Programação Orientada à Objetos (POO)

CIÊNCIA DA COMPUTAÇÃO

Prof. Dr. João Paulo Aramuni

UNIVERSIDADE FUMEC

Sumário

* **Interface Gráfica com Usuário**
  * Conceitos Básicos de Interface com Usuário
  * Introdução ao SWING
  * Componentes

Interface Gráfica com Usuário

* **Conceitos Básicos de Interface com Usuário**
  * Geralmente quando se está começando a programar, o desenvolvedor começa a fazer códigos que são retornados no console em formato de texto, pois muitos são códigos de aprendizagem.

  * Mas quando é necessário desenvolver sistemas que precisam de alguma interação mais aprimorada com o usuário, utiliza-se as interfaces gráficas.

Interface Gráfica com Usuário

* **Conceitos Básicos de Interface com Usuário**
  * A **I**nterface **G**ráfica com **U**suário (Graphical User Interface – GUI) é formada através de componentes GUI.
  * Esses componentes são objetos que fazem a interação com usuário por teclado, mouse ou outros dispositivos que venham a servir para entrada de dados.

Interface Gráfica com Usuário

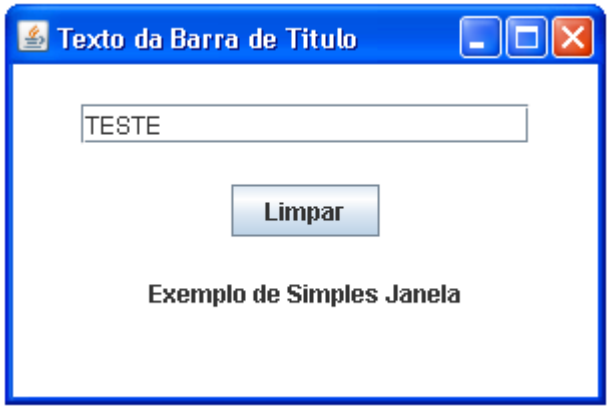
- **Conceitos Básicos de Interface com Usuário**
  - Ao projetar os formulários da aplicação, o desenvolvedor deve considerar:
    - Conteúdo e Estética
      - Os dados que estão no formulário atendem as necessidades dos usuários.
      - Distribuição, Agrupamento, Dimensionamento, Fonte, Cor, Espaçamento e Alinhamento.
    - Facilidade Operacional (produtividade e confiabilidade)
      - Solicitação de serviços do aplicativo (Menu, Barra de Ferramentas, Botões, Atalhos...)
      - Preenchimento dos dados (Validação)
      - Interpretação de resultados (Formatação)

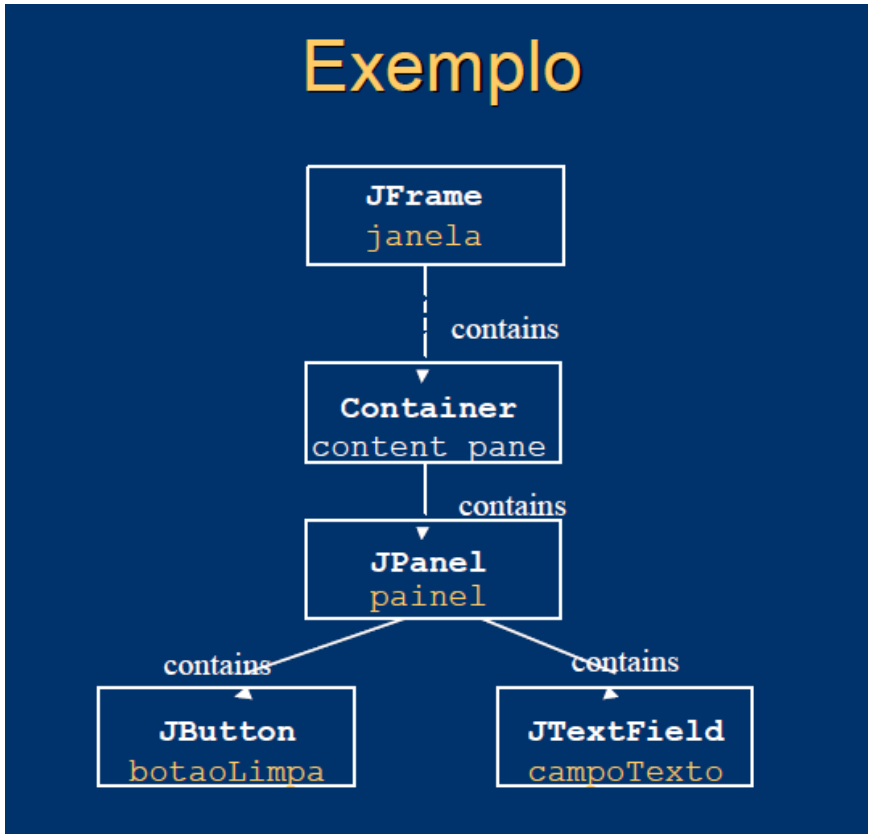
Interface Gráfica com Usuário

- **Conceitos Básicos de Interface com Usuário**
  - Processo Básico: OO + Eventos
    - 1) Instanciar os componentes de interface
      - Por exemplo, janelas, botões, campos de textos, etc.
    - 2) Adicionar os componentes em containers.
      - Por exemplo, como os componentes podem ser agrupados e qual o layout de diagramação.
    - 3) Estabelecer o tratamento de eventos de interface
      - Por exemplo, o deve ocorrer quando o usuário clicar em um botão ou como alterar o conteúdo de um componente quando um outro sofre alguma alteração.

Interface Gráfica com Usuário

* **Conceitos Básicos de Interface com Usuário**
  * Exemplo





Interface Gráfica com Usuário

* **Código**

```
public class ExemploJanela extends JFrame {
    //Atributos
    private JPanel painel = new JPanel();
    private JButton jButtonLimpar = new JButton("Limpar");
    private JTextField jTextFieldTexto = new JTextField("TESTE",20);
    private JLabel jLabelMensagem = new JLabel("Exemplo Simples de Janela");
    // Construtor
    public ExemploJanela() {
        this.setTitle("Texto da Barra de Titulo");
        this.setSize(300, 200);
        painel.setLayout(new FlowLayout(FlowLayout.CENTER,40,20));
        painel.setBackground(new Color(255,255,255));
        painel.add(jTextFieldTexto);
        painel.add(jButtonLimpar);
        painel.add(jLabelMensagem);
        this.getContentPane().add(painel);
        this.setLocationRelativeTo(null); // Centralizar janela
        this.setVisible(true); // Exibir janela
    }
    public static void main(String[] args) {
        new ExemploJanela();
    }
}
```

Interface Gráfica com Usuário

* **Introdução ao SWING**
  * Atualmente, o Java suporta, oficialmente, dois tipos de bibliotecas gráficas: **AWT** e **Swing**. A AWT (Abstract Window Toolkit) foi a primeira API para interfaces gráficas a surgir no Java e foi, mais tarde, superada pelo Swing (a partir do Java 1.2), que possui diversos benefícios em relação ao seu antecessor.

Interface Gráfica com Usuário

* **Introdução ao SWING**
  * As bibliotecas gráficas são bastante simples no que diz respeito a conceitos necessários para usá-las. A complexidade no aprendizado de interfaces gráficas em Java reside no tamanho das bibliotecas e no enorme mundo de possibilidades; isso pode assustar, em um primeiro momento.

  * AWT e Swing são bibliotecas gráficas oficiais inclusas em qualquer JRE ou JDK. Além destas, existem algumas outras bibliotecas de terceiros, sendo a mais famosa, o SWT - desenvolvida pela IBM e utilizada no Eclipse e em vários outros produtos.

