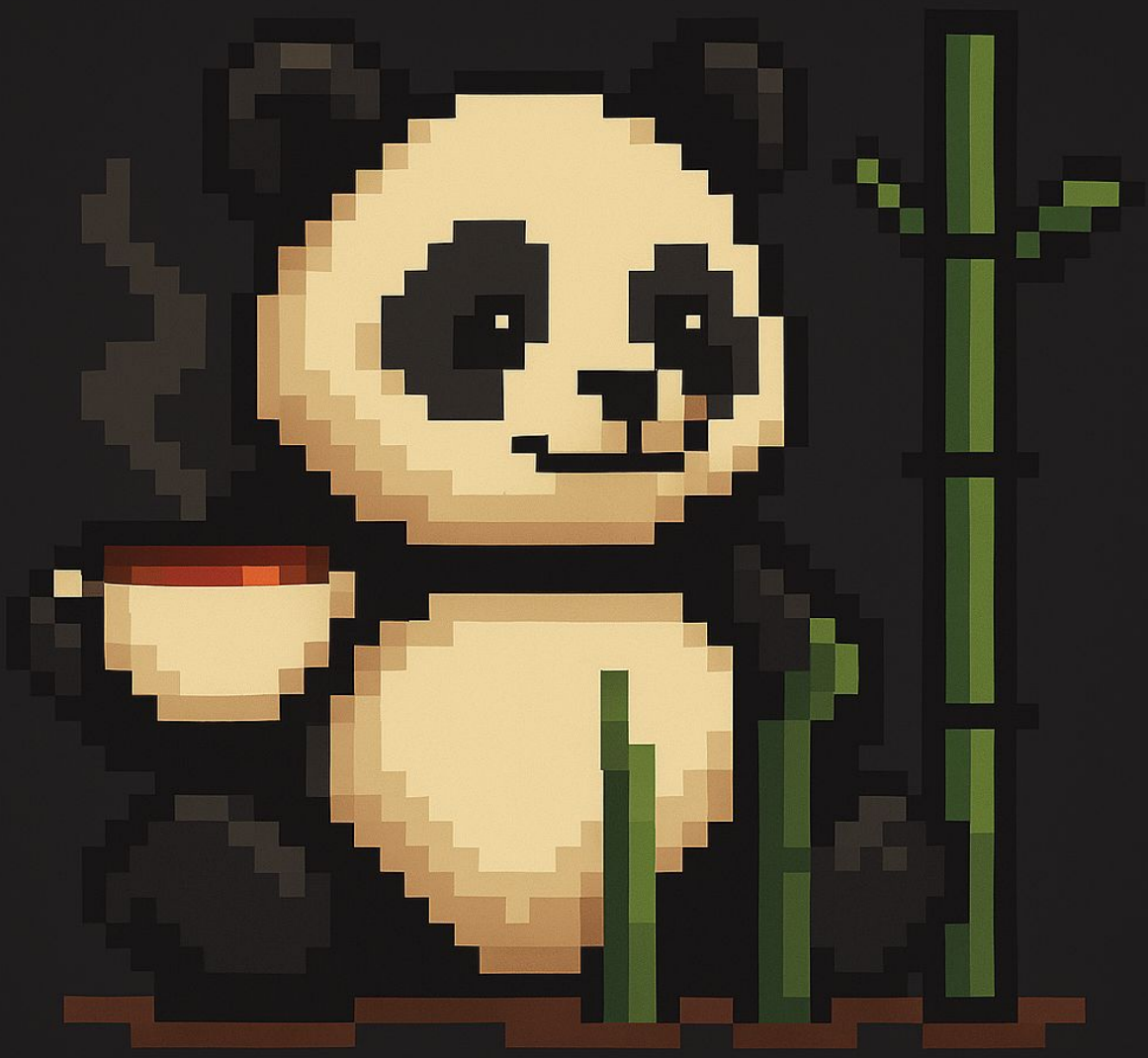
DESVENDANDO O JAVA

Aprenda com Foco e Serenidade

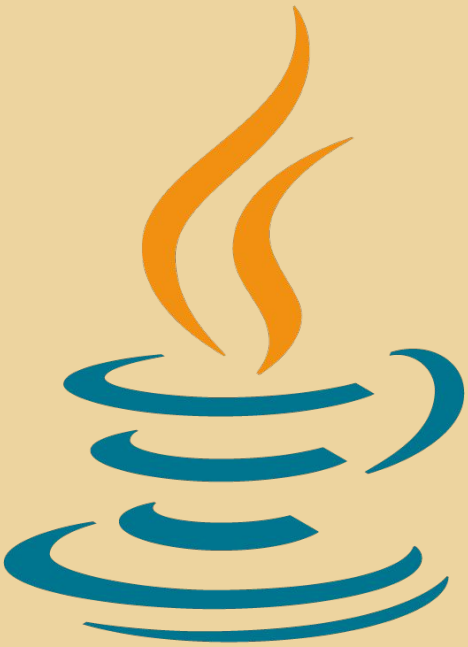


Eric Gravatá

Fundamentos do Java

Variáveis, Tipos de Dados e Classes

Java é uma linguagem poderosa e versátil, utilizada em diversos contextos como aplicações web, mobile e desktop. Para programar de forma eficiente, é fundamental entender variáveis, tipos de dados e classes. Vamos explorar cada um desses conceitos com exemplos práticos!



01

VARIÁVEIS

As variáveis são usadas para armazenar valores que podem ser alterados durante a execução do programa. Em Java, toda variável precisa ter um tipo definido.

GUARDANDO INFORMAÇÕES

Armazenando a idade de um usuário.

Exemplo prático:

Criando uma Classe chamada 'Pessoa'

```java
int idade = 25;
System.out.println("Idade do usuário: " + idade);
```

Aqui, `idade` é uma variável do tipo `int`, ideal para armazenar números inteiros.

Podemos interpretar da seguinte forma:
O valor **‘25’** está **armazenado** dentro da variável **‘idade’**. Dessa forma, poderemos utilizar a variável posteriormente em qualquer situação que for preciso utilizar o valor **‘25’**.

02

TIPOS DE DADOS

Java possui diferentes tipos de dados, divididos em primitivos e referenciais.

DEFININDO O TIPO DE INFORMAÇÃO

Tipos Primitivos

- **Inteiros:** `byte`, `short`, `int`, `long` – usados para números inteiros.
- **Ponto Flutuante:** `float`, `double` – para números decimais.
- **Texto:** `char` e `String` – usados para caracteres e textos.
- **Booleano:** `boolean` – armazena `true` ou `false`.

Exemplo Prático:

Criando uma Classe chamada 'Pessoa'

```java
String nome = "Alice";
double altura = 1.65;
