

The logo for Oracle Academy. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is centered on a light gray background, which is framed by dark gray horizontal bars at the top and bottom.

ORACLE

Academy

Java Foundations

3-5

Entrada do Teclado

ORACLE
Academy



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

Objetivos

- Esta lição abrange os seguintes objetivos:
 - Entender a entrada do usuário
 - Criar um JOptionPane para coletar a entrada do usuário
 - Usar um Scanner para coletar a entrada do console
 - Usar um Scanner para coletar a entrada de um arquivo
 - Entender como um Scanner trata tokens e delimitadores



Por Que Você Deve Obter uma Entrada do Usuário?

- Quando você atribui manualmente valores a variáveis, esse procedimento é conhecido como hard-coding de valores:

```
String input = "Isto é uma String";
```

- É possível alterar facilmente valores submetidos a hard-code porque você tem o código-fonte e um IDE Java:

```
String input = "Isto é uma String diferente";
```

- Mas, quando você distribui um software, seus usuários não podem se dar a esse luxo

Tipos de Entrada de Usuário

- Exemplos de entrada de usuário incluem o seguinte...
 - Pressionar um botão em um controlador de jogo
 - Inserir um endereço em um GPS
 - Inserir números e funções em uma calculadora
 - Informar seu nome às pessoas
- Mas sem entrada do usuário...
 - Quando o jogo fará com que seu personagem pule?
 - Aonde seu GPS guiará você?
 - Que números sua calculadora triturrará?
 - Do que as pessoas chamarão você?

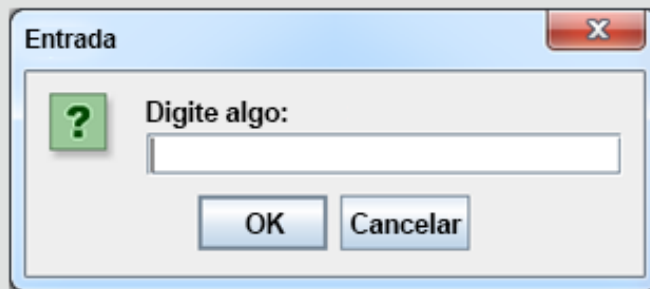
Como Obter a Entrada do Usuário

- Existem várias maneiras de obter a entrada do usuário:
 - Botões (físicos ou virtuais)
 - Discos e mostradores
 - Reconhecimento de voz
 - Caixas de diálogo de texto
 - Arquivos de propriedade
- O Java oferece muitas maneiras de obter entrada do usuário, inclusive...
 - Swing JOptionPane
 - JavaFX (um sucessor do Swing, abordado mais adiante)
 - Scanner

JOptionPane

- Essa é uma maneira simples de obter entrada dos usuários:

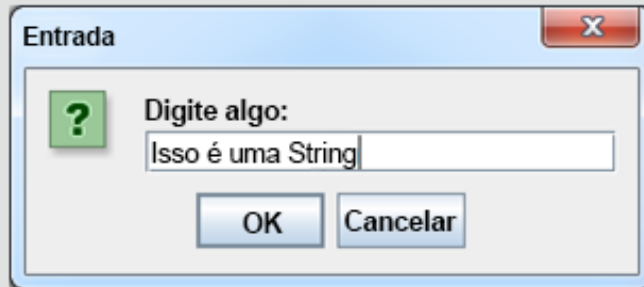
```
JOptionPane.showInputDialog("Digite algo:");
```



O JOptionPane Retorna Strings

- A entrada pode ser armazenada como uma String:

```
String input = JOptionPane.showInputDialog("Digite algo:");
```



- Isso é equivalente a escrever:

```
String input = "Isto é uma String";
```


Exercício 1, Parte 1

- Crie um novo projeto e adicione o arquivo `Input01.java` a ele
- Crie um `JOptionPane`:
 - O NetBeans reclamará
 - Siga a sugestão do NetBeans para importação do `javax.swing.JOptionPane`
 - Abordaremos a importação em outra seção

Exercício 1, Parte 2

- Armazene essa entrada como uma String
- Imprima a variável da String
- Faça parse da String como uma variável int separada
 - Você precisará inserir um valor que possa ser analisado
 - Imprima esse valor +1
- Tente criar uma caixa de diálogo, fazendo parse dela e inicializando um int em uma linha individual
- Só pode haver um ponto e vírgula (;)

Código Condensado

- Você poderia propagar sua entrada, fazendo parse e calculando em várias linhas:

```
String inputString = JOptionPane.showInputDialog("??");  
int input = Integer.parseInt(inputString);  
input++;
```