Interface Gráfica com Usuário

* **Introdução ao SWING**
  * Grande parte da complexidade das classes e métodos do Swing está no fato da API ter sido desenvolvida tendo em mente o máximo de portabilidade possível.

  * Favorece-se, por exemplo, o posicionamento relativo de componentes, em detrimento do uso de posicionamento fixo, que poderia prejudicar usuários com resoluções de tela diferentes da prevista.

Interface Gráfica com Usuário

* **Introdução ao 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 ambientes.

Interface Gráfica com Usuário

* **Componentes**
  * O Swing traz muitos componentes para usarmos: botões, entradas de texto, tabelas, janelas, abas, scroll, árvores de arquivos e muitos outros.

  * Durante o curso, veremos alguns componentes importantes que nos ajudarão a fazer algumas telas.

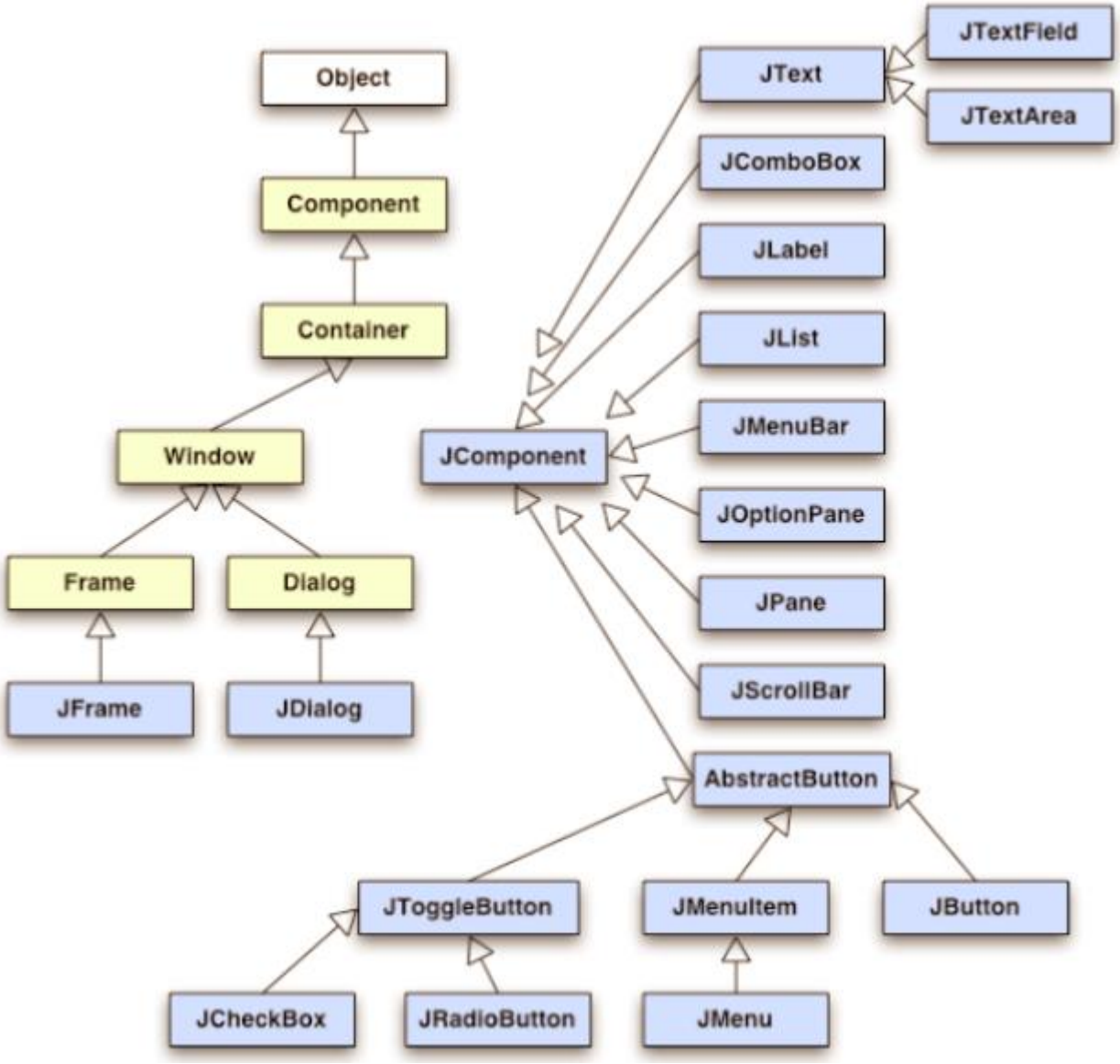
Interface Gráfica com Usuário

* **Componentes**
  * A biblioteca do Swing está no pacote javax.swing (inteira, exceto a parte de acessibilidade, que está em javax.accessibility). Alguns componentes não são do tipo GUI Swing e sim componentes **AWT**.

  * 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

- Diagrama resumido de classes AWT e Swing.



Interface Gráfica com Usuário

* **Componentes**
  * 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.

Interface Gráfica com Usuário

- **Componentes**
  - Abaixo são mostrados alguns dos componentes mais usados:
    - **JLabel** - Exibe texto não editável ou ícones.
    - **JTextField** – Insere dados do teclado e serve também para exibição do texto editável ou não editável.
    - **JButton** – Libera um evento quando o usuário clicar nele com o mouse.
    - **JCheckBox** – Especifica uma opção que pode ser ou não selecionada.
    - **JComboBox** – Fornece uma lista de itens onde possibilita o usuário selecionar um item ou digitar para procurar.
    - **JList** – Lista de itens onde pode ser selecionado vários itens.
    - **JPanel** – É a área onde abriga e organiza os componentes inseridos.

Interface Gráfica com Usuário

* **Componentes**
  * Outros componentes:

Componente	Classe Swing
Janela (Container Principal)	JFrame / JDialog
Painel (Container Intermediário)	JPanel
Painel com Scroll	JScrollPane
Painel Tabulado	JTabbedPane
Barra de Menu	JMenuBar
Barra de Ferramentas	JToolBar
Menu	JMenu
Item de Menu	JMenuItem
Separação de item de Menu	JSeparator
Popup Menu	JPopupMenu

