UNIP
UNIVERSIDADE PAULISTA

UNICLUSTER

JAVA *SWING*

Aplicações de Linguagem de Programação Orientada a Objetos

Carlos Arruda Baltazar

UNIP

Componentes *SWING*

Os componentes GUI Swing estão dentro do pacote javax.swing que são utilizados para construir as interfaces gráficas. Alguns componentes não são do tipo GUI Swing e sim componentes AWT. Antes de existir o GUI Swing, o Java tinha componentes AWT (Abstract Windows Toolkit) que faz parte do pacote javax.awt.

Componentes *SWING*

A diferença entre o GUI Swing e AWT, é na aparência e comportamento dos componentes, ou seja, quando criado por AWT, a aparência e comportamento de seus componentes são diferentes para cada plataforma e enquanto feito por GUI Swing, a aparência e comportamento funcionam da mesma forma para todas as plataformas.

Componentes *SWING*

Os componentes AWT são mais pesados, pois requerem uma interação direta com o sistema de janela local, podendo restringir na aparência e funcionalidade, ficando menos flexíveis do que os componentes GUI Swing.

Com Swing, não importa qual sistema operacional, qual resolução de tela, ou qual profundidade de cores: sua aplicação se comportará da mesma forma em todos os ambiente.

Gerenciador de *Layout*

Todos os gerenciadores de layout implementam a interface *LayoutManager* que faz parte do pacote *java.awt*. Existe o método *setLayout* da classe Container que aceita um objeto que implementa a interface *LayoutManager* como um argumento.

Gerenciador de *Layout*

Veja três maneiras básicas de organizar componentes em uma GUI:

- **Posicionamento absoluto:** fornece o maior nível de controle sobre a aparência de uma GUI, pois configura o layout de um Container como **null** podendo especificar a posição absoluta de cada componente GUI em relação ao canto superior esquerdo do Container usando os métodos Component **setSize** e **setLocation** ou **setBounds**.
- **Gerenciadores de layout:** é mais simples e mais rápido para criar uma posição GUI com posicionamento absoluto, mas acaba perdendo controle sobre tamanho e o posicionamento dos componentes GUI.
- **Programação visual em uma IDE:** facilita o desenvolvimento em GUI, pois toda IDE tem uma ferramenta de design que permite arrastar e soltar (**drag and drop**) os componentes para uma área de desenho.

Gerenciador de *Layout*

Cada contêiner individual pode ter apenas um gerenciador de ***layout***, mas vários contêineres no mesmo aplicativo podem utilizar cada um gerenciador de ***layout***. Os gerenciadores de layout são diversos, mas nesse artigo vamos conhecer o ***FlowLayout***, ***BorderLayout*** e ***GridLayout*** como apresentado abaixo.

FlowLayout

Ocorre quando os componentes GUI são colocados em um contêiner da esquerda para a direita na ordem em que são adicionados no contêiner. Quando a borda do contêiner é alcançada, os componentes continuarão a ser exibidos na próxima linha. A classe ***FlowLayout*** permite aos componentes GUI ser alinhados à esquerda, centralizados (padrão) e alinhados à direita.

FlowLayout

Os componentes ficam centralizados por padrão. A ação desse aplicativo será relacionada aos botões. Quando o usuário clicar no botão “Direita” o alinhamento do gerenciador muda para um ***FlowLayout*** à direita e assim acontece com os outros botões quando clicado, cada um com seu alinhamento.

FlowLayout

Cada botão tem seu próprio ***handler*** de evento que é declarado com um uma classe interna anônima que implementa ***ActionListener***.

BorderLayout

É um gerenciador de *layout* que organiza os componentes, sendo a parte superior do contêiner dividida em cinco regiões:

- NORTH
- SOUTH
- EAST
- WEST
- CENTER

A classe *BorderLayout* estende a *Object* e implementa a interface *LayoutManager2* sendo uma subinterface de *LayoutManager* que adiciona vários métodos para obter um processamento de *layout* aprimorado.

GridLayout

É um gerenciador de *layout* que divide o contêiner em uma grade de modo que os componentes podem ser colocados nas linhas e colunas. A classe *GridLayout* estende a classe *Object* e implementa a interface *LayoutManager*.

Cada componente no *GridLayout* tem os mesmos tamanhos, onde podem ser inserida uma célula na parte superior esquerda da grade que prossegue da esquerda para a direita até preencher por completa.

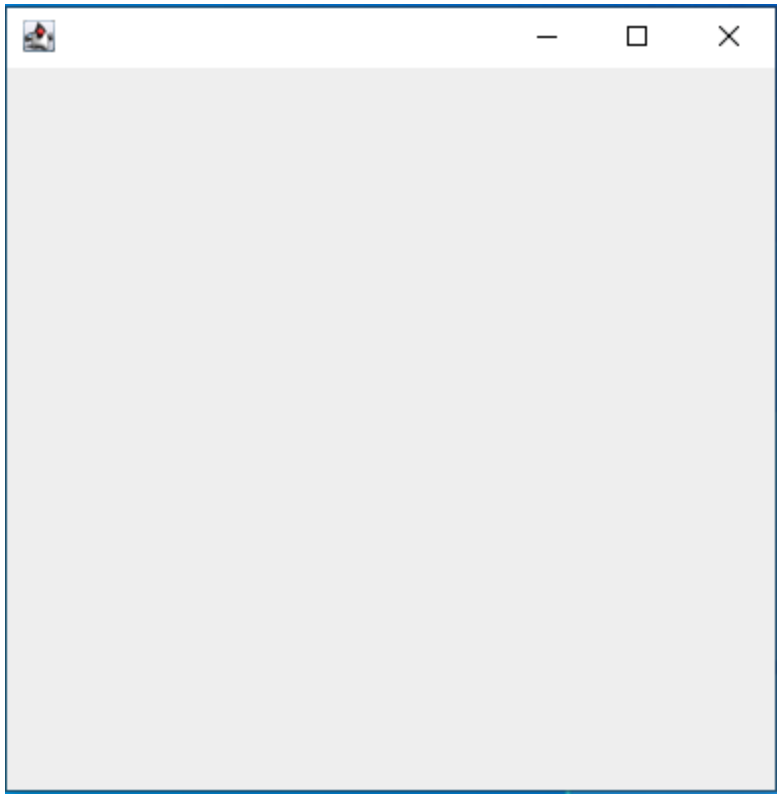
JFrame

JFrame é um objeto para instanciar a janela do programa, contendo componentes como: barra de título; ícone; botões; painéis etc. Entre os principais métodos temos:

- ***pack()*** que compacta a janela para o tamanho dos componentes;
- ***setSize(int, int)*** que define as medidas largura e altura da janela em pixels;
- ***setTitle(string)*** que define o título da janela;
- ***setLocation(int, int)*** que define a posição da janela na tela por coordenadas **(x,y)** em pixels;
- ***setBounds(int, int, int, int)*** que agrega as propriedades dos métodos ***setLocation*** e ***setSize***;
- ***setVisible(boolean)*** que define se a janela será visível ou não;
- ***setDefaultCloseOperation(int)*** que define o que ocorre quando o usuário tenta fechar a janela;
- ***setLayout(LayoutManager)*** que altera o tipo de gerenciador de layout.

