Introdução a Interface Gráfica em Java

Por Jonas Nascimento e Jorge Victor

IFPB – Campus Campina Grande
Disciplina Programação Orientada a Objetos
Professor Katyusco – 3º Período

GUI (Graphical User Interface)

- Objeto no qual o usuário interage através de, por exemplo:
  - Mouse;
  - Teclado;
  - Alguma forma de entrada;
- Gera mais familiaridade pelo usuário.
- Tempo de aprendizado do programa pelo usuário é reduzido.

AWT e Swing

AWT (Abstract Windowing toolkit)

- Conjunto de ferramentas para interfaces gráficas do Java.
- Limitado em recursos devido a depender de suporte de cada plataforma para os componentes oferecidos.
- Bugs e incompatibilidades entre plataformas.

Swing

- Coleção de componentes que oferece uma interface muito mais rica.
- Muitas vezes importa-se java.awt e javax.swing para usar o Swing.

- JDK 1.0
  - Interface que roda de forma medíocre em todas as plataformas
  - Modelo de eventos arcaico
- JDK 1.1
  - Melhora do modelo de eventos.
- JDK 1.2
  - Nova biblioteca feita do zero.
  - Mantém e estende a interface de eventos e layout

Biblioteca Java Swing

javax.swing.*

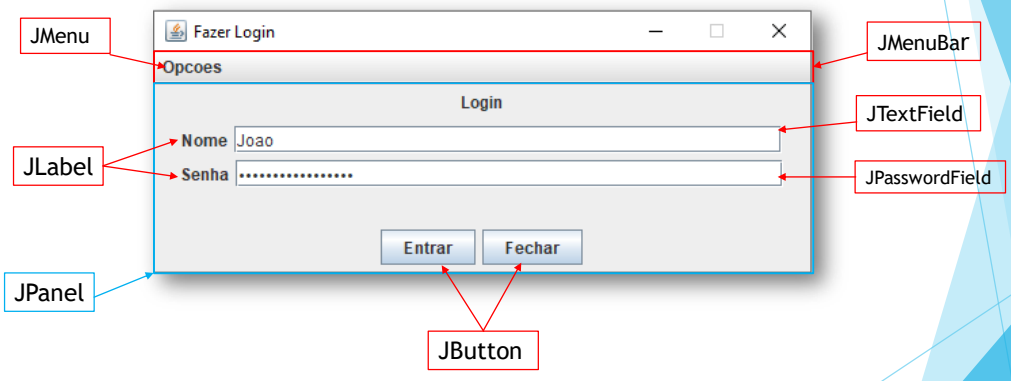
Porque swing e não AWT?

- AWT é a versão mais velha e mais primitiva.
- Swing é mais poderosa.
- Swing pode ser enviado com as aplicações, sendo não dependente de máquina

