



Componentes Parte 01

Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro
Prof. Edwar Saliba Júnior
Setembro de 2012



Reflexão

“Merecemos a culpa por não termos facilitado o seu aprendizado. No tocante aos recursos, o produto era fantástico, mas no que se refere à facilidade dos primeiros passos, não nos saímos muito bem.”

(Bill Gates sobre a frustração parcial de alguns usuários com o Word 2.0)



Dica

Interfaces consistentes permitem que o usuário aprenda mais rápido e com mais facilidade novos aplicativos.




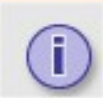


Entrada/saída baseada em GUI simples com JOptionPane

- Caixas de diálogo:
 - utilizadas pelas aplicações para interagir com o usuário;
 - fornecidas pela classe `JOptionPane` (pacote `javax.swing`):
 - contêm diálogos de entrada e diálogos de mensagem.
 - sintaxe:

```
JOptionPane.showMessageDialog(...);  
JOptionPane.showInputDialog(...);
```



Constantes JOptionPane static para diálogos de mensagem

Tipo de diálogo de mensagem	Ícone	Descrição
ERROR MESSAGE		Um diálogo que indica um erro para o usuário.
INFORMATION MESSAGE		Um diálogo com uma mensagem informativa para o usuário.
WARNING MESSAGE		Um diálogo que adverte o usuário de um problema potencial.
QUESTION MESSAGE		Um diálogo que impõe uma pergunta ao usuário. Normalmente, esse diálogo exige uma resposta, como clicar em um botão Yes ou No.
PLAIN MESSAGE	Nenhum ícone	Um diálogo que contém uma mensagem, mas nenhum ícone..

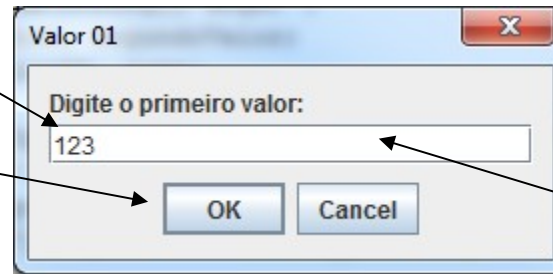


Exercício 01 - JOptionPane

Diálogo de entrada exibido nas linhas 14 e 15

Cursor para o usuário

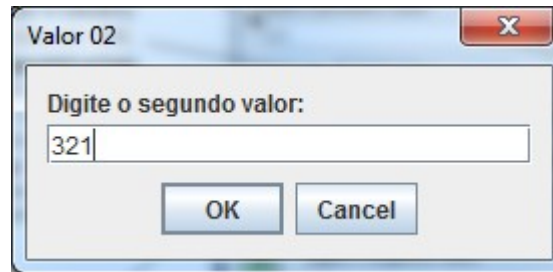
Quando o usuário clica em **OK**, **showInputDialog** retorna ao programa o **123** digitado pelo usuário como uma **String**. O programa deve converter a **String** em um **float**



Campo de texto em que o usuário digita um valor

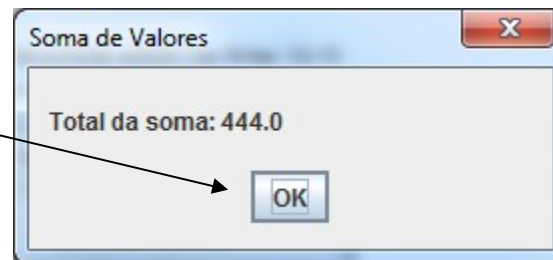
Diálogo de entrada exibido na linha 16 e 17

barra de título



Diálogo de entrada exibido nas linhas 22-23

Quando o usuário clicar em **OK**, o diálogo de mensagem é fechado (é removido da tela)





