

# Programação Orientada a Objetos - Java Swing

Professor Isaac

# Interface Gráfica em Java usando o Swing do NetBeans

# Java Swing

- Swing é uma biblioteca gráfica oficial do Java inclusa em qualquer JRE (Java Runtime Environment) ou JDK (Java Development Kit):
  - JRE é uma implementação do Java Virtual Machine que na verdade executa programas Java
  - JDK é um pacote de software que você pode usar para desenvolver aplicativos baseados em Java
- Com Swing a aplicação se comportará da mesma forma em todos os ambientes, independente de: sistema operacional, resolução de tela ou profundidade de cores

# Criação do Projeto e do JFrame

- Clique em Novo Projeto
- Escolha Java -> Aplicação Java e clique em Próximo
- Atribua o nome ao Projeto, **DESMARQUE a opção Criar Classe Principal** e clique em Finalizar
- Clique com o botão direito **no pacote “telas”, selecione Novo e clique em Form JFrame**
- Escreva o nome da Classe e clique em Finalizar

**Paleta:** componentes para serem adicionados ao JFrame

**Propriedades:** características de cada componente que podem ser alteradas

Saída - CadastroCliente (run)

```
run:
CONSTRUÍDO COM SUCESSO (tempo total: 22 segundos)
```

# Algumas **Propriedades** do JFrame

- **resizable**: permite ou não redimensionar a janela durante a execução do projeto
- **location**: localização do canto superior esquerdo da janela na tela
- **locationByPlatform**: padrão de localização de acordo com a plataforma
- **defaultCloseOperation**: define a como reagir a operação de fechar o formulário quando tem mais de uma janela aberta

# Alguns componentes da **Paleta**

- **Painel (JPanel):** Painel criado sobre o JFrame para organizar melhor sua interface
- **Label (JLabel):** Permite inserir textos e imagem no JFrame que são somente visualizados pelo usuário



The image shows a snippet of a Java Swing window. On the left side, there are three labels: 'Fone:', 'Estado:', and 'Sexo:'. These labels are enclosed in a red rectangular box. To the right of 'Fone:' is a text input field. To the right of 'Estado:' is a dropdown menu showing 'AC' with a downward arrow. To the right of 'Sexo:' are two radio buttons labeled 'Masculino' and 'Feminino'.

# Alguns componentes da **Paleta**

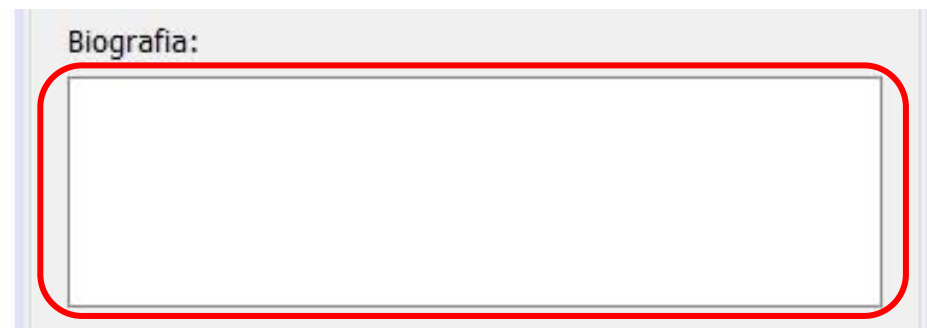
- **Campo de Texto (JTextField):** Permite ao usuário digitar os dados de entrada em tempo de execução



A screenshot of a Java Swing window containing two text input fields. The first field is labeled 'Nome:' and the second is labeled 'Endereço'. Both fields are highlighted with a red rounded rectangle, indicating they are JTextField components.

**IMPORTANTE:** todos os dados digitados em um Campo de Texto são tratados como String, sendo necessária a conversão caso a variável que recebe o dado seja de tipo diferente

- **Área de texto (JTextArea):** Campo semelhante ao Campo de texto, usado para entrada de textos maiores



A screenshot of a Java Swing window containing a large text area. The label 'Biografia:' is positioned to the left of the text area. The text area is highlighted with a red rounded rectangle, indicating it is a JTextArea component.



# Alguns componentes da **Paleta**

- **Grupo de Botões (JButtonGroup):** Utilizado para criar grupos de Botões de Rádio
- **Botão de Rádio (JRadioButton):** Permite o usuário selecionar apenas uma opção quando relacionado a um Grupo de Botões



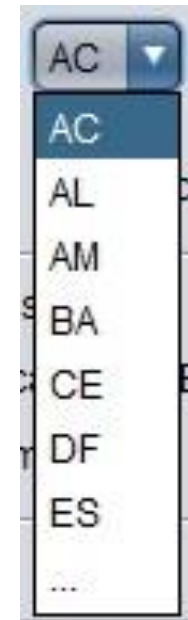
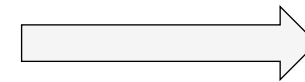
☒ Masculino ☐ Feminino

# Alguns componentes da **Paleta**

- **Caixa de Seleção (JCheckBox):** Permite ao usuário marcar várias opções ao mesmo tempo



- **Caixa de Combinação (JComboBox):** Permite ao usuário escolher apenas uma opção



- **Botão (JButton):** dispara um evento ao ser clicado pelo usuário



# Exemplo de formulário com os componentes:

- Painel
- Label
- Campo de Texto
- Área de Texto
- Botão de Rádio
- Caixa de Combinação
- Caixa de Seleção
- Botão

Cadastro

Nome:

Endereço:

E-mail:

Fone:

Estado:  ▼

Sexo: ☐ Masculino ☐ Feminino

Interesses

☐ Música ☐ Esportes

☐ Cinema ☐ Informática

Biografia:

# Tela do NetBeans com o Formulário criado

ProjetoSwing - NetBeans IDE 8.1

Arquivo Editar Exibir Navegar Código-Fonte Refatorar Executar Depurar Perfil Equipe Ferramentas Janela Ajuda

Página Inicial x TelaCadastro.java x Cadastro.java x

Código-Fonte Projeto Histórico

Para adicionar um componente várias vezes, selecione-o clicando na paleta e, em seguida, clique com a tecla Shift pressionada na tela de design.

**Navegador: nomes dos componente utilizados**

**Clicar no nome para selecionar o componente em Propriedades (exemplo: JFrame)**

**Paleta**

- Contêineres Swing
  - Painel
  - Dividir Painel
  - Barra de Ferramentas
  - Quadro Interno
  - Painel com Guias
  - Painel de Rolagem
  - Painel da Área de Trabalho
  - Painel em Camadas
- Controles Swing
  - Label
  - Botão Alternar
  - Botão de Rádio
  - Caixa de Combinação
  - Campo de Texto
  - Barra de Rolagem
  - Barra de Progresso
  - Campo de Senha
  - Botão
  - Caixa de Seleção
  - Grupo de Botões
  - Listar
  - Área de Texto
  - Controle Deslizante
  - Campo Formatado
  - Controle Giratório

**[JFrame] - Propriedades**

Propriedades Vinculação Eventos Código

Propriedades

- defaultCloseOperation: EXIT\_ON\_CLOSE
- title
- Outras Propriedades
  - alwaysOnTop
  - alwaysOnTopSupported
  - autoRequestFocus
  - background
  - bounds: [240,240,240]
  - cursor: Cursor Padrão
  - enabled
  - extendedState: n

**[JFrame]**

Form Cadastro

Outros Comp

JFrame

JPanel1 [JPanel]

- jLabel1 [JLabel]
- jLabel2 [JLabel]
- jLabel3 [JLabel]
- jLabel4 [JLabel]
- jLabel5 [JLabel]
- jLabel6 [JLabel]
- jLabel7 [JLabel]
- JTextField1 [JTextField]
- JTextField2 [JTextField]
- JTextField3 [JTextField]
- JTextField4 [JTextField]
- JComboBox1 [JComboBox]
- JRadioButton1 [JRadioButton]
- JRadioButton2 [JRadioButton]
- JPanel2 [JPanel]
- JScrollPane1 [JScrollPane]
- JTextArea1 [JTextArea]
- JButton1 [JButton]
- JButton2 [JButton]
- jLabel8 [JLabel]

Salvar Limpar

Ant -f C:\Users\marmo\Documents\NetBeansProjects\ProjetoSwing -Dnb.internal.action.name=run run

init:

Deleting: C:\Users\marmo\Documents\NetBeansProjects\ProjetoSwing\build\build-jar.properties

deps-jar:

Updating property file: C:\Users\marmo\Documents\NetBeansProjects\ProjetoSwing\build\build-jar.properties

Compiling 1 source file to C:\Users\marmo\Documents\NetBeansProjects\ProjetoSwing\build\classes

warning: [options] bootstrap class path not set in conjunction with -source 1.7

1 warning

compile:

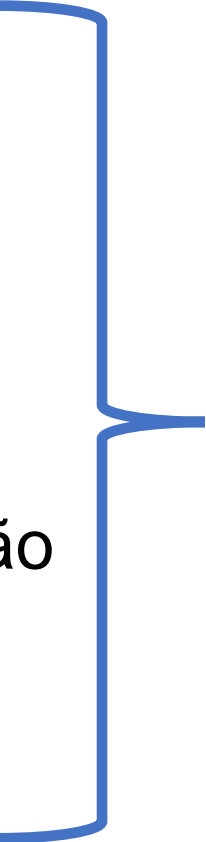
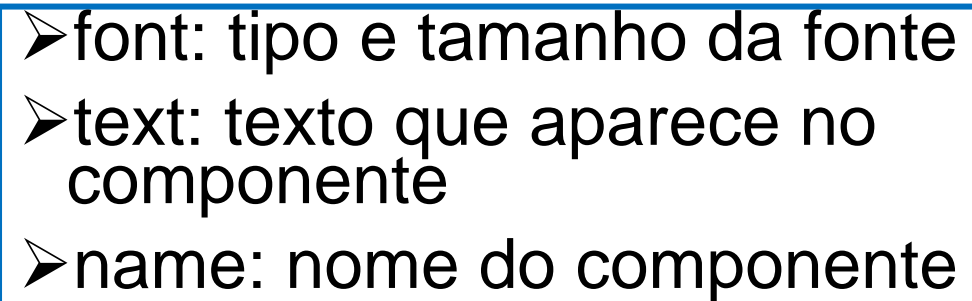
run:

CONSTRUÍDO COM SUCESSO (tempo total: 1 minuto 15 segundos)

258:1

# Propriedades comuns dos componentes da Paleta

- Painel
- Label
- Campo de Texto
- Área de Texto
- Grupo de Botões
- Botões de Rádio
- Caixa de Combinação
- Caixa de Seleção
- Botão

