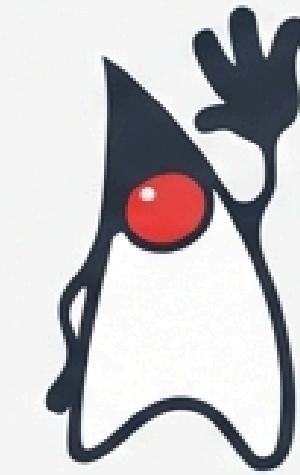


Java: Do Green Project à Máquina Virtual

Uma jornada visual pela linguagem que conecta o mundo.

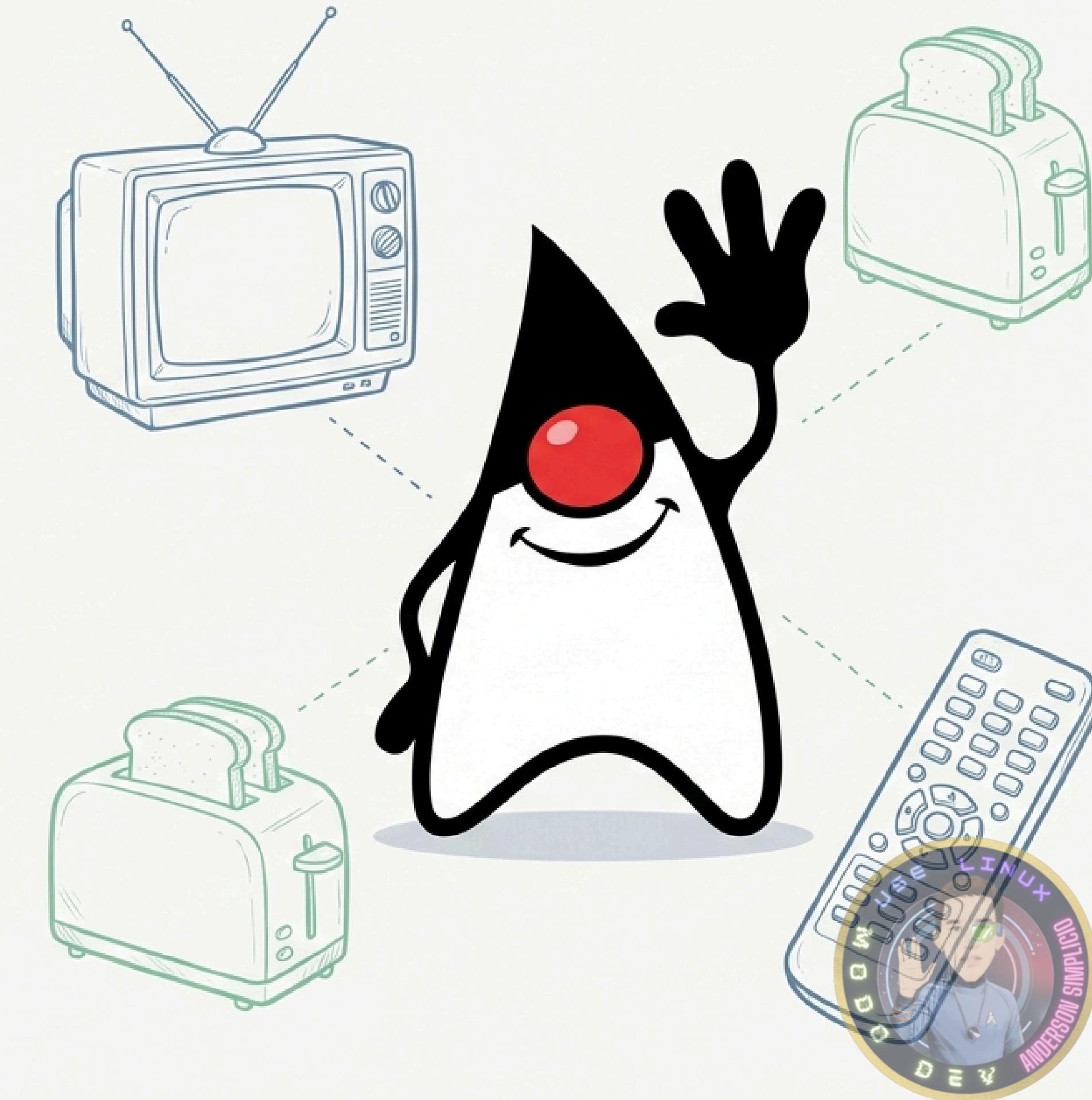


O Início Inesperado: Eletrodomésticos Inteligentes

A história não começou nos computadores. Na década de 90, o 'Green Project' da Sun Microsystems, liderado por James Gosling, tinha um objetivo diferente: conectar aparelhos domésticos e televisões a cabo.

Inter

A interface original chamava-se 'Star7', e foi nessa época que o mascote 'Duke' nasceu para guiar os usuários em dispositivos portáteis.



Do 'Oak' ao Café: Uma Mudança de Identidade



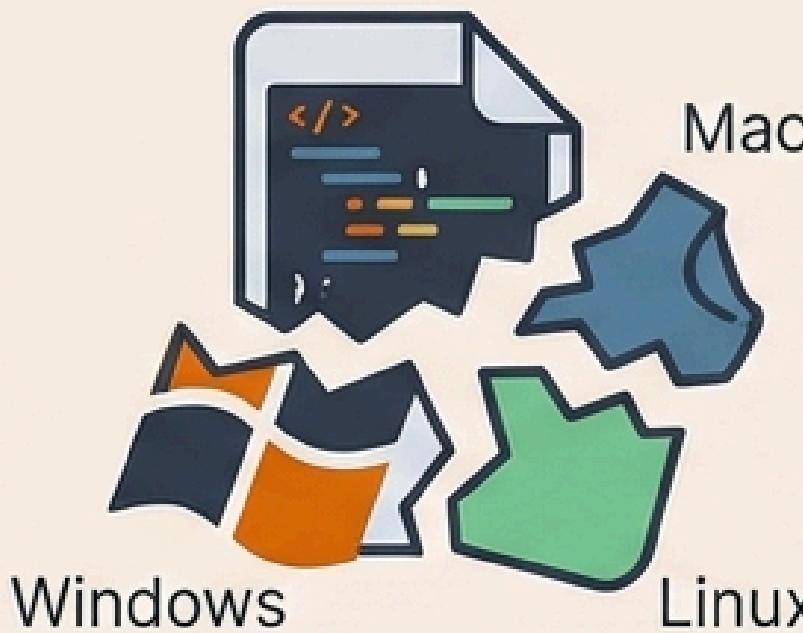
A Origem: **Oak** (Carvalho). Inspirado em uma árvore que Gosling via da janela. O nome já estava registrado.

A Solução: **Java**. Em uma reunião em uma cafeteria, a equipe buscou vivacidade. Homenagem ao café da ilha da Indonésia.





A Revolução da Portabilidade

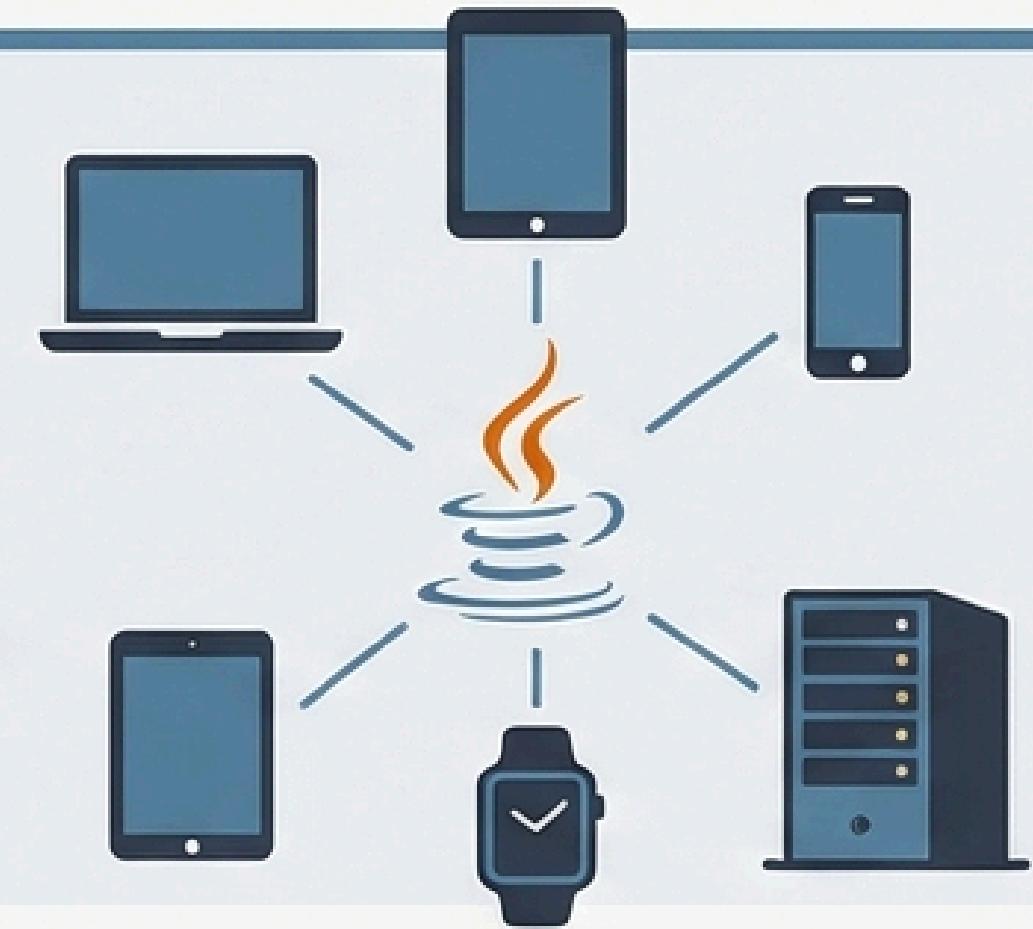


O Problema

Antes, um programa precisava ser reescrito ou recompilado para cada sistema operacional.

A Solução

"Write Once, Run Anywhere"
(Escreva uma vez, execute em qualquer lugar).



Aplicações Bancárias:
Segurança extrema.



Android: Base do ecossistema mobile.



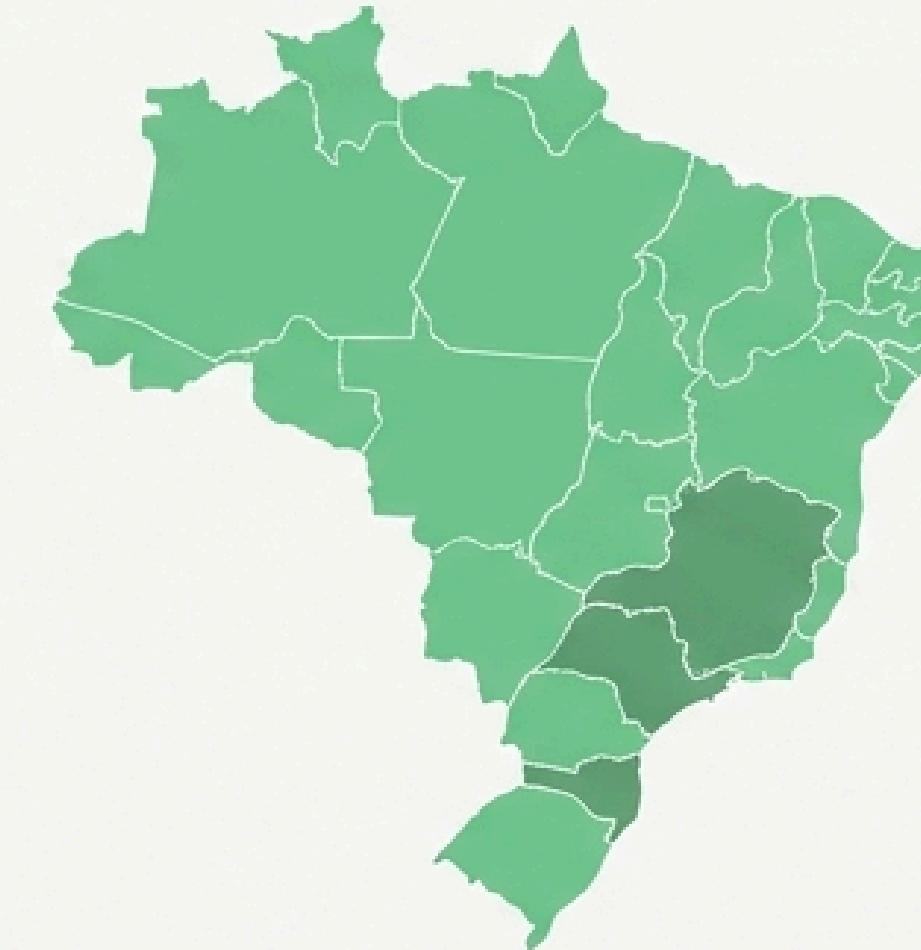
Big Data:
Processamento massivo.

