



Componentes GUI

(Parte 1)

Richarlyson A. D'Emery

site: <https://sites.google.com/site/profricodemery/mpoo>






grupo: http://groups.google.com/group/mpoo_uast

email grupo: mpoo_uast@googlegroups.com

contato: rico_demery@yahoo.com.br

Sumário



-  AWT
-  Swing
-  Containers
-  Frame
-  Layout

O AWT

- Fornece um conjunto de componentes para a construção de interface gráficas (GUI)
- Todos os componentes da GUI são subclasses de **Component** ou de **MenuComponent**
- Possui uma classe **Container** que é uma subclasse abstrata de **Component**. **Container** possui duas subclasses:
 - **Panel**
 - **Window**

O Swing

- Fornece um conjunto mais rico que o AWT
- Não usa a mesma aparência dos componentes
- Baseado na Classe **Jcomponent** que é subclasse de Container

Containers

- Servem como um repositório para elementos gráficos
- Elementos são adicionados com o método **add()**
- Os principais tipos de Containers são **Window** e **Panel**
- Um **Window** pode existir independentemente
- Um **Panel** deve existir no contexto de outro container

Frame - JFrame

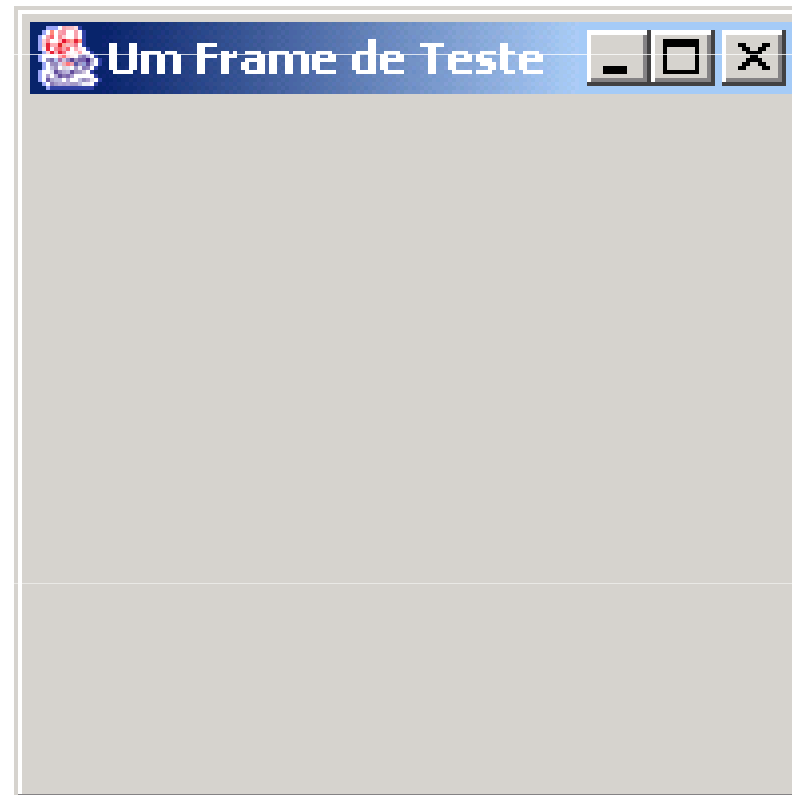
- **Frame** é uma subclasse de Window
- Possui **título** e **bordas** para redimensionamento
- O gerenciador de layout default é o **BorderLayout**
- Um **Jframe** é uma subclasse de Frame

