

Interface Gráfica - Swing

'Botão de opção' (JRadioButton)

- ❑ O componente '**Botão de opção**', também conhecido como **RadioButton**, é utilizado para você **selecionar uma e somente uma** opção entre **várias** oferecidas em um **determinado** grupo;



Propriedades JRadioButton

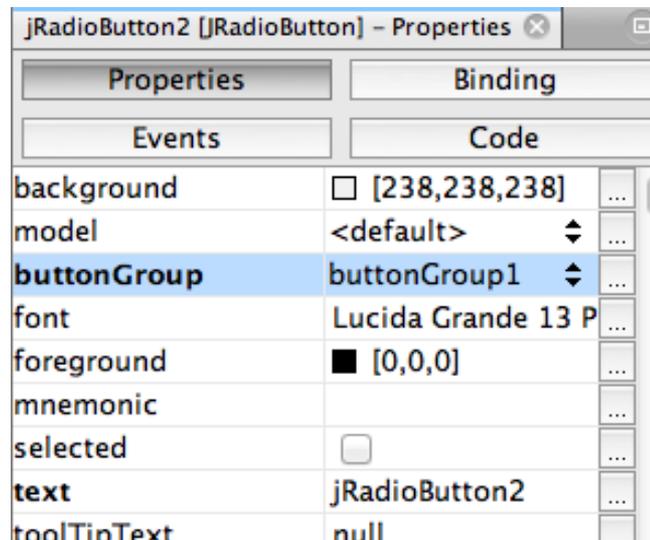
- ❑ **buttonGroup** – essa **propriedade** é utilizada para **agrupar** vários '**Botões de opção**' e **garantir** que apenas um **componente** do grupo poderá ser **selecionado**.
- ❑ **selected** – se o check box dessa **propriedade** for **selecionado**, significa que o **componente** '**Botão de opção**' que estiver sendo configurado será mostrado já marcado quando a **aplicação** for **executada**.
- ❑ **text** – essa **propriedade** é utilizada para **identificar** o **componente** com um **rótulo**.

'Grupo de botões' (ButtonGroup)

- ❑ Se você **clicar** em cada um **desses botões** você deverá **notar** que **todos** os **três botões** poderão ser **marcados**;
- ❑ Isso **acontece** porque **cada um** desses botões estão **isolados**, eles não **fazem** parte do mesmo **grupo**.
- ❑ **Grupo de botões' (ButtonGroup)**, tem a finalidade de **agrupar** e **gerenciar** o **estado** de cada um **deles** que **fizerem** parte do mesmo grupo, **garantido** que um **único botão** do grupo possa ser **selecionado** de cada vez.

'Grupo de botões' (ButtonGroup)

- ❑ **Arraste** um **componente 'Grupo de botões'** para dentro do **painel** na área de **projeto**.
- ❑ Ao fazer isso, no combo box da **propriedade buttonGroup** dos botões de **opção** será **acrescentado** um novo item **chamado 'buttonGroup1'**



'Caixa de combinação' (JComboBox)



- Esse **componente** é **utilizado** para **armazenar informações** em forma de uma **lista**, sendo cada **item** em uma **linha**, e **apenas** uma **seleção** será **permitida**;



Propriedades JComboBox

- ❑ **editable** – se o **check box** dessa **propriedade** for **selecionado**, significa que os itens do **componente** '**Caixa de combinação**' poderão ser **editados**;
- ❑ **maximumRowCount** – essa propriedade **informa** o **número máximo** de **itens** que poderão ser **visualizados** quando a **lista** do **componente** for aberta. Se o **número** de **itens** da lista **ultrapassar** esse valor, uma **barra** de **rolagem vertical** será **criada automaticamente**. O valor padrão é 8.
- ❑ **model** – essa **propriedade** é **utilizada** para **preencher** o **componente** com os itens necessários.