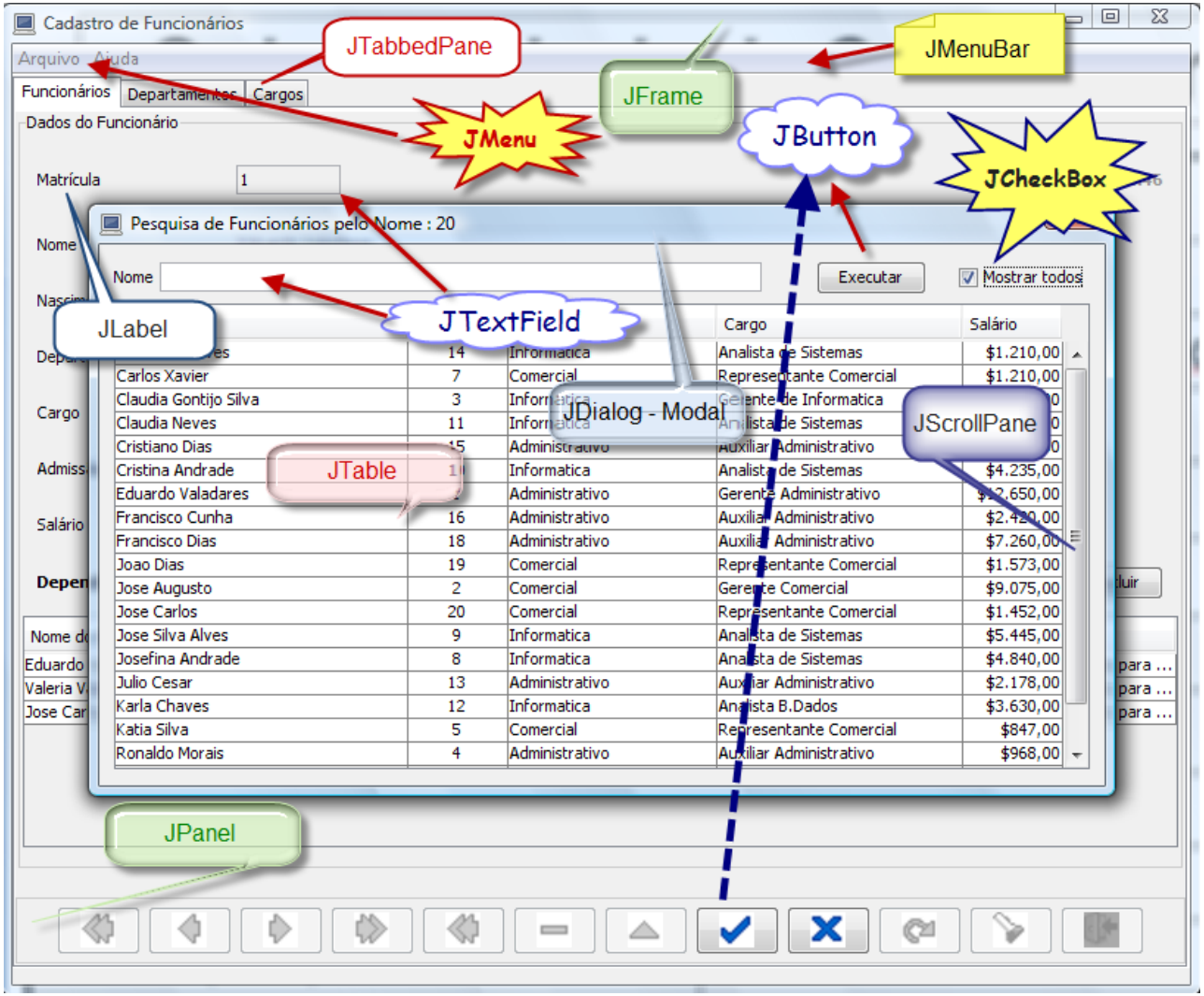
Interface Gráfica com Usuário

* **Componentes**
  * Outros componentes:

Componente	Classe Swing
Rótulo	JLabel
Campo de Texto / Formatado	JTextField / JFormattedTextField
Campo de Senha	JPasswordField
Área de Texto	JTextArea
Botão (c/ ou s/ imagem)	JButton
Caixa de Opção	JCheckBox
Botão de Rádio	JRadioButton , ButtonGroup
Lista	JList
Caixa de Seleção	JComboBox
Tabela (Grid)	JTable
Árvore	JTree

Interface Gráfica com Usuário

* **Exemplo:**

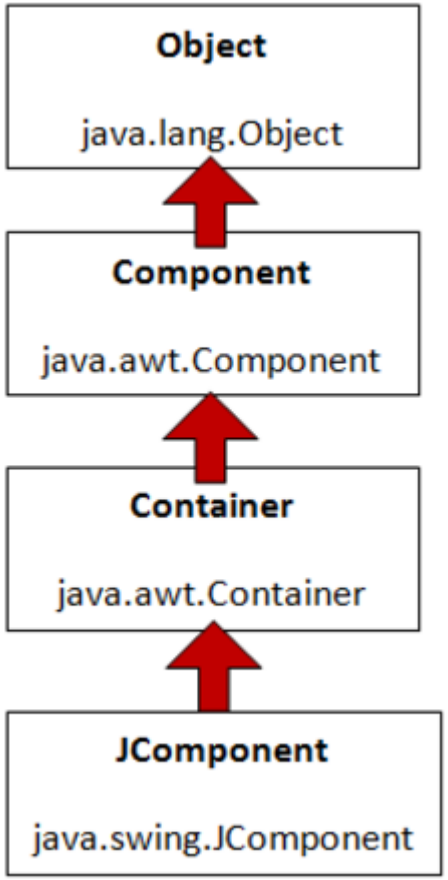


Interface Gráfica com Usuário

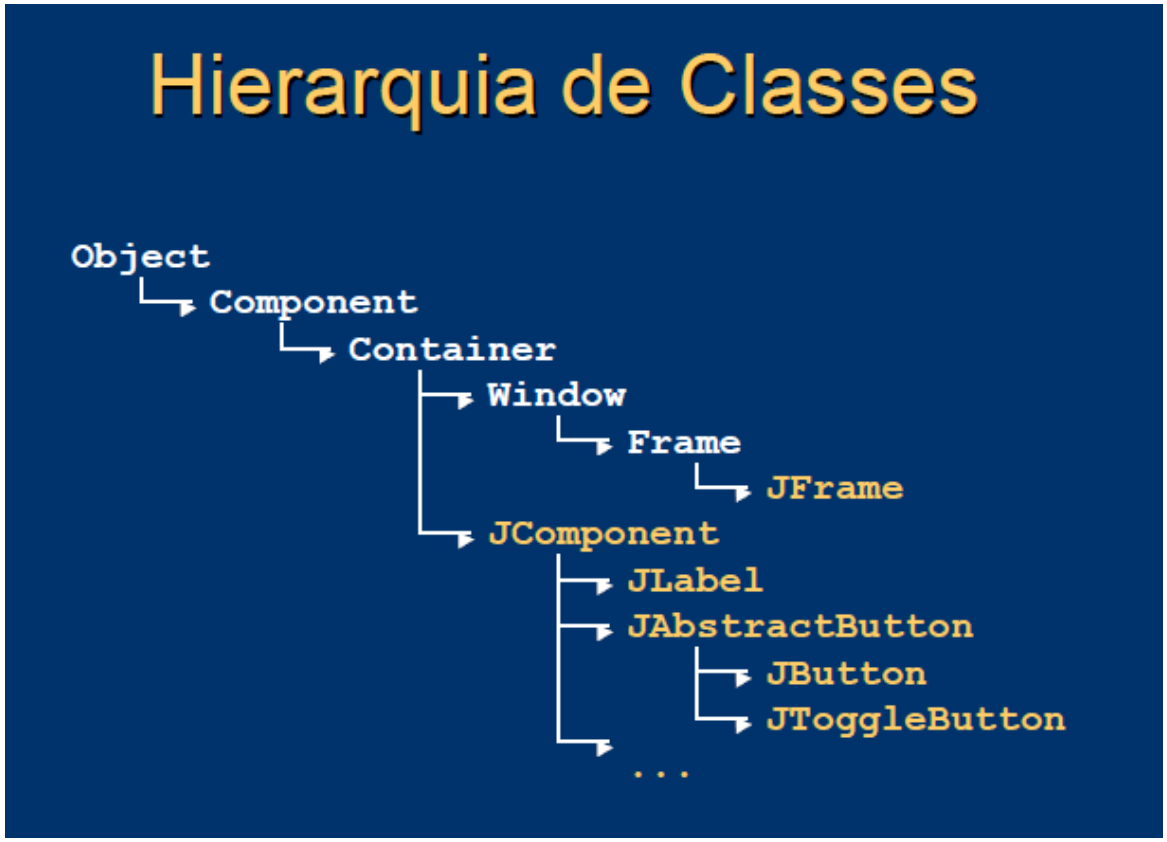
- **Componentes**
  - Principais métodos:
    - JLabel – Nome de campo na tela
      - setText(String)
    - JTextField /JTextArea – Entrada e Saída de Dados
      - getText() / setText(String) / requestFocus()
    - JButton – Botão de comando
      - setText(String) ; isEnabled() ; setEnabled(boolean)
    - jCheckBox – Opções Múltiplas
      - boolean isSelected()
      - setSelected(boolean)
    - ButtonGroup / JRadioButton – Opções Exclusivas
      - objButtonGroup.clearSelecion()
      - objRadioButton.isSelected()