JFrame - Exemplo

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

public class ExemploFrame{
    public static void main(String [] args){
        JFrame f = new JFrame("Um Frame de Teste");
        f.setSize(200,200);
        f.getContentPane().setBackground(Color.green);
        f.setVisible(true); // Igual a f.show();
    }
}
```

JFrame - Exemplo

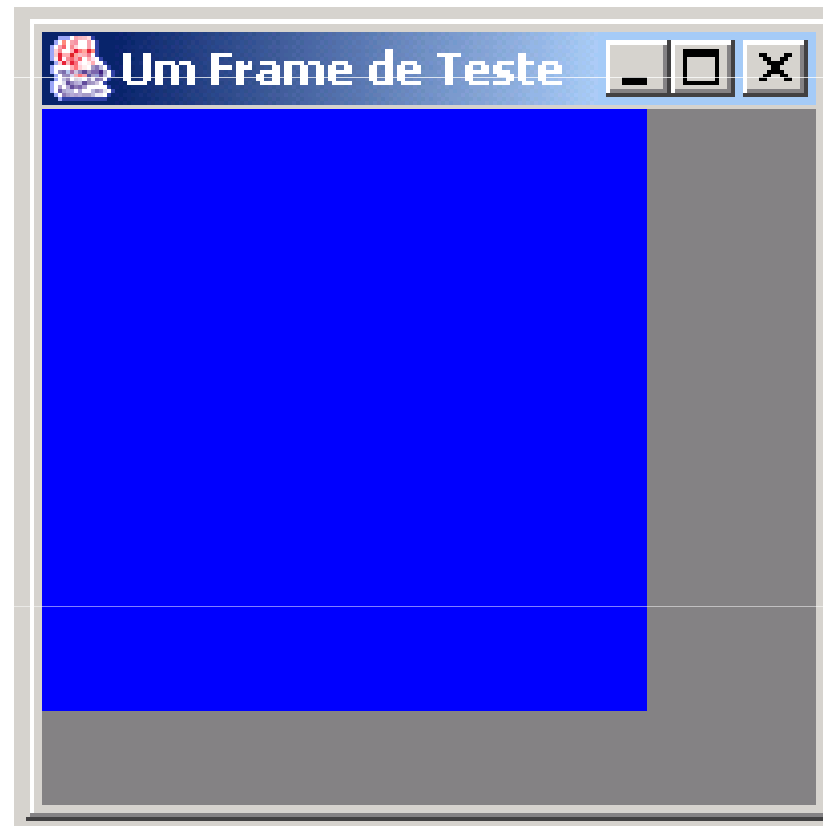


JFrame com JPanel

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

public class ExemploFrameComJPanel{
    public static void main(String [] args){
        JFrame f = new JFrame("Um Frame de Teste");
        JPanel p = new JPanel();
        f.setSize(200,200);
        p.setSize(150,150);
        f.getContentPane().setBackground(Color.gray);
        p. setBackground(Color.blue);
        f.getContentPane().setLayout(null);
        f.getContentPane().add(p);
        f.setVisible(true);
    }
}
```

JFrame com JPanel



Gerenciador de Layout

- FlowLayout
- BorderLayout
- GridLayout
- CardLayout
- GridBagLayout

FlowLayout

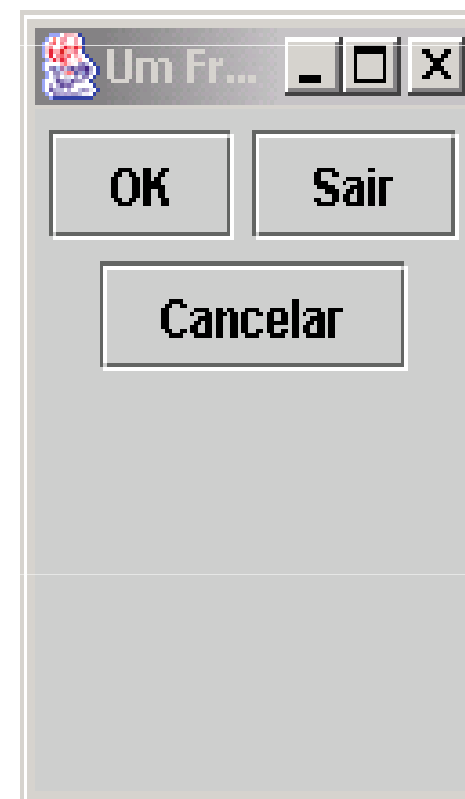
- É o gerenciador default da classe Panel
- Componentes são adicionados da esquerda para a direita
- O alinhamento default é centralizado