Aplicação JComboBox

- ❑ Agora veremos como **utilizar** esse **componente** em uma **aplicação prática** juntamente com um componente '**Rótulo**' para **mostrar** o **resultado** quando um **item** da **lista** for **selecionado**:
 - Crie um novo projeto (**Arquivo** → **Novo projeto**).
 - Salve seu projeto com o nome de '**ComponenteCaixaDeCombinacao**' em uma pasta de sua escolha.
 - Clique com o botão direito na aplicação e crie um novo **Formulário JFrame** com o nome de **CaixaDeCombinacao**.
 - A partir da **Paleta** de componentes, na seção **Controles Swing**, clique e arraste um componente '**Caixa de combinação**' (**JComboBox**) para a área do projeto.
 - Altere (usando a alça da direita) a largura desse componente para um tamanho um pouco maior do que o padrão.
 - Altere sua propriedade **maximumRowCount** para 4.

Aplicação JComboBox

- Para **preencher** a **lista** da **caixa de combinação**, clique na sua propriedade **model**. Na janela **apresentada**, apague os **itens existentes** e crie a seguinte lista de itens:

- **Nossa Empresa**
Produtos
Cadastro
Clientes
Fornecedores
Localização
Fale Conosco

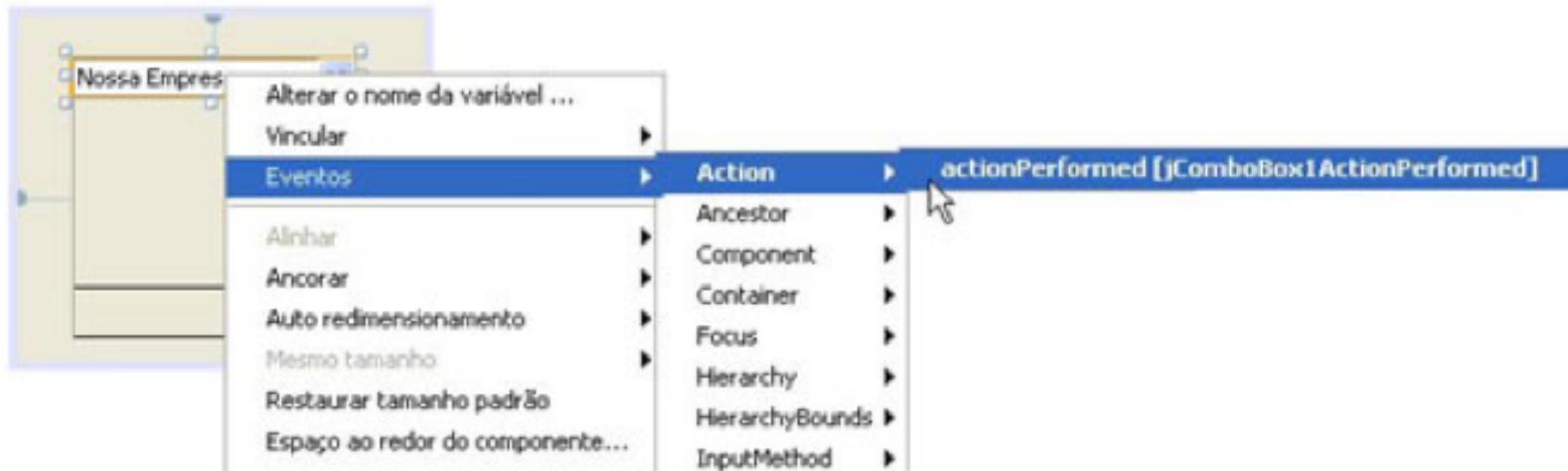


- Agora **arraste** um **componente 'Rótulo'** e insira-o bem **abaixo** da **caixa de combinação**. Crie uma **borda** para que ele **possa** ser **visualizado**, e **apague** o seu **conteúdo** (propriedade **text**)

Aplicação JComboBox

- ❑ Clique com o botão direito sobre a '**Caixa de combinação**'.

