

Introdução ao Desenvolvimento Orientado a Objetos

**Conceitos Básicos da Linguagem
Java - Parte I**

A Linguagem Java

- ▷ Java é uma **linguagem de programação orientada a objetos**
 - Desenvolvida na empresa **Sun Microsystems** na década de 90 por uma equipe de programadores chefiada por James Gosling
 - Java foi **adquirida** pela empresa **Oracle Corporation** em **2008**.
 - Java é **compilada** para um **bytecode** que é **interpretado por uma máquina virtual** (*Java Virtual Machine*, abreviada **JVM**).

Máquina Virtual Java

- ▷ Em muitas linguagens de programação, o código quando é compilado produz um executável em linguagem de máquina.
- ▷ O Java utiliza o conceito de máquina virtual. Esta máquina virtual é uma camada que fica entre o programa e o sistema operacional e traduz os comandos do programa para o sistema operacional em que ela está no momento.
 - Para que rode na máquina virtual, o código Java é compilado para uma linguagem intermediária conhecida pelo nome de bytecode.

Máquina Virtual Java

- ▷ A máquina virtual Java JVM (Java Virtual Machine) gerencia memória, threads, a pilha de execução, dentre outras tarefas.
- ▷ Dessa forma, a aplicação é executada sem nenhum contato direto com o sistema operacional. A aplicação se comunica apenas com a JVM.

Máquina Virtual Java

- ▷ A JVM utiliza uma tecnologia chamada Hotspot para detectar pontos da sua aplicação que são executados com frequência repetidas vezes.
 - Dessa forma, a JVM pode optar por compilar esse trecho para código nativo da plataforma para melhorar o desempenho.
 - O compilador utilizado para isso é conhecido como JIT (Just in Time Compiler)

Obtendo o Java

- ▷ Como baixar o Java?
 - JRE: Java Runtime Environment.
 - Ambiente de execução Java, formado pela JVM e bibliotecas.
 - É o que você precisa para executar uma aplicação Java.
 - JDK: Java Development Kit.
 - Utilizado por desenvolvedores
 - É formado pelo JRE somado às ferramentas como o compilador.

Variáveis e Constantes

Declaração e Atribuição

Variáveis

- ▷ Java é uma linguagem fortemente tipada com suporte à tipificação dinâmica.
- ▷ Para declararmos uma variável em Java, precisamos fornecer o seu tipo.
 - Uma vez declarada uma variável em Java, apenas valores do tipo declarado podem ser atribuídos à variável em questão.

Variáveis

- Uma variável em Java pode ser declarada da seguinte forma:

`tipo identificador;`

Variáveis

- Uma variável em Java pode ser declarada da seguinte forma:

tipo identificador;

- Tipo de dado da variável.
 - Identifica que tipo de valor esta variável pode armazenar ou referenciar.

Variáveis

- ▷ Uma variável em Java pode ser declarada da seguinte forma:

tipo **identificador**

- Nome ou identificador da variável.
 - Utilizado para quando se deseja ler ou atualizar o valor da variável.

Variáveis

- ▶ Uma variável em Java pode ser declarada da seguinte forma:

tipo identificador;

- O ponto e vírgula identifica o fim do comando.
 - Não se esqueça de utilizá-lo.

Tipos Primitivos

- ▷ Java possui 8 tipos primitivos. Em azul estão destacados os tipos que utilizaremos nesta disciplina:
 - `boolean`
 - `byte`
 - `short`
 - `int`
 - `long`
 - `char`
 - `float`
 - `double`

Tipos Primitivos

		Valores possíveis				
Tipos	Primitivo	Menor	Maior	Valor Padrão	Tamanho	Exemplo
Inteiro	byte	-128	127	0	8 bits	byte ex1 = (byte)1;
	short	-32768	32767	0	16 bits	short ex2 = (short)1;
	int	-2.147.483.648	2.147.483.647	0	32 bits	int ex3 = 1;
	long	-9.223.372.036.854.770.000	9.223.372.036.854.770.000	0	64 bits	long ex4 = 1l;
Ponto Flutuante	float	-1,4024E-37	3.40282347E + 38	0	32 bits	float ex5 = 5.50f;
	double	-4,94E-307	1.79769313486231570E + 308	0	64 bits	double ex6 = 10.20d; ou double ex6 = 10.20;
Caractere	char	0	65535	\0	16 bits	char ex7 = 194; ou char ex8 = 'a';
Booleano	boolean	false	true	false	1 bit	boolean ex9 = true;

Declarando Variáveis

- ▷ É possível declarar uma variável para armazenar a idade de uma pessoa em anos da seguinte forma:

```
int idade;
```

- ▷ Já a variável para armazenar o salário de uma pessoa pode ser declarada da seguinte forma:

```
double salario;
```

Operadores

O operador de atribuição. Operadores aritméticos.