Exemplo 01 - JOptionPane

```
1 package pv_unidade_02_ex01_joptionpane;
2
3 import javax.swing.JOptionPane;
4
5 /**
6  * @author Edwar Saliba Júnior - http://www.esj.eti.br
7  */
8 public class PV_Unidade_02_Ex01_JOptionPane {
9
10     public static void main(String[] args) {
11         String primeiroValor, segundoValor;
12         float valor01, valor02, soma;
13
14         primeiroValor = JOptionPane.showInputDialog (null, "Digite o primeiro valor: ",
15             "Valor 01", JOptionPane.PLAIN_MESSAGE);
16         segundoValor = JOptionPane.showInputDialog (null, "Digite o segundo valor: ",
17             "Valor 02", JOptionPane.PLAIN_MESSAGE);
18
19         valor01 = Float.parseFloat(primeiroValor);
20         valor02 = Float.parseFloat(segundoValor);
21
22         soma = valor01 + valor02;
23
24         JOptionPane.showMessageDialog (null, "Total da soma: " + soma,
25             "Soma de Valores", JOptionPane.PLAIN_MESSAGE);
26     }
27 }
```

Mostra o diálogo de entrada para receber o primeiro valor

Mostra o diálogo de entrada para receber o segundo valor

Mostra o diálogo de mensagem para gerar a saída da soma para o usuário



Visão Geral de Componentes Swing

- Componentes Swing GUI:
 - declarado no pacote `javax.swing`;
 - a maioria dos componentes Swing são componentes *Java puros* — escritos, manipulados e exibidos em Java;
 - fazem parte das bibliotecas do Java (Java Foundation Classes (JFC)) para desenvolvimento de GUI para múltiplas plataformas.



Alguns Componentes Básicos

Componente	Descrição
JLabel	Exibe texto não-editável ou ícones.
TextField	Permite ao usuário inserir dados do teclado. Também pode ser utilizado para exibir texto editável ou não editável.
Button	Desencadeia um evento quando o usuário clicar nele com o mouse.
CheckBox	Especifica uma opção que pode ser ou não selecionada.
ComboBox	Fornecer uma lista drop-down de itens a partir da qual o usuário pode fazer uma seleção clicando em um item ou possivelmente digitando na caixa.
List	Fornecer uma lista de itens a partir da qual o usuário pode fazer uma seleção clicando em qualquer item na lista. Múltiplos elementos podem ser selecionados.
Panel	Fornecer uma área em que os componentes podem ser colocados e organizados. Também pode ser utilizado como uma área de desenho para imagens gráficas.



Swing versus AWT

- *Abstract Window Toolkit (AWT):*
 - precursor do Swing;
 - declarado no pacote `java.awt`;
 - não fornece aparência e comportamento consistentes para diversas plataformas.



Dica de Portabilidade

Os componentes Swing são implementados com Java; desse modo, eles são mais portáveis e flexíveis do que os componentes Java do pacote `java.awt`.



Componentes Leves X Pesados

- Componentes leves:
 - Não associados diretamente a componentes GUI suportados pela plataforma subjacente;
- Componentes pesados:
 - Associados diretamente à plataforma local;
 - Componentes AWT;
 - Alguns componentes Swing.

