Interface Gráfica do Usuário (GUI)

Prof. Grace

Interface Gráfica com Usuário (GUI)

- Componentes que possuem representação gráfica;
- Podem ser exibidos na tela e fornecem meios de interação com o usuário;
- A plataforma Java oferece recursos para construção de interfaces gráficas de usuário (GUI) a partir do framework Java Foundation Classes (JFC):
  - Java 2D: Criação de imagens e gráficos 2D
  - AWT: Componentes básicos para janelas Desktop. Base para Swing API.
  - Swing: Componentes sofisticados para construir uma interface gráfica de usuário

Container

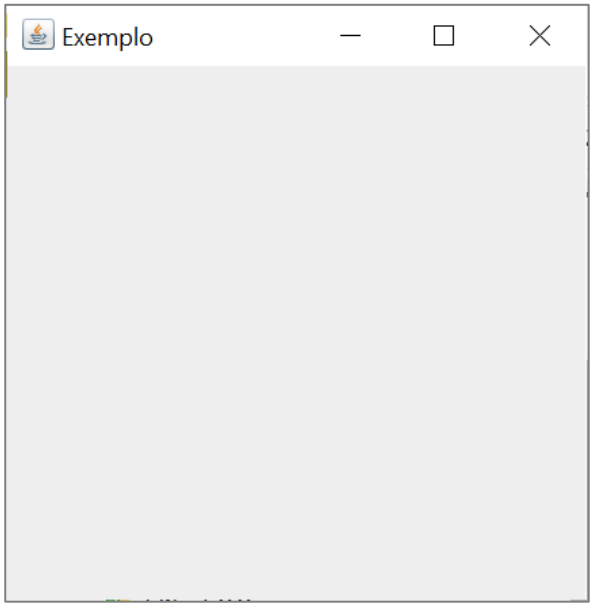
- Local em que são adicionados os componentes visualizados na tela;
- Principais containers:
  - Frame
  - Dialog
  - Applets

Container: Frames

- Janelas usadas para aplicações que construímos.
- Sintaxe:
  - JFrame fr = new JFrame(“Exemplo”);
  - fr.setSize(colunas, linhas);
  - fr.setVisible(true);

Exemplo Frame

```
import javax.swing.*;

public class TesteFrame {

    public static void main(String[] args) {
        JFrame fr = new JFrame("Exemplo");
        fr.setSize(300, 300);
        fr.setVisible(true);
    }
}
```



Container: Caixas de diálogo

- JOptionPane:
  - showMessageDialog(): informativo (só 1 botão)
  - showConfirmDialog(): possui opções padrão
  - showInputDialog(): entrada de dados
  - showOptionDialog(): com várias opções

Tipos de mensagem

- Define o ícone a ser apresentado:
  - ERROR_MESSAGE
  - INFORMATION_MESSAGE
  - WARNING_MESSAGE
  - QUESTION_MESSAGE
  - PLAIN_MESSAGE

Exemplo: Tipo MessageDialog

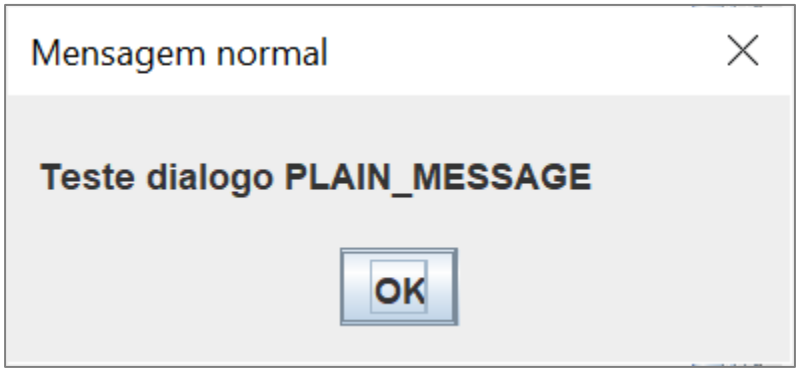
```
import javax.swing.*;

public class TesteShowMessage {

    public static void main(String[] args) {
        JOptionPane.showMessageDialog(null,
            "Teste dialogo PLAIN_MESSAGE",
            "Mensagem normal",
            JOptionPane.PLAIN_MESSAGE);
    }
}
```

