JAVA SWING E PRINCÍPIOS DE PROGRAMAÇÃO ORIENTADA A OBJETOS

PROF. ME. HÉLIO ESPERIDIÃO

INTERFACES GRÁFICAS

- A criação de interfaces gráficas, também chamado de Graphical User Interface (GUI), é um processo que necessita de grande dedicação, pois por meio delas o usuário percebe e interage com o sistema.

- Todo o processo de criação de uma interface é necessário seguir as orientações de boas práticas na utilização de componentes, inclusive quanto ao posicionamento e agrupamento.

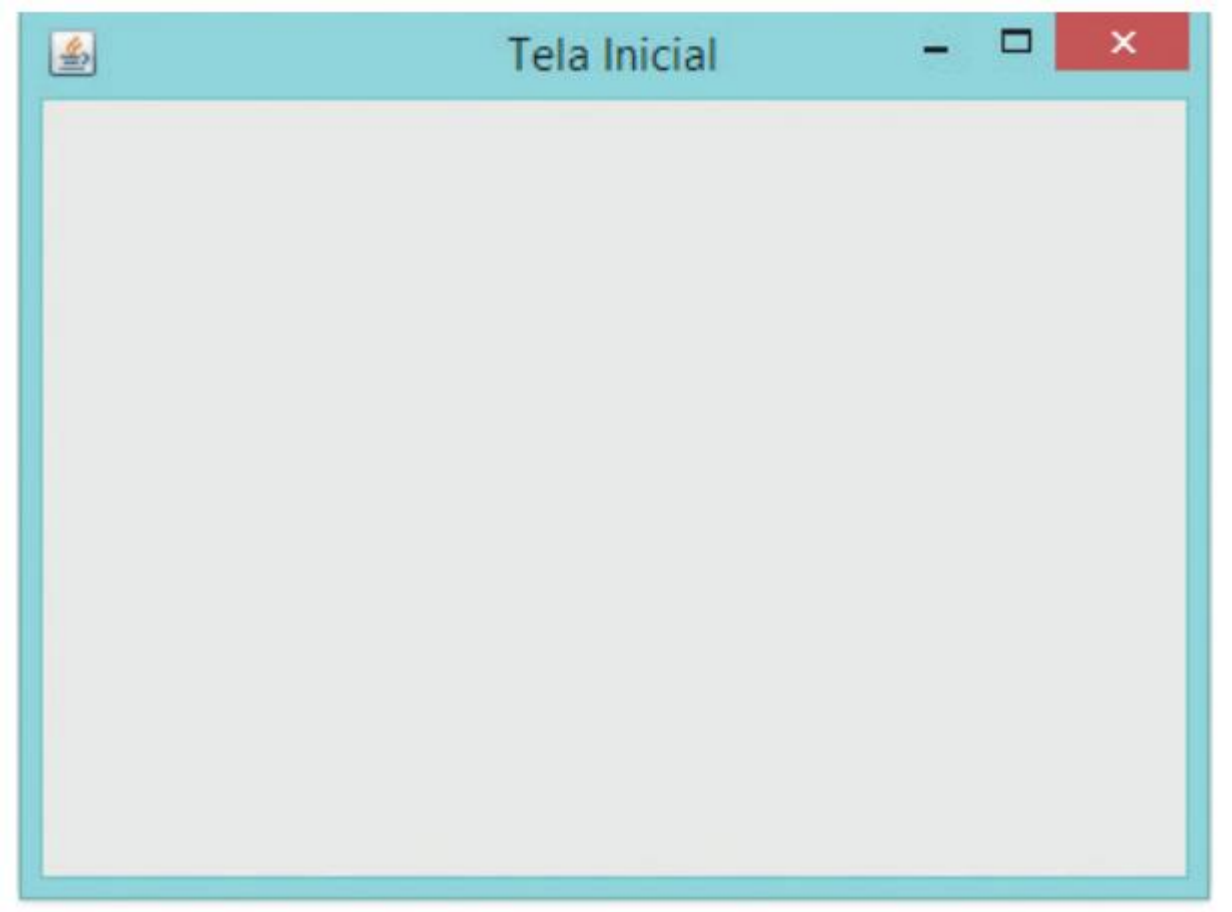
A CONSTRUÇÃO

- A construção de interfaces gráficas é feita a partir da inserção de componentes em uma tela, como por exemplo, botões, caixas de seleção e a própria representação da tela, dentre outros. Todos esses componentes estão disponíveis dentro de bibliotecas específicas

