Desenvolvimento Desktop

Aula 03 - Componentes – Visão Geral e Componente Rótulo (JLabel)

Apresentação

Esta aula dará a você o suporte necessário para a utilização dos componentes Swing da linguagem Java utilizando o NetBeans. Além de uma visão geral sobre esses componentes Swing, você verá também como utilizar um dos componentes mais utilizados em qualquer aplicação, o **JLabel**, ou rótulo, e aplicar algumas de suas propriedades em uma aplicação. Outros componentes serão mostrados no decorrer das próximas aulas

Vídeo 01 - Apresentação

Objetivos

- Diferenciar componente nativo e componente Swing de um sistema operacional e identificar por que os componentes Swing tem a mesma aparência em sistemas operacionais diferentes
- Definir conceito de portabilidade.
- Organizar hierarquicamente os componentes Swing e saber o que são componentes Top-Level.
- Reconhecer elementos da classe **JComponent** e saber utilizar o componente JLabel.

Componentes Swing – Uma Visão Geral

Os componentes do Swing têm uma aparência própria, diferente dos componentes equivalentes que estão ligados diretamente a um sistema operacional, como os componentes do Windows, por exemplo. Veja o exemplo da **Figura 1**:

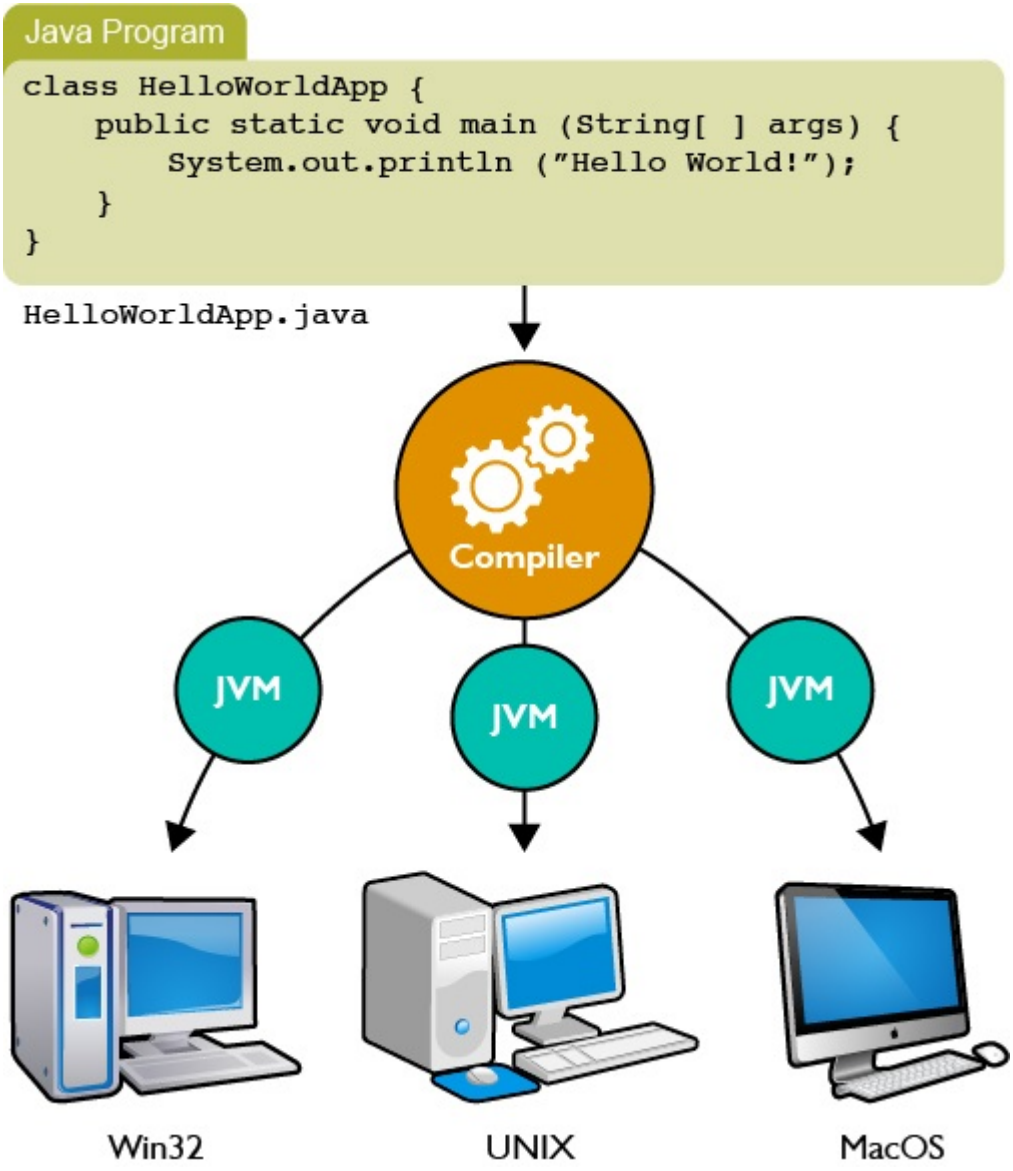
Figura 01 - Aparência dos componentes do Swing comparados aos componentes nativos do Windows

