

# Interface Gráfica



# Java GUI Frameworks

- Abstract Window Toolkit (java.awt)
  - Fez parte do primeiro framework GUI do java.
  - Ligado ao sistema operacional (windows system manager)
- Swing (Java Foundation Classes)
  - Segunda geração de desenvolvimento gráficos para GUI Java. Introduzida no JDK 1.2.
  - Independente do sistema operacional



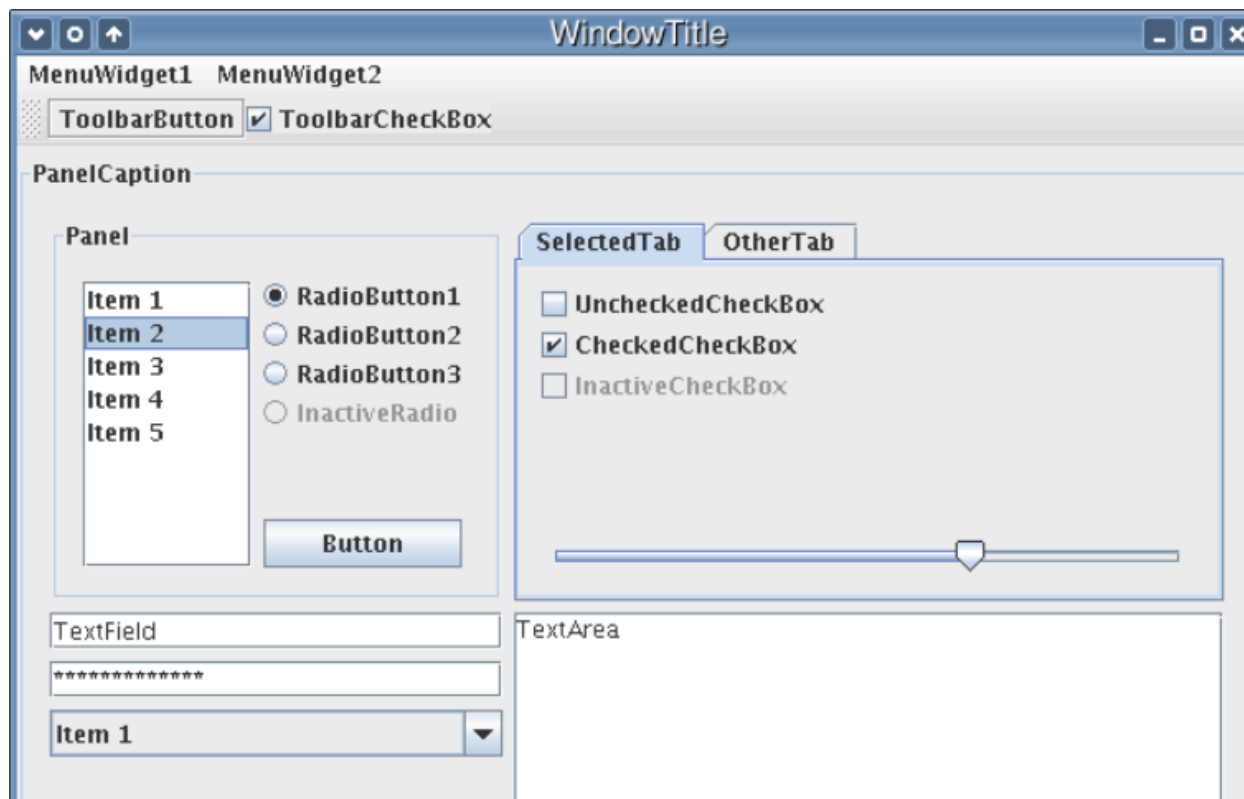
# AWT

- Criada em 1995 para J2SE 1.0
- Camada de abstração sobre GUI nativa
- Maior integridade com aplicações nativas
- Diferença na execução entre diferentes sistemas



