

# 11

## Aula 1

# Componentes GUI:

## Parte 1



# OBJETIVOS

- Neste capítulo, você aprenderá:
- Os princípios do projeto de interfaces gráficas com o usuário (*graphical user interfaces* – GUIs).
- Como construir GUIs e tratar eventos gerados por interações de usuário com GUIs.
- Como entender os pacotes que contêm componentes GUI, interfaces e classes de tratamento de evento.
- Como criar e manipular botões, rótulos, listas, campos de texto e painéis.
- Como tratar eventos de mouse e eventos de teclado.
- Como utilizar gerenciadores de layout para organizar componentes GUI.



- 11.1** Introdução
- 11.2** Entrada/saída baseada em GUI simples com JOptionPane
- 11.3** Visão geral de componentes Swing
- 11.4** Exibição de texto e imagens em uma janela



# 11.1 Introdução

- **Interface gráfica com o usuário (*graphical user interface* – GUI):**
  - **Apresenta um mecanismo amigável ao usuário para interagir com uma aplicação.**
  - **Freqüentemente contém barra de título, barra de menus que contém menus, botões e caixas de combinação.**
  - **É construída a partir de componentes GUI.**



# Componentes Swing

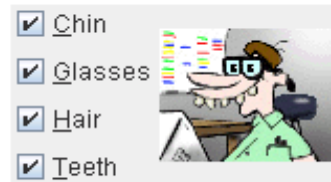
<http://java.sun.com/docs/books/tutorial/uiswing/components/index.html>

## Basic Controls

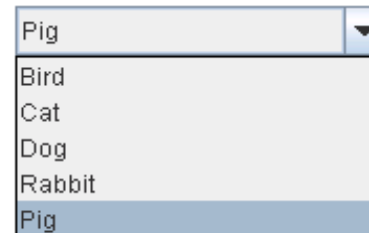
Simple components that are used primarily to get input from the user; they may also show simple state.



[JButton](#)



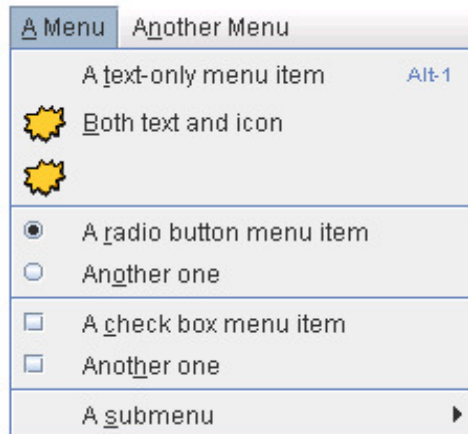
[JCheckBox](#)



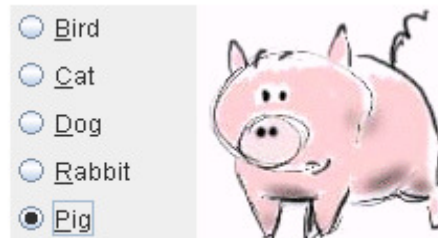
[JComboBox](#)



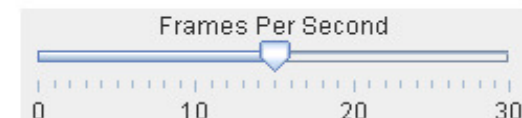
[JList](#)



[JMenu](#)



[JRadioButton](#)



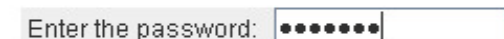
[JSlider](#)



[JSpinner](#)



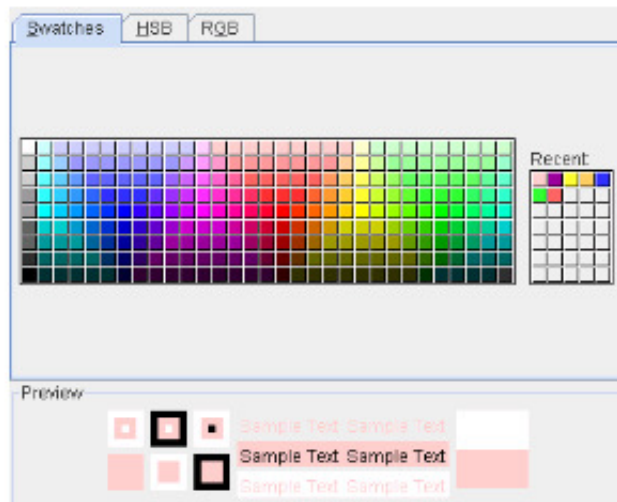
[JTextField](#)



[JPasswordField](#)

## Interactive Displays of Highly Formatted Information

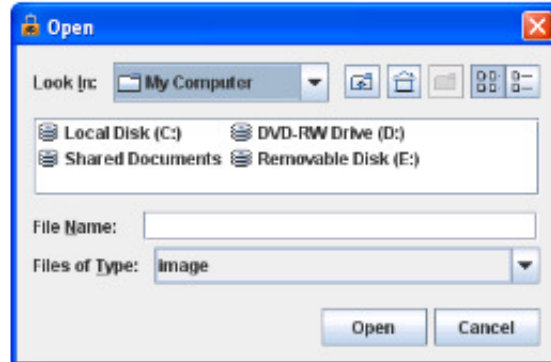
These components display highly formatted information that (if you choose) can be modified by the user.