Eventos → Action → actionPerformed[jComboBox1ActionPerformed]

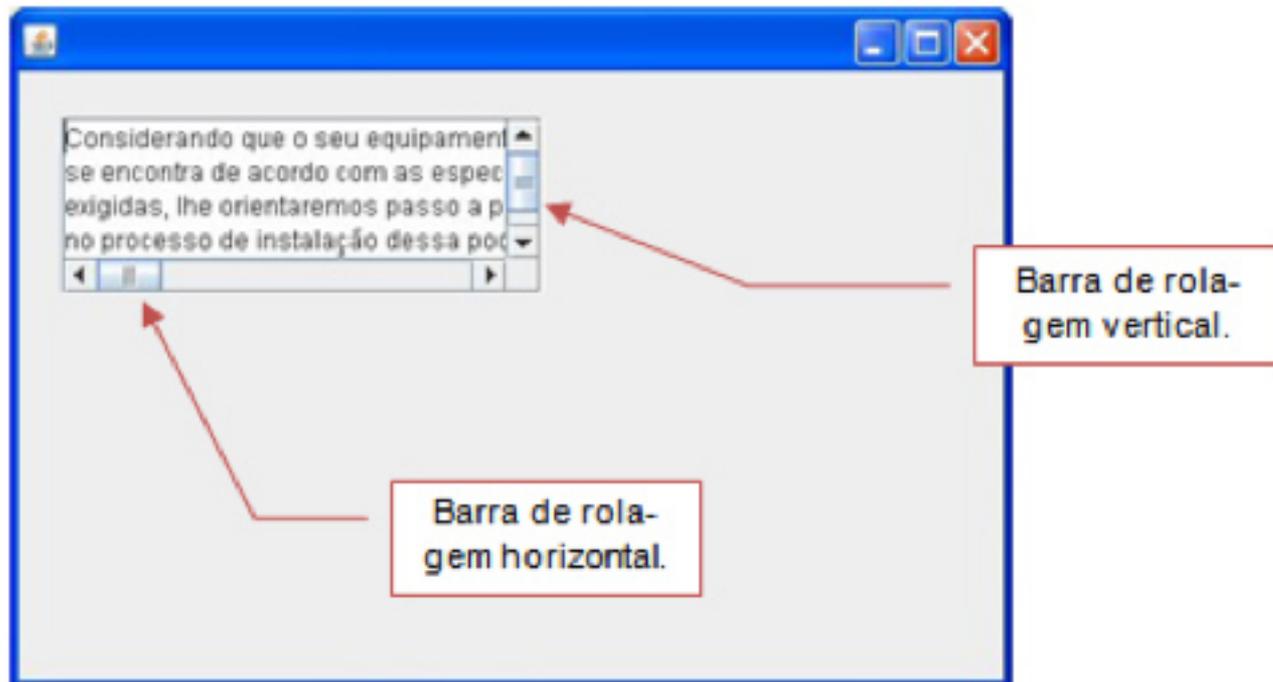


- ❑ Dentro do método criado para o evento, adicionem as seguintes linhas de código:

```
String a = (String)jComboBox1.getSelectedItem();  
jLabel1.setText(a);
```

'Área de Texto' (JTextArea)

- ❑ O componente '**Área de Texto**' é nada mais nada menos do que uma **caixa de texto** com maiores **recursos** do que uma caixa de texto **convencional (JTextField)**;