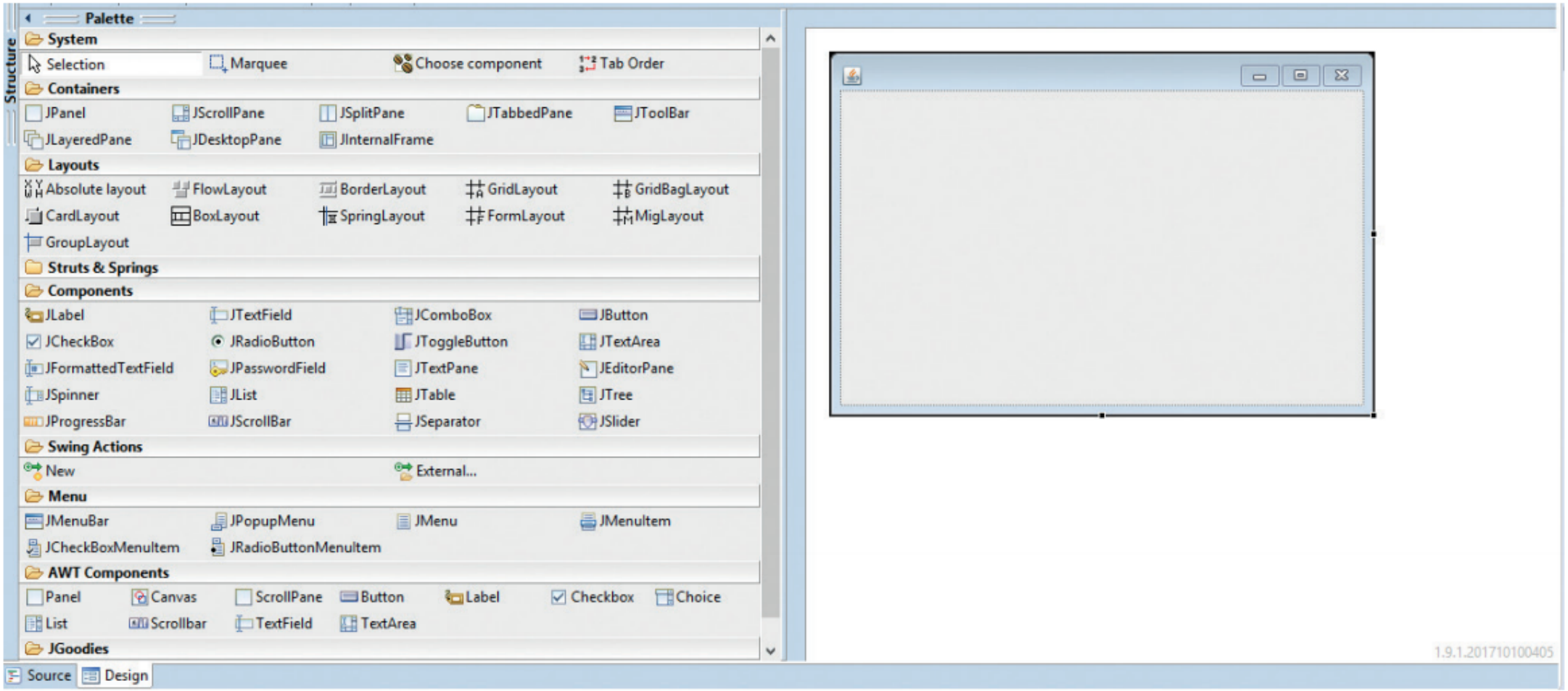
JAVA SWING

- É uma biblioteca que vem incorporada no Java Development Kit, dispondo de diversos elementos para a produção de telas.

EXEMPLO DE FRAME OU TELA UTILIZANDO O SWING DO JAVA



EDITOR PARA INTERFACES GRÁFICAS ACOPLADO AO ECLIPSE



```
package view;
import javax.swing.JFrame;

public class InterfaceGrafica extends JFrame{
   public InterfaceGrafica(){
        setSize(400,500);
        setTitle("Tela Inicial");
        setVisible(true);
   }
   public static void main(String[] args) {
        InterfaceGrafica telaInicial = new Interfa
ceGrafica();
   }
}
```