JFrame


```java
package Pck_Swing;

import javax.swing.*;
import java.awt.*;

public class MeuPrimeiroJFrame extends JFrame
{
    public MeuPrimeiroJFrame()
    {
        this.setVisible(true);
        this.setSize(400,400);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setLayout(new FlowLayout());
    }
    public static void main (String args[])
    {
        MeuPrimeiroJFrame frame = new MeuPrimeiroJFrame();
    }
}
```



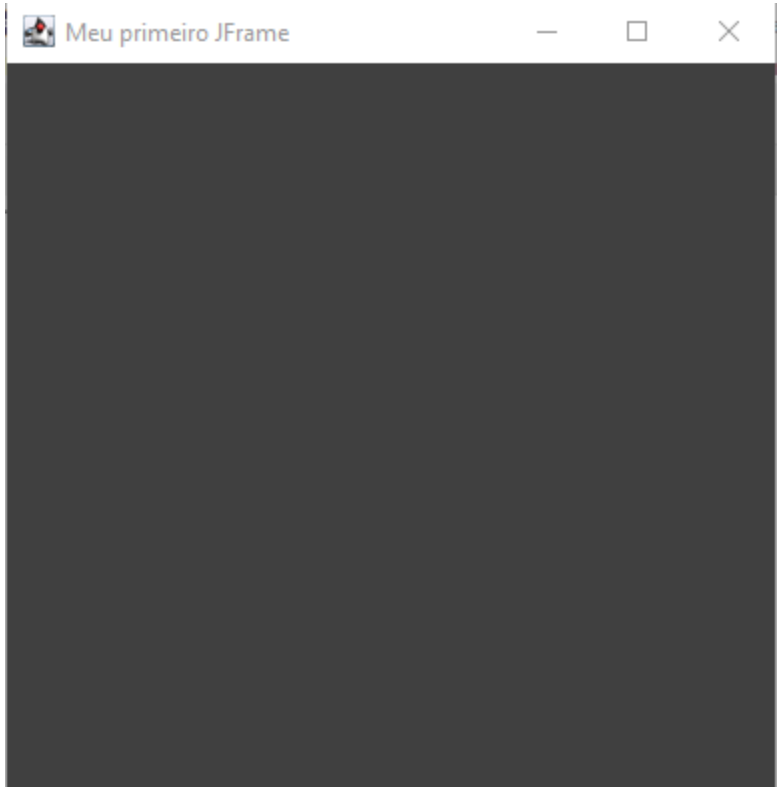
JPanel

JPanel representa um tipo básico de container para inserção de componentes, inclusive de outro ***JPanel***. Entre os principais métodos temos:

- ***add(Component)*** que adiciona um componente ao painel;
- ***setLayout(LayoutManager)*** que altera o tipo de gerenciador de layout.

JPanel


```java
package Pck_Swing;

import javax.swing.*;
import java.awt.*;

public class MeuPrimeiroJFrame extends JFrame
{
    public JPanel meuPainel;


    public MeuPrimeiroJFrame()
    {
        this.setVisible(true);
        this.setTitle("Meu primeiro JFrame");
        this.setSize(400,400);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setLayout(new BorderLayout());

        meuPainel = new JPanel();
        meuPainel.setLayout(new FlowLayout());
        meuPainel.setBackground(Color.DARK_GRAY);
        this.add(meuPainel, BorderLayout.CENTER);
    }
    public static void main (String args[])
    {
        MeuPrimeiroJFrame frame = new MeuPrimeiroJFrame();
    }
}
```



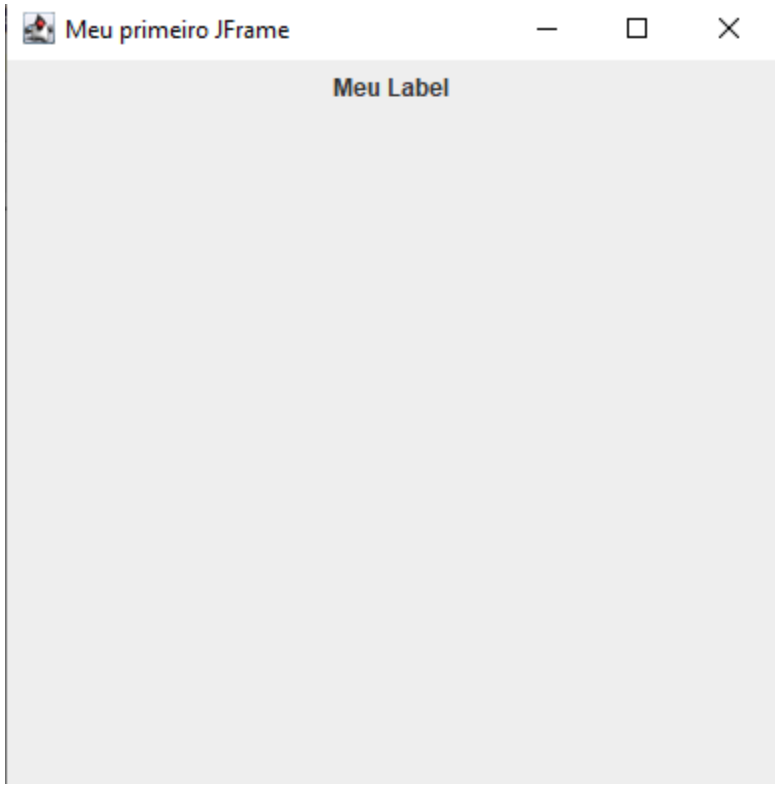
JLabel

JLabel representa um rótulo/etiqueta de texto. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto à etiqueta (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***getText()*** que retorna o texto contido na etiqueta.
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;