Operador de Atribuição

- ▷ Após declarar uma variável, é possível atribuir um valor à mesma utilizando para isso o operador de atribuição (=).

```
idade = 20;
```

Operador de Atribuição

- ▷ Após declarar uma variável, é possível atribuir um valor à mesma utilizando para isso o operador de atribuição (=).

idade = 20;

- Operador de atribuição.
 - À esquerda deste operador deve ser informada a variável que receberá o valor.
 - À direita deste operador deve ser informada a expressão cujo valor resultante será atribuído à variável.

Operador de Atribuição

- ▷ Após declarar uma variável, é possível atribuir um valor à mesma utilizando para isso o operador de atribuição (=).

idade = 20

- Valor a ser atribuído.
 - Neste caso foi atribuído um valor constante.
 - Constante é o valor que não é alterado durante a execução do programa.
 - Note que a variável pode ter o seu valor alterado, porém o número 20 fornecido sempre terá o mesmo valor.

Operador de Atribuição

- ▷ Após declarar uma variável, é possível atribuir um valor à mesma utilizando para isso o operador de atribuição (=).

idade = 20

- O valor a ser atribuído pode ser uma constante, uma variável, uma chamada a uma função ou uma expressão envolvendo constantes, variáveis, operadores e/ou chamadas à funções.

Operador de Atribuição

- ▷ É possível atribuir valor a uma variável na declaração da mesma:

```
int idade = 20;
```

Operador de Atribuição

- ▷ Exemplos de atribuições:

`int idade = 20;`

- Constante

Operador de Atribuição

- ▷ Exemplos de atribuições:

```
int idade = 20;
```

```
int idade2 = idade;
```

- Variável

Operador de Atribuição

- ▷ Exemplos de atribuições:

```
int idade = 20;
```

```
int proximaldade = obterIdade();
```

- Função

Operador de Atribuição

- ▷ Exemplos de atribuições:

```
int idade = 20;
```

```
int proximaldade = idade + 1;
```

- Expressão

Operadores Numéricos

- ▷ Você pode utilizar os seguintes operadores:
 - **+** (soma)
 - **-** (subtração)
 - ***** (multiplicação)
 - **/** (divisão)
 - **%** (resto da divisão)

Impressão

Apresentando valores para o usuário

Impressão

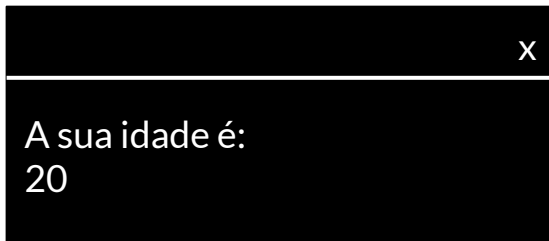
- ▷ Para apresentar um valor na tela do terminal, utilize o comando `System.out.println(VALOR_A_SER_IMPRESSO);`
 - O valor a ser impresso pode ser uma constante, uma variável, o retorno de uma função ou outra expressão qualquer.

Impressão

- ▷ Veja o código abaixo:

```
int idade = 20;  
System.out.println("A sua idade é: ");  
System.out.println(idade);
```

- ▷ Após a sua execução, será impresso:



A screenshot of a Java application window. The window has a title bar with a close button (X) in the top right corner. The main content area of the window displays the text "A sua idade é:" followed by the number "20" on the next line.

Impressão

- ▷ Agora note a seguinte alteração::

```
int idade = 20;  
System.out.print("A sua idade é: ");  
System.out.println(idade);
```

- ▷ Após a sua execução, será impresso:



A sua idade é: 20



Qual a diferença entre
print e *println*?