O Valor da Expertise no Mercado Brasileiro

Média Salarial por Nível (Fonte: Código Fonte TV)



Destaques Regionais



- 01** São Paulo (SP): R\$ 11.028,40
- 02** Minas Gerais (MG): R\$ 9.958,17
- 03** Santa Catarina (SC): R\$ 9.306,56



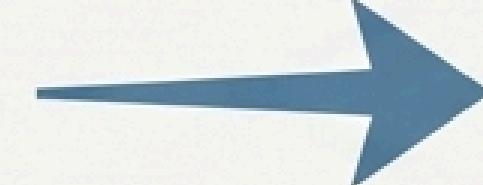
A demanda por especialistas (Pleno/Sênior) domina o mercado, mostrando um caminho claro de crescimento.

O Esperanto da Computação: Entendendo o Bytecode

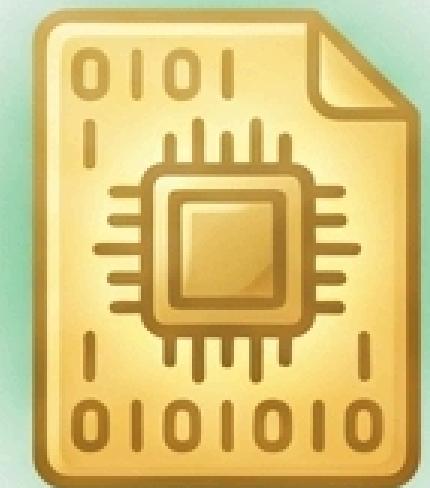
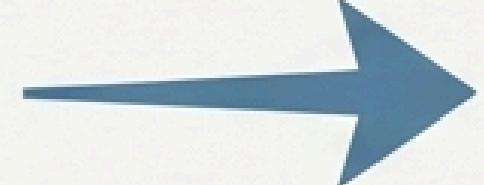


Código Fonte (.java)

O que você escreve



Compilador (javac)



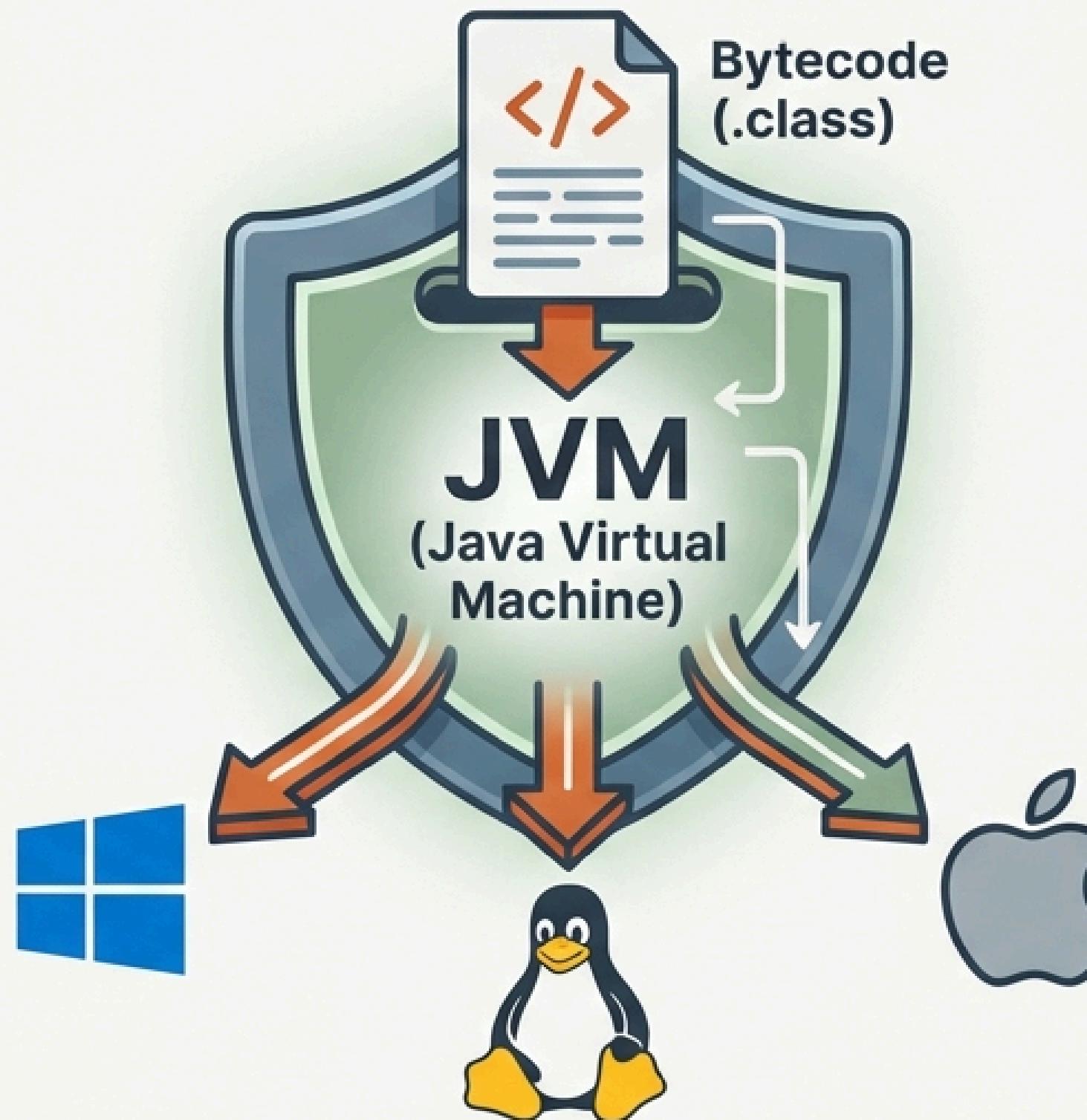
Bytecode (.class)

Código Universal

O Bytecode não é binário puro nem linguagem humana. É um código intermediário otimizado para a Máquina Virtual, independente se o sistema é Windows, Linux ou Mac.



JVM: A Tradutora Universal e Segura



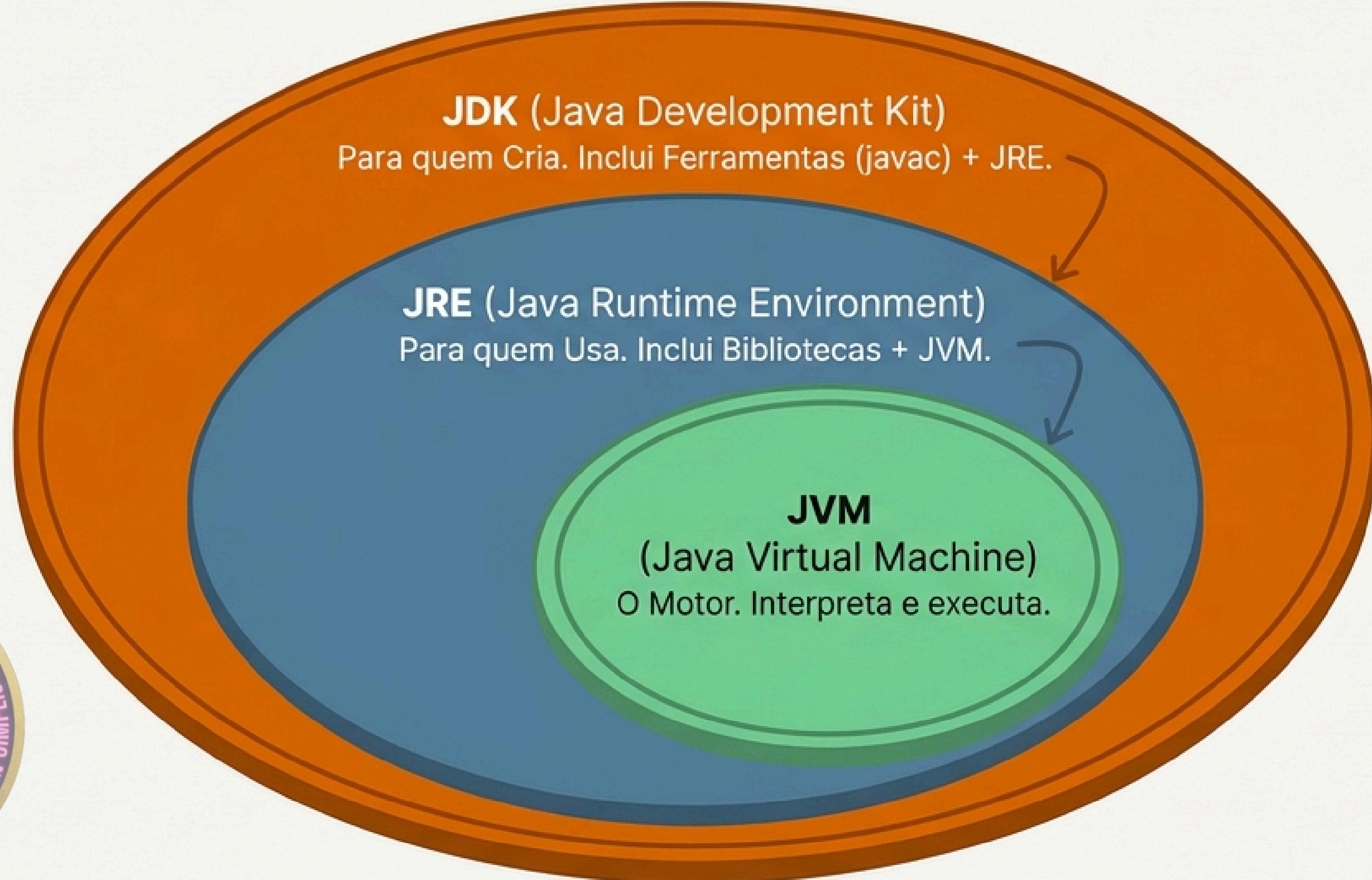
A Tradutora

A JVM “finge” ser um computador e traduz o bytecode em tempo real para o hardware específico.

O Sandbox (Caixa de Areia)

Segurança total: o código roda isolado dentro da JVM, sem acesso perigoso à memória física do computador.