Componentes swing do Java | Componentes nativos do Windows

Perceba que, visualmente, os componentes são diferentes. Isso acontece porque no caso do Swing, os componentes visuais fazem parte da máquina virtual (JVM – Java Virtual Machine) e isso o deixa totalmente independente do sistema operacional que está por trás disso tudo.

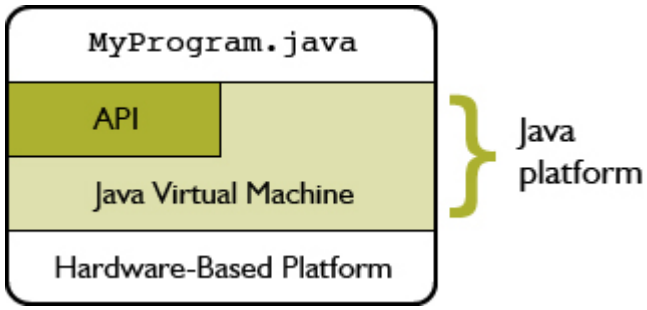
Como consequência, temos a vantagem de ter um programa feito em Java utilizando componentes Swing com, exatamente, a mesma aparência quando executado em diversos sistemas operacionais. A **Figura 2** mostra que nos vários sistemas operacionais (Windows, Unix, MacOS etc.) o ambiente encontrado pelo programa Java é o mesmo, ou seja, ele é compilado para ser executado em uma máquina virtual, não importando em que arquitetura computacional essa esteja inserida.

Figura 02 - A plataforma Java (JVM) permite manter a mesma aparência independentemente do Sistema Operacional (SO)



Já a figura a seguir (**Figura 3**) ilustra as camadas de software e hardware de um sistema computacional, destacando a máquina virtual como intermediária entre um programa feito em Java e o hardware que, efetivamente o executará. Como a figura sugere, fica a cargo da máquina virtual a tradução do programa Java para a plataforma na qual a JVM está instalada. Essa solução gera o que chamamos de **portabilidade**, que nada mais é do que a capacidade de se implementar e compilar um programa apenas uma vez e poder executá-lo em várias plataformas de hardware e software diferentes.

Figura 03 - Destaque da plataforma Java como intermediária entre o hardware e o software



Vídeo 02 - Componentes Swing

Usando Contêineres Top-level

Contêineres são componentes que têm a capacidade de incluir outros componentes no seu interior. É o mesmo conceito para os contêineres da vida real, aquelas enormes caixas utilizadas para armazenar mercadorias transportadas por navios e caminhões de um lugar para outro. O Swing provê três classes que são consideradas contêineres de mais alto nível (conhecidas também como contêineres top-level): **JFrame**, **JDialog** e **JApplet**. Isso quer dizer que:

- Para aparecer na tela, cada componente deve fazer parte de uma hierarquia de contêiner. Uma hierarquia de contêiner consiste em uma árvore de componentes que tem um contêiner top-level como raiz. Mostraremos um exemplo mais adiante.