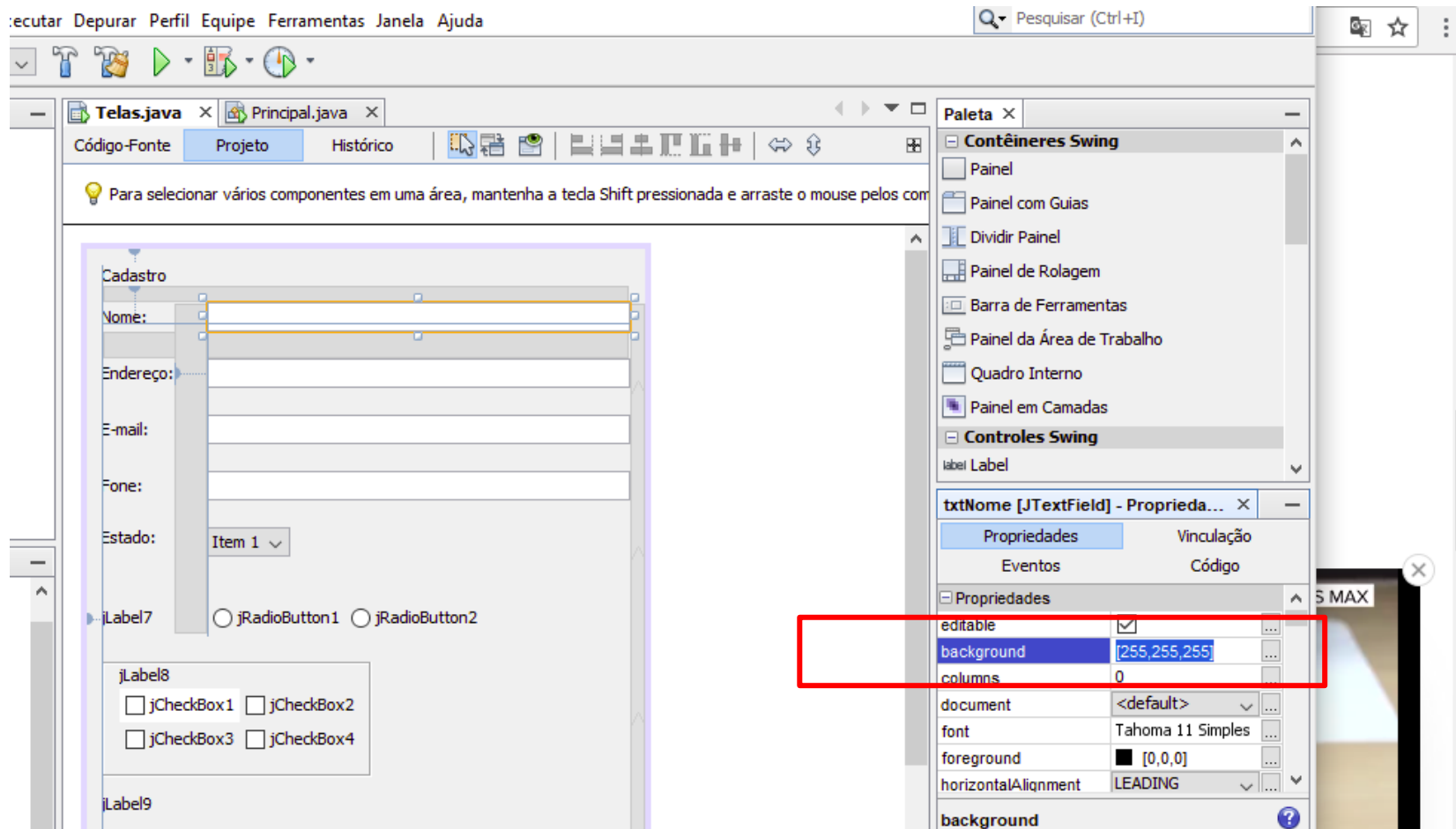
- 
- 
- font: tipo e tamanho da fonte
  - text: texto que aparece no componente
  - name: nome do componente

# Algumas Propriedades dos componentes da Paleta

- Painel (JPanel)
  - border: define o tipo de borda do Painel
- Label (JLabel)
  - background: muda a cor de fundo
  - foreground: altera a cor do conteúdo escrito
  - icon: adiciona uma figura

# Algumas Propriedades dos componentes da Paleta

- Background: mudando a cor de fundo.



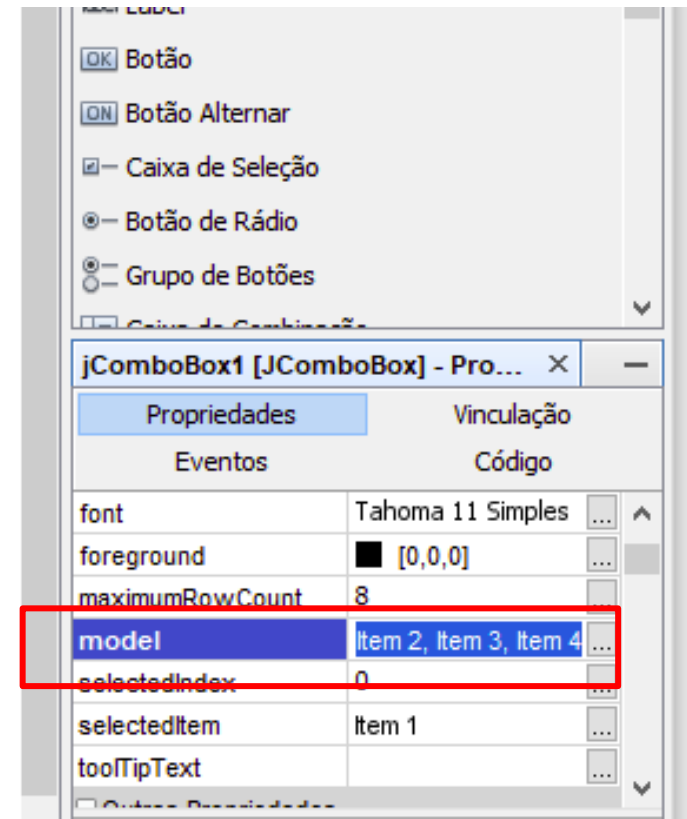
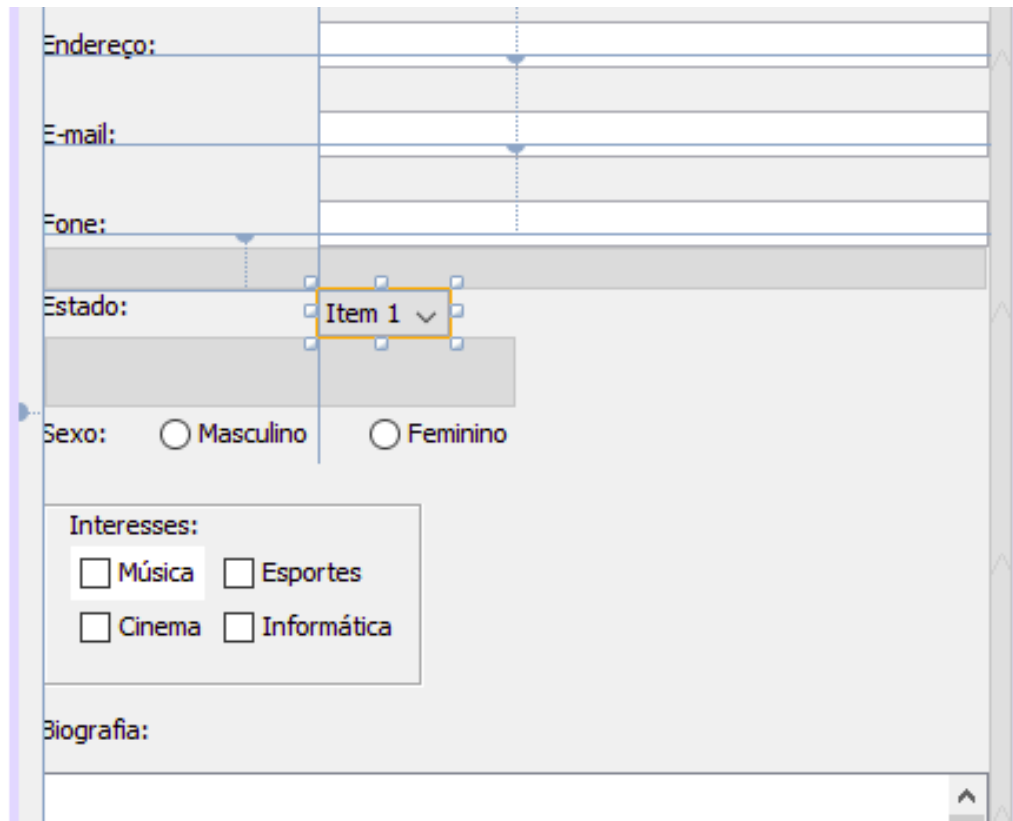
# Algumas Propriedades dos componentes da Paleta

- Caixa de Seleção (JCheckBox):
    - selected: informação se a Caixa está ou não selecionada
  - Caixa de Combinação (JComboBox):
    - selected: informação se a Caixa está ou não selecionada
    - model: lista separada por vírgulas das opções da Caixa
  - Botões de Rádio (JRadioButton):
    - selected: informação se a Caixa está ou não selecionada
    - buttonGroup: nome do grupo de botões que faz parte
- IMPORTANTE: os botões tem que ser de um mesmo grupo para obrigar a escolher apenas um deles



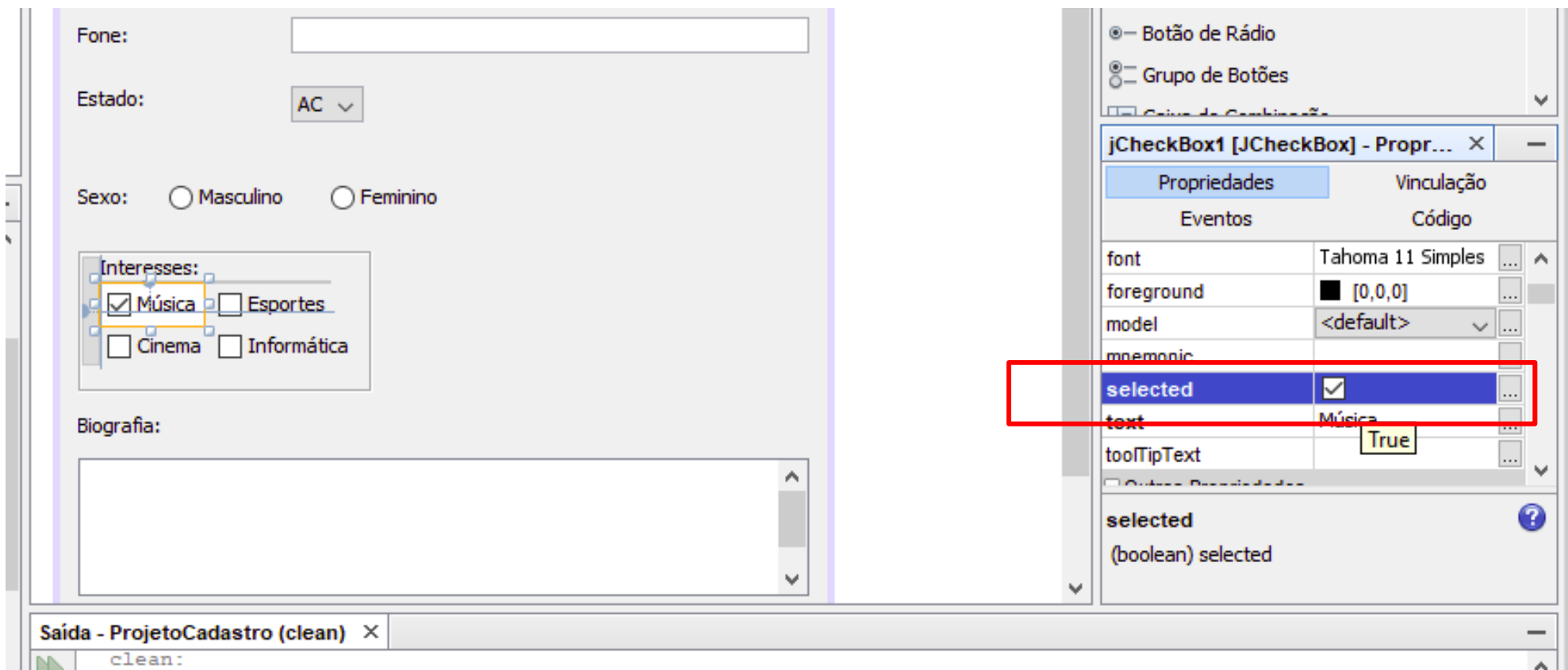
# Algumas Propriedades dos componentes da Paleta

- **model:** listando as opções da caixa.