# Swing

- Escrito em Java
- Aparência consistente em plataformas diferentes

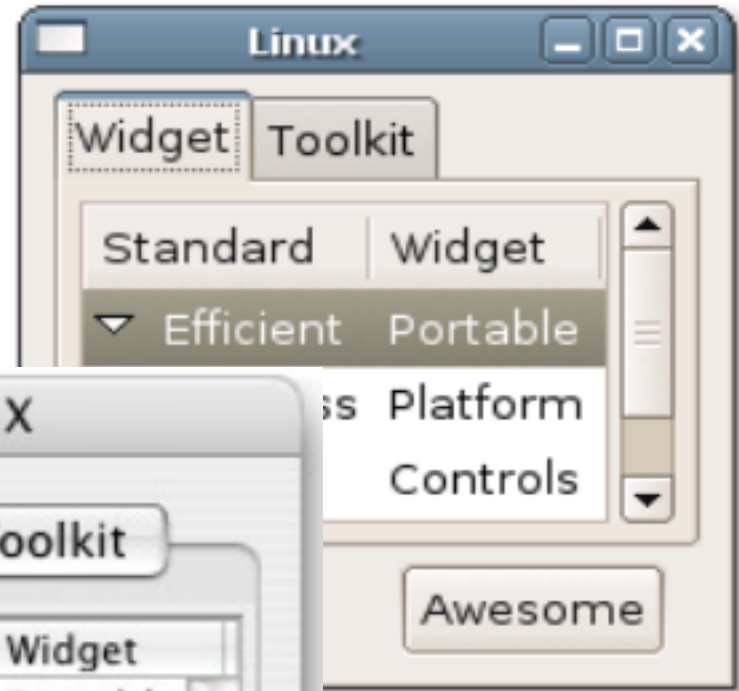
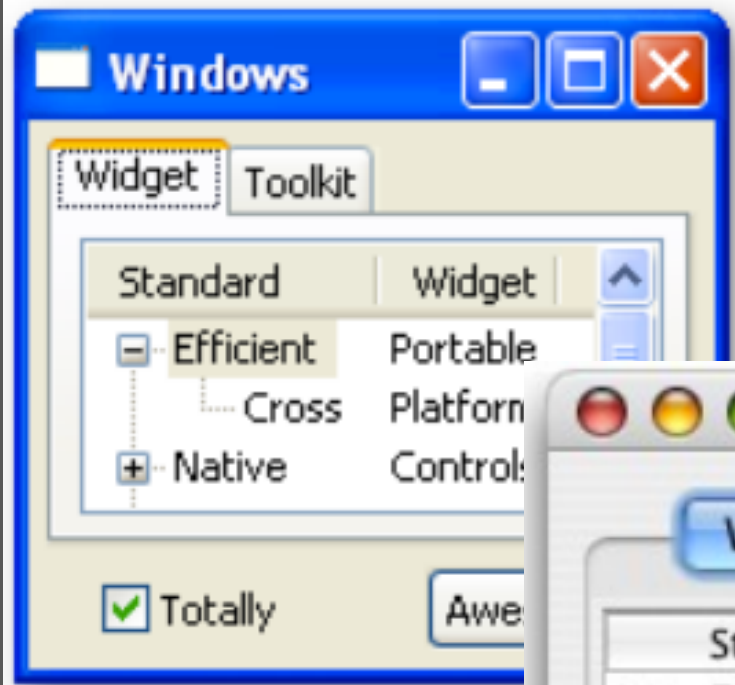


# SWT – Standard Widget Toolkit

- Abertura do IBM Visual Age (IDE) – Eclipse
- Proposta intermediária entre o AWT e o Swing
  - JNI para toolkits nativos
  - Implementa códigos próprios
- JFace
  - Classes utilitárias para implementação de tarefas complexas em SWT



# SWT – Standard Widget Toolkit



# Java FX

- **JavaFX Script** é uma linguagem de script.
- O domínio da plataforma engloba o desenvolvimento de RIAs (Rich Internet Application) para desktops e dispositivos móveis.
- Você pode criar aplicativos para diversas plataformas usando JavaFX: desktop, celular, web, televisão digital.
- A compatibilidade de JavaFX através das plataforma é de 80%, isso quer dizer que seu código Desktop pode ser o mesmo que vai rodar no celular, ou com pequenas alterações.