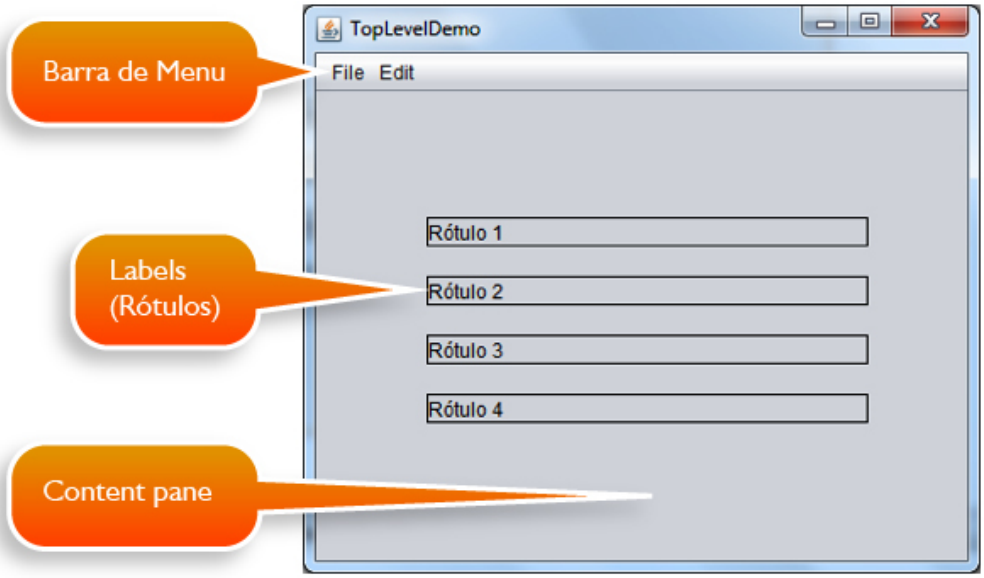
- Cada componente da GUI (Graphics User Interface, ou seja, Interface Gráfica do Usuário) deve ser inserido apenas uma vez. Se um componente já estiver em um contêiner e você tentar adicioná-lo em outro contêiner, este componente será removido do primeiro contêiner e adicionado ao segundo.

- Cada contêiner top-level tem um **content pane** que contém, direta ou indiretamente, os componentes visíveis daquele contêiner top-level.

- Você pode, opcionalmente, adicionar uma barra de menus a um contêiner top-level. A barra de menus é posicionada dentro do contêiner top-level, mas fora do **content pane**.

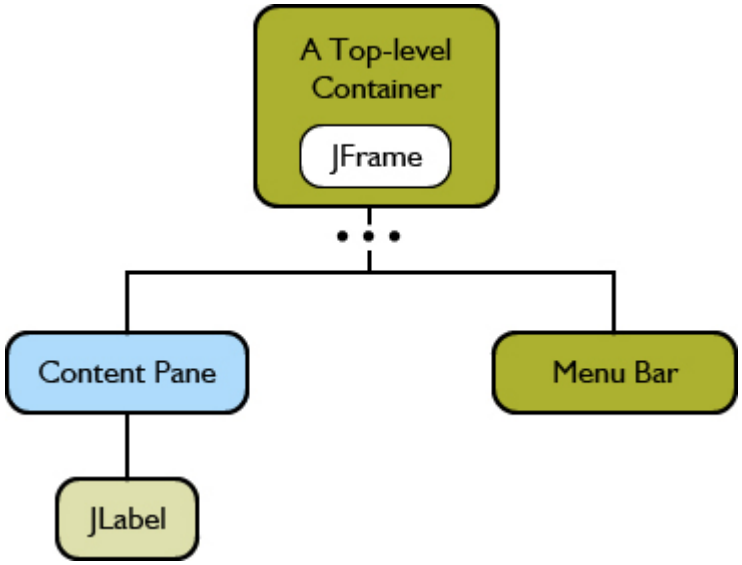
A **Figura 4** representa um **JFrame** criado por uma aplicação. Esse frame contém uma **barra de menus** (com as opções padrões) e, no **content pane** do frame, alguns **labels** (rótulos).