boolean estudante = true;

System.out.println("Nome: " + nome);
System.out.println("Altura: " + altura + "m");
System.out.println("É estudante? " + estudante);
```

Cada variável possui um tipo de dado específico para sua função, isto é, valores diretos.

DEFININDO O TIPO DE INFORMAÇÃO

Tipos Referenciais

Eles incluem:

- **Classes;**
- **Interfaces;**
- **Arrays e;**
- **Tipos específicos da API coleções.**

Falaremos um pouco sobre cada um na sequência.

03

CLASSES

Em Java, uma classe é um molde para criar objetos. Ela define atributos (variáveis) e métodos (funções).

DEFININDO O TIPO DE INFORMAÇÃO

Tipos Referenciais

Classes e Objetos

Toda classe em Java pode ser usada para criar objetos que armazenam múltiplos dados e comportamentos.

Exemplo prático: Classe [`Carro`]

Criando uma Classe chamada 'Pessoa'

```java
class Carro {  // Definição da classe Carro
    String modelo;  // Variável para armazenar o modelo do carro
    int ano;  // Variável para armazenar o ano do carro

    void detalhes() {  // Método para exibir os detalhes do carro
        System.out.println("Modelo: " + modelo + ", Ano: " + ano);
    }
}

public class Main {  // Classe principal onde o programa é executado
    public static void main(String[] args) {  // Método principal
        Carro meuCarro = new Carro();  // Criando um objeto da classe Carro
        meuCarro.modelo = "Honda Civic";  // Definindo o modelo do carro
        meuCarro.ano = 2022;  // Definindo o ano do carro
        meuCarro.detalhes();  // Chamando o método para exibir os detalhes
    }
}
```

Aqui, `Carro` é um tipo referencial, e `meuCarro` é uma referência para um objeto da classe.