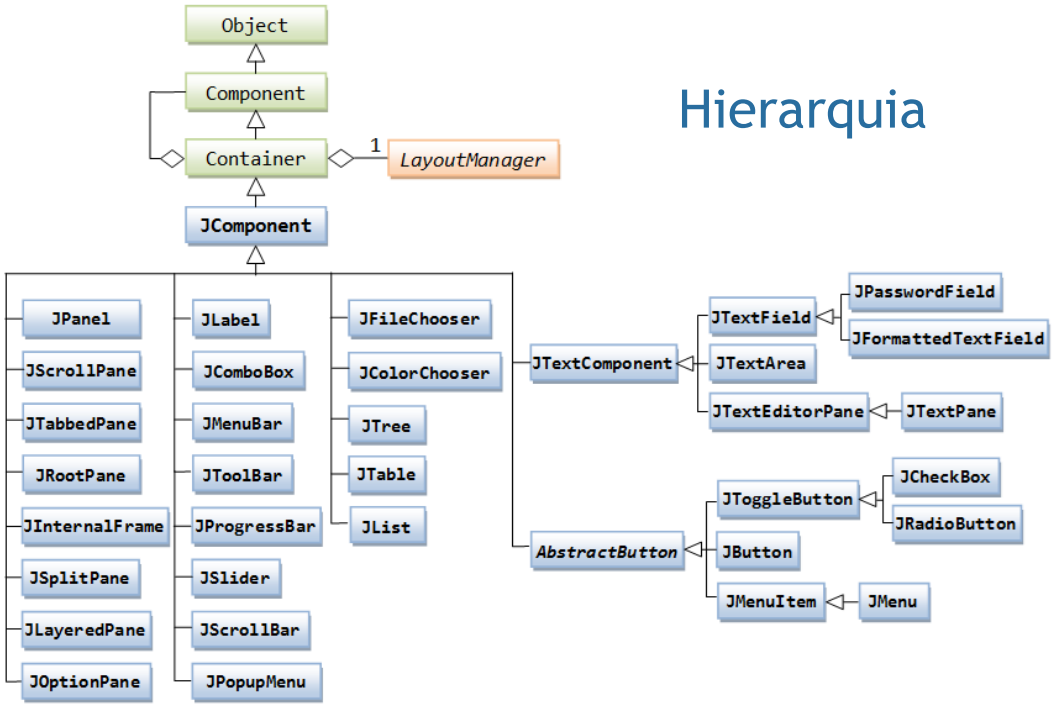
Elementos de uma GUI usando Swing

- **Componentes:** Elementos desenhados na tela.
  - Ex: botão, textbox, label, etc.
- **Containers:** Elementos que servem como agrupadores lógicos para componentes.
  - Ex: Panel.
- **Containers de alto nível:** Suporte que os componentes swing necessitam para realizar o desenho na tela e o tratamento de eventos.
  - Ex: Frames , Diálogos.

Elementos de uma GUI usando Swing



Hierarquia

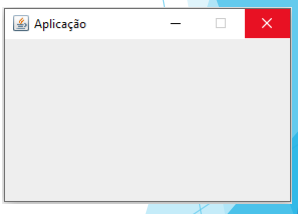


Hierarquia

```
java.lang.Object
  +--java.awt.Component
     +--java.awt.Container
     +--javax.swing.JComponent
         | +--javax.swing.JButton
         | +--javax.swing.JLabel
         | +--javax.swing.JMenuBar
         | +--javax.swing.JOptionPane
         | +--javax.swing.JPanel
         | +--javax.swing.JTextField
         |
        +--java.awt.Window
           +--java.awt.Frame
              +--javax.swing.JFrame
```

Classe JFrame

- Componente swing que cria janelas para a aplicação com atributos e elementos básicos de uma janela.
  - Barra de Títulos
  - Botões de Minimizar, Maximizar, e Fechar a janela.
- Acessórios totalmente dependentes de plataforma.
- Toda aplicação com uma GUI usa pelo menos um frame.

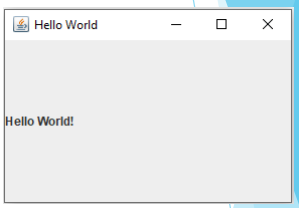


Métodos essenciais da classe JFrame

- setTitle(String title):
  - Coloca um titulo na barra de titulo da janela
- setDefaultCloseOperation(int operation):
  - Define o que acontece quando clicar no botão X de fechar da janela.
  - Mais usado: JFrame.EXIT_ON_CLOSE
- setSize(int x , int y):
  - Define o tamanho da janela.
- setVisible(boolean v):
  - Faz com que o frame se torne visível(true) ou não(false).

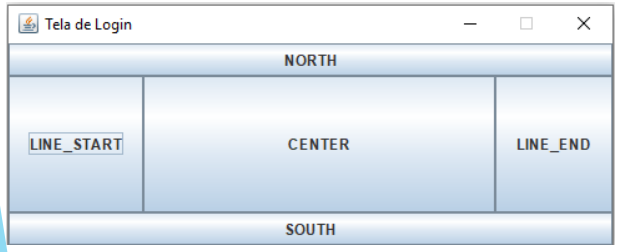
Programa básico em Java Swing

```
import javax.swing.*;

public class Run {
    public static void main(String[] args) {
        JFrame janela = new JFrame("Hello World");

        janela.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        janela.setSize(300,200);
        janela.setVisible(true);

        JLabel hello = new JLabel("Hello World!");
        janela.add(hello);
    }
}
```

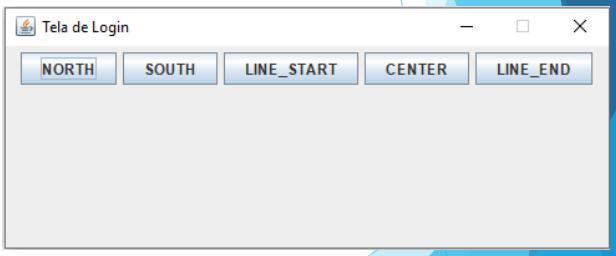


Gerenciadores de Layout

- Cada contêiner individual pode ter apenas um gerenciador de layout, mas vários contêineres no mesmo aplicativo podem utilizar cada um gerenciador de layout.