Figura 04 - JFrame mostrando a barra de menus e o content pane acomodando os labels



A **Figura 5** mostra a hierarquia de contêineres para a GUI da figura anterior.

Figura 05 - Hierarquia de contêineres utilizada no exemplo



Top-level Contêineres e Hierarquias de Contêineres

Todo programa que usa componentes do Swing tem pelo menos um contêiner top-level. Esse é a raiz da hierarquia de contêiner e esta última, por sua vez, contém todos os componentes do Swing que aparecerão dentro do contêiner top-level.

Uma aplicação para desktop cuja GUI utiliza o Swing tem pelo menos uma hierarquia de contêineres com um **JFrame** como sua raiz. Por exemplo, se uma aplicação tem uma janela principal e duas caixas de diálogo, então, a aplicação tem três hierarquias de contêiner. Uma que tem um **JFrame** como raiz e duas com um **JDialog** como raiz.

A Classe JComponent

Com exceção dos contêineres top-level, todos os componentes do Swing cujo nome começa com "**J**" são descendentes da classe **JComponent**. Por exemplo, os componentes **JPanel**, **JScrollPane**, **JButton** e **JTable** herdam todas as características de **JComponent**. Já o **JFrame** e o **JDialog** não, porque eles são contêineres top-level.

A classe **JComponent** provê as seguintes características a seus descendentes:

- **Tool tips**: essa propriedade lhe permite criar uma mensagem do tipo **String** (texto) no componente com a finalidade de mostrar ao usuário uma mensagem informativa sobre o seu conteúdo. Por exemplo, você poderia inserir uma mensagem do tipo: "**Informe o CPF no formato: 999.999.99-99**", quando o componente fosse selecionado.

- **Bordas e Desenho**: permite especificar as bordas de um componente, do tipo: borda de linha, borda composta, borda de título, entre outras. É possível também desenhar dentro de um componente.

- **Look-and-feel**: possibilita a mudança do visual e do comportamento de um componente.

- **Propriedades customizadas**: você poderá associar uma ou mais propriedades a cada **JComponent**.

- **Suporte a layout**: possibilita a mudança das características de layout dos componentes, tais como tamanho mínimo do componente, alinhamento etc.

- **Suporte a acessibilidade**: provê funcionalidades de apoio a tecnologias assistidas como leitores de tela terem acesso às informações do Swing.

- **Drag and drop**: provê recursos de arrastar e soltar os componentes em outras posições da tela.

- **Buffer duplo**: suaviza o aparecimento dos componentes na tela.

- **Key binding**: associa teclas do teclado a eventos dos componentes.

Componentes do Swing

Nesta aula será apresentado o mais, digamos assim, indispensável dos componentes: o rótulo (**JLabel**). Outros componentes serão apresentados nas próximas aulas. Começaremos pelos componentes mais usados, explicando seus principais métodos e propriedades, e, com o desenvolver das aulas, veremos alguns componentes de uso mais específico e que serão utilizados em estudos posteriores. Serão mostrados exemplos do uso de cada componente, assim como serão cobrados exercícios para todos eles. Para isso, utilizaremos o NetBeans para o desenvolvimento de todos os exercícios do presente curso.

Vídeo 03 - Container e Jcomponent

Atividade 01

---

1. Que característica da classe **JComponent** possibilita a mudança do visual e do comportamento de um componente?

2. Que componentes são considerados contêineres top-level e não herdam as mesmas características de **JComponent**?

3. O que você entende por contêiner?
4. Cite três componentes que herdam as mesmas características de **JComponent**.
5. Qual a finalidade do recurso **Tool** tips da classe **JComponent**?

