Principais Componentes Swing

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
PROF. VITOR YANO

Principais componentes Swing

- **JFrame** – janela com barra de título, ícone, botões de comando.
- Construtor:
  - JFrame() – constrói a janela invisível e sem título
  - JFrame(String) – constrói a janela invisível e atribui o título
- Alguns métodos importantes:
  - pack() – compacta a janela para o tamanho dos componentes
  - setSize(int, int) – define a largura e altura da janela
  - setLocation(int, int) – define a posição da janela na tela (x,y)
  - setBounds(int, int, int, int) – define posição e tamanho
  - setVisible(boolean) – exibe a janela

JFrame

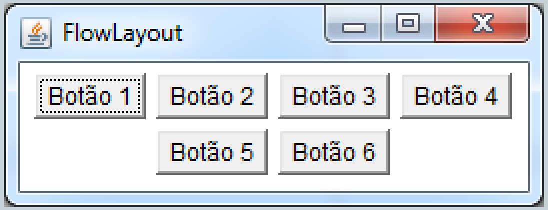
- Alguns métodos importantes:
  - setDefaultCloseOperation(int) – define o que ocorre quando o usuário tenta fechar a janela. As opções são: DO_NOTHING_ON_CLOSE, HIDE_ON_CLOSE, DISPOSE_ON_CLOSE, EXIT_ON_CLOSE.
  - setIconImage(Image) – altera o ícone da janela
  - setTitle(String) – altera o título da janela
  - setJMenuBar(JMenuBar) – associa uma barra de menus à janela

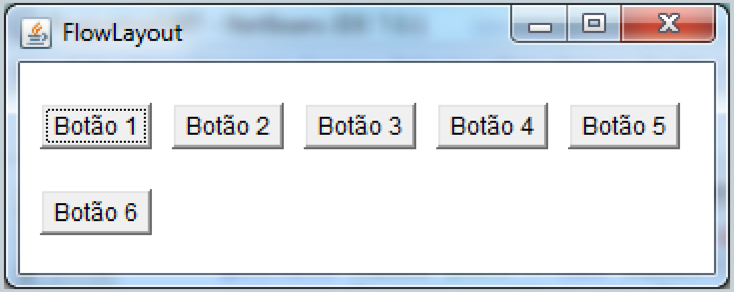
JPanel

- **JPanel** – tipo básico de container para inserção de componentes.
- Construtor:
  - JPanel() – cria um painel com layout padrão (FlowLayout)
  - JPanel(LayoutManager) – define o tipo de layout do painel
- Alguns métodos importantes:
  - add(Component, int) – adiciona o componente, definindo sua posição. O argumento int é opcional e depende do tipo de layout usado.
  - setLayout(LayoutManager) – altera o tipo de layout

Gerenciadores de layout

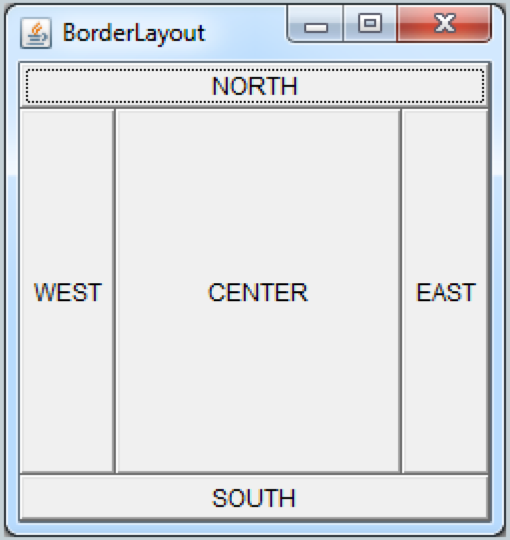
- **FlowLayout** – layout de fluxo. Os componentes são adicionados da esquerda para a direita. Caso não caiba na largura do container, o componente é adicionado na linha abaixo.
- Exemplos:
  - panel1.setLayout(new FlowLayout());
  - panel1.setLayout(new FlowLayout(FlowLayout.LEFT, 10, 10));