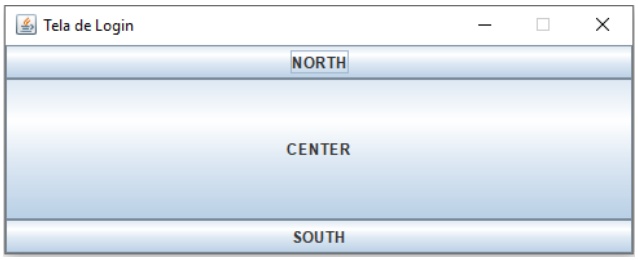


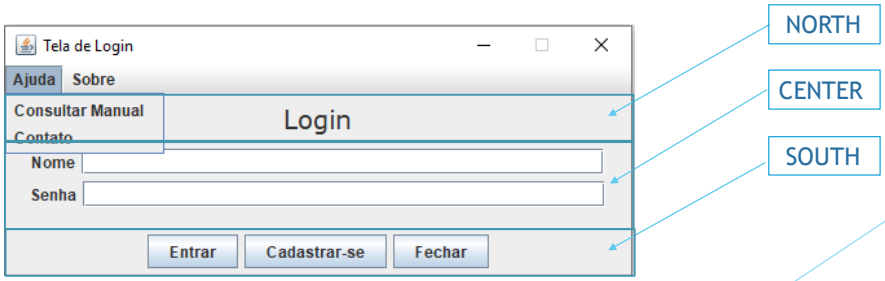
BorderLayout



FlowLayout

Gerenciadores de Layout





Estrutura de um JFrame.

- Cada Objeto que herda da classe JFrame será uma janela.
- Todos os Componentes que forem ser adicionados na janela estarão nessa classe

loginFrame.java

```
Import javax.swing.*;

public class loginFrame extends JFrame{
        loginFrame(){
                super("Tela de Login");
                init();
                setVisible(true);
        }
        private void init(){
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setSize(300,200);
        }
}
```

Run.java

```
Import javax.swing.*;

class Run{
        public static void main(String[] args){
                loginFrame login = new loginFrame();
        }
}
```

JPanel

- Container que é capaz de organizar um grupo de componentes.
- Vários layouts podem ser definidos para um JPanel. Padrão: FlowLayout.

Construtores:

- JPanel() : Cria um novo painel com FlowLayout.
- JPanel(LayoutManager l) : Cria um novo painel com um layout especifico.

Métodos mais usados:

- add(Component c): Adiciona componentes a um container especifico.
- setLayout(LayoutManager l): Define o Layout de um container especifico.

JLabel (rótulos)

- Usado para mostrar uma pequena String ou um icone, ou os dois
- Não pode ser atribuído nenhum evento. Ex: cliques do mouse , ou entrada no teclado.

Construtores:

- JLabel() : Cria um rótulo em branco sem texto nem imagem.
- JLabel(String s): Cria um novo rótulo com um texto específico.
- JLabel(Icon i): Cria um novo rótulo com uma imagem.
- JLabel(String s, Icon i, int align) : Cria um novo rótulo com uma String, uma imagem e um alinhamento (horizontal ou vertical).

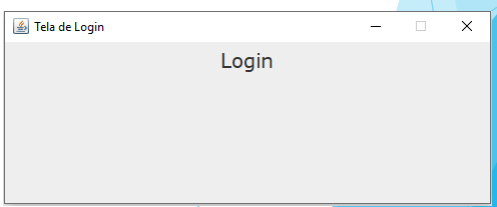
Métodos mais usados:

- setText(String s): Define o texto do rótulo.
- setIcon(Icon i) : Define o Icone do rótulo.

JPanel, JLabel, Layout

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

public class loginFrame extends JFrame{
    loginFrame(){
        super("Tela de Login");
        init();
        criarComponentes();
        setVisible(true);
    }
    private void init(){
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(500,200);
        setResizable(false);
        setLocationRelativeTo(null);
    }
```

```
private void criarComponentes(){
    setLayout(new BorderLayout());

    JPanel painelTitulos = new JPanel();
    JPanel painelLogin = new JPanel();
    JPanel paineBotoes = new JPanel();