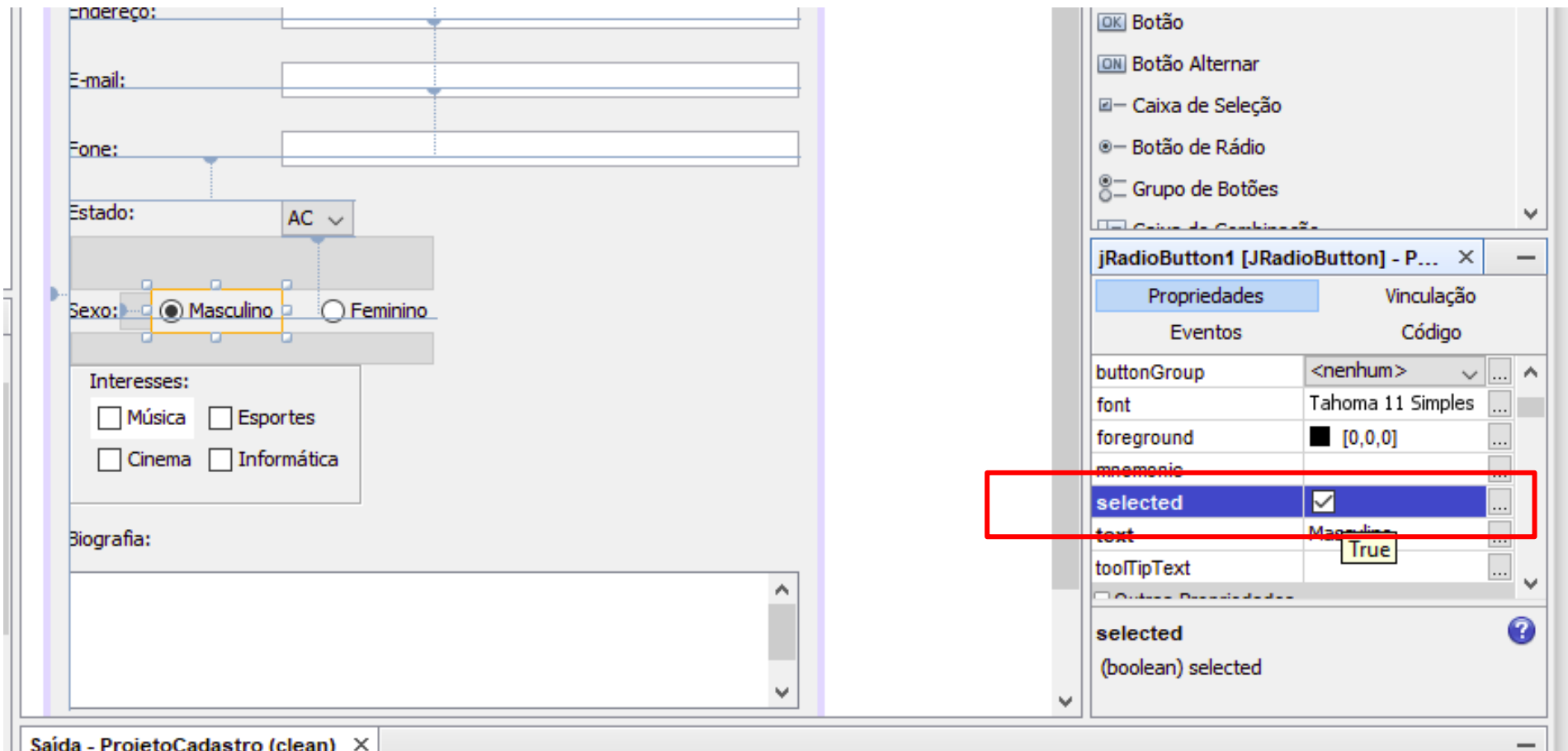
# Algumas Propriedades dos componentes da Paleta

- **selected**: informação se a Caixa está ou não selecionada.



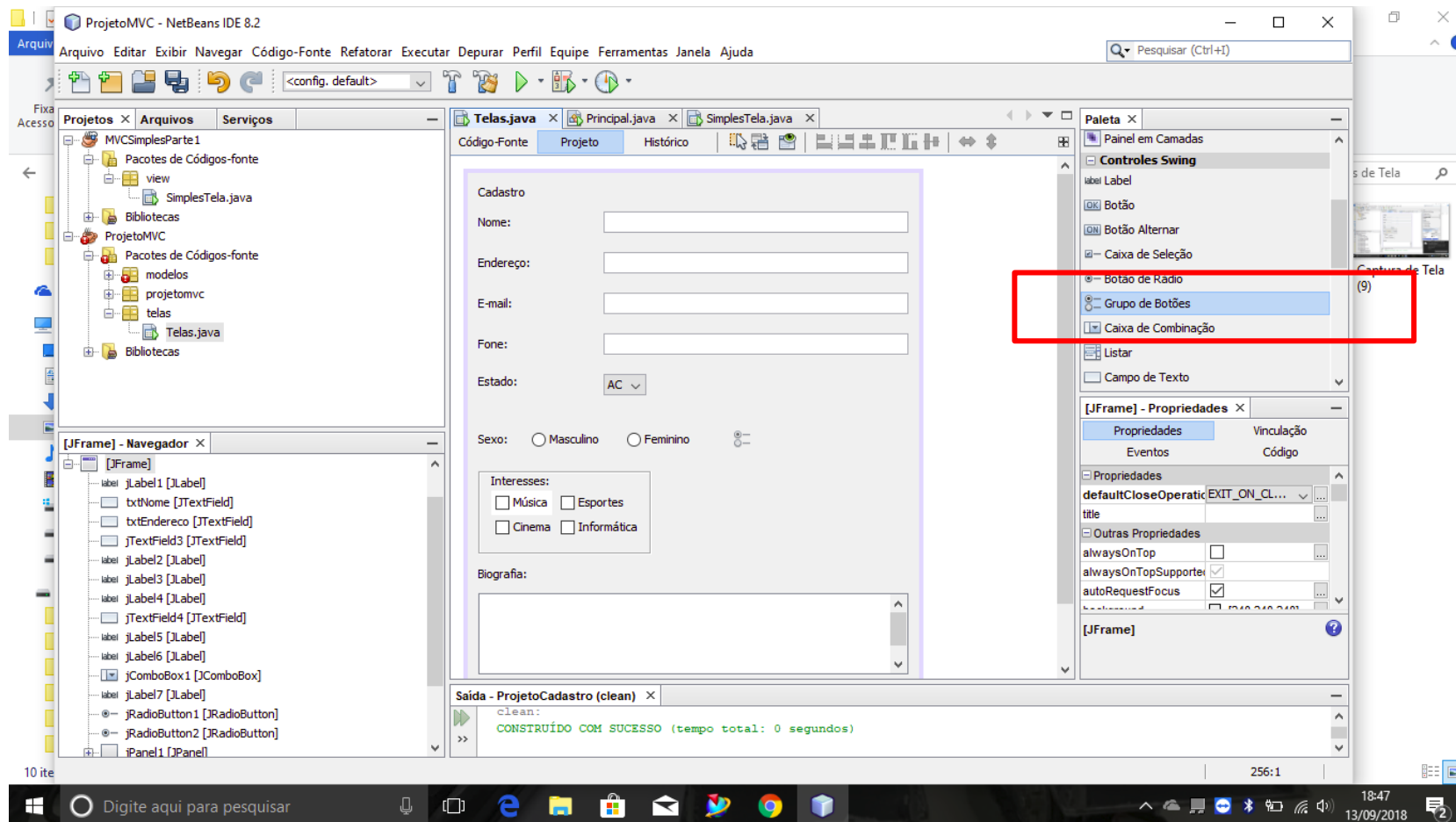
# Algumas Propriedades dos componentes da Paleta

- **selected**: informação se a Caixa está ou não selecionada.



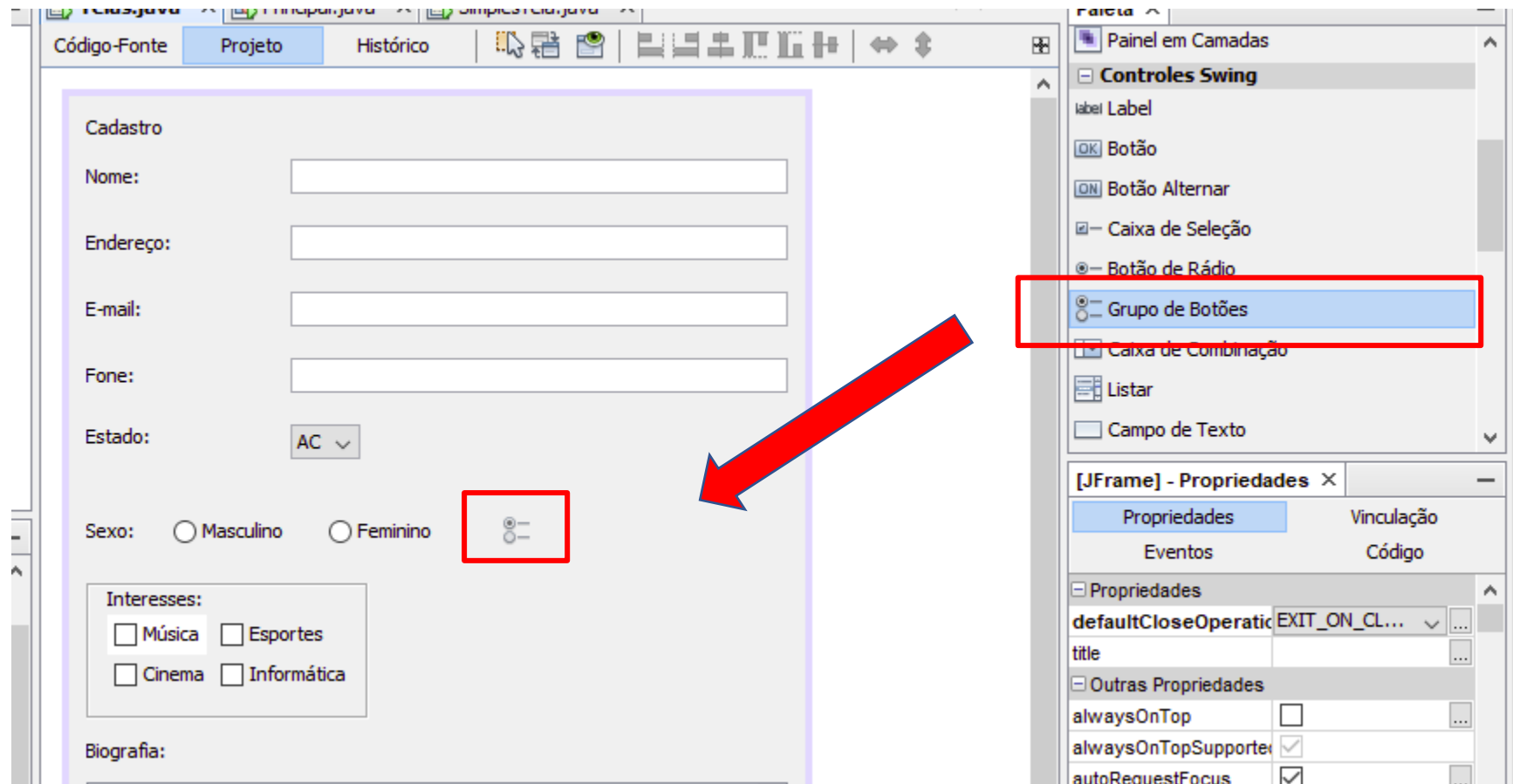
# Algumas Propriedades dos componentes da Paleta

- **buttonGroup**: nome do grupo de botões que faz parte.