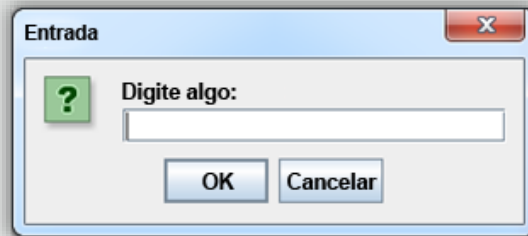
- Ou condensá-la em uma única linha:

```
int input = Integer.parseInt(JOptionPane.showInputDialog("??")) + 1;
```

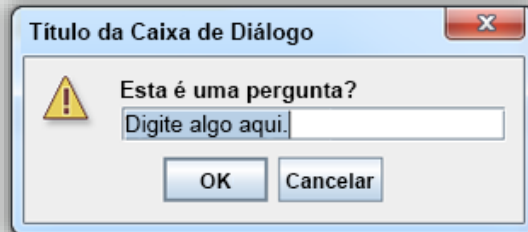
- Essa opção é uma questão de preferência pessoal
 - Mas, se você precisar fazer referência a determinados valores novamente mais tarde, convém armazená-los em uma variável

InputDialogs (caixas de diálogo de entrada) Diferentes

- Criamos uma InputDialog simples:



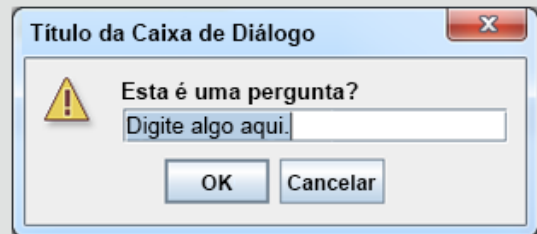
- Com um código mais complexo, podemos personalizar mais a InputDialog:



Mais Opções com InputDialogs

- Esta versão de uma InputDialog não retorna uma String
- O resultado deve ser convertido em uma String para que possa ser utilizado:

Conversão

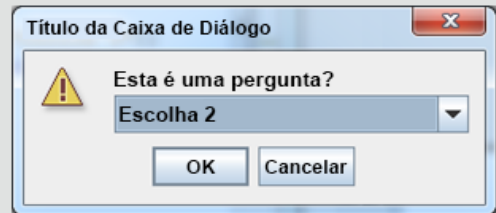


```
String input = (String)JOptionPane.showInputDialog(null,  
"Esta é uma pergunta?",  
"Título da Caixa de Diálogo",  
2,  
null,  
null,  
"Digite algo aqui.");
```

Você está confuso com este código? Não se preocupe. Mesmo os programadores experientes podem ficar confusos quando veem um novo código. Uma maneira muito útil de desenvolver seu conhecimento é modificar o código existente e observar o que acontece. Faremos isso no próximo exercício.

Mais Opções com InputDialogs

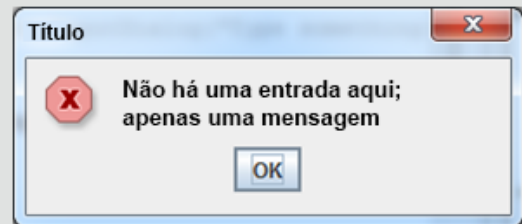
- Para evitar uma entrada indesejada, é possível fornecer somente valores aceitáveis para os usuários
- Parte dessa sintaxe será discutida em detalhes na Seção 8



```
String[] acceptableValues = {"Choice 1", "Choice 2", "Choice 3"};
Input2 string= (String)JOptionPane.showInputDialog(null,
    "Esta é uma pergunta?",
    "Título da Caixa de Diálogo",
    2,
    null,
    acceptableValues,
    acceptableValues[1]);
```


showMessageDialog

- Uma showMessageDialog não fornece um campo de entrada
- Existem muitas outras variações de JOptionPane



```
JOptionPane.showMessageDialog(  
    null,  
    "Não há uma entrada aqui; apenas uma mensagem",  
    "Título",  
    0);
```

Exercício 2

- Crie um novo projeto e adicione o arquivo `Input02.java` a ele
- Teste o código e tente alterar...
 - O título da mensagem
 - A mensagem
 - Qualquer texto de entrada padrão 
 - O ícone da caixa de diálogo
- Faça parse, manipule e imprima qualquer entrada

Dica: Ignore os nulos. Se você precisar de ajuda, a documentação do Java poderá ser útil:
<https://docs.oracle.com/en/java/javase/17/docs/api/java.desktop/javax/swing/package-summary.html>.