ORGANIZANDO O CÓDIGO

Tipos Referenciais

Vejamos mais um exemplo de classe:

Criando uma Classe chamada 'Pessoa'

```java
class Pessoa {
    String nome;
    int idade;

    void apresentar() {
        System.out.println("Olá, meu nome é " + nome + " e tenho " + idade + " anos.");
    }
}

public class Main {
    public static void main(String[] args) {
        Pessoa p1 = new Pessoa();
        p1.nome = "Lucas";
        p1.idade = 30;
        p1.apresentar();
    }
}
```

Aqui, a classe `Pessoa` define um objeto com os atributos `nome` e `idade`, além do método `apresentar()`.

04

INTERFACES

Uma interface em Java é um tipo referencial que define um conjunto de métodos a serem implementados por classes.

DEFININDO O TIPO DE INFORMAÇÃO

Tipos Referenciais

Interfaces

Exemplo Prático – Interface `Animal`

Criando uma Classe chamada 'Pessoa'

```java
interface Animal {  // Definição da interface Animal
    void emitirSom();  // Método abstrato a ser implementado
}

class Cachorro implements Animal {  // Classe Cachorro implementa a interface Animal
    public void emitirSom() {  // Implementação do método emitirSom
        System.out.println("Au Au!");  // Exibe som característico do cachorro
    }
}

public class Main {  // Classe principal do programa
    public static void main(String[] args) {  // Método principal
        Animal meuCachorro = new Cachorro();  // Criando um objeto da classe Cachorro
        meuCachorro.emitirSom();  // Chamando o método para emitir som
    }
}
```

`meuCachorro` é uma referência para um objeto da classe `Cachorro`, que implementa a interface `Animal`.

05

ARRAYS

Arrays são usados para armazenar múltiplos valores de um mesmo tipo dentro de uma estrutura ordenada.

DEFININDO O TIPO DE INFORMAÇÃO

Tipos Referenciais

Arrays

Exemplo Prático – Lista de Notas

Criando uma Classe chamada 'Pessoa'

```java
int[] notas = {85, 90, 78, 92};  // Declaração e inicialização de um array de notas

for (int nota : notas) {  // Loop para percorrer o array
    System.out.println("Nota: " + nota);  // Exibição de cada nota
}
```

O array `notas` é um tipo referencial que aponta para uma sequência de valores.
Os valores dentro das chaves sempre seguirão uma ordem contagem de posições chamados **índice** e a sequência sempre começa a contar a partir de **0**.
Por exemplo: o valor **‘85’** está na posição de índice 0, o valor **‘90’** está na posição de índice 1, e assim sucessivamente.

06

COLEÇÕES

Java possui estruturas de dados avançadas, como `ArrayList`, `HashMap`, entre outras.

DEFININDO O TIPO DE INFORMAÇÃO

Tipos Referenciais

Tipos específicos da API de coleções

Exemplo Prático – Lista de Nomes

Criando uma Classe chamada 'Pessoa'

```java
import java.util.ArrayList;  // Importação da classe ArrayList

ArrayList<String> nomes = new ArrayList<>();  // Criando uma lista dinâmica de Strings
nomes.add("Lucas");  // Adicionando o nome "Lucas" à lista
nomes.add("Ana");  // Adicionando o nome "Ana" à lista
nomes.add("Pedro");  // Adicionando o nome "Pedro" à lista

for (String nome : nomes) {  // Loop para percorrer a lista
    System.out.println("Nome: " + nome);  // Exibição de cada nome armazenado
}
```

O `ArrayList` permite manipulação dinâmica de elementos sem a limitação de tamanho fixo dos arrays.

AGRADECIMENTOS

OBRIGADO POR LER ATÉ AQUI

Esse Ebook foi gerado por IA, e diagramado por humano.

O passo a passo se encontra no meu Github.

•

Esse conteúdo foi gerado com fins didáticos de construção, não foi realizado uma validação cuidadosa humana no conteúdo e pode conter erros gerados por uma IA.

https://github.com/ericgravata/DIO-as-create-a-ebook.git