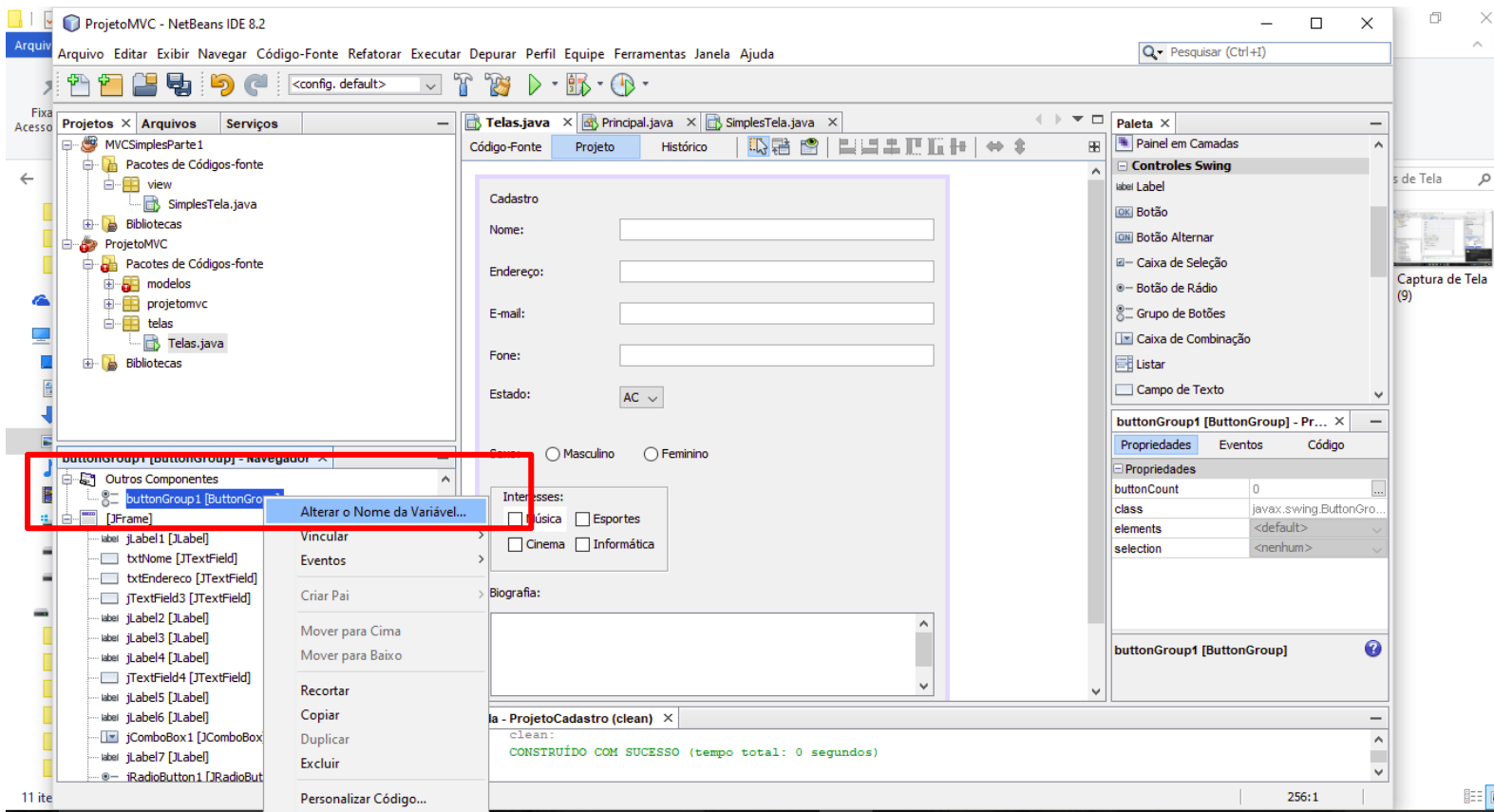
# Algumas **Propriedades** dos componentes da **Paleta**

- **buttonGroup**: coloque um buttonGroup no seu JFrame.



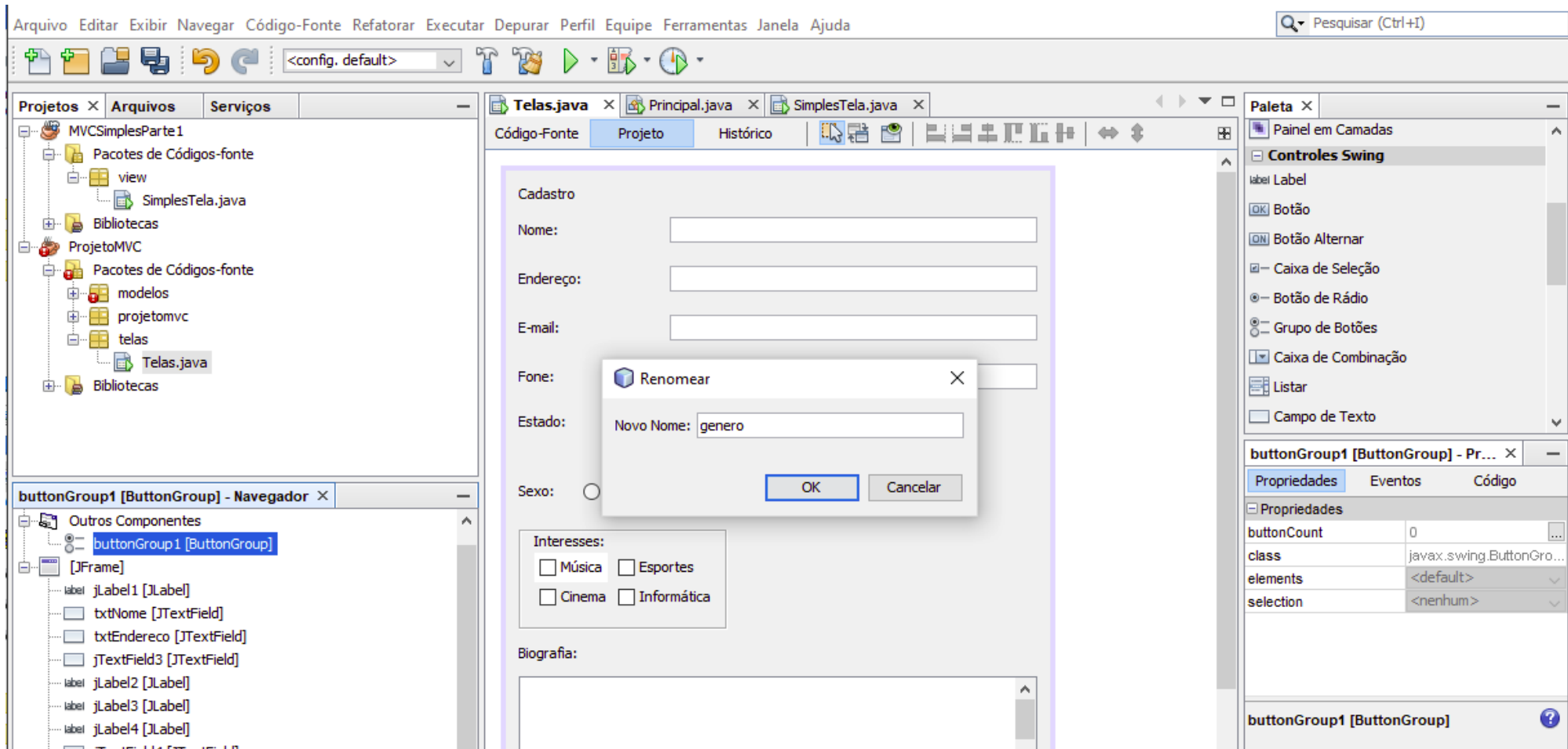
# Algumas Propriedades dos componentes da Paleta

- **buttonGroup**: altere o nome da variável.



# Algumas Propriedades dos componentes da Paleta

- **buttonGroup**: Coloque o nome **genero**.



# Algumas Propriedades dos componentes da Paleta

- **buttonGroup**: No JRadioButton troque para **genero**.

The screenshot displays an IDE with a Java Swing application. The central canvas shows a registration form with fields for Name, Address, Email, Phone, State (dropdown), Sex (radio buttons for Masculino and Feminino), Interests (checkboxes for Música, Esportes, Cinema, Informática), and a Biography text area. The 'Sexo' section is highlighted with a yellow box. On the left, the 'Outros Componentes' palette shows a tree structure with 'JFrame' containing various labels and text fields. On the right, the 'Painel em Camadas' palette shows 'Controles Swing' components. The 'jRadioButton1 [JRadioButton] - P...' component is selected, and its properties are shown in the 'Propriedades' tab. The 'buttonGroup' property is set to '<nenhum>' and the 'foreground' property is set to 'genero'. A red box highlights these two properties.

**Propriedades**

Propriedades	Vinculação
action	<nenhum>
background	[240,240,240]
model	<default>
buttonGroup	<nenhum>
font	<nenhum>
foreground	genero

**buttonGroup**  
Grupo de botões ao qual este botão pertence



Manipular os dados do formulário da  
Swing do NetBeans

# Nome de Variável dos Componentes

- O nome do componente é utilizado para manipular suas propriedades e eventos no código do programa
- A alteração do nome é feita clicando com o botão direito sobre o componente e depois em “Alterar o Nome da Variável”
- O nome da variável deve seguir um padrão (uma sugestão de padrões é definida no próximo slide)
- Não é necessário modificar os nomes de todos os componentes do formulário, somente os nomes daqueles que serão manipuladas pelo código do programa

# Padrões para nomes de variáveis

- Painel (JPanel): pnlNomeDoPainel
  - Exemplo: pnlPrincipal
- Label(JLabel): lblNomeDoLabel
  - Exemplo: lblEmail
- Campo de Texto (JTextField): txtNomeDoCampo
  - Exemplo: txtEndereco
- Área de Texto(JTextArea): areaNomeDaArea
  - Exemplo: areaBiografia

# Padrões para nomes de variáveis

- Grupo de Botões (JGroupButton): grpNomeDoGrupo
  - Exemplo: grpSexo
- Botões de Rádio (JRadioButton): rbtNomeDaOpção
  - Exemplo : rbtMasculino
- Caixa de Combinação (JComboBox): cmbxNomeDaOpção
  - Exemplo: cmbxEstado
- Caixa de Seleção JCheckBox: chbxNomeDaOpção
  - Exemplo: chbxMusica
- Botão (JButton): btnNomeDoBotão
  - Exemplo: btnSalvar

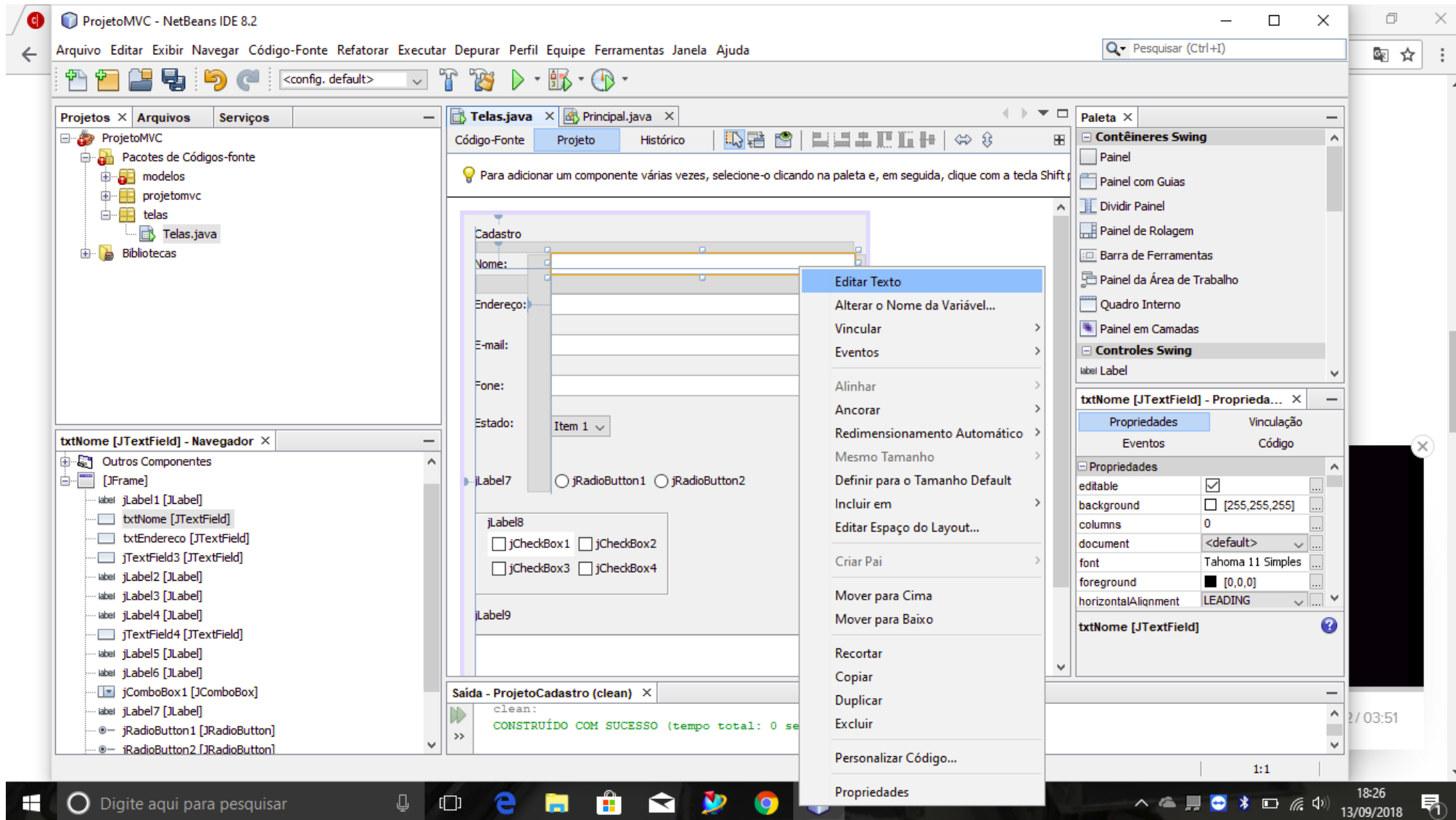
# Projeto Cadastro:

## Alterar os nomes dos componentes

- Campos de Texto: txtNome, txtEndereco, txtEmail, txtFone
- Caixa de Combinação: cmbxEstado
- Botões de Rádio: rbtMasculino, rbtFeminino
- Grupo de Botões: grpSexo
- Caixas de Seleção: chbxMusica, chbxEsportes, chbxCinema, chbxInformatica
- Área de Texto: areaBiografia
- Botão: btnSalvar, btnLimpar

# Projeto Cadastro:

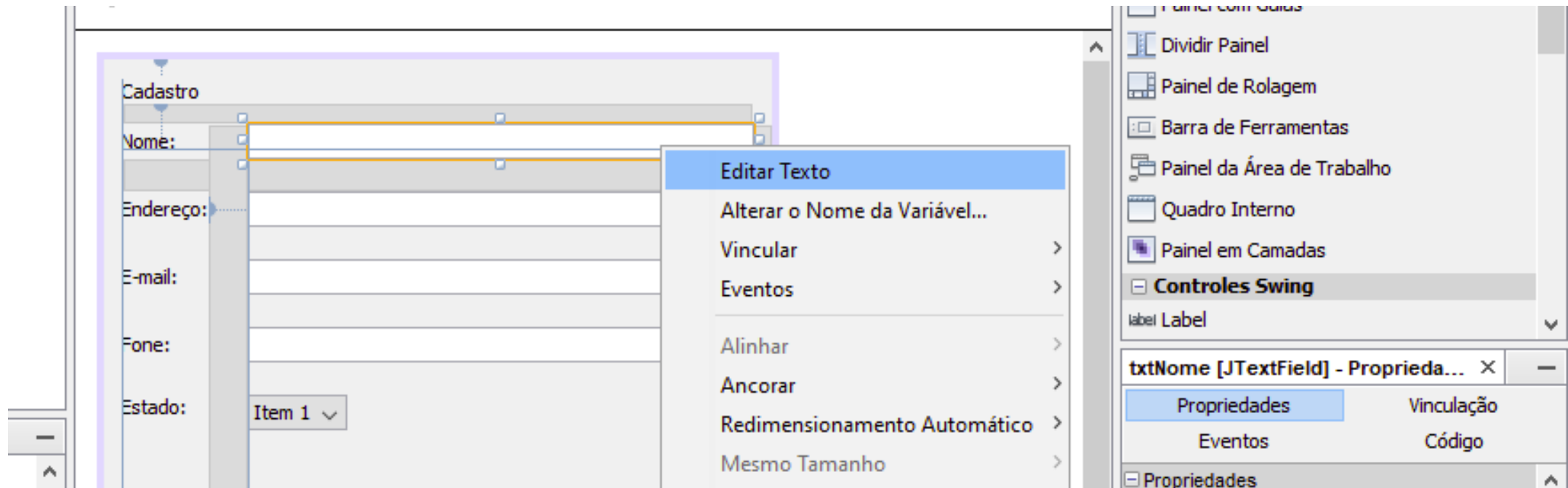
## Alterar os nomes dos componentes



# Projeto Cadastro:

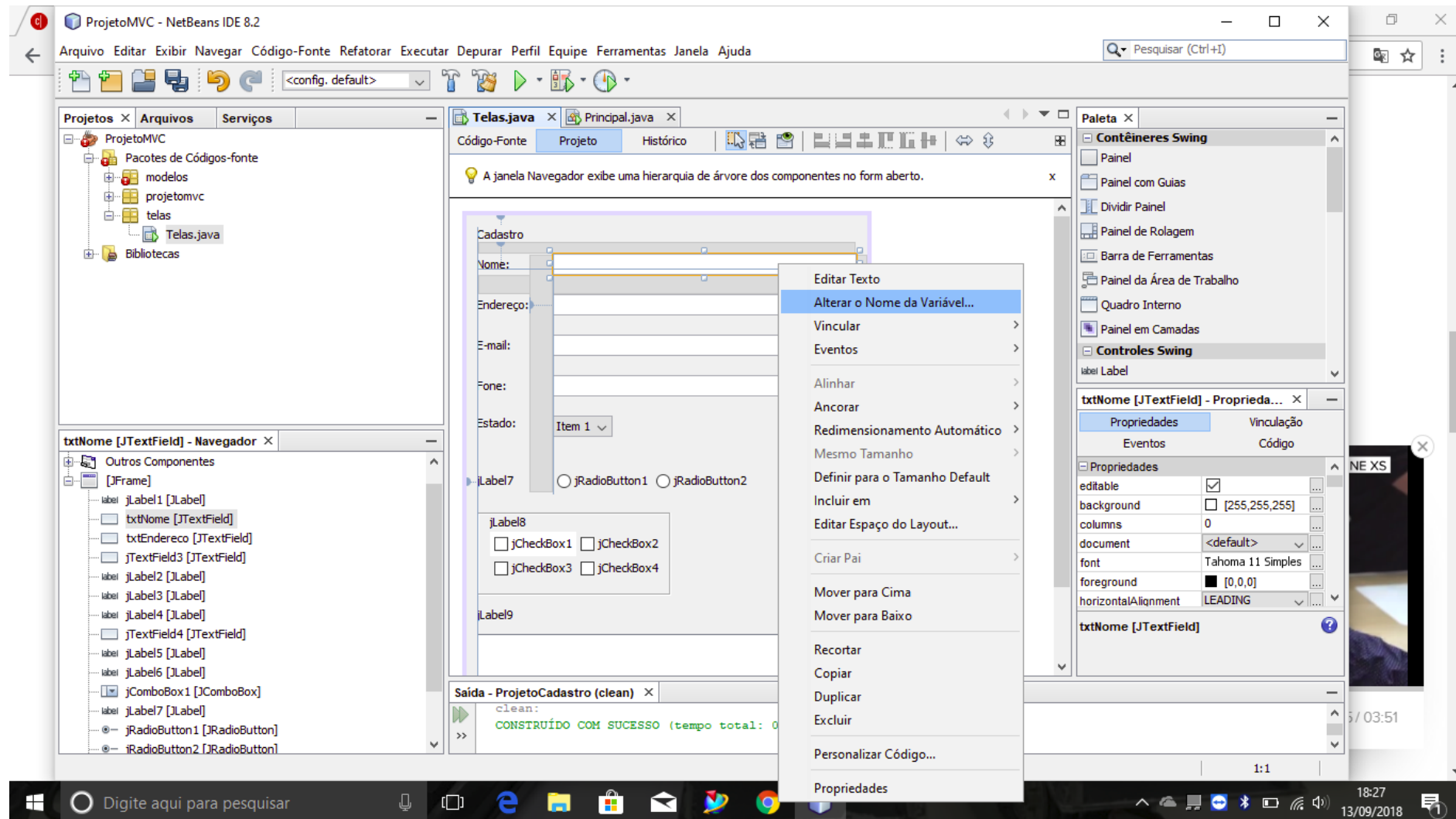
## Alterar **Texto** dos componentes

- A alteração do texto é feita clicando com o botão direito sobre o componente e depois em “Alterar o Nome da Variável”.



# Projeto Cadastro:

## Alterar os **nomes** dos componentes

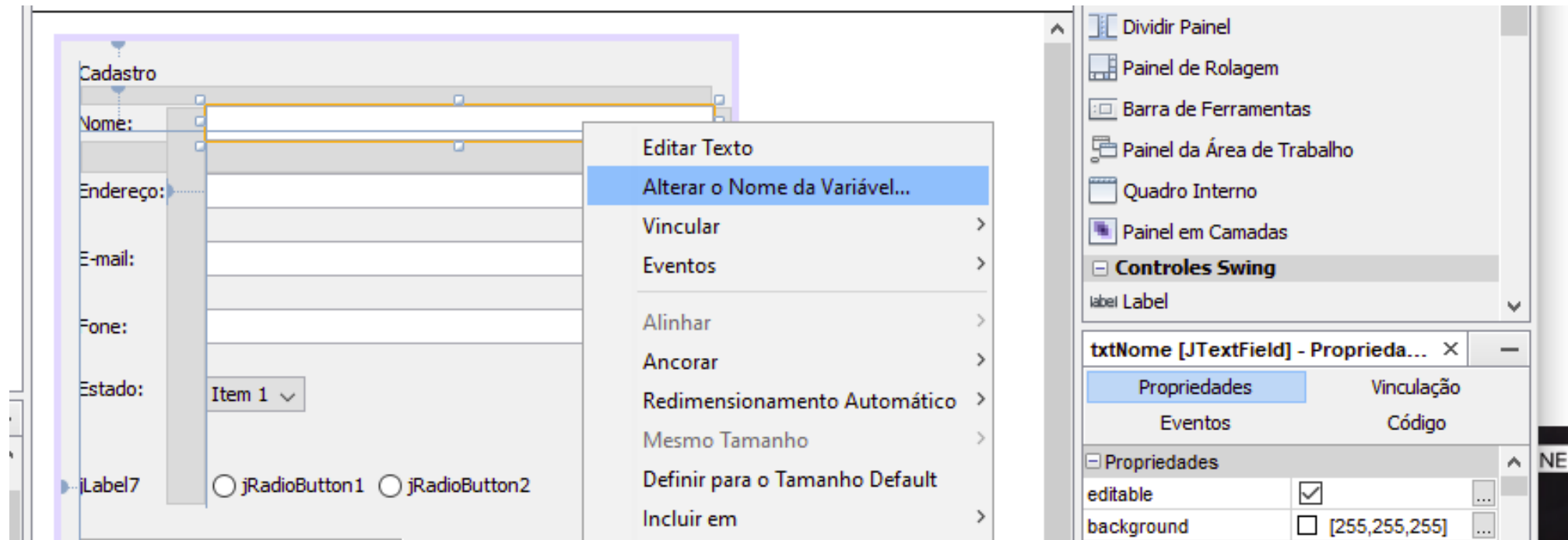




# Projeto Cadastro:

## Alterar os **nomes** dos componentes

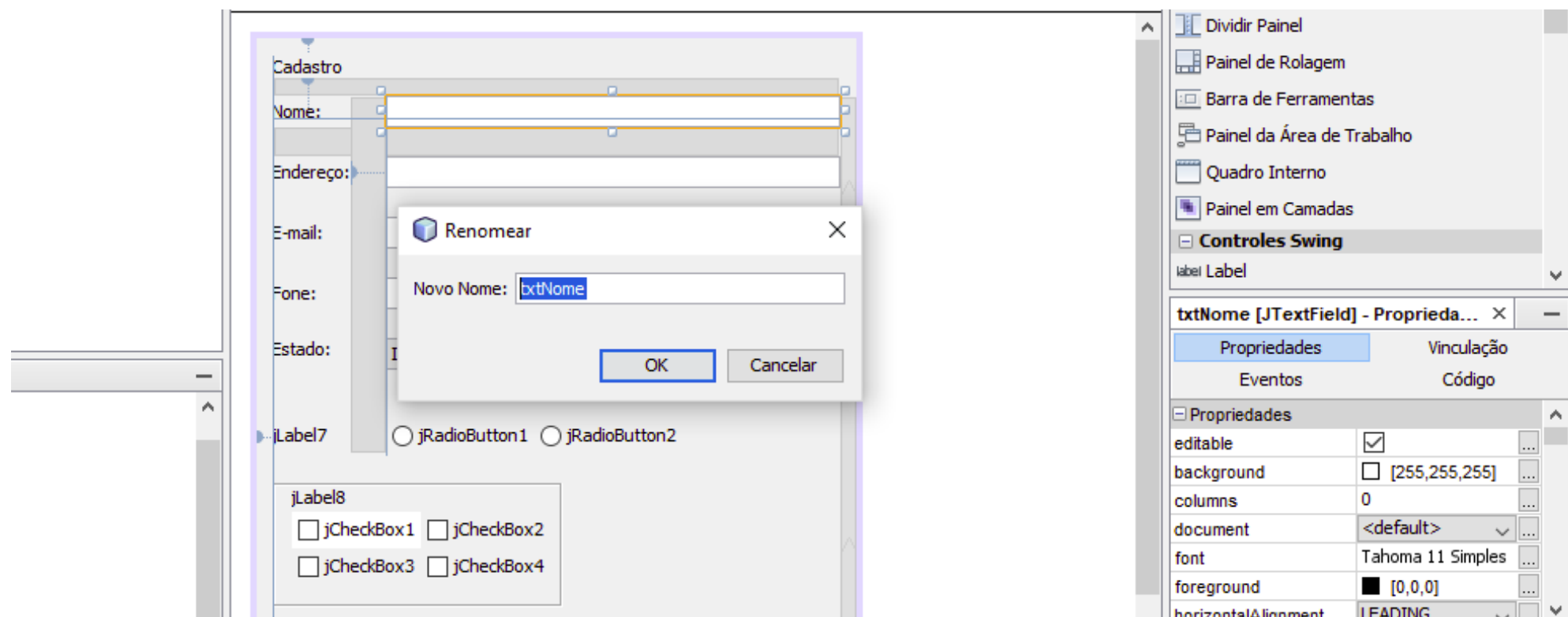
- A alteração do nome é feita clicando com o botão direito sobre o componente e depois em “Alterar o Nome da Variável”.