Interface Gráfica com Usuário

* **Componentes**
  * Os componentes possuem classes que definem quais serão seus estados e comportamentos.
  * Veja ao lado como é a hierarquia dessas classes:



Interface Gráfica com Usuário



Interface Gráfica com Usuário

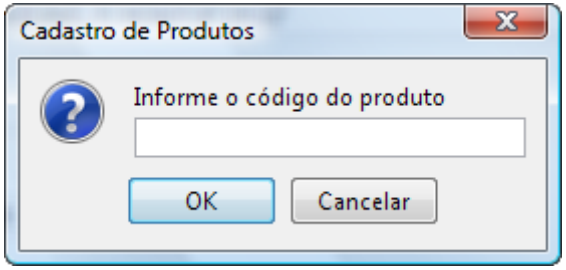
* **Componentes**
  * **Componente** é qualquer elemento de interface.
  * **Container** é um componente que agrega Componentes.
  * Uma **janela** é um *top-level* container: onde os outros componentes são desenhados.
  * Um **painel** é um container intermediário: serve para facilitar o agrupamento de outros componentes.
  * Botões e campos de texto são exemplos de **componentes atômicos**: elementos de interface que não agrupam outros componentes. Mostram para o usuário e/ou obtêm dele alguma informação. A API do Swing oferece muitos componentes atômicos.

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo Caixa de Entrada:

    String codigo = JOptionPane.showInputDialog(null, “Informe o código do produto”, “Cadastro de Produtos” , JOptionPane.QUESTION_MESSAGE);

    if (codigo == null || !codigo.matches(“[0-9]*”)) **return;**
    exibirProdutoCodigo(Integer.parseInt(codigo));



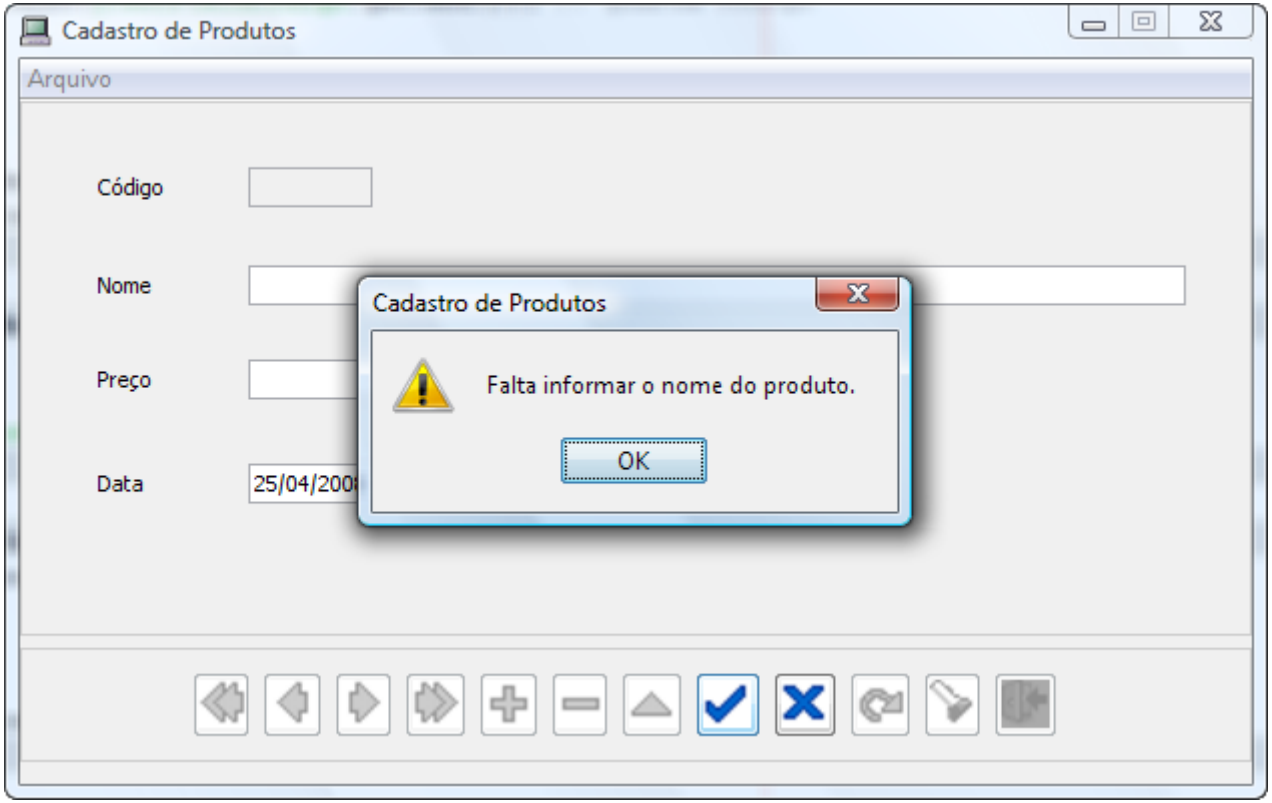
Interface Gráfica com Usuário

* **Componentes**
  * Exemplo Caixa de Mensagem:

```
if ( jFormattedTextFieldNome.getText().trim().equals(“”)) {
   JOptionPane.showMessageDialog(null, “Falta informar o nome do
     produto.”, “Cadastro de Produtos”,
     JOptionPane.WARNING_MESSAGE);
   jFormattedTextFieldNome.requestFocus();
  return false;
}
```

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo Caixa de Mensagem:



Interface Gráfica com Usuário

* **Componentes**
  * Exemplo Caixa de Confirmação:

```
if (JOptionPane.showConfirmDialog(null, “Confirmar a exclusão do
    produto”+ jFormattedTextFieldNome.getText().trim() + “?” , “Cadastro
    de Produtos”, JOptionPane.YES_NO_OPTION) ==
    JOptionPane.NO_OPTION ) {
        return;
}
```

