VOLTAR

Tipos de dados em Java

Programar em Java requer domínio da linguagem e, portanto, conhecimento dos principais tipos de dados que se pode usar. Nesta aula serão apresentados os principais tipos de dados e exemplos de uso.

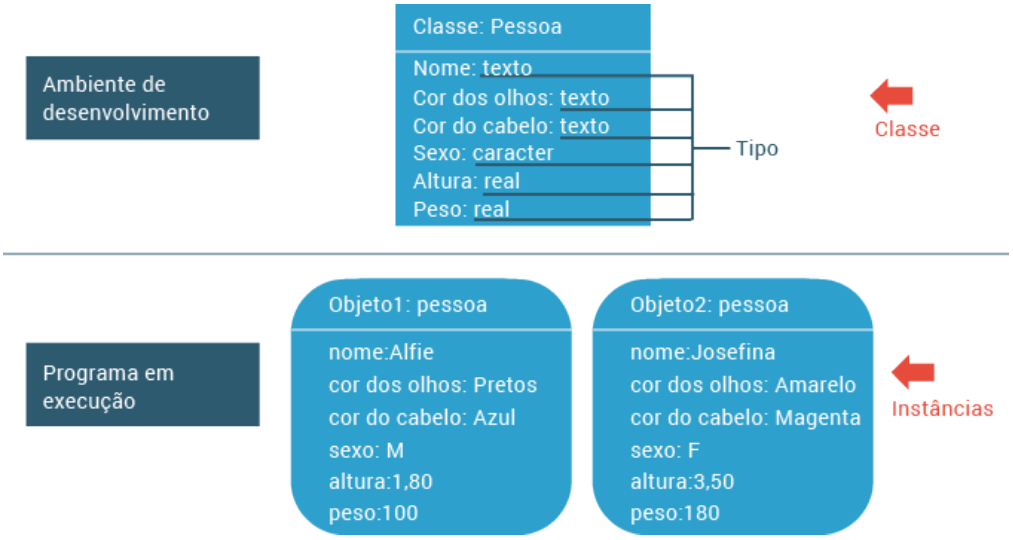
NESTE TÓPICO

O que são tipos de dados

Sempre que estamos programando, em Java ou qualquer outra linguagem, usamos variáveis e atributos o tempo todo. Essas variáveis e atributos podem ser de vários tipos: texto, caractere, número inteiro, número real, verdadeiros ou falso, etc.

Lembre-se do conceito de orientação a objetos, onde cada **Classe** representa algo que virará um ou mais **objetos**, quando seu programa estiver sendo executado. O objeto, então tem duas coisas principais: Características (**atributos**) e comportamentos (**métodos**).

Para se definir os atributos de uma classe usamos as variáveis, que podem ser de vários tipos: Texto, número inteiro, número real, verdadeiro ou falso, caractere etc. Só que usaremos estes tipos em inglês. Veja a imagem que ilustra a orientação a objetos e perceba o destaque no tipo de cada um dos atributos da classe:



Repare nos tipos de dados do cenário proposto

Tipos de dados em Java

Segundo Teruel, é muito importante saber que em Java existem dois tipos de dados: Primitivo e referência.

A tabela abaixo mostra os tipos primitivos implementados pelo Java (Teruel):

	Tipo primitivo	Espaço em bytes usado na memória	Exemplo
	char	2	char sexo= 'M';
Inteiros	byte	1	byte idade=55;
	short	2	short x=3456;
	int	4	int y= 678934;
	long	8	long cod=1756453;
Reais	float	4	float pi=3.1415F;
	double	8	double valor=34.56;
	boolean	1	boolean casado=true;

Principais tipos de dados em Java

Note que cada tipo usa uma quantidade diferente de bytes em memória para armazenamento das informações. Hoje em dia, com computadores poderosos, pode não ser tão preocupante usar uma variável do tipo long para se armazenar um inteiro curto (como idade, por exemplo), mas lembre-se que se você estiver programando para dispositivos móveis ou legados, a quantidade de memória, muitas vezes, pode ser limitada. Isso quer dizer que é muito importante usar o tipo certo para armazenar a informação desejada, independente de onde será executado seu programa.

