

# Java SWING

## MATA55 - Programação Orientada a Objetos

Profº Frederico Durão  
freddurao@dcc.ufba.br

Instituto de Matemática  
Departamento de Ciência da Computação  
Universidade Federal da Bahia

23 de abril de 2014



# Agenda I

## 1 Introdução

- O que é Java Swing
- Por que estudar Java Swing
  - Swing vs. others
- Como usar o Java Swing
  - Codificando no Editor convencional
  - Metodo Drop & Drag - Plugin WindowBuilder para Eclipse
  - WindowBuilder - Instalando
  - WindowBuilder - Perspectiva
  - Botões
- Eventos
- Javadoc
- Exemplos

## 2 Referências

# Introdução

## O que é Java Swing

- Swing é uma API Java para interfaces gráficas.
- Faz parte da JFC (Java Foundation Classes) que encapsula um grupo de 'features' para GUIs (Graphical User Interfaces).

# Introdução

Por que estudar Java Swing

[enumerar vantagens]

# Introdução

Por que estudar Java Swing

## Swing vs. outros

- Aprendendo Swing aprende todos
- EXT-GWT, Android, JSF + PrimeFaces, etc.

# Introdução

## Como usar o Java Swing

- Editor convencional
- Plugin Window Builder

# Introdução I

## Como usar o Java Swing

### Codificando no Editor convencional

---

```
package br.ufba.mata55;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.SwingUtilities;

public class HelloWorldSwing extends JFrame {

    public HelloWorldSwing() {
        initUI();
    }

    private void initUI() {

        JPanel panel = new JPanel();
        getContentPane().add(panel);
```

# Introdução II

## Como usar o Java Swing

```
panel.setLayout(null);

setTitle("Hello World!");
setSize(300, 200);
setLocationRelativeTo(null);
setDefaultCloseOperation(EXIT_ON_CLOSE);
}

public static void main(String[] args) {

    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            HelloWorldSwing ex = new HelloWorldSwing();
            ex.setVisible(true);
        }
    });
}
```



# Introdução

## Como usar o Java Swing

- Ao executar como projeto **Java** normal, o resultado será:

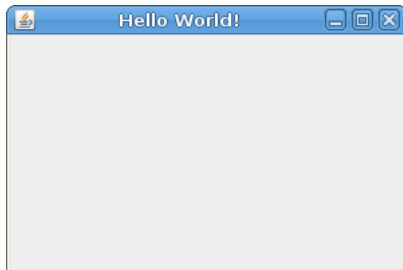


Figura: HelloWorldSwing.java

# Introdução

## Criando Projeto

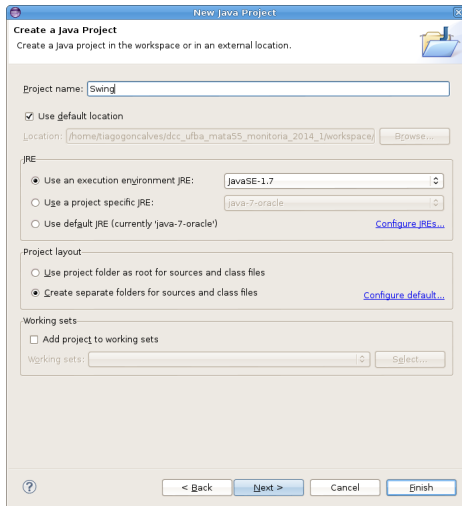


Figura: Projeto Java

# Introdução

## Criando Classe Principal

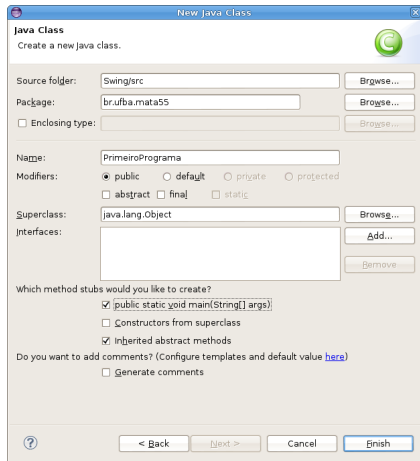


Figura: PrimeiroPrograma.java

# Página principal do Window Builder para Eclipse

<https://www.eclipse.org/windowbuilder/>

**eclipse**

Visit other Eclipse Sites

Home Downloads Users Members Committers Resources Projects About Us

Google™ Custom Search Search

Onde baixar >>

Como usar >>

**Download**  
Eclipse Distribution,  
Update Site, Dropins

**Documentation**  
Tutorials, Examples,  
Videos, Online Reference

**Support**  
Bug Tracker, Newsgroup  
Professional Support

**Getting Involved**  
CVS, Workspace Setup,  
Wiki, Committers

**WindowBuilder - is a powerful and easy to use bi-directional Java GUI designer**

This project was just provisioned. You can see the proposal [here](#).