# Ferramentas

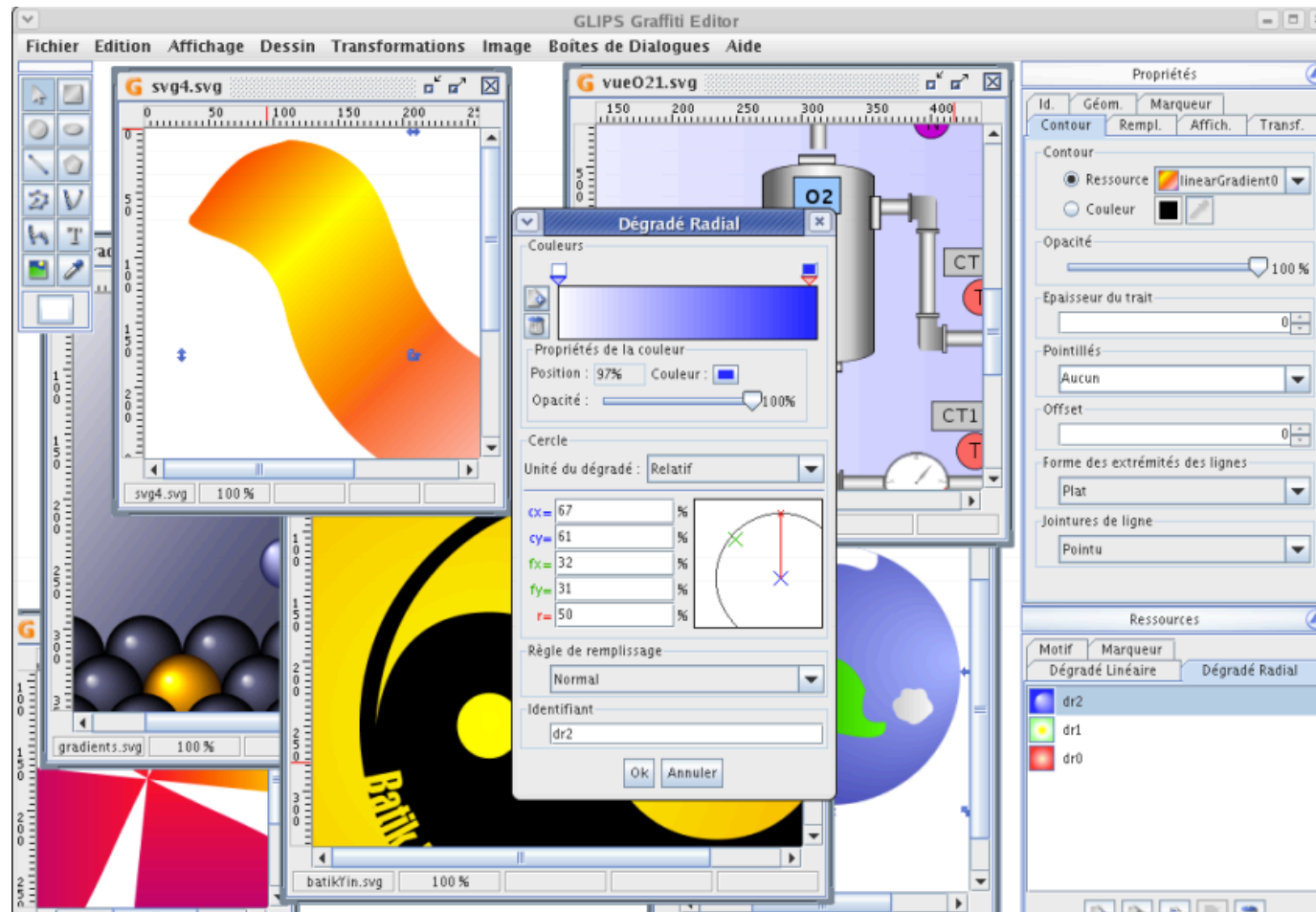
- Integradas às IDEs
  - Netbeans (Matisse)
  - Eclipse (Visual Editor, Matisse4Eclipse)
- E muitas outras:
  - Swing Designer
  - JFormDesigner
  - FormLayoutMaker
  - Abeille
  - Etc.





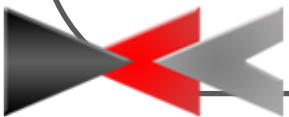
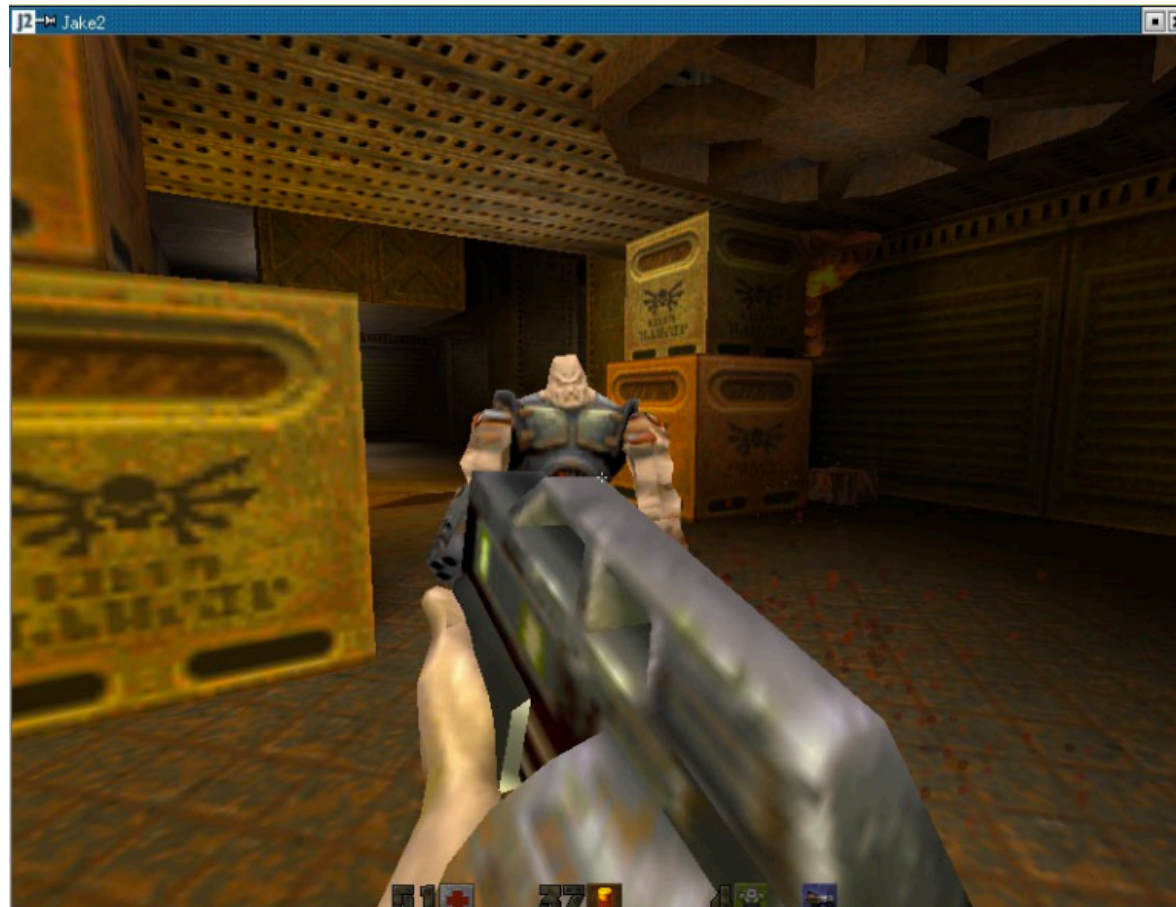
# Swing - Aplicações

