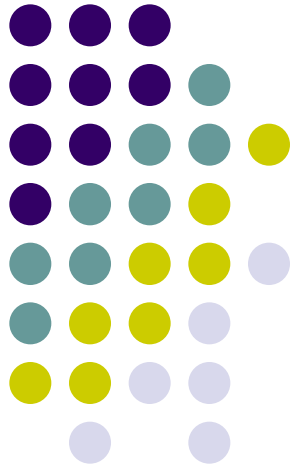
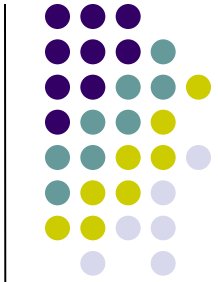
COM220

Aula 18: Componentes Básicos de Controle – Parte 1

Prof. Laércio Baldochi



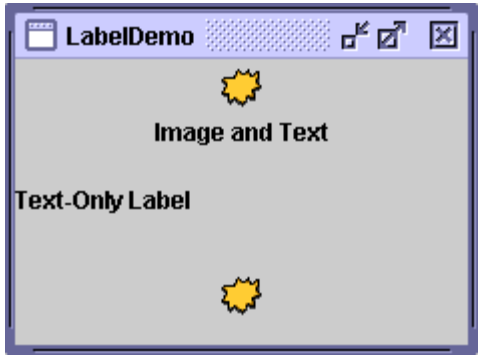
Componentes básicos de controle
Parte 1



- Rótulos
  - JLabel
- Botões
  - JButton
  - JToggleButton
  - JCheckBox
  - JRadioButton
- Campos de texto
  - JTextField
  - JTextArea

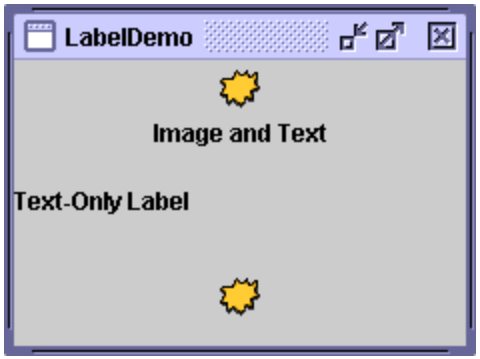
JLabel

- Geralmente usados para mostrar informação textual em containeres
- Podem ser alinhados horizontal e verticalmente
  - setHorizontalAlignment(alignment)
  - setVerticalAlignment(alignment)
- Podem estar associados a uma imagem (icon)

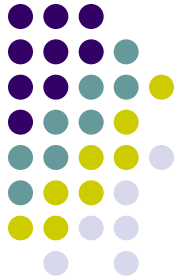


JLabel
Exemplo

```
// Cria um objeto contendo a imagem indicada
ImageIcon icon = new ImageIcon("images/middle.gif");

// Cria o primeiro rótulo, setando o texto e o alinhamento
JLabel lbl1 = new JLabel("Image and Text", JLabel.CENTER);
lbl1.setIcon(icon); //Seta o ícone

// Ajusta a posição do texto em relação ao ícone
lbl1.setVerticalTextPosition(JLabel.BOTTOM);
lbl1.setHorizontalTextPosition(JLabel.CENTER);

// Cria o segundo rótulo setando apenas o texto
JLabel lbl2 = new JLabel("Text-Only Label");

// Cria o terceiro rótulo e seta o alinhamento
JLabel lbl3 = new JLabel(icon);
lbl3.setHorizontalAlignment(JLabel.CENTER);
```



Botões

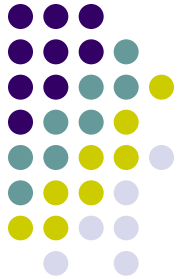


- Componentes de interface mais utilizados
- Classe AbstractButton
  - Define o conjunto de métodos e propriedades comuns que os tipos especializados de botões devem implementar
    - JButton
    - JToogleButton
    - JCheckBox
    - JRadioButton

JButton

- Podem exibir texto ou ícone
- São geralmente associados a algum tipo de listener para tratamento de eventos
- Podem ser habilitados e desabilitados