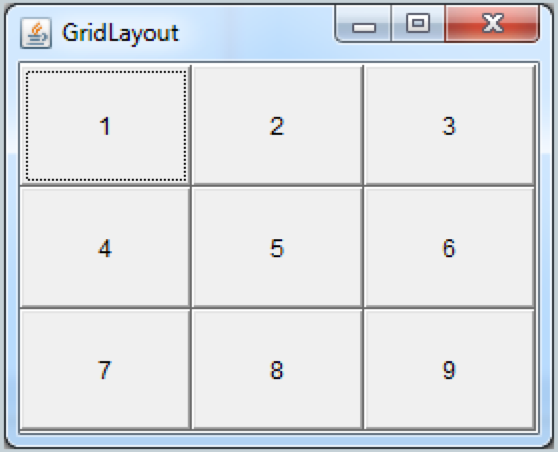
Gerenciadores de layout

- **BorderLayout** – layout de borda. O container é dividido em 5 áreas: norte, sul, leste, oeste e centro. Ao adicionar o componente, deve-se indicar a área.
- Exemplos:
  - panel1.setLayout(new BorderLayout());
  - btn1.add(panel1, BorderLayout.NORTH);
  - btn2.add(panel1, BorderLayout.SOUTH);
  - btn3.add(panel1, BorderLayout.CENTER);
  - btn4.add(panel1, BorderLayout.NORTH);
  - btn5.add(panel1, BorderLayout.NORTH);



Gerenciadores de layout

- **GridLayout** – layout de grade. Deve ser indicado o número de linhas e o número de colunas.
- Exemplos:
  - panel1.setLayout(new GridLayout(3, 3));



Gerenciadores de layout

- **GridBagLayout** – extensão layout de grade. Principais diferenças:
- Linhas e colunas podem ter diferentes dimensões, um componente pode ocupar mais de uma célula, o componente não precisa ocupar a célula inteira, podendo ser alinhado à borda da célula como.
- É o gerenciador que permite maior personalização, porém muito mais trabalhoso de lidar.

Gerenciadores de layout

- Os gerenciadores de layout usados pelos componentes Swing são classes pertencentes ao pacote AWT.
- Isto significa que, para usar um gerenciador de layout, além de importar o pacote **javax.swing**, deve-se importar também **java.awt**.

ImageIcon

- **ImageIcon** – um ícone de imagem, pequeno elemento gráfico que pode ser usado em rótulos de texto, botões, etc.
- Construtor:
  - ImageIcon(String) – modo mais comum de construção. A String indica o nome do arquivo de imagem a ser carregado.

JLabel

- **JLabel** – um rótulo de texto. Um texto informativo que pode passar uma orientação ao usuário.
- Construtor:
  - JLabel(String) – cria o rótulo com o texto especificado
  - JLabel(String, int) – cria o rótulo de texto especificando o alinhamento, que pode ser: SwingConstants.LEFT, SwingConstants.CENTER, SwingConstants.RIGHT
  - JLabel(String, Icon, int) – cria o rótulo com texto, ícone e alinhamento
- Alguns métodos importantes:
  - setText(String) – altera o texto.
  - getText() – retorna o texto atual.

JTextField

- **JTextField** – um campo de texto. Uma caixa onde o usuário pode informar um texto em uma linha.
- Construtor:
  - JTextField() – cria o campo de texto vazio.
  - JTextField(int) – cria o campo de texto com a largura especificada.
  - JTextField(String, int) – cria o campo de texto com um texto inicial e a largura especificada.
- Alguns métodos importantes:
  - setText(String) – altera o texto.
  - getText() – retorna o texto atual.
  - getSelectedText() – retorna o texto selecionado pelo usuário.

JPasswordField

- **JPasswordField** – um campo de texto protegido. É uma subclasse de JTextField.
- Alguns métodos importantes:
  - setEchoChar(char) – define o caractere que aparece ao digitar um texto

JTextArea

- **JTextArea** – uma caixa onde o usuário pode informar várias linhas de texto.
- Construtor:
  - JTextArea(int, int) – cria a área de texto especificando o número de linhas e de colunas.
  - JTextArea(String, int, int) – cria a área especificando o texto, o número de linhas e de colunas.
- Alguns métodos importantes:
  - setText(String) – altera o texto.
  - getText() – retorna o texto atual.
  - getSelectedText() – retorna o texto selecionado pelo usuário.
  - append(String) – adiciona um texto ao final do texto atual.
  - insert(String, int) – insere um texto na posição especificada.
  - setLineWrap(boolean) – ativa a quebra automática de linha.
  - setWrapStyleWord(boolean) – define se a quebra se dará por palavra ou por caractere

JScrollPane

- **JScrollPane** – um tipo de painel que possibilita o uso de barras de rolagem.
- Construtor:
  - JScrollPane(Component) – cria o painel contendo o componente especificado.
  - JScrollPane(Component, int, int) – cria o painel configurando as barras de rolagem vertical e horizontal. Deve-se usar:
  - ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS, ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED, ScrollPaneConstants.VERTICAL_SCROLLBAR_NEVER,
  - ScrollPaneConstants.HORIZONTAL_SCROLLBAR_ALWAYS, ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED, ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER,