[JColorChooser](#)



[JEditorPane](#) and [JTextPane](#)



[JFile Chooser](#)

Host	User	Password	Last Modified
Biocca Games	Freddy	#asf6Awzsb	Mar 18, 2006
zabble	ichabod	Taz0!344fZ	Mar 6, 2006
Sun Developer	fraz@hotmail.co...	AasW541!fbZ	Feb 22, 2006
Heirloom Seeds	shams@gmail...	bkz!ADF78!	Jul 28, 2005
Pacific Zoo Shop	seal@hotmail.c...	mbA!124%z	Feb 22, 2006

[JTable](#)

*This is an editable JTextArea. A text area is a "plain" text component, which means that although it can display text in any font, all of the text is in the same font.*

[JTextArea](#)



[JTree](#)

# Componentes Swing



# Componentes Swing

**Uneditable Information Displays**  
 These components exist solely to give the user information.

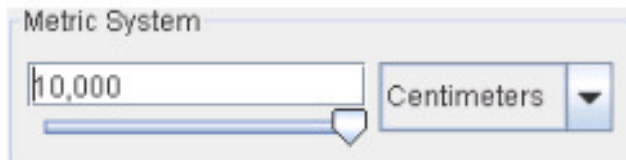
[JLabel](#)      [JProgressBar](#)      [JSeparator](#)      [JToolTip](#)

**Top-Level Containers**  
 At least one of these components must be present in any Swing application.

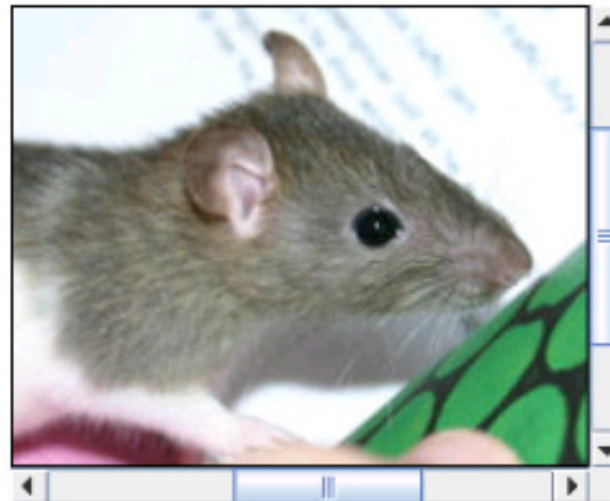
[JApplet](#)      [JDialog](#)      [JFrame](#)

## General-Purpose Containers

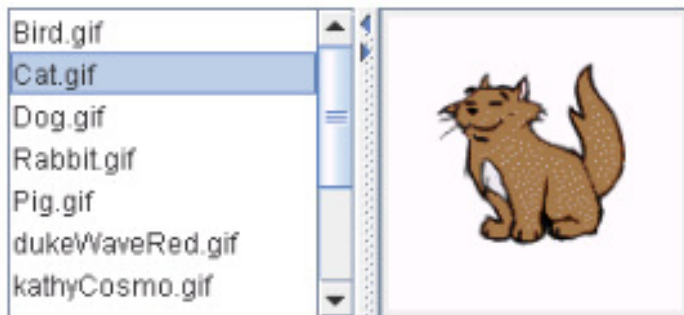
These general-purpose containers are used in most Swing applications.



[JPanel](#)



[JScrollPane](#)



[JSplitPane](#)



[JTabbedPane](#)



[JToolBar](#)