- GLIPS



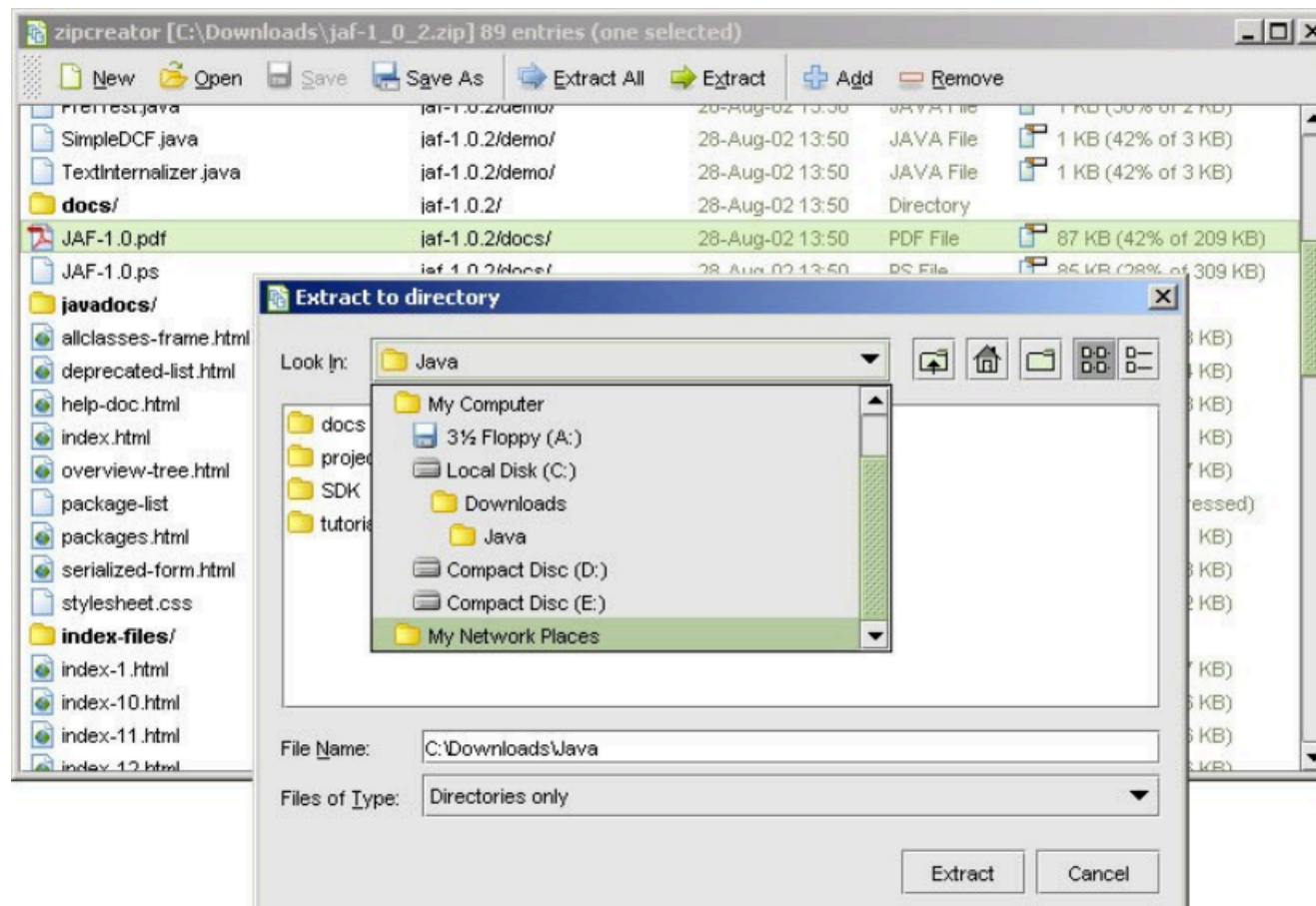
# Swing - Aplicações

- Jogos



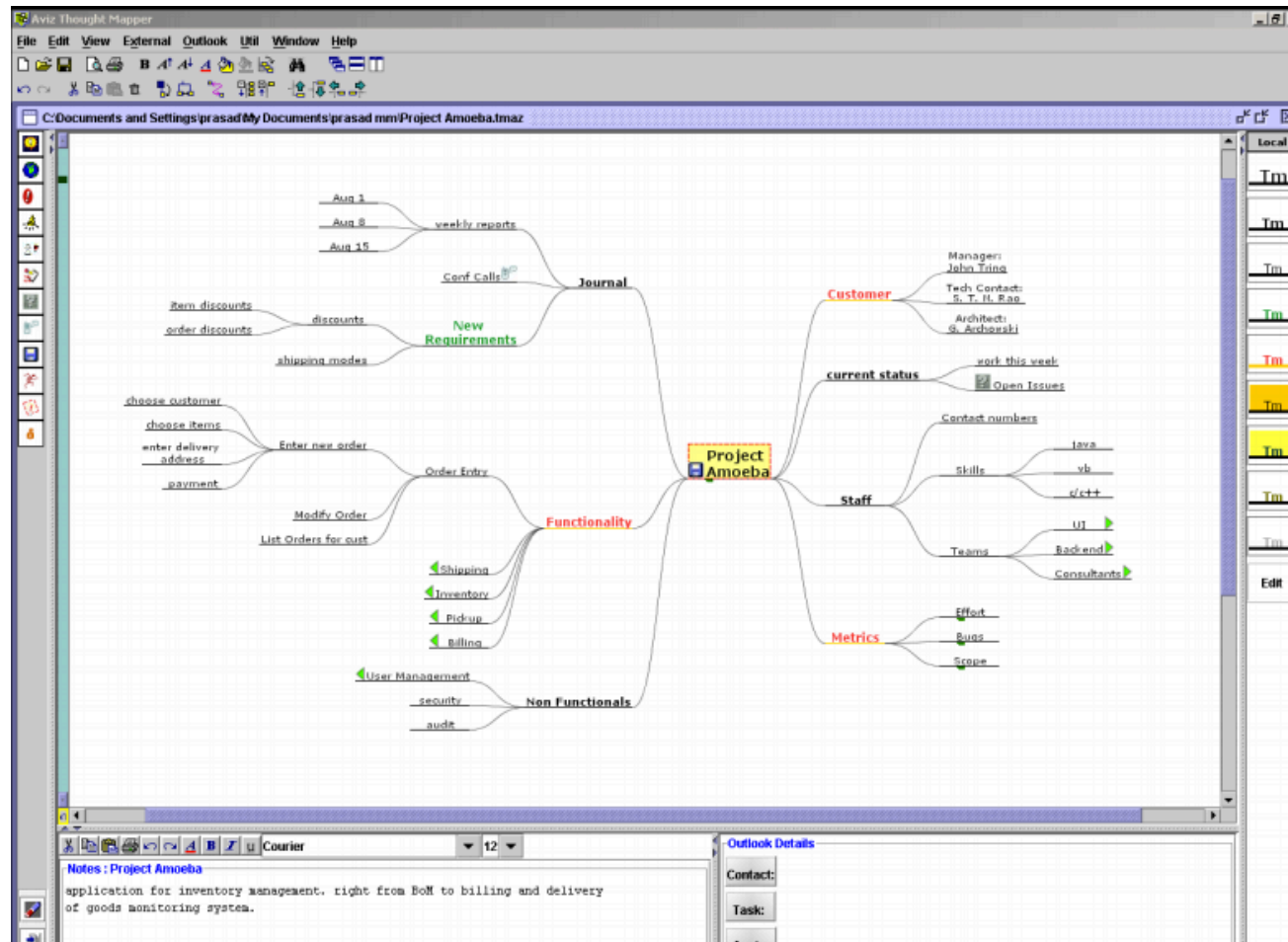
# Swing - Aplicações

- ZipCreator



# Swing - Aplicações

- Thought Mapper



# Swing - Aplicações

- E muitas outras:
  - Azureus (Bit Torrent)
  - Receita federal
  - NetBeans
  - Etc.
- Java Foundation Class
  - Ferramenta Matisse