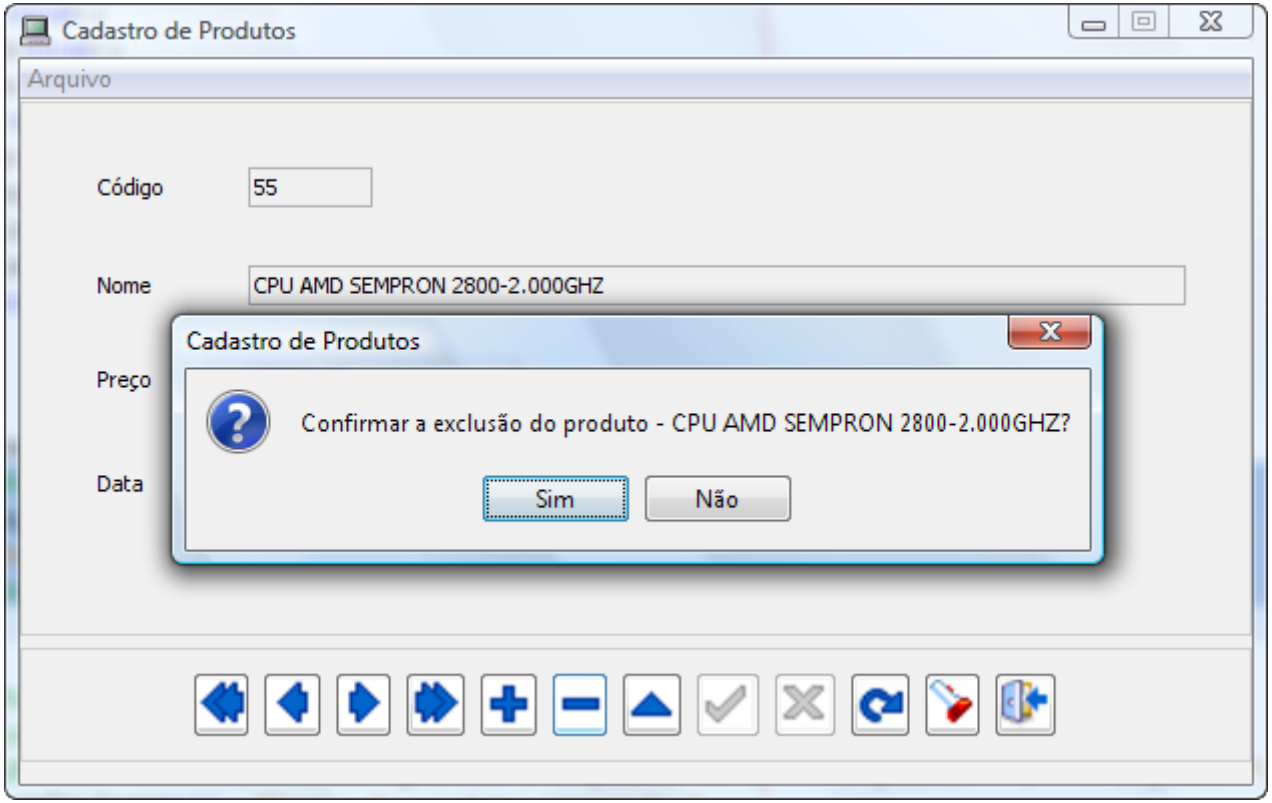
JOptionPane.YES_NO_OPTION

JOptionPane.YES_OPTION

JOptionPane.NO_OPTION

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo Caixa de Confirmação:



Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de Label:

```
/* Cria um label com texto */
JLabel label1 = new JLabel("Label1: Apenas Texto");

/* Cria um label com texto e imagem */
JLabel label2 = new JLabel("Label2: Imagem e texto",
                    new ImageIcon("javalogo.gif"),
                    JLabel.CENTER);
label2.setVerticalTextPosition(JLabel.BOTTOM);
label2.setHorizontalTextPosition(JLabel.CENTER);
```

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de TextField:

```
/* Cria um campo de nome */
JTextField campoNome = new JTextField(10);
JLabel labelNome = new JLabel ("Nome: ");
labelNome.setLabelFor (campoNome);
labelNome.setDisplayedMnemonic('n'); // Alt-n

/* Cria um campo de email */
JTextField campoEmail = new JTextField(10);
JLabel labelEmail = new JLabel ("Email: ");
labelEmail.setLabelFor (campoEmail);
labelEmail.setDisplayedMnemonic('E'); // Alt-e
```

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de Button:

```
/* Cria um botao com texto */
JButton botao1 = new JButton ("Botão Desabilitado");
botao1.setEnabled(false);
botao1.setToolTipText("Exemplo de um botão de texto");
botao1.setMnemonic(KeyEvent.VK_D);  // Alt-D

/* Cria um botao com texto e imagem */
JButton botao2 = new JButton("Botão Habilitado", new
   ImageIcon("javalogo.gif"));
botao2.setToolTipText("Botão de texto e imagem");
botao2.setMnemonic(KeyEvent.VK_H);  // Alt-H
botao2.setPressedIcon(new ImageIcon("javalogo2.gif"));
```

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de Checkbox:

```
JFrame f = new JFrame("Teste");
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JCheckBox ci = new JCheckBox("Inglês",true);
JCheckBox ce = new JCheckBox("Espanhol",true);
JCheckBox cf = new JCheckBox("Francês");
Container cp = f.getContentPane();
cp.setLayout(new FlowLayout());
cp.add(ci);
cp.add(ce);
cp.add(cf);
f.pack();
f.show();
```

Interface Gráfica com Usuário

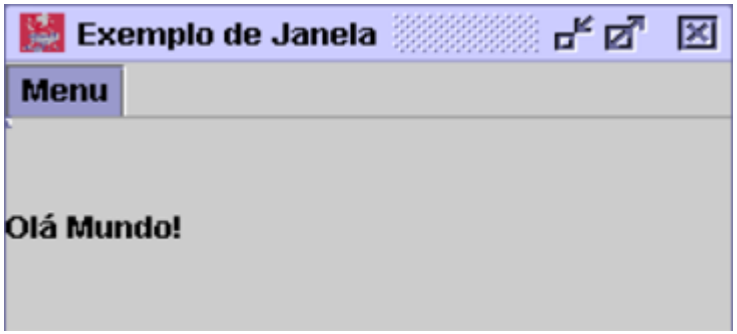
* **Componentes**
  * Exemplo de RadioButton:

```
JFrame f = new JFrame("Teste");
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JRadioButton bm = new JRadioButton("Masculino",true);
JRadioButton bf = new JRadioButton("Feminino");
ButtonGroup bg = new ButtonGroup();
bg.add(bm);
bg.add(bf);
Container cp = f.getContentPane();
cp.setLayout(new FlowLayout());
cp.add(bm);
cp.add(bf);
f.pack();
f.show();
```

Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de JFrame:

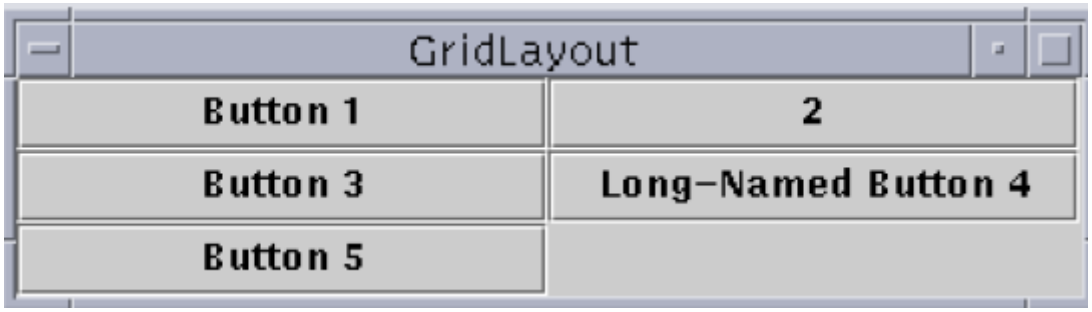
```
JFrame janela = new JFrame("Exemplo de Janela");
janela.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JLabel mensagem = new JLabel("Olá Mundo!");
janela.getContentPane().add(mensagem);
janela.setLocationRelativeTo(null); // centraliza
janela.setIconImage(new
   ImageIcon("javalogo2.gif").getImage());
JMenuBar menuBar = new JMenuBar();
menuBar.add(new JMenu("Menu"));
janela.setJMenuBar (menuBar);
janela.pack();
janela.show();
```



Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de Grid:

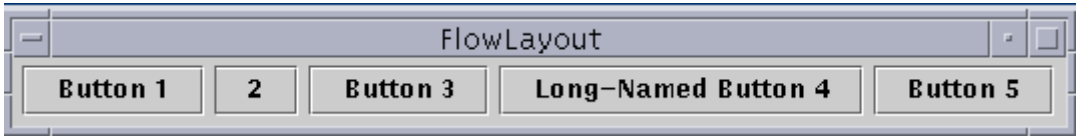
```
Container contentPane = janela.getContentPane();

contentPane.setLayout(new GridLayout(0,2));

contentPane.add(new JButton("Button 1"));
contentPane.add(new JButton("2"));
contentPane.add(new JButton("Button 3"));
contentPane.add(new JButton("Long-Named Button 4"));
contentPane.add(new JButton("Button 5"));
```



Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de Flow:

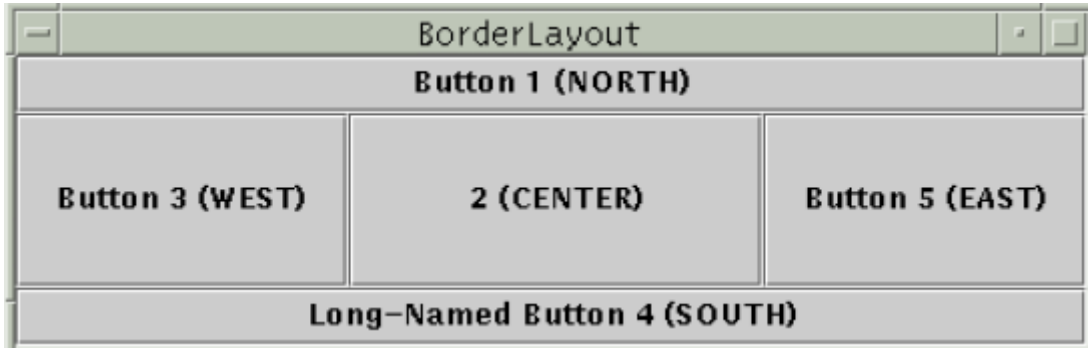
```
Container contentPane = janela.getContentPane();
contentPane.setLayout(new FlowLayout());

contentPane.add(new JButton("Button 1"));
contentPane.add(new JButton("2"));
contentPane.add(new JButton("Button 3"));
contentPane.add(new JButton("Long-Named Button 4"));
contentPane.add(new JButton("Button 5"));
```



Interface Gráfica com Usuário

* **Componentes**
  * Exemplo de Border:

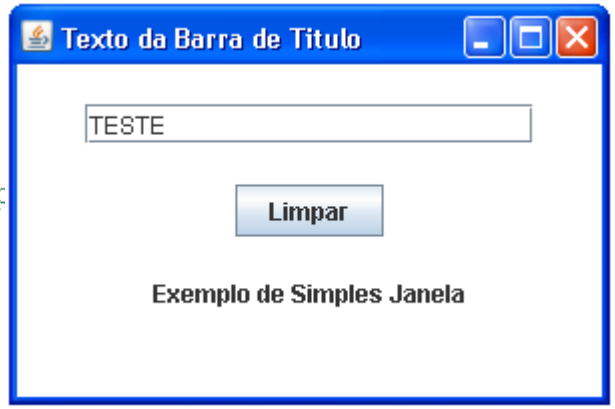
```
Container contentPane = janela.getContentPane();
//contentPane.setLayout(new BorderLayout()); // Desnecessário

contentPane.add(new JButton("Button 1 (NORTH)"),
   BorderLayout.NORTH);
contentPane.add(new JButton("2 (CENTER)"),
   BorderLayout.CENTER);
contentPane.add(new JButton("Button 3 (WEST)"),
   BorderLayout.WEST);
contentPane.add(new JButton("Long-Named Button 4 (SOUTH)"),
   BorderLayout.SOUTH);
contentPane.add(new JButton("Button 5 (EAST)"),
   BorderLayout.EAST);
```



Interface Gráfica com Usuário

* **Orientação por eventos**
  * Um modelo de programação que tornou-se bastante difundido com o uso de interfaces gráficas foi a programação orientada por eventos.
  * Segundo esse modelo, o programa deixa de ter o controle do fluxo de execução, que passa a ser um sistema encarregado de gerenciar a interface.
  * Assim, o programa passa a ser chamado pelo sistema quando algum evento é gerado na interface.

```
public class TesteJanela extends JFrame {
    //Atributos
    private JPanel painel = new JPanel();
    private JButton jButtonLimpar = new JButton("Limpar");
    private JTextField jTextFieldTexto = new JTextField("TESTE",20);
    private JLabel jLabelMensagem = new JLabel("Exemplo de Simples Janela");
    public TesteJanela() {                          // Construtor
        this.setTitle("Exemplo de Interface Gráfica");
        this.setSize(400, 200);
        configurarComponentes();
        this.setLocationRelativeTo(null); // Centralizar janela na tela
        this.setVisible(true); }// Exibir janela
    private void configurarComponentes() {
        jButtonLimpar.setToolTipText("Limpar formulário");
        jButtonLimpar.setCursor(new Cursor(Cursor.HAND_CURSOR));
        jButtonLimpar.setMnemonic('L');
        jButtonLimpar.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
                limparTela();
            }});
        painel.setLayout(new FlowLayout(FlowLayout.CENTER,100,20));
        painel.setBackground(new Color(255,255,255));
        jTextFieldTexto.setFont(new Font("Arial",Font.BOLD,16));
        painel.add(jTextFieldTexto);
        painel.add(jButtonLimpar);
        painel.add(jLabelMensagem);
        this.getContentPane().add(painel);
    }
    private void limparTela() {                 // Limp
        jTextFieldTexto.setText("");
        jTextFieldTexto.requestFocus(); }
    public static void main(String[] args) {
        new TesteJanela();
    }
}
```



Orientação por eventos

- Tabela de ações para eventos (Pacote java.awt.event):

Ação que dispara o evento	**Tipo de listener (que escuta a ação)**
Usuário clica em um botão, pressiona return em uma caixa de texto, ou seleciona um item de menu.	ActionListener
Usuário fecha um frame (janela principal da aplicação).	WindowListener
Usuário pressiona o botão do mouse enquanto o cursor está sobre um componente.	MouseListener
Usuário move o mouse sobre um componente.	MouseMotionListener
Usuário move o wheel sobre um componente.	MouseWhellListener
Componente se torna visível.	ComponentListener
Componente obtém o foco do teclado.	FocusListener
Usuário pressiona alguma tecla.	KeyListener
Item selecionado muda em uma tabela ou lista	ListSelectionListener

Interface Gráfica com Usuário

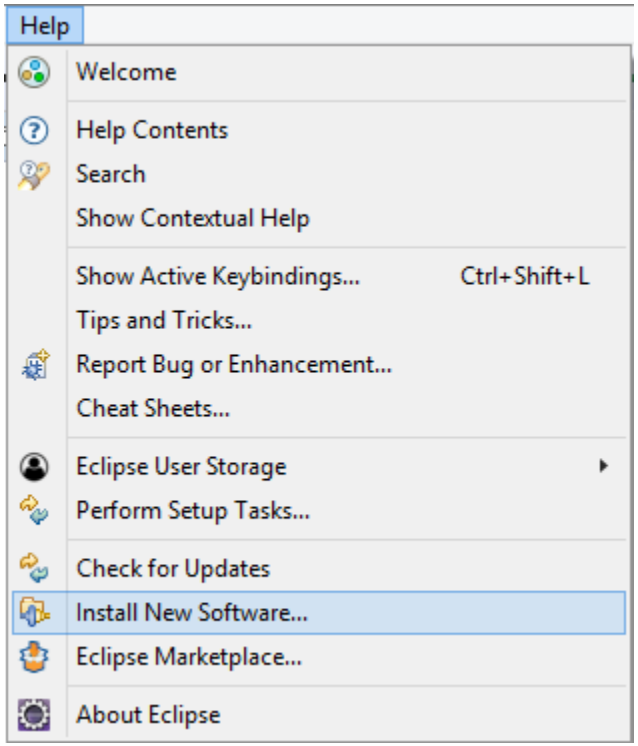
* **Ambiente de Desenvolvimento**
  * Seria extremamente complexo desenvolver interfaces gráficas utilizando somente o código-fonte textual.
  * Por esse motivo, precisamos de ferramentas ou frameworks que facilitem a visualização prévia da interface gráfica durante o trabalho do programador.
  * Essas ferramentas são conhecidas como ferramentas de "drag-and-drop" de componentes. Ou seja, arrastar e soltar componentes para dentro da sua aplicação.

Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * No caso deste curso, iremos utilizar o Eclipse WindowBuilder.
  * O WindowBuilder possui acesso fácil aos componentes SWT e Swing.
  * Link:
    * https://projects.eclipse.org/projects/tools.windowbuilder

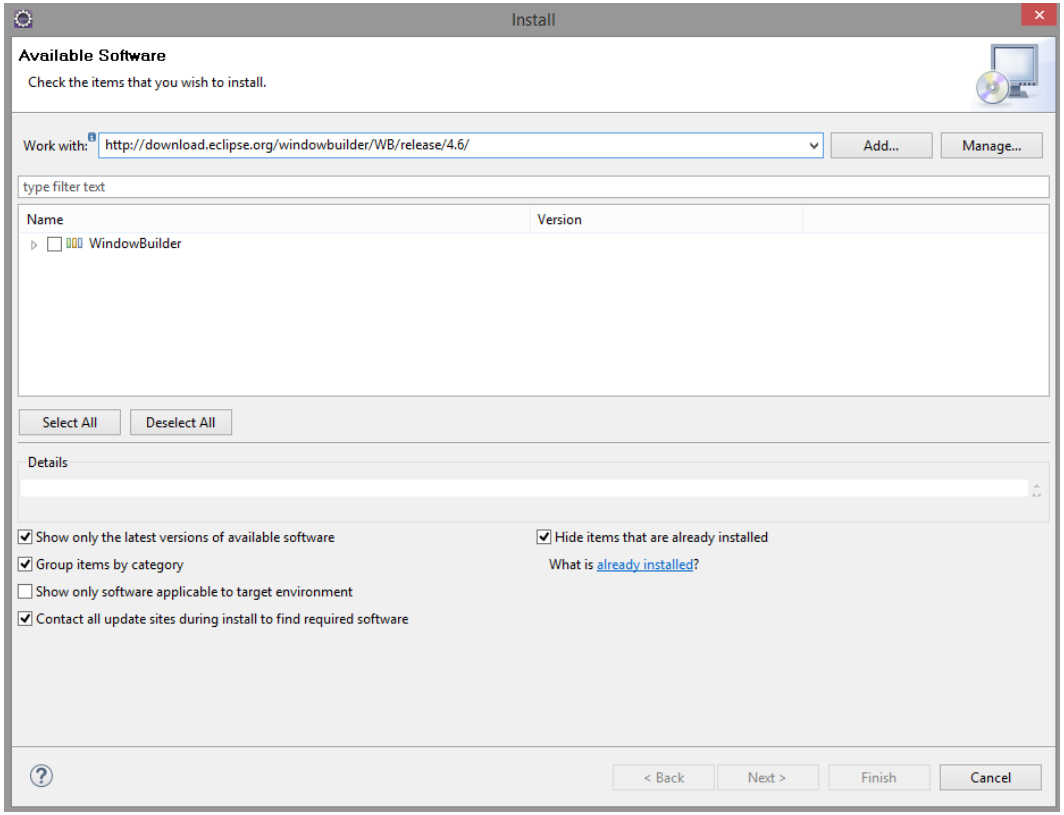
Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * Como instalar:
    * Abra o eclipse e vá em Help > Install New Software.



Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * Como instalar:
    * Cole o link: http://download.eclipse.org/windowbuilder/WB/release/4.6/

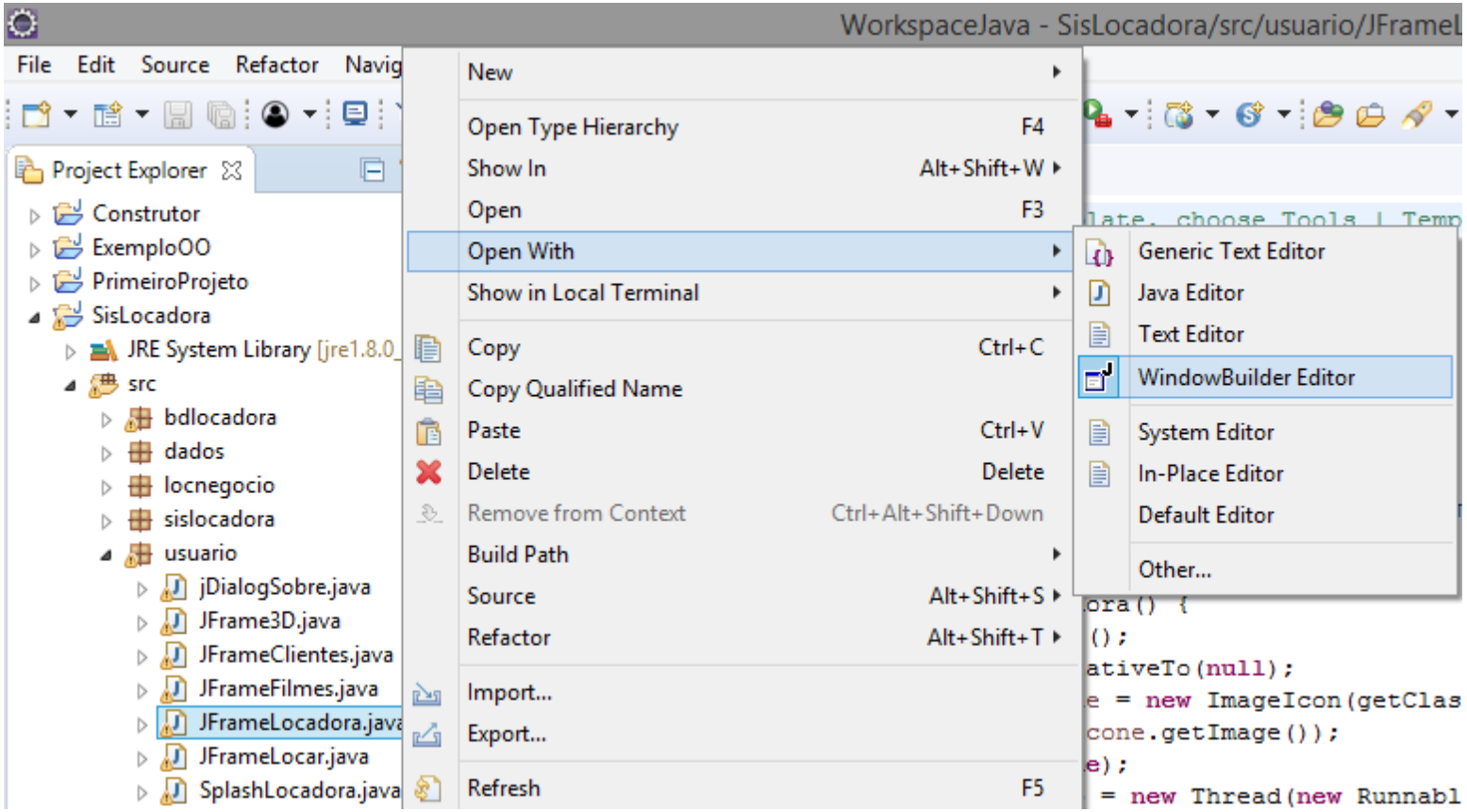


Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * A instalação costuma levar alguns minutos.
  * Após instalado abra o arquivo .Java com o WindowBuilder Editor, conforme mostrado a seguir.