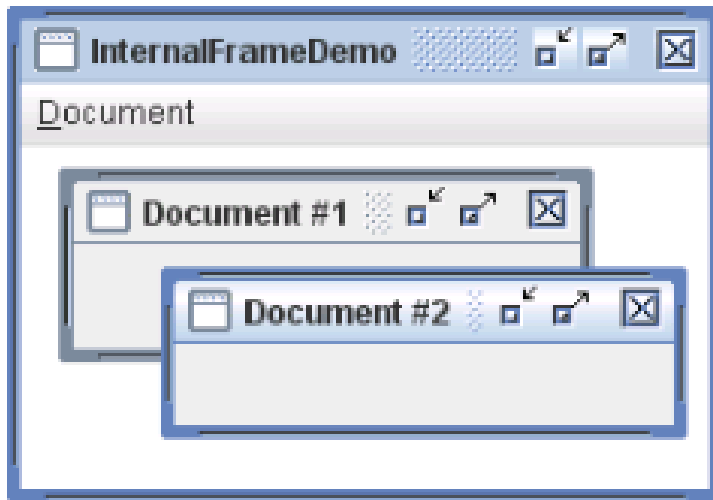
# Componentes Swing



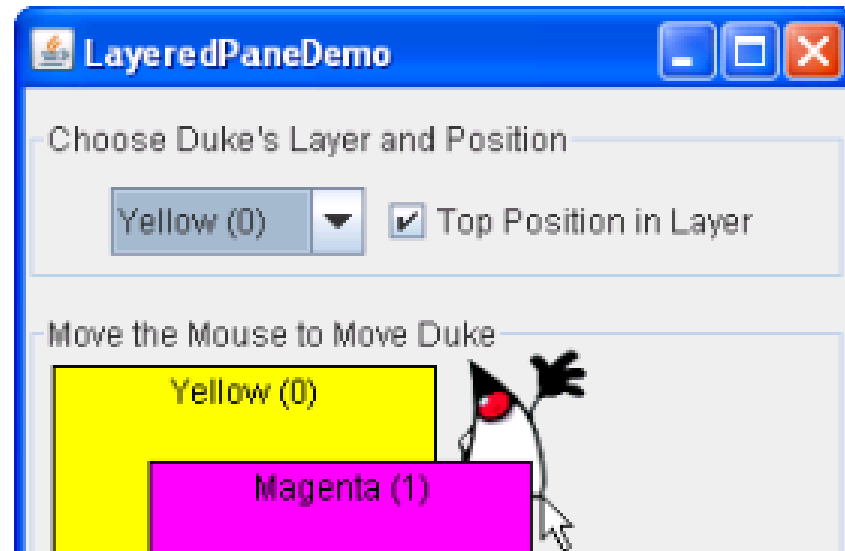


### Special-Purpose Containers

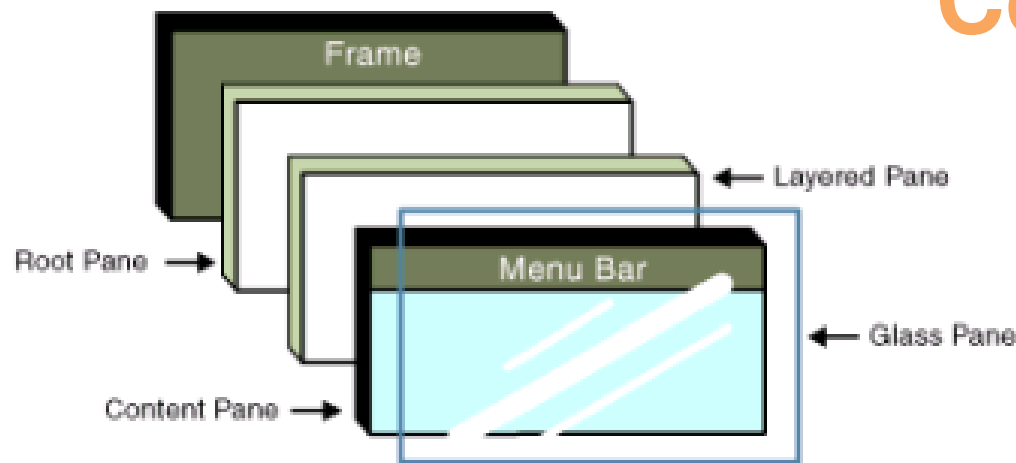
These special-purpose containers play specific roles in the UI.



[JInternalFrame](#)



[JLayeredPane](#)



[Root pane](#)

# Componentes Swing



# Componentes Swing

- **Etapas básicas para utilizar componentes Swing:**

1. **Declarar cada componente Swing**

```
private JLabel lblNome;
```

2. **Alocar valores à cada componente Swing**

```
lblNome = JLabel("Nome");
```

3. **Configurar o posicionamento de cada componente no Frame (quando utilizar Layout absoluto)**

```
lblNome.setBounds(15, 90, 100, 25);
```

4. **Adicionar cada componente no Frame:**

```
Add(lblNome);
```



# Componentes Swing – Exemplo (parte 1)

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

public class Exemplo1 extends JFrame {
    private JLabel lblTitulo, lblRA, lblNome;
    private JTextField edtRA, edtNome;

    public Exemplo1() {
        setTitle("Exemplo de Componentes Swing em Java"); // título do frame
        setPreferredSize (new Dimension (600, 277)); // ajuste do tamanho e layout
        setLayout (null); // aqui a definicao para utilizacao de layout absoluto

        CriarComponentes(); // instancia componentes (objetos Swing);
        AdicionarComponentes(); // adiciona componentes no frame
        ConfigurarPosicoes(); // configura as posices dos componentes;
        pack(); // reorganiza os componentes (objetos) no frame
        setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
        setVisible (true); // deixa o frame visível
    }

    public void CriarComponentes() {
        lblTitulo = new JLabel ("Utilizacao de Componentes Swing em Java");
        lblRA = new JLabel ("R.A."); // Label do R.A.
        edtRA = new JTextField (5); // TextField do R.A.

        lblNome = new JLabel ("Nome"); // Label do Nome
        edtNome = new JTextField (50); // TextField do Nome
    }
}
```



# Componentes Swing – Exemplo (parte 2)

```
public void AdicionarComponentes() {
    add (lblTitulo);    // adiciona o Label do Título no Frame
    add (lblRA);        // adiciona o Label do RA no Frame
    add (lblNome);      // adiciona o Label do Nome no Frame

    add (edtRA);        // adiciona o TextField do RA no Frame
    add (edtNome);      // adiciona o TextField do Nome no Frame
}

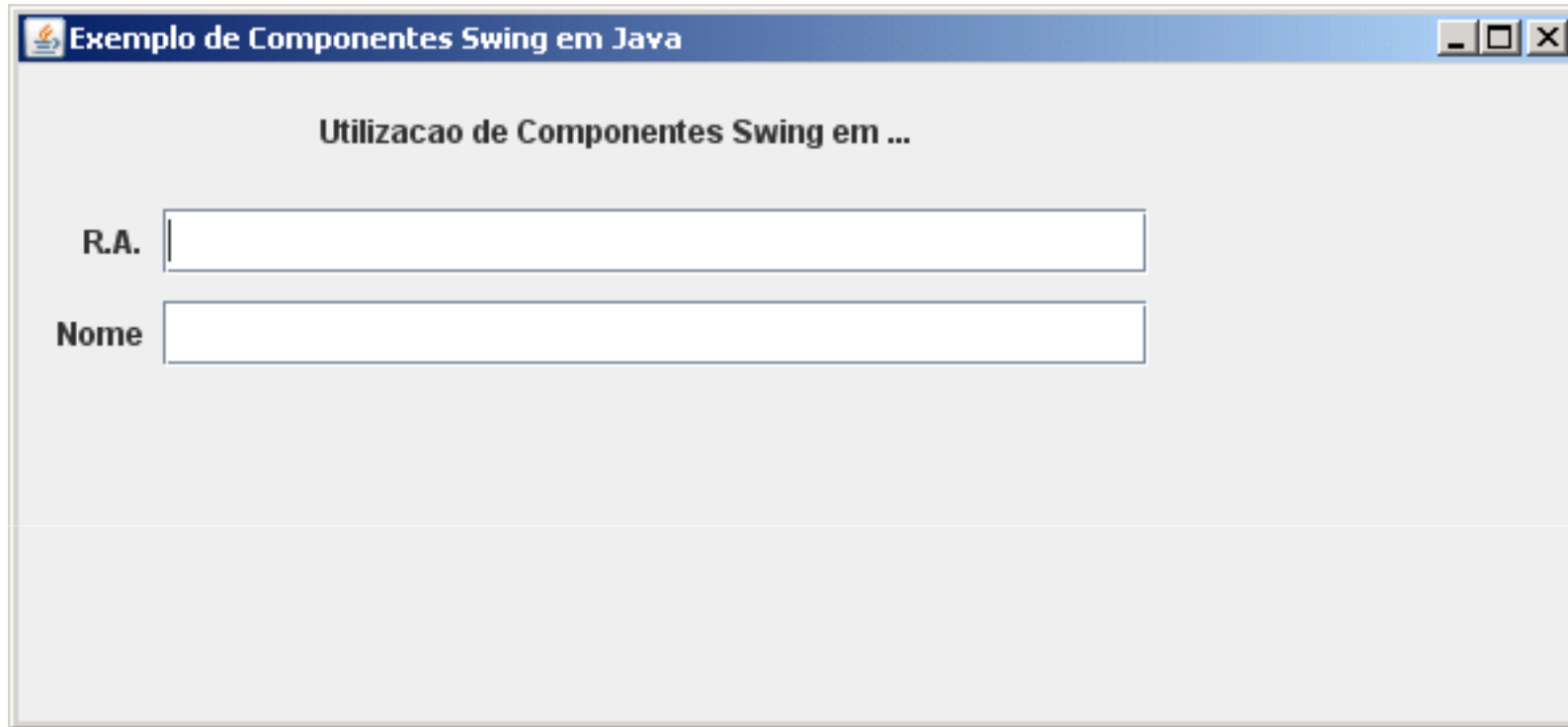
private void ConfigurarPosicoes() {
    // Setbounds( posicao x, posicao y, largura, altura )
    lblTitulo.setBounds (115, 15, 225, 20);
    lblRA.setBounds     ( 25, 55, 100, 25);
    lblNome.setBounds   ( 15, 90, 100, 25);