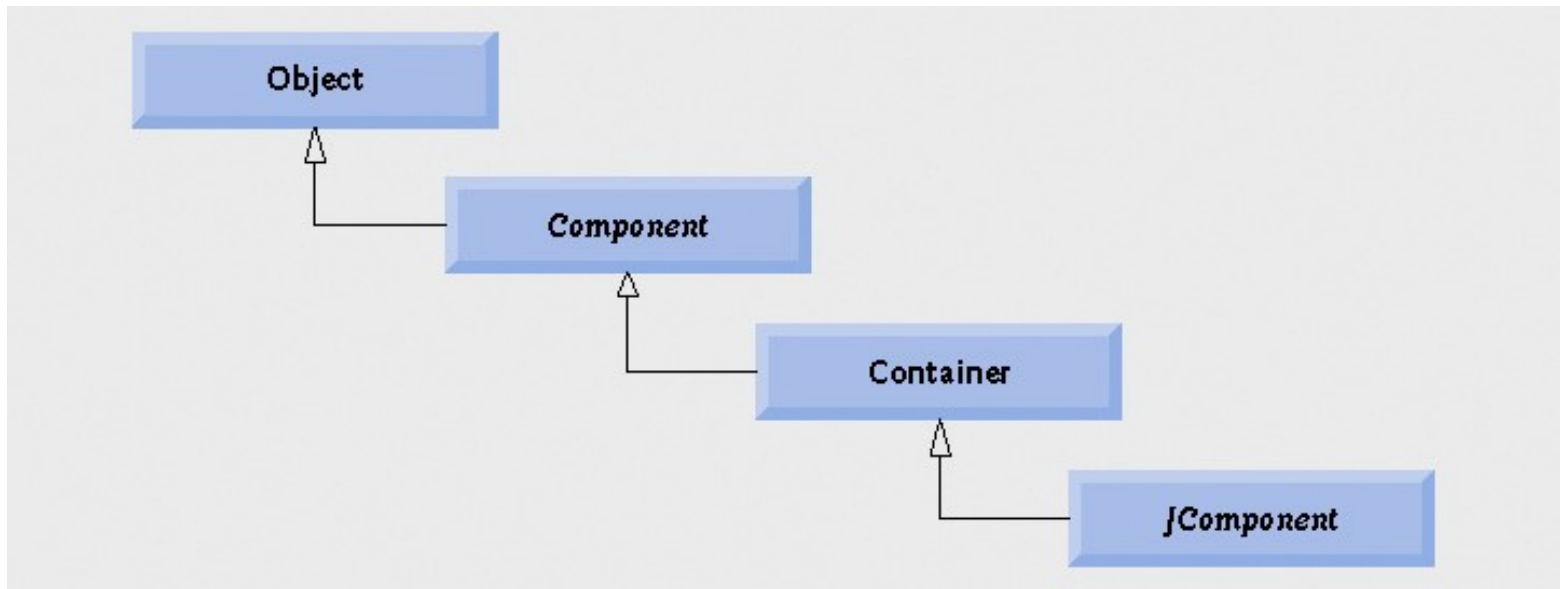


Observação sobre Aparência e Comportamento

- Componentes Pesados:
 - a aparência e o comportamento de uma interface definida com componentes GUI do pacote `java.awt`, podem variar entre plataformas (S.O.);
 - são acoplados à GUI da plataforma local, assim sendo, sua aparência e comportamento variam entre plataformas.



Superclasses Comuns de Muitos dos Componentes do Swing





Superclasses de Componentes GUI Leves do Swing

- Classe Component (pacote `java.awt`):
 - subclasse de `Object`;
 - declara muitos comportamentos e atributos comuns a componentes GUI.
- Classe Container (pacote `java.awt`):
 - subclasse de `Component`;
 - organiza componentes.
- Classe JComponent (pacote `javax.swing`):
 - subclasse de `Container`;
 - superclasse de todos os componentes Swing leves.



Superclasses de Componentes GUI Leves do Swing

- Recursos dos componentes leves comuns:
 - aparência e comportamento adaptáveis para personalizar a aparência dos componentes;
 - teclas de atalho (chamadas *mnemônicas*);
 - capacidades comuns de tratamento de eventos;
 - breves descrições do propósito de um componente GUI (*dicas de ferramenta*) e
 - suporte para *localização* de interface com o usuário.



Componentes Visuais



JFrame

- Usado para exibição de texto e imagens em uma janela;
- classe JFrame:
 - a maioria das janelas é uma instância ou subclasse dessa classe;
 - fornece a barra de título e
 - fornece botões para minimizar, maximizar e fechar a aplicação.



Métodos Importantes

- JFrame:
 - `setDefaultCloseOperation`
 - determina como a aplicação reage quando o usuário clica no botão fechar;
 - `setSize`
 - especifica a largura e altura da janela e
 - `setVisible`
 - determina se a janela é exibida (`true`) ou não (`false`).



Métodos Importantes Comuns a Maioria dos Componentes Visuais

- `setToolTipText`
 - quando preenchido, apresenta um dica ao usuário do *software*, quando este para o *mouse* sobre o componente;
- `setEnabled`
 - habilita ou desabilita um componente;
- `setFocusable`
 - habilita ou desabilita a possibilidade de um componente receber foco e
- `setNextFocusableComponent`
 - desvia a sequência de foco entre os componentes habilitados a recebê-lo.



JLabel

- Rótulo;
- classe JLabel:
 - instruções de texto ou informações que declaram o propósito de cada componente;
 - muito utilizado também em campos que têm o objetivo único de visualização de dados.



Métodos Importantes

- `JLabel`:
 - `getText` e `setText`
 - para colocar e recuperar o texto de um rótulo;
 - `getIcon` e `setIcon`
 - para colocar e recuperar o ícone exibido no rótulo;
 - `getHorizontalTextPosition` e `setHorizontalTextPosition`
 - para configurar e recuperar a posição horizontal do texto exibido no rótulo e
 - `setDisplayMnemonic` e `setLabelFor`
 - estes métodos funcionam em conjunto. O primeiro configura um atalho no texto apresentado no `JLabel` e o segundo é o método que aponta para o componente que deverá receber o foco.