Posição

Mensagem

Título

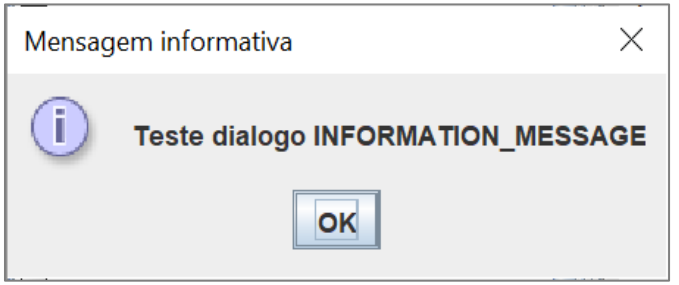
Sem ícone



Mensagens

```
JOptionPane.showMessageDialog(null,
    "Teste dialogo ERROR_MESSAGE",
    "Mensagem de erro",
    JOptionPane.ERROR_MESSAGE );
```

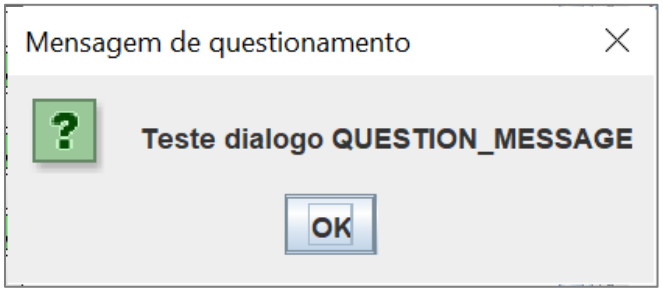

```
JOptionPane.showMessageDialog(null,
    "Teste dialogo INFORMATION_MESSAGE",
    "Mensagem informativa",
    JOptionPane.INFORMATION_MESSAGE );
```



Mensagens

```
JOptionPane.showMessageDialog(null,
    "Teste dialogo WARNING_MESSAGE",
    "Mensagem de alerta",
    JOptionPane.WARNING_MESSAGE );
```


```
JOptionPane.showMessageDialog(null,
    "Teste dialogo QUESTION_MESSAGE",
    "Mensagem de questionamento",
    JOptionPane.QUESTION_MESSAGE );
```



Caixa de diálogo: ConfirmDialog

- JoptionPane.showConfirmDialog:
  - DEFAULT_OPTION;
  - YES_NO_OPTION;
  - YES_NO_CANCEL_OPTION;
  - OK_CANCEL_OPTION.

Exemplo - ConfirmDialog

```
public class TesteConfirmDialog {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Exemplo");
        frame.setSize(300, 300);
        frame.setVisible(true);
        int op = JOptionPane.showConfirmDialog(frame,
                "Deseja finalizar a execução?",
                "Saída",
                JOptionPane.YES_NO_OPTION,
                JOptionPane.QUESTION_MESSAGE);
        if (op==0)
            JOptionPane.showMessageDialog(null,
                "Encerrando programa",
                "Saída",
                JOptionPane.PLAIN_MESSAGE);
        else
            JOptionPane.showMessageDialog(null,
                "Voltando ao programa",
                "Retorna",
                JOptionPane.PLAIN_MESSAGE);
    }
}
```