JLabel


```
package Pck_Swing;

import javax.swing.*;
import java.awt.*;

public class MeuPrimeiroJFrame extends JFrame
{
    public JPanel meuPainel;
    public JLabel meuLabel;

    public MeuPrimeiroJFrame()
    {
        this.setVisible(true);
        this.setTitle("Meu primeiro JFrame");
        this.setSize(400,400);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setLayout(new BorderLayout());

        meuPainel = new JPanel();
        meuPainel.setLayout(new FlowLayout());
        this.add(meuPainel, BorderLayout.CENTER);

        meuLabel = new JLabel("Meu Label");
        meuLabel.setSize(30,70);
        meuPainel.add(meuLabel);
    }
    public static void main (String args[])
    {
        MeuPrimeiroJFrame frame = new MeuPrimeiroJFrame();
    }
}
```



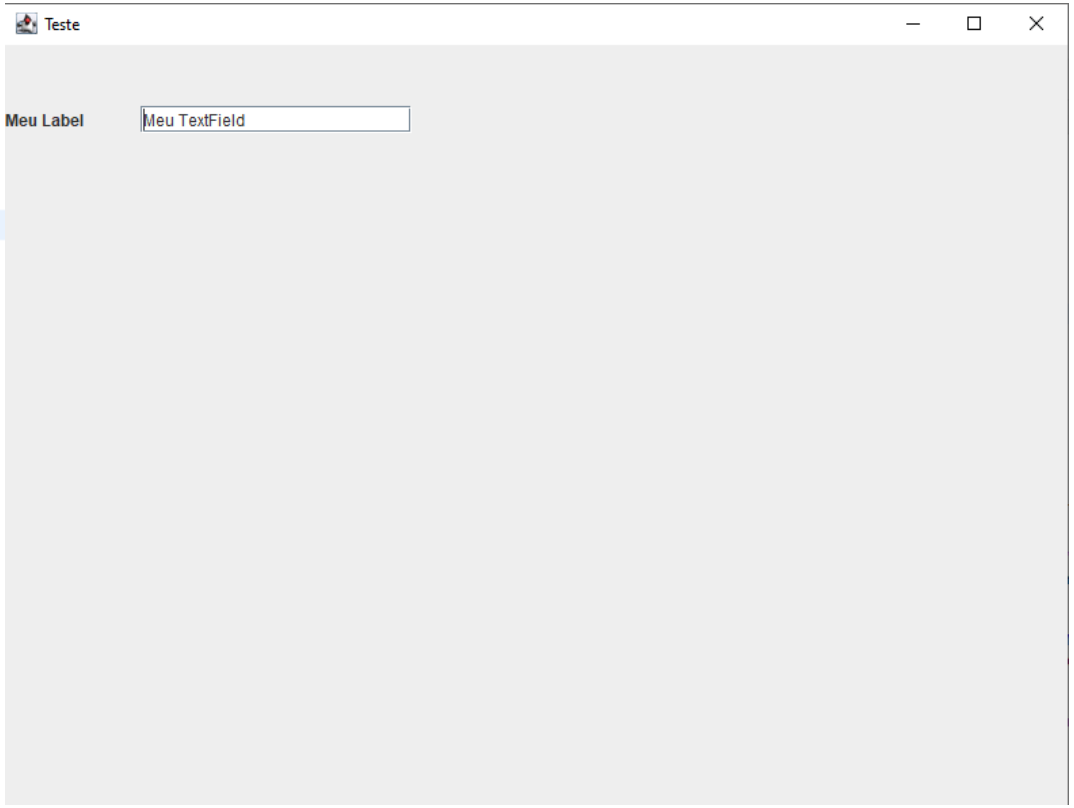
JTextField

JTextField representa um campo para preenchimento de textos curtos de única linha. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto padrão ao campo (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***getText()*** que retorna o texto inserido no campo.
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;

JTextField


```
import javax.swing.*;
import java.awt.*;

public class Tela extends JFrame
{
    private JPanel jp_principal;

    private JLabel jl_meuLabel;
    private JTextField jtf_meuTextField;

    public Tela()
    {
        this.setTitle("Teste");
        this.setSize(800,600);
        this.setLayout(new BorderLayout());

        jp_principal = new JPanel();
        jp_principal.setLayout(new FlowLayout(0,0,30));


            jl_meuLabel = new JLabel("Meu Label");
            jl_meuLabel.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jl_meuLabel);

            jtf_meuTextField = new JTextField("Meu TextField");
            jtf_meuTextField.setPreferredSize(new Dimension(200,20));
            jp_principal.add(jtf_meuTextField);

        this.getContentPane().add("Center",jp_principal);
        this.setVisible(true);
    }
}
```



