14.3 Entrada/saída baseada em GUI simples com JOptionPane

Os aplicativos nos capítulos 2 a 10 exibem o texto na janela de comando e obtêm a entrada a partir da janela de comando. A maioria dos aplicativos que você utiliza diariamente utiliza janelas ou **caixas de diálogo** (também chamadas de **diálogos**) para interagir com o usuário. Por exemplo, um programa de email permite digitar e ler mensagens em uma janela fornecida pelo programa. Em geral, as caixas de diálogo são janelas em que programas exibem mensagens importantes para o usuário ou obtêm informações do usuário. A classe **JOptionPane** do Java (pacote `javax.swing`) fornece caixas de diálogo pré-construídas tanto para entrada como para saída. Esses diálogos são exibidos invocando métodos `JOptionPane static`. A Figura 14.2 apresenta um aplicativo de adição simples que utiliza dois **diálogos de entrada** para obter inteiros do usuário e um **diálogo de mensagem** para exibir a soma dos inteiros que o usuário insere.

```
1  // Figura 14.2: Addition.java
2  // Programa de adição que utiliza JOptionPane para entrada e saída.
3  import javax.swing.JOptionPane; // programa utiliza JOptionPane
4
5  public class Addition
6  {
7     public static void main( String[] args )
8     {
9        // obtém a entrada de usuário a partir dos diálogos de entrada JOptionPane
10       String firstNumber =
11          JOptionPane.showInputDialog( "Enter first integer" );
12       String secondNumber =
13          JOptionPane.showInputDialog( "Enter second integer" );
14
15       // converte String em valores int para utilização em um cálculo
16       int number1 = Integer.parseInt( firstNumber );
17       int number2 = Integer.parseInt( secondNumber );
18
19       int sum = number1 + number2; // soma os números
20
21       // exibe o resultado em um diálogo de mensagem JOptionPane
22       JOptionPane.showMessageDialog( null, "The sum is " + sum,
23          "Sum of Two Integers", JOptionPane.PLAIN_MESSAGE );
24    } // fim do método main
25 } // fim da classe Addition
```

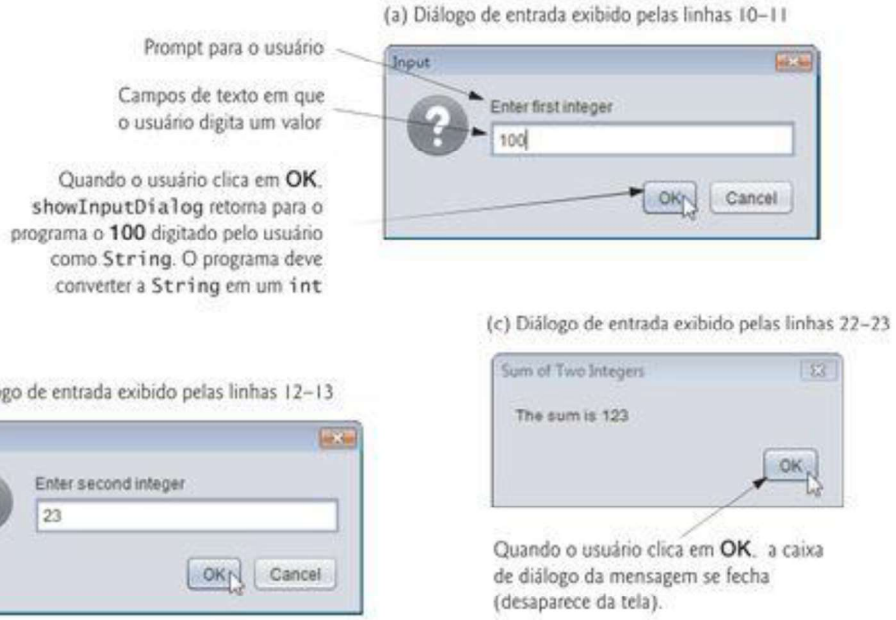

Figura 14.2 | Programa de adição que utiliza `JOptionPane` para entrada e saída.