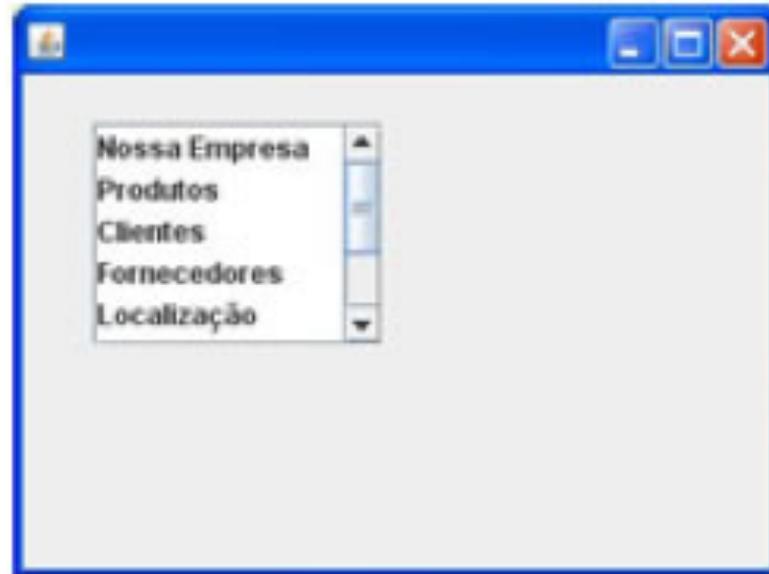


Propriedades do JTextArea

- ❑ **editable** – se esse check box estiver **marcado**, significa que o **componente** poderá ser **preenchido** e editado;
- ❑ **lineWrap** – se esse check box estiver marcado, significa que as **linhas** serão quebradas automaticamente quando o texto **atingir** a **margem direita** da caixa de texto;
- ❑ **text** – essa propriedade é **utilizada** para você **especificar** um **texto padrão** para o componente quando a aplicação for executada, ao invés de ficar em branco.

Componente 'Lista' (JList)

- Ele poderá ser **utilizado** para o **preenchimento** de **faixas de datas**, lista de **produtos**, lista dos **meses do ano**, lista de **profissões**, entre outras finalidades. Nesses casos, o usuário **seleciona** um ou mais itens da **lista**, **alternados** ou não.



Propriedades do JList

- ❑ **model** – essa **propriedade** é utilizada para **preencher** os **itens** do componente **Lista**, sendo um **item** por cada **linha**. Similar ao **ComboBox**.

- ❑ **selectionMode** – essa **propriedade** permite-lhe **selecionar** uma das **seguintes situações** para os itens da lista:
 - **SINGLE** – selecionando essa opção, o usuário só poderá **selecionar apenas** um **item** de cada vez.
 - **SINGLE_INTERVAL** – essa opção permite ao **usuário** **selecionar vários itens por vez**, mas somente em sequência, com o auxílio da tecla **Shift** ou **Ctrl**. Não permite múltiplas seleções alternadas.
 - **MULTIPLE_INTERVAL** – essa opção **permite** ao usuário **selecionar vários itens por vez em sequência ou alternados** com o auxílio da tecla **Shift** ou **Ctrl**.

Caixa de Seleção' (JCheckBox)

- ❑ O componente **Caixa de Seleção**, ou **CheckBox**, é um **pequeno** quadrado que **pode** ser **marcado** ou **desmarcado**.



Propriedades do JCheckBox

- ❑ **selected** – se você quiser que alguns dos seus **check boxes** apareçam já **marcados** em sua **aplicação**, utilize essa **propriedade** para fazer isso. Clique em cada **componente** e **marque** essa **opção**.
- ❑ **text** – essa **propriedade** é **utilizada** para **alterar** o **rótulo** do **check box**.

'Campo de senha' (JPasswordField)

- ❑ Muito **comum** em **qualquer interface** que **solicite** uma **senha** de **acesso**;
- ❑ É uma **caixa de texto** em que o **conteúdo** digitado **não é exibido** como em uma **caixa de texto** convencional (**JTextField**);