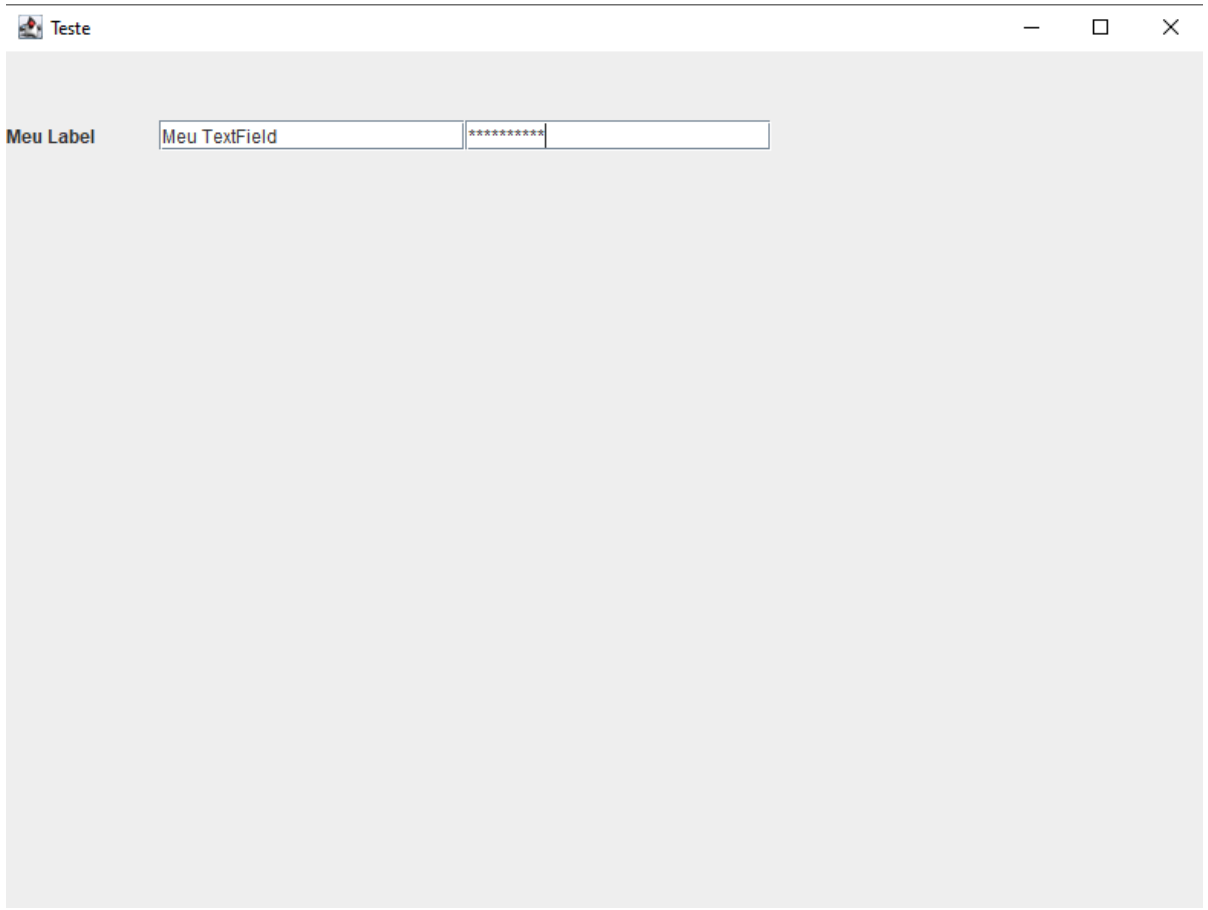
JPasswordField

JPasswordField representa um campo de texto protegido, subclasse de JTextField. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto padrão ao campo (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***getText()*** que retorna o texto inserido no campo.
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;
- ***setEchoChar(char)*** que define o caractere que aparece ao digitar um texto.

JPasswordField


```
package Pck_JFrame;

import javax.swing.*;
import java.awt.*;

public class Tela extends JFrame
{
    private JPanel jp_principal;

    private JLabel jl_meuLabel;
    private JTextField jtf_meuTextField;
    private JPasswordField jpf_meuPass;

    public Tela()
    {
        this.setTitle("Teste");
        this.setSize(800,600);
        this.setLayout(new BorderLayout());

        jp_principal = new JPanel();
        jp_principal.setLayout(new FlowLayout(0,0,30));


            jl_meuLabel = new JLabel("Meu Label");
            jl_meuLabel.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jl_meuLabel);

            jtf_meuTextField = new JTextField("Meu TextField");
            jtf_meuTextField.setPreferredSize(new Dimension(200,20));
            jp_principal.add(jtf_meuTextField);

            jpf_meuPass = new JPasswordField();
            jpf_meuPass.setEchoChar('*');
            jpf_meuPass.setPreferredSize(new Dimension(200,20));
            jp_principal.add(jpf_meuPass);

        this.getContentPane().add("Center",jp_principal);
        this.setVisible(true);
    }
}
```



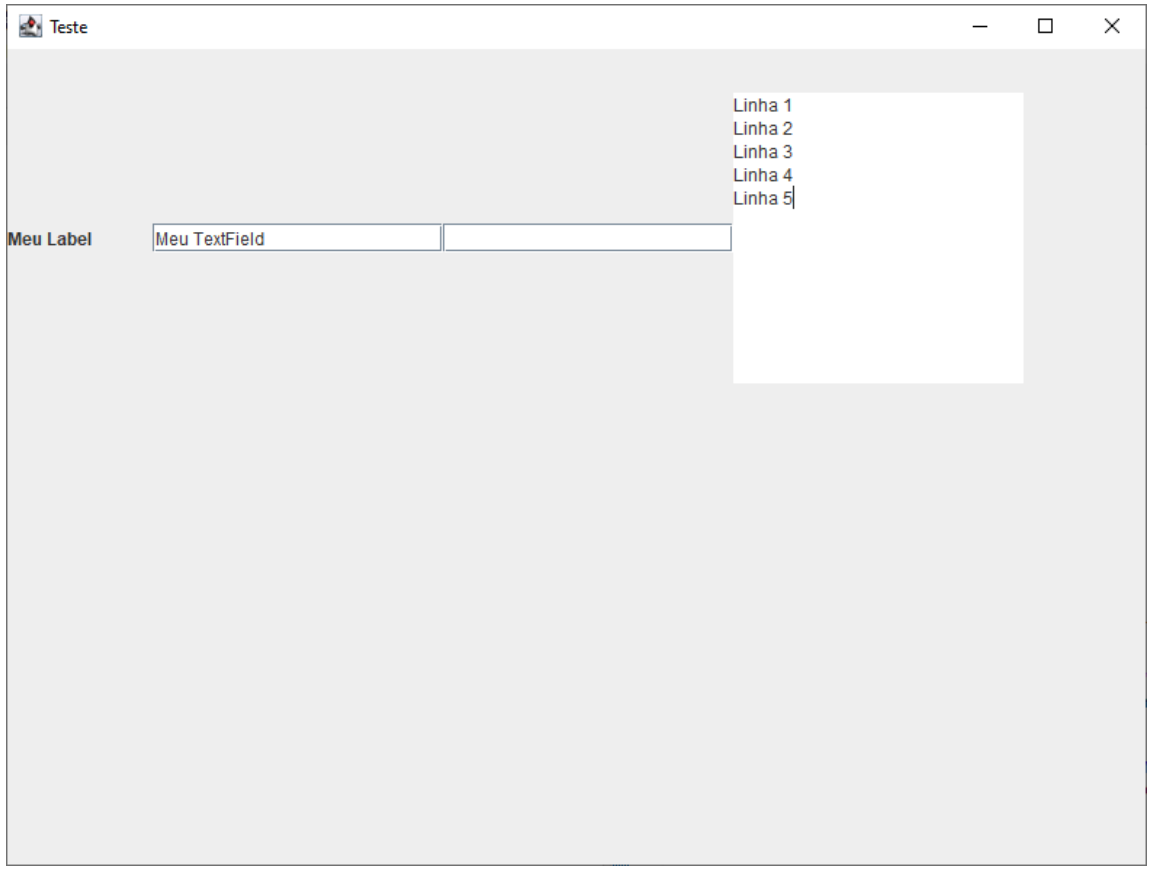
JTextArea

JTextArea representa um campo que pode ser preenchido com várias linhas de texto. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto padrão ao campo (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***getText()*** que retorna o texto inserido no campo.
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;
- ***getSelectedText()*** que retorna o texto selecionado pelo usuário;
- ***insert(String, int)*** que insere um texto na posição especificada.