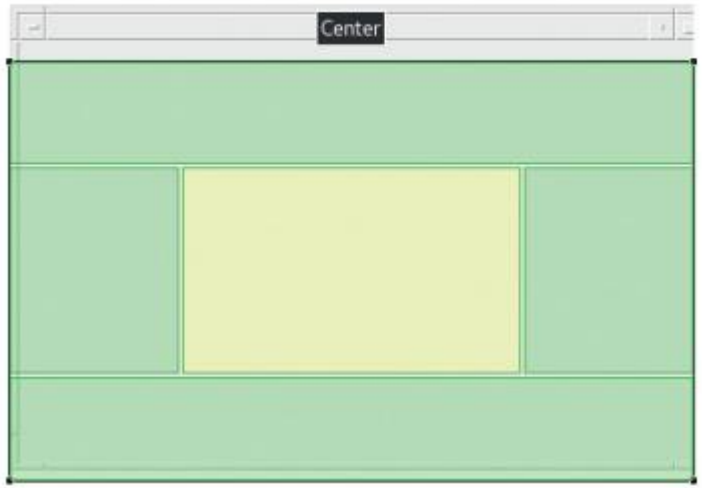
```java
package view;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class PrimeiraTela extends JFrame {
        private JLabel lblNome;
        private JTextField txtNome;

public PrimeiraTela() {
        lblNome = new JLabel("Nome");
        txtNome = new JTextField();

        setSize(400,200);
        setTitle("Tela Inicial");
        setVisible(true);
        setLayout(null);

        lblNome.setBounds(10,10,100,25);
        txtNome.setBounds(50,10,200,25);
        getContentPane().add(lblNome);
        getContentPane().add(txtNome);
    }
   public static void main(String[] args) {
        PrimeiraTela t1 = new PrimeiraTela();
    }
}
```

```
setBounds(Rectangle r)                                  void
setBounds(int x, int y, int width, int height) void
Membros da Instância; Pressione 'Ctrl+SPACE' Novamente para Todos os Itens
```

Componente	Descrição
JButton	Objeto usado para criar botões.
JCheckBox	Objeto usado para oferecer uma opção para o usuário. Normalmente é representado por uma caixa de seleção, que quando está com "check" representa "sim" e quando está com "não" representa "não".
JComboBox	Objeto usado para oferecer mais de uma opção para o usuário em forma de lista *drop-down*. Para cada lista somente um item pode ser selecionado.
JList	Objeto usado para oferecer mais de uma opção para o usuário, mas, diferentemente do JComboBox, esse componente permite a seleção de mais que uma opção.
JPanel	Objeto usado para organizar diversos componentes.

BORDERLAYOUT

Divisão da classe `BorderLayout` do Java Swing



- NORTH, SOUTH, EAST, WEST e CENTER

```
setLayout(new BorderLayout());
add(new Button("Sim"), BorderLayout. SOUTH);
```

JFORMATTEDTEXTFIELD

- MaskFormatter (“###.###.###-##”)

Segundo Deitel e Deitel (2016), para a máscara do objeto `JFormattedTextField` é possível utilizar os seguintes codificadores:

- **#** para representar a entrada de números.
- **U** para letras em caixa alta.
- **L** para letras em caixa baixa.
- **A** para qualquer número ou letra.
- **?** para qualquer caractere.
- ***** para qualquer elemento.
- **H** para entrada em hexadecimal.

```
import java.awt.Container; //biblioteca para
containers
import java.text.ParseException;
import javax.swing.*; //simplificando a
inclusão de bibliotecas
import javax.swing.text.MaskFormatter;


public class PrimeiraTela extends JFrame {


private JLabel lblNome;
private JTextField txtNome;
private JLabel lblCPF;
private JFormattedTextField txtCPF;
private JLabel lblTipo;
private JComboBox cmbTipo;
private final String[] tiposUsuarios =
{"Adminstrador","Geral"};
private JButton btnOK;
private Container ctn;
```

```
public PrimeiraTela(){
        setSize(400,300);
        setTitle("Tela Inicial");
        ctn = getContentPane();
        lblNome = new JLabel("Nome");
        txtNome = new JTextField();
        lblCPF = new JLabel("CPF");
        try {
            txtCPF = new JFormattedTextField(new
            MaskFormatter("###.###.###-##"));
```

```
    } catch (ParseException e) {
            e.printStackTrace();
    }
    lblTipo = new JLabel("Tipo de usuário"
    cmbTipo = new JComboBox(tiposUsuarios)
    btnOK = new JButton("Enviar");
    ctn.setLayout(null);
    lblNome.setBounds(0, 0, 100, 25);
    txtNome.setBounds(150,0,200,25);
    lblCPF.setBounds(0,50,100,25);
    txtCPF.setBounds(150,50,200,25);
    lblTipo.setBounds(0,100,200,25);
    cmbTipo.setBounds(150,100,200,25);
    btnOK.setBounds(150, 150, 100, 100);
```

```
        ctn.add(lblNome);
        ctn.add(txtNome);
        ctn.add(lblCPF);
        ctn.add(txtCPF);
        ctn.add(lblTipo);
        ctn.add(cmbTipo);
        ctn.add(btnOK);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_
        ON_CLOSE);
}
```

```
    public static void main(String[] args) {
            PrimeiraTela t1 = new PrimeiraTela();
    }
}
```