# Projeto Cadastro:

## Alterar os **nomes** dos componentes

- A alteração do nome é feita clicando com o botão direito sobre o componente e depois em “Alterar o Nome da Variável”.



# Eventos

- Um evento é acionado quando alguma ação é realizada no formulário, como por exemplo:
  - Clicar em botões
  - Apertar teclas do teclado
  - Rolar a barra de informações
  - Marcar, selecionar, escrever, minimizar, fechar e uma infinidade de possibilidades...
- Quando um evento é acionado durante a execução do sistema o seu código é executado
- Qualquer código Java deve digitado dentro de algum evento quando usamos um formulário (**Form JFrame**)

# Acionar o evento ao clicar no Botão

- Dê dois cliques no botão para surgir o código do evento com o nome daquele botão
- Exemplo:

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
}
```

- Nesse local devem ser codificadas em Java as instruções seguidas quando o botão for clicado durante a execução do sistema

# Acessar os dados digitados (**getText**)

- Para acessar os dados que são digitados em um Campo de Texto ou Área de Texto use o método **getText()**
- Esse método captura os dados no formato de texto (String)
- Sintaxe: **nomeDoComponente.getText();**
- Exemplo: **campoEndereco.getText();**

# Atribuir dados digitados a uma variável

- Para atribuir o texto que foi “pego” usando `getText` a outra variável use a sintaxe:

```
nomeDaVariável = nomeDoComponente.getText();
```

- Exemplo:

```
String endereco = campoEndereco.getText();
```

- Caso a variável não seja do tipo **String** é necessário modificar o tipo do texto que foi capturado pelo método **getText()**

# Modificar o tipo

- Para converter em **double** use o método **Double.parseDouble()**

- Exemplo de conversão para double:

**double valor = Double.parseDouble(campoValor.getText())**

# Modificar o tipo

- Para converter em **inteiro** use o método **Integer.parseInt()**

- Exemplo de conversão para inteiro:

```
int quant = Integer.parseInt(campoQuantidade.getText())
```



# Modificar o tipo

- Para converter em **String** use o método **String.valueOf()**
- Exemplo de conversão para String:

**String numero = String.valueOf(100)**

**String numero = String.valueOf(3.14)**

# Projeto Cadastro:

## Capturar os dados digitados

- Clicar duas vezes no botão Salvar para criar o evento `btnSalvarActionPerformed`
- Criar variáveis da classe String e atribuir os dados digitados no formulário nos campos de texto e na área de texto
  - Exemplo 1: **`String nome = txtNome.getText();`**
  - Exemplo 2: **`String endereco = txtEndereco.getText();`**

# Projeto Cadastro:

## Capturar os dados digitados

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String nome = txtNome.getText();  
    String endereco = txtEndereco.getText();  
    String email = txtEmail.getText();  
    String fone = txtFone.getText();  
    String genero;  
    if (rbtFeminino.isSelected()) {  
        genero = "feminino";  
    } else {  
        genero = "masculino";  
    }  
}
```

# Manipular as Escolhas nos Formulários da Swing do NetBeans

# Acessar as escolhas feitas no formulário

- Para acessar a escolha feitas nas Caixas e Botões do formulário usamos métodos que retornam o objeto selecionado ou a escolha feita
- Caixa de Combinação: método **getSelectedItem()**
- Botões de Radio ou na Caixa de Seleção: método **isSelected()**

# Acessar as escolhas da Caixa de Combinação

- O método `getSelectedItem()` retorna o objeto selecionado na caixa
- Para atribuir esse valor a uma variável do tipo `String` devemos usar o modificador `(String)` antes de fazer a atribuição
- Sintaxe: `(Modificador) nomeDoComponente.getSelectedItem()` ;
- Exemplo: `(String) cmbxEstado.getSelectedItem();`

# Atribuir a escolha feita na Caixa de Combinação a uma variável

- Para atribuir o texto que foi selecionado a outra variável do tipo String use a sintaxe:

```
String nomeDaVariável = (Modificador)  
                           nomeDoComponente.getSelectedItemAt() ;
```

- Exemplo:

```
String estado = (String) cmbxEstado.getSelectedItemAt();
```

# Acessar as escolhas do Botão de Rádio e da Caixa de Seleção

- O método `isSelected()` retorna um valor *boolean*:
  - Retorna o valor *true* caso o botão ou a caixa estejam selecionados
  - Retorna o valor *false* caso o botão ou a caixa NÃO estejam selecionados
- Para verificar se o valor está ou não selecionado podemos usar o comando *if*
- Sintaxe: `nomeDoComponente.isSelected()` ;
- Exemplo: **`chbxMusica.isSelected()`**;



# Atribuir a escolha feita nos Botões de Rádio e nas Caixas de Seleção a variáveis

- Podemos atribuir o valor do botão ou da caixa diretamente a uma variável do tipo *boolean*
  - Sintaxe: `boolean nomeDaVariavel = nomeDoComponente.isSelected();`
  - Exemplo: `boolean musica = chbxMusica.isSelected();`
- Para verificar se o botão ou a caixa de seleção foi selecionada podemos usar o comando *if* e atribuir um valor ou realizar uma ação a partir do resultado
- Exemplo:

```
if (cnbxMusica.isSelected()) {  
    interesses = interesses + "Música ";  
}
```

# Projeto Cadastro:

## Capturar os dados digitados

- Clicar duas vezes no botão Salvar para acessar o evento `btnSalvarActionPerformed`
- Criar variável da classe `String` e atribuir a seleção feita no formulário na Caixa de Combinação
- Exemplo: **`String estado = (String) cmbxEstado.getSelectedItem();`**

# Projeto Cadastro:

## Capturar os dados digitados

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String nome = txtNome.getText();  
    String endereco = txtEndereco.getText();  
    String email = txtEmail.getText();  
    String fone = txtFone.getText();  
    String genero;  
    if (rbtFeminino.isSelected()) {  
        genero = "feminino";  
    } else {  
        genero = "masculino";  
    }  
    String naturalDe = (String) cmbxEstado.getSelectedItem();  
    String interesses = "";  
    if (cnbxInformatica.isSelected())  
        interesses = interesses + "Informatica ";  
    if (cnbxMusica.isSelected())  
        interesses = interesses + "Música ";  
    if (cnbxEsporte.isSelected())  
        interesses = interesses + "Esporte ";  
    if (cnbxCinema.isSelected())  
        interesses = interesses + "Cinema ";  
}
```

# Caixa de Diálogo usando JOptionPane

- As caixas de diálogo são usadas para interagir com o usuário
- Tem várias utilizações como mostrar uma mensagem, mostrar valores de variáveis, confirmar uma mensagem, requisição de algum dado.
- A classe JOptionPane proporciona uma série de métodos estáticos que ao serem invocados criam caixas de diálogos simples e objetivas
- Para usar JOptionPane temos sempre que primeiro importar o pacote **javax.swing.JOptionPane**

# Tipos de Caixas de Diálogo

- Caixa de Confirmação:
  - **showConfirmDialog**: Solicita uma confirmação, por exemplo SIM/NÃO/CANCELAR
- Caixa de Entrada:
  - **showInputDialog**: Solicita alguma entrada
- Caixa de Mensagem:
  - **showMessageDialog**: Mostra algum aviso ao usuário
- Caixa de Opções:
  - **showOptionDialog**: É uma unificação dos três métodos anteriores

# Caixa de Diálogo de Mensagem

- Serve para emitir uma mensagem
- Configurável e versátil
- Muitas situações distintas, tais como:
  - Informação
  - Alerta
  - Mensagem de erro
- Método para mostrar a caixa de diálogo: **showMessageDialog**

# Sintaxe da Caixa de Diálogo de Mensagem

- `showMessageDialog (null, “mensagem”);`
- `showMessageDialog (null, “mensagem”, “título”, messageType);`
- `showMessageDialog (null, “mensagem”, “título”, messageType, icone);`

Sendo que:

- `null`: indica que não existe dependência com outras janelas
- `“mensagem”`: é a mensagem que será mostrada
- `“título”`: é o título da caixa de diálogo
- `messageType`: é o tipo da caixa de diálogo
- `icone`: Caso queira mostra um ícone na mensagem

# Tipos da Caixa de Diálogo

- `showMessageDialog(null, "mensagem", "título", messageType, icon)`

## Exemplo:

```
JOptionPane.showMessageDialog (null, "Mensagem", "título",  
JOptionPane.INFORMATION_MESSAGE);
```



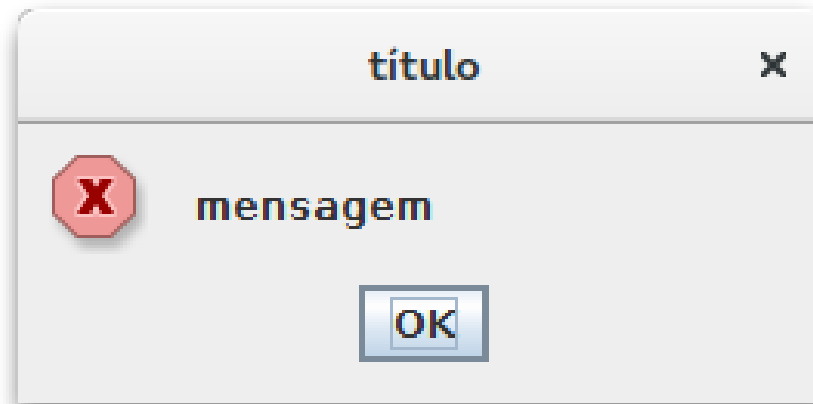


# Tipos da Caixa de Diálogo de Mensagem

- Erro: `JOptionPane.ERROR_MESSAGE`
- Informação: `JOptionPane.INFORMATION_MESSAGE`
- Aviso: `JOptionPane.WARNING_MESSAGE`
- Pergunta: `JOptionPane.QUESTION_MESSAGE`
- Sem ícones: `JOptionPane.PLAIN_MESSAGE`

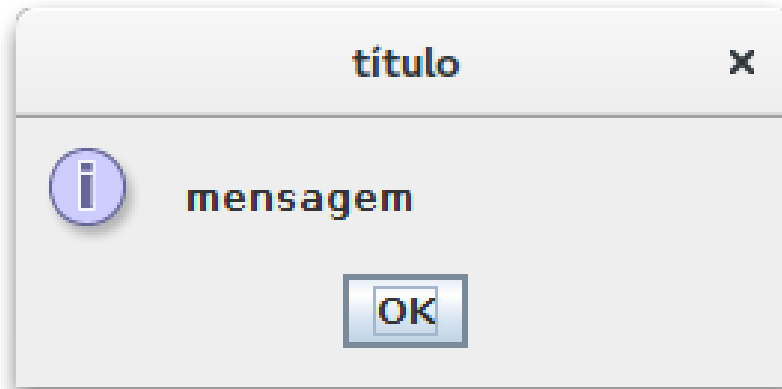
# Tipos da Caixa de Diálogo de Mensagem

- Erro:



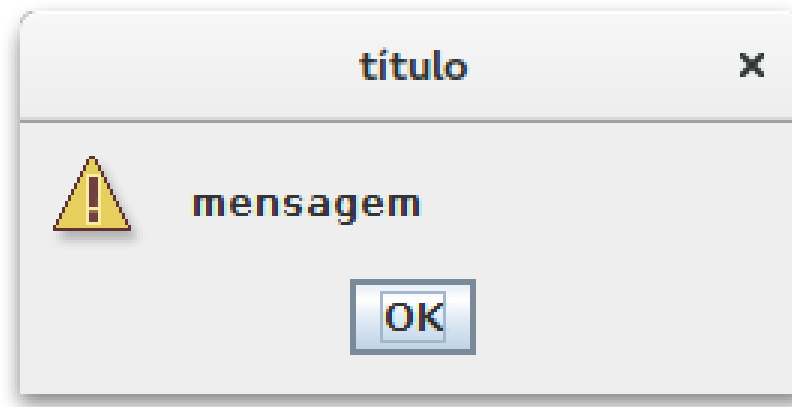
# Tipos da Caixa de Diálogo de Mensagem

- Informação:



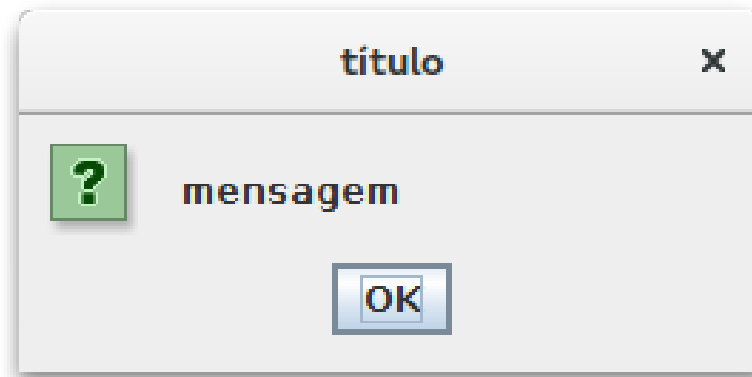
# Tipos da Caixa de Diálogo de Mensagem

- Aviso:



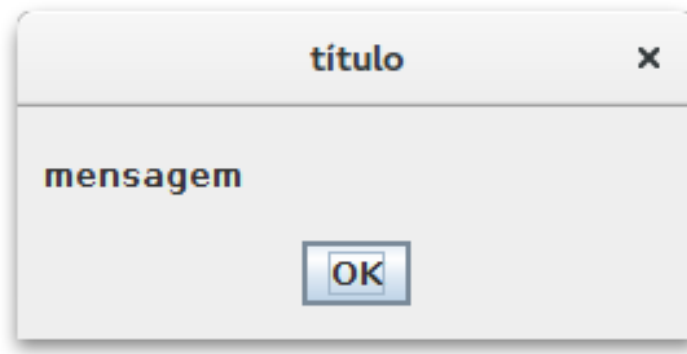
# Tipos da Caixa de Diálogo de Mensagem

- Pergunta:



# Tipos da Caixa de Diálogo de Mensagem

- Sem ícone:

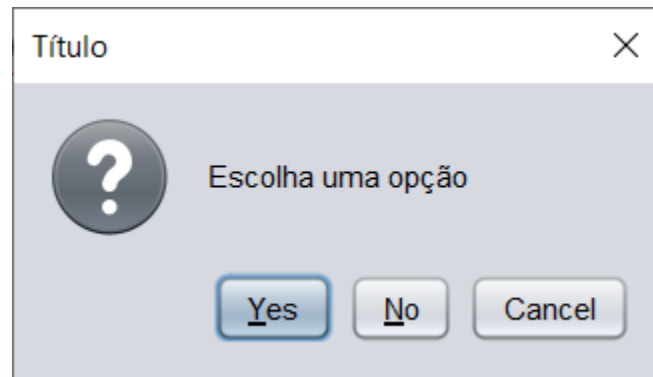


# Tipos da Caixa de Diálogo

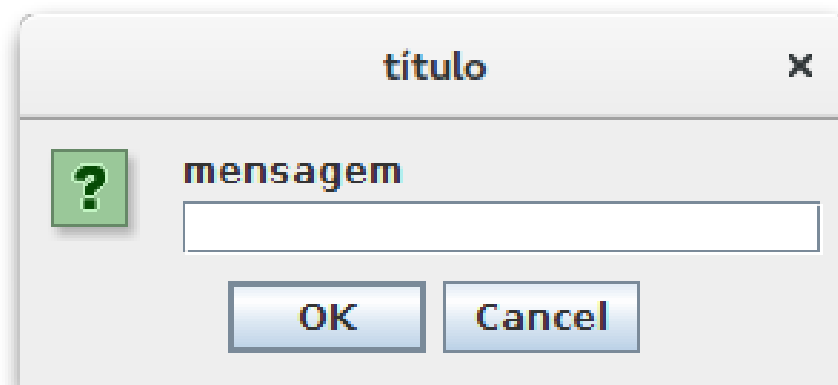
- `showOptionDialog(null, "mensagem" , "título", optionType, messageType, icon)`

## Exemplo:

```
int cap = JOptionPane.showConfirmDialog(null,  
    "Escolha uma opção", "Título",JOptionPane.YES_NO_CANCEL_OPTION);
```



# Caixas de Entrada de Dados





# Caixas de Entrada de Dados

- Serve para capturar uma mensagem em formato String.
- Configurável e versátil
- Muitas situações distintas, tais como:
  - Informação
  - Alerta
  - Mensagem de erro
- Método para mostrar a caixa de diálogo: `showInputDialog`

# Sintaxe da Caixa de Entrada de Dados

- `showInputDialog(null, “mensagem”);`
- `showInputDialog(null, “mensagem”, “título”, messageType);`
- `showInputDialog(null, “mensagem”, “título”, messageType, icone);`

Sendo que:

- `null`: indica que não existe dependência com outras janelas
- `“mensagem”`: é a mensagem que será mostrada
- `“título”`: é o título da caixa de diálogo
- `messageType`: é o tipo da caixa de diálogo
- `icone`: Caso queira mostra um ícone na mensagem

Retorna o texto no formato `String`.

# Sintaxe da Caixa de Opções

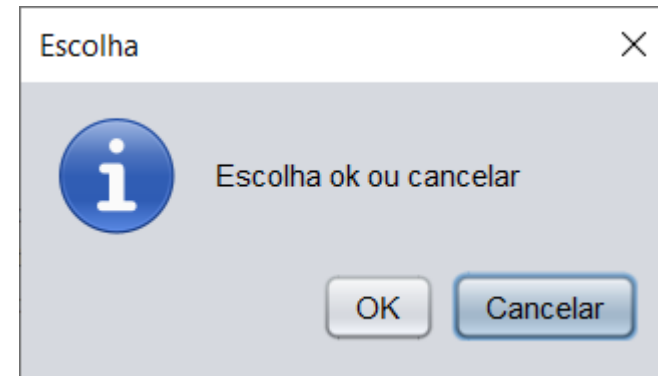
- `showOptionDialog(null, “mensagem”, "Escolha", “título”, optionType, messageType, icon, Object[] options, Object initialValue)`
  - Sendo que:
    - `null`: indica que não existe dependência com outras janelas
    - “mensagem”: é a mensagem que será mostrada
    - “título”: é o título da caixa de diálogo
    - `messageType`: é o tipo da caixa de diálogo
    - `icone`: Caso queira mostra um ícone na mensagem
    - `options`: Para mostrar as opções de botões.
    - `initialValue`: Qual botão começa selecionado.
- Retorna o index (int) do options selecionado pelo usuário.

# Sintaxe da Caixa de Opções

- `showOptionDialog(null, "mensagem", "Escolha", "título", optionType, messageType, icon, Object[] options, Object initialValue)`

## Exemplo:

```
String[] opcoes = {"OK", "Cancelar"};
```



```
int cap = JOptionPane.showOptionDialog(null, "Escolha ok ou cancelar",  
"Escolha", JOptionPane.DEFAULT_OPTION,  
JOptionPane.INFORMATION_MESSAGE, null, opcoes, opcoes[1]);
```

# Projeto Cadastro: Pegar os dados digitados e mostrar na caixa de diálogo

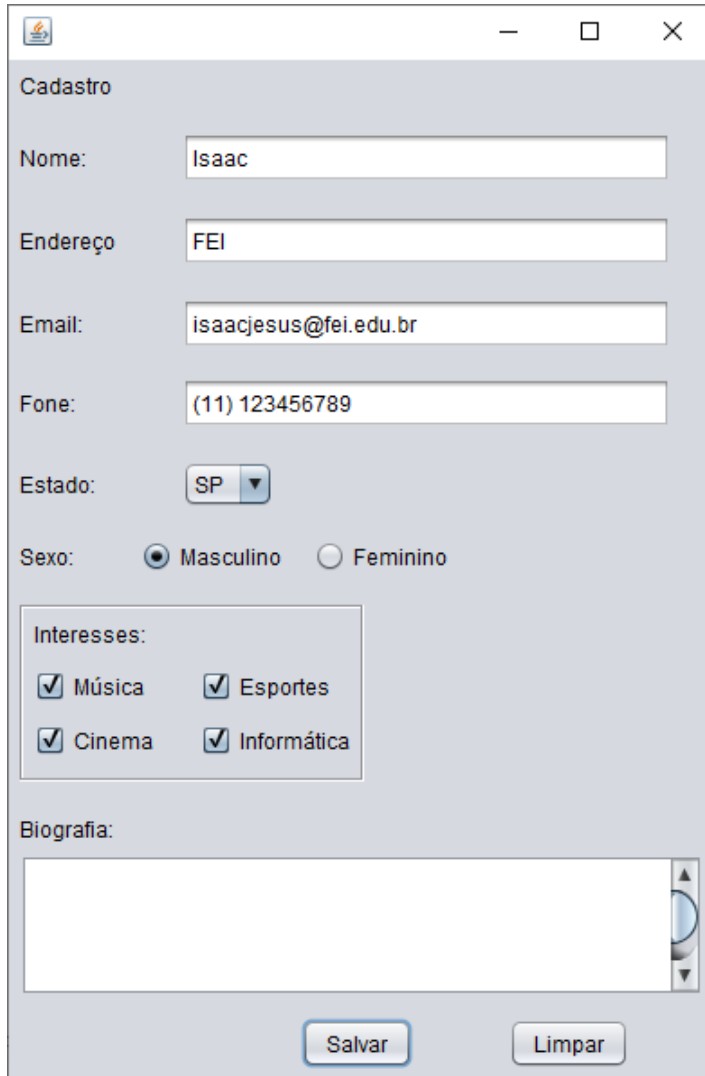
- Clicar duas vezes no botão Salvar para acessar o evento `btnSalvarActionPerformed`
- Mostrar na caixa de diálogo as variáveis que receberam os valores digitados nos campos de texto e na área de texto do formulário
  - Exemplo:  