FlowLayout - Exemplo

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

public class ExemploFlowLayout{
    public static void main(String [] args){
        JFrame f = new JFrame("Um Frame de Teste");
        JButton botao1 = new JButton("OK");
        JButton botao2 = new JButton("Sair");
        JButton botao3 = new JButton("Cancelar");
        f.getContentPane().setLayout(new FlowLayout());
        f.setSize(200,200);
        f.getContentPane().add(botao1);
        f.getContentPane().add(botao2);
        f.getContentPane().add(botao3);
        f.setVisible(true); // Igual a f.show();
    }
}
```

FlowLayout - Exemplo



BorderLayout

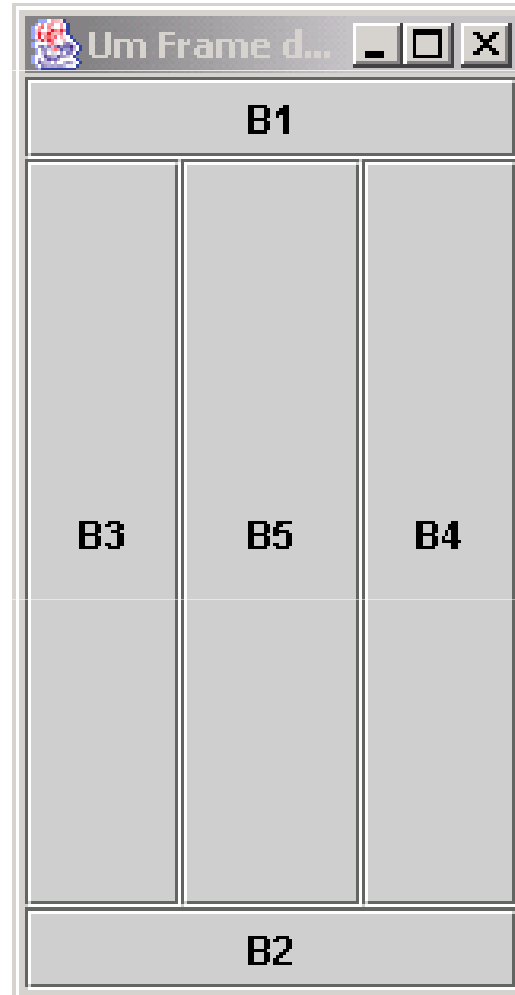
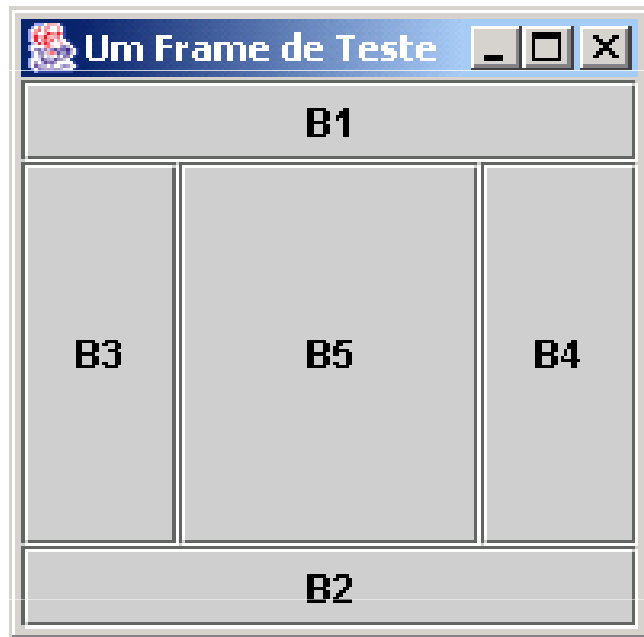
- É o gerenciador default da classe Frame
- É dividido em 5 regiões:
 - North
 - South
 - East
 - West
 - Center
- Componentes são adicionados em uma dessas regiões

BorderLayout - Exemplo

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

public class ExemploBorderLayout{
    public static void main(String [] args){
        JFrame f = new JFrame("Um Frame de Teste");
        JButton botao1 = new JButton("B1");
        JButton botao2 = new JButton("B2");
        JButton botao3 = new JButton("B3");
        JButton botao4 = new JButton("B4");
        JButton botao5 = new JButton("B5");
        f.setSize(200,200);
        f.getContentPane().add(botao1, BorderLayout.NORTH);
        f.getContentPane().add(botao2, BorderLayout.SOUTH);
        f.getContentPane().add(botao3, BorderLayout.WEST);
        f.getContentPane().add(botao4, BorderLayout.EAST);
        f.getContentPane().add(botao5, BorderLayout.CENTER);
        f.setVisible(true);
    }
}
```


