

Input Dialogs

Caixas de Entrada de Dados

Como foi introduzido anteriormente, em Java, podemos usufruir de classes e objetos do pacote Swing, o qual facilita a criação de interface gráfica sem a necessidade de ficar horas programando.

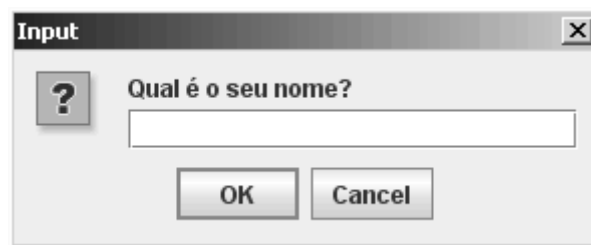
Neste artigo, nos aprofundaremos no uso de `JOptionPane`, começando a personalizar algumas caixas de diálogo mais simples.

Caixas de diálogo de entrada de dados são importantes para obter informações ou requisitar algo do usuário.

No objeto `JOptionPane`, há o método `showInputDialog()` que é responsável em criar uma caixa de diálogo requisitando uma entrada de dado. Este método é sobrecarregado de várias maneiras. A forma mais simples de seus argumentos é:

1. A mensagem que deve ser exibida para o usuário.

Com apenas este argumento é possível criar uma caixa de diálogo com o título `Input`, um ícone de interrogação, uma caixa de texto, uma mensagem e dois botões. Igual a figura abaixo:



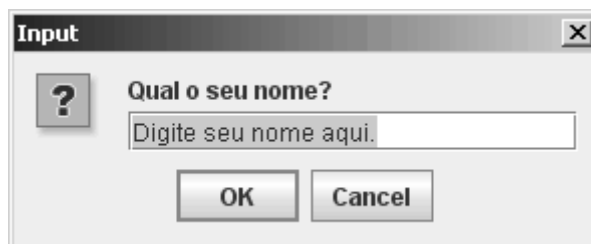
Porém, há como alterarmos a aparência dessa caixa, costumizando-a completamente.

Outra forma é utilizar dois argumentos, sendo:

1. A mensagem que deve ser exibida ao usuário.
2. O valor inicial da caixa de texto.

O valor inicial da caixa de texto é a string que deve aparecer dentro do campo onde digitamos nossa entrada. Assim que aparece a caixa, seu campo está preenchido com um valor inicial já selecionado.

Ex.: O código `JOptionPane.showInputDialog("Qual o seu nome?", "Digite seu nome aqui.")` geraria a seguinte caixa:



Uma das formas mais completas desse método inclui alterar, inclusive, o título da caixa. Assim, usa-se 4 argumentos:

1. De qual frame a caixa de diálogo é dependente, ou seja, qual a janela principal que chamou a caixa `Input Dialog`. Caso a caixa de diálogo não dependa de nenhum frame ou janela principal, basta utilizarmos o valor `null` para este argumento.
2. A mensagem que deve ser exibida ao usuário.
3. O título da caixa de texto.
4. Que tipo de mensagem é. O tipo de mensagem define qual o ícone será utilizado, podendo ser utilizados os números inteiros representados pelas constantes:

- PLAIN_MESSAGE (valor: -1): Mensagem limpa, sem nenhum ícone.
- ERROR_MESSAGE (valor: 0): Mensagem de erro.
- INFORMATION_MESSAGE (valor: 1): Mensagem informativa.
- WARNING_MESSAGE (valor: 2): Mensagem de alerta.
- QUESTION_MESSAGE (valor: 3): Mensagem de requisição ou pergunta. Esta é a opção padrão do método `showInputDialog()`.

Ex.: O código `JOptionPane.showInputDialog(null, "Qual o seu Nome?", "Pergunta", JOptionPane.PLAIN_MESSAGE)` geraria a seguinte caixa:



Obter valor de `showInputDialog`

O método `showInputDialog` pode retornar dois valores: ou uma string ou null.

Se o botão OK for clicado a string contida na caixa de texto será retornada, se o botão Cancel for clicado o valor null será retornado. Sabendo disso, podemos usar uma variável string para obter o valor e tratarmos da forma que quisermos. Vejamos o exemplo abaixo:

```
import javax.swing.JOptionPane;

public class CaixasDeInput {
    public static void main(String[] args) {
        String nome = null;
        while (nome == null || nome.equals("")) {
            nome = JOptionPane.showInputDialog("Qual o seu nome?");
            if (nome == null || nome.equals("")) {
                JOptionPane.showMessageDialog(null,
                    "Você não respondeu a pergunta.");
            }
        }
        JOptionPane.showMessageDialog(null, "Seu nome é " + nome);
    }
}
```

Input Dialog com lista de opções

Outra forma de caixa de diálogo de entrada de dados é a Input Dialog com lista de opções.

É o mesmo método `showInputDialog`, mas com mais argumentos, sendo um deles uma lista de objetos. Esta lista de objetos fará com que a caixa de diálogo venha com um combo box ao invés de um campo de texto.

