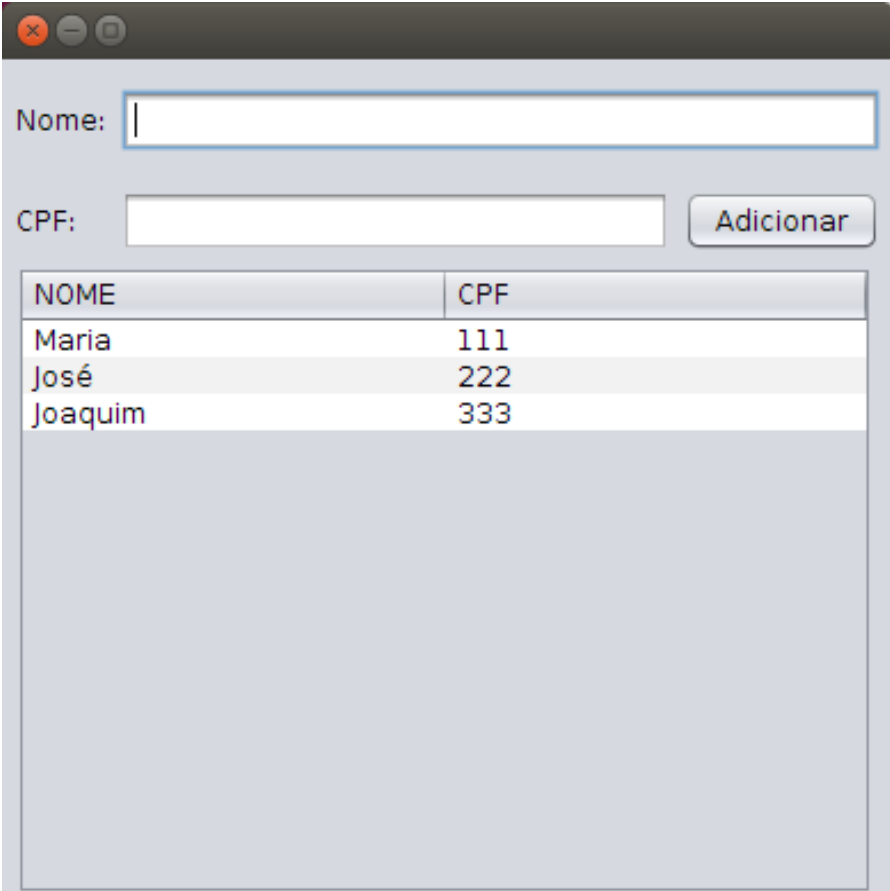
JTable – preenchendo

- Para preencher a tabela vamos precisar de um vetor de valores das colunas e uma matriz de dados:

```
public JTable() {  
    initComponents();  
  
    // rótulos das colunas da tabela  
    String[] nomesColunas = new String[]{"NOME", "CPF"};  
  
    // dados da tabela  
    String[][] dados = {"Maria", "111"}, {"José", "222"}, {"Joaquim", "333"}; }  
  
    // inserindo o modelo na tabela  
    tabela.setModel(new DefaultTableModel(dados, nomesColunas));  
}
```

JTable – preenchendo (cont.)

- Veja como ficou nossa tabela ao executarmos a janela:



The screenshot shows a Java Swing window with a light gray background. At the top, there are standard window control buttons (red, yellow, and green). Below the title bar, there is a form with two text input fields. The first field is labeled 'Nome:' and is empty. The second field is labeled 'CPF:' and is also empty. To the right of the 'CPF:' field is a button labeled 'Adicionar'. Below the form is a JTable with two columns: 'NOME' and 'CPF'. The table contains three rows of data: 'Maria' with '111', 'José' with '222', and 'Joaquim' with '333'. The table has a light gray header and alternating row colors (white and light gray). Below the table is a large, empty rectangular area.