Alinhado ao frame

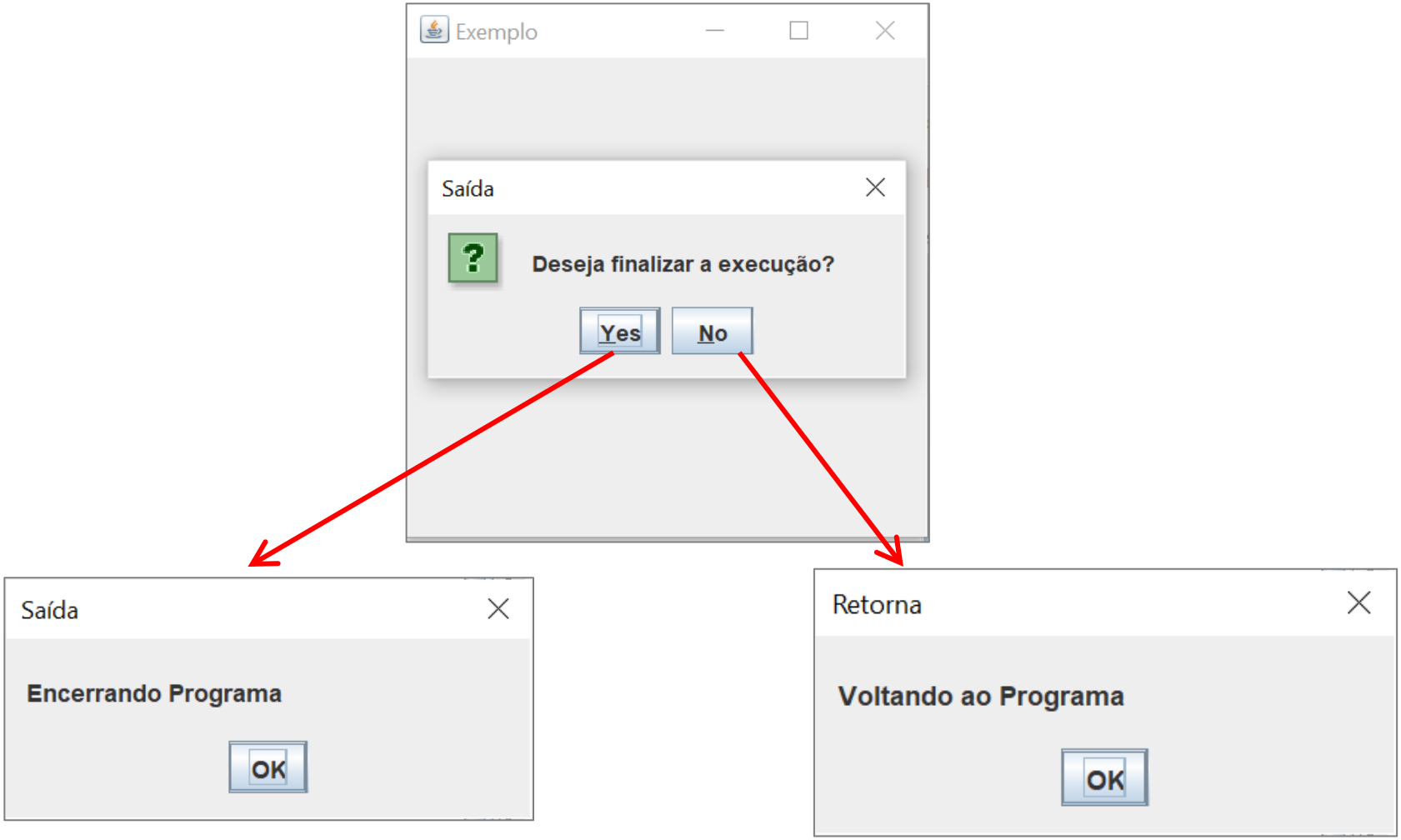
Mensagem

Título

Botões

Ícone interrogação

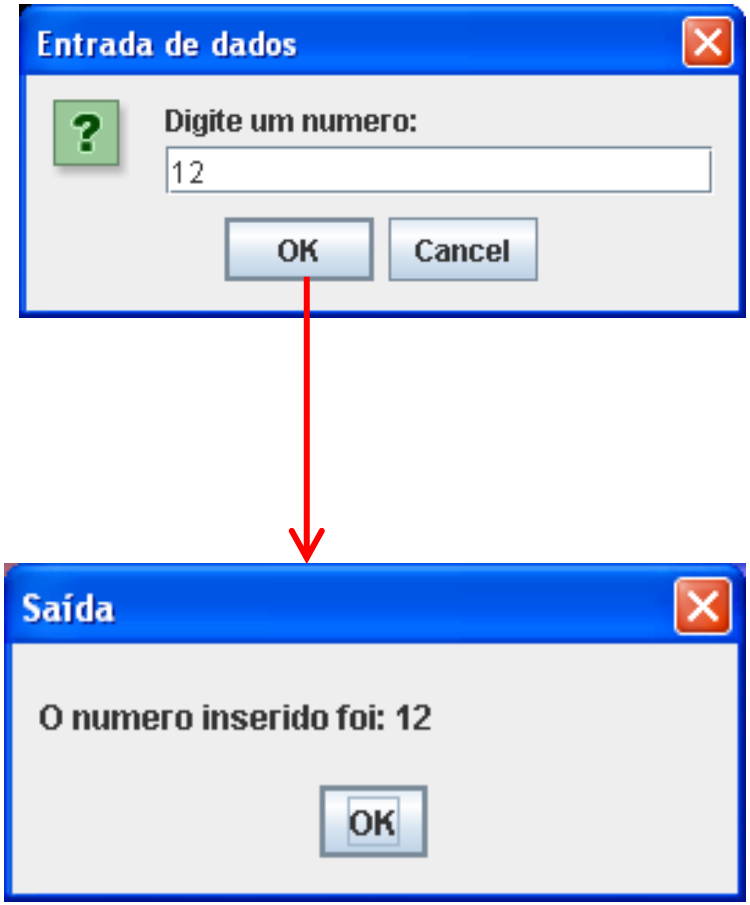
Exemplo executando



Caixa de diálogo: InputDialog

```
import javax.swing.*;

public class TesteInputDialog {

    public static void main(String[] args) {
        String s = JOptionPane.showInputDialog(null,
                "Digite um numero:",
                "Entrada de dados",
                JOptionPane.QUESTION_MESSAGE);

        JOptionPane.showMessageDialog(null,
                "O numero inserido foi: "+s,
                "Saída",
                JOptionPane.PLAIN_MESSAGE);
    }
}
```

Exemplo executando



Caixa de diálogo: OptionDialog

```
Object[] options = { "OK", "CANCELA" };
JOptionPane.showOptionDialog(null,
    "Clique OK para continuar",
    "Cuidado",
    JOptionPane.DEFAULT_OPTION,
    JOptionPane.WARNING_MESSAGE,
    null, options, options[0]);
```

Vetor com opções

Posição

Mensagem