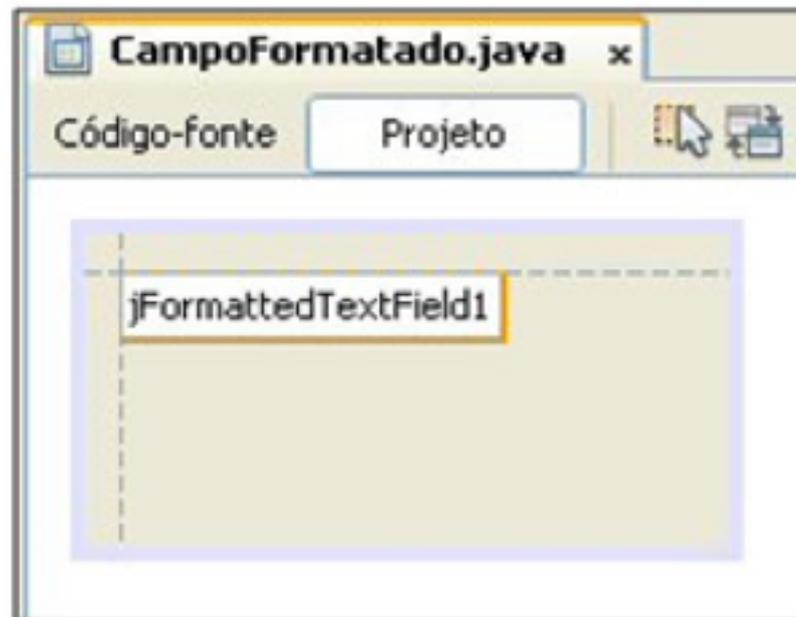


Propriedades JPasswordField

- ❑ **preferredSize** – permite **definir** os **valores** de **largura** e **altura** do **componente** em **número** de **pixels**.
- ❑ **caretPosition** – permite **definir** em que **posição** o **cursor estará**, quando a **aplicação** for **executada**.
- ❑ **caretColor** – funciona da **mesma forma** que outras **propriedades** de **definição** de **cores**, só que essa **determina** a **cor** do cursor **dentro** do **componente**.

'Campo formatado' (JFormattedText)

- Campos **formatados** servem para **especificar** a **formatação** de **caracteres** em um campo de texto (**TextField**). A **formatação** pode **servir**, por exemplo, para definir **formatos** de **datas**, **número** de **telefones**, **número** do **CEP**, **CPF** etc.