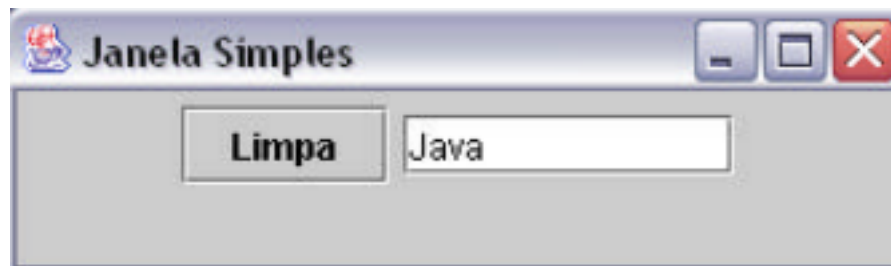


# Processo Básico

- OO + Eventos
- Instanciar os componentes da interface
  - Janelas, botões, campos de texto etc.
- Adicionar os componentes em containers
  - Como os componentes podem ser agrupados e qual o layout de diagramação
- Especificar o tratamento de eventos da interface
  - O que deve acontecer quando o usuário clica em um botão?



# Exemplo



# Exemplo

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

public class JanelaSimples {

    public JanelaSimples() {
        /* Cria o botão */
        final JButton botaoLimpa = new JButton("Limpa");
        /* Cria o campo de texto */
        final JTextField campoTexto = new JTextField(10);
        /* Cria uma janela */
        final JFrame janela = new JFrame ("Janela Simples");
        janela.setSize(300,100);
        /* Adiciona os componentes na janela */
        JPanel painel = new JPanel();
        painel.add (botaoLimpa);
        painel.add (campoTexto);
        janela.getContentPane().add(painel);
    }
}
```





# Exemplo

```
/* Quando o usuário clicar no botao, limpa o campo de texto */
    botaoLimpa.addActionListener (new ActionListener() {
        public void actionPerformed (ActionEvent e) {
            campoTexto.setText("");
        }
    });
/* Exibe a janela */
    janela.setVisible(true);
}