JButton
Exemplo

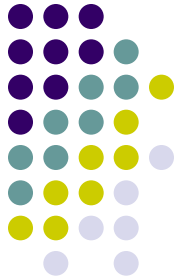


```
b1 = new JButton();                          // Cria o botao esquerdo
b1.setText("Disable middle button");         // Seta o texto
b1.setActionCommand("DISABLE");              // Define o comando
b1.setMnemonic(KeyEvent.VK_D);               // Seta a tecla de atalho

icon = new ImageIcon("images/middle.gif");   // Cria o icone
b2 = new JButton("Middle button", icon);     // Cria o botao do meio
b2.setMnemonic(KeyEvent.VK_M);               // Seta o atalho
b2.setVerticalTextPosition(JLabel.BOTTOM);   // Ajusta a posicao

b3 = new JButton("Enable middle button");    // Cria o botao direito
b3.setEnabled(false);                        // Desabilita o botao
b3.setActionCommand("ENABLE");               // Seta o commando
b3.setMnemonic(KeyEvent.VK_E);               // Seta o atalho
b1.addActionListener(this);                  // Adiciona o listener ao botao
b3.addActionListener(this);                  // Adiciona o listener ao botao
```

JButton Exemplo

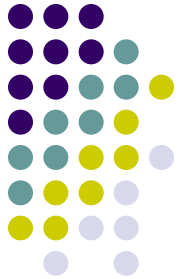
```
// Implementa o listener

public void actionPerformed(ActionEvent e) {
   String command = e.getActionCommand();
   // Se clicou em desabilitar
   if ("DISABLE".equals(command)) {
      b1.setEnabled(false);
      b2.setEnabled(false);
      b3.setEnabled(true);
   }
   // Senao, clicou em habilitar
   else {
      b1.setEnabled(true);
      b2.setEnabled(true);
      b3.setEnabled(false);
   }
}
```

JToggleButton



- Implementa um tipo de botão que possui dois estados: *selecionado* e *não-selecionado*
- É superclasse de JCheckBox e JRadioButton
- Ações de seleção e deseleção podem ser capturadas pela interface itemListener
- itemListener
  - Define apenas um método que deve ser sobrescrito
    - itemStateChanged

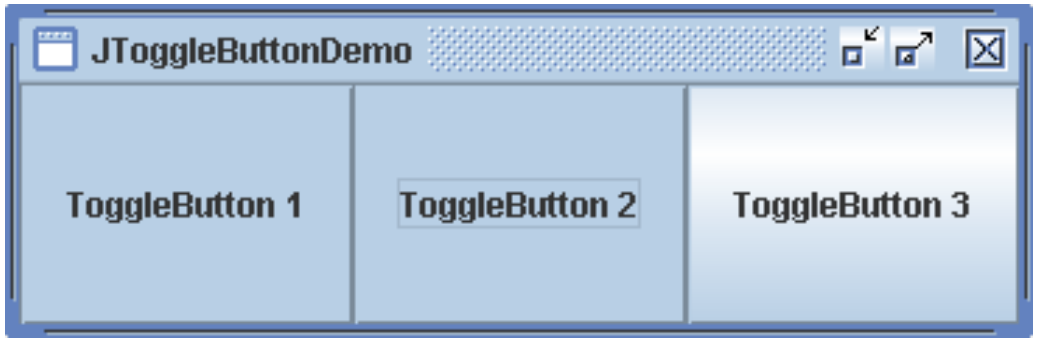
JToggleButton Exemplo



```
// Cria os botoes
JToggleButton b1 = new JToggleButton("ToggleButton 1");
JToggleButton b2 = new JToggleButton("ToggleButton 2");
JToggleButton b3 = new JToggleButton("ToggleButton 3");

// Adiciona o listener aos botoes
b1.addItemListener(this);
b2.addItemListener(this);
b3.addItemListener(this);

// Adiciona os componentes ao container
getContentPane().setLayout(new GridLayout(1, 3));
getContentPane().add(b1);
getContentPane().add(b2);
getContentPane().add(b3);
...
```

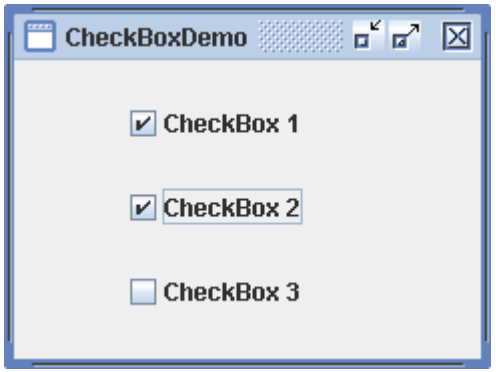
JToggleButton Exemplo

```
// Implementa a interface ItemListener

public void itemStateChanged(ItemEvent e) {
   JToggleButton button = (JToggleButton) e.getSource();
   String text = button.getText();
   String sel = button.isSelected() ? " selected." : " deselected.";
   System.out.println(text + sel);
}
```