O Componente JLabel (Rótulo)

Esse é um componente bastante comum. Podemos dizer até que ele está presente em toda interface gráfica. O rótulo, mais conhecido pelo termo **JLabel** no NetBeans, é, geralmente, utilizado para criar títulos, como também identificar campos de formulários em uma aplicação, entre outras funções. Veja um exemplo na **Figura 6**.

Figura 06 - Exemplos de utilização do componente JLabel

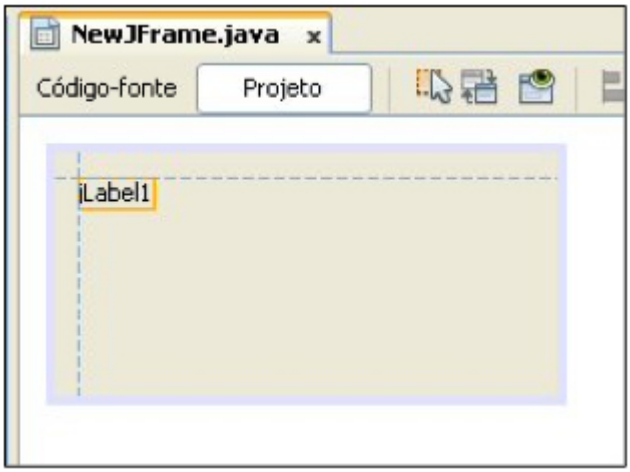
Como Implementar um Rótulo na minha Aplicação?

Acompanhe os procedimentos a seguir e veja como implementar, de uma maneira fácil, um rótulo em uma aplicação:

1. Execute o NetBeans e crie uma aplicação java, conforme foi explicado na aula "**Introdução ao Swing**". Para o nome do projeto informe: **LabelDemo** (apenas uma sugestão, pode ser o nome que você achar mais conveniente).
2. Em seguida, clique com o botão direito sobre o nome do projeto e adicione um **Formulário JFrame** ao projeto. Dê o mesmo nome para o formulário.

3. Após o projeto ser criado, selecione o **JFrame** e arraste um componente **Rótulo** da paleta **Controles Swing** para qualquer posição da tela, conforme mostra a (**Figura 7**):

Figura 07 - Inserindo um rótulo no JFrame



4. Observe as linhas tracejadas à medida que você arrasta o componente. Elas lhe auxiliam no alinhamento em relação ao formulário e outros componentes existentes nele (se houver).

Vejamos agora algumas propriedades que podem ser utilizadas nesse componente:

- **font** – com essa propriedade é possível configurar diversas outras propriedades em relação à fonte, como o tipo, estilo e tamanho.

- **foreground** – utilizado para alterar a cor da fonte.

- **horizontalAlignment** – permite o alinhamento horizontal do texto, da propriedade text em relação às dimensões do componente. Algumas opções disponíveis: LEFT, RIGHT e CENTER, entre outras.

- **verticalAlignment** – permite o alinhamento vertical do texto, da propriedade **text** em relação às dimensões do componente. As opções disponíveis para essa propriedade são: **BOTTOM**, **TOP** e **CENTER**.

- **icon** – permite inserir um ícone no rótulo. Formatos aceitáveis: **GIF**, **JPG** ou **PNG**.

- **border** – permite aplicar uma borda em volta do componente. Há varias opções como Borda de linha, Borda de título, Borda fosca, Borda composta, entre outras.