WindowBuilder is composed of SWT Designer and Swing Designer and makes it very easy to create Java GUI applications without spending a lot of time writing code. Use the WYSIWYG visual designer and layout tools to create simple forms to complex windows; the Java code will be generated for you. Easily add controls using drag-and-drop, add event handlers to your controls, change various properties of controls using a property editor, internationalize your app and much more.

WindowBuilder is built as a plug-in to Eclipse and the various Eclipse-based IDEs (RCP, RSE, MyEclipse, JBuilder, etc.).

**Current Status**

Welcome to the new WindowBuilder homepage! We are thrilled to see WindowBuilder emerge as a new open source project, and we are excited to work with the Eclipse community to grow and evolve the tool.

We are in the process of getting the initial code contributions into Eclipse and into the IP review process. In the meantime, if you are interested in the project, you can see the current docs for

Figura: Página principal do Window Builder

# WindowBuilder - Instalando

## Help > Instal New Software

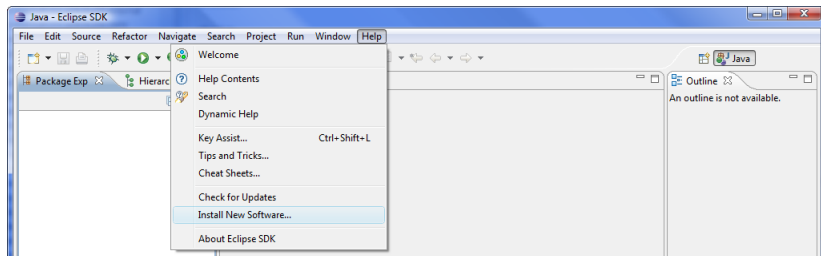


Figura: Instalando Window Builder

# WindowBuilder - Instalando

Clique em **Add**.

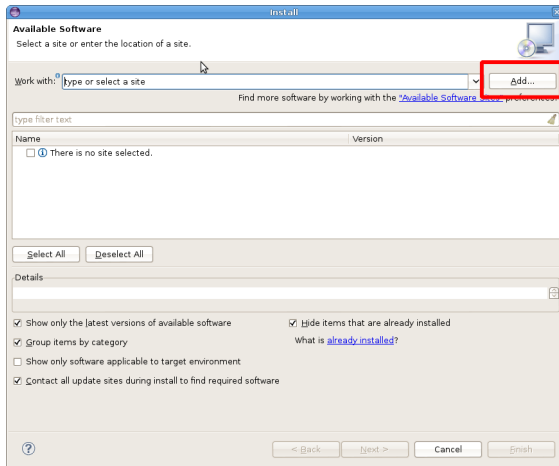


Figura: Instalando Window Builder

# WindowBuilder - Instalando

Em **Name** digite **WindowBuilder** (pode ser qualquer nome).

Em **Location** digite a url

<http://download.eclipse.org/windowbuilder/WB/release/R201309271200/4.3/>

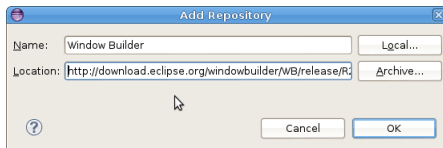


Figura: Instalando Window Builder

# Abrindo Editor do Window Builder

Clique com o botão direito na classe `PrimeiroPrograma.java` . Selecione **Open with** e depois **WindowBuilder Editor**.

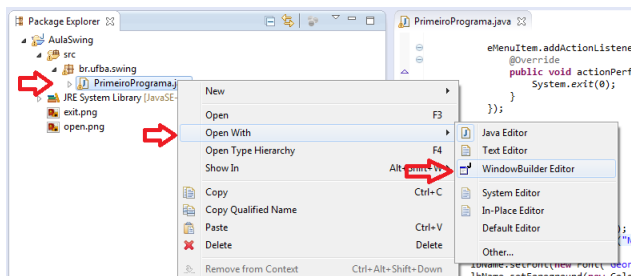


Figura: Abrindo Editor do Window Builder



# Abrindo Editor do Window Builder

- **Source** - Como o Editor padrão do Eclipse.
- **Design** - Editor gráfico do Window Builder.