JCheckBox

- **JCheckBox** – caixa de seleção, permite selecionar ou não uma opção.
- Construtor:
  - JCheckBox(String) – cria a caixa com o texto especificado
  - JCheckBox(String, boolean) – cria a caixa com o texto especificado e a opção de seleção (true/false)
  - JCheckBox(Icon) – cria a caixa com o ícone especificado
  - JCheckBox(Icon, boolean)
  - JCheckBox(String, Icon)
  - JCheckBox(String, Icon, boolean)

JCheckBox

- Alguns métodos importantes:
  - setSelected(boolean) – altera o estado da caixa de seleção
  - isSelected() – retorna *true* se a caixa estiver marcada, e *false* se não estiver

JRadioButton

- **JRadioButton** – componente semelhante ao JCheckBox, inclusive com os mesmos construtores e métodos.
- A diferença é que um botão de opção pode ser agrupado a outros, tornando-o exclusivo dentro do grupo. Para isso, deve-se usar a classe ButtonGroup.
- Exemplo:
  - ButtonGroup escolha = new ButtonGroup();
  - JRadioButton opcao1 = new JRadioButton(“Sim”);
  - JRadioButton opcao2 = new JRadioButton(“Não”);
  - JRadioButton opcao3 = new JRadioButton(“Talvez”);
  - escolha.add(opcao1);
  - escolha.add(opcao2);
  - escolha.add(opcao3);

JButton

- **JButton** – um botão destinado a executar uma ação.
- Construtor:
  - JButton() – cria um botão vazio
  - JButton(String) – cria um botão com texto
  - JButton(Icon) – cria um botão com um ícone
  - JButton(String, Icon) – cria um botão com texto e ícone
- Alguns métodos importantes:
  - setText(String) – altera o texto do botão
  - setIcon(Icon) – altera o ícone do botão

JComboBox

- **JComboBox** – uma caixa de combinação, da qual o usuário pode selecionar uma opção.
- Construtor:
  - JComboBox() – cria a caixa de combinação
- Alguns métodos importantes:
  - addItem(Object) – adiciona um item à lista de opções
  - setEditable(boolean) – permite que o usuário digite uma opção
  - getItemAt(int) – retorna o item que está na posição especificada
  - getItemCount() – retorna o número de itens na lista
  - getSelectedIndex() – retorna a posição do item atualmente selecionado
  - getSelectedItem() – retorna o texto do item atualmente selecionado
  - setSelectedIndex(int) – seleciona o item da posição especificada
  - setSelectedIndex(Object) – seleciona o objeto especificado na lista

JList

- **JList** – uma lista de opções que permite a seleção de mais de um item simultaneamente.
- Construtor:
  - JList() – cria uma lista vazia
  - JList(Object[]) – cria uma lista que contém um array de objetos (String, por exemplo)
- Alguns métodos importantes:
  - setListData(Object[]) – preenche ou altera os itens de uma lista
  - getSelectedValues() – retorna um array de objetos contendo os itens selecionados na lista

JMenuBar, JMenu, JMenuItem

```
JMenuBar barra = new JMenuBar();
JMenu m1 = new JMenu("Arquivo");
JMenuItem m11 = new JMenuItem("Novo");
JMenuItem m12 = new JMenuItem("Abrir");
JMenuItem m13 = new JMenuItem("Salvar");
JMenuItem m14 = new JMenuItem("Sair");
JMenu m2 = new JMenu("Ajuda");
JMenuItem m21 = new JMenuItem("Conteudo");
JMenuItem m22 = new JMenuItem("Sobre");
m1.add(m11);
m1.add(m12);
m1.add(m13);
m1.addSeparator();
m1.add(m14);
m2.add(m21);
m2.addSeparator();
m2.add(m22);
barra.add(m1);
barra.add(m2);
janela.setJMenuBar(barra);
```

Caixas de diálogo

- Para pequenas mensagens ou questões ao usuário, existem caixas de diálogo padrão, implementadas pela classe JOptionPane.
- Os métodos que criam as caixas de diálogo são estáticos, por isso a classe não precisa ser instanciada para chamá-los.
- Existem quatro caixas de diálogo padrão:
  - ConfirmDialog
  - InputDialog
  - MessageDialog
  - OptionDialog