JOPTIONPANE
CAIXA DE MENSAGEM SIMPLES

```
JOptionPane.showInputDialog("Qual é o seu nome?");
```

SHOWINPUTDIALOG

```
String nome = null;
nome = JOptionPane.showInputDialog("Qual é o seu nome?");
```

SHOWCONFIRMDIALOG

```
int resposta = JOptionPane.showConfirmDialog(null, "Deseja realmente excluir?");
```

INTERFACE GRÁFICA X INTERFACE

- Para evitar confusões na definição dos termos, chamaremos de interface os elementos da orientação a objetos que descrevem um conjunto de métodos, e de interface gráfica os elementos como botões, janelas, campos e outros. Ao se construir uma interface gráfica usando Java Swing, acrescentamos as interfaces para definir qual método será utilizado no momento de chamada que um evento é iniciado

INTERFACE

```
public interface AcessoElementos {
    public int getElemento (int index);
    public void setElemento (int index);
}
```

USO DE UMA INTERFACE

```
import javax.swing.*;
public class Aluno extends JFrame implements AcessoElementos{
        @Override
        public int getElemento(int index) {
                return 0;
        }
        @Override
        public void setElemento(int index) {
         }
}
```

TRATAMENTO DE EVENTOS

Evento	Descrição
`ActionListener`	Evento gerado pelo clique com o mouse em um botão.
`ItemListener`	Evento gerado quando um item em uma lista é selecionado.
`FocusListener`	Evento gerado quando um elemento ganha ou perde foco.
`WindowListener`	Evento gerado quando ocorre uma mudança na janela.

```
import java.awt.Container;
import javax.swing.*;
public class PrimeiraTela extends JFrame{
        private JButton btnok;
        private JTextField txtNome;
        private JLabel lblNome;
        private Container ctn;
        public PrimeiraTela() {
                setSize(300,140);
                setTitle("Eventos em Java Swing");
```

```
        ctn = getContentPane();
        ctn.setLayout(null);
        btnok = new JButton("Enviar");
        lblNome = new JLabel("Nome");
        txtNome = new JTextField();
        lblNome.setBounds(10,10,100,25);
        txtNome.setBounds(70,10,200,25);
        btnok.setBounds(90,50,80,40);
        ctn.add(lblNome);
        ctn.add(txtNome);
        ctn.add(btnok);


}
```

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

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class PrimeiraTela Tela extends JFrame implements ActionListener{
        btnok.addActionListener(this);
        @Override
        public void actionPerformed(ActionEvent e)  {
                // caso seja necessário tratar eventos de mais de um botão
                if(e.getActionCommand().equals("Enviar")) {
                        txtNome.setText(“Botão clicado”);
                }
        }
}
```

UTILIZANDO CLASSE ANÔNIMA

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class Tela extends JFrame {
        public Tela() {
                btnok.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                trataBotaoOk();
                        }
                });
        }
        public void trataBotaoOk() {
                txtNome.setText("Botão clicado");
        }
}
```

TRATAR EVENTO DE MUDANÇA DE TEXTO EM UM JTEXTFIELD

```
import javax.swing.event.DocumentEvent;
import javax.swing.event.DocumentListener;
        txtNome.getDocument().addDocumentListener(new DocumentListener() {
                public void removeUpdate(DocumentEvent e) {
                                // ações quando texto for apagado }
                public void insertUpdate(DocumentEvent e) {
                // ações quando texto for inserido
                }
                public void changedUpdate(DocumentEvent e) {
                // ações quando texto for alterado
                }
        });
```

TRATAMENTO DE SELEÇÃO DE UM JCOMBOBOX

```
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
        jmbTipos.addItemListener(new ItemListener() {
                public void itemStateChanged(ItemEvent e) {
                        if (e.getStateChange() == ItemEvent. SELECTED) {
                                trataJmbTipos();
                                }
                        }
                });
        public void trataJmbTipos() {
                JOptionPane.showMessageDialog(this, "Item selecionado: "+ jmbTipos.
getSelectedItem());
        }
```

EVENTO DE PERDA OU GANHO DE FOCO SOB UM COMPONENTE

```
txtNome.addFocusListener(new FocusListener() {
    public void focusLost(FocusEvent e) {

     }
        public void focusGained(FocusEvent e) {
        }
});
```