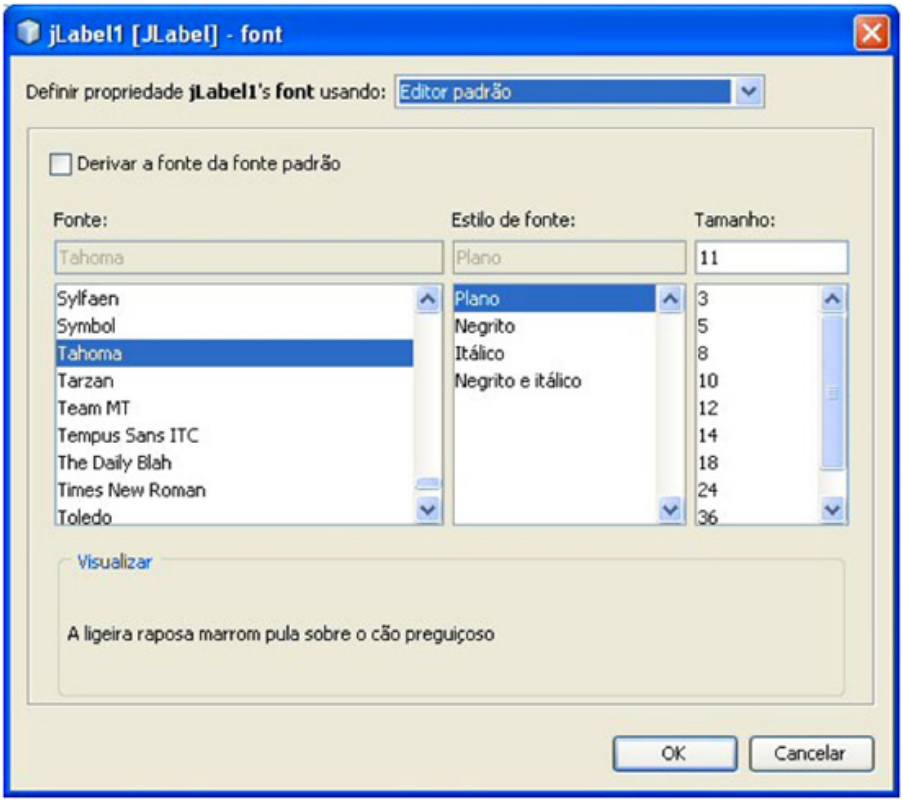
Como Mudar o Tipo, Estilo e Tamanho da Fonte

1. No painel **Propriedades**, clique na opção **font** e, em seguida, clique no pequeno botão (de reticências) à sua direita, conforme mostra a **Figura 8**:

Figura 08 - Alterando a propriedade font do componente rótulo

2. Na caixa de diálogo apresentada (**Figura 9**):

Figura 09 - Caixa de diálogo para alterar o tipo, o estilo e o tamanho da fonte

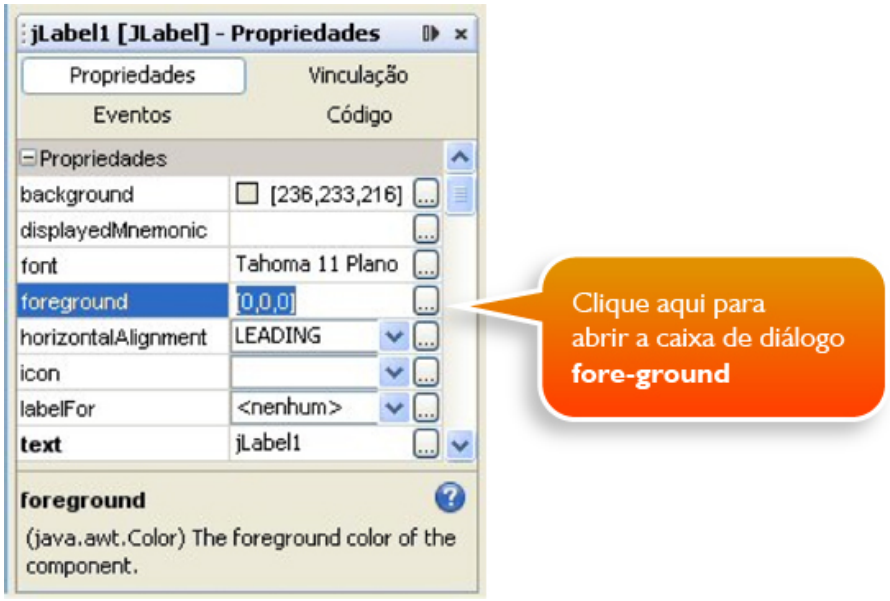


3. Selecione o tipo de fonte, o estilo e o tamanho que achar mais conveniente para o seu **rótulo**, em seguida clique em **OK** para confirmar.

