

# ILP-010 – Linguagem de Programação

---

ADS – Análise e Desenvolvimento de Sistemas  
Prof. Vagner Macedo

Aula 04  
Entrada e saída de dados

# Saída de Dados pelo Console

- Para mostrar uma informação no console, devemos utilizar o objeto `System.out`;
  - A função `print(String mensagem)` mostra o texto recebido entre os parênteses no console;
  - A função `println(String mensagem)` mostra o texto recebido entre os parênteses no console, quebrando linha ao final.

# Saída de Dados pelo Console

```
1 package jappentradadedados;
2
3 public class Main {
4     public static void main(String[] args) {
5         System.out.print("O rato");
6         System.out.print(" roeu a roupa");
7         System.out.println(" do rei de roma");
8
9         System.out.println("Ser ou não ser?");
10        System.out.println("...eis a questão!");
11    }
12 }
```

Saída - jappEntradaDeDados (run-single)

```
init:
deps-jar:
Compiling 1 source file to C:\Documents and Settings\ricard
compile-single:
run-single:
O rato roeu a roupa do rei de roma
Ser ou não ser?
...eis a questão!
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)
```

# Entrada de Dados pelo Console

- A entrada de dados por console deve ser feita através do objeto `System.in`;
- A alternativa mais simples é utilizar somente um objeto do tipo `Scanner` (em conjunto com o objeto `System.in`);
- A classe `Scanner` deve ser importada no início do arquivo.

# Entrada de Dados pelo Console

```
2 package exemplo;
3
4 import java.util.Scanner;
5
6 public class EntradaDados {
7
8     public static void main(String[] args) {
9         // declara leitor para entrada de dados via texto
10        Scanner leitor = new Scanner(System.in);
11        // faz a leitura de dados em formatos diferentes
12        System.out.println("Informe seu nome: ");
13        String nome = leitor.next();
14        System.out.println("Informe sua idade: ");
15        int idade = leitor.nextInt();
16        // exibe os valores
17        System.out.println("Nome: " + nome + "; " + idade + " anos.");
18    }
19 }
```

# Entrada de Dados pelo Console

- Uma alternativa mais robusta e que prevê tratamento de erros também pode ser feita com os objetos `Scanner` e `System.in`;
- Um objeto `BufferedReader` representa a entrada de dados através do teclado;
- Neste caso, para garantir maior robustez, é necessário tratar uma possível exceção (“exception”). Essa estrutura condicional será vista em detalhes no futuro.

# Entrada e Saída de Dados utilizando Swing

- A biblioteca Swing oferece algumas funções muito práticas para ler e mostrar dados para o usuário
  - O JOptionPane é uma classe que oferece funções prontas que não precisam de tratamento de I/O:
    - showMessageDialog;
    - showInputDialog.

# Saída de Dados utilizando Swing

- A função `showMessageDialog` possui três variações, com diferentes configurações de parâmetros:

```
JOptionPane.showMessageDialog
```

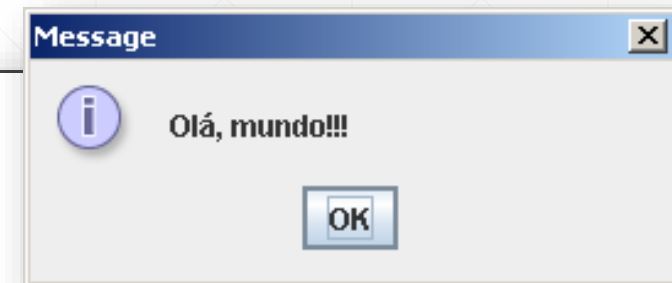
```
• showMessageDialog(Component parentComponent, Object message)  
• showMessageDialog(Component parentComponent, Object message, String title,  
• showMessageDialog(Component parentComponent, Object message, String title,
```



# Saída de Dados utilizando Swing

- A primeira variação recebe entre parênteses o componente pai (não nos preocuparemos com isso por enquanto e enviaremos sempre “null”) e a mensagem que deverá ser mostrada;

```
1 package jappentradadedados;
2
3 import javax.swing.JOptionPane;
4
5 public class Main {
6     public static void main(String[] args) {
7         JOptionPane.showMessageDialog(null, "Olá, mundo!!!");
8     }
9 }
```