NOME	CPF
Maria	111
José	222
Joaquim	333

JTable - Preenchimento dinâmico

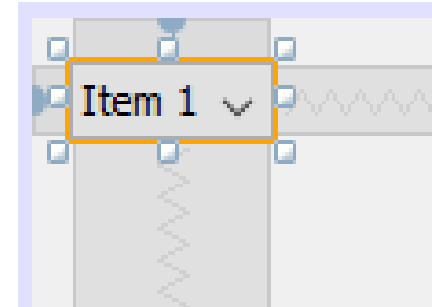
```
private void btnAdicionarActionPerformed(java.awt.event.ActionEvent evt) {  
    // modelo da nova tabela  
    DefaultTableModel tabelaNova = new DefaultTableModel();  
    // rótulos das colunas da tabela  
    tabelaNova.setColumnIdentifiers(new String[]{"NOME", "CPF"});  
    // modelo atual da tabela  
    TableModel tabelaAtual = tabela.getModel();  
    // recuperando dados da tabela atual  
    for (int i=0; i<tabelaAtual.getRowCount(); i++){  
        tabelaNova.addRow(  
            new String[]{  
                tabelaAtual.getValueAt(i, 0).toString(), // nome  
                tabelaAtual.getValueAt(i, 1).toString() // cpf  
            });  
    }  
    // adicionando os novos dados digitados  
    String[] linha = {txtNome.getText(), txtCpf.getText()};  
    tabelaNova.addRow(linha);  
    // atualizando a tabela  
    tabela.setModel(tabelaNova);  
}
```

A13ex07.jar

- Crie um programa que contenha um JTable e um campo de texto para nome e outro para CPF;
- Inicialize a tabela no construtor do JTable e permita inclusão de novos nomes/cpfs na tabela.

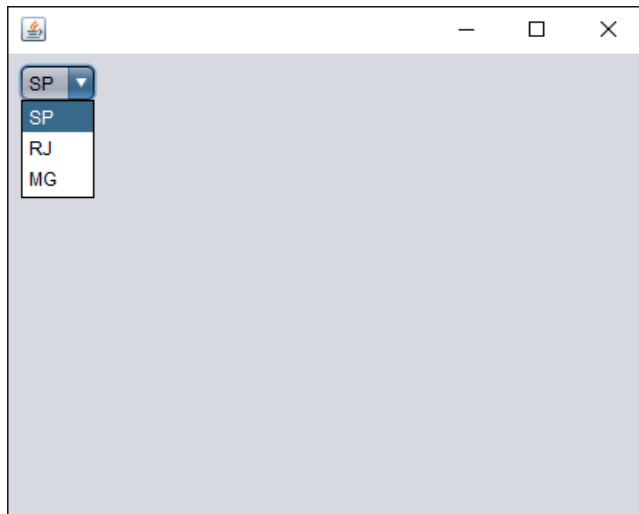
JComboBox

- Apresenta um conjunto de valores para que o usuário escolha;



Caixa de Combinação

Exemplo JComboBox



```
public Janela() {  
    initComponents();  
  
    cmbEstados.removeAllItems();  
    cmbEstados.addItem("SP");  
    cmbEstados.addItem("RJ");  
    cmbEstados.addItem("MG");  
}
```


Evento no JComboBox

```
private void cmbEstadosItemStateChanged(java.awt.event.ItemEvent evt) {  
    System.out.println(cmbEstados.getSelectedIndex());  
}
```

JComboBox com MVC

```
public Janela() {  
    initComponents();  
  
    // limpa o combo  
    cmbEstados.removeAllItems();  
    // busca os dados  
    estados = EstadoController.getLista();  
    // adiciona os dados no combo  
    for (Estado e: estados)  
        cmbEstados.addItem(e.getNome());  
}
```

```
private void cmbEstadosItemStateChanged(java.awt.event.ItemEvent evt) {  
    if (cmbEstados.getSelectedIndex() >= 0)  
        System.out.println(estados.get(cmbEstados.getSelectedIndex()).getNome());  
}
```

Cadastro de Cidades



Estado: SP ▼

Cidade:

Salvar

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    CidadeController.add(new Cidade(  
        0,  
        txtCidade.getText(),  
        estados.get(cmbEstados.getSelectedIndex())  
    ));  
}
```

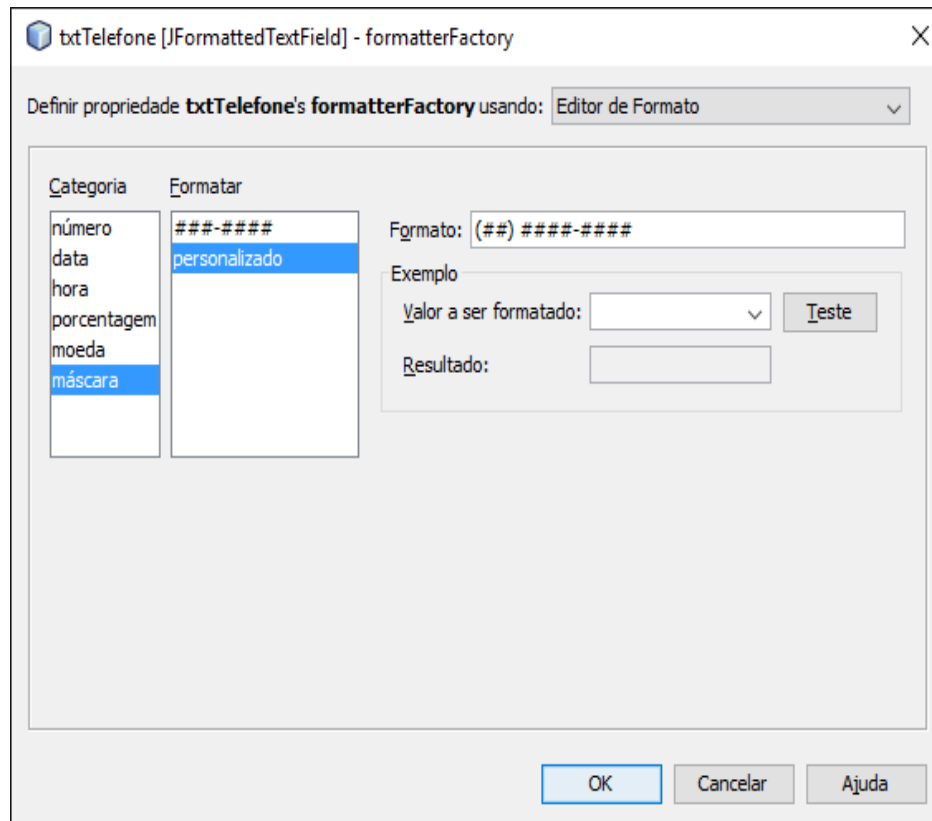
JTable – item selecionado

```
private void tblCidadesMouseClicked(java.awt.event.MouseEvent evt) {  
    System.out.println(cidades.get(tblCidades.getSelectedRow()).getNome());  
    System.out.println(cidades.get(tblCidades.getSelectedRow()).getEstado().getNome());  
}
```

Cidade	Estado
Barretos	SP
Belo Horizonte	MG

```
// gera o arraylist de cidades (importante para referenciar os dados)  
cidades = CidadeController.getList();  
// vetor com os nomes das colunas  
String[] colunas = CidadeController.getColunas();  
// matriz da tabela  
String[][] linhas = CidadeController.getLinhas();  
// gera o modelo a partir das linhas e colunas  
DefaultTableModel modelo = new DefaultTableModel(linhas, colunas);  
tblCidades.setModel(modelo);
```

JFormattedTextField



 Campo Formatado

Nome:

Telefone:

Exemplo campo Telefone

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    System.out.println(txtTelefone.getText());  
    // limpa o telefone para armazenar no bd  
    System.out.println(txtTelefone.getText().replaceAll("-", "").replaceAll("\\(", "").replaceAll("\\)", ""));  
}
```

ExemploCombo (run) x

```
run:  
(11) 1111-1111  
1111111111  
|
```

JCheckBox

☒ - Caixa de Seleção

☒ conferido

```
if (chkConferido.isSelected())  
    System.out.println("conferido!!!");
```

ExemploCombo (run) x

```
run:  
(11) 1111-1111  
1111111111  
(11) 1111-1111  
1111111111  
conferido!!!  
|
```