Como Mudar a Cor do Rótulo

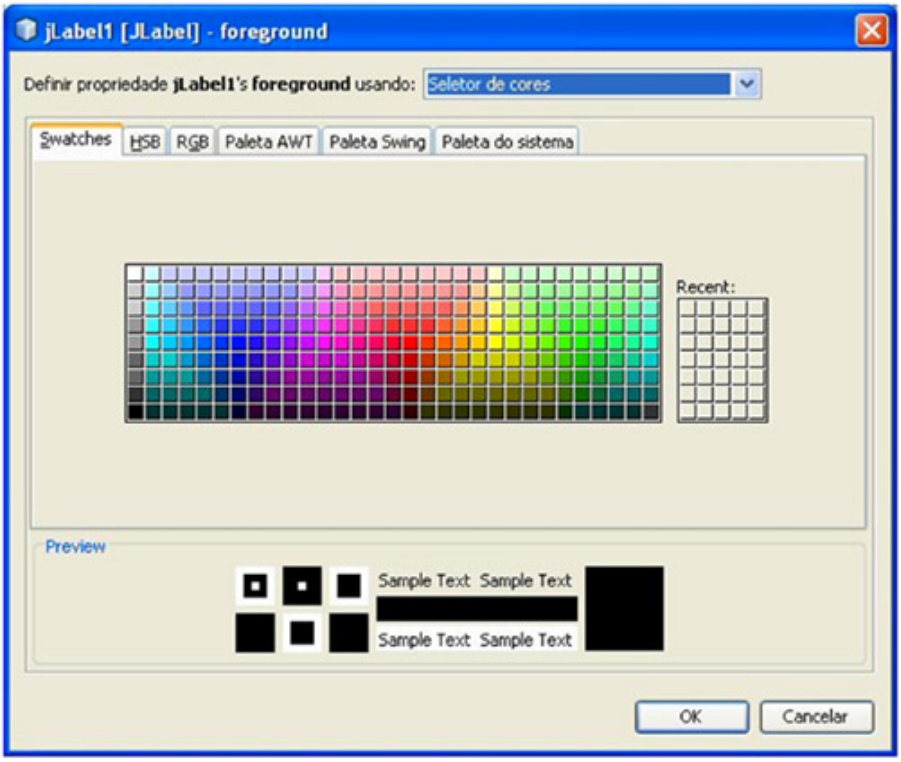
1. Clique na opção **foreground** e, em seguida, clique no pequeno botão (de reticências) à sua direita para abrir a caixa de diálogo, conforme mostra a **Figura 10**:

Figura 10 - Opção para alterar a propriedade foreground do rótulo



2. Na caixa de diálogo apresentada (**Figura 11**):

Figura 11 - Caixa de diálogo para alterar a cor do rótulo



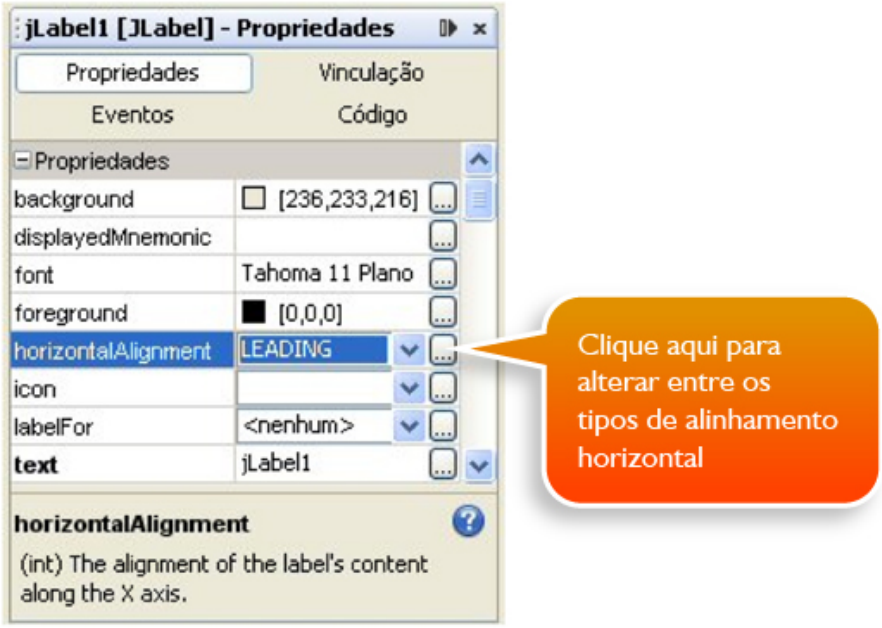
3. Selecione a cor que achar mais conveniente para o seu texto, em seguida, clique em **OK** para confirmar.