BorderLayout - Exemplo



GridLayout

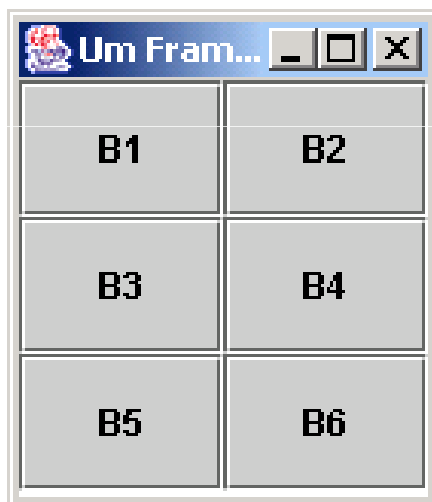
- É dividido de tamanhos iguais
- Componentes são adicionados da esquerda para a direita, de cima para baixo
- O construtor especifica as linhas e as colunas
 - `New GridLayout(5,2);`

GridLayout - Exemplo

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

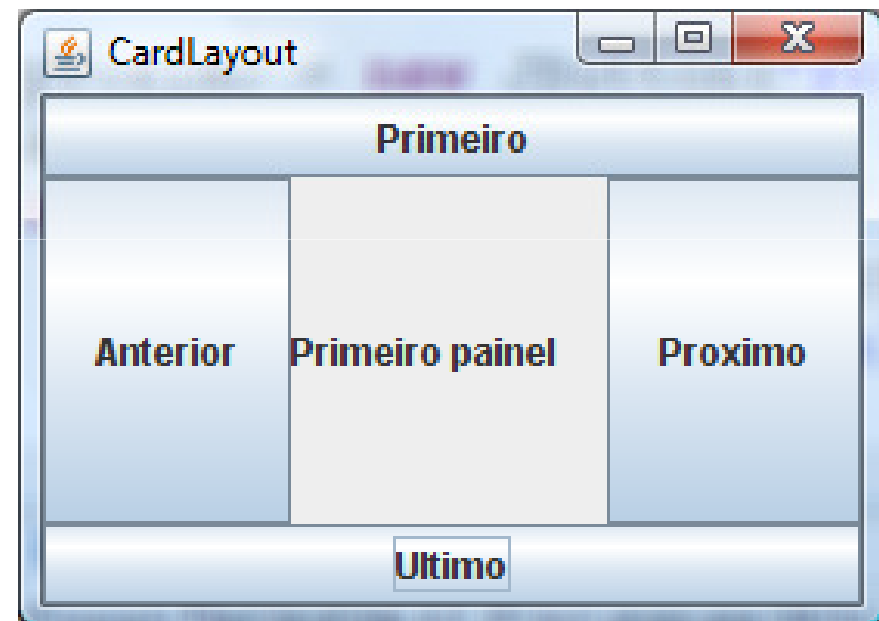
public class ExemploGridLayout{
    public static void main(String [] args){
        JFrame f = new JFrame("GridLayout");
        JButton botao1 = new JButton("B1");
        JButton botao2 = new JButton("B2");
        JButton botao3 = new JButton("B3");
        JButton botao4 = new JButton("B4");
        JButton botao5 = new JButton("B5");
        JButton botao6 = new JButton("B6");
        f.setSize(200,200);
        f.setLayout(new GridLayout(3,2));
        f.getContentPane().add(botao1);
        f.getContentPane().add(botao2);
        f.getContentPane().add(botao3);
        f.getContentPane().add(botao4);
        f.getContentPane().add(botao5);
        f.getContentPane().add(botao6);
        f.setVisible(true);
    }
}
```

GridLayout - Exemplo



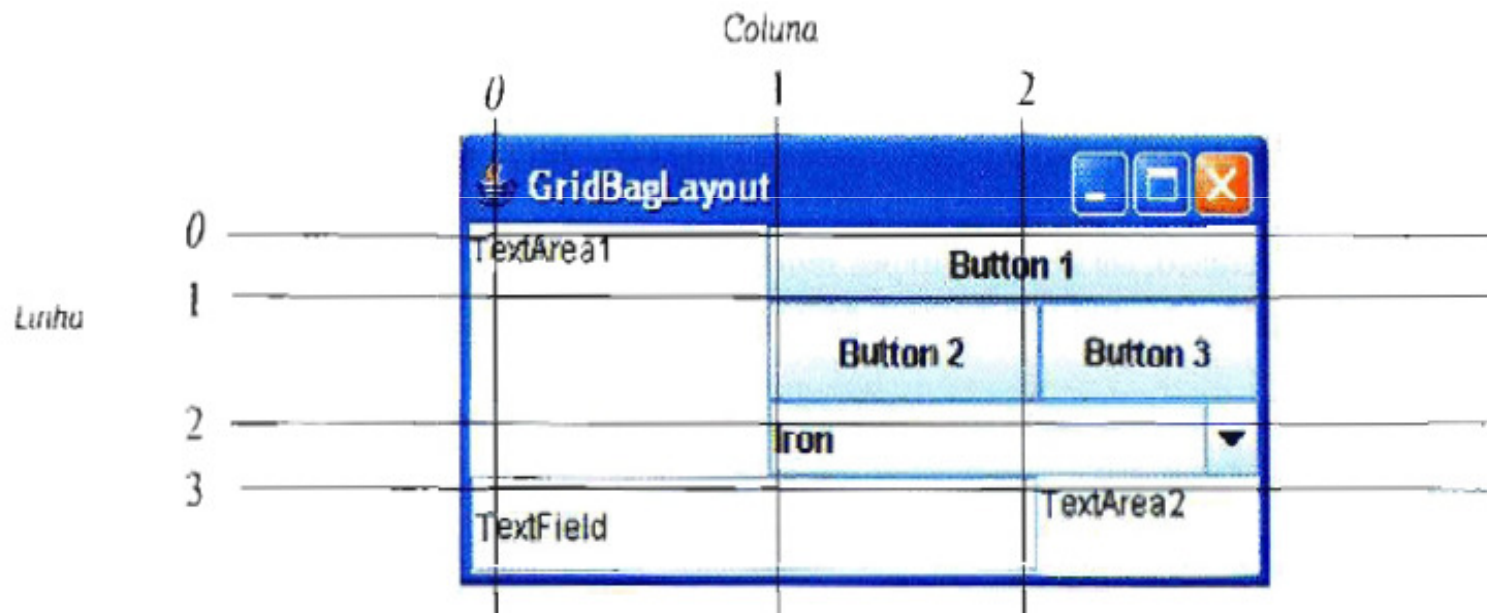
CardLayout

- O **CardLayout** permite que Panels sejam apresentados de forma alternada