A Hierarquia de Execução (JDK vs JRE vs JVM)

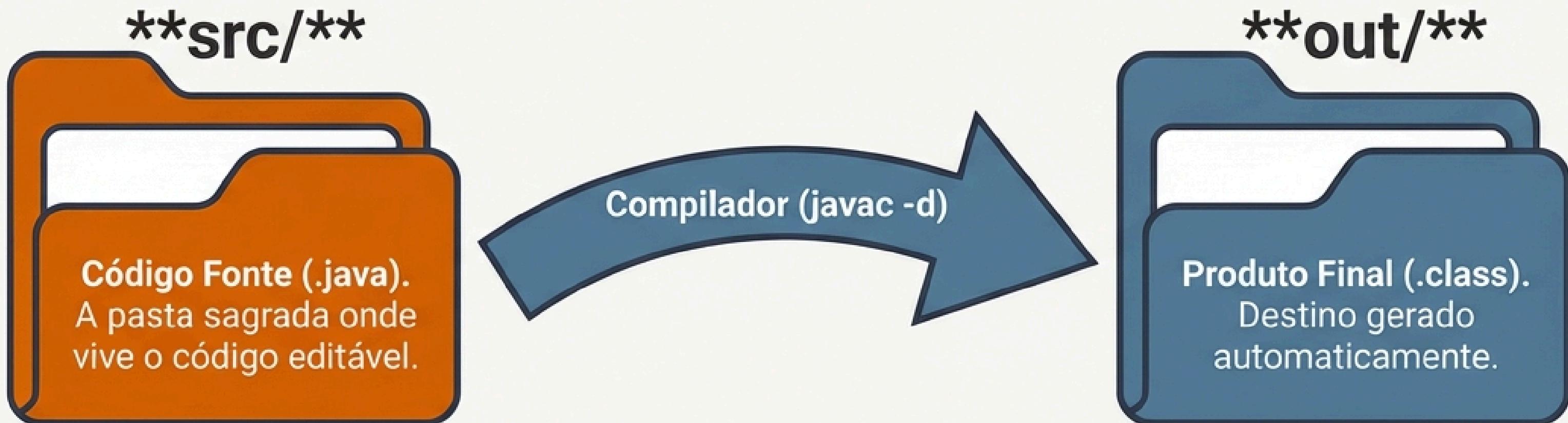


Desde o Java 11, o foco da distribuição é no JDK, permitindo pacotes de execução personalizados.



Organizando a Casa: Compilação Limpa

```
javac -d out src/Main.java
```



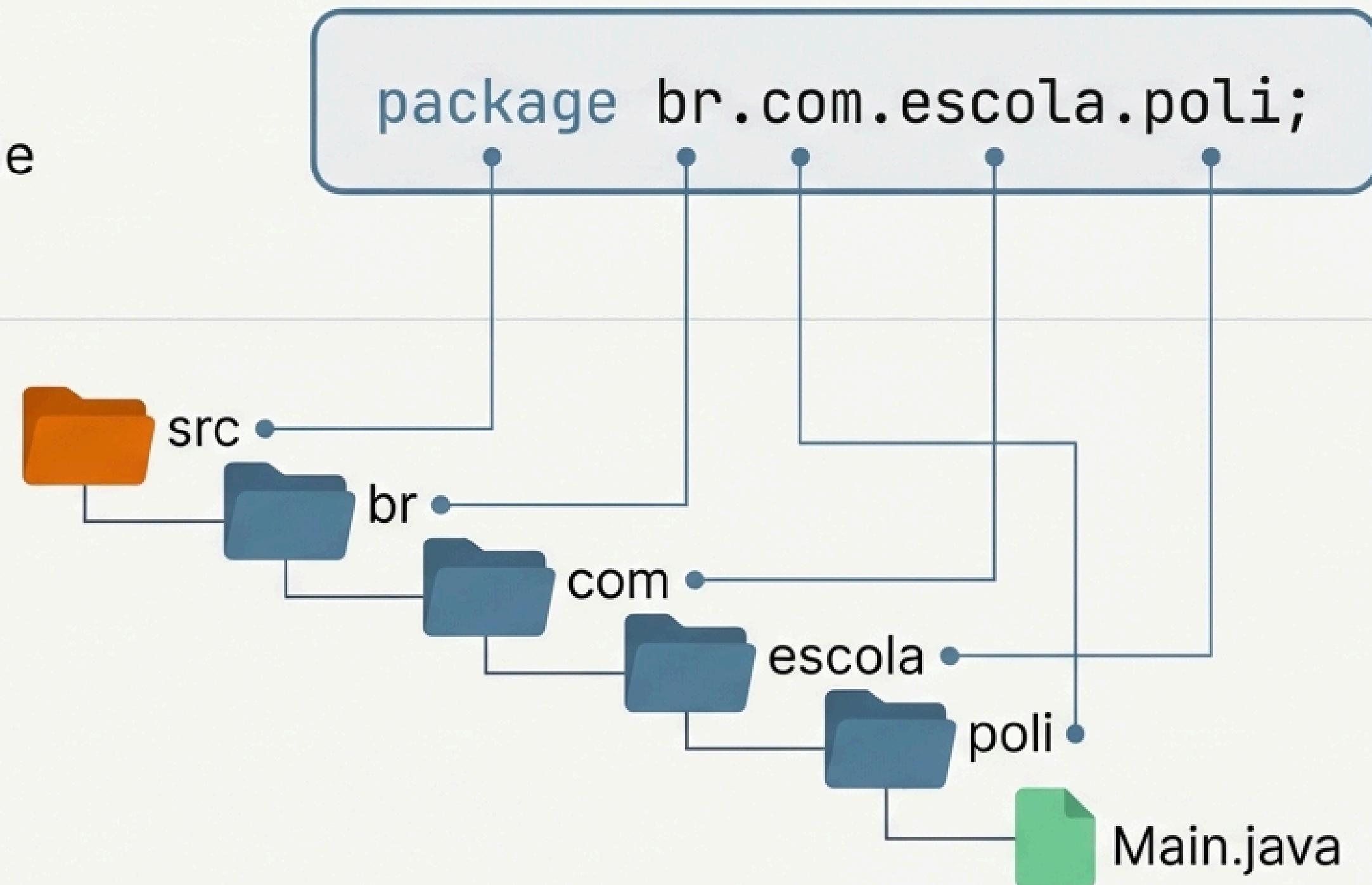
O parâmetro **-d** (*directory*) garante que o compilador não misture arquivos gerados com seu código fonte, mantendo o projeto organizado.

Pacotes: A Lógica de Organização

Logical View

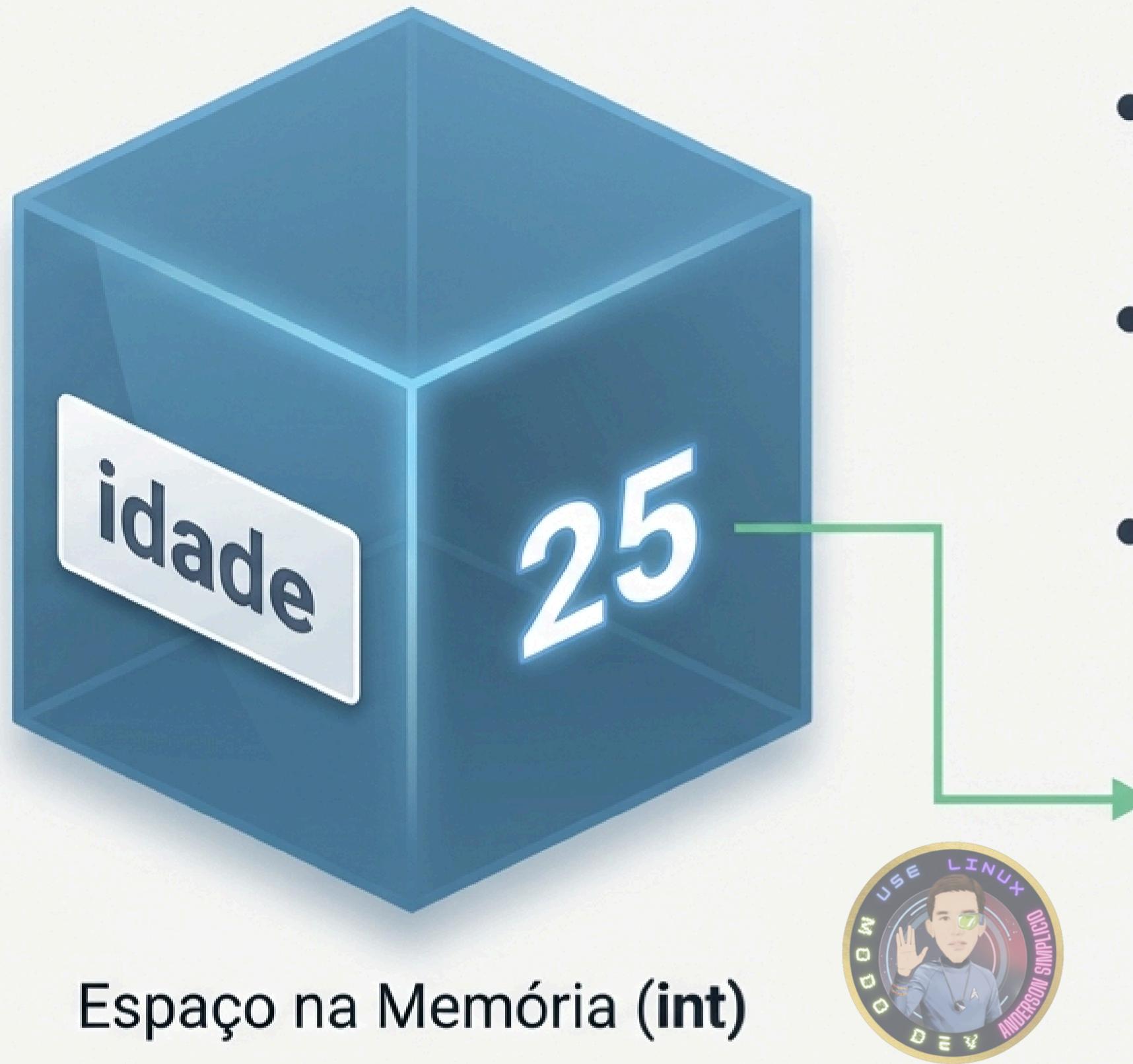
Pacotes agrupam classes por funcionalidade e evitam conflitos de nomes (ex: br.com.escola).

Physical View



Para rodar: java -cp out br.com.escola.poli.Main

Variáveis: As Caixas Etiquetadas da Memória



- **A Caixa (Memória):** Espaço reservado.
- **A Etiqueta (Nome):** Como encontramos o dado.
- **O Conteúdo (Valor):** O dado guardado.

Variável: Se `idade = 26`, o 25 é descartado e o 26 assume o lugar.



Regras e Etiqueta do Código

! Obrigatório (Regras)

- ✓ Começar com letra ou `_`.
- ✗ Começar com números (ex: `1nota`).
- ✗ Espaços ou caracteres especiais (@, #).
- ⚠ Case Sensitive: `Idade` ≠ `idade`.