Constantes

Constantes ou Literais

Constantes

- ▷ Até o momento falamos de constantes, porém não especificamos como representar um valor constante nos tipos mais utilizados.
- ▷ Uma constante do tipo
 - int é representada por um número inteiro
 - Exemplo: 20
 - double é representada por um número com casas decimais (separador .)
 - Exemplo: 12.5

Constantes

- ▷ Uma constante do tipo
 - int é representada por um número inteiro
 - Exemplo: 20
 - double é representada por um número com casas decimais (separador .)
 - Exemplo: 12.5
 - boolean é representada pelos valores *true* ou *false*
 - char é representada pelo caractere entre aspas simples
 - Exemplo: 'a'

Exercícios

Atenção: ainda não foi explicada a leitura de dados, logo cada resposta para esses exercícios ainda não deve solicitar ao usuário nenhuma interação com o sistema após a execução do mesmo.

1) Faça um programa que imprima o seu nome na primeira linha e o seu sobrenome na segunda linha.

2) Faça um programa que calcule e imprima o Índice de Massa Corporal (IMC) de uma pessoa que possui 1,60 metros de altura e pesa 60 quilos.

$$\text{IMC} = \text{peso} / (\text{altura}^2)$$

Conversões

Conversão implícita e explícita

Conversões implícitas e explícitas

- ▶ Uma conversão implícita pode ocorrer quando atribuímos um int a um double, pois o conjunto dos inteiros (int) está contido no conjunto dos números reais (double).

```
double salario = 20000;  
System.out.println(salario);
```

Conversões implícitas e explícitas

- ▷ O contrário do exemplo anterior não pode ser feito:

```
int idade = 20.1;  
System.out.println(idade);
```



Erro!

Conversões implícitas e explícitas

- ▷ Para este caso, precisamos de uma conversão explícita:

```
int idade = (int) 20.1;  
System.out.println(idade);
```

- ▷ As casas decimais serão descartadas e a variável inteira ficará com o valor 20.

Entrada

Obtendo valores informados pelo usuário

Entrada de Dados

- ▷ Para trabalharmos com entrada de dados, utilizaremos a classe Scanner.
 - Para isso, você deve importar a classe Scanner do pacote java.util:

```
import java.util.Scanner;  
  
public class Main {  
    public static void main(String[] args) {  
    }  
}
```

Entrada de Dados

- ▷ O primeiro passo é instanciar a classe Scanner passando para o construtor da mesma a entrada padrão do sistema.

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

Entrada de Dados

- ▷ Veja como é fácil a entrada de dados:

```
nome = sc.nextLine();  
  
idade = sc.nextInt();  
  
salario = sc.nextDouble();
```

Entrada de Dados

- ▷ Veja como é fácil a entrada de dados:

```
nome = sc.nextLine();
```

```
idade = sc.nextInt();
```

```
salario = sc.nextDouble();
```

- O método `nextLine` retorna uma `String` informada pelo usuário no terminal.

Entrada de Dados

- ▷ Veja como é fácil a entrada de dados:

```
nome = sc.nextLine();
```

```
idade = sc.nextInt();
```

```
salario = sc.nextDouble();
```

- O método `nextInt()` retorna um `int` informado pelo usuário no terminal.

Entrada de Dados

- ▷ Veja como é fácil a entrada de dados:

```
nome = sc.nextLine();
```

```
idade = sc.nextInt();
```

```
salario = sc.nextDouble();
```

- O método `nextDouble()` retorna um `double` informado pelo usuário no terminal.

Exercícios

3) Faça um programa que leia o nome e o sobrenome de uma pessoa (nome em uma variável e sobrenome em outra variável). O programa deve imprimir o nome completo da pessoa com um espaço entre o nome e o sobrenome.