    JLabel tituloText = new JLabel("Login");
    tituloText.setFont(new Font("Verdana",Font.PLAIN,20));
    painelTitulos.add(tituloText);
    add( painelTitulos, BorderLayout.NORTH );
```



JTextField

- Componente que permite editar uma linha de texto.

Construtores mais usados:

- JTextField() : Cria um novo campo de texto vazio sem tamanho.
- JTextField(String text): Cria um novo campo de texto inicializado com a String dada.
- JTextField(int columns): Cria um novo campo de texto vazio com um tamanho especifico de colunas.
- JTextField(String text, int columns): Cria um novo campo de texto vazio com uma String dada e um numero especifico de colunas.

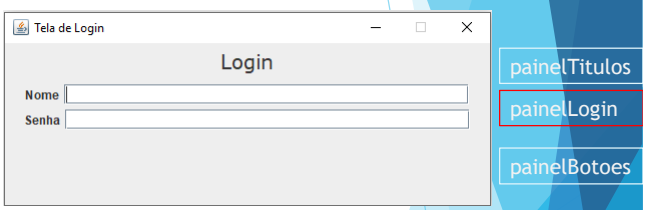
Alguns métodos são:

- setColumns(int n): Define o numero de colunas de um campo de texto. (o tamanho de um TextField é medido em colunas.
- setFont(Font f) : Define a fonte do texto mostrado no campo de texto.
- addActionListener(ActionListener l): define um ActionListener para o campo de texto.

JTextField, JPasswordField

Ainda no método criarComponentes()

```
private void criarComponentes(){
    ........

    JLabel nomeText = new JLabel("Nome");
    painelLogin.add(nomeText);

    JTextField nomeField = new JTextField(40);
    painelLogin.add(nomeField);

    JLabel senhaText = new JLabel("Senha");
    painelLogin.add(senhaText);

    JPasswordField senhaField = new JPasswordField(40);
    painelLogin.add(senhaField);

    add(painelLogin, BorderLayout.CENTER);
    ........
```



JButton

- Componente em que o usuário pode interagir
- Quando clicado gera um evento.
- Pode conter um texto ou um icone, ou ambos.

Construtores:

- JButton(): Cria um botão sem texto ou icone definidos.
- JButton(String text): Cria um botão com um texto.
- JButton(String text, Icon icon): Cria um botão com um texto inicial e um icone.

Método setActionListener(ActionListener a): Método que captura um evento.

JButton

Ainda no método criarComponentes()

```
private void criarComponentes(){
        ........

        JButton entrarBotao = new JButton("Entrar");
        painelBotoes.add(entrarBotao);

        JButton cadastrarBotao = new JButton("Cadastrar-se");
        painelBotoes.add(cadastrarBotao);

        JButton fecharBotao = new JButton("Fechar");
        painelBotoes.add(fecharBotao);

        add(painelBotoes, BorderLayout.SOUTH);

}
```

Fim do método.

Tratar Eventos em Java Swing

- Interface ActionListener: java.awt.event.ActionListener

```
public interface ActionListener extends EventListener{
        void actionPerformed(ActionEvent e);
}
```

- Interface que recebe e processa eventos.
- Toda classe que é destinada a processar um evento implementa essa classe.
- Usando o método de componentes: addActionListener, a classe que implementa ActionListener pode ser associada ao componente
- Implementa o método actionPerformed que é chamado quando ocorre alguma ação.

Tratar Eventos em Java Swing

- Criar classe que implementa a interface ActionListener e sobrescrever o método actionPerformed(ActionEvent e).
- Associar a classe ao componente por meio do método:
  Componente.addActionListener(ActionListener a);

```
private class fecharAction implements ActionListener{
      @Override
      public void actionPerformed(ActionEvent e) {
         System.exit(1);
      }
   }
```

Fecha toda a aplicação

```
private void criarComponentes(){
        ……..

        JButton entrarBotao = new JButton("Entrar");
        painelBotoes.add(entrarBotao);

        JButton cadastrarBotao = new JButton("Cadastrar-se");
        painelBotoes.add(cadastrarBotao);

        JButton fecharBotao = new JButton("Fechar");
        fecharBotao.addActionListener(new fecharAction());
        painelBotoes.add(fecharBotao);

        add(painelBotoes, BorderLayout.SOUTH);

}
```

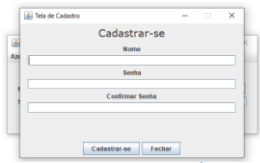
Tratar Eventos em Java Swing

Outros exemplos

```
private void criarComponentes(){
        ……..

        JButton entrarBotao = new JButton("Entrar");
        painelBotoes.add(entrarBotao);

        JButton cadastrarBotao = new JButton("Cadastrar-se");
        cadastrarBotao.addActionListener(new cadastroAction());
        painelBotoes.add(cadastrarBotao);

        JButton fecharBotao = new JButton("Fechar");
        fecharBotao.addActionListener(new fecharAction());
        painelBotoes.add(fecharBotao);

        add(painelBotoes, BorderLayout.SOUTH);

}
```

```
private class cadastroAction implements ActionListener{
    @Override
    public void actionPerformed(ActionEvent e) {
        cadastroFrame cadastroFrame = new cadastroFrame();
    }
}
```

Abre outra janela

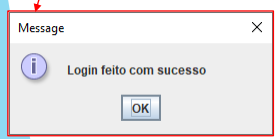


Tratar Eventos em Java Swing

Outros exemplos

```
private void criarComponentes(){
        ……..