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

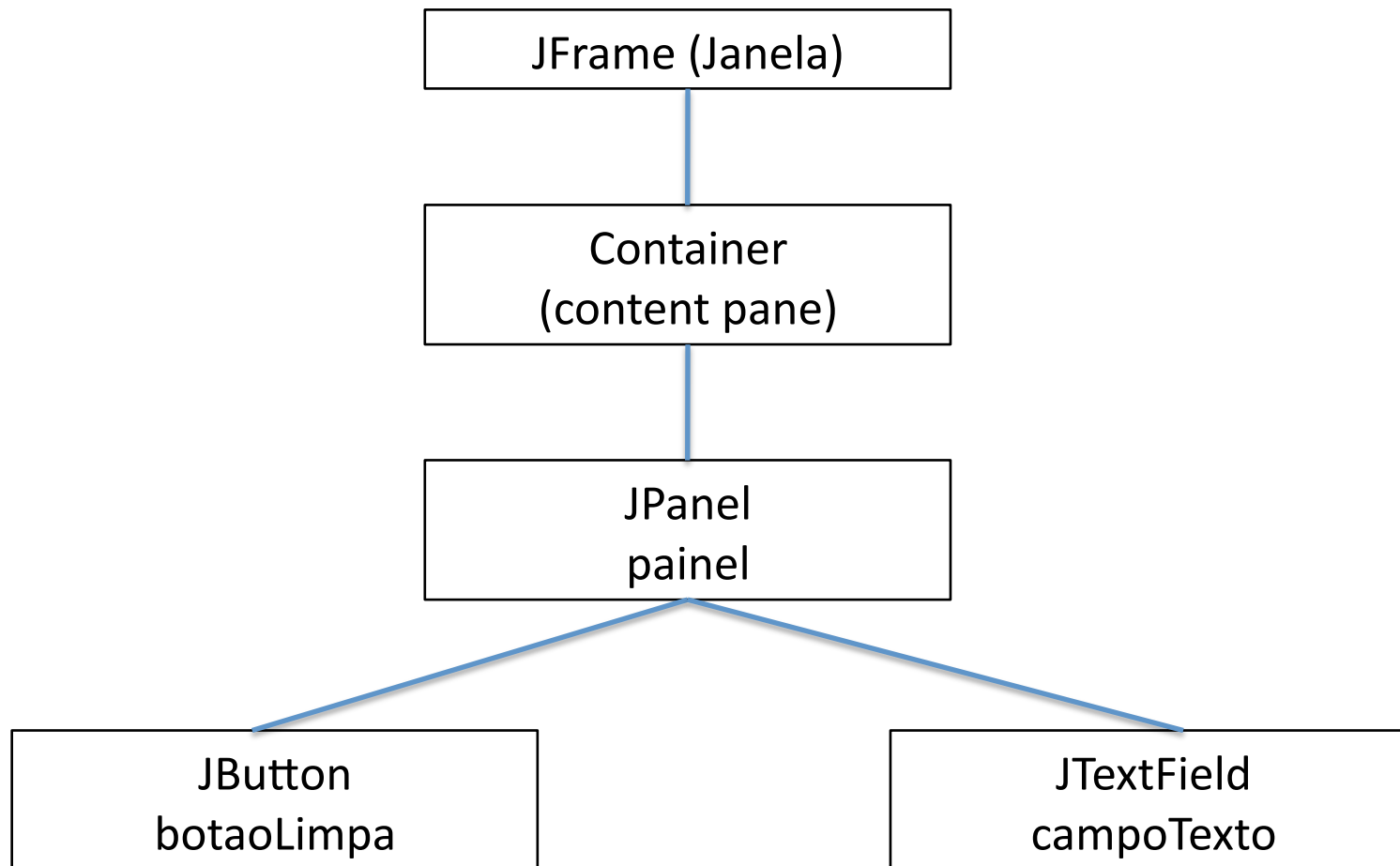


# Hierarquia de Composição

- Componente
  - Qualquer elemento de interface
- Container
  - É um componente
  - Agrega Componentes



# Exemplo



# Elementos Swing

- 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 elementos de componentes atômicos: elementos de interface que não agrupam outros componentes



# Exemplos de Componentes Atômicos

- Campo de Texto
- Label
- Botão
- CheckBox
- Lista
- Lista DropDown
- Canvas



# Classe JComponent

- Superclasse de muitos elementos do Swing
- Funcionalidade comum aos componentes
- Disponibiliza métodos para controlar:
  - Tool tips
  - Bordas
  - Propriedades
  - Etc.



# javax.swing.JLabel

- Modela um texto e/ou imagem não editável, isto é, sem interação com o usuário



# Exemplo - JLabel

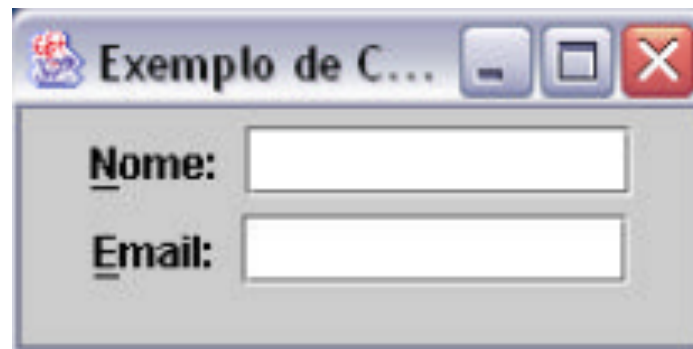
```
/* Cria um label com texto */  
JLabel label1 = new JLabel("Label1: Apenas Texto");  
  
/* Cria um label com texto e imagem */  
JLabel label2 = new JLabel("Label2: Imagem e texto", new  
    ImageIcon("javalogo.gif"), JLabel.CENTER);  
  
label2.setVerticalTextPosition(JLabel.BOTTOM);  
label2.setHorizontalTextPosition(JLabel.CENTER);
```





# javax.swing.JTextField

- Modela um campo de edição de texto de uma linha



# Exemplo - JTextField

```
/* 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
```



# javax.swing.JButton

- Modela um push-button



# Exemplo - JButton

```
/* 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"));
```



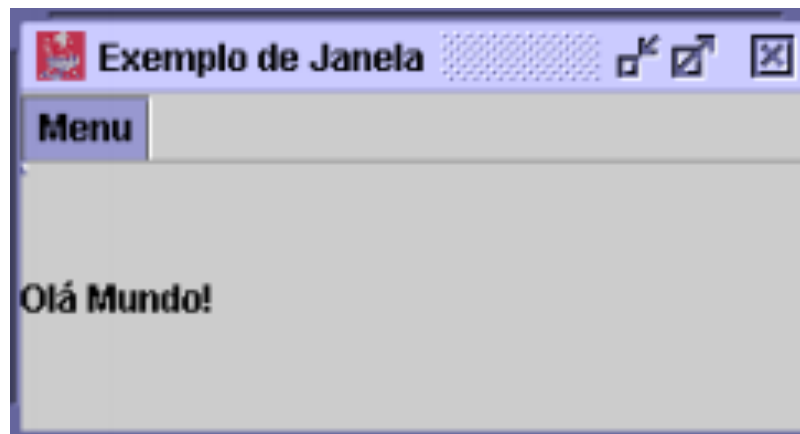
# Alguns Containers

- Top Level Containers
  - Janela
  - Diálogo
  - Applet
- Containers Intermediários
  - Painel
  - Scroll Pane



# javax.swing.JFrame

- Representa uma janela do sistema nativo
- Possui título e borda
- Pode possuir menu