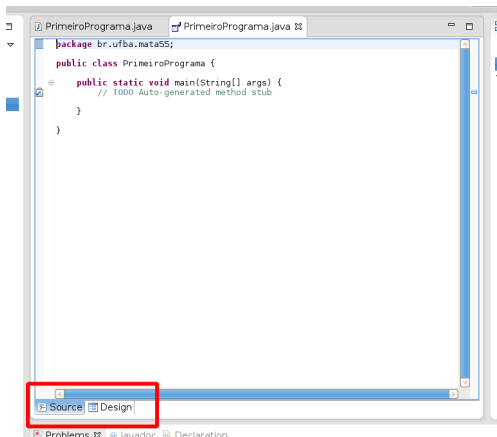


Figura: Editor do Windows Builder

# Aba Design do Editor do Window Builder

- Visualizador gráfico.

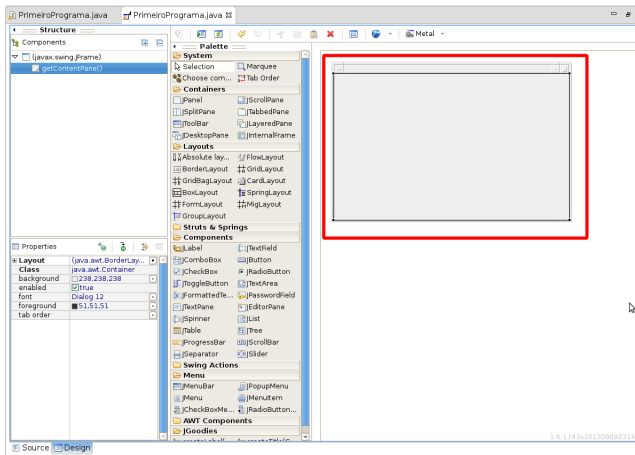


Figura: Aba Design

# Aba Design do Editor do Window Builder

- Paleta com os principais componentes do Swing.

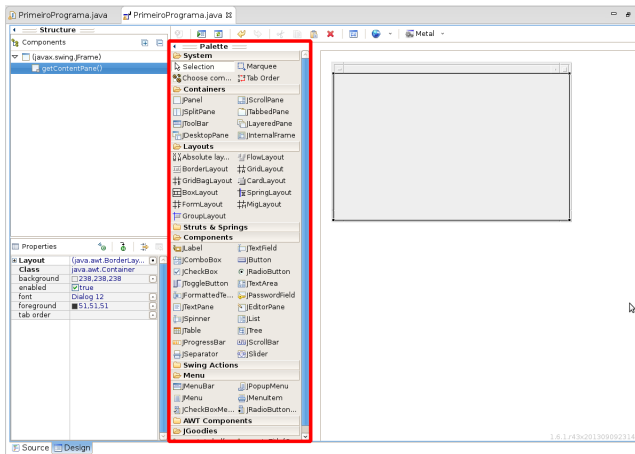
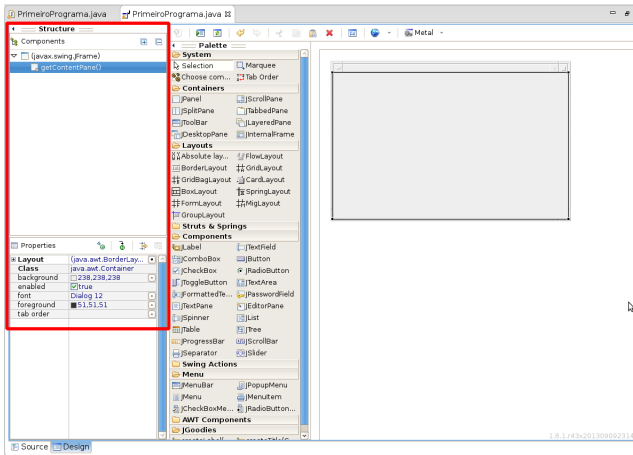


Figura: Aba Design

# Aba Design do Editor do Window Builder

- **Structure** - estrutura da tela com seus componentes.
- **Properties** - propriedades do componente selecionado.



