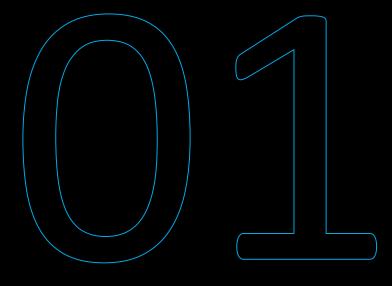
# JAVA

Um guia completo para iniciantes





# O QUE É JAVA E POR QUE APRENDER?

### Primeiros Passos com Java

#### Conhecendo a linguagem Java

Java é uma linguagem de programação muito usada no mercado, presente em aplicações web, mobile (como apps Android) e sistemas corporativos. Ela é conhecida por ser robusta, orientada a objetos e multiplataforma.

Exemplo real: Cálculo de pedido Imagine um sistema de pedidos de comida por aplicativo. O código que calcula o valor total do pedido, com frete e desconto, pode ser feito em Java no backend.

```
public class Pedido {
    public static void main(String[] args) {
        double valorPedido = 50.0;
        double frete = 10.0;
        double desconto = 5.0;
        double total = valorPedido + frete - desconto;

        System.out.println("Total do pedido: R$ " + total);
    }
}
```



# SEU PRIMEIRO PROGRAMA EM JAVA

## Seu primeiro programa em Java

```
"Olá, Mundo!" — o clássico
```

Todo aprendizado em linguagens começa com imprimir algo na tela. Isso serve para garantir que seu ambiente de desenvolvimento está funcionando corretamente e para apresentar a estrutura básica de um programa Java.

```
public class OlaMundo {
    public static void main(String[] args) {
        System.out.println("Olá, Mundo!");
    }
}
```

#### Como executar seu código Java

- Salve o arquivo como OlaMundo.java.
- > Compile: javac OlaMundo.java
- Execute: java OlaMundo



# VARIÁVEIS E TIPOS DE DADOS

# Variáveis e Tipos de Dados

### Guardando informações

Em Java, variáveis são espaços na memória usados para guardar dados. Cada variável tem um tipo, que define que tipo de dado ela pode armazenar. Entender isso é essencial para manipular informações corretamente no seu programa.

```
int idade = 25;
double salario = 3000.50;
String nome = "Ana";
boolean ativa = true;

System.out.println("Funcionária: " + nome);
```

Exemplo real: Cálculo de bônus

```
double salario = 3000.0;
double bonus = salario * 0.10;
System.out.println("Bônus: R$ " + bonus);
```



# CONDICIONAIS (if,else)

## Condicionais (if, else)

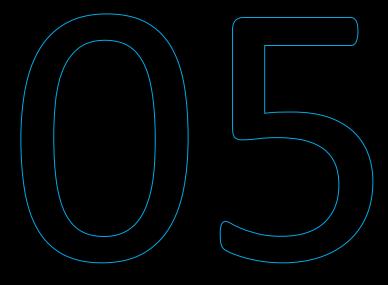
#### Tomando decisões

Condicionais permitem que o programa tome decisões baseadas em condições lógicas. É com if e else que controlamos o fluxo de execução e criamos comportamentos dinâmicos.

```
int idade = 18;
if (idade >= |18) {
    System.out.println("Maior de idade");
} else {
    System.out.println("Menor de idade");
}
```

Exemplo real: Frete grátis

```
double valorPedido = 120.0;
if (valorPedido >= 100.0) {
    System.out.println("Frete grátis!");
} else {
    System.out.println("Frete: R$ 15,00");
}
```



# LAÇOS DE REPETIÇÃO (for, while)

## Laços de Repetição (for, while)

### Repetindo ações com for

Laços de repetição são usados para executar blocos de código várias vezes. Com for e while, você pode percorrer listas, repetir cálculos ou exibir dados sequencialmente.

```
for (int i = 1; i <= 5; i++) {
    System.out.println("Pedido no: " + i);
}</pre>
```

Exemplo real: Listar itens do pedido

```
String[] itens = {"Pizza", "Refrigerante", "Sobremesa"};
for (String item : itens) {
    System.out.println("Item: " + item);
}
```

# CRIANDO MÉTODOS

### Criando Métodos

### Reutilizando código com funções

Métodos são blocos de código que realizam uma tarefa específica. Eles ajudam a organizar o programa, evitam repetição de código e facilitam a manutenção.

```
public class Calculadora {
    public static double somar(double a, double b) {
        return a + b;
    }

public static void main(String[] args) {
        double resultado = somar(10, 5);
        System.out.println("Resultado: " + resultado);
    }
}
```

Exemplo real: Método de desconto

```
public static double aplicarDesconto(double valor, double desconto) {
    return valor - desconto;
}
```



# CLASSES E OBJETOS

## Classes e Objetos

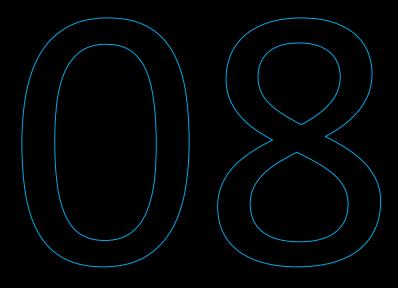
#### Fundamentos da Orientação a Objetos

Java é uma linguagem orientada a objetos. Isso significa que organizamos o código em classes (estruturas que definem atributos e comportamentos) e objetos (instâncias dessas classes). Esse modelo facilita o reaproveitamento e a escalabilidade do código.

```
public class Produto {
    String nome;
    double preco;

    void mostrarInfo() {
        System.out.println("Produto: " + nome + ", Preço: R$" + preco);
    }
}

public class Loja {
    public static void main(String[] args) {
        Produto p = new Produto();
        p.nome = "Camiseta";
        p.preco = 49.90;
        p.mostrarInfo();
    }
}
```



# CONCLUSÃO E PRÓXIMOS PASSOS

## Conclusão e Próximos Passos

#### O que você aprendeu até aqui

Você agora entende o básico de Java! Aprendeu como funciona a estrutura de um programa, como manipular dados, tomar decisões, repetir ações, criar métodos reutilizáveis e aplicar conceitos de orientação a objetos.

Continue praticando e explore temas mais avançados como:

- Arrays
- Listas e Coleções Manipulação de arquivos
- Integração com banco de dadosDesenvolvimento web com Spring Boot

Dica final

Crie pequenos projetos, como:

- Uma calculadora
- Um controle de tarefas
- Um sistema simples de pedidos

Com o tempo, esses projetos podem se transformar em portfólio para mostrar suas habilidades.