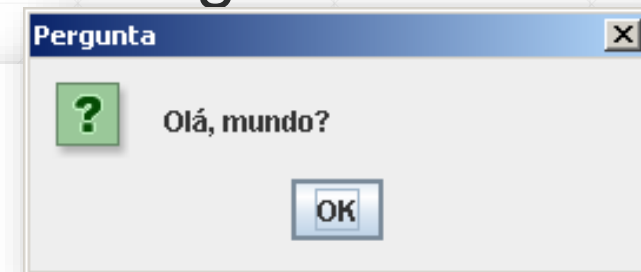


# Saída de Dados utilizando Swing

- Se quisermos incrementar a mensagem, podemos usar a variação que envia entre os parênteses também o título da janela e o tipo de mensagem

```
1  package jappentradadedados;
2
3  import javax.swing.JOptionPane;
4
5  public class Main {
6      public static void main(String[] args) {
7          JOptionPane.showMessageDialog(
8              null
9              , "Olá, mundo?"
10             , "Pergunta"
11             , JOptionPane.QUESTION_MESSAGE
12             );
13      }
14  }
```

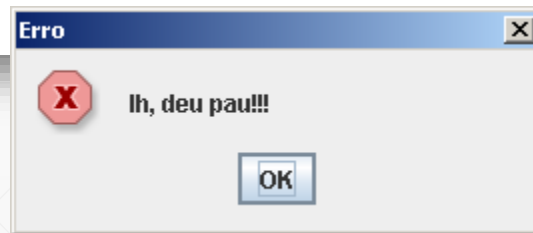
//Mensagem  
//Título da janela  
//Tipo de mensagem



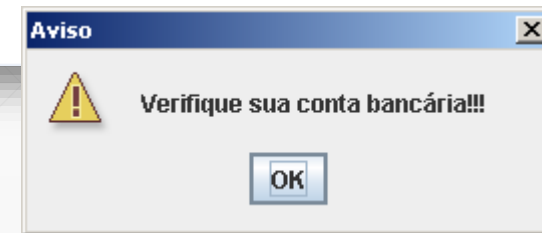
# Saída de Dados utilizando Swing

- Outros tipos de mensagem são Informação (padrão), Questão, Aviso, Erro e Vazio

```
1 package jappentradadedados;
2
3 import javax.swing.JOptionPane;
4
5 public class Main {
6     public static void main(String[] args) {
7         JOptionPane.showMessageDialog(
8             null
9             , "Ih, deu pau!!!"
10            , "Erro"
11            , JOptionPane.ERROR_MESSAGE
12        );
13    }
14 }
```



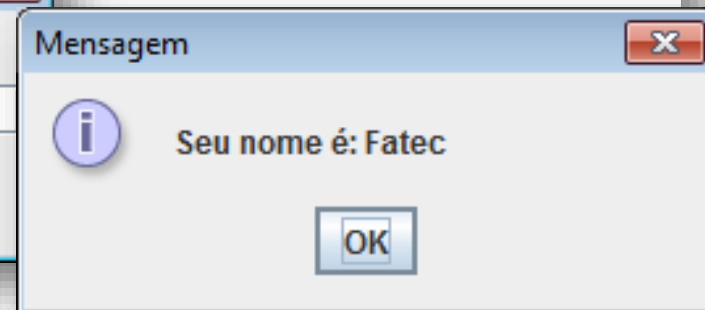
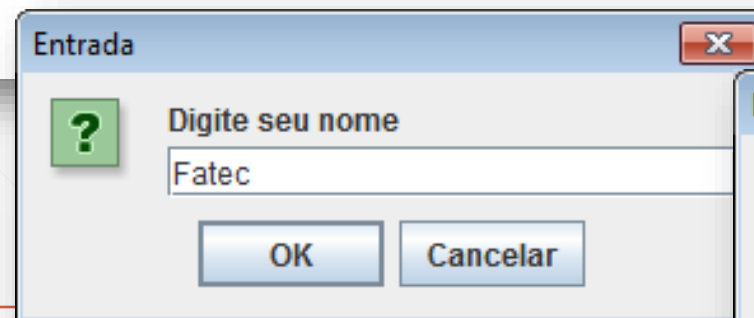
```
1 package jappentradadedados;
2
3 import javax.swing.JOptionPane;
4
5 public class Main {
6     public static void main(String[] args) {
7         JOptionPane.showMessageDialog(
8             null
9             , "Verifique sua conta bancária!!!"
10            , "Aviso"
11            , JOptionPane.WARNING_MESSAGE
12        );
13    }
14 }
```