### Figura: Estrutura e propriedades

# Construindo a janela

Inserindo um título à janela.

Selecione a janela e insira em **Title** na aba **Properties**.

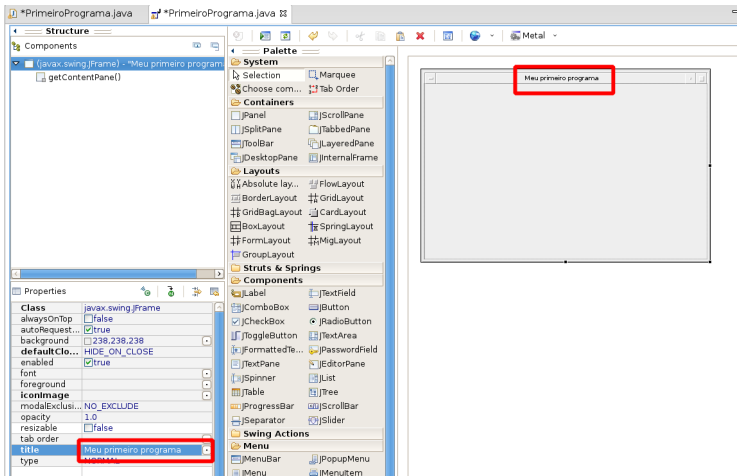


Figura: Estrutura e propriedades

# Construindo a janela

O equivalente em código seria.

---

```
public PrimeiroPrograma() {  
    setTitle("Meu primeiro programa");  
}
```

---

# Construindo uma tela - JPanel

- É um **simple container** (recipiente) **genérico**.
- É como se fosse um receptáculo onde os componentes podem ser agrupados.
- Pode ter seu próprio **layout**, portanto suas próprias regras.

# Construindo uma tela - JPanel

- O Window Builder utiliza o método **Drag & Drop** (clica e solta)
- As áreas verdes são os locais onde o componente pode ser inserido
- As divisões entre as áreas verdes são ditadas pelo **layout** escolhido
- Por padrão o layout é o **BorderLayout** que divide a tela em **NORTH**, **SOUTH**, **WEST**, **CENTER** e **EAST**.

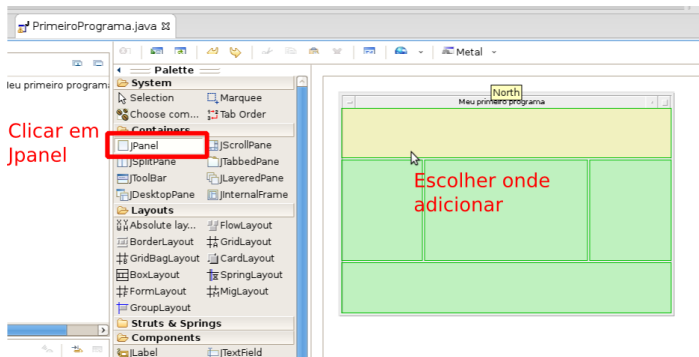


Figura: JPanel



# Construindo uma tela - JPanel

Em código:

---

```
package br.ufba.mata55;

import javax.swing.JFrame;
import javax.swing.JPanel;
import java.awt.BorderLayout;

public class PrimeiroProgramaPura extends JFrame {
    public PrimeiroProgramaPura() {

        JPanel panel = new JPanel();
        getContentPane().add(panel, BorderLayout.NORTH);
    }
}
```

---

# Construindo uma tela - JLabel

- É um rótulo.
- Pode exibir um texto, uma imagem ou ambos.

# Construindo uma tela - JLabel

- Ao inserir o JLabel já defina o nome!