Ícones

- Interface Icon:
 - pode ser adicionada a um JLabel com o método setIcon;
 - implementado pela classe ImageIcon.



Ícones no Apache Netbeans (AN)

- Para que o AN reconheça os ícones que serão colocados nos JLabel's, deve-se fazer o seguinte:
 - usar o caminho absoluto da imagem (ícone) no sistema operacional ou
 - colocar as seguintes linhas no arquivo pom.xml:

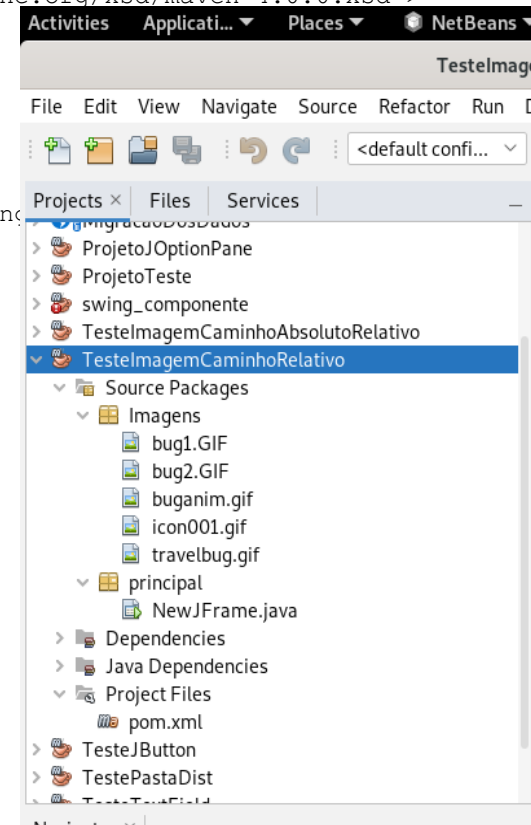
```
<build>
  <resources>
    <resource>
      <directory>src/main/java</directory>
      <includes>
        <include>/*.java</include>
        <include>nomeDoDiretorio/*.gif</include>
        <include>nomeDoDiretorio/*.GIF</include>
      </includes>
    </resource>
  </resources>
</build>
```




Ícones no Apache Netbeans (AN)

