Hortolândia, 30 de outubro de 2013.

Aulas 53 e 54 – Classes Gráficas em Java – Parte II

1. Introdução

A presente aula apresenta uma segunda forma de se trabalhar com Interfaces Gráficas em Java, que é por meio do uso da IDE NetBeans e seu suporte à criação de formulários.

A IDE NetBeans, na verdade, apenas facilita o acesso e a utilização das classes dos pacotes awt e Swing, vistos na última aula prática.

O NetBeans, no caso, oferece a comodidade de arrastar e soltar componentes a fim de se criar cada interface gráfica.

Primeiramente, na Seção 2, será demonstrado como se criar uma classe gráfica qualquer a partir do NetBeans.

Em seguida, na Seção 3, será criada uma aplicação de exemplo, contendo uma Tela de Cadastro, e uma classe principal Cadastro, a qual será responsável pela exibição dos dados obtidos do usuário na Tela de Cadastro.

A Seção 4 apresenta as referências bibliográficas utilizadas.

2. Criação de Aplicação Gráfica no NetBeans

2.1. Novo Projeto: AplicacaoGráfica

A fim de se demonstrar como utilizar o NetBeans para a criação de aplicações gráficas, será criado primeiramente um novo projeto.

Este novo projeto pode ser criado seguindo-se os seguintes passos:

Arquivo → Novo Projeto → Aplicação Java

Em seguida, define-se o nome da aplicação – no caso, chamou-se de "**AplicacaoGrafica**" e da classe principal. Neste exemplo, manteve-se a sugestão do NetBeans para a classe principal da aplicação: **aplicacaografica.AplicacaoGrafica**.

Até o momento, o código gerado pelo NetBeans para a aplicação será semelhante ao seguinte:

```
package aplicacaografica;

public class AplicacaoGrafica {

    public static void main(String[] args) {

        // TODO code application logic here

    }

}
```

A fim de se criar uma interface gráfica (uma tela gráfica), pode-se tomar o seguinte caminho:

a) Clicar com o botão direito no nome do pacote (no caso, "aplicacaografica");

b) Escolher a opção **Novo → Form JFrame**

> "Form JFrame" é como o NetBeans denomina a classe JFrame, do pacote Swing.
>
> A classe JFrame corresponde a uma janela gráfica, conforme visto na última aula prática.

c) Escolher o nome da classe gráfica (por exemplo, Tela1) e clicar em "Finalizar".

O NetBeans a seguir exibirá, no centro da IDE (onde normalmente exibe o código-fonte das classes), a representação gráfica de uma Tela vazia, contendo apenas o JFrame associado à mesma. Esta é a **área de desenho** (ou de **projeto**) da interface gráfica.

Além disso, no lado direito da tela, aparecerá um menu (**paleta**), contendo os diversos componentes gráficos do pacote Swing. Para se utilizar cada um desses elementos, basta selecionar o elemento com o botão esquerdo do mouse, e arrastá-lo para a área de desenho.

Se quisermos, por exemplo, adicionar um rótulo, um JTextField e um botão à interface gráfica, podemos fazê-lo, arrastando cada uma dos elementos citados para a área de desenho.

A Figura 1 ilustra a interface gráfica desejada neste primeiro exemplo.

Figura 1: Interface Gráfica de Exemplo.

O rótulo é chamado de "Label". Para se alterar o texto do rótulo, basta clicar-se duplamente sobre o texto do mesmo.

A classe JTextField é chamada de "Campo de Texto".

O botão é chamado de "Botão". Para programar a ação do botão, basta clicar duplamente por sobre o mesmo. O método de ação do botão (jButtonActionPerformed()) aparecerá na tela, permitindo a alteração de seu conteúdo.

Outra informação importante é que se pode chavear entre a visão de projeto (a aparência da classe gráfica, do JFrame) e a visão de código-fonte, bastando-se clicar, na área de projeto, nos botões "projeto" ou "código-fonte", respectivamente.

3. Aplicação de Cadastro de Usuário

3.1. Novo Projeto: CadastroUsuario

A presente Seção demonstra como se pode criar uma Tela de Cadastro, contendo diversos campos de textos, correspondentes a entradas de um cadastro de usuário.

Também se acrescentarão rótulos, para cada campo de texto, e dois botões: Limpar e Enviar.