✓ Boas Práticas (Etiqueta)

Padrão "corcova de camelo" para facilitar a leitura.

mediaFinal

nomeDoAluno

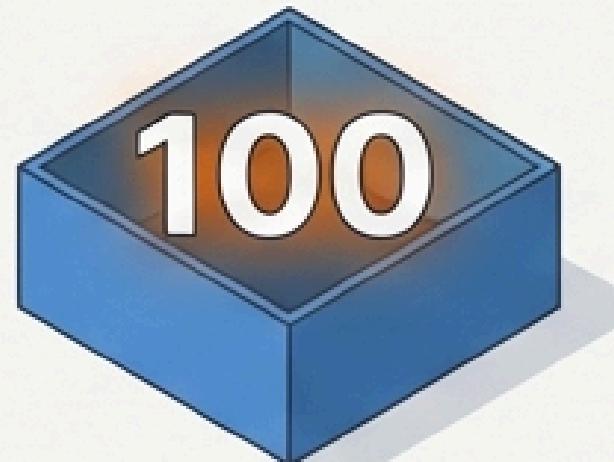
numeroDeAlunos



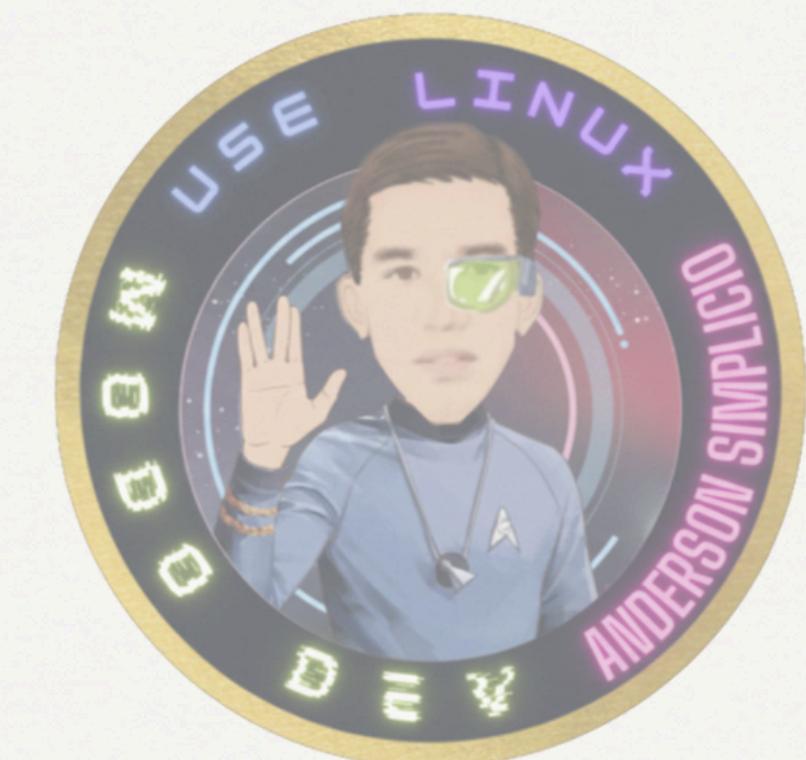
✓ Use nomes significativos. Evite `x` ou `y`.

Tipos de Dados: O que tem dentro da caixa?

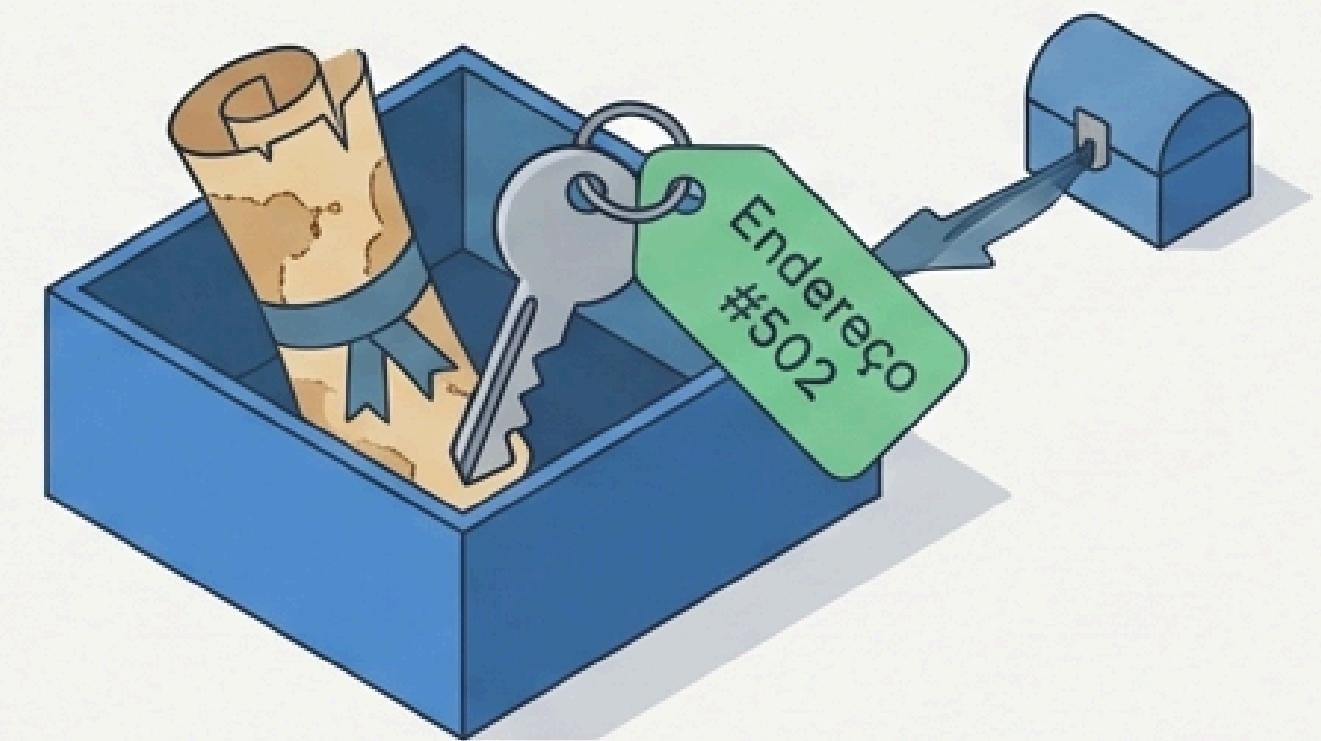
Tipos Primitivos (Valores Diretos)



- int (Inteiros)
- double (Decimais)
- boolean (True/False)
- char (Caractere único)



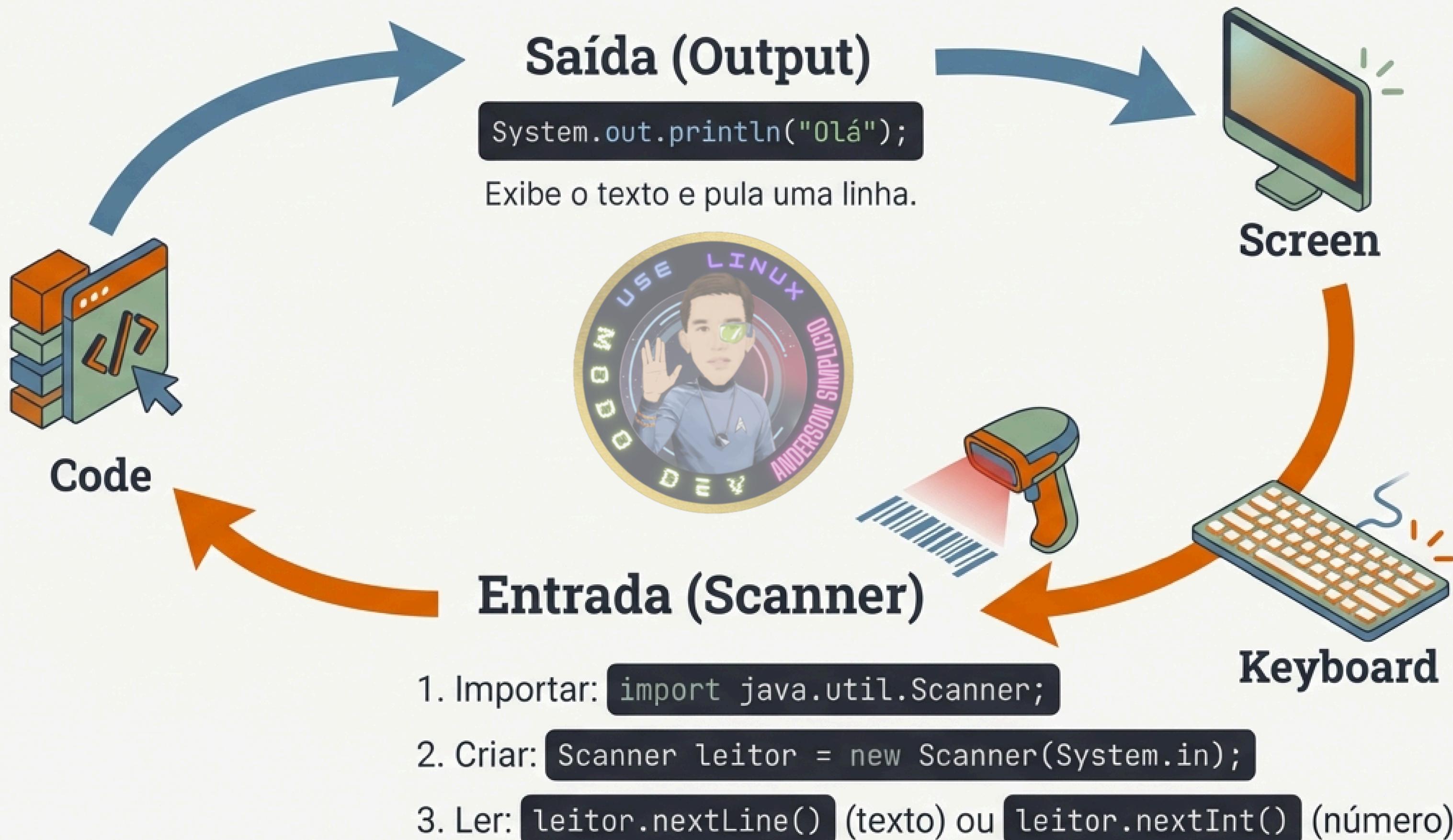
Tipos de Referência (Endereços)



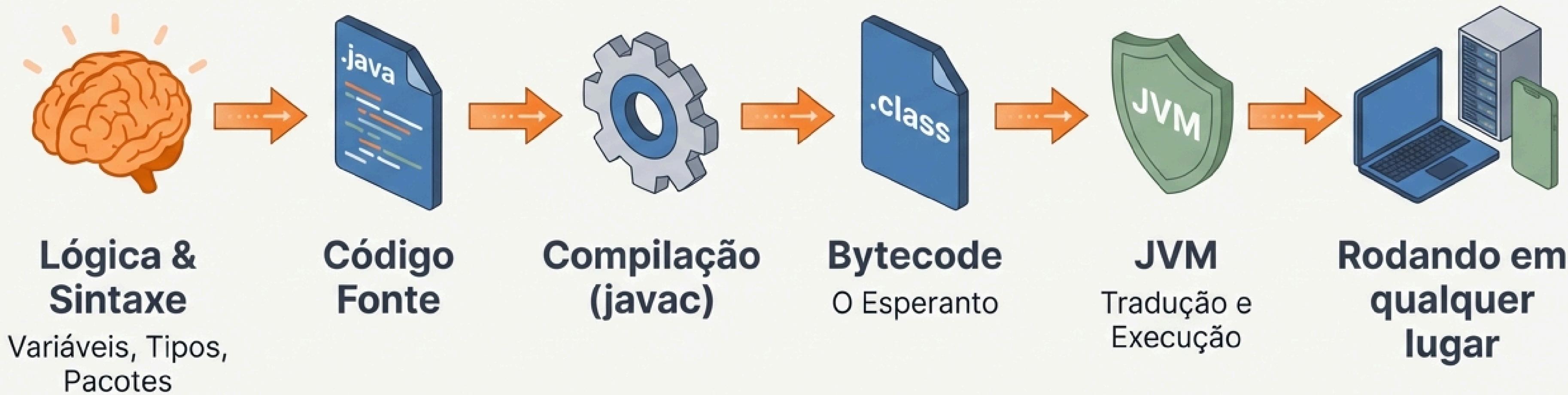
- String (Textos)
- Scanner
- Objetos

Primitivo guarda o dinheiro. Referência guarda o mapa do cofre.