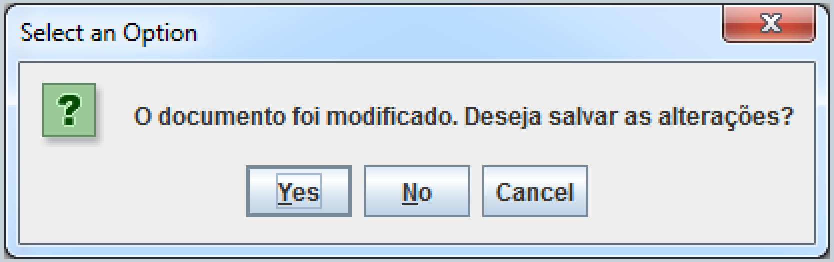
ConfirmDialog

- Método:
  - JOptionPane.showConfirmDialog(Component, Object, String, int, int)
    - **Component**: indica qual é o container que chama a caixa de diálogo, pode ser **null**.
    - **Object**: a mensagem a ser exibida. Pode ser uma String ou uma imagem, por exemplo.
    - **String**: opcional. Título da janela.
    - **int**: opcional. Botões que serão mostrados: JOptionPane.YES_NO_CANCEL_OPTION, JOptionPane.YES_NO_OPTION.
    - **int**: opcional. Tipo da caixa de diálogo (ícone mostrado): JOptionPane.ERROR_MESSAGE, JOptionPane.INFORMATION_MESSAGE, JOptionPane.PLAIN_MESSAGE, JOptionPane.QUESTION_MESSAGE, JOptionPane.WARNING_MESSAGE

ConfirmDialog

- O retorno da função é a opção escolhida pelo usuário, que pode ser: JOptionPane.YES_OPTION, JOptionPane.NO_OPTION ou JOptionPane.CANCEL_OPTION
- Exemplo:

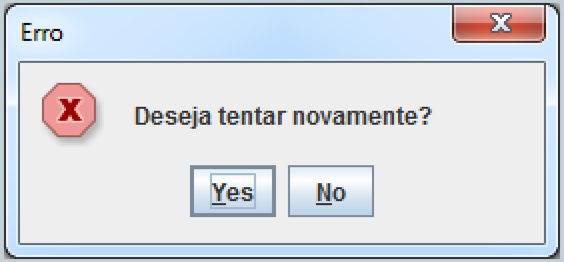
```
int resposta;
resposta = JOptionPane.showConfirmDialog(null, "O documento
foi modificado. Deseja salvar as alterações?");
```



ConfirmDialog

- Exemplo:

```
resposta = JOptionPane.showConfirmDialog(null, "Deseja
tentar novamente?", "Erro", JOptionPane.YES_NO_OPTION,
JOptionPane.ERROR_MESSAGE);
```



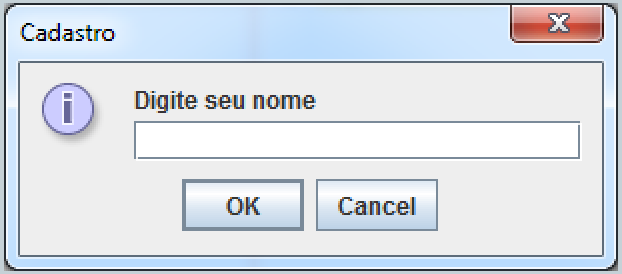
InputDialog

- Método:
  - JOptionPane.showInputDialog(Component, Object, String, int)
    - **Component**: indica qual é o container que chama a caixa de diálogo, pode ser **null**.
    - **Object**: a mensagem a ser exibida. Pode ser uma String ou uma imagem, por exemplo.
    - **String**: opcional. Título da janela.
    - **int**: opcional. Tipo da caixa de diálogo (ícone mostrado), mesmas opções do showConfirmDialog.

InputDialog

- O retorno da função é uma String contendo o texto digitado pelo usuário. Caso o usuário clique em Cancel ou feche a janela, é retornado **null**.
- Exemplo:

```
String resposta;
resposta = JOptionPane.showInputDialog(null, "Digite seu
nome", "Cadastro", JOptionPane.INFORMATION_MESSAGE);
```



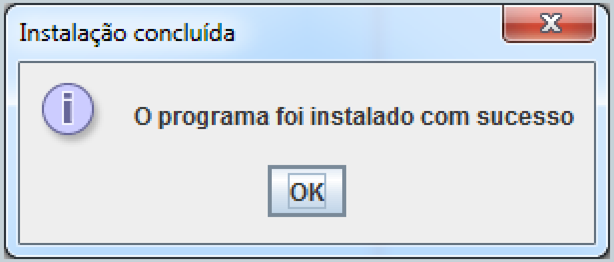
MessageDialog

- Método:
  - JOptionPane.showMessageDialog(Component, Object, String, int)
    - **Component**: indica qual é o container que chama a caixa de diálogo, pode ser **null**.
    - **Object**: a mensagem a ser exibida. Pode ser uma String ou uma imagem, por exemplo.
    - **String**: opcional. Título da janela.
    - **int**: opcional. Tipo da caixa de diálogo (ícone mostrado), mesmas opções do showConfirmDialog e showInputDialog.