Abaixo a explicação um pouco mais detalhada sobre cada tipo de dados primitivos em Java (Teruel):

- Tipo *char*: Este tipo é destinado para se armazenar um único caractere a atribuição de valor deve ser feita sempre com aspas simples;
- Tipo *byte*: Este tipo armazena valores inteiros, mas somente com um byte alocado em memória. Pode armazenar valores de -128 até 127 (o zero conta);
- Tipo *short*: O short, assim como o byte, armazena valores inteiros, porém utilizando 2 bytes de memória, ou seja, pode armazenar valores entre -32,768 e 32,767;
- Tipo *int*: O int representa um número inteiro de até 4 bytes, ou seja 2^{32} = 4294967296, isso quer dizer que o int pode armazenar valores inteiros de -2147483648 até 2147483647. Se você precisar de algo maior que isso (inteiro, também), você deve usar o tipo *long*;
- Tipo *long*: Assim como o int, o long armazena valores inteiros, porém com 8 bytes de alocação em memória, ou seja, pode-se armazenar valores de -9223372036854775808 até 9223372036854775807, mas somente valores inteiros;
- Tipo *float*: Este tipo deve ser utilizado para se armazenar valores de ponto flutuante (daí a origem do nome do tipo), ou seja, valores reais, que necessitam de casas decimais. Utiliza-se o ponto para separação do decimal. O tipo float utiliza 4 bytes para armazenamento de informação;
- Tipo *double*: Assim como o float, o double é utilizado para armazenar valores reais, com ponto flutuante, contudo com o dobro de espaço para armazenando (8 bytes);
- Tipo *boolean*: Este tipo armazena informações que podem ser apenas de dois tipos: verdadeiro (true) ou falso (false), por isso utiliza apenas 1 byte de espaço em memória.

Conforme mencionado, há também os tipos de referência, ou seja, que não existem nativamente na linguagem, mas podem ser implementados por ela ou por você. Veja alguns exemplos:

- Tipo *String*: Este tipo de dados é um dos mais usados para se armazenar texto. Esse tipo é implementado nativamente pelo Java, utilizando a classe String.java, como uma cadeia de caracteres e, internamente, é manipulado

dessa forma. Deve-se utilizar aspas duplas para atribuição de valores a este tipo. Exemplo: String nome = new String("Joana Dark da Silva Sauro");

- Tipo *Date*: O tipo date é utilizado para armazenamento de datas e, assim como o String, é implementado pelo Java através da classe Date.java. Exemplo: Date dtNascimento = new Date("10/10/2020").

> **MUITO IMPORTANTE!**
>
> Note que todos os tipos primitivos são declarados com letra minúscula e os tipos de de referência com letra maiúscula. Isso reforça o conceito de boas práticas de programação onde classes devem ser declaradas com a primeira letra maiúscula, como a String, que é uma classe do Java, contudo int ou float, por exemplo, não são classes, são tipos primitivos.

Convertendo tipos (casting)

Em Java é muito comum a necessidade de converter um tipo em outro, e isso vale tanto para tipos primitivos quanto tipos de referência (inclusive implementador por você!). Isso, em programação, chama-se "*casting*".

Para fazer isso, será exemplificada a conversão de um valor float para int:

```
1.  float X = 10;
2.  int Y = (int) X;
```

Contudo, se o valor for um número real, a conversão irá pegar somente a parte inteira do valor, ou seja:

```
1.  float X = 20.5F;
2.  int Y = (int)X; //neste caso o Y valerá 20
```

No exemplo acima, foi preciso usar a letra F junto ao número real, para "dizer" ao java que isso é um número real.