JRadioButton

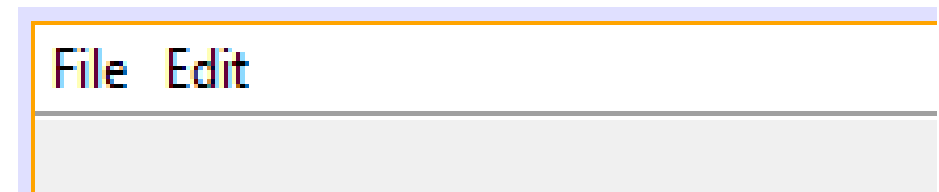
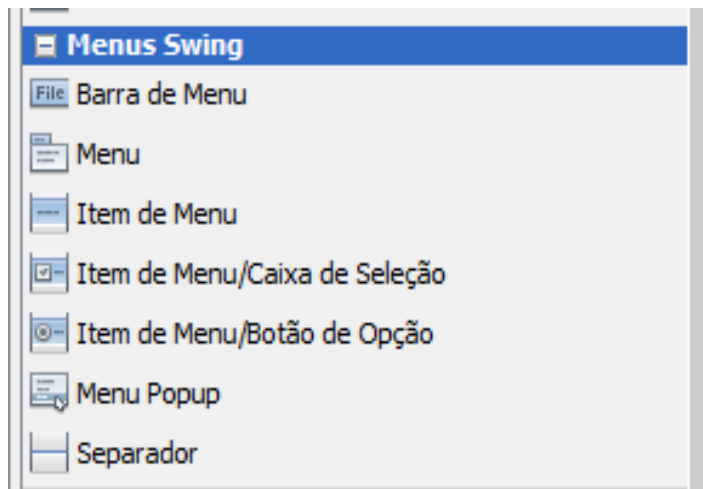
```
public PessoaCadastro() {  
    initComponents();  
  
    // cria um grupo de botões radio  
    ButtonGroup grupo = new ButtonGroup();  
    // adiciona os radios ao grupo  
    grupo.add(radFem);  
    grupo.add(radMasc);  
}
```

```
if (radFem.isSelected())  
    System.out.println("Feminino");  
else if (radMasc.isSelected())  
    System.out.println("Masculino");
```

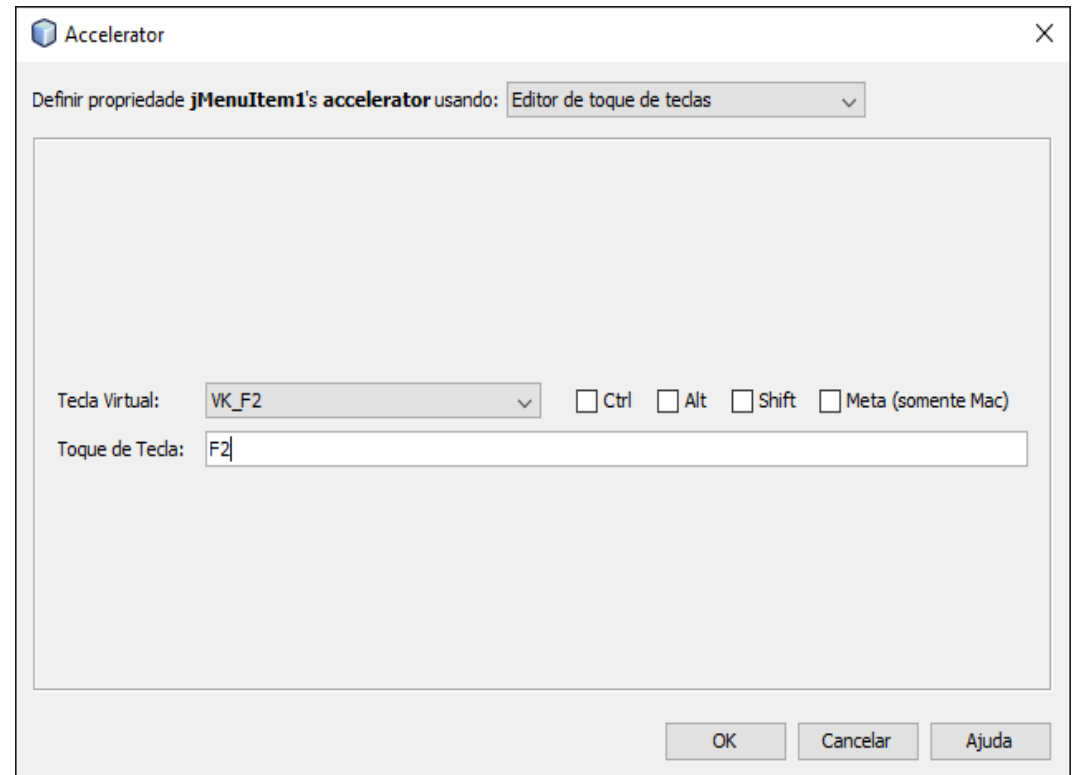
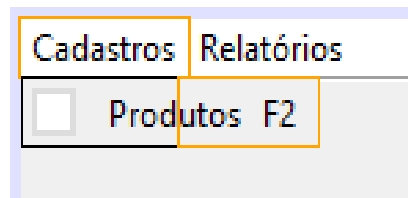
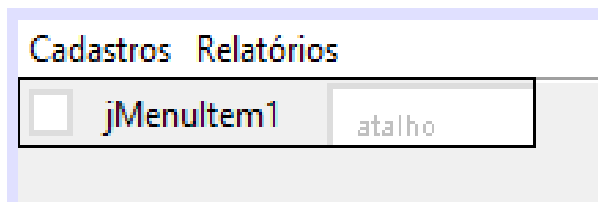
 – Botão de Rádio