Obs.: Conforme vimos na propriedade **foreground**, se você já souber o código da cor que deseja, basta digitar os valores "**[0,0,0]**" diretamente na caixa de texto e apertar **Enter**. A cor da fonte será alterada para a cor especificada.

Como Alinhar o Texto na Horizontal

1. Localize a opção **horizontalAlignment** e, em seguida, clique na pequena seta mostrada na **Figura 12**:

Figura 12 - Alterando o alinhamento horizontal do rótulo



2. Dentre as principais opções temos:

    - **RIGHT** – essa opção permite ao usuário alinhar o texto à direita.
    - **LEFT** – essa opção permite alinhar o texto à esquerda.
    - **CENTER** – essa opção permite ao usuário alinhar o texto no centro da área definida para o **rótulo**.

3. Selecione o alinhamento que achar mais conveniente para o seu texto. Automaticamente, o texto ficará alinhado de acordo com a seleção.

Vídeo 04 - Jlabel

Atividade 02

1. Para alterarmos a cor dos caracteres de um rótulo (**JLabel**), precisamos utilizar a propriedade **font** desse componente. Essa afirmação é verdadeira ou falsa? Se falsa, que propriedade devemos utilizar?

2. Quais as opções disponíveis na propriedade **verticalAlignment** de um componente **rótulo**?

    a. LEFT, CENTER e RIGHT

    b. BOTTOM, TOP e CENTER

    c. RIGHT, BOTTOM e CENTER

    d. LEFT, CENTER e TOP

3. Que formato de imagem não é considerado aceitável para ser utilizado com a propriedade **icon** de um rótulo?

    a. JPG

    b. GIF

    c. IMG

    d. PNG

4. Responda **V** (Verdadeiro) ou **F** (Falso), de acordo com as situações referentes ao uso de um componente **JLabel** (rótulo):

    a. Para se alterar o tamanho da fonte utilizamos a propriedade **font**. ( )

    b. A cor da fonte pode ser alterada com a propriedade **text**. ( )

c. A propriedade **icon** só permite inserir um ícone no formato **.ico**. ( )

d. A propriedade **border** só permite a opção **Borda de linha**. ( )

Conclusão

Nas próximas aulas você vai conhecer outros componentes do NetBeans, suas propriedades mais importantes e como implementá-los.

Resumo

Nessa aula você viu como funciona o Swing e porque ele mantém a mesma aparência de suas aplicações em diferentes sistemas operacionais. Você aprendeu que portabilidade é a possibilidade de implementar e executar um programa uma única vez, gerando um aplicativo capaz de ser executado em plataformas diferentes. Viu também como os componentes do Swing se organizam, além de aprender um pouco sobre a classe **JComponent** e o que são os componentes Top-level. Por fim, você aprendeu como inserir um componente **Label** no formulário da sua aplicação e algumas de suas propriedades.

Autoavaliação

1. Quais as três classes de contêineres de mais alto nível (top-level)?
2. O que acontece se um componente já estiver em um contêiner e você tentar adicioná-lo em outro contêiner?
3. O que você entende por portabilidade?

Referências

Lesson: Using Swing Components. http://download.oracle.com/javase/tutorial/uiswing/components/index.html. Acesso em: 05 abr. 2012.

How to Use Labels. http://download.oracle.com/javase/tutorial/uiswing/components/label.html Acesso em: 05 abr. 2012.