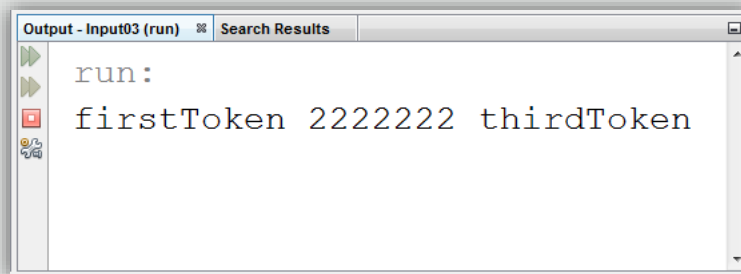
Obtendo Entrada com um Scanner

- Um objeto Scanner abre um fluxo para coleta da entrada:
 - System.in lê o Scanner para coletar entrada do console
 - Digite a entrada na janela de saída de seu IDE
 - Também é possível usar o Scanner sem um IDE
- Uma boa prática é fechar o fluxo do Scanner quando terminar

```
public static void main(String[] args) {  
    Scanner sc = new Scanner(System.in);  
  
    sc.close();  
}//fim do método main
```

Lendo a Entrada com um Scanner

- O Scanner procura tokens
- Os tokens são separados por um delimitador
 - O delimitador padrão é um espaço



The screenshot shows a window titled "Output - Input03 (run)" with a "Search Results" tab. The output text is as follows:

```
run:  
firstToken 2222222 thirdToken
```

A Classe Scanner

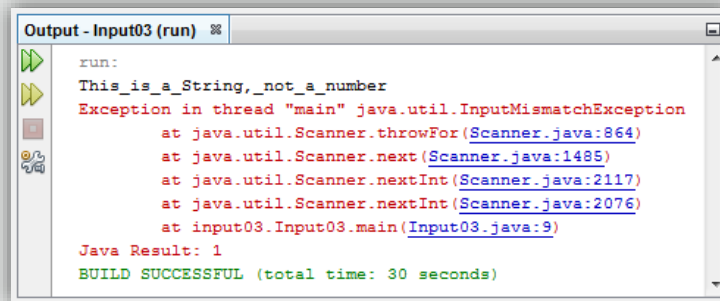
- A classe Scanner, assim como qualquer outra, tem campos e métodos
- Alguns métodos úteis do Scanner...
 - nextInt() lê o próximo token como um int
 - nextDouble() lê o próximo token como um double
 - next() lê o próximo token como uma String

```
public static void main(String[] args) {  
    Scanner sc = new Scanner(System.in);  
    int x = sc.nextInt();  
    double y = sc.nextDouble();  
    String z = sc.next();  
    sc.close();  
} //fim do método main
```

Exercício 3

- Crie um novo projeto e adicione o arquivo `Input03.java` a ele
- Crie um Scanner:
 - Your IDE will complain
 - Seu IDE reclamará
 - Siga a sugestão de seu IDE para importação do `java.util.Scanner`
- Use Scanner e `System.in` para escrever um programa que...
 - Localize e imprime a soma de três números inteiros inseridos pelo usuário
- Tente inserir menos que três tokens
- Tente inserir um token que pode ser analisado como `int`

Exceções: InputMismatchException



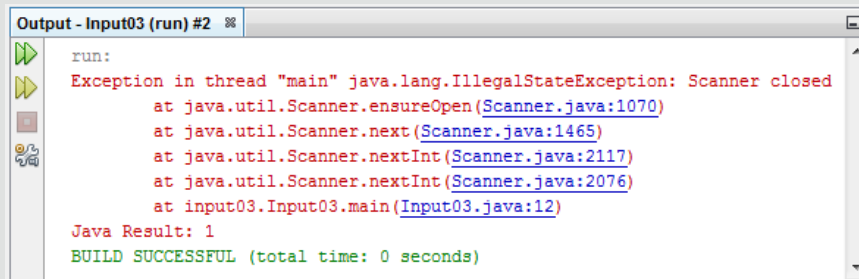
The screenshot shows a Java IDE output window titled "Output - Input03 (run)". The output text is as follows:

```
run:
This_is_a_String,_not_a_number
Exception in thread "main" java.util.InputMismatchException
    at java.util.Scanner.throwFor(Scanner.java:864)
    at java.util.Scanner.next(Scanner.java:1485)
    at java.util.Scanner.nextInt(Scanner.java:2117)
    at java.util.Scanner.nextInt(Scanner.java:2076)
    at input03.Input03.main(Input03.java:9)
Java Result: 1
BUILD SUCCESSFUL (total time: 30 seconds)
```