        JButton entrarBotao = new JButton("Entrar");
        entrarBotao.addActionListener(new loginAction());
        painelBotoes.add(entrarBotao);

        trarBotao = new JButton("Cadastrar-se");
        o.addActionListener(new cadastroAction());
        add(cadastrarBotao);

        rBotao = new JButton("Fechar");
        ddActionListener(new fecharAction());
        add(fecharBotao);

        add(painelBotoes, BorderLayout.SOUTH);

}
```

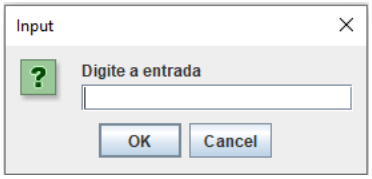
```
private class loginAction implements ActionListener{
    @Override
    public void actionPerformed(ActionEvent e) {
        JOptionPane.showInternalMessageDialog(null, "Login feito com sucesso");

        aplicaçãoFrame app = new aplicaçãoFrame();

        loginFrame.this.dispose();
    }
}
```



Fecha somente a janela, sem fechar a aplicação.

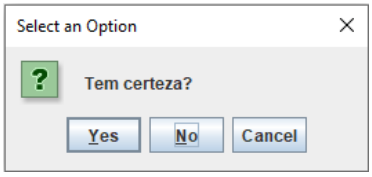
JOptionPane

- Caixas de Diálogo.
- Classes prontas.
- Interfaces mais simples.
- Uma linha de código.

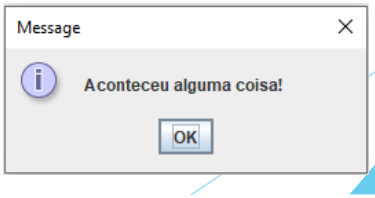



```
showInputDialog(null,"Digite a entrada", "Input");
```
Componente pai — Mensagem — Titulo

```
showConfirmDialog(new loginFrame(),"Tem certeza?");
```
Componente pai — Mensagem




```
showMessageDialog(null, "Aconteceu alguma coisa!");
```
Componente pai — Mensagem



JMenuBar

- Barra de menus do Java Swing.
- Adicionado ao JFrame pelo método setJMenuBar(JMenuBar m).

Construtor:

- JMenuBar(): inicializa a barra de menus.

JMenu

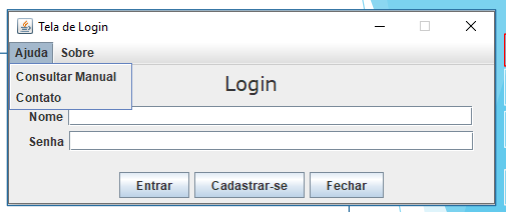
- Componente que é adicionado à barra de menus através do método add(Jmenu m).

Construtores:

- JMenu(): Constrói um novo menu sem texto.
- JMenu(String s) : Constrói um novo menu com texto como seu titulo.
- JMenu(String s, boolean b): Constói um novo menu com texto como seu titulo e com um valor boleano que diz se o menu vai poder ser movido pela tela(true) ou não(false).

JMenuBar, JMenu, JMenuItem

```
private void criarMenu(){
        JMenuBar menuBar = new JMenuBar();
        setJMenuBar(menuBar);

        JMenu ajudaMenu = new JMenu("Ajuda");
        menuBar.add(ajudaMenu);

        JMenu sobreMenu = new JMenu("Sobre");
        menuBar.add(sobreMenu);

        JMenuItem consultarManual_ajuda = new JMenuItem("Consultar Manual");
        ajudaMenu.add(consultarManual_ajuda);

        JMenuItem contato_ajuda = new JMenuItem("Contato");
        ajudaMenu.add(contato_ajuda);

   }
```



Menus
painelTitulos
painelLogin
painelBotoes

Fim.

Referencias

- Geeks for Geeks. https://www.geeksforgeeks.org/
- Java API. https://docs.oracle.com/javase/7/docs/api/
- LNCC. https://www.lncc.br/~rogerio/poo/04a%20-%20Programacao_GUI.pdf
- Yuri Lacerda. https://www.youtube.com/watch?v=p9mZHeWVsAg&t=1536s