JCheckBox

- Implementa a metáfora das caixas de controle / verificação
- Herda propriedades da classe JToggleButton
- Pode ter seus eventos capturados por meio da interface ItemListener
- Métodos isSelected() e setSelected() são usados para verificar/ajustar o estado das caixas

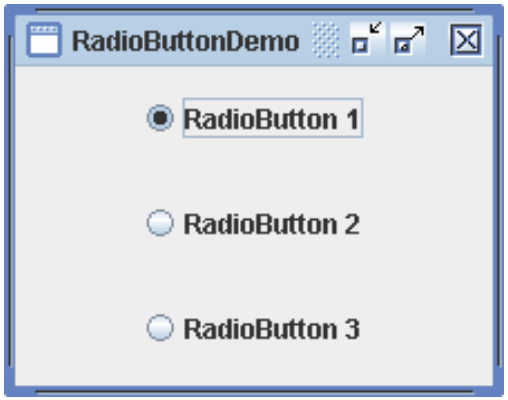


JCheckBox
Exemplo

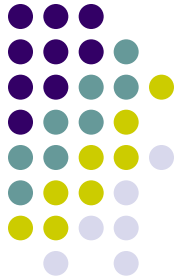
```
cb1 = new JCheckBox();          // Estado default é "deselected" (false)
cb1.setText("Checkbox 1");
cb2 = new JCheckBox("Checkbox 2");
cb3 = new JCheckBox("Checkbox 3");
cb3.setSelected(true);          // Seta o estado para "selected" (true)
// Registra o listener nos checkboxes
cb1.addItemListener(this);
cb2.addItemListener(this);
cb3.addItemListener(this);
...
// Implementa a interface ItemListener
public void itemStateChanged(ItemEvent e) {
   Object src = e.getItemSelectable();
   if (src == cb1) {
      if (e.getStateChange() == ItemEvent.SELECTED)
         System.out.println("Checkbox 1 selected.");
      else System.out.println("Checkbox 1 unselected.");
   } else if (src == cb2) {
      ...
   } else if (src == cb3) {
      ...
   }
}
```

JRadioButton

- Permite agrupar botões
  - Em cada grupo, apenas 1 botão pode estar selecionado em um dado momento
    - Classe ButtonGroup cuida do agrupamento de botões
  - É subclasse de JToggleButton
    - Eventos são capturados por meio da interface ItemListener



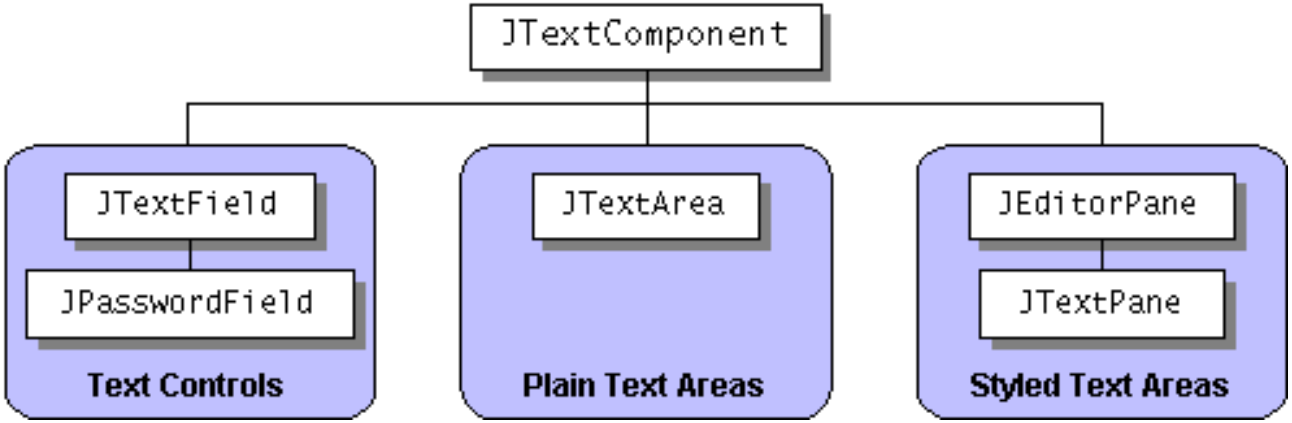
JRadioButton Exemplo



```
// Cria os botoes de radio
JRadioButton rb1 = new JRadioButton("RadioButton 1");
JRadioButton rb2 = new JRadioButton("RadioButton 2");
JRadioButton rb3 = new JRadioButton("RadioButton 3");
// Adiciona os listeners
rb1.addItemListener(this);
rb2.addItemListener(this);
rb3.addItemListener(this);

// Cria o button group e adiciona os botoes
ButtonGroup grp = new ButtonGroup();
grp.add(rb1);
grp.add(rb2);
grp.add(rb3);

// Implementa a interface ItemListener
public void itemStateChanged(ItemEvent e) {
  JToggleButton button = (JToggleButton) e.getSource();
  if (e.getStateChange() == ItemEvent.SELECTED)
    System.out.println(button.getText()+ "selected.");
}
```