- Ocorre porque a entrada não pode ser analisada como o tipo esperado:

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println(sc.nextInt());
    sc.close();
} //fim do método main
```

Exceções: IllegalStateException



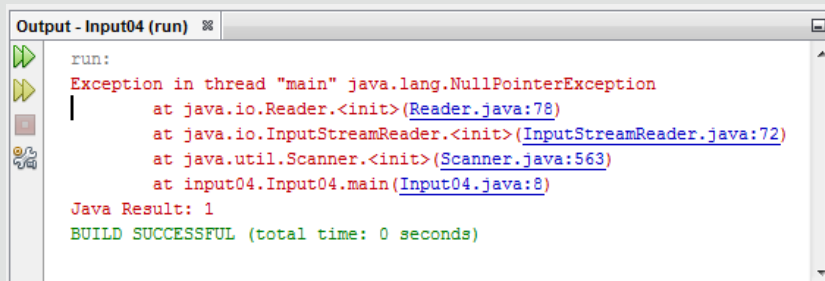
The screenshot shows an IDE output window titled "Output - Input03 (run) #2". It displays the following text:

```
run:
Exception in thread "main" java.lang.IllegalStateException: Scanner closed
    at java.util.Scanner.ensureOpen(Scanner.java:1070)
    at java.util.Scanner.next(Scanner.java:1465)
    at java.util.Scanner.nextInt(Scanner.java:2117)
    at java.util.Scanner.nextInt(Scanner.java:2076)
    at input03.Input03.main(Input03.java:12)
Java Result: 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Ocorre porque o fluxo é acessado depois de ter sido fechado:

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    sc.close();
    System.out.println(sc.nextInt());
} //fim do método main
```

Exceções: NullPointerException



```
Output - Input04 (run)
run:
Exception in thread "main" java.lang.NullPointerException
    at java.io.Reader.<init>(Reader.java:78)
    at java.io.InputStreamReader.<init>(InputStreamReader.java:72)
    at java.util.Scanner.<init>(Scanner.java:563)
    at input04.Input04.main(Input04.java:8)
Java Result: 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Ocorre porque "fakeFile.txt" não existe. Também é um erro comum esquecer a extensão .txt

```
public static void main(String[] args) {
    Scanner sc = new Scanner(
        Input04.class.getResourceAsStream("fakeFile.txt"));
    sc.close();
} //fim do método main
```

Lembre-se da extensão

Lendo de um Arquivo

- O Java oferece várias maneiras de ler arquivos
- Os métodos Scanner mais úteis incluem o seguinte:
 - `nextLine()` avança esse scanner de volta à linha atual e retorna a entrada que foi ignorada
 - `findInLine("String a ser Localizada")` Tenta localizar a próxima ocorrência de um padrão construído com base na String especificada ignorando delimitadores

```
public static void main(String[] args) {  
    Scanner sc = new Scanner(  
        Input04.class.getResourceAsStream("fakeFile.txt"));  
    int x = sc.nextInt();  
    String entireLine = sc.nextLine();  
    sc.close();  
} //fim do método main
```


Exercício 4, Parte 1

- Crie um novo projeto e adicione o arquivo `Input04.java` a ele
- Execute o código e examine a saída
- Leia cada linha até encontrar "BlueBumper"
- Os dois números após o "**BlueBumper**" são `xPositon` e `yPosition` do objeto
- Armazene essas coordenadas como números inteiros e imprima-as
- Examine `input04text.txt`, se necessário

Exercício 4, Parte 2

- Examine `Level105.txt` se estiver curioso:
 - É assim que os dados de nível são armazenados para o Java Puzzle Ball
 - Ler e fazer parse dos dados de nível é um pouco mais complicado do que você fez neste exercício
 - Mas, se terminou este exercício, você está perto de entender como ele é feito

Resumo

- Nesta lição, você deverá ter aprendido a:
 - Entender a entrada do usuário
 - Criar um JOptionPane para coletar a entrada do usuário
 - Usar um Scanner para coletar a entrada do console
 - Usar um Scanner para coletar a entrada de um arquivo
 - Entender como um Scanner trata tokens e delimitadores



The Oracle Academy logo is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by two horizontal dark gray bars, one at the top and one at the bottom.

ORACLE

Academy