Diálogos de entrada

A linha 3 importa a classe `JOptionPane`. As linhas 10–11 declaram a variável `String` local `firstNumber` e atribuem a ela o resultado da chamada ao método `JOptionPane` `static` **`showInputDialog`**. Esse método exibe um diálogo de entrada (ver a primeira captura de tela na Figura 14.2), utilizando o argumento `String` do método (`"Enter first integer"`) como um prompt.

Observações sobre a aparência e comportamento 14.2

Em geral, o prompt em um diálogo de entrada emprega ***maiúsculas e minúsculas no estilo de frases*** *— um estilo que emprega a maiúscula inicial apenas na primeira palavra da frase a menos que a palavra seja um nome próprio (por exemplo, Jones).*

O usuário digita caracteres no campo de texto, depois clica em OK ou pressiona a tecla *Enter* para enviar a `String` para o programa. Clicar em OK também **fecha (oculta) o diálogo**. [*Nota:* se você digitar no campo de texto e não aparecer nada, ative o campo de texto clicando nele com o mouse.] Ao contrário de `Scanner`, que pode ser utilizado para inserir valores de vários tipos do usuário no teclado, um diálogo de entrada pode inserir somente `String`s. Isso é comum na maioria dos componentes GUI. O usuário pode digitar qualquer caractere no campo de texto do diálogo de entrada. Nosso programa assume que o usuário insere um inteiro válido. Se o usuário clica em Cancel, `showInputDialog` retorna `null`. Se o usuário digitar um valor não inteiro ou clicar no botão Cancel no diálogo de entrada, ocorrerá uma exceção e o programa não irá operar corretamente. O Capítulo 11, "Tratamento de exceções", discutiu como tratar esses erros. As linhas 12–13 exibem outro diálogo de entrada que pede ao usuário para inserir o segundo inteiro. Note que cada diálogo `JOptionPane` que você exibe é a chamada **caixa de diálogo modal** — enquanto o diálogo está na tela, o usuário não pode interagir com o restante do aplicativo.

Observações sobre a aparência e comportamento 14.3

Não use excessivamente caixas de diálogo modais uma vez que elas podem reduzir a usabilidade dos aplicativos. Utilize uma caixa de diálogo modal somente quando for necessário impedir usuários de interagir com o restante de um aplicativo até que eles descartem o diálogo.

Convertendo `String`s em valores `int`

Para realizar o cálculo, convertemos as `String`s que o usuário inseriu em valores `int`. Lembre-se de que o método `static` `parseInt` da classe `Integer` converte seu argumento `String` em um valor `int`. As linhas 16–17 atribuem os valores convertidos às variáveis locais `number1` e `number2`. Em seguida, a linha 19 soma esses valores e atribui o resultado à variável local `sum`.

Diálogos de mensagem

As linhas 22–23 utilizam método `static` `JOptionPane` **`showMessageDialog`** para exibir um diálogo de mensagem (a última captura de tela da Figura 14.2) que contém a soma. O primeiro argumento ajuda o aplicativo Java a determinar onde posicionar a caixa de diálogo. Um diálogo é tipicamente exibido a partir de um aplicativo GUI em uma janela própria. O primeiro argumento refere-se a essa janela (conhecida como a janela pai) e faz com que o diálogo pareça centrado sobre a janela pai (como faremos na Seção 14.9). Se o primeiro argumento for `null`, a caixa de diálogo será exibida no centro da tela. O segundo argumento é a mensagem a exibir — nesse caso, o resultado da concatenação de `String` `"The sum is "` e do valor de `sum`. O terceiro argumento — `"Sum of Two Integers"` — é a `String` que deve aparecer na barra de título em cima do diálogo. O quarto argumento — **`JOptionPane.PLAIN_MESSAGE`** — é o tipo do diálogo de mensagem a exibir. O diálogo `PLAIN_MESSAGE` não exibe um ícone à esquerda da mensagem. A classe `JOptionPane` fornece várias versões sobrecarregadas dos métodos `showInputDialog` e `showMessageDialog`, bem como os métodos que exibem outros tipos de diálogo. Para informações completas sobre a classe `JOptionPane`, visite `java.sun.com/javase/6/docs/api/javax/swing/JOptionPane.html`.

