

Comandos Condicionais em Dart

Os comandos condicionais são a espinha dorsal de qualquer aplicação robusta, permitindo que o código reaja a diferentes situações e tome decisões inteligentes. Eles são fundamentais para o controle de fluxo lógico e essenciais para construir aplicativos Dart reativos e dinâmicos.



O Que São Comandos Condicionais?



Blocos de Código Seletivos

Executam código apenas se uma condição booleana específica for verdadeira, garantindo a execução lógica.



Caminhos de Execução Divergentes

Permitem que o programa siga diferentes rotas, adaptando seu comportamento dinamicamente.



Base da Lógica de Negócios

Constituem a fundação para 90% da inteligência e interatividade em aplicações modernas.



Melhora de Interatividade

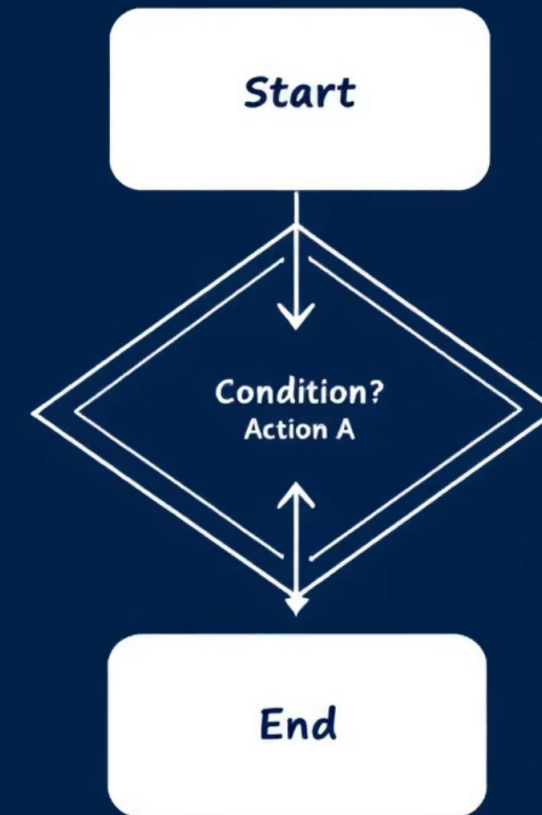
Tornam o software mais responsivo e "inteligente" ao usuário, oferecendo experiências personalizadas.

`if` e `else`: A Decisão Binária

O comando `if` é a estrutura condicional mais básica e fundamental em Dart. Ele permite que um bloco de código seja executado somente se uma condição booleana específica for avaliada como `true`.

Complementarmente, o bloco `else` é executado caso a condição do `if` seja `false`, oferecendo um caminho alternativo de execução.

Esta sintaxe é simples e direta, ideal para decisões binárias onde há apenas dois resultados possíveis.



```
if (idade >= 18) {  
    print('Maior de idade');  
} else {  
    print('Menor de idade');  
}
```

`else if`: Múltiplas Condições em Sequência

#

Verificação Sequencial

Permite verificar diversas condições em uma ordem específica, otimizando o fluxo de decisão.

∞

Ideal para N Opções

Perfeito quando o cenário apresenta mais de duas alternativas de execução para o programa.

≡

Avaliação Top-Down

As condições são avaliadas de cima para baixo, e o primeiro bloco true é executado, ignorando os demais.

```
double nota = 85.0;
if (nota >= 90) {
    print('Conceito: A');
} else if (nota >= 70) {
    print('Conceito: B');
} else if (nota >= 50) {
    print('Conceito: C');
} else {
    print('Conceito: D');
}
```

`switch` Statement: Casos Discretos

O comando switch é uma alternativa elegante para lidar com múltiplos valores discretos, como enums, strings ou inteiros. Ele avalia uma única expressão e compara seu resultado com os valores definidos nos blocos case.

É crucial usar break ao final de cada case (em Dart <2.12) para evitar o "fall-through", onde a execução continuaria para o próximo case. O bloco default é opcional e serve como um fallback para casos não correspondidos.

```
String diaSemana = 'Ter';
```

```
switch (diaSemana) {
  case 'Seg': print('É segunda-feira.');
```

```
      break;
```

```
  case 'Ter': print('É terça-feira.');
```