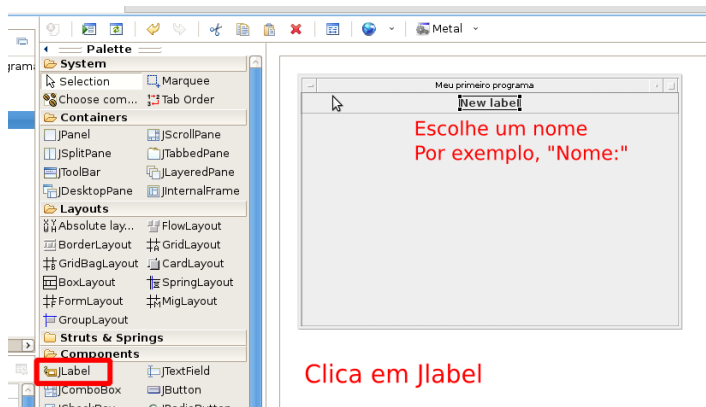


Figura: JLabel

# Construindo uma tela - **JTextField**

- É um componente que nos permite a edição de um simples linha de texto

# Construindo uma tela - JTextField

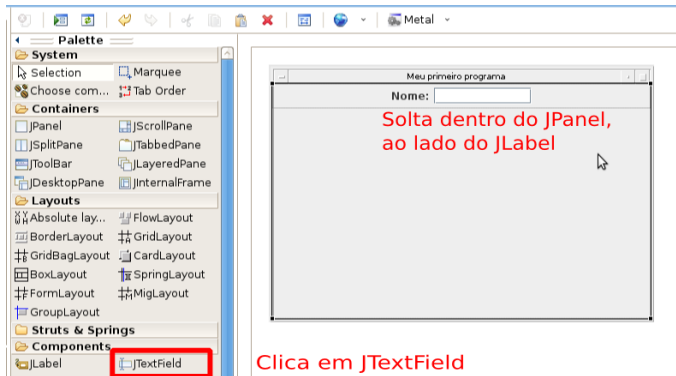


Figura: JTextField

# Construindo uma tela - Alinhando os componentes

- Os tipos de alinhamento disponíveis são : **LEFT**, **RIGHT**, **CENTER**, **LEADING**(bottom) e **TRAILING** (top)

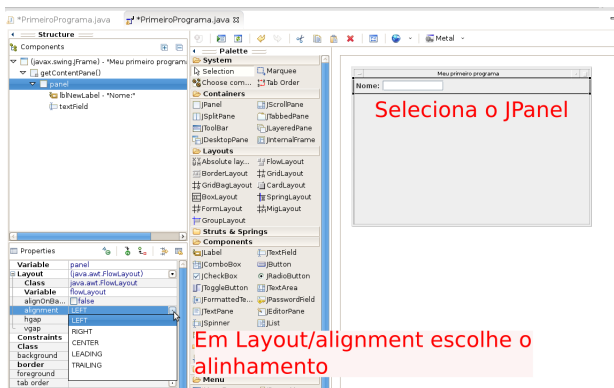


Figura: Alinhando componentes

# Construindo uma tela - Alinhamento

Em código:

---

```
JPanel panel = new JPanel();
FlowLayout flowLayout = (FlowLayout) panel.getLayout();
flowLayout.setAlignment(FlowLayout.LEFT);
getContentPane().add(panel, "1, 1, fill, top");
```

---

# Construindo uma tela - JPasswordField

- Um JTextField não é o componente mais adequado para um campo de senha, por exemplo.
- Para isso existe o componente JPasswordField.



# Construindo uma tela - JPasswordField

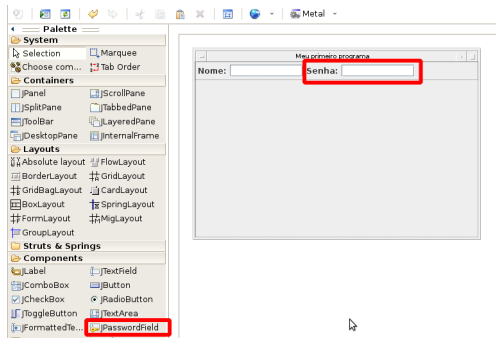


Figura: JPassword

# Construindo uma tela - Simulação

- É possível simular o funcionamento da tela antes mesmo de executar o código.

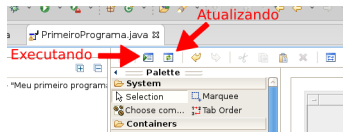


Figura: Simulando

# Construindo uma tela - Simulação do JPasswordField

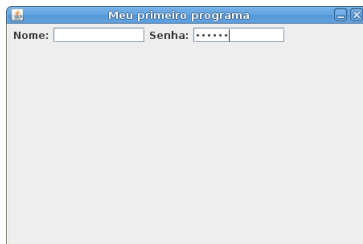


Figura: Simulando o uso do JPasswordField

# Construindo uma tela - JPasswordField

- A implementação do JPasswordField é semelhante ao do JTextField.