JTextArea


```
package Pck_JFrame;

import javax.swing.*;
import java.awt.*;

public class Tela extends JFrame
{
    private JPanel jp_principal;

    private JLabel jl_meuLabel;
    private JTextField jtf_meuTextField;
    private JPasswordField jpf_meuPass;
    private JTextArea jta_meuTextArea;

    public Tela()
    {
        this.setTitle("Teste");
        this.setSize(800,600);
        this.setLayout(new BorderLayout());

        jp_principal = new JPanel();
        jp_principal.setLayout(new FlowLayout(0,0,30));


            jl_meuLabel = new JLabel("Meu Label");
            jl_meuLabel.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jl_meuLabel);

            jtf_meuTextField = new JTextField("Meu TextField");
            jtf_meuTextField.setPreferredSize(new Dimension(200,20));
            jp_principal.add(jtf_meuTextField);

            jpf_meuPass = new JPasswordField();
            jpf_meuPass.setEchoChar('*');
            jpf_meuPass.setPreferredSize(new Dimension(200,20));
            jp_principal.add(jpf_meuPass);

            jta_meuTextArea = new JTextArea();
            jta_meuTextArea.setPreferredSize(new Dimension(200,200));
            jp_principal.add(jta_meuTextArea);

        this.getContentPane().add("Center",jp_principal);
        this.setVisible(true);
    }
}
```



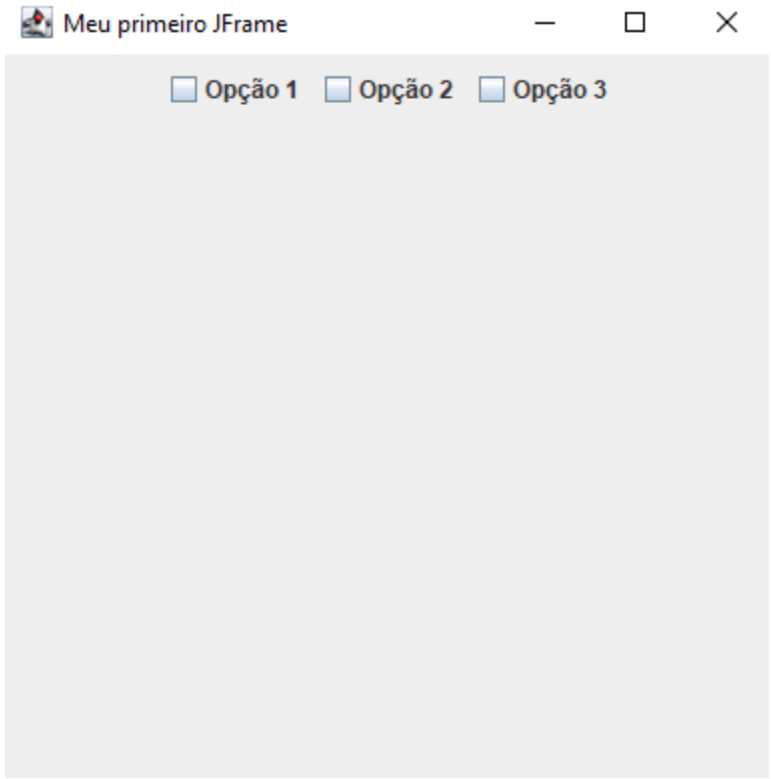
JCheckBox

JCheckBox representa uma caixa de seleção e permite selecionar mais de uma opção. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto padrão ao campo (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;
- ***setSelected(boolean)*** que altera o estado da caixa de seleção;
- ***isSelected()*** que retorna ***true*** se a caixa estiver marcada e *false* se não estiver marcada.

JCheckBox


```java
package Pck_Swing;

import javax.swing.*;
import java.awt.*;

public class MeuPrimeiroJFrame extends JFrame
{
    public JPanel meuPainel;
    public JCheckBox meuCheckBox1;
    public JCheckBox meuCheckBox2;
    public JCheckBox meuCheckBox3;

    public MeuPrimeiroJFrame()
    {
        this.setVisible(true);
        this.setTitle("Meu primeiro JFrame");
        this.setSize(400,400);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setLayout(new BorderLayout());

        meuPainel = new JPanel();
        meuPainel.setLayout(new FlowLayout());
        this.add(meuPainel, BorderLayout.CENTER);

        meuCheckBox1 = new JCheckBox();
        meuCheckBox1.setText("Opção 1");
        meuPainel.add(meuCheckBox1);

        meuCheckBox2 = new JCheckBox();
        meuCheckBox2.setText("Opção 2");
        meuPainel.add(meuCheckBox2);

        meuCheckBox3 = new JCheckBox();
        meuCheckBox3.setText("Opção 3");
        meuPainel.add(meuCheckBox3);
    }
    public static void main (String args[])
    {
        MeuPrimeiroJFrame frame = new MeuPrimeiroJFrame();
    }
}
```