    edtRA.setBounds     (55, 55, 375, 25);
    edtNome.setBounds   (55, 90, 375, 25);
}

public static void main (String[] args) {
    Exemplo1 obj = new Exemplo1();
}
}
```



# Componentes Swing – Exemplo



Exemplo de Componentes Swing em Java

Utilizacao de Componentes Swing em ...

R.A.

Nome

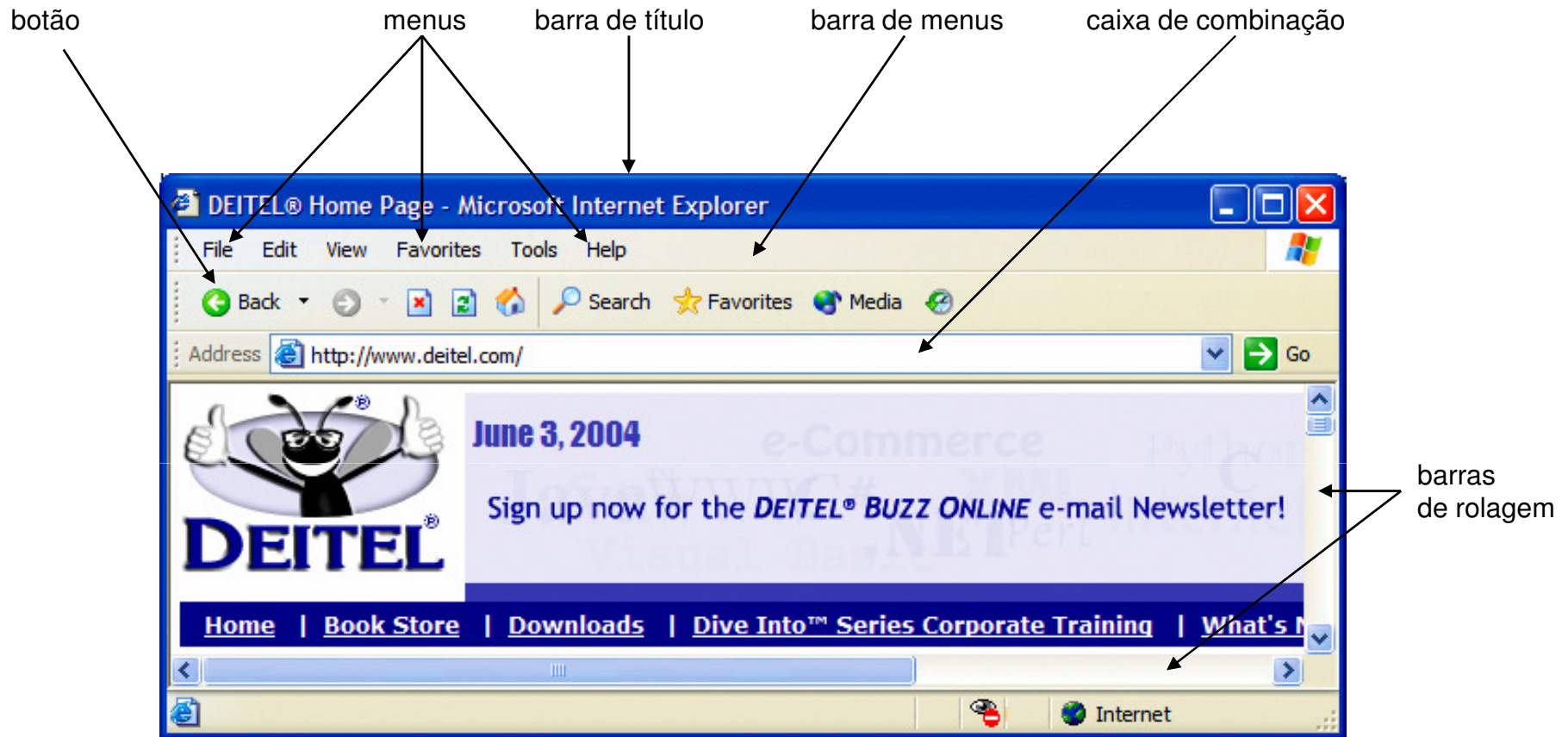


# Observação sobre aparência e comportamento 11.1

---

**Interfaces com o usuário consistentes permitem que o usuário aprenda mais rápido novos aplicativos.**





**Figura 11.1 | Janela do Internet Explorer com componentes GUI.**



## 11.2 Entrada/saída baseada em GUI simples com JOptionPane

- **Caixas de diálogo:**
  - Utilizadas pelas aplicações para interagir com o usuário.
  - Fornecidas pela classe JOptionPane do Java (pacote javax.swing).
    - Contém diálogos de entrada e diálogos de mensagem.