 - Exemplo – na Aula_16 (após Eventos)
- Componentes ficam empilhados como um baralho
- Seus métodos:
 - first()
 - next()
 - previous()
 - last



GridBagLayout

- O **GridBagLayout** permite uma elaboração mais sofisticada da interface gráfica
- Um componente pode se estender por mais de uma célula
 - Button1, por exemplo, ocupa as colunas 1 e 2



GridBagLayout

Campos	Descrição
anchor	Especifica a posição relativa (NORTH, NORTHEAST, EAST, SOUTHEAST, SOUTH, SOUTHWEST, WEST, NORTHWEST, CENTER) do componente em uma área que ele não preenche.
fill	Redimensiona o componente na direção especificada (NONE, HORIZONTAL, VERTICAL, BOTH) quando a área de exibição for maior que o componente.
gridx	A coluna em que o componente será colocado.
gridy	A linha em que o componente será colocado.
gridwidth	O número de colunas que o componente ocupa.
gridheight	O número de linhas que o componente ocupa.
weightx	A quantidade de espaço extra a alocar horizontalmente. O componente na grade pode tomar-se mais largo se houver espaço extra disponível
weighty	A quantidade de espaço extra a alocar verticalmente. O componente na grade pode tornar-se mais alto se houver espaço extra disponível

Componentes do Swing

■ JTextField

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