Observações sobre a aparência e comportamento 14.4

Em geral, a barra de título de uma janela adota o uso de letras ***maiúsculas e minúsculas de título de livro*** *— um estilo que emprega a inicial maiúscula em cada palavra significativa no texto e não termina com pontuação (por exemplo, Uso de Letras Maiúsculas e Minúsculas no Título de um Livro).*

Constantes de diálogo de mensagem `JOptionPane`

As constantes que representam os tipos de diálogo de mensagem são mostradas na Figura 14.3. Todos os tipos de diálogo de mensagem exceto `PLAIN_MESSAGE` exibem um ícone à esquerda da mensagem. Esses ícones fornecem uma indicação visual da importância da mensagem para o usuário. Observe que um ícone `QUESTION_MESSAGE` é o ícone padrão de uma caixa de diálogo de entrada (ver Figura 14.2).

Tipo de diálogo de mensagem	Ícone	Descrição
`ERROR_MESSAGE`		Indica um erro ao usuário.
`INFORMATION_MESSAGE`		Indica uma mensagem informativa ao usuário.
`WARNING_MESSAGE`		Alerta o usuário de um potencial problema.
`QUESTION_MESSAGE`		Propõe uma questão 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.

Figura 14.3 | Constantes `JOptionPane` `static` para diálogo de mensagem.

14.4 Visão geral de componentes Swing

Embora seja possível realizar entrada e saída utilizando os diálogos `JOptionPane` apresentados na Seção 14.3, a maioria dos aplicativos GUI exige interfaces com o usuário mais elaboradas e personalizadas. O restante deste capítulo discute muitos componentes GUI que permitem aos desenvolvedores de aplicações criar GUIs robustas. A Figura 14.4 lista vários **componentes GUI Swing** do pacote **`javax.swing`** que são utilizados para construir GUIs Java. A maioria dos componentes Swing são componentes Java puros — eles são completamente escritos, manipulados e exibidos em Java. Eles fazem parte das **Java Foundation Classes (JFC)** — bibliotecas do Java para desenvolvimento de GUI para múltiplas plataformas. Visite `java.sun.com/javase/technologies/desktop/` para obter mais informações sobre tecnologias desktop JFC e Java.

Componente	Descrição
`JLabel`	Exibe texto não editável ou ícones.
`JTextField`	Permite ao usuário inserir dados do teclado. Também pode ser utilizado para exibir texto editável ou não editável.
`JButton`	Desencadeia um evento quando o usuário clicar nele com o mouse.
`JCheckBox`	Especifica uma opção que pode ser ou não selecionada.
`JComboBox`	Fornece 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.
`JList`	Fornece 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.
`JPanel`	Fornece uma área em que os componentes podem ser colocados e organizados. Também pode ser utilizado como uma área de desenho para imagens gráficas.

Figura 14.4 | Alguns componentes GUI básicos.

Swing versus AWT

Há realmente dois conjuntos de componentes GUI no Java. Antes de o Swing ter sido introduzido no J2SE 1.2, as Java GUIs eram construídas com componentes do **Abstract Window Toolkit (AWT)** no pacote **`java.awt`**. Quando um aplicativo Java com uma AWT GUI executa em diferentes plataformas Java, os componentes GUI do aplicativo exibem diferentemente em cada plataforma. Considere um aplicativo que exibe um objeto do tipo `Button` (pacote `java.awt`). Em um computador que executa o sistema operacional do Microsoft Windows, `Button` terá a mesma aparência dos botões de outros aplicativos Windows. De maneira semelhante, em um computador que executa o sistema operacional Mac OS X da Apple, `Button` terá a mesma aparência e comportamento dos botões em outros aplicativos Macintosh. Às vezes, a maneira como um usuário pode interagir com um componente AWT particular difere entre plataformas.