# Exemplo - 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();
```



# javax.swing.JPanel

- Modela um container sem decoração
- Representa um grupo de elementos
- Normalmente usado para estruturar a interface
  - Associado a um diagramador





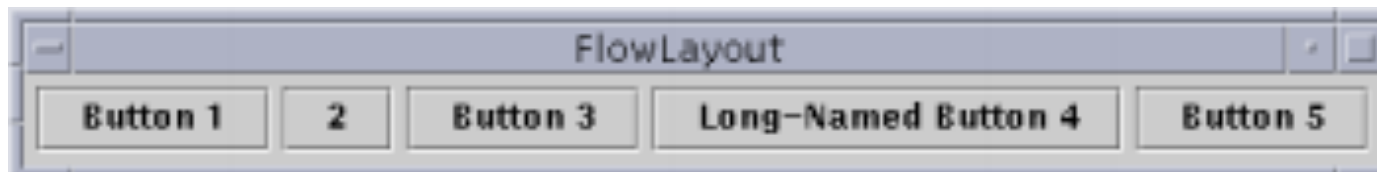
# Diagramadores

- Arrumam um grupo de elementos
- Estão associados aos containers
- Diferentes estilos de arrumação
  - Fluxo de texto
  - Orientado pelas bordas
  - Em forma de grade
  - ...



# java.awt.FlowLayout

- Dispõe os componentes lado a lado, uma linha após a outra
- Alinhamento: centralizado (default), à esquerda ou à direita
- Default para JPanel



# Exemplo - FlowLayout

```
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"));
```



# java.awt.BorderLayout

- Divide o container em 5 áreas: norte, sul, leste, oeste e centro
- Default para o content pane do JFrame



# Exemplo - BorderLayout

```
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);
```



# java.awt.GridLayout

- Células do mesmo tamanho especificadas pelo número de linhas e colunas



# Exemplo - GridLayout

```
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"));
```



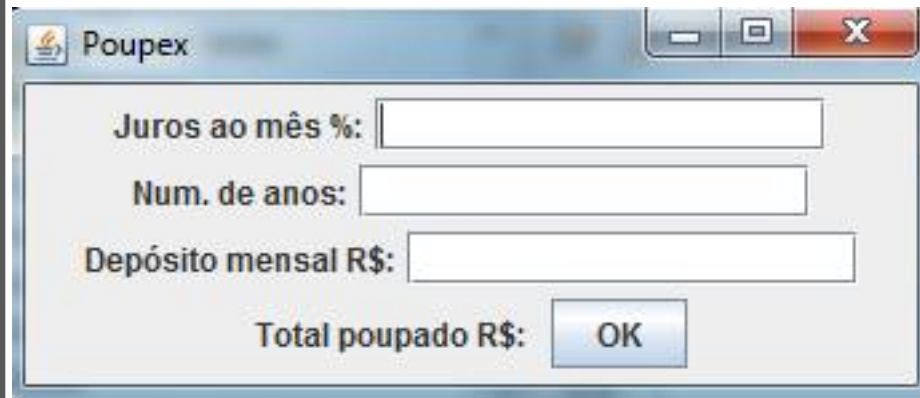
# Exercício

- Calculadora de Poupanças
  - Aplicativo para cálculo dos ganhos com aplicações em poupança
  - Para o cálculo, precisamos saber o período (em anos) da aplicação, o valor dos juros mensais e a quantidade que é depositada mensalmente na poupança.
- Realize esse exercício construindo as seguintes classes
  - Poupança: que possui os atributos desejados e o método de cálculo
  - InterfacePoupança: que possui a interface gráfica





# Exercício



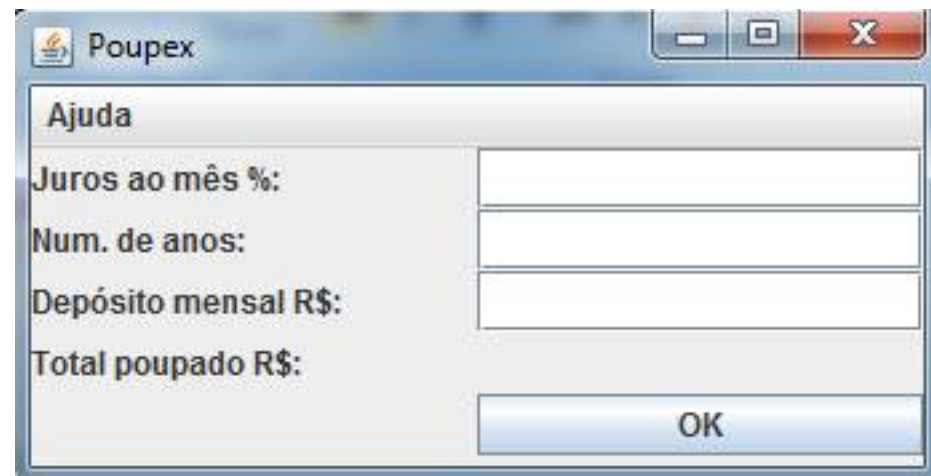
Poupex

Juros ao mês %:

Num. de anos:

Depósito mensal R\$:

Total poupado R\$:  OK



Poupex

Ajuda

Juros ao mês %:

Num. de anos:

Depósito mensal R\$:

Total poupado R\$:  OK