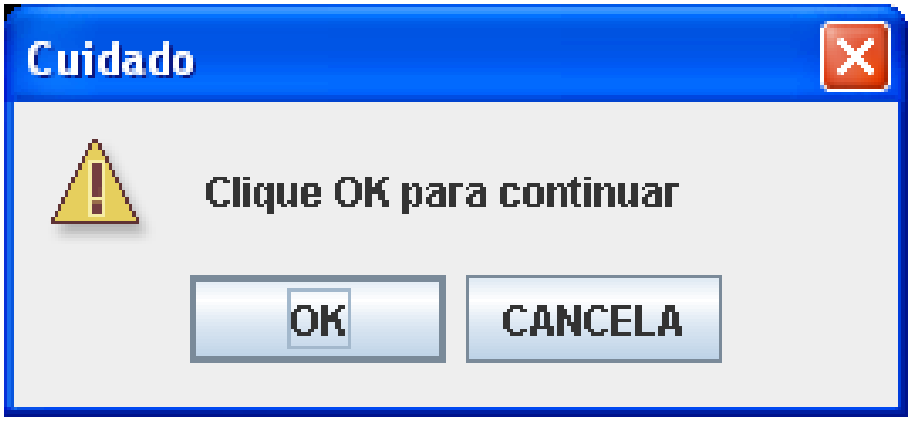
Título

Típo de opção

Icone cuidado!

Vetor opções, opção inicial

Outro ícone



Sugestão de Atividade

- Escreva um programa que use o inputDialog para ler dois valores inteiros e exiba uma caixa de diálogo com sua soma.
- Dica: use Integer.parseInt(s) para converter a String em inteiro;

Tratamento de Eventos

- Eventos: são acionados pelos programas com interface gráfica
- Eventos mais comuns
  - Gerados pelo teclado ou mouse
  - A partir de botões
  - A partir de menu

Event Listener

- É um objeto preparado para receber informações de eventos;
- A função de um *listener* é “escutar” eventos;
- Espera um evento ocorrer a partir de uma ou mais fontes;
- Para processar informações, precisamos implementar métodos (a partir de interfaces *listeners*);

Interfaces Java

- São modelos de comportamentos;
- Definem e padronizam serviços que uma classe pode oferecer;
- Implementar uma interface é como assinar um contrato!