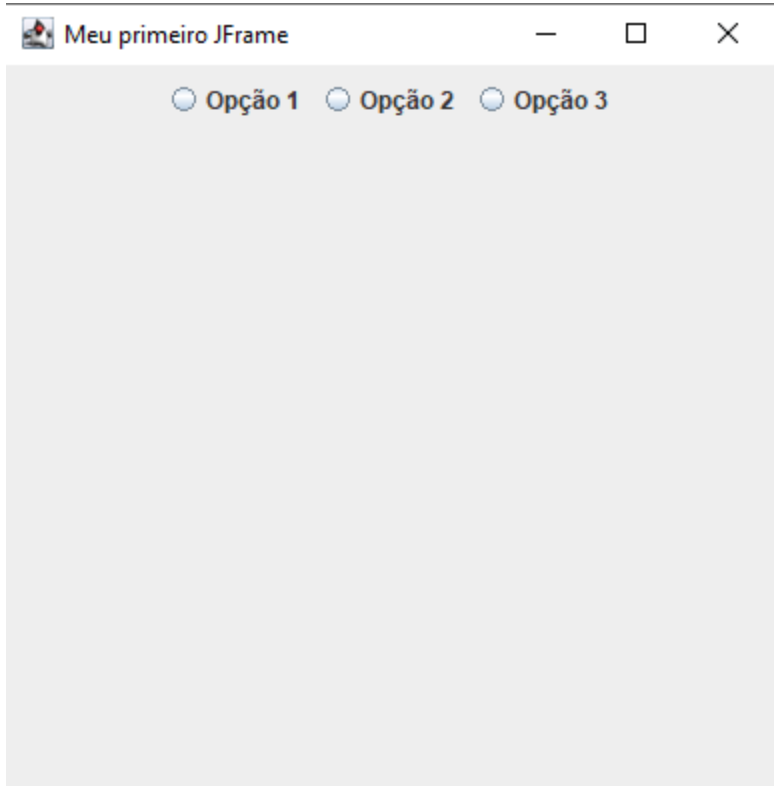
JRadioButton

JRadioButton representa um componente que permite seleciona uma única opção. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto padrão ao campo (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;
- ***setSelected(boolean)*** que altera o estado da caixa de seleção;
- ***isSelected()*** que retorna ***true*** se a caixa estiver marcada e *false* se não estiver marcada.

JRadioButton


```java
package Pck_Swing;

import javax.swing.*;
import java.awt.*;

public class MeuPrimeiroJFrame extends JFrame
{
    public JPanel meuPainel;
    public JRadioButton meuRadioButton1;
    public JRadioButton meuRadioButton2;
    public JRadioButton meuRadioButton3;

    public MeuPrimeiroJFrame()
    {
        this.setVisible(true);
        this.setTitle("Meu primeiro JFrame");
        this.setSize(400,400);
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        this.setLayout(new BorderLayout());

        meuPainel = new JPanel();
        meuPainel.setLayout(new FlowLayout());
        this.add(meuPainel, BorderLayout.CENTER);

        meuRadioButton1 = new JRadioButton();
        meuRadioButton1.setText("Opção 1");
        meuPainel.add(meuRadioButton1);

        meuRadioButton2 = new JRadioButton();
        meuRadioButton2.setText("Opção 2");
        meuPainel.add(meuRadioButton2);

        meuRadioButton3 = new JRadioButton();
        meuRadioButton3.setText("Opção 3");
        meuPainel.add(meuRadioButton3);
    }
    public static void main (String args[])
    {
        MeuPrimeiroJFrame frame = new MeuPrimeiroJFrame();
    }
}
```



JComboBox

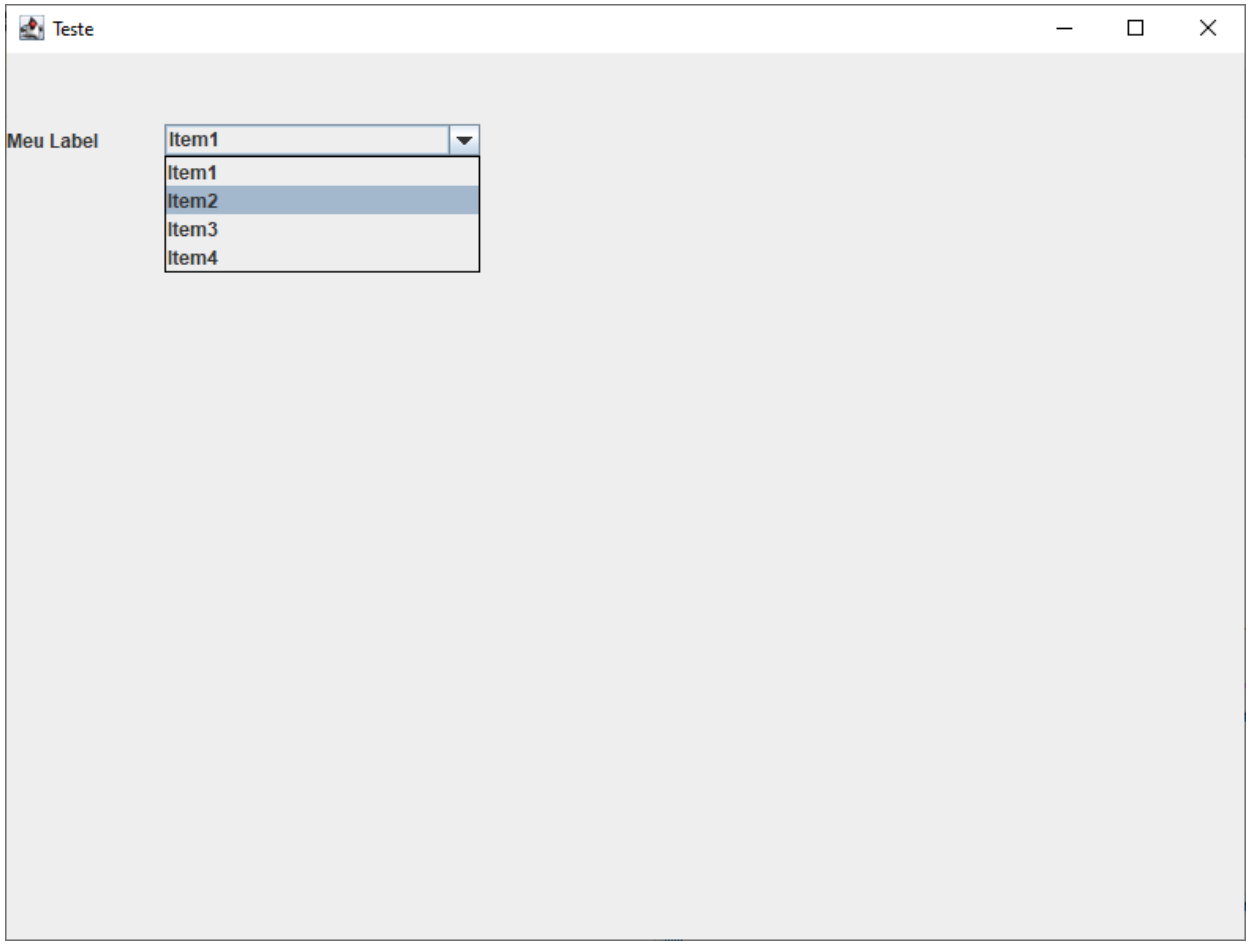
JComboBox representa uma caixa de seleção, da qual pode ser selecionada uma única opção.