Interatividade: Entrada e Saída de Dados



O Ciclo de Vida do Código Java

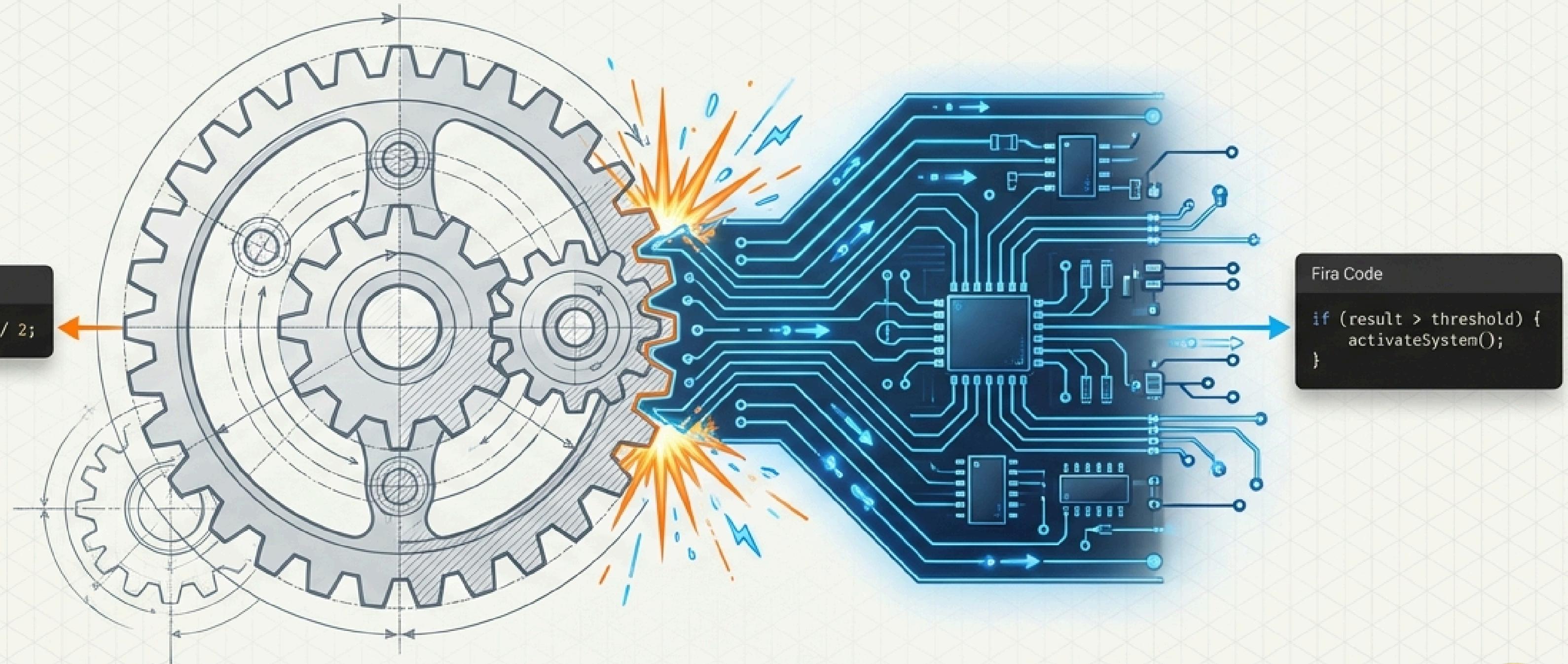


Java: Uma linguagem projetada para robustez, segurança e portabilidade global.



Java Fundamentals: Matemática e Controle de Fluxo

Como construir a “Mecânica Lógica” do seu código.



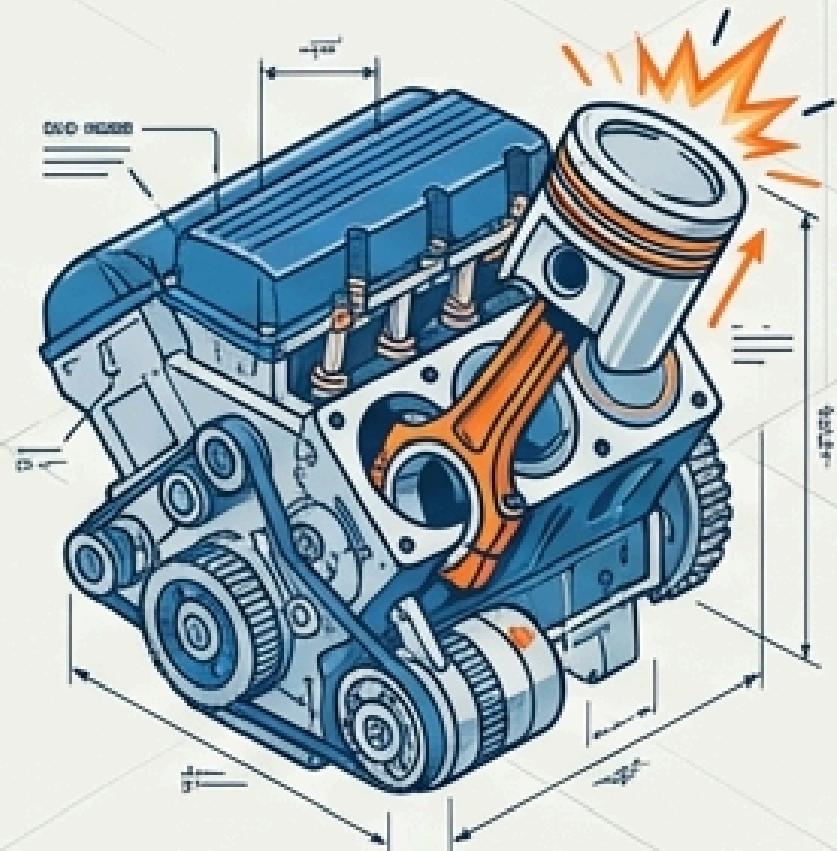
Conceitos Essenciais: Aritmética, Precedência e Condicionais



Programar é ensinar o computador a pensar.

Roboto Serif

1. Cálculo (O Motor)



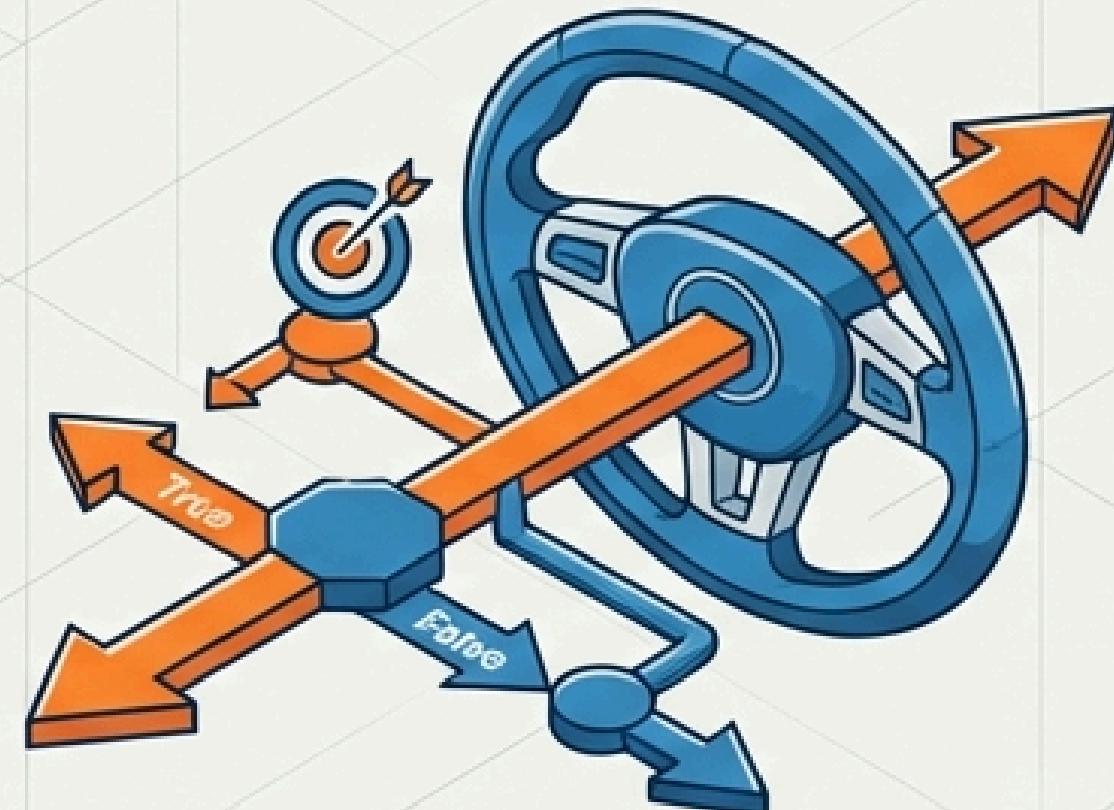
Processar números e dados para encontrar respostas.

```
int x = 10 + 5;
```

Fira Code

Roboto Serif

2. Lógica (O Volante)



Tomar decisões e escolher caminhos baseados nesses dados.

```
if (x > 10) { ...
```

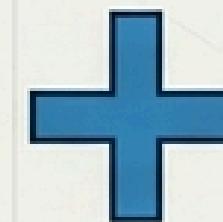
Fira Code



O Motor Aritmético: As 5 Operações Básicas

Roboto Serif

As regras são parecidas com a álgebra escolar, mas com símbolos específicos para o processador.



Adição

10 + 5
Fira Code

→ 15



Subtração

10 - 5
Fira Code

→ 5



Multiplicação

10 * 5
Fira Code

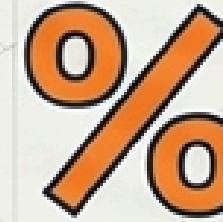
→ 50



Divisão

10 / 4
Fira Code

→ Depende do tipo de dado
(Ver próximo slide)



Módulo (Resto)

10 % 3
Fira Code

→ 1



A Regra de Ouro: Cuidado com a Divisão Inteira



O Java observa o TIPO dos números antes de calcular. Se você divide dois inteiros, o resultado será truncado (cortado), não arredondado.

Inteiro / Inteiro

$$5 \text{ / } 2$$

$$= 2$$

O Java joga fora o 0.5

Fira Code

Decimal / Inteiro

$$5.0 \text{ / } 2$$

$$= 2.5$$

Fira Code

Basta um número ser double para manter a precisão



Dica: Para cálculos de dinheiro ou médias, sempre utilize tipos double ou float.