```

1 // Fig. 11.2: Addition.java
2 // Programa de adição que utiliza JOptionPane para entrada e saída.
3 import javax.swing.JOptionPane; // programa utiliza JOptionPane
4
5 public class Addition
6 {
7     public static void main( String args[] )
8     {
9         // obtém a entrada de usuário a partir dos diálogos
10        String firstNumber =
11            JOptionPane.showInputDialog( "Enter first integer" );
12        String secondNumber =
13            JOptionPane.showInputDialog( "Enter second integer" );
14
15        // converte String em valores int para utilização em um cálculo
16        int number1 = Integer.parseInt( firstNumber );
17        int number2 = Integer.parseInt( secondNumber );
18
19        int sum = number1 + number2; // adiciona números
20
21        // exibe o resultado em um diálogo de mensagem JOptionPane
22        JOptionPane.showMessageDialog( null, "The sum is " + sum,
23            "Sum of Two Integers", JOptionPane.PLAIN_MESSAGE );
24    } // fim do método main
25 } // fim da classe Addition

```

Mostra o diálogo de entrada para receber o primeiro inteiro

Mostra o diálogo de entrada para receber o segundo inteiro

Mostra o diálogo de mensagem para gerar a saída da soma para o usuário

# Resumo

## Addition.java

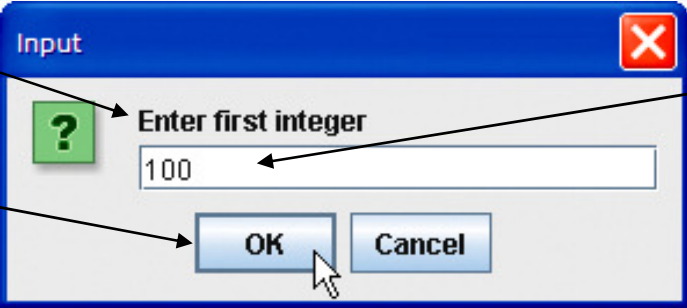
(2 de 2)

Diálogo de entrada exibido nas linhas 10–11

Prompt para o usuário

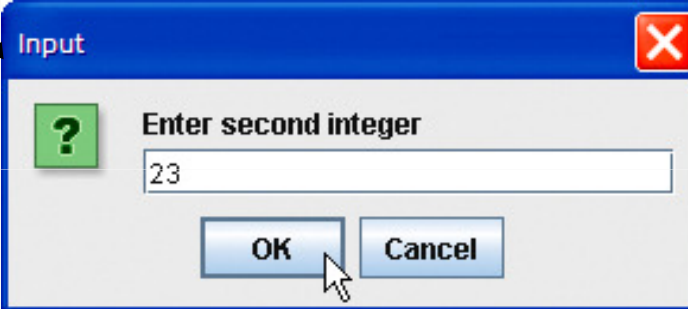
Quando o usuário clica em **OK**, `showInputDialog` retorna ao programa o **100** digitado pelo usuário como uma **String**. O programa deve converter a **String** em um **int**

Campo de texto em que o usuário digita um valor



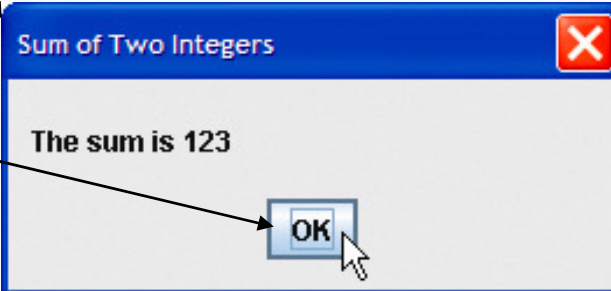
Diálogo de entrada exibido nas linhas 12–13





barra de título



Diálogo de entrada exibido nas linhas 22–23

Quando o usuário clique em **OK**, o diálogo de mensagem é fechado (é removido da tela)




Tipo de diálogo de mensagem	Ícone	Descrição
<code>ERROR_MESSAGE</code>		Um diálogo que indica um erro para o usuário.
<code>INFORMATION_MESSAGE</code>		Um diálogo com uma mensagem informativa para o usuário.
<code>WARNING_MESSAGE</code>		Um diálogo que adverte o usuário de um problema potencial.
<code>QUESTION_MESSAGE</code>		Um diálogo que impõe uma pergunta ao usuário. Normalmente, esse diálogo exige uma resposta, como clicar em um botão Yes ou No.
<code>PLAIN_MESSAGE</code>	Nenhum ícone	Um diálogo que contém uma mensagem, mas nenhum ícone..

**Figura 11.3 | Constantes JOptionPane static para diálogos de mensagem.**



# 11.3 Visão geral de componentes Swing

- **Componentes Swing GUI:**
  - Declarado no pacote `javax.swing`.
  - A maioria dos componentes Swing são componentes *Java puros* — escritos, manipulados e exibidos em Java.
  - Fazem parte das **Java Foundation Classes (JFC)** — bibliotecas do Java para desenvolvimento de GUI para múltiplas plataformas.



Componente	Descrição
<code>JLabel</code>	Exibe texto não-editável ou ícones.
<code>TextField</code>	Permite ao usuário inserir dados do teclado. Também pode ser utilizado para exibir texto editável ou não editável.
<code>Button</code>	Desencadeia um evento quando o usuário clicar nele com o mouse.
<code>CheckBox</code>	Especifica uma opção que pode ser ou não selecionada.
<code>ComboBox</code>	Fornecer uma lista drop-down de itens a partir da qual o usuário pode fazer uma seleção clicando em um item ou possivelmente digitando na caixa.
<code>List</code>	Fornecer uma lista de itens a partir da qual o usuário pode fazer uma seleção clicando em qualquer item na lista. Múltiplos elementos podem ser selecionados.