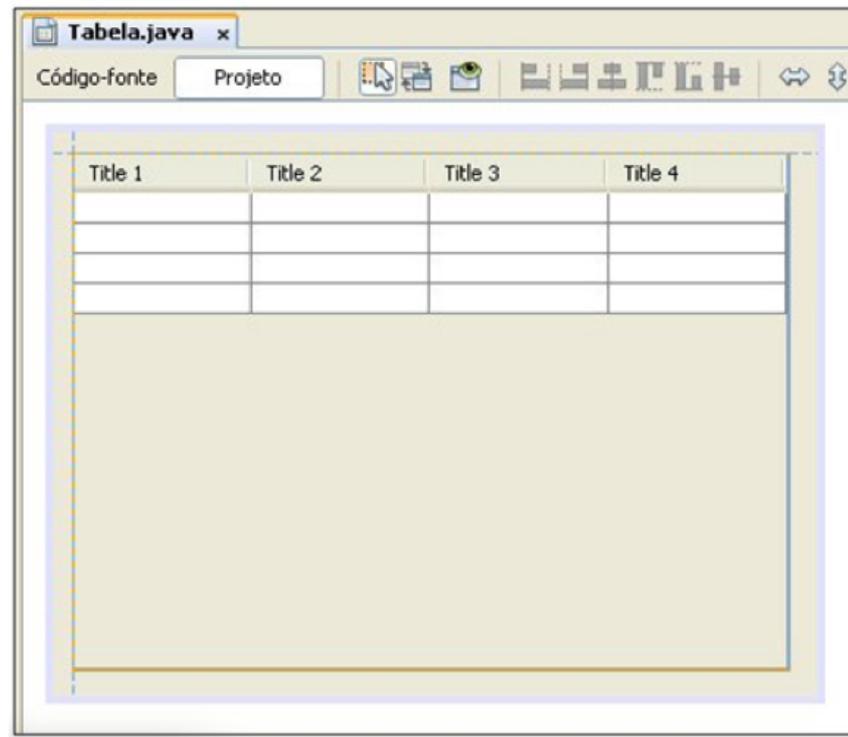


Propriedades

- ❑ **formatterFactory** – permite a **configuração** do **formato** a ser **estipulado** para o **componente**. Entre eles, temos:
 - **NÚMERO** – dispõe **diversos formatos** de **números**;
 - **DATA** – nessa **categoria**, você poderá **definir** entre os **formatos** de **data disponíveis**
 - **HORA** – essa categoria traz os **principais formatos** de **hora** utilizados, do mais **reduzido** ao mais **completo**.
 - **PORCENTAGEM** – basicamente, os **dois formatos** mais **utilizados**: **sem** casas decimais e **com duas** casas decimais
 - **MOEDA** – **formatos** de **moeda com** e **sem** os **centavos**.
 - **MÁSCARA** – nessa **categoria**, você **define** uma **máscara qualquer**. Os exemplos mais **comuns** incluem **número** de **CEP** e **CPF**.

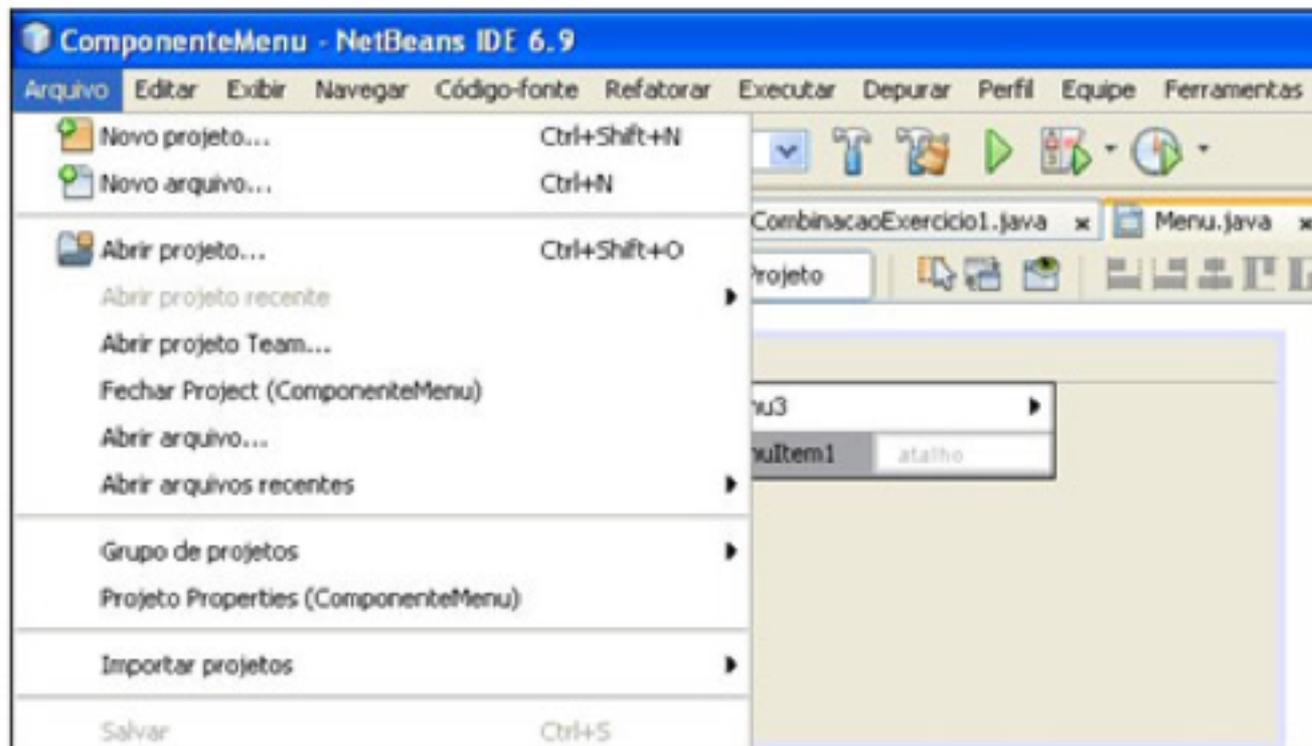
'Tabela' (JTable)