- Ex: Interfaces gráficas (GUI) => Listener

Exemplo:
Encerrar aplicação ao fechar o Frame

```
import javax.swing.*;

public class TesteWindowListener {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Exemplo");
        frame.setSize(300, 300);
        frame.setVisible(true);

        TrataWindow tw = new TrataWindow();
        frame.addWindowListener(tw);
    }
}
```

tw = objeto do tipo eventListener

Classe TrataWindow: Implementando interface WindowListener

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

public class TrataWindow implements WindowListener {

    public void windowClosing(WindowEvent e) {
        System.exit(0);
    }
    public void windowIconified(WindowEvent e) {}
    public void windowOpened(WindowEvent e) {}
    public void windowClosed(WindowEvent e) {}
    public void windowDeiconified(WindowEvent e) {}
    public void windowActivated(WindowEvent e) {}
    public void windowDeactivated(WindowEvent e) {}
}
```

Outros Componentes GUI

- JLabel, JButton
- JTextField e JTextArea
- JScrollPane
- JCheckBox e JComboBox
- JList e JTable
- JMenuItem, JMenu e JMenuBar

Para adicionar componente a um container:

<container>.add(componente1);

Layout managers

- Auxiliam containers no posicionamento de seus componentes;
- Tipos:
  - FlowLayout
  - BorderLayout
  - GridLayout
  - GridBagLayout
  - CardLayout

FlowLayout

- Padrão para:
  - Panel
  - Jpanel
- Adiciona componentes da esquerda para direita até fim da linha;
- Então passa para a próxima linha, repetindo o processo;

BorderLayout

- Divide o container em 5 regiões:
  - Norte, Sul, leste, Oeste e Centro;
- Padrão para JFrames e JApplets;
- Permite apenas 1 componente para cada região do container;

GridLayout

- Divide o container como uma grade com tamanhos e espaços iguais entre os componentes
- Estes, por sua vez dividem-se em linhas e colunas do layout;

GridBagLayout

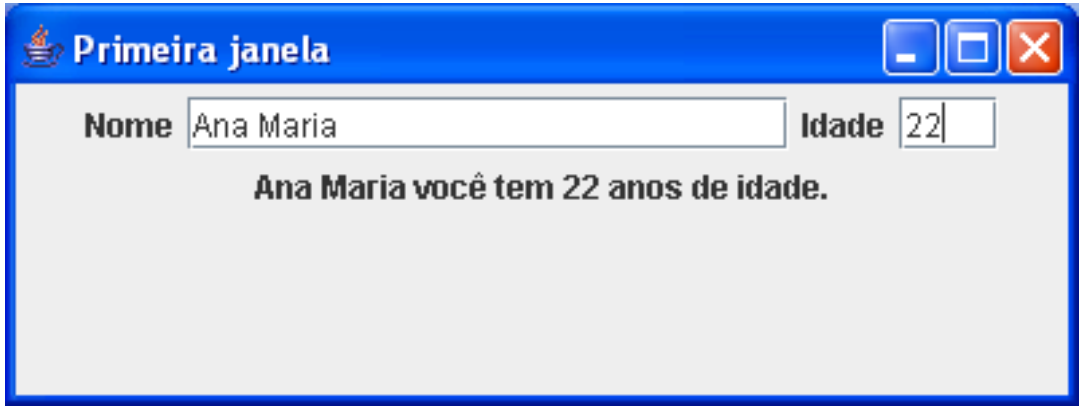
- Semelhante ao GridLayout, porém, permite divisões com tamanhos diferentes;

CardLayout

- Usado para exibir um componente de cada vez como em uma pilha de cartas;
- Somente o objeto que estiver no topo será visível;

Exemplo 01:
Listener no “Enter” da caixa de texto

- Fazer um programa que leia o nome e a idade de uma pessoa e exiba a mensagem: “Fulano” você tem XX anos de idade.”.



Exemplo 01

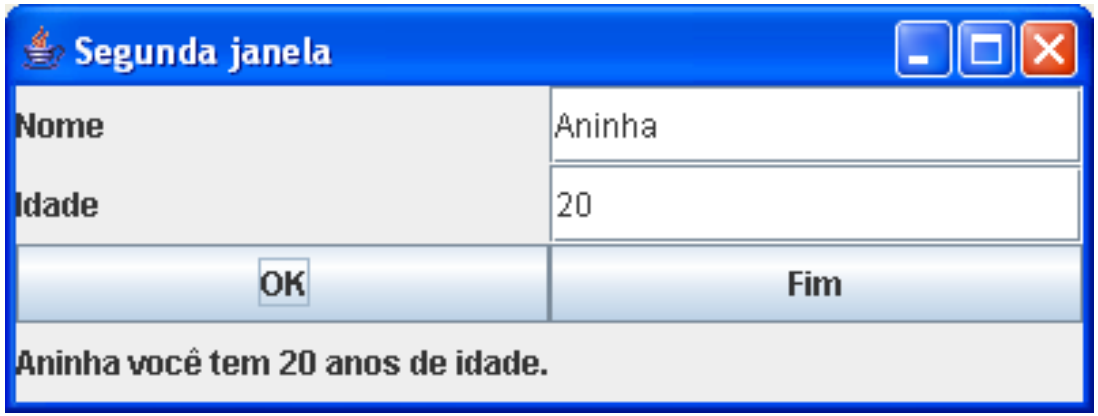
```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class JanelaEx1 extends JFrame
{
    //Inicialização dos objetos gráficos da janela
    private JLabel     lblNome, lblIdade, lblMensagem;
    private JTextField txtNome, txtIdade;
    //construtor
    public JanelaEx1()
    {
        super("Primeira janela");
        setLayout(new FlowLayout());
        //Instancia componentes
        lblNome     = new JLabel("Nome");
        lblIdade    = new JLabel("Idade");
        lblMensagem = new JLabel();
        txtNome   = new JTextField(20);
        txtIdade = new JTextField(3);
        //Adiciona cada componente ao frame
        add(lblNome);
        add(txtNome);
        add(lblIdade);
        add(txtIdade);
        add(lblMensagem);
        //Instancia e registra listener do TextField
        TrataTextField ttf = new TrataTextField();
        txtIdade.addActionListener(ttf);
    }
```

Listener e método main