```
JOptionPane.showMessageDialog(null, nome + "\n" + endereco, "Cadastro", JOptionPane.PLAIN_MESSAGE);
```
- Executar o projeto e testar o botão Salvar

# Projeto Cadastro: Pegar os dados digitados e mostrar na caixa de diálogo

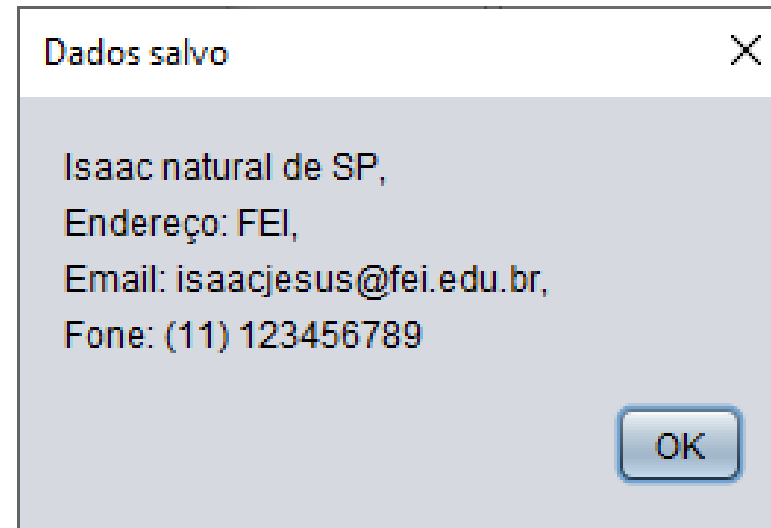
```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String nome = txtNome.getText();  
    String endereco = txtEndereco.getText();  
    String email = txtEmail.getText();  
    String fone = txtFone.getText();  
    String genero;  
    if (rbtFeminino.isSelected()) {  
        genero = "feminino";  
    } else {  
        genero = "masculino";  
    }  
    String naturalDe = (String) cmbxEstado.getSelectedItem();  
  
    String s = nome + " natural de " + naturalDe + ",\nEndereço: "  
        + endereco + ",\n"  
        + "Email " + email + ",\nFone" + fone;  
  
    JOptionPane.showMessageDialog(null, s, "Dados salvo", JOptionPane.PLAIN_MESSAGE);  
}
```

# Projeto Cadastro: Pegar os dados digitados e mostrar na caixa de diálogo



A screenshot of a registration form window titled "Cadastro". The form contains several input fields and controls:

- Nome:** Text field with "Isaac" entered.
- Endereço:** Text field with "FEI" entered.
- Email:** Text field with "isaacjesus@fei.edu.br" entered.
- Fone:** Text field with "(11) 123456789" entered.
- Estado:** Dropdown menu showing "SP".
- Sexo:** Radio buttons for "Masculino" (selected) and "Feminino".
- Interesses:** A group box containing four checked checkboxes: "Música", "Esportes", "Cinema", and "Informática".
- Biografia:** A large text area for a biography, currently empty.
- Buttons:** "Salvar" (Save) and "Limpar" (Clear) buttons at the bottom.



# Projeto Cadastro: Capturar os dados digitados e mostrar na caixa de diálogo

```
private void btnSalvarActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    String nome = txtNome.getText();  
    String endereco = txtEndereco.getText();  
    String email = txtEmail.getText();  
    String fone = txtFone.getText();  
    String genero;  
    if (rbtFeminino.isSelected()) {  
        genero = "feminino";  
    } else {  
        genero = "masculino";  
    }  
    String naturalDe = (String) cmbxEstado.getSelectedItem();  
    String interesses = "";  
    if (cnbxInformatica.isSelected())  
        interesses = interesses + "Informatica ";  
    if (cnbxMusica.isSelected())  
        interesses = interesses + "Música ";  
    if (cnbxEsporte.isSelected())  
        interesses = interesses + "Esporte ";  
    if (cnbxCinema.isSelected())  
        interesses = interesses + "Cinema ";  
    String s = nome + " natural de " + naturalDe + "\nEndereço: "  
        + endereco + "\n"  
        + "Email " + email + "\nFone: " + fone + "\n"  
        + "Sexo: " + genero + "\n"  
        + "Interesses: " + interesses;  
    JOptionPane.showMessageDialog(null, s, "Dados salvo", JOptionPane.PLAIN_MESSAGE);  
}
```



# Projeto Cadastro: Capturar os dados digitados e mostrar na caixa de diálogo



Cadastro

Nome:

Endereço:

E-mail:

Fone:

Estado:

Sexo: ☒ Masculino ☐ Feminino

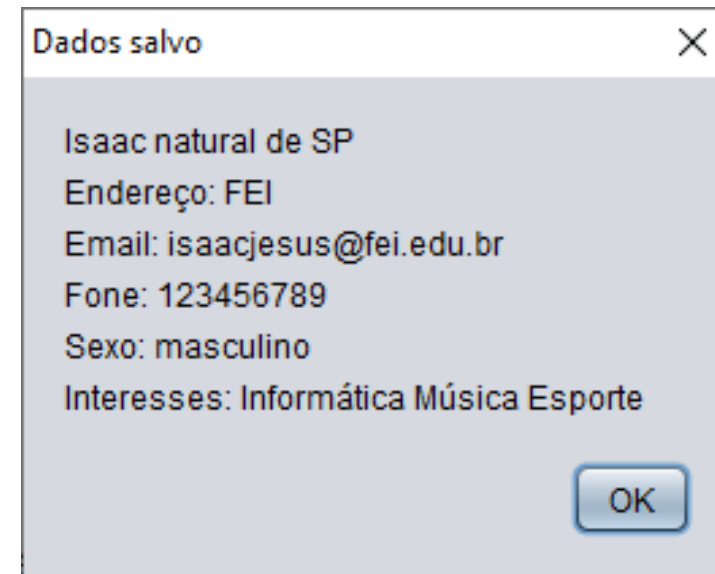
Interesses

☒ Música ☒ Esportes

☐ Cinema ☒ Informática

Biografia:

Salvar Limpar



Dados salvo

Isaac natural de SP

Endereço: FEI

Email: isaacjesus@fei.edu.br

Fone: 123456789

Sexo: masculino

Interesses: Informática Música Esporte

OK

Modificar os textos e escolhas  
dos componentes

# Colocar dados nos componentes (setText)

- Para “colocar” dados nos componentes usamos o método setText()
- Podemos usar esse método em componentes como Label, Campo de Texto e Área de Texto
- Sintaxe:

`nomeDoComponente.setText(<texto>);`

- Exemplos:
  - `txtNome.setText("Isaac");`
  - `lblValor.setText(12345.67);`

# Formatar o valor “colocado” no componente

- Para formatar o valor é necessário importar uma biblioteca e usar o método `format()`, conforme exemplo a seguir:

```
import java.text.NumberFormat;
...
//retorna uma string no formato da moeda atual
Double numero = 3.1415
String valor = NumberFormat.getCurrencyInstance().format(numero);
//Retorna a String valor = “R$ 3.14”

//String.format funciona como o printf
lblValor.setText(String.format("%.2f", valor));
```

# Projeto Cadastro: Limpar os valores dos campos e áreas de texto

- Clicar duas vezes no botão Limpar para criar o evento `btnSalvarActionPerformed`
- Atribuir valor em branco ("" ) para o texto de todos os componentes campo de texto e área de texto
  - Exemplo: `txtNome.setText("");`
- Executar o projeto e testar o botão Limpar

# Projeto Cadastro: Limpar os valores dos campos e áreas de texto

```
private void btnLimparActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    txtNome.setText("");  
    txtEndereco.setText("");  
    txtEmail.setText("");  
    txtFone.setText("");  
}
```

# Limpar as seleções feitas nos Botões e Caixas

- Para limpar a seleção feita a Caixa de Combinação devemos atribuir o valor -1 ao índice da caixa
- Sintaxe:  
`nomeDoComponente.setSelectedIndex(-1);`
- Exemplo:  
`cmbxEstado.setSelectedIndex(-1);`

# Limpar as seleções feitas nos Botões e Caixas

- Para limpar a seleção feita no Grupo de Botões usamos o método `clearSelection()`;

- Sintaxe:

```
nomeDoComponente.clearSelection();
```

- Exemplo:

```
if (rdbtMasculino.isSelected() || rdbtFeminino.isSelected()) {  
    grpSexo.clearSelection();  
}
```



# Limpar as seleções feitas nos Botões e Caixas

- Para limpar a seleção feita Caixa de Seleção usamos o método `setSelected()` e atribuímos valor *false* à Caixa

- Sintaxe:

```
nomeDoComponente.setSelected(false);
```

- Exemplo:

```
if (chbxMusica.isSelected()) {  
    chbxMusica.setSelected(false);  
}
```

# Projeto Cadastro: Limpar as seleções feitas nos Botões e Caixas

- Clicar duas vezes no botão Limpar para acessar o evento `btnSalvarActionPerformed`
- Atribuir valor -1 para a Caixa de Combinação que seleciona o Estado  
Exemplo: `cmbxEstado.setSelectedIndex(-1);`
- Limpar a seleção do Grupo de Botões que seleciona o Sexo. Exemplo:  

```
if (rdbtMasculino.isSelected() || rdbtFeminino.isSelected()) {  
    grpSexo.clearSelection();  
}
```

# Projeto Cadastro: Limpar as seleções feitas nos Botões e Caixas

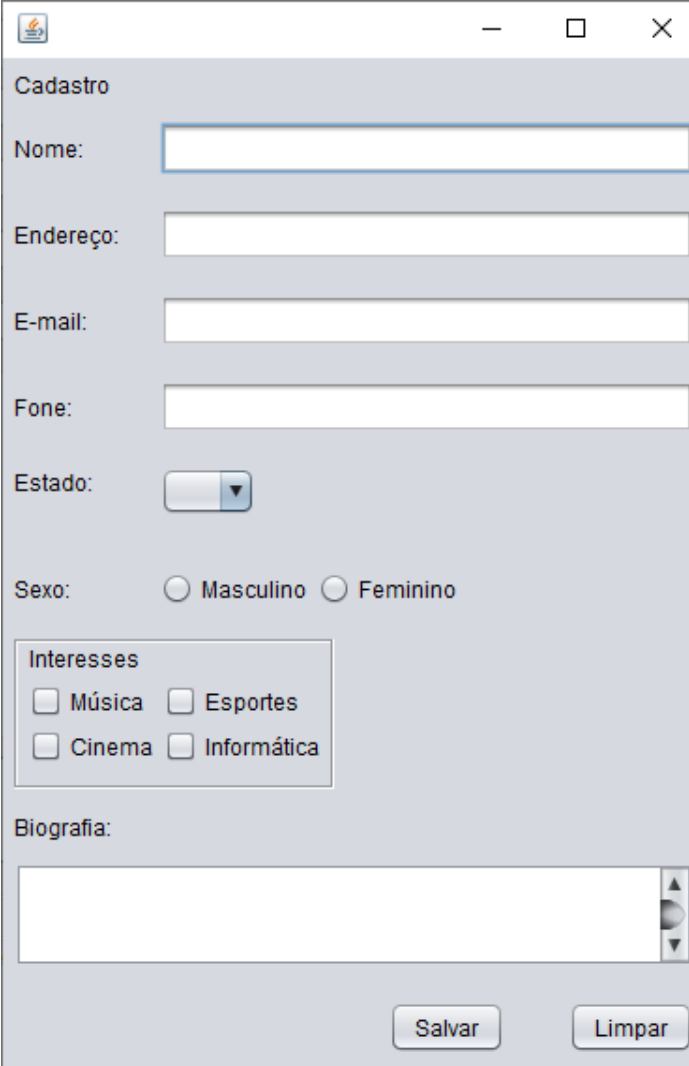
- Atribuir *false* às Caixas de Seleção que selecionam os Interesses. Exemplo:

```
if (chbxMusica.isSelected()) {  
    chbxMusica.setSelected(false);  
}
```
- Executar o projeto e testar o botão Limpar

# Projeto Cadastro: Limpar as seleções feitas nos Botões e Caixas

```
private void btnLimparActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    txtNome.setText("");  
    txtEndereco.setText("");  
    txtEmail.setText("");  
    txtFone.setText("");  
    grpSexo.clearSelection();  
    cmbxEstado.setSelectedIndex(-1);  
    cnbxCinema.setSelected(false);  
    cnbxEsporte.setSelected(false);  
    cnbxInformatica.setSelected(false);  
    cnbxMusica.setSelected(false);  
}
```

# Projeto Cadastro: Limpar as seleções feitas nos Botões e Caixas



A screenshot of a software window titled "Cadastro" (Registration). The window contains the following elements:

- Nome:** A text input field.
- Endereço:** A text input field.
- E-mail:** A text input field.
- Fone:** A text input field.
- Estado:** A dropdown menu.
- Sexo:** Two radio buttons labeled "Masculino" and "Feminino".
- Interesses:** A group box containing four checkboxes: "Música", "Esportes", "Cinema", and "Informática".
- Biografia:** A multi-line text area with a vertical scrollbar.
- Buttons:** Two buttons at the bottom right, labeled "Salvar" (Save) and "Limpar" (Clear).