- Entende-se por **tabela** como sendo uma **espécie** de **planilha** cuja utilidade **principal** é **organizar** informações **provenientes** de alguma **fonte** de dados, como **bancos** de **dados** ou de um **arquivo texto**, por exemplo.



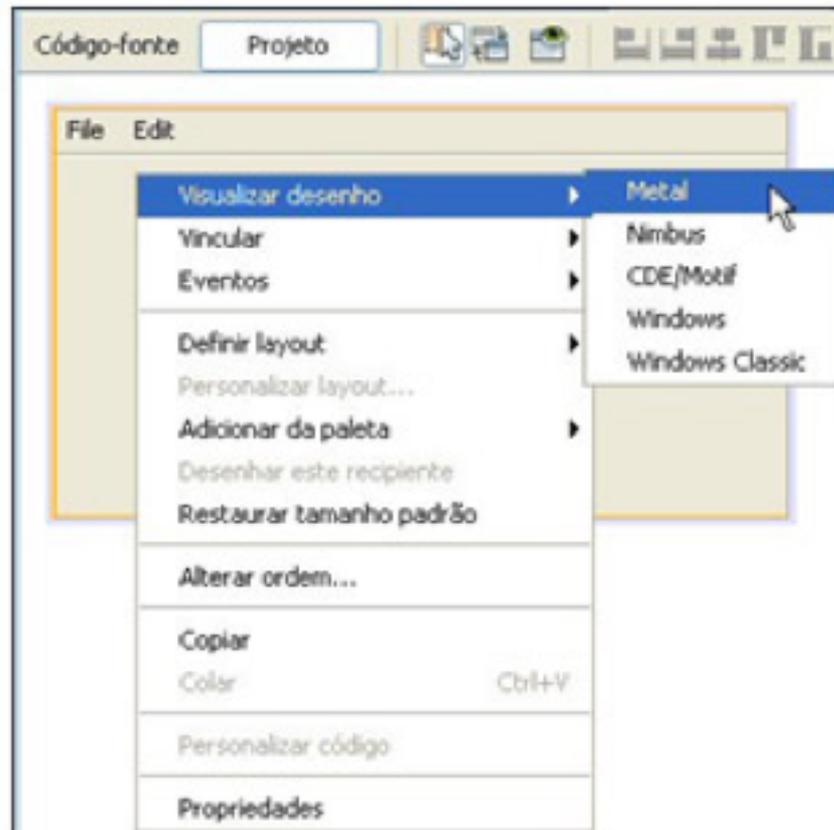
'Menu' (JMenu)

- Os **menus** podem ser **basicamente** de dois tipos: aqueles que ficam **localizados** no **topo** da janela (**barra de menus**) da aplicação e aqueles do tipo **pop-up**.
- Exemplo de Barra de menus:



'Menu' (JMenu)

- **Menu** de **Pop-up/contexto**, é aquele que só aparece quando você **clica** com o **botão direito** sobre o **objeto**.