Operações que podem ser realizadas

Como qualquer linguagem de programação, você pode realizar uma série de operações com os valores armazenados em suas variáveis. Segundo Teruel, ? os principais tipos de operações que podem ser realizadas sobre os dados são por meio de expressões aritméticas, relacionais e lógicas?.

Expressões aritméticas

Segundo Teruel, as expressões aritméticas são utilizadas para cálculos matemáticos convencionais (soma, subtração, divisão e multiplicação). Para utilizar uma dessas expressões, basta aplicar o operador relacionado. Veja a tabela abaixo ilustrando as quatro operações e exemplos em Java:

Operador	Significado	Exemplo de uso em Java

*	Multiplicação	double quantidade=20;
		double valor = 109.6;
		double total = quantidade * valor;

/	Divisão	double salario = 1987.7;
		double percentual = 10.0;
		double aumento = salario * percentual / 100;

+	Soma	int a = 10;
		int b = 20;
		int c = a + b;

-	Subtração	double salario = 2456.76;
		double desconto = 200.57;
		double salLiquido = salario ? desconto;

Expressões aritméticas em Java

Em Java há, também, uma forma simples de escrever operações, facilitando muito na hora de digitar. Podemos dizer que é uma forma reduzida de se escrever uma operação, mas só vale para uma única operação simples; se você precisa de mais operandos, é melhor digitar a equação completa.

Por exemplo, se for preciso incrementar 1 em uma variável camada a, normalmente escrevemos assim:

```
1.  //...
2.       a = a + 1;
3.  //...
```

A forma reduzida dessa mesma expressão é:

```
1.  //...
2.        a++;
3.  //...
```

E se precisarmos incrementar essa mesma variável de 2, podemos escrever da seguinte forma:

```
1.  //...
2.        a += 2;
3.  //...
```

O resultado será o mesmo e isso vale para qualquer operação aritmética. Veja mais exemplos abaixo (Teruel):

a = a * 4; —> a *= 4;

X = X / Y —> X /= Y;

F = F - 1; —> F- -;

//etc...

Expressões relacionais

As expressões relacionas envolvem a comparação de dois valores. Em Java podemos comparar duas variáveis e obter verdadeiro (true) ou falso (false) como resultado dessa comparação.

Teruel propões o seguinte cenário para exemplificação:

Imagine as três variáveis declaradas em Java. A tabela a seguir mostra diversas comparações envolvendo essas três variáveis:

```
1.  //...
2.        int a, b, c; //Se você possui mais de uma variável do mesmo tipo, pode-se decl
    ara-las dessa forma
3.        a = 2;
4.        b = 5;
5.        c = 5;
6.  //...
```

Vamos agora ás comparações, em Java, envolvendo essas três variáveis:

Operador	Significado	Exemplo	Resultado
==	Igual	a==b b==c	false true
>	Maior	b > a	true
>=	Maior ou igual	b >= c a >= c	true false
<	Menor	b < c	false
<=	Menor ou igual	a <= b	true
!=	Diferente	a != b b != c	true false

Expressões relacionais em Java

Expressões lógicas

Segundo Teruel, as expressões lógicas “são aquelas que envolvem a comparação de dois valores boolean e resultam em um terceiro valor boolean”. Em Java podemos utilizar o “Não”, o “ou” e o “e”, conforme ilustrado pelo cenário abaixo.

```
//...
      boolean a, b;
      a = true;
      b = false;
//...
```

Operador	Significado	Exemplo	Resultado
!	Não	!a !b	false true
&&	E	a && b a && a	false true
||	OU	a || b b || b	true false

Expressões lógicas em Java

Vamos praticar

Para praticar, vamos criar um pequeno programa em Java que é capaz calcular o IMC (índice de massa corporal) de uma pessoa. Para calcular o IMC, é preciso ler do teclado duas informações sobre a pessoa: Seu **peso** e sua **altura**.