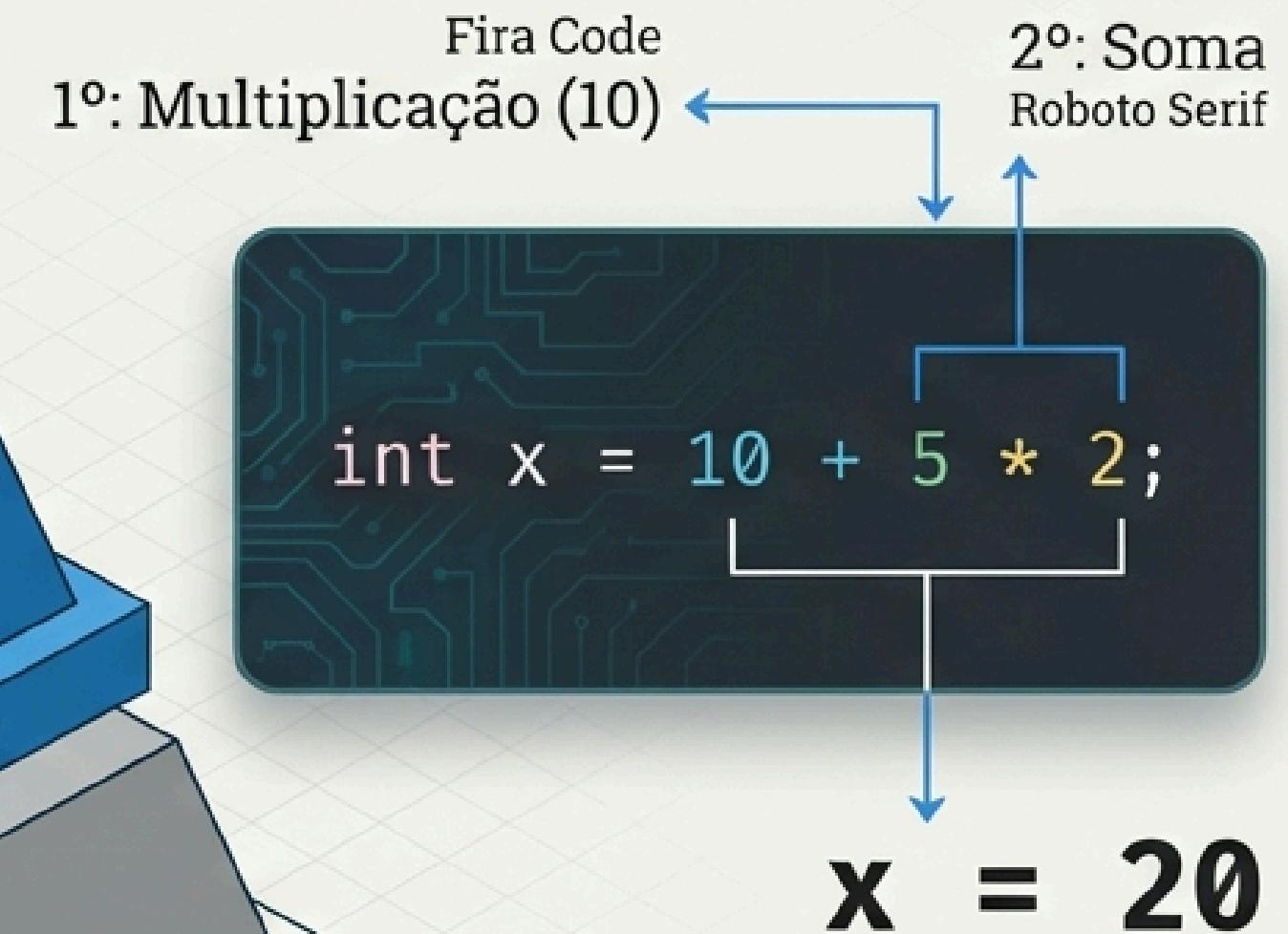


Quem Manda Primeiro? (Precedência de Operadores)

Helvetica Now Display

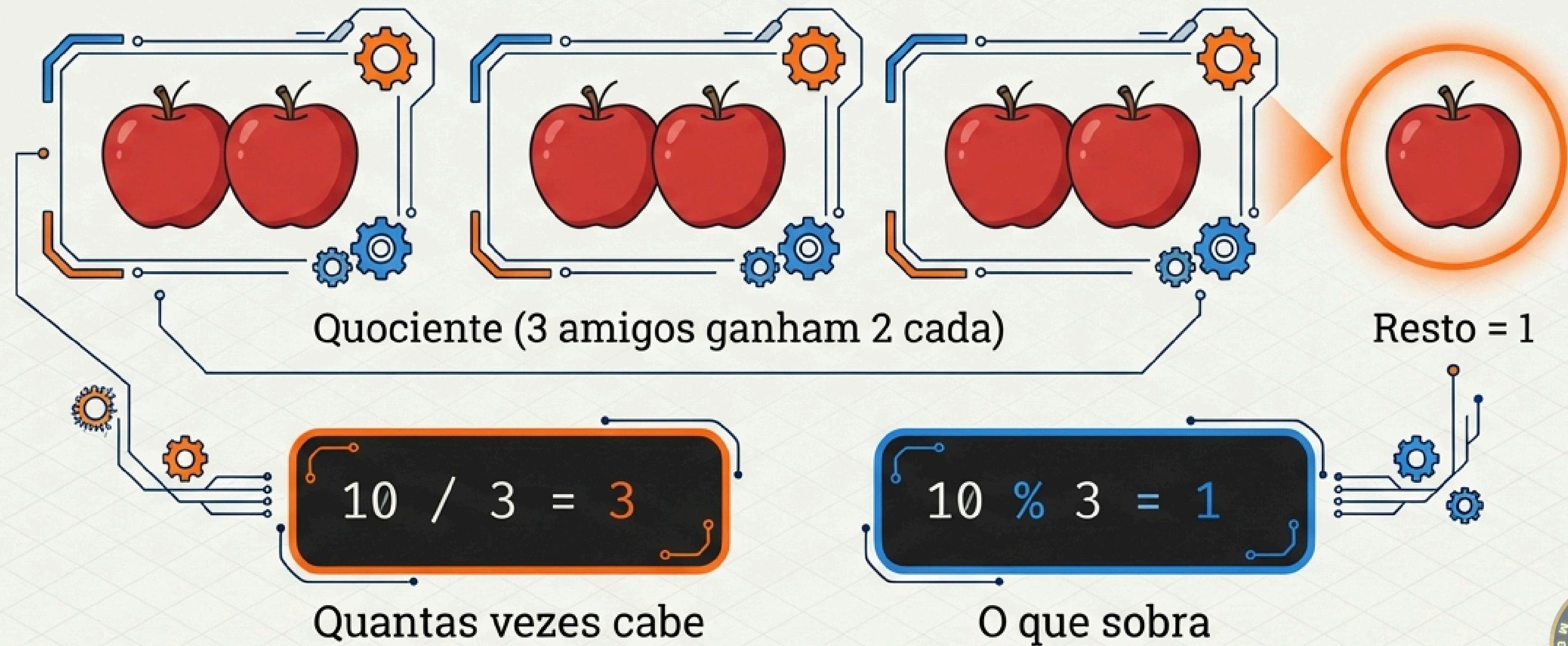
Roboto Serif

O Java segue uma hierarquia estrita de importância, de cima para baixo.



Entendendo o Módulo (%)

O operador que olha para o que SOBRA, não para o resultado.



Para que serve o Módulo na prática?

O **%** resolve problemas lógicos fundamentais.



Par ou Ímpar

`numero % 2 == 0 → Par`
`numero % 2 == 1 → Ímpar`

Ciclos e Limites

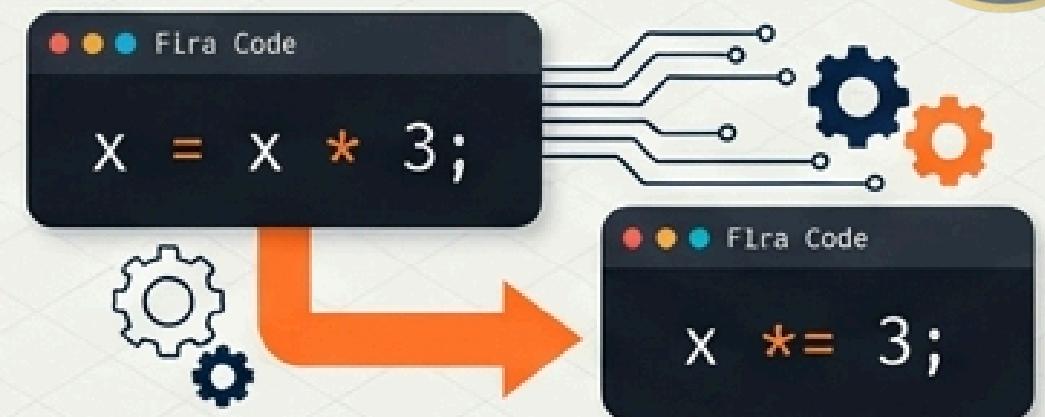
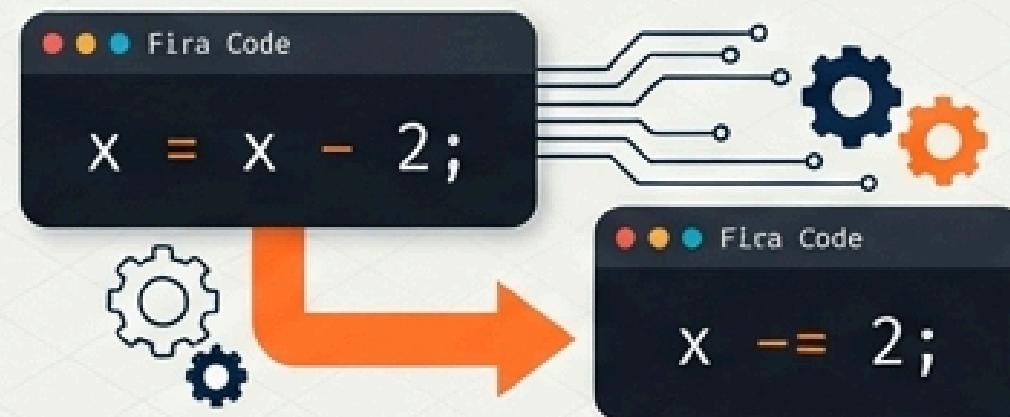
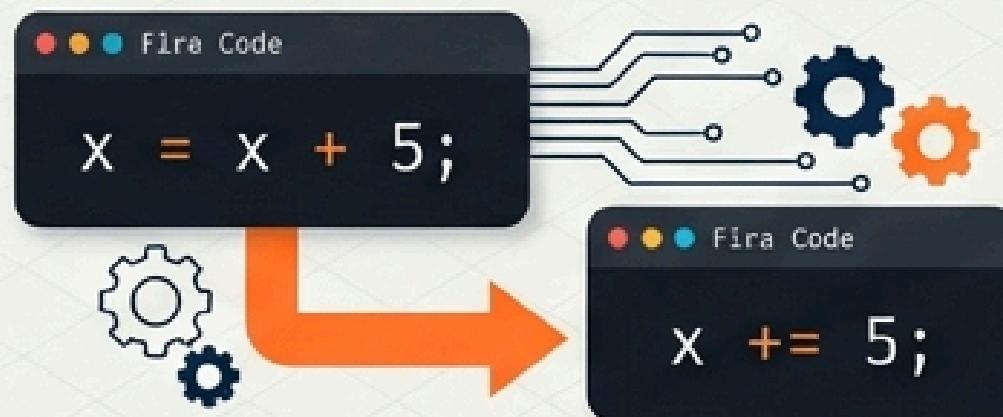
Ideal para zerar contagens (relógio, carrossel).
`contador % 10 → O valor nunca passará de 9.`