Em seguida serão programadas as ações de cada botão.

A Interface Gráfica proposta está representada na Figura 2.

Figura 2: Tela de Cadastro de Usuário.

Para se criar o novo projeto, pode-se seguir os passos:

a) **Arquivo → Novo Projeto → Aplicação Java**

b) O nome da aplicação pode ser "**CadastroUsuario**".

c) A seguir, clicar com o botão direito no nome do pacote "**cadastrousuario**", e selecionar **Novo → Form JFrame**. Escolher um nome para a tela, por exemplo, **TelaCadastro**.

d) Arrastar os seguintes componentes, da paleta para dentro da área de desenho: quatro rótulos, três campos de texto e dois botões. Renomear os rótulos, de acordo com o que está representado na Figura 2.

3.2. Programação dos Botões

A etapa seguinte à da confecção da Interface Gráfica pode ser a de programação da ação dos botões.

Conforme ensinado na aula passada, cada botão ao ser acionado (clicado pelo usuário, quando a classe gráfica estiver em execução), gera no Java uma ação de botão acionado.

Cada botão, por sua vez, está associado a um "ouvidor de ações" (ActionListener), que se encarregará de verificar a ocorrência da ação de clique no botão. Assim que tal ação for detectada, o método **actionPerformed()** correspondente será executado.

A IDE NetBeans já deixa preparado o objeto ActionListener para cada botão existente na interface, restando apenas ao programador definir as ações necessárias no interior do método **actionPerformed()** correspondente ao botão.

E para se programar tal método, basta clicar-se no botão, dentro da área de desenho.

O conteúdo do método **actionPerformed( )** será exibido. A Listagem 2 exibe o código criado automaticamente pelo NetBeans quando se cria um novo botão.

```
    private   void   jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {

            // TODO add your handling code here:

        }
```

Listagem 2: Código-fonte do método jButton1ActionPerformed().

Para o botão **Limpar** (**jButton1**), as ações correspondentes são de limpar os campos de texto do formulário.

Considerando-se os valores padrão de nomes de variáveis de campos de texto, esses últimos terão os nomes **jTextField1**, **jTextField2** e **jTextField3**.

Portanto, o método actionPerformed() do botão **Limpar** ficará conforme representado na Listagem 3.

```
    private   void   jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {

            jTextField1.setText("");

            jTextField2.setText("");

            jTextField3.setText("");

        }
```

Listagem 3: Código-fonte do método jButton1ActionPerformed() (botão Limpar).

Para se testar o funcionamento da classe TelaCadastro, precisa-se completar a programação da classe principal da aplicação, que é CadastroUsuario.

O código-fonte da classe **CadastroUsuario** está representado na Listagem 4.

```
    package cadastrousuario;

    public class CadastroUsuario {

        public static void main(String[] args) {

           TelaCadastro cadastro = new TelaCadastro();

           cadastro.setVisible(true);

        }

    }
```

Listagem 4: Código-fonte da classe **CadastroUsuario**.

O método **setVisible()** com o argumento **true** faz com que o JFrame fique visível.

Para o botão de Enviar, a programação sugerida será ler os valores digitados pelo usuário, e imprimir na linha de comandos.

O método **jButtonActionPerformed**() correspondente ao botão **Enviar** está descrito na Listagem 5.

```
    private   void   jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {

            String nome, endereco, telefone;

            nome = jTextField1.getText();

            endereco = jTextField2.getText();

            telefone = jTextField3.getText();

            System.out.println("Nome: " + nome);

            System.out.println("Endereço: " + endereco);

            System.out.println("Telefone: " + telefone);

        }
```

Listagem 5: Código-fonte do método jButton2ActionPerformed() (botão Enviar).

Por fim, executar a classe principal (CadastroUsuario) e verificar o comportamento da aplicação.

4. Referências Bibliográficas

BORATTI, Isaias Camilo. **Programação Orientada a Objetos usando Delphi.** Quarta Edição. Editora Visual Books. Florianópolis, 2007.

DEITEL, H.M., DEITEL, P.J. **Java – Como programar**. Terceira edição. Porto Alegre: Bookman Editora, 2001.

SANTOS, Rafael. **Introdução à Programação orientada a objetos usando Java**. 8ª Reimpressão. Rio de Janeiro: Campus - Elsevier, 2003.