<code>Panel</code>	Fornecer uma área em que os componentes podem ser colocados e organizados. Também pode ser utilizado como uma área de desenho para imagens gráficas.

**Figura 11.4 | Alguns componentes GUI básicos.**



# Swing *versus* AWT

- **Abstract Window Toolkit (AWT):**
  - **Precursor do Swing.**
  - **Declarado no pacote `java.awt`.**
  - **Não fornece aparência e comportamento consistentes para diversas plataformas.**



## Dica de portabilidade 11.1

---

**Os componentes Swing são implementados no Java; desse modo, eles são mais portáveis e flexíveis do que os componentes Java GUI originais de pacotes `java.awt`, que foram baseados nos componentes GUI da plataforma subjacente. Por essa razão, os componentes Swing GUI geralmente são preferidos.**

---



# Componentes GUI leves *versus* pesados

- **Componentes leves:**
  - Não associados diretamente a componentes GUI suportados pela plataforma subjacente.
- **Componentes pesados:**
  - Associados diretamente à plataforma local.
  - Componentes AWT.
  - Alguns componentes Swing.





## Observação sobre aparência e comportamento 11.4

---

**A aparência e o comportamento de uma GUI definida com componentes GUI pesados no pacote `java.awt` podem variar entre plataformas. Como os componentes pesados são acoplados à GUI da plataforma local, a aparência e o comportamento variam entre plataformas.**



# Superclasses de componentes GUI leves do Swing

- **Classe Component (pacote java.awt):**
  - Subclasse de Object.
  - Declara muitos comportamentos e atributos comuns a componentes GUI.
- **Classe Container (pacote java.awt):**
  - Subclasse de Component.
  - Organiza Components.
- **Classe JComponent (pacote javax.swing):**
  - Subclasse de Container.
  - Superclasse de todos os componentes Swing leves.

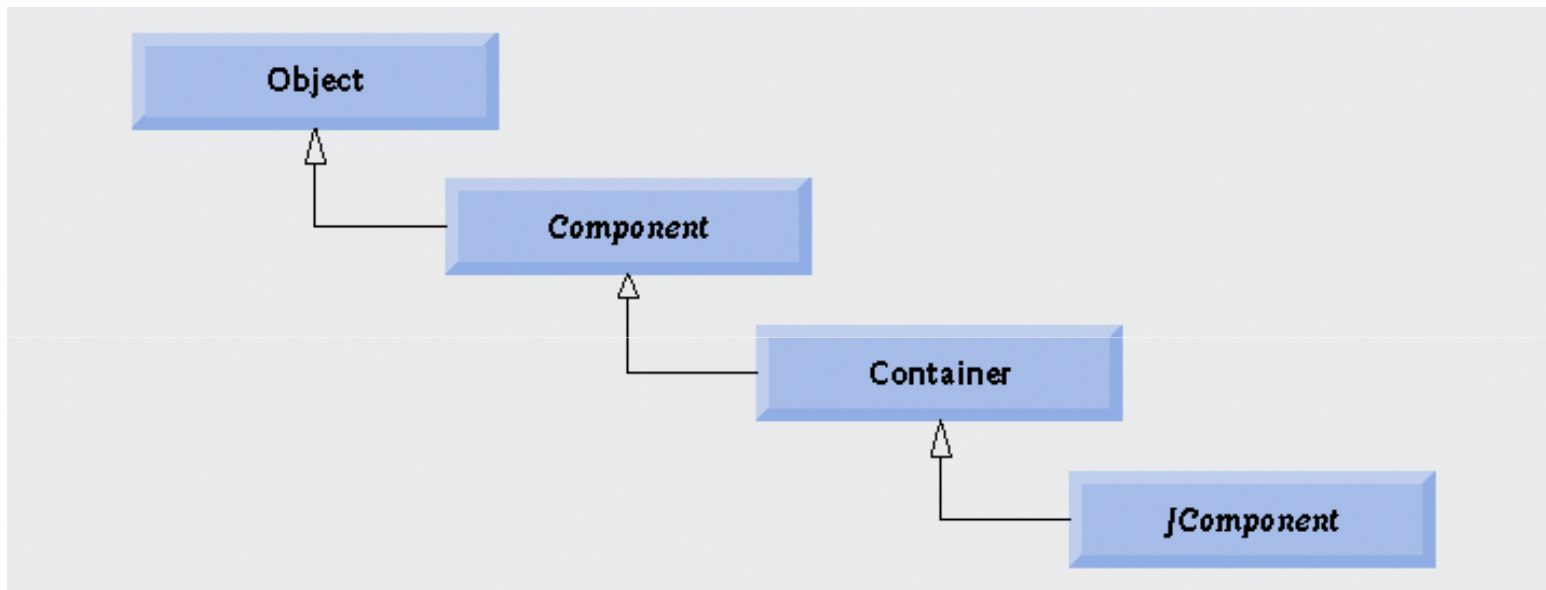


# Observação de engenharia de software 11.1

---

**Estude os atributos e comportamentos das classes na hierarquia de classe da Figura 11.5. Essas classes declaram os recursos que são comuns à maioria dos componentes Swing.**





**Figura 11.5** | Superclasses comuns de muitos dos componentes do Swing.



# Superclasses de componentes GUI leves do Swing

- **Recursos dos componentes leves comuns:**
  - Aparência e comportamento plugáveis para personalizar a aparência dos componentes.
  - Teclas de atalho (chamadas *mnemônicas*).
  - Capacidades comuns de tratamento de eventos.
  - Breves descrições do propósito de um componente GUI (chamadas *dicas de ferramenta*).
  - Suporte para *localização* de interface com o usuário.



# 11.4 Exibição de texto e imagens em uma janela

- **Classe JFrame:**
  - **A maioria das janelas é uma instância ou subclasse dessa classe.**
  - **Fornece a barra de título.**
  - **Fornece botões para minimizar, maximizar e fechar a aplicação.**



# Rotulando componentes GUI

- **Rótulo:**
  - **Instruções de texto ou informações que declaram o propósito de cada componente.**
  - **Criadas com a classe `JLabel`.**



# Observação sobre aparência e comportamento 11.5

---

**Normalmente, o texto em um JLabel emprega maiúsculas e minúsculas no estilo de frases.**





# Especificando o layout

- **Organização dos contêineres:**
  - **Determina onde os componentes são colocados no contêiner.**
  - **Feita no Java com gerenciadores de layout.**
    - Um dos quais é a classe `FlowLayout`.
  - **Configure com o método `setLayout` da classe `JFrame`.**



# Resumo

## LabelFrame.java

(1 de 2)