Divisibilidade

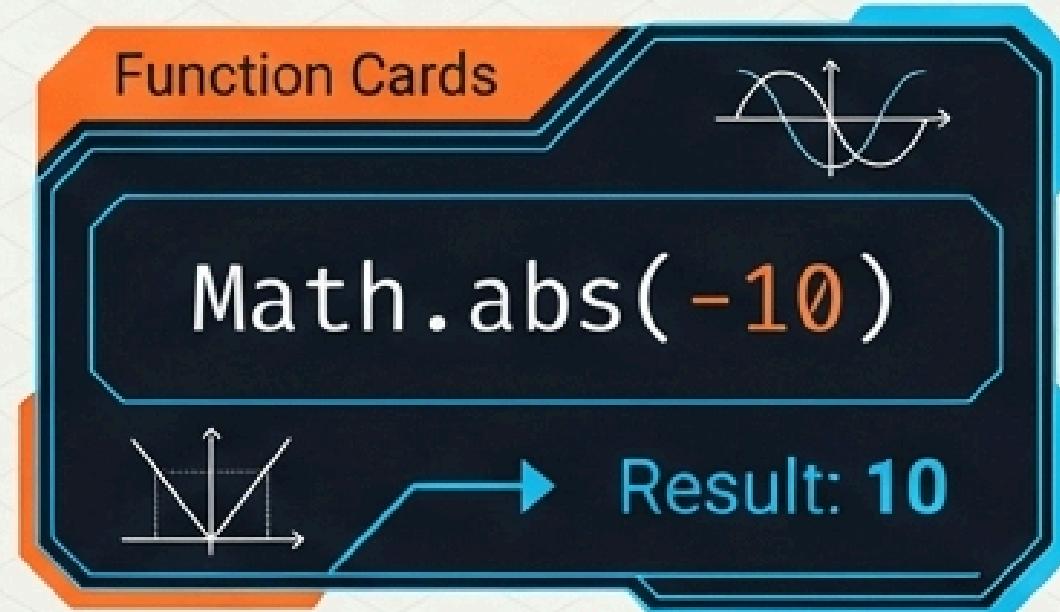
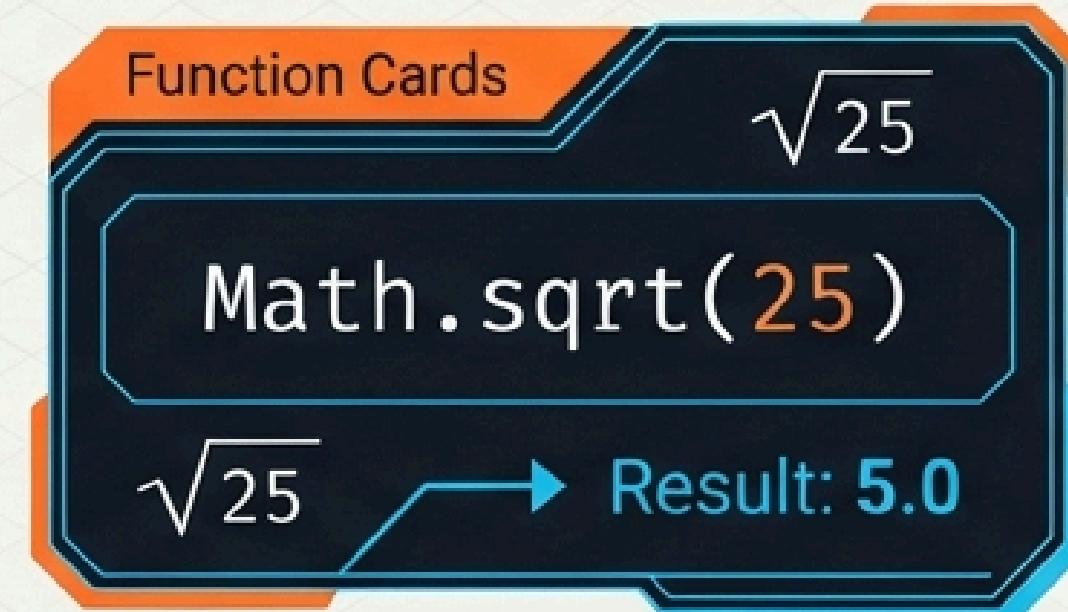
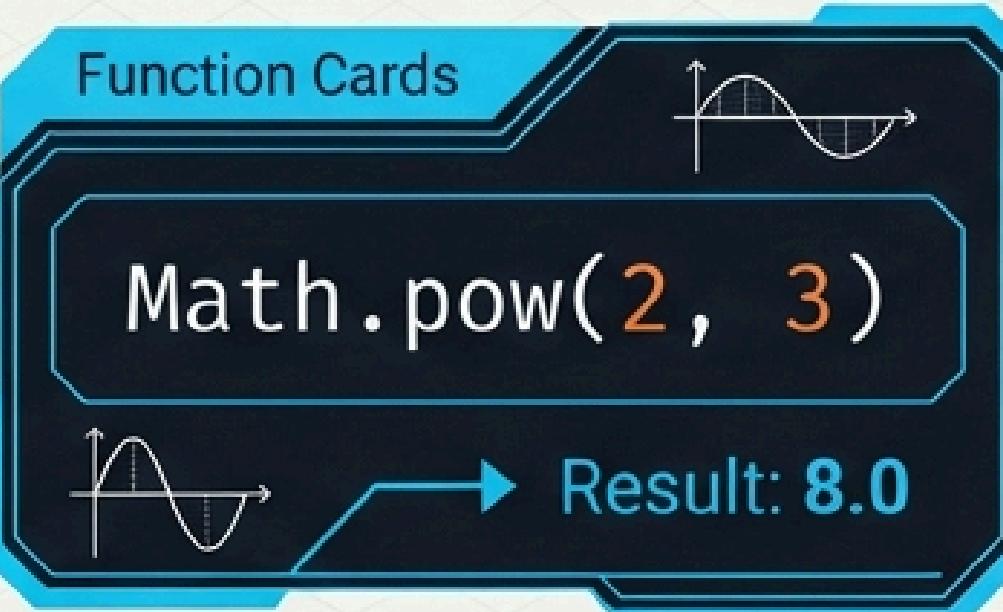
Verificar múltiplos.
`n % 5 == 0 → É múltiplo de 5.`

Ferramentas de Eficiência e Cálculos Avançados

Operadores de Atribuição (Shortcuts)

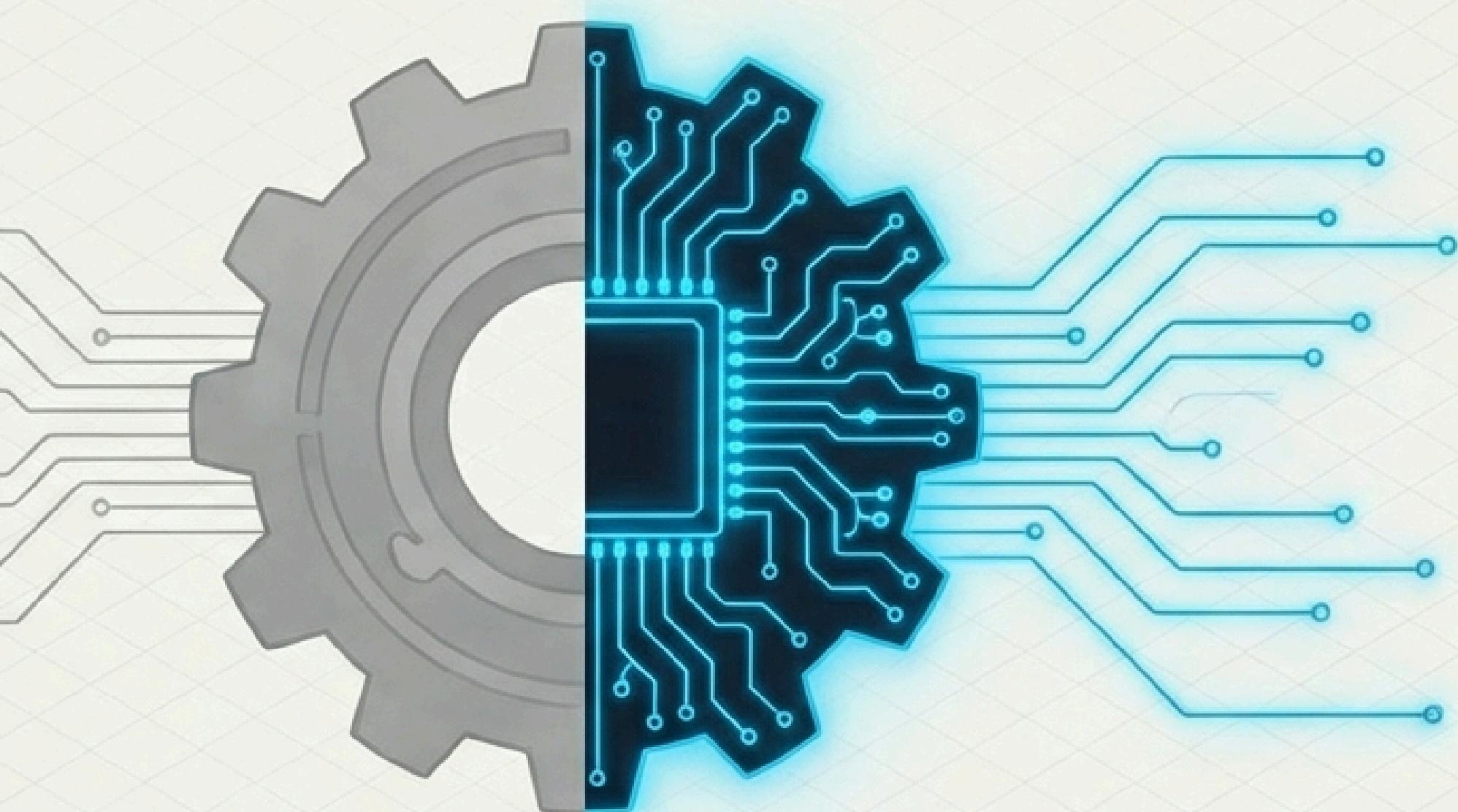


A Classe Math (Cálculos Científicos)



Parte II: Tomando Decisões

Agora que sabemos calcular, precisamos ensinar o código a reagir a diferentes situações.



O “E se?” do Java.



O Porteiro do Código: if e else

A Porta de Entrada
Roboto Serif

```
Fira Code  
if (idade >= 18) {  
    System.out.println('Pode entrar.');//  
} else {  
    System.out.println('Entrada proibida.');//  
}
```

O Plano B
Roboto Serif



Os Juízes: Operadores de Comparaçāo

Para o **if** funcionar, ele precisa de uma pergunta que resulte em Sim (True) ou Não (False).

`==`

Igual a

Atenção: Use dois iguais!

`!=`

Diferente de

`>`

Maior que

`<`

Menor que

`>=`

Maior ou igual

`<=`

Menor ou igual



Lógica Composta: &&, || e !

Combinando cenários: "Se eu tiver dinheiro" E "se a loja estiver aberta".

Fira Code

```
if (temDinheiro && lojaAberta) {  
    System.out.println('Vou comprar o jogo!');  
}
```

&& (AND)



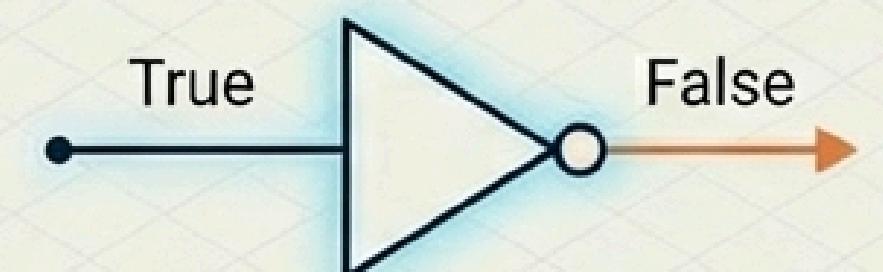
Exigente. As duas condições precisam ser verdadeiras.

|| (OR)



Flexível. Basta uma condição ser verdadeira.