Para criar um Input Dialog com um combo box devemos usar os seguintes argumentos na respectiva ordem:

1. De qual frame a caixa de diálogo é dependente, ou seja, qual a janela principal que chamou a caixa Input Dialog. Caso a caixa de diálogo não dependa de nenhum frame ou janela principal, basta utilizarmos o valor null para este argumento.
2. A mensagem que deve ser exibida ao usuário.
3. O título da caixa de texto.
4. Que tipo de mensagem é. O tipo de mensagem define qual o ícone será utilizado, podendo ser utilizados os números inteiros representados pelas constantes da mesma forma como foi mostrada anteriormente.

5. O quinto argumento é representado pelo objeto Icon, que é um ícone que podemos criar a partir de um jpg, gif, png, etc. O objeto Icon será comentado com mais detalhes nos próximos artigos.
6. O segredo do combo box está neste argumento. Aqui virá um array (vetor) de objetos que serão nossos valores pré-definidos.
7. O último argumento serve apenas para indicar qual elemento do array (vetor) deve vir selecionado no início. Caso não desejarmos que um ítem seja selecionado no início basta utilizarmos null.

O array (vetor) de objetos deve ser genérico, portanto, utilizamos a classe Object para criar este array.

O método showInputDialog com combo box se diferencia do showInputDialog com caixa de texto pelo seguinte fato: o que é retornado dessa vez não será uma string, mas um objeto. Isso faz sentido se percebermos que agora estamos escolhendo um item dentro de uma lista de objetos. Portanto, o que será retornado será um objeto dessa lista, não uma string como acontecia com o Input Dialog com caixa de texto.

Então, se quisermos utilizar o objeto genérico como algum outro tipo de dado, devemos antes fazer uma indução de tipo ou typecasting.

Vejam os exemplos abaixo:

```
import javax.swing.JOptionPane;

public class CaixaComComboBox {
    public static void main(String[] args) {
        Object[] opcoes = { "sim", "não" };
        Object resposta;
        do {
            resposta = JOptionPane.showInputDialog(null,
                "Deseja finalizar o programa?",
                "Finalização",
                JOptionPane.PLAIN_MESSAGE,
                null,
                opcoes,
                "não");
        } while (resposta == null || resposta.equals("não"));
    }
}
```

No exemplo acima, criamos uma lista com dois objetos: "sim" e "não". E já definimos "não" como opção pré-selecionada. Removemos qualquer ícone com as opções PLAIN_MESSAGE e ícone null. Criamos um laço (loop) com while que sempre irá repetir a mesma caixa enquanto o botão Cancel ou a opção "não" forem selecionados.

Abaixo está um exemplo mais funcional, divertido e simples de fazer e entender. Trata-se de um jogo de adivinhar onde um número é escolhido e temos que adivinhar qual número foi sorteado. No exemplo, serão usados elementos já estudados no site como a classe Math e Integer.

```
import javax.swing.JOptionPane;

public class JogoDeAdivinhar {
    public static void main(String[] args) {
        // define um número qualquer entre 0 e 10
        int rndNr = (int) Math.ceil(Math.random() * 10);
        // lista de opções para o combo box da caixa de diálogo
        Object[] opcoes = { "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
            "10" };
        // string onde será retornado o resultado
        String resposta;
```



```
while (true) {
// loop para evitar que o jogo feche depois da primeira resposta
  resposta = (String) JOptionPane.showInputDialog(null,
      "Em que número estou pensando?", "Jogo de Advinhar",
      JOptionPane.QUESTION_MESSAGE, null, opcoes, null);
  if (resposta == null) {
    /*
     * se clicar no botão Cancel, mostrar uma mensagem de Game Over
     * e sair do loop para finalizar o programa
     */
    JOptionPane.showMessageDialog(null,
        "Game Over!\nVocê desistiu do jogo!");
    break;
  }
  if (Integer.valueOf(resposta) > rndNr) {
    /*
     * Interpreta string como inteiro e compara com o número sorteado
     * para ver se é maior
     */
    JOptionPane.showMessageDialog(null,
        "Errado!\nO número que eu pensei é menor.");
  } else if (Integer.valueOf(resposta) < rndNr) {
    /*
     * Interpreta string como inteiro e compara com o número sorteado
     * para ver se é maior
     */
    JOptionPane.showMessageDialog(null,
        "Errado!\nO número que eu pensei é maior.");
  } else {
    /*
     * se não for nem maior e nem menor, então é igual.
     * Finaliza o jogo saindo do loop
     */
    JOptionPane.showMessageDialog(null,
        "Parabéns\nVocê adivinhou!\n"
        + "Eu realmente pensei no número " + rndNr);
    break;
  }
}
}
```



Autor: Denys William Xavier

Este artigo está sob Licença Creative Commons

*Para ver uma cópia desta licença, visite <http://creativecommons.org/licenses/by-nc-sa/2.5/br/>
ou envie uma carta para Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.*