A forma para cálculo de IMC é:

$$IMC = peso/altura^2$$

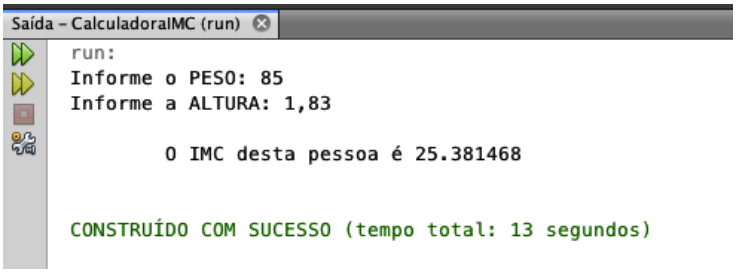
Para fazer leitura do teclado via console, em Java, utilizaremos um objeto da classe Scanner. O código abaixo mostro a programa em java comentado para a cálculo do IMC:

```java
//importa a classe Scanner para ser utilizada aqui:
import java.util.Scanner;

public class CalculadoraSimples {

    public static void main(String[] args) {

        //Declaracao do scanner:
        Scanner leitor = new Scanner(System.in);

        //Declaracao das variavies que serao utilizadas:
        float peso, altura, imc;

        //Leitura do teclado:
        //Informacao de instrucoes para o usuário:
        System.out.print("Informe o PESO: ");

        //le e armazena o valor do peso:
        peso = leitor.nextFloat(); //Le um valor de ponto flutuante

        //le e armazena o valor da altura:
        System.out.print("Informe a ALTURA: ");
        altura = leitor.nextFloat(); //Le um valor de ponto flutuante

        //calcula:
        imc = peso / (altura * altura);

        //Imprime o resultado
        //Note que pode-se concatenar o resultado com a impressao de texto
        System.out.println("\n\tO IMC desta pessoa é " + imc + "\n\n");

    }
}
```

BOAS PRÁTICAS DE PROGRAMAÇÃO

Note a indentação (formatação) do código. Em boas práticas de programação, o código deve ser corretamente indentado. No Netbeans, você pode utilizar a tecla de atalho “Ctrl + Shift + F” para que a indentação seja corrigida automaticamente.

Veja este programa em execução, ou seja, o resultado deste código:



Exemplos de execução do código acima, calculadora IMC

Lembre-se que a execução é feita no console do Netbeans.

Resumo dessa aula

Nesta aula você foi apresentado aos conceitos de tipos de dados em Java, onde destacam-se assuntos importantes, como:

- Os principais tipos primitivos de dados: char, byte, short, int, float, long, double, boolean.
- Os principais tipos de referência, como o String e o Date
- A comparação de valores do tipo primitivo, que podem ocorrer com operadores matemáticos (==, <, >, <=, >=, !=)
- A comparação de Strings, que devem ocorrer com o método equals ou equalsIgnoreCase
- As possíveis operações matemáticas com os valores numéricos (+, -, /, *)
- A redução de equações em Java
- Comparações lógicas (&&, !, ||)

Pronto. Agora você já está programando em Java de uma forma um pouco mais avançada! Para praticar, tente criar outros programas como uma calculadora com as quatro operações, uma que resolva tabuadas etc. Boa programação e bons estudos!

Quiz

Exercício Final

Tipos de dados em Java

INICIAR

Referências

Teruel, E. C., 2015, Programação Orientada a Objetos com Java - sem mistérios - 1ª Ed., Editora Uninove

Deitei P. e Deitel H., 2010, Java : Como programar, 8ª Edição, Pearson Pretice Hall

Schildt, H., 2015, Schildt, Java para iniciantes : crie, compile e execute programas Java rapidamente, Bookman

Avalie este tópico

ANTERIOR
Introdução à Programação Orientada a Objetos com Java

Índice

PRÓXIMO
Estruturas de Controle de Fluxo e Laços

Biblioteca (https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove (http://www.uninove.br)
Mapa do Site

Ajuda? (https://ava.un idCurso=)