---

```
passwordField = new JPasswordField();  
passwordField.setColumns(10);  
panel.add(passwordField);
```

---

# Construindo uma tela - JTextArea

- É uma área multilinha que mostra um texto.
- Dentro de um **JScrollPane** um texto relativamente grande pode caber em uma tela pequena.

# Construindo uma tela - JTextArea

- Foi adicionado um outro **JPane** simples que abrange todo o resto da tela.
- Dentro deste **JPane** foi criado o **JLabel** Descrição.
- Dentro deste **JPane** foi adicionado um **JScrollPane**.
- Finalmente dentro deste **JScrollPane** foi adicionado o **JTextArea**

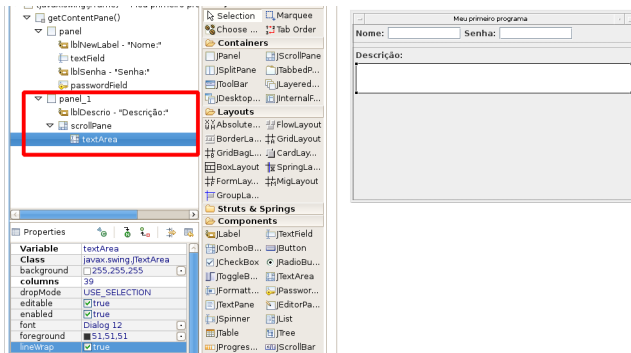


Figura: Estrutura da tela

# Construindo uma tela - JTextArea

- Em **Properties** > **columns** foi definida a 'largura' do **JTextArea**.
- Em **Properties** > **lineWrap** foi definida a quebra de linha do texto.

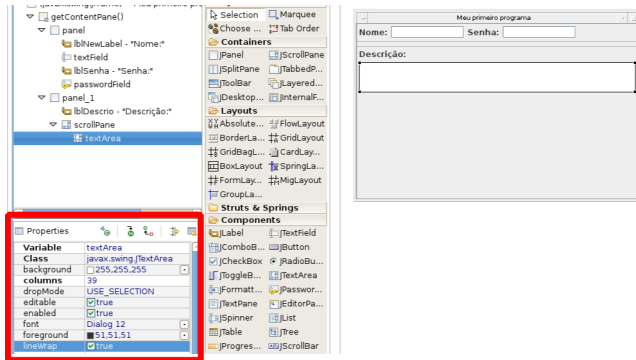


Figura: Propriedades do JTextArea





# Construindo uma tela - JTextArea

Em código seria:

---

```
JScrollPane scrollPane = new JScrollPane();  
panel_1.add(scrollPane);
```

```
JTextArea textArea = new JTextArea();  
textArea.setLineWrap(true);  
textArea.setWrapStyleWord(true);  
textArea.setRows(3);  
textArea.setColumns(39);  
scrollPane.setViewportView(textArea);
```

---

# Construindo uma tela - **JButton**

- Um **JButton** é um botão convencional.
- Podem ser controlados por **Eventos**.

# Construindo uma tela - JButton

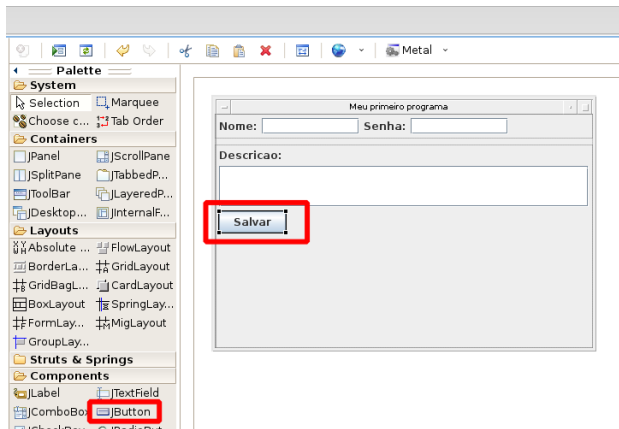


Figura: JButton

# Construindo uma tela - JButton

---

```
JButton btnSalvar = new JButton("Salvar");  
panel_1.add(btnSalvar);
```

---

# Construindo uma tela - JCheckBox

- É uma caixa de seleção.
- Um **JCheckBox** pode ser marcado ou desmarcado.