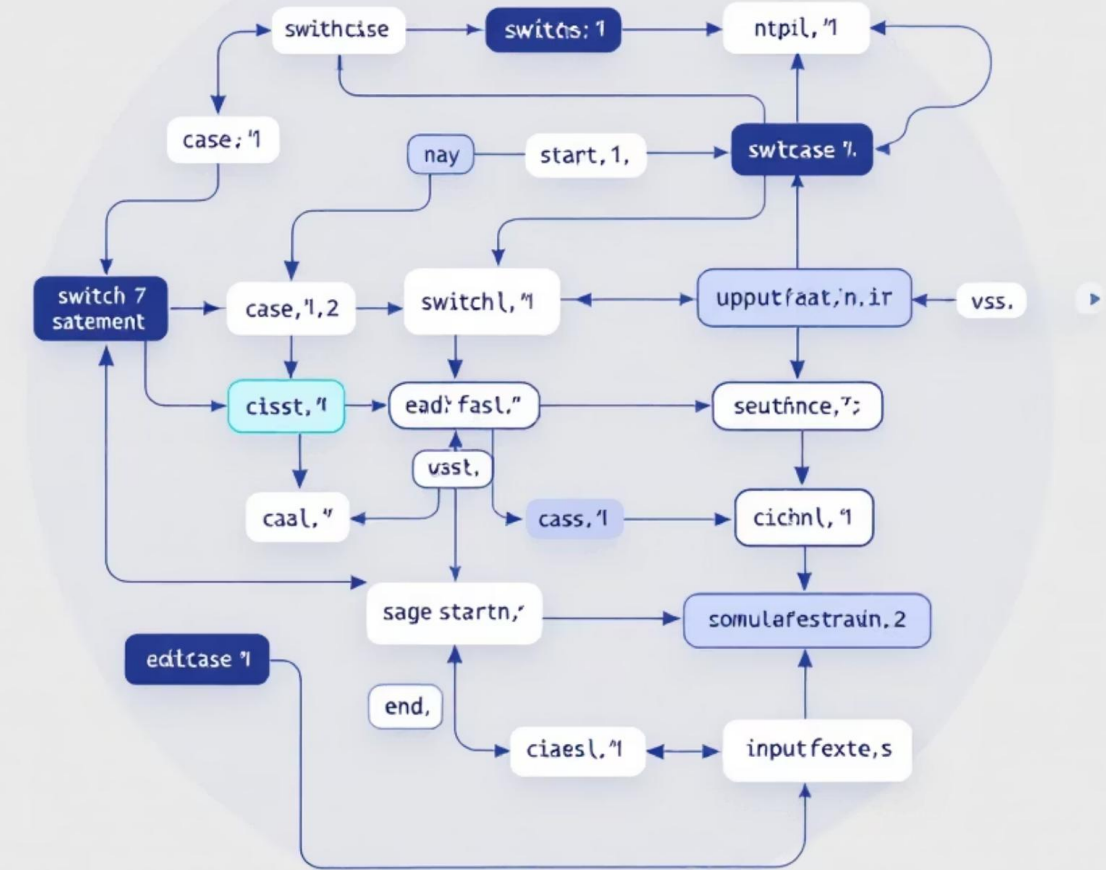
```
      break;
```

```
  case 'Qua': print('É quarta-feira.');
```

```
      break;
```

```
  default: print('Outro dia da semana.');
```

```
}
```



Operador Ternário: Concision na Atribuição

Sintaxe Compacta

Oferece uma maneira concisa de escrever uma instrução `if-else` em uma única linha.

Atribuições e Expressões

Ideal para atribuir valores a variáveis ou retornar expressões baseadas em uma condição.

Legibilidade Aprimorada

Aumenta a clareza do código para lógicas condicionais simples, reduzindo a verbosidade.

Redução de Linhas

Pode reduzir em até 70% o número de linhas para certas operações condicionais, tornando o código mais enxuto.

```
double saldo = 1500.0;
String status = (saldo > 0) ? 'Ativo' : 'Inativo';
print('Status da conta: $status');
int idade = 20;
String mensagem = (idade >= 18) ? 'Pode votar' : 'Não pode votar';
print(mensagem);
```


Boas Práticas e Erros Comuns

Clareza na Expressão

Use parênteses `()` para desambiguar expressões complexas, melhorando a leitura.

Entendimento do Escopo

Variáveis declaradas dentro de blocos condicionais são **locais** para esses blocos.

Gerenciamento de `null` Safety


Utilize `!` (assert), `?` (nullable) e `??` (fallback) para lidar com valores `null`.

Evitar Aninhamento Excessivo

Refatore seu código com funções ou `switch` para melhorar a **legibilidade**.

Ordem do `else if`

Sempre posicione as condições **mais específicas** primeiro para evitar erros lógicos de avaliação.



Conclusão: Dominando a Lógica Condicional

Os comandos condicionais são pilares essenciais da programação em Dart. Através do uso estratégico de `if-else`, `else if`, `switch` e do operador ternário, você pode construir código robusto, flexível e eficiente.

A prática constante dessas estruturas de controle de fluxo é o caminho para a maestria, permitindo que seus aplicativos tomem decisões inteligentes e forneçam experiências de usuário dinâmicas.