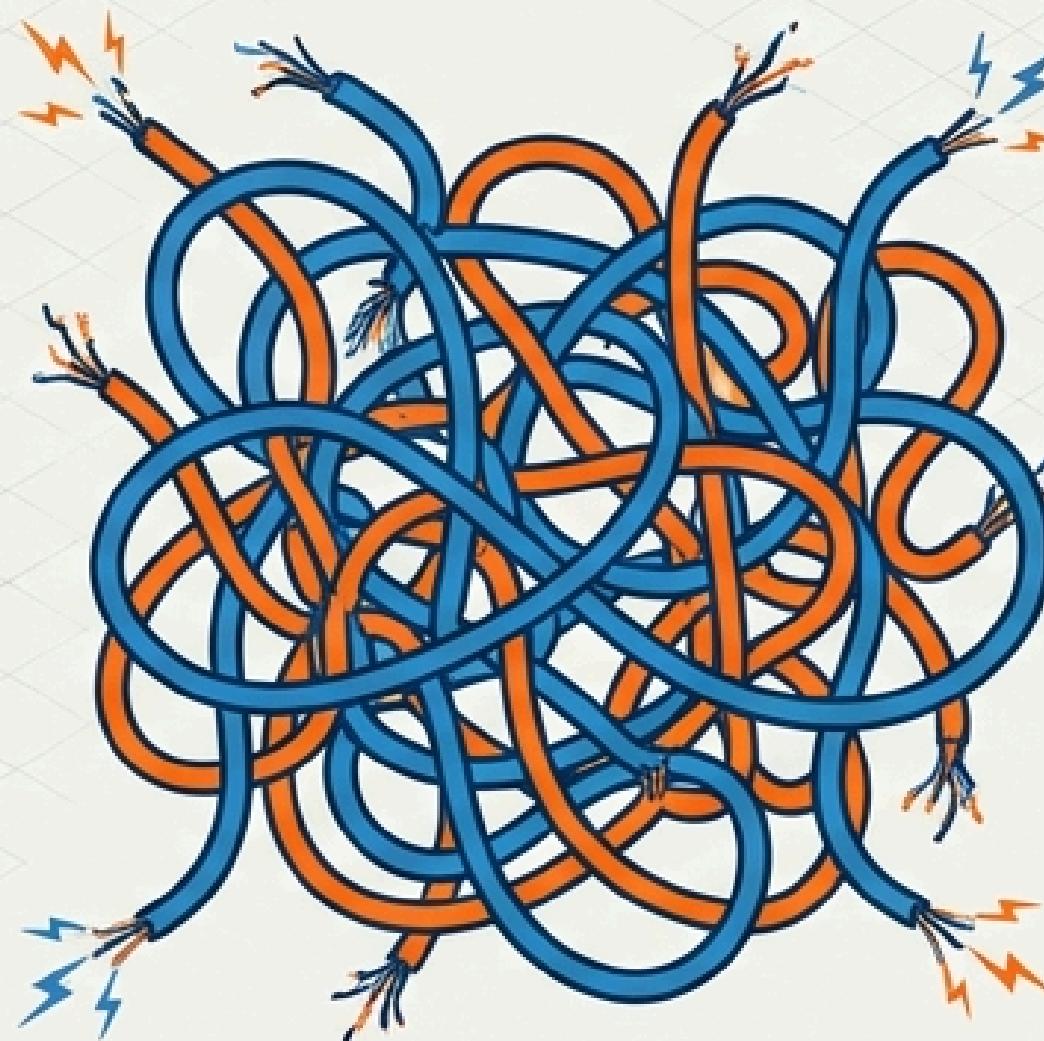
! (NOT)



Do Contra. Inverte o valor (True vira False).

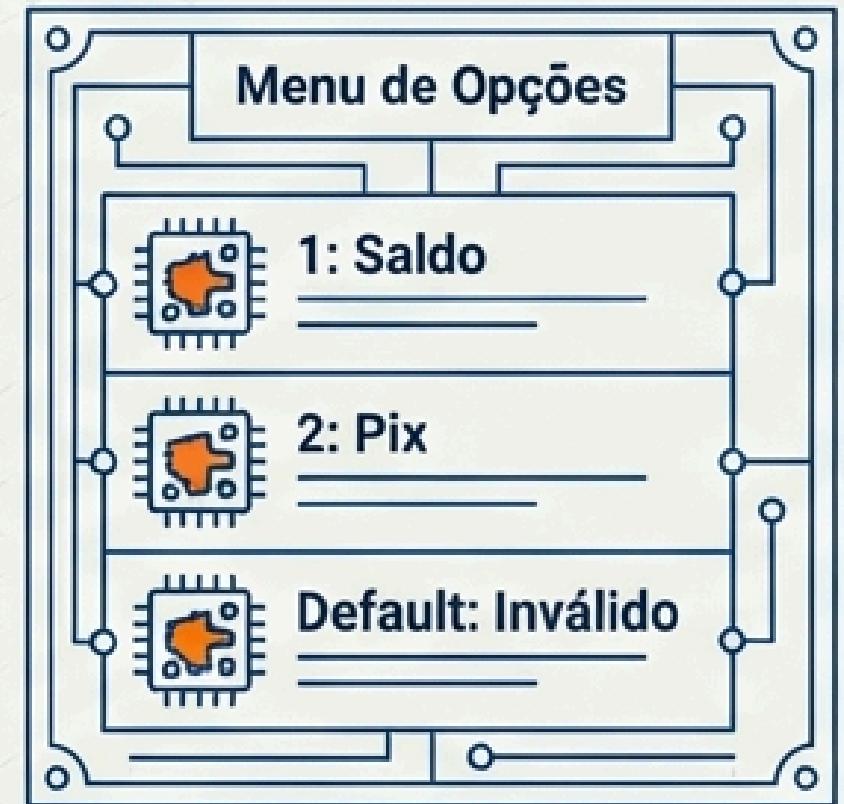
O Menu de Opções: Comando switch

Vários if / else



Bagunçado para muitas opções fixas.

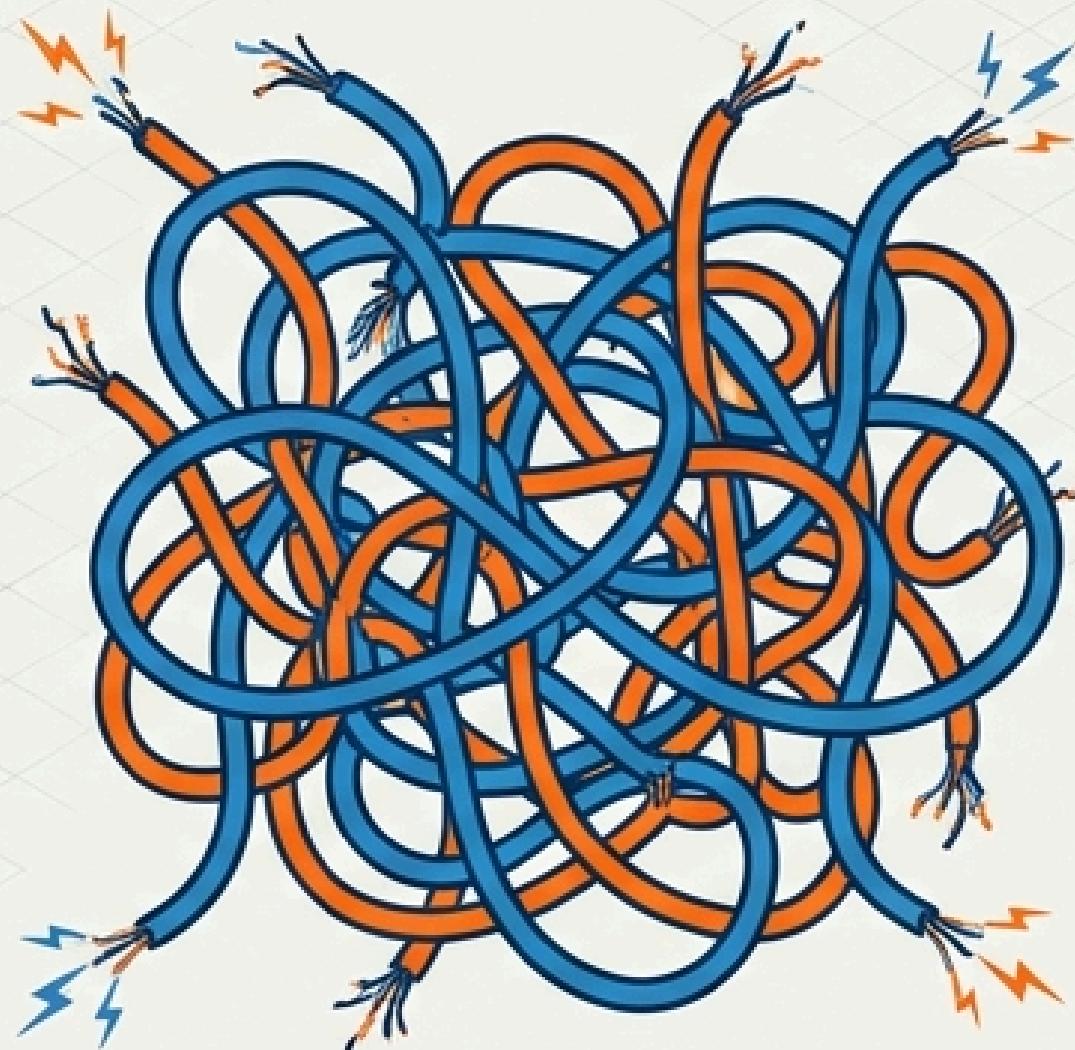
Estrutura Switch



```
switch (opcao) {  
    case 1: System.out.println('Saldo'); break;  
    case 2: System.out.println('Pix'); break;  
    default: System.out.println('Inválido');  
}
```

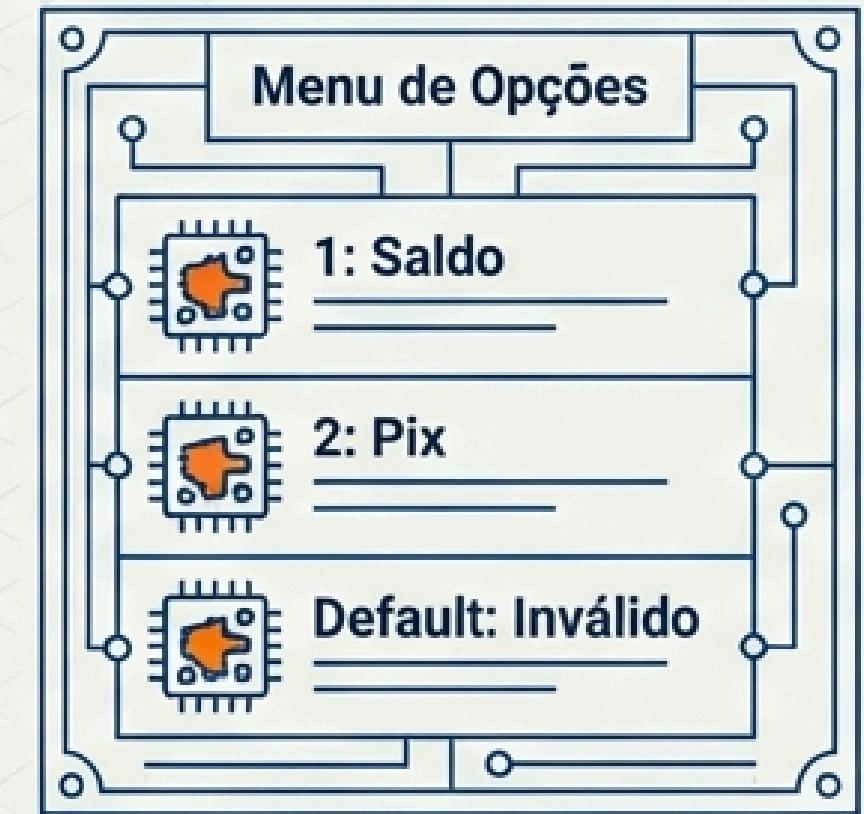
O Menu de Opções: Comando switch

Vários if / else



Bagunçado para muitas opções fixas.

Estrutura Switch



```
switch (opcao) {  
    case 1: System.out.println('Saldo'); break;  
    case 2: System.out.println('Pix'); break;  
    default: System.out.println('Inválido');  
}
```

O Freio de Mão: A Importância do break

```
case 1:  
    System.out.println('Saldo');  
    break;  
=====
```



Sem o break, o Java sofre de 'Fall-through' – ele continua executando os casos de baixo automaticamente!

Regra: Sempre “freue” seus casos no switch.

Resumo da Ópera



Atenção aos Tipos

Divisão de inteiros corta os decimais ($5/2 = 2$). Use 'double' para precisão.



O Poder do Resto

Use o Módulo (%) para lógica de ciclos, par/ímpar e divisibilidade.



Controle o Fluxo

Use if/else para condições binárias e switch para menus. Não esqueça o break!