# Entrada de Dados utilizando Swing

- A entrada de dados pode ser feita utilizando a função `showInputDialog`

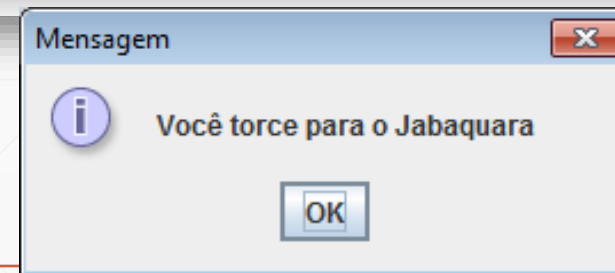
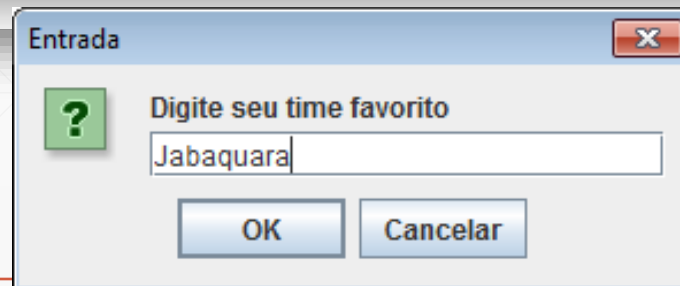
```
1  package jappentradadedados;
2
3  import javax.swing.JOptionPane;
4
5  public class Main {
6      public static void main(String[] args) {
7          String entrada = JOptionPane.showInputDialog("Digite seu nome");
8          JOptionPane.showMessageDialog(null, "Seu nome é: " + entrada);
9      }
10 }
```



# Entrada de Dados utilizando Swing

- Você pode utilizar uma variação do `showInputDialog()` para definir um valor padrão de entrada

```
1 package jappentradadedados;
2
3 import javax.swing.JOptionPane;
4
5 public class Main {
6     public static void main(String[] args) {
7         String entrada = JOptionPane.showInputDialog("Digite seu time favorito", "Jabaquara");
8         JOptionPane.showMessageDialog(null, "Você torce para o " + entrada);
9     }
10 }
```



# Conversão de Tipo

- No `showInputDialog`, as entradas sempre são `Strings`. Para convertê-las para os tipos de dados pretendidos, existem classes específicas:

```
1  package jappentradadedados;
2
3  import javax.swing.JOptionPane;
4
5  public class Main {
6      public static void main(String[] args) {
7          String entrada = JOptionPane.showInputDialog("Digite um valor");
8          int inteiro = Integer.parseInt(entrada);
9          float pontoFlutuante = Float.parseFloat(entrada);
10         double realPreciso = Double.parseDouble(entrada);
11         boolean lógico = Boolean.parseBoolean(entrada);
12     }
13 }
```

# Conversão de Tipo

- Na entrada de dados, você pode também fazer a conversão de tipo na mesma linha, melhorando a redigibilidade do código.

```
1 package jappentradadedados;
2
3 import javax.swing.JOptionPane;
4
5 public class Main {
6     public static void main(String[] args) {
7         String nome = JOptionPane.showInputDialog(null, "Digite seu nome:");
8         int idade = Integer.parseInt(JOptionPane.showInputDialog(null, "Digite sua idade:"));
9         JOptionPane.showMessageDialog(null, "Seu nome é " + nome + " e você tem " + idade + " anos.");
10    }
11 }
```

# Conversão de Tipo

- Também é possível fazer conversão de tipos via type-casting, que podem ser implícitos ou explícitos:

<b>PARA:</b>	<b>byte</b>	<b>short</b>	<b>char</b>	<b>int</b>	<b>long</b>	<b>float</b>	<b>double</b>
<b>DE:</b>							
<b>byte</b>	----	<i>Impl.</i>	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
<b>short</b>	(byte)	----	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
<b>char</b>	(byte)	(short)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
<b>int</b>	(byte)	(short)	(char)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
<b>long</b>	(byte)	(short)	(char)	(int)	----	<i>Impl.</i>	<i>Impl.</i>
<b>float</b>	(byte)	(short)	(char)	(int)	(long)	----	<i>Impl.</i>
<b>double</b>	(byte)	(short)	(char)	(int)	(long)	(float)	----

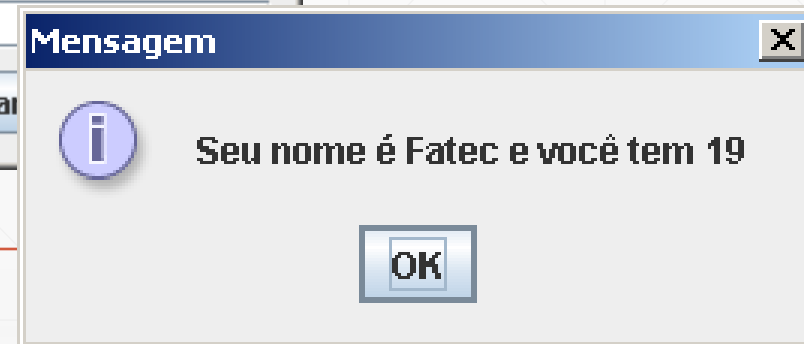
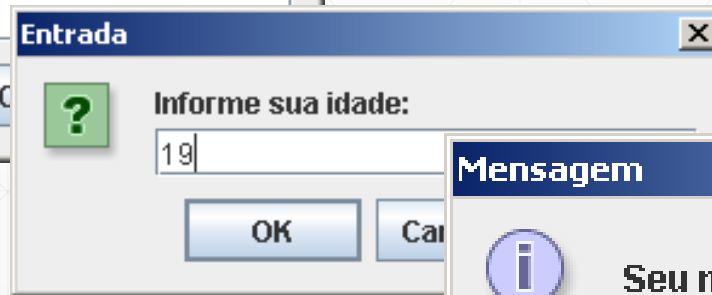
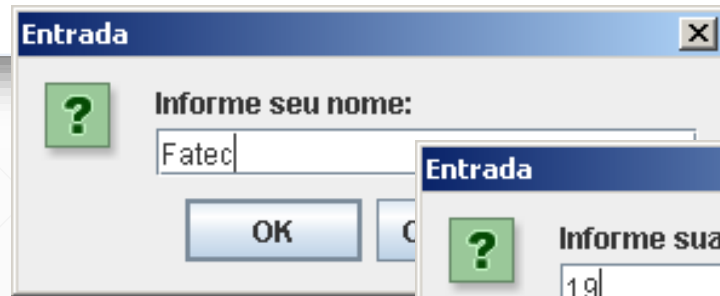


# Concatenação de Dados

- Em JAVA, a “adição” de Strings com outros tipos de dados é interpretada automaticamente como concatenação e, portanto, a conversão de outros tipos para String muitas vezes não é necessária.
- No exemplo a seguir, vemos a concatenação de Strings ser feita com o comando “String.format”. Também é possível ocorrer com o operador “+”. Outra forma bastante utilizada é com a classe StringBuilder.

# Concatenação de Dados

```
1 package entradadedados;
2
3
4 import javax.swing.JOptionPane;
5
6 public class Principal {
7
8     public static void main(String[] args) {
9         String nome = JOptionPane.showInputDialog("Informe seu nome: ");
10        String idadeTemp = JOptionPane.showInputDialog("Informe sua idade: ");
11        int idade = Integer.parseInt(idadeTemp);
12        JOptionPane.showMessageDialog(null, String.format("Seu nome é %s e você tem %d", nome, idade));
13    }
14
15 }
```



**Obrigado!**