- Exemplo de arquivo pom.xml modificado para aceitar as imagens que foram importadas para a pasta "Imagens" do projeto:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>br.edu.iftm.testeimagemcaminhorelativo</groupId>
  <artifactId>TesteImagemCaminhoRelativo</artifactId>
  <version>1.0-SNAPSHOT</version>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
  </properties>
  <build>
    <resources>
      <resource>
        <directory>src/main/java</directory>
        <includes>
          <include>/*.java</include>
          <include>Imagens/*.gif</include>
          <include>Imagens/*.GIF</include>
        </includes>
      </resource>
    </resources>
  </build>
</project>
```





Constantes

- Interface `SwingConstants`:
 - declara um conjunto de constantes inteiras comuns, como as utilizadas para configurar o alinhamento dos componentes;
 - pode ser utilizada com os métodos:
 - `setHorizontalAlignment` e
 - `setVerticalAlignment`.



Algumas Constantes

Constante	Descrição
<i>Constantes de posição horizontal</i>	
<code>SwingConstants.LEFT</code>	Coloca o texto à esquerda.
<code>SwingConstants.CENTER</code>	Coloca o texto no centro.
<code>SwingConstants.RIGHT</code>	Coloca o texto à direita.
<i>Constantes de posição vertical</i>	
<code>SwingConstants.TOP</code>	Coloca o texto na parte superior.
<code>SwingConstants.CENTER</code>	Coloca o texto no centro.
<code>SwingConstants.BOTTOM</code>	Coloca o texto na parte inferior.



Exercício 02 - JLabel

- Desenvolva um *software* utilizando apenas:
 - um componente JFrame,
 - três componentes JLabel e
 - uma imagem.
- Seu *software* deverá produzir a seguinte saída:





Exemplo 02 - JLabel

- ♦ Parte 01 - Classe Main.
- ♦ Parte 02 - Classe LabelFrame.



Eventos Comuns

- Eventos comuns nos componentes GUI:
 - clicar em um botão,
 - digitar em um campo de texto,
 - selecionar um item de *menu*,