```
1 // Fig. 11.6: LabelFrame.java
2 // Demonstrando a classe JLabel.
3 import java.awt.FlowLayout; // especifica como os componentes são organizados
4 import javax.swing.JFrame; // fornece recursos básicos de janela
5 import javax.swing.JLabel; // exibe texto e imagens
6 import javax.swing.SwingConstants; // constantes comuns utilizadas com Swing
7 import javax.swing.Icon; // interface utilizada para manipular imagens
8 import javax.swing.ImageIcon; // carrega imagens
9
10 public class LabelFrame extends JFrame
11 {
12     private JLabel label1; // JLabel apenas com texto
13     private JLabel label2; // JLabel construído com texto e ícone
14     private JLabel label3; // JLabel com texto e ícone adicionados
15
16     // Construtor LabelFrame adiciona JLabels a JFrame
17     public LabelFrame()
18     {
19         super( "Testing JLabel" );
20         setLayout( new FlowLayout() ); // configura o layout de frame
21
22         // Construtor JLabel com um argumento de string
23         label1 = new JLabel( "Label with text" );
24         label1.setToolTipText( "This is label1" );
25         add( label1 ); // adiciona label1 a JFrame
26
```



# Resumo

## LabelFrame.java

(2 de 2)

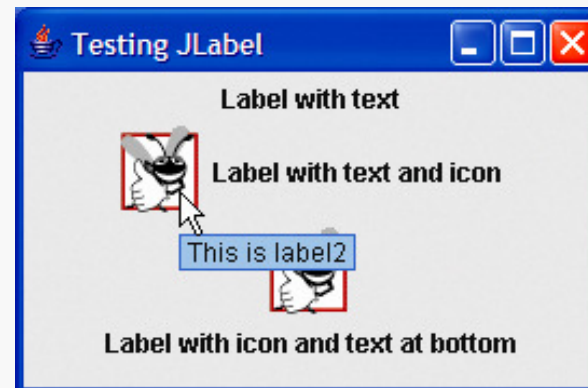
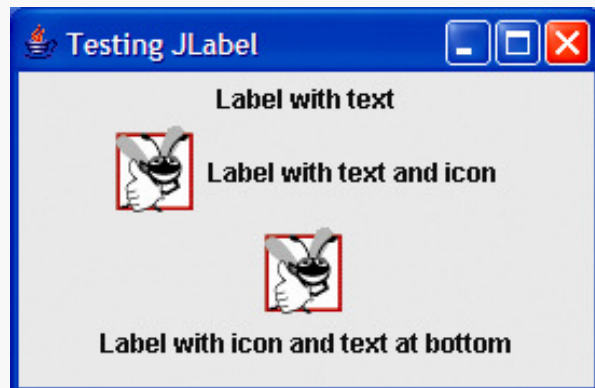
```
27 // construtor JLabel com string, Icon e argumentos de alinhamento
28 Icon bug = new ImageIcon( getClass().getResource( "bug1.gif" ) );
29 label2 = new JLabel( "Label with text and icon", bug,
30     SwingConstants.LEFT );
31 label2.setToolTipText( "This is label2" );
32 add( label2 ); // adiciona label2 a JFrame
33
34 label3 = new JLabel(); // construtor JLabel sem argumentos
35 label3.setText( "Label with icon and text at bottom" );
36 label3.setIcon( bug ); // adiciona ícone a JLabel
37 label3.setHorizontalTextPosition( SwingConstants.CENTER );
38 label3.setVerticalTextPosition( SwingConstants.BOTTOM );
39 label3.setToolTipText( "This is label3" );
40 add( label3 ); // adiciona label3 a JFrame
41 } // fim do construtor LabelFrame
42 } // fim da classe LabelFrame
```



# Resumo

## LabelTest.java