Entre os principais métodos temos:

- ***addItem(Object)*** que adiciona um item à lista de opções;
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;
- ***setEditable(boolean)*** que permite ao usuário digitar uma opção;
- ***getSelectedIndex()*** que retorna a posição do item atualmente selecionado;
- ***getSelectedItem()*** que retorna o texto do item atualmente selecionado;
- ***setSelectedIndex(int)*** que seleciona o item da posição especificada;
- ***setSelectedIndex(Object)*** que seleciona o objeto especificado na lista.

JComboBox


```
package Pck_JFrame;

import javax.swing.*;
import java.awt.*;

public class Tela extends JFrame
{
    private JPanel jp_principal;

    private JLabel jl_meuLabel;
    private JComboBox jb_meuCombo;

    public Tela()
    {
        this.setTitle("Teste");
        this.setSize(800,600);
        this.setLayout(new BorderLayout());

        jp_principal = new JPanel();
        jp_principal.setLayout(new FlowLayout(0,0,30));


            jl_meuLabel = new JLabel("Meu Label");
            jl_meuLabel.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jl_meuLabel);

            jb_meuCombo = new JComboBox();
            jb_meuCombo.addItem("Item1");
            jb_meuCombo.addItem("Item2");
            jb_meuCombo.addItem("Item3");
            jb_meuCombo.addItem("Item4");
            jb_meuCombo.setPreferredSize(new Dimension(200,20));
            jp_principal.add(jb_meuCombo);

        this.getContentPane().add("Center",jp_principal);
        this.setVisible(true);
    }
}
```



JList

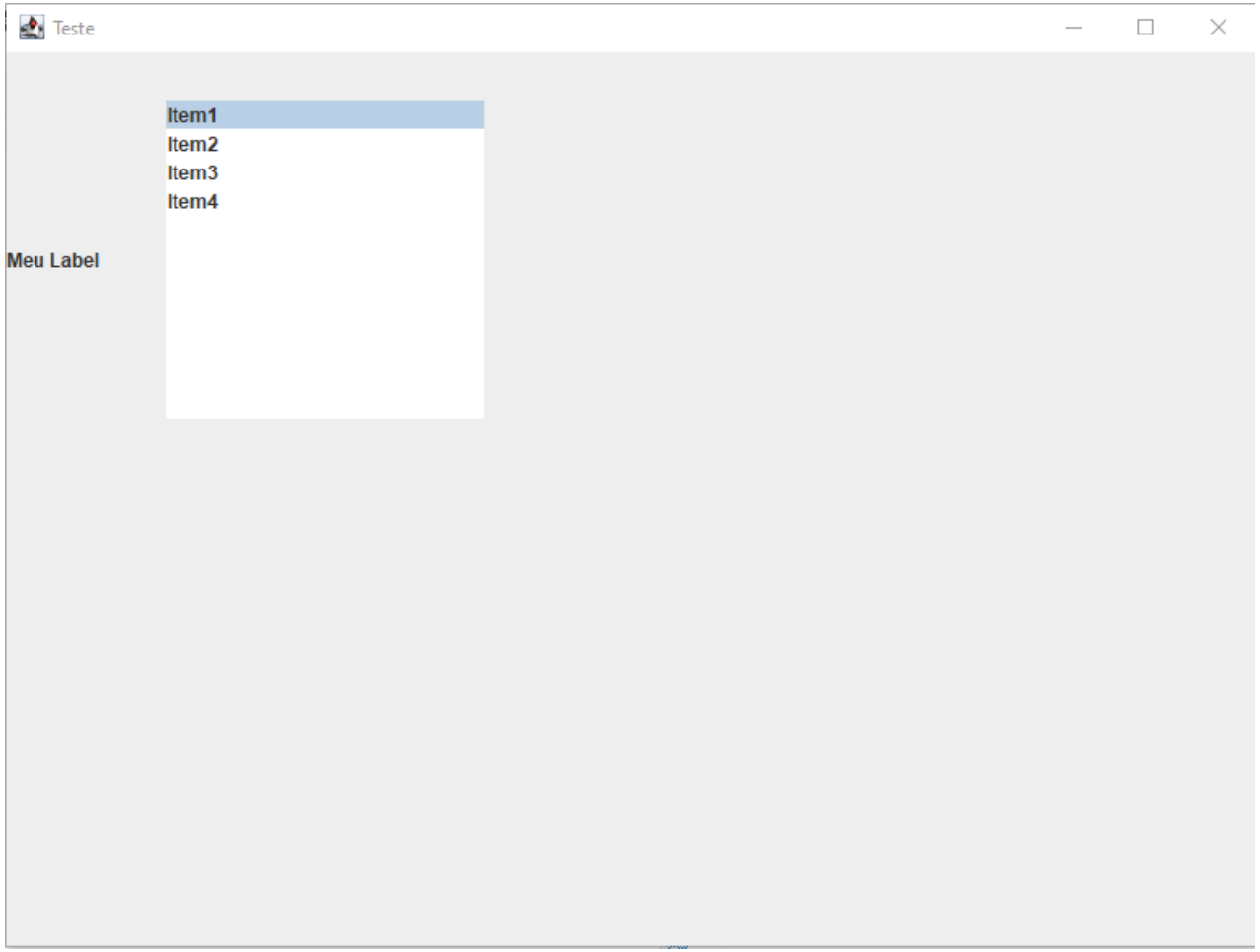
JList representa uma lista de opções que permite a seleção de mais de um item simultaneamente.

Entre os principais métodos temos:

- ***setListData(Object[])*** que preenche ou altera os itens de uma lista;
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels
- ***getSelectedValues()*** que retorna um array de objetos contendo itens selecionados na lista;
- ***getSelectedIndex()*** que retorna a posição do item atualmente selecionado;
- ***getSelectedItem()*** que retorna o texto do item atualmente selecionado;
- ***setSelectedIndex(int)*** que seleciona o item da posição especificada;
- ***setSelectedIndex(Object)*** que seleciona o objeto especificado na lista.