4) Faça um programa que leia o peso e a altura da pessoa e calcule e imprima o Índice de Massa Corporal (IMC) da mesma.

$$\text{IMC} = \text{peso} / (\text{altura}^2)$$

Formatando a Impressão

Imprimindo dados em formatos mais adequados

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- Nesta função é possível informar uma string contendo flags que identificam o tipo de formatação.
- Além das flags, também é possível informar o tamanho e a precisão.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- String contendo dados a serem impressos e, quando aplicável, seus marcadores, tamanho e precisão.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- Flag que indica que nesta posição o dado será impresso como uma string.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- Flag que indica que nesta posição será impresso um número inteiro.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- %f é a flag que indica que nesta posição será impresso um número real.
 - As outras informações entre o % e o f serão apresentadas à seguir.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- Tamanho mínimo a ser ocupado pela informação. Se a informação a ser impressa ultrapassar este tamanho, a informação continuará sendo impressa com todos os seus caracteres.
- Se a informação a ser impressa for menor que este tamanho, serão utilizados zeros à esquerda até completar o tamanho mínimo.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

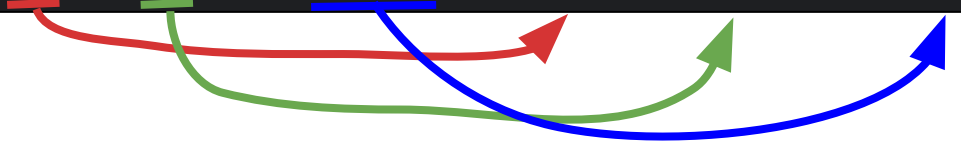
```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```

- Precisão que será utilizada na impressão.
 - O número que vem após o ponto indica o número de casas decimais.

Formatar Impressão

- ▷ É possível imprimir dados formatados com a função *printf*:

```
System.out.printf("%s - %d - R$%1.2f", nome, idade, salario);
```



- O conteúdo a ser impresso é passado por parâmetro como expressões após a string.
- A ordem em que os flags aparecem no texto indicam qual expressão será impressa nos mesmos.

Formatar Impressão

- ▷ Quebra de linha pode ser feita utilizando o flag %n:

```
System.out.printf("Primeira linha%nSegunda linha");
```

Formatar Impressão

- ▷ Quebra de linha pode ser feita utilizando o flag %n:

```
System.out.printf("Primeira linha%nSegunda linha");
```

- Flag que indica que nesta posição será feita uma quebra de linha.

Formatar Impressão

- ▷ A impressão em colunas pode ser feita utilizando um número após o % do flag.
 - Números positivos alinham à direita e números negativos alinham à esquerda.

```
System.out.printf("%-17s|%6d%n", "Linha 1 coluna 1", 123);  
System.out.printf("%-17s|%6d%n", "L 2 C 1", 45678);
```

		x
Linha 1	coluna 1	123
L 2	C 1	45678

Formatar Impressão

- ▷ A impressão em colunas pode ser feita utilizando um número após o % do flag.
 - Números positivos alinham à direita e números negativos alinham à esquerda.

```
System.out.printf("%-17s|%6s%n", "Linha 1 coluna 1", 123);  
System.out.printf("%-17s|%6s%n", "L 2 C 1", 45678);
```

	x
Linha 1 coluna 1	123
L 2 C 1	45678

- Impressão em 17 colunas alinhada à esquerda.

Formatar Impressão

- ▷ A impressão em colunas pode ser feita utilizando um número após o % do flag.
 - Números positivos alinham à direita e números negativos alinham à esquerda.

```
System.out.printf("%-17s|%6s%n", "Linha 1 coluna 1", 123);  
System.out.printf("%-17s|%6s%n", "L 2 C 1", 45678);
```

Linha 1 coluna 1	123
L 2 C 1	45678

- Impressão em 6 colunas alinhada à direita.

Formatar Impressão

- ▷ Flags:
 - %b: booleano
 - %s: string
 - %S: string em maiúsculas
 - %d: número inteiro
 - %f: número real
 - %n: quebra de linha

Exercícios

5) Faça um programa que leia o nome e a altura de 2 atletas e imprima os dados de cada atleta em uma linha utilizando colunas para obter o formato de tabela, conforme exemplo abaixo. A Altura deve ser impressa com duas casas decimais.

x	
Michael Jordan	1,98
Fernando Alonso	1,70

Exercícios

6) Faça um programa que leia uma temperatura em graus Celsius e imprima esta temperatura convertida para graus Fahrenheit, utilizando 1 casa decimal. A fórmula para conversão é:

$$F = C \times 1,8 + 32$$

Fim de material

Dúvidas?