```
1 // Fig. 11.7: LabelTest.java
2 // Testando JLabelFrame.
3 import javax.swing.JFrame;
4
5 public class LabelTest
6 {
7     public static void main( String args[] )
8     {
9         JLabelFrame labelFrame = new JLabelFrame(); // cria JLabelFrame
10        labelFrame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
11        labelFrame.setSize( 275, 180 ); // configura tamanho do frame
12        labelFrame.setVisible( true ); // exhibe frame
13    } // fim de main
14 } // fim da classe LabelTest
```



# Criando e anexando JLabel

- **Método setToolTipText da classe JComponent:**
  - Especifica a dica de ferramenta.
- **Método add da classe Container:**
  - Adiciona um componente a um contêiner.



# Erro comum de programação 11.1

---

**Se você não adicionar explicitamente um componente GUI a um contêiner, o componente GUI não será exibido quando o contêiner aparecer na tela.**



# Observação sobre a aparência e comportamento 11.6

---

**Utilize as dicas de ferramenta para adicionar texto descritivo aos componentes GUI. Esse texto ajuda o usuário a determinar o propósito do componente GUI na interface com o usuário.**



# Criando e anexando JLabel

- **Interface ICON:**
  - Pode ser adicionado a uma `JLabel` com o método `setIcon`.
  - Implementado pela classe  `ImageIcon`.
- **Interface SwingConstants:**
  - Declara um conjunto de constantes inteiras comuns, como as utilizadas para configurar o alinhamento dos componentes.
  - Pode ser utilizada com os métodos `setHorizontalAlignment` e `setVerticalAlignment`.





# Criando e anexando JLabel

- **Outros métodos JLabel:**
  - `getText` e `setText`
    - **Para configurar e recuperar o texto de um rótulo.**
  - `setIcon` e `setIcon`
    - **Para configurar e recuperar o ícone exibido no rótulo.**
  - `getHorizontalAlignment` e `setHorizontalAlignment`
    - **Para configurar e recuperar a posição horizontal do texto exibido no rótulo.**



Constante	Descrição
<i>Constantes de posição horizontal</i>	
<code>SwingConstants.LEFT</code>	Coloca o texto à esquerda.
<code>SwingConstants.CENTER</code>	Coloca o texto no centro.
<code>SwingConstants.RIGHT</code>	Coloca o texto à direita.
<i>Constantes de posição vertical</i>	
<code>SwingConstants.TOP</code>	Coloca o texto na parte superior.
<code>SwingConstants.CENTER</code>	Coloca o texto no centro.
<code>SwingConstants.BOTTOM</code>	Coloca o texto na parte inferior.

Figura 11.8 | Alguns componentes GUI básicos.



# Criando e exibindo uma janela LabelFrame

- **Outros métodos JFrame:**
  - `setDefaultCloseOperation`
    - **Determina como a aplicação reage quando o usuário clica no botão de fechar.**
  - `setSize`
    - **Especifica a largura e altura da janela.**
  - `setVisible`
    - **Determina se a janela é exibida (`true`) ou não (`false`).**



# Exemplo de Componentes Swing

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

public class LayoutAbsoluto extends JFrame {
    private JLabel lblTitulo, lblRA, lblCurso, lblNome, lblSemestre;
    private JTextField edtRA, edtNome, edtCurso;
    private JComboBox boxSemestre;
    private JButton btnGravar, btnSair;
    private String[] boxBimestreItems = {"Primeiro", "Segundo", "Terceiro", "Quarto"};

    public LayoutAbsoluto() {
        setTitle("Exemplo de Layout Absoluto em Java");

        //ajuste do tamanho e layout
        setPreferredSize (new Dimension (456, 277));
        setLayout (null); // aqui a definicao para utilizacao de layout absoluto

        CriarComponentes(); // instancia componentes (objetos Swing);
        AdicionarComponentes(); // adiciona componentes no frame
        ConfigurarPosicoes(); // configura as posicoes dos componentes;
        pack(); // reorganiza os componentes (objetos) no frame
        setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
        setVisible (true);
    }
}
```



# Exemplo de Componentes Swing

```
public void CriarComponentes() {
    lblTitulo = new JLabel ("Utilizao de Leioute Absoluto em Java");
    edtRA = new JTextField (5);
    edtNome = new JTextField (5);
    edtCurso = new JTextField (5);
    lblRA = new JLabel ("R.A.");
    lblCurso = new JLabel ("Curso");
    lblNome = new JLabel ("Nome");
    lblSemestre = new JLabel ("Semestre");
    boxSemestre = new JComboBox (boxBimestreItems);
    btnGravar = new JButton ("Gravar");
    btnSair = new JButton ("Sair");
}

public void AdicionarComponentes() {
    add (lblTitulo);
    add (edtRA);
    add (edtNome);
    add (edtCurso);
    add (lblRA);
    add (lblCurso);
    add (lblNome);
    add (lblSemestre);
    add (boxSemestre);
    add (btnGravar);
    add (btnSair);
}
```



# Exemplo de Componentes Swing

```
private void ConfigurarPosicoes() {
    //Configura a posicao dos objetos no frame
    // Setbounds( posicao x, posicao y, largura, altura )
    lblTitulo.setBounds (115, 15, 225, 20);
    lblRA.setBounds ( 25, 55, 100, 25);
    lblNome.setBounds ( 15, 90, 100, 25);
    lblCurso.setBounds ( 15, 125, 100, 25);
    lblSemestre.setBounds(290, 125, 65, 25);

    edtRA.setBounds (55, 55, 375, 25);
    edtNome.setBounds (55, 90, 375, 25);
    edtCurso.setBounds(55, 125, 220, 25);

    boxSemestre.setBounds(350, 125, 80, 25);
    btnGravar.setBounds (100, 190, 100, 25);
    btnSair.setBounds (260, 190, 100, 25);
}

public static void main (String[] args) {
    LaiouteAbsoluto obj = new LaiouteAbsoluto();
}
}
```



# Exercícios

- **Implemente os seguintes programas:**
  - **Exemplo1.java** (páginas 12 e 13)
  - **Addition.java** (17)
  - **LabelFrame.java** (páginas 34 e 36)
  - **LabelTest.java** (páginas 44, 45 e 46).
- **Implemente um programa que contenha diversos componentes Swing (utilize a maior diversidade possível).**