```
public class ExemploFrameJTextField{  
    public static void main(String [] args){  
        JFrame f = new JFrame("TextField");  
        JTextField texto = new JTextField("Exemplo  
de TextField",30);  
        f.setSize(200,75);  
        f.add(texto);  
        f.setBackground(Color.green);  
        f.setVisible(true);  
    }  
}
```


Componentes do Swing

■ JComboBox

```
import javax.swing.*;
```

```
public class ExemploFrameComboBox{  
    public static void main(String [] args){  
        JFrame f = new JFrame("JComboBox");  
        JComboBox combo = new JComboBox();  
  
        combo.addItem("Primeiro");  
        combo.addItem("Segundo");  
        combo.addItem("Terceiro");  
  
        f.setSize(200, 70);  
        f.getContentPane().add(combo);  
        f.setVisible(true);  
    }  
}
```

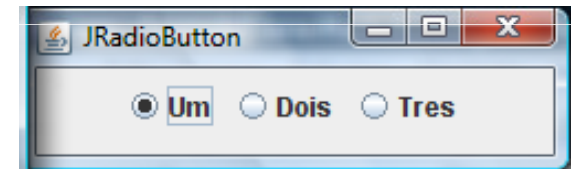


Componentes do Swing

■ JRadioButton

```
import javax.swing.ButtonGroup;
import javax.swing.JFrame;
import javax.swing.JRadioButton;
import java.awt.FlowLayout;

public class ExemploFrameJRadioButton{
    public static void main(String [] args){
        JFrame f = new JFrame ("JRadioButton");
        ButtonGroup bg = new ButtonGroup();
        JRadioButton um = new JRadioButton("Um",true);
        JRadioButton dois = new JRadioButton("Dois",false);
        JRadioButton tres = new JRadioButton("Tres",false)
//Criação do grupo - elementos sob mesma condição de seleção
        bg.add(um);
        bg.add(dois);
        bg.add(tres);
        f.getContentPane().add(um);
        f.getContentPane().add(dois);
        f.getContentPane().add(tres);
        f.setSize(250,75);
        f.setLayout(new FlowLayout());
        f.setVisible(true);
    }
}
```



Componentes do Swing com GridBagLayout



```
//Demonstrando GridBagLayout.
import java.awt.GridBagLayout;
import java.awt.GridBagConstraints;
import java.awt.Component;
import javax.swing.JFrame;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.JButton;
import javax.swing.JComboBox;

public class ExemploGridBagLayout extends JFrame{
    private GridBagLayout layout;//layout do frame
    private GridBagConstraints retricos; // restrições do layout

    //configura a GUI
    public ExemploGridBagLayout() {
        super ( "GridBagLayout");

        layout = new GridBagLayout();
        setLayout(layout); // configura o layout de frame
        retricos = new GridBagConstraints(); //restrições

        //Componentes GUI
        JTextArea textArea1 = new JTextArea("TextAreal", 5, 10);
        JTextArea textArea2 = new JTextArea("TextArea2", 2, 2);
        String opcoesComboBox[] = {"opcao1", "opcao2", "opcao3"};
        JComboBox comboBox = new JComboBox(opcoesComboBox);
        JTextField textField = new JTextField();
        JButton botao1 = new JButton ("botao1");
        JButton botao2 = new JButton ("botao2");
        JButton botao3 = new JButton ("botao3");

        //padrão para todos os componentes: anchor = CENTER
        //padrão para o textArea1: weightx=0 e weighty=0
        retricos.fill = GridBagConstraints.BOTH;
        addComponent(textArea1, 0, 0, 1, 3);

        //padrão para botão1: weightx=0 e weighty=0
        retricos.fill = GridBagConstraints.HORIZONTAL; //só pode
        crescer na horizontal
        addComponent(botao1, 0, 1, 2, 1);
```

```
//padrão para o comboBox: weightx=0 e weighty=0, fill é HORIZ.
addComponent(comboBox, 2, 1, 2, 1 );

//botão2
retricos.weightx = 1000; // pode crescer na largura
retricos.weighty = 1;    // pode crescer na altura
retricos.fill = GridBagConstraints.BOTH;
addComponent(botao2, 1, 1, 1, 1 );

//botão três NÃO cresce na largura pois botão2 tem weightx=1000
retricos.weightx = 0;
retricos.weighty = 0;
addComponent(botao3, 1, 2, 1, 1);

//weightx=0, weighty=0 (NÃO AUMENTA) e preenchimento BOTH para
textField e textArea2
addComponent(textField, 3, 0, 2, 1 );
addComponent(textArea2, 3, 2, 1, 1);

setSize(300,150);
setVisible(true);
setLocationRelativeTo(null);
} //fim do construtor ExemploGridBagLayout

// método para adicionar os GUIs com restrições configuradas
private void addComponent(Component gui, int linha, int coluna, int
width, int height){
    retricos.gridx = coluna;    //configura gridx
    retricos.gridy = linha;     // configura gridy
    retricos.gridwidth = width; //configura gridwidth
    retricos.gridheight = height; //configura gridheight
    layout.setConstraints(gui, retricos); //configura constraints
    add(gui); //adiciona componente ao frame
} //fim do método addComponent

public static void main(String [] args){
    ExemploGridBagLayout aplicacao = new ExemploGridBagLayout();
    aplicacao.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
}
```



FIM

Prof. Richarlyson D'Emery

site: <https://sites.google.com/site/profricodemery/mpoo>

grupo: http://groups.google.com/group/mpoo_uast

email grupo: mpoo_uast@googlegroups.com

contato: rico_demery@yahoo.com.br