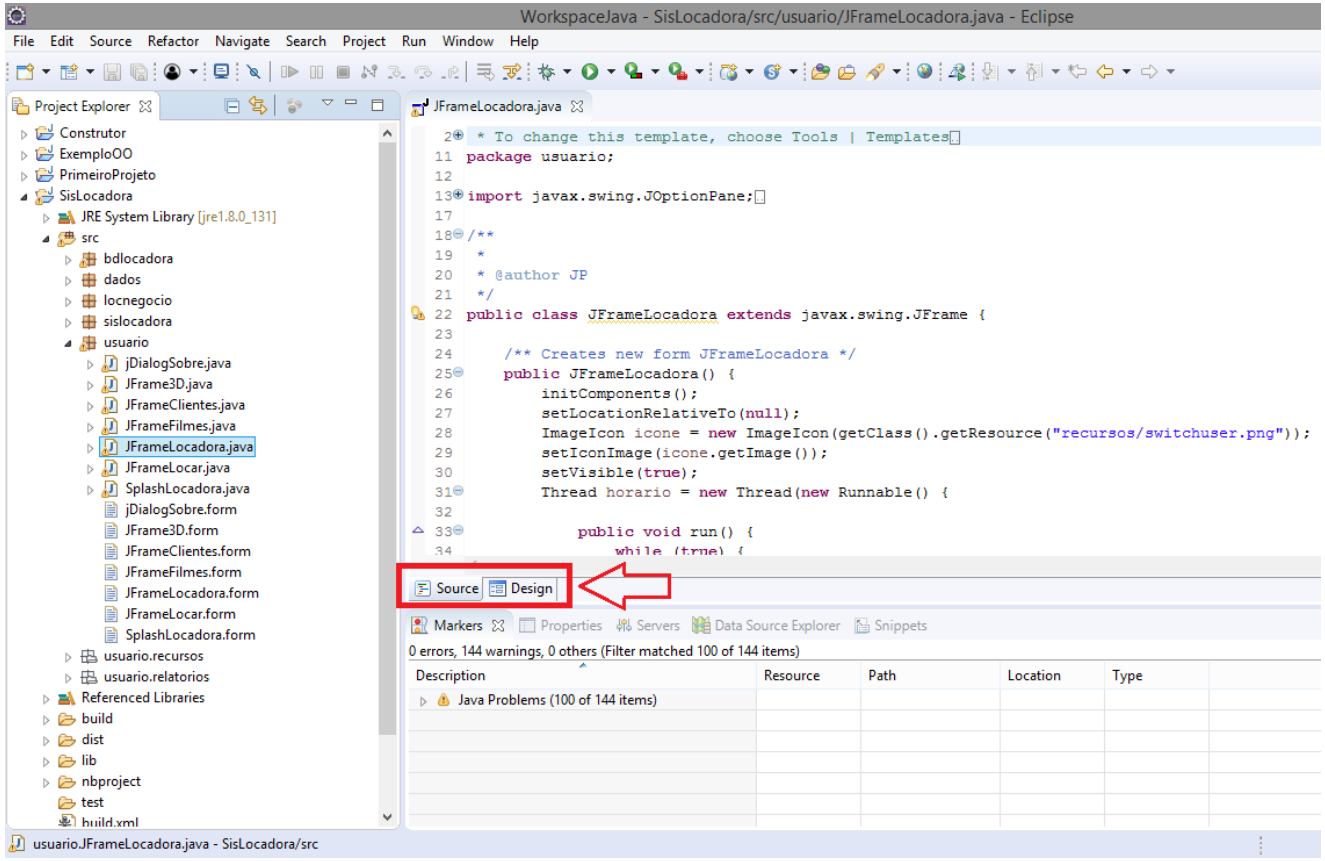
Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * Como abrir:



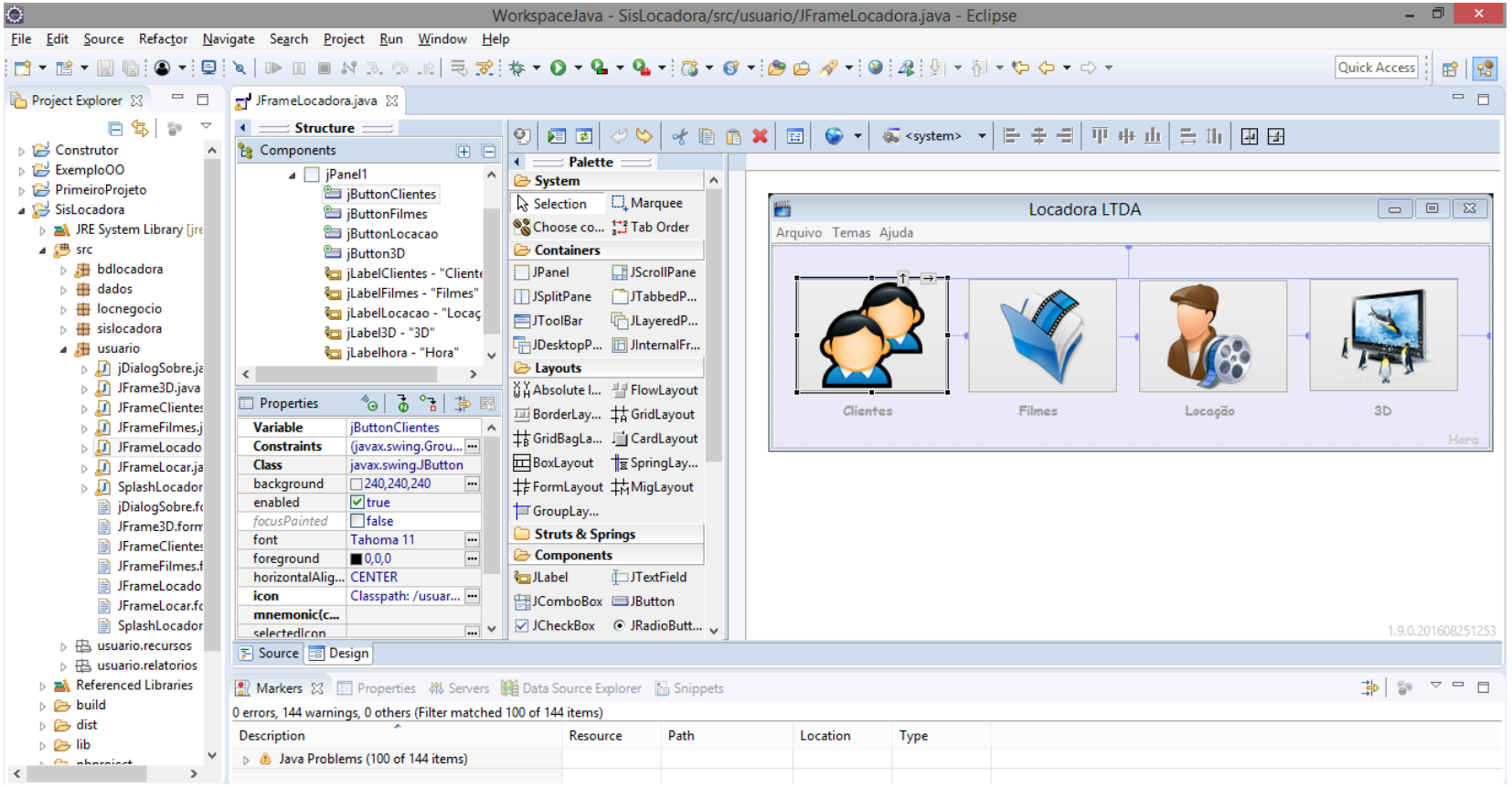
Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * Duas caixas serão apresentadas, ‘Source’ e ‘Design’.



Interface Gráfica com Usuário

* **Ambiente de Desenvolvimento**
  * Clique em 'Design' e comece a experimentar os componentes.



Obrigado.

joaopauloaramuni@gmail.com
joaopauloaramuni@fumec.br

UNIVERSIDADE FUMEC
50 Anos