 - fechar uma janela e
 - mover um *mouse*.
- O código que realiza uma tarefa em resposta a um evento é chamado de: *handler* de evento.



Campos de Texto

- A classe `JTextComponent`
 - é superclasse de `JTextField`
 - é superclasse de `JPasswordField`,
 - que adiciona o caractere de eco para ocultar a entrada de texto no componente;
 - permite que o usuário insira texto no componente quando o componente tem o foco da aplicação.



Conceito

- Classe de primeiro nível:
 - não declarada dentro de uma outra classe.
- Classes aninhadas:
 - declaradas dentro de uma outra classe;
 - classes aninhadas não-`static` são chamadas classes internas;
 - frequentemente utilizadas para tratamento de eventos.



Observação

Uma classe interna tem permissão de acessar diretamente variáveis e métodos de sua classe de primeiro nível, mesmo se eles forem `private`.



JTextField e JPasswordField

- `JTextField` e `JPasswordField`,
 - pressionar `<Enter>` dentro de um desses campos dispara um `ActionEvent`,
 - este será processado pelos objetos que implementam a interface `ActionListener`.

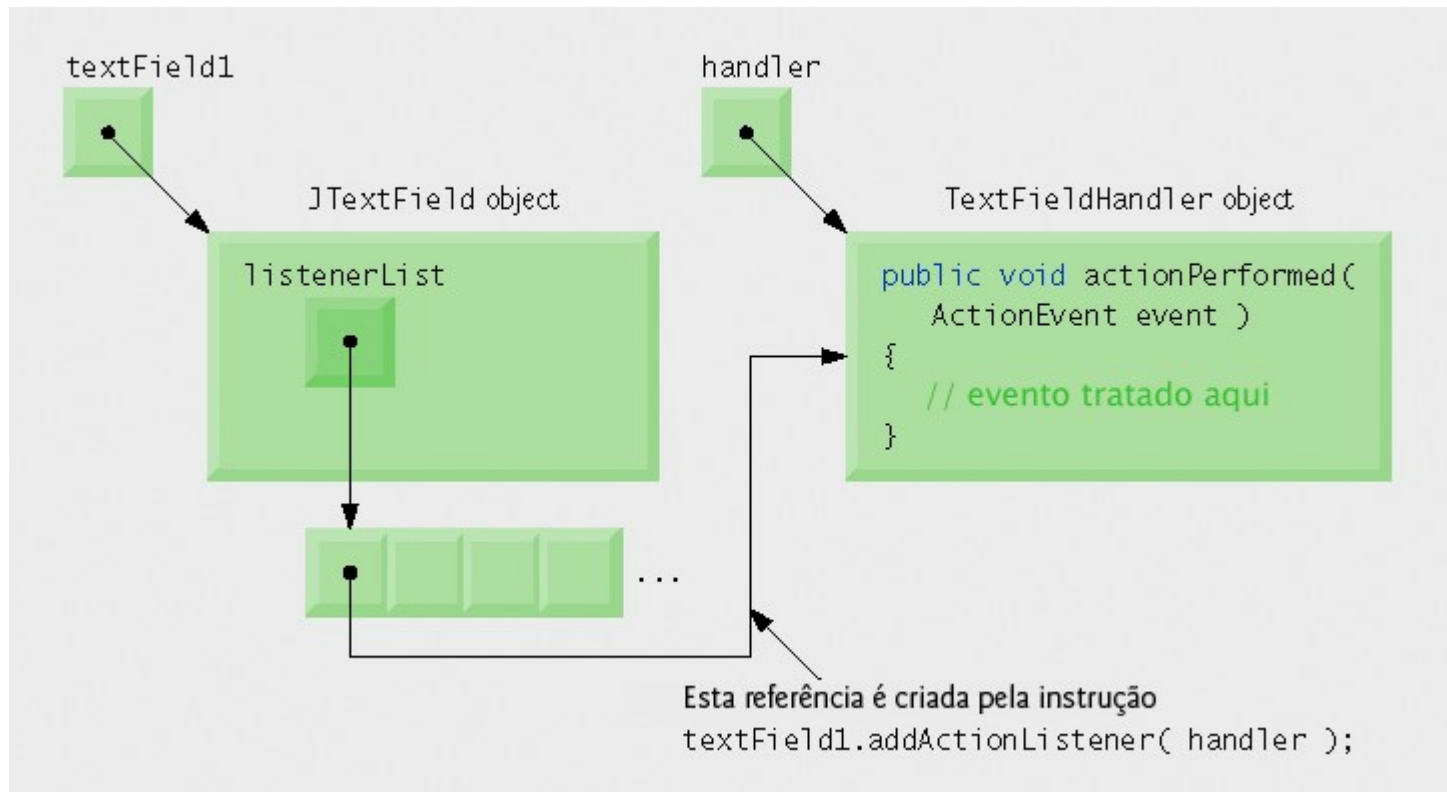


actionPerformed

- Fonte do evento:
 - componente a partir do qual o evento se origina;
 - pode ser determinado utilizando o método `getSource`;
 - o texto em um `JTextField` pode ser adquirido utilizando o método `getActionCommand` do evento (por meio do parâmetro do próprio evento)
 - o texto em um `JPasswordField` pode ser adquirido utilizando o método `getPassword`, da seguinte maneira:
 - `new String(objeto.getPassword())`



Exemplo de Registro de Evento



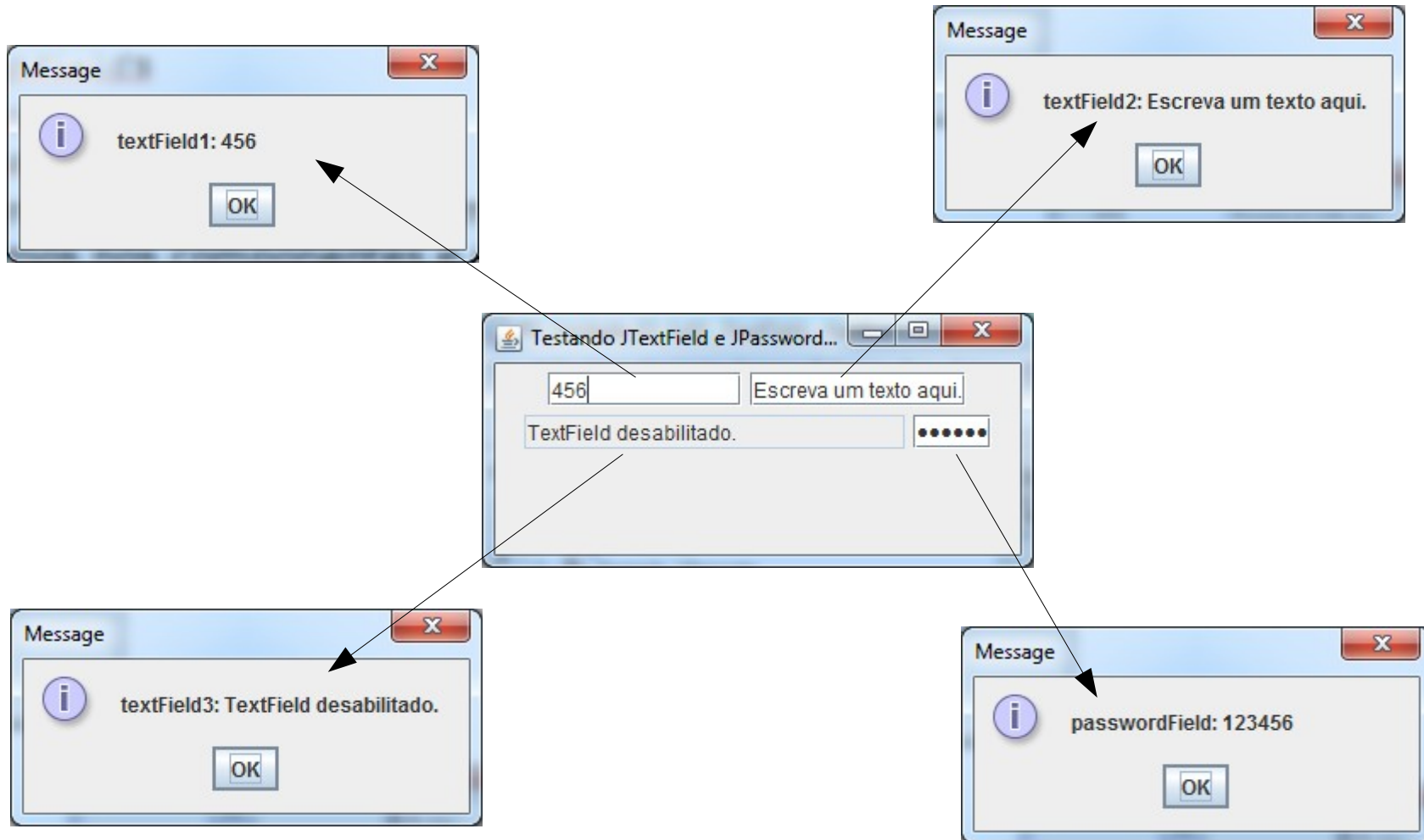


Exercício 03 - JTextField

- Desenvolva um *software* que possua:
 - 03 JTextField's
 - 01 JPasswordField e
 - 01 JFrame.
- O *software* deverá funcionar assim:
 - ao escrevermos nos componentes e/ou apertarmos a tecla <Enter>, o nome e o valor no componente deverá ser mostrado num JOptionPane;