Sexo: ☒ Masc ☐ Fem

JMenuBar



JMenuItem

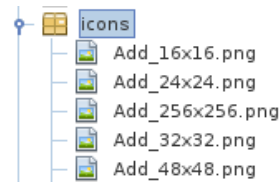


```
private void mnuProdutosActionPerformed(java.awt.event.ActionEvent evt) {  
    new Janela().setVisible(true);  
}
```

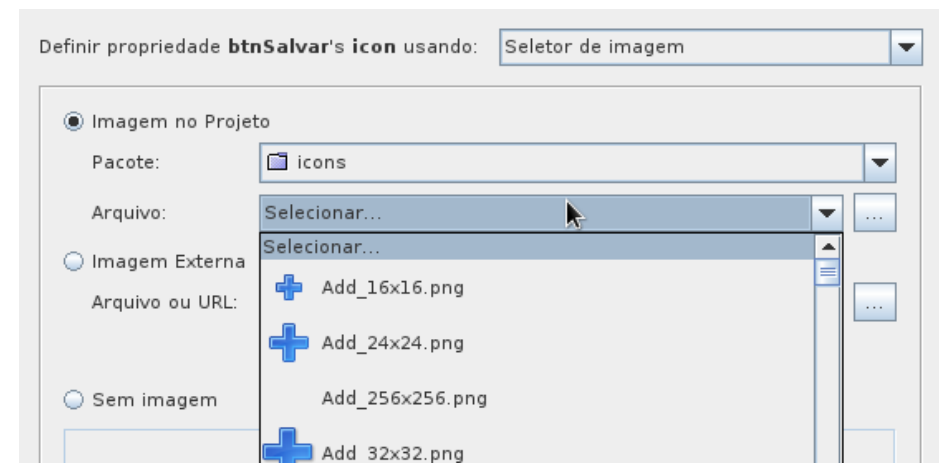
Ícones

Baixando ícones da internet, podemos incluir em nossos objetos: botões, menus etc;

Arraste a pasta dos ícones para a pasta “Pacotes de código fonte” do Netbeans para criar um pacote.
Renomeie o pacote se preferir:



Selecione o objeto e, no item “icon” das propriedades, clique no botão “...” e inclua o



A13ex08

Crie um programa em MVC que faça um CRUD de Produtos aplicando todas as funcionalidades da aula de hoje:

- Combo de Fabricantes;

- JTable com alteração dos campos ao clicar na linha;

- Campo formatado para data de vencimento;

- Checkbox para ativar/desativar do estoque;

- Radio para selecionar se o produto é perecível;

- Menu;

- Atalhos de teclado no menu;

- Ícones.

○ que aprendemos?

- Classes de componentes gráficos;
- Desenvolvimento de UI no NetBeans;
- Eventos.

Na próxima aula...

- Biblioteca iText;