# Construindo uma tela - JCheckBox

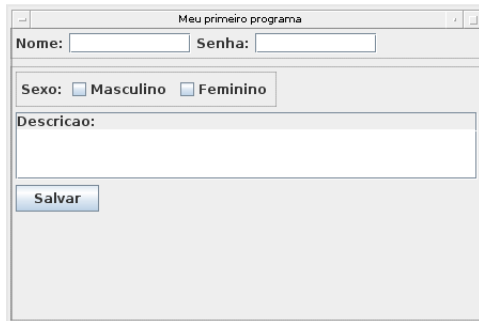
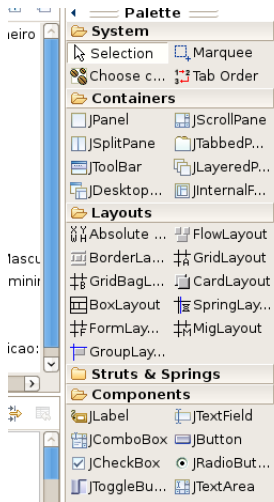


Figura: JCheckBox

# Construindo uma tela - JCheckBox

---

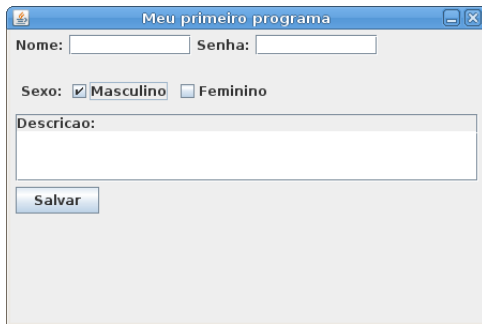
```
JCheckBox chckbxMasculino = new JCheckBox("Masculino");  
panel_2.add(chckbxMasculino);
```

```
JCheckBox chckbxFeminino = new JCheckBox("Feminino");  
panel_2.add(chckbxFeminino);
```

---

# Construindo uma tela - JCheckBox

- Exemplo de seleção do sexo do usuário.



The image shows a Java Swing window titled "Meu primeiro programa". Inside the window, there is a form with the following elements:

- Two text input fields: "Nome:" and "Senha:".
- A "Sexo:" label followed by two radio buttons: "Masculino" (which is checked) and "Feminino" (which is unchecked).
- A text area labeled "Descricao:".
- A "Salvar" button at the bottom left.

Figura: Simulação do JCheckBox



## Construindo uma tela - JCheckBox

- Exemplo de seleção do sexo do usuário.
- Existe um problema em utilizar **JCheckBox** nesse exemplo: **É possível selecionar mais de uma caixa de seleção.**
- A solução é usar **JRadioButton**.

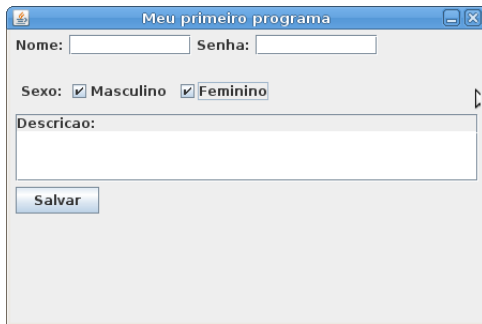


Figura: Simulação do JCheckBox

# Construindo uma tela - **JRadioButton**

[O objetivo é seguir a mesma lógica acima]

# Construindo uma tela - JMenuitem

[O objetivo é seguir a mesma lógica acima]

# Construindo uma tela - **JToggleButton**

[O objetivo é seguir a mesma lógica acima]

[Inserir eventos nos elementos que já existem na tela]

# Javadoc

Explorando o javadoc

[Mostrar o javadoc]

# Exemplos

## Exemplos

[Mostrar a lista de componentes e applets do site oficial]

- Window Builder - <https://www.eclipse.org/windowbuilder/>
- Swing tutorial I - <http://www.tutorialspoint.com/swing/>
- Swing tutorial II - <http://www.wikihow.com/Create-a-Swing-GUI-in-Java>
- Swing tutorial III - <http://zetcode.com/tutorials/javaswingtutorial/>
- Componentes do Swing -  
<http://docs.oracle.com/javase/tutorial/uiswing/components/>
- Página oficial do Swing - <http://docs.oracle.com/javase/tutorial/uiswing/>
- Swing javadoc -  
<http://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html>