MessageDialog

- O método showMessageDialog não retorna valor.
- Exemplo:

```
JOptionPane.showInputDialog(null, "O programa foi instalado
com sucesso", "Instalação concluída",
JOptionPane.INFORMATION_MESSAGE);
```



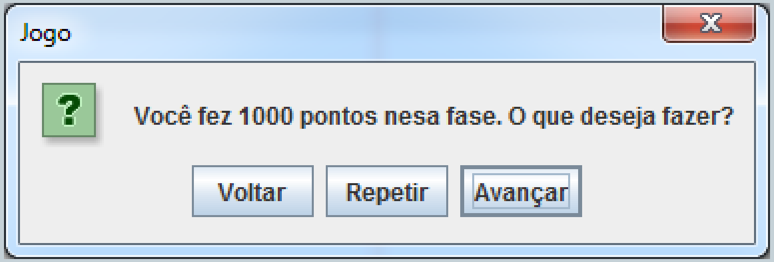
OptionDialog

- Método:
  - JOptionPane.showOptionDialog(Component, Object, String, int, int, Icon, Object[], Object)
    - **Component**: indica qual é o container que chama a caixa de diálogo, pode ser **null**.
    - **Object**: a mensagem a ser exibida. Pode ser uma String ou uma imagem, por exemplo.
    - **String**: título da janela.
    - **int**: botões que serão mostrados, mesmas opções do showConfirmDialog, ou 0 (zero) se outros botões forem usados.
    - **int**: ícone da caixa de diálogo, mesmas opções do showConfirmDialog, ou 0 (zero) se for usado outro ícone.
    - **Icon**: ícone a ser usado se a opção anterior for 0.
    - **Object[]** – array contendo os textos dos botões, caso não seja usado um dos padrões.
    - **Object** – um elemento da array indicando qual é o botão se seleção padrão.

OptionDialog

- O retorno da função é um inteiro, indicando a posição do array relativa ao botão selecionado.
- Exemplo:

```
String[] opcoes = {"Voltar", "Repetir", "Avançar"};
int resposta;
resposta = JOptionPane.showOptionDialog(null, "Você fez 1000
pontos nesa fase. O que deseja fazer?", "Jogo", 0,
JOptionPane.QUESTION_MESSAGE, null, opcoes, opcoes[2]);
```

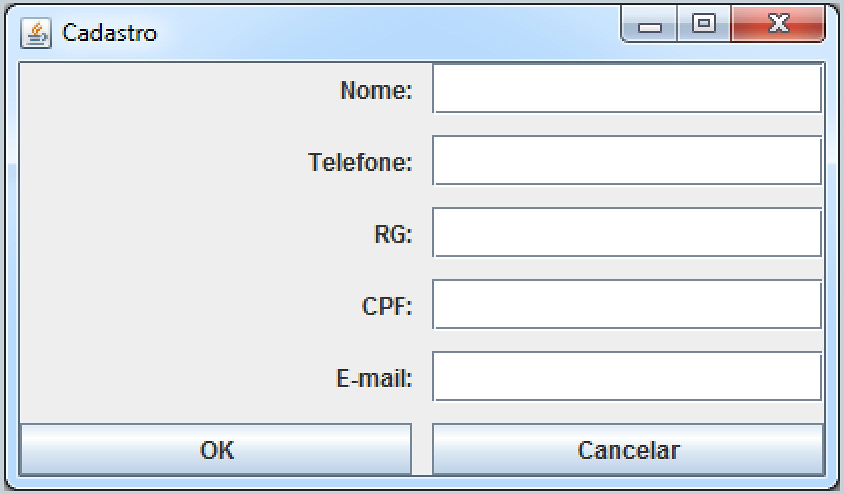


Outros componentes Swing

- Há diversos outros componentes Swing que podem ser úteis dependendo da aplicação. Exemplos: JSlider, JProgressBar, JToolBar, JTabbedPane
- http://download.oracle.com/javase/7/docs/api/javax/swing/package-summary.html

Exercício

- Monte a seguinte janela usando componentes Swing:



- Não é preciso adicionar nenhuma funcionalidade.