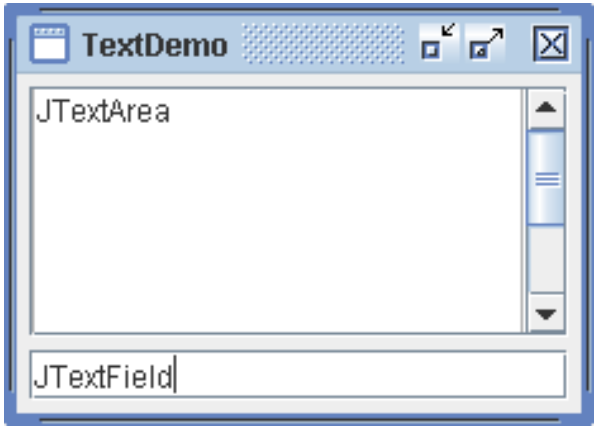
Campos de texto

- Derivam da classe JTextComponent
- Exibem textos que podem ser selecionados e editados
- Podem suportar textos estilizados e campos de password

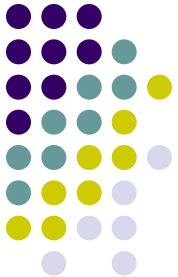


Campos de texto

- JTextField
  - Única linha para entrada de dados
- JTextArea
  - Permite múltiplas linhas
  - Permite a adição de painéis de scroll
    - JScrollPane

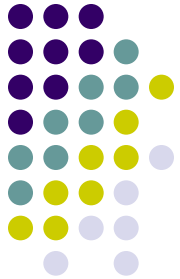


Campos de texto
Exemplo



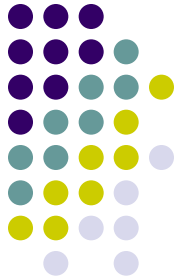
```
// Cria o textfield e adiciona o listener
textField = new JTextField(20);
textField.addActionListener(this);
// Cria a textarea, seta a borda e adiciona o scrollpane
textArea = new JTextArea("JTextArea");
textArea.setBorder(BorderFactory.createEtchedBorder());
scrollPane = new JScrollPane(textArea,
                    JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
                    JScrollPane.HORIZONTAL_SCROLLBAR_NEVER);

// Adiciona os componentes ao painel
JPanel panel = new JPanel(new BorderLayout(5, 5));
panel.setBorder(BorderFactory.createEmptyBorder(5, 5, 5, 5));
panel.add(scrollPane, BorderLayout.CENTER);
panel.add(textField, BorderLayout.PAGE_END);
// Implementa a interface ActionListener
public void actionPerformed(ActionEvent evt) {
   String text = textField.getText();
   textArea.append(text + "\n");
}
```

Exercícios
Entrega: 05/06



1) Crie uma aplicação que simula uma tela de *login*. Ela deve ter um JTextField para entrada do nome do usuário, um campo de senha, implementado pela classe JPasswordField, e um botão. Quando o usuário clica no botão, os dados são comparados com uma lista interna contendo 5 entradas e o programa exibe uma mensagem de sucesso se o usuário e a senha conferem, ou uma mensagem de erro caso contrário.
2) Crie um painel contendo quatro JToggleButtons, de tal forma que apenas um deles possa estar selecionado em um dado momento.

Exercícios
Entrega: 05/06



3) Crie uma aplicação com um painel e um botão. Quando o usuário clicar no botão, um objeto da classe JColorChooser é exibido e o usuário escolhe uma das cores. O painel deve ser pintado com a cor escolhida.

4) Crie um JLabel que carrega uma imagem. O programa deve imprimir em outro JLabel a posição (x,y) do *mouse* quando o usuário clicar sobre a imagem.