JList


```
package Pck_JFrame;

import javax.swing.*;
import java.awt.*;
import java.util.ArrayList;

public class Tela extends JFrame
{
    private JPanel jp_principal;

    private JLabel jl_meuLabel;
    private JList jli_minhaLista;
    DefaultListModel model;

    public Tela()
    {
        this.setTitle("Teste");
        this.setSize(800,600);
        this.setLayout(new BorderLayout());

        jp_principal = new JPanel();
        jp_principal.setLayout(new FlowLayout(0,0,30));


            jl_meuLabel = new JLabel("Meu Label");
            jl_meuLabel.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jl_meuLabel);

            jli_minhaLista = new JList();
                model = new DefaultListModel<String>();
                model.add(0, "Item1");
                model.add(1, "Item2");
                model.add(2, "Item3");
                model.add(3, "Item4");
            jli_minhaLista.setModel(model);
            jli_minhaLista.setPreferredSize(new Dimension(200,200));
            jp_principal.add(jli_minhaLista);

        this.getContentPane().add("Center",jp_principal);
        this.setVisible(true);
    }
}
```



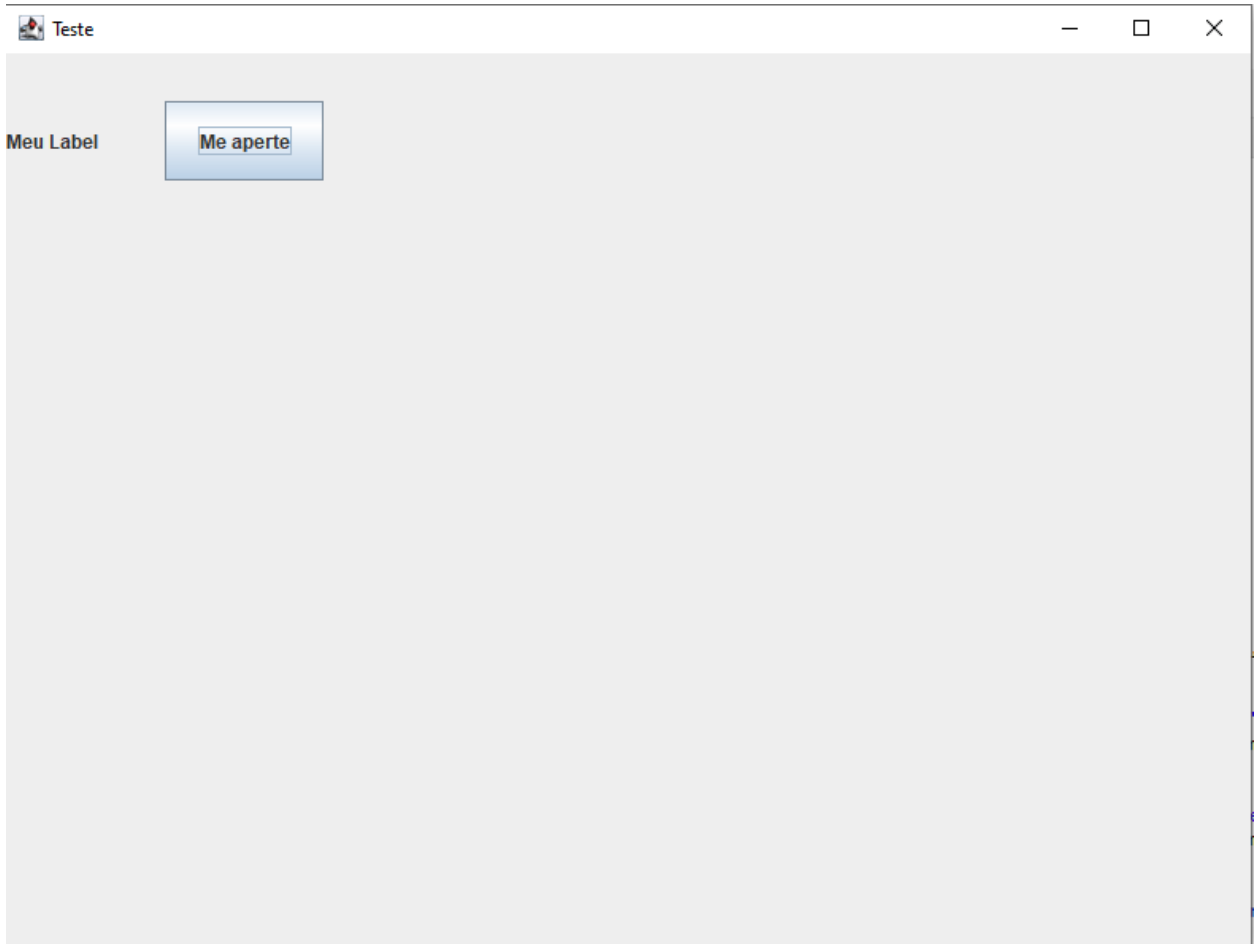
JButton

JButton representa um botão destinado a executar uma ação. Entre os principais métodos temos:

- ***setText(String)*** que insere um texto padrão ao campo (o texto também pode ser inserido diretamente no método construtor do objeto);
- ***setSize(int, int)*** que define as medidas largura e altura da etiqueta em pixels;
- ***setIcon(Icon)*** que altera o ícone do botão.

JButton


```
package Pck_JFrame;

import javax.swing.*;
import java.awt.*;
import java.util.ArrayList;

public class Tela extends JFrame
{
    private JPanel jp_principal;

    private JLabel jl_meuLabel;
    private JButton jb_meuBotao;

    public Tela()
    {
        this.setTitle("Teste");
        this.setSize(800,600);
        this.setLayout(new BorderLayout());

        jp_principal = new JPanel();
        jp_principal.setLayout(new FlowLayout(0,0,30));


            jl_meuLabel = new JLabel("Meu Label");
            jl_meuLabel.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jl_meuLabel);

            jb_meuBotao = new JButton("Me aperte");
            jb_meuBotao.setPreferredSize(new Dimension(100,50));
            jp_principal.add(jb_meuBotao);

        this.getContentPane().add("Center",jp_principal);
        this.setVisible(true);
    }
}
```



OBRIGADO