- Conforme imagem no próximo *slide*:



Exercício 03 - Resultado Visual





Exemplo 03 - JTextField e JPasswordField

- Parte 01 - Classe Main.
- Parte 02 - Classe TextFieldFrame.



JButton

- Botão:
 - o usuário do *software* clica no botão para desencadear uma ação específica;
 - pode ser botão de comando, caixa de seleção, botão de alternância ou botão de opção e
 - os tipos de botões são subclasses da classe `AbstractButton`.



JButton

- Método interessante:
 - `setMnemonic`
 - este método configura uma tecla de atalho (alt + “um caractere”) no texto apresentado no rótulo do JButton.



JButton

- JButton's podem ter um ícone de *rollover*:
 - este ícone aparece quando o *mouse* é posicionado sobre o botão;
 - pode ser adicionado a um JButton pelo método `setRolloverIcon`.

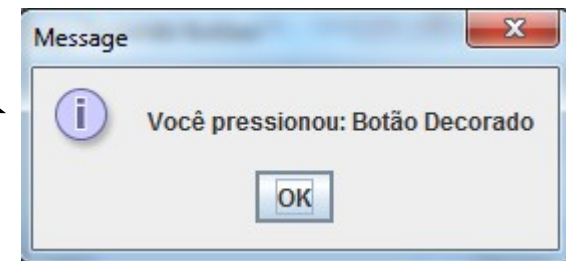
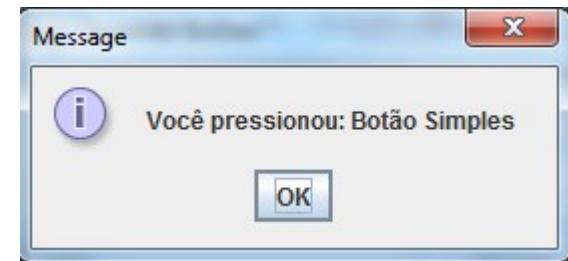


Exercício 04 - JButton

- Desenvolva um *software* utilizando:
 - 01 JFrame,
 - 02 JButton's e
 - 02 figuras.
- Seu *software* deverá possuir um botão com imagem e outro sem. E deverá funcionar assim:
 - ao clicar em um botão, deverá ser mostrado em um JOptionPane uma mensagem contendo o nome do componente e seu texto;
 - no botão com imagem, ao passarmos o *mouse* sobre o componente, uma imagem deverá sobrepor a outra.
- Seu *software* deverá possuir a seguinte aparência (*slide* a seguir):



Exercício 04 - Resultado Visual





Exemplo 04 - JButton

- Parte 01 - Classe Main.
- Parte 02 - Classe ButtonFrame.



Exercícios

- Resolver os exercícios da Lista 01



Bibliografia

- DEITEL, H. M.; DEITEL, P. J. **Java Como Programar**; 6. ed., São Paulo: Pearson, 2005.
- SALIBA JÚNIOR, E. *Slides* da disciplina de Programação de Computadores II - CEFET-MG. Disponível em: http://www.esj.eti.br/CEFETMG/Disciplinas/PC2/PC2_Unidade_04.pdf
Acesso em: 20 set. 2012.