```
    private class TrataTextField implements ActionListener
    {
        public void actionPerformed( ActionEvent e )
        {
            if (e.getSource() == txtIdade)
                lblMensagem.setText(txtNome.getText() + " você tem " +
                                    txtIdade.getText() + " anos de idade.");
            else
                System.exit(0);
        }
    }

    public static void main(String args[])
    {
        JanelaEx1 janela = new JanelaEx1();
        janela.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
        janela.setSize(400, 150);
        janela.setVisible( true );
    }
}
```

Exemplo 02:
Listener nos botões Ok e Fim

- Fazer um programa que leia o nome e a idade de uma pessoa usando botões para o tratamento do evento.



Exemplo 02

```
public class JanelaEx2 extends JFrame
{
    private JLabel     lblNome, lblIdade, lblMensagem;
    private JButton    btnOK, btnFim;
    private JTextField txtNome, txtIdade;
    //construtor
    public JanelaEx2()
    {
        super("Segunda janela");
        setLayout(new GridLayout(4,2));
        //Instancia componentes
        lblNome     = new JLabel("Nome");
        lblIdade    = new JLabel("Idade");
        lblMensagem = new JLabel();
        btnOK  = new JButton("OK");
        btnFim = new JButton("Fim");
        txtNome  = new JTextField(20);
        txtIdade = new JTextField(3);
        //adiciona cada componente ao frame
        add(lblNome);
        add(txtNome);
        add(lblIdade);
        add(txtIdade);
        add(btnOK);
        add(btnFim);
        add(lblMensagem);
        //Instancia e registra listener aos botoes
        TrataBtn tBtn = new TrataBtn();
        btnOK.addActionListener(tBtn);
        btnFim.addActionListener(tBtn);
    }
```

Listener e método main

```
    private class TrataBtn implements ActionListener
    {
        public void actionPerformed( ActionEvent e )
        {
            if (e.getSource() == btnOK)
                lblMensagem.setText(txtNome.getText() + " você tem " +
                                    txtIdade.getText() + " anos de idade.");
            else
                System.exit(0);
        }
    }

    public static void main(String args[])
    {
        JanelaEx2 janela = new JanelaEx2();
        janela.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
        janela.setSize(400, 150);
        janela.setVisible( true );
    }
}
```