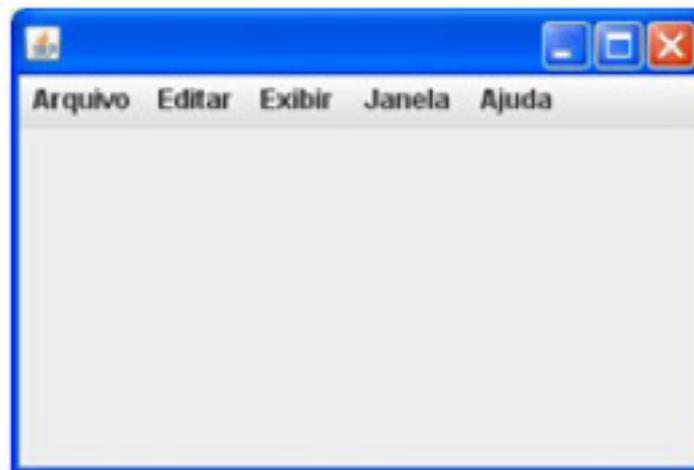


Aplicação com Menus

- ❑ Na verdade um menu em Java não é criado apenas com um componente, mas por vários componentes que em conjunto formam o menu propriamente dito.
- ❑ Iremos criar um menu passo a passo semelhante ao menu do próprio NetBeans.
 - Criem um novo projeto
 - Clique com o botão direito no nome da aplicação e crie um novo **Formulário JFrame**.
 - A partir da **Paleta** de componentes, na seção **Menus Swing**, clique e arraste um componente '**Barra de menu**' para a área do projeto. Observe que quando ele é solto na área do projeto ele é atraído para o topo da janela automaticamente. Essa barra representa o menu principal de qualquer aplicação, e o componente já traz duas opções como padrão

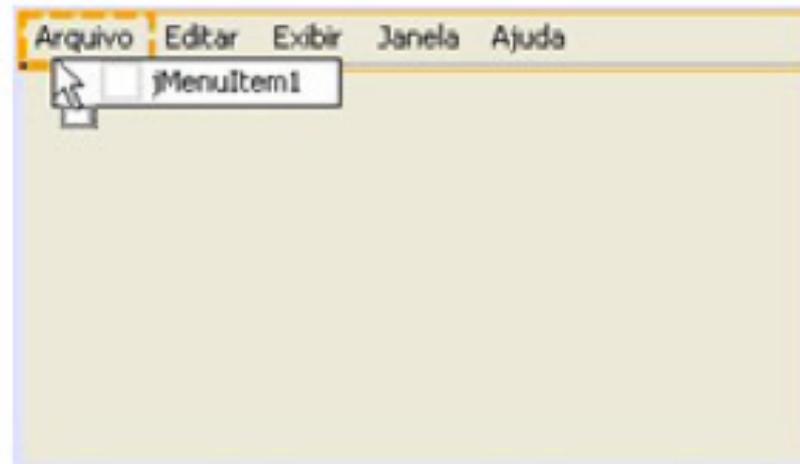
Aplicação com Menus

- Clique no item **'File'** e, na sua propriedade **'text'**, digite **'Arquivo'**. Tecele **'Enter'** para confirmar. Faça a mesma coisa com o item **'Edit'**, alterando para **'Editar'**.
- Para acrescentar mais um item ao menu principal, arraste um componente **'Menu'** e insira-o na frente do item **'Editar'**.
- Altere a propriedade **'text'** do novo item para **Exibir**.
- Acrescente mais dois itens ao menu e altere suas propriedades **'text'** para **'Janela'** e **'Ajuda'**, respectivamente.



Aplicação com Menus

- ❑ Se você clicar em algum item ele ficará com a cor de fundo um pouco azulada, mas nada acontecerá ainda. Precisamos criar, agora, os submenus de cada item do menu;
 - Arraste um componente '**Item de menu**' e posicione-o sobre o item '**Arquivo**' do menu principal, de forma que ele fique selecionado com uma borda laranja tracejada;



Aplicação com Menus

- ❑ Agora clique no seu rótulo (**jMenuItem1**) e altere sua propriedade '**text**' para '**Novo arquivo**'.
- ❑ Arraste outro componente '**Item de menu**' e adicione ao menu principal '**Arquivo**'. Altere sua propriedade '**text**' para '**Abrir arquivo**'.
- ❑ Acrescente mais dois '**Item de menu**' no item '**Arquivo**' e altere suas propriedades '**text**' para '**Fechar arquivo**' e '**Sair**', respectivamente.