



# If Ternário e Switch



**Certified  
Developer**  
The Ultimate Tech Degree

**DigitalHouse** >  
Coding School



# Temas

**1**

If Ternário

**2**

Switch



**1**

**If Ternário**

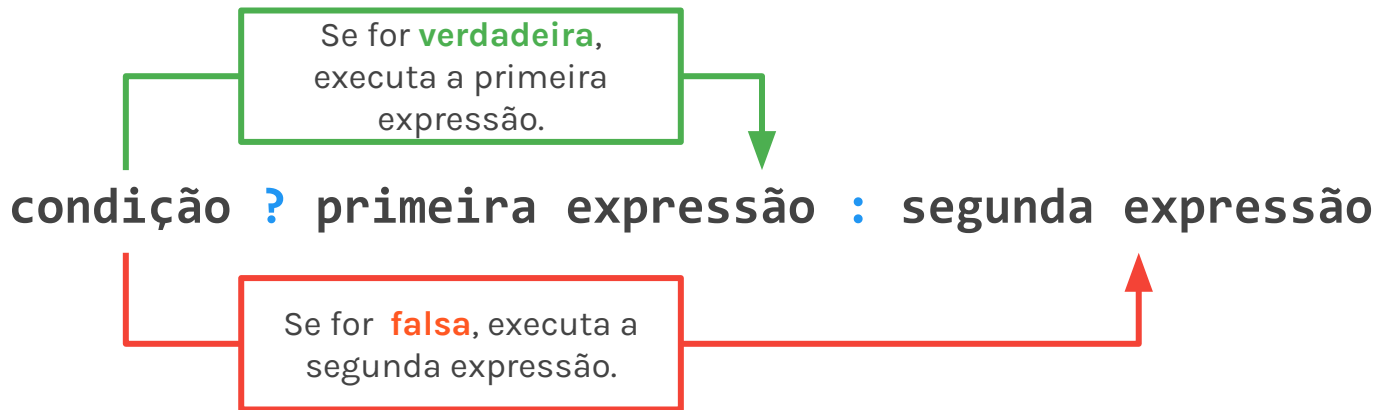


Como mencionamos antes: se algo é **muito usado** em programação, as linguagens costumam nos dar uma **versão abreviada**.



# Estrutura básica

Ao contrário de um if tradicional, o **if ternário** é escrito **horizontalmente**. Como o if tradicional, ele tem o mesmo fluxo (se esta condição for verdadeira, faça isso, se não, faça aquilo), mas neste caso, você não precisa escrever a palavra **if** ou a palavra **else**.



# Estrutura básica

Para o ternário, é obrigatório colocar código na **segunda expressão**. Se não quisermos que nada aconteça, podemos usar uma string vazia `''`.

```
{ } 4 > 10 ? '0 4 é o maior' : '0 10 é o maior';
```

## Condição

Declaramos uma expressão, que retornará **true** ou **false**.

## Primeira expressão

Se a condição for verdadeira, o código após o ponto de interrogação será executado.

## Segunda expressão

Se a condição for falsa, o código após os dois pontos será executado.

É obrigatório escrevê-lo.

## 2 | Switch



O switch nos oferece uma sintaxe  
**mais legível** para os casos em que  
queremos avaliar **muitas**  
**possibilidades** de um único valor.





# Estrutura básica

O **switch** é composto por um valor a ser avaliado, seguido de diferentes **cases** (caso, em português). Podemos adicionar quantos quisermos, cada um contemplando um cenário diferente.

Cada **case** deve terminar com a palavra-chave **break** para evitar que o próximo bloco seja executado.

```
{  
  switch (valor) {  
    case valorA:  
      // código a ser executado se a expressão for igual a valorA.  
      break;  
    case valorB:  
      // código a ser executado se a expressão for igual a valorB.  
      break;  
  }  
}
```

# Agrupamento de casos

O switch também nos permite **agrupar casos** e executar o mesmo bloco de código para qualquer caso naquele grupo.

```
{}  
switch (valor) {  
    case valorA:  
    case valorB:  
        // código a ser executado se a expressão for igual a valorA  
        // ou valorB.  
        break;  
    case valorC:  
        // código a ser executado se valorC for verdadeiro  
        break;  
}
```

# {código}

```
let idade = 5;
```

Definimos a variável **idade** e atribuímos o número 5 a ela.

```
switch (idade) {  
  case 10:  
    console.log('Tem 10 anos');  
    break;  
  case 5:  
    console.log('Tem 5 anos');  
    break;  
}
```

# {código}

```
let idade = 5;
```

```
switch (idade) {  
  case 10:  
    console.log('Tem 10 anos');  
    break;  
  case 5:  
    console.log('Tem 5 anos');  
    break;  
}
```

Começamos a condicional com a palavra reservada **switch** e, entre parênteses, a expressão/condição que queremos avaliar.

Neste caso, vamos **avaliar o valor da variável idade**.

# {código}

```
let idade = 5;
```

```
switch (idade) {
```

```
  case 10:
```

```
    console.log('Tem 10 anos');
```

```
    break;
```

```
  case 5:
```

```
    console.log('Tem 5 anos');
```

```
    break;
```

```
}
```

Para cada caso, escrevemos a palavra reservada **case** e, em seguida, o valor que queremos avaliar.

Nesse caso, **perguntamos se o valor da variável idade é 10.**

Como este caso **NÃO é verdadeiro**, o JavaScript ignora o código neste **case** e avança para avaliar o próximo.



# {código}

```
let idade = 5;
```

```
switch (idade) {  
  case 10:  
    console.log('Tem 10 anos');  
    break;  
  
  case 5:  
    console.log('Tem 5 anos');  
    break;  
}
```

Este caso é verdadeiro, portanto o **código do bloco** será executado.

A palavra-chave **break** encerra a execução.

Se esquecermos o **break**, os blocos continuarão a ser executados independentemente de os casos serem atendidos ou não.



# O bloco default

Se quisermos considerar a possibilidade de que **nenhum dos casos** seja verdadeiro, usamos a palavra-chave **default** seguida por dois pontos **:** e o bloco de código que queremos executar.

Normalmente **escrevemos o bloco default por último**. Nesse caso, não é necessário escrever a palavra reservada **break**.

```
{}  
switch (valor) {  
    case valorA:  
        // código a ser executado se valorA for verdadeiro.  
        break;  
    default:  
        // código a ser executado se nenhum dos casos for verdadeiro.  
}
```

# {código}

```
let fruta = 'wefwef';  
switch (fruta) {  
  case 'banana':  
    console.log('Uma fruta amarela');  
    break;  
  case 'laranja':  
    console.log('Bem ácida!');  
    break;  
  default:  
    console.log('Qual fruta é?');  
    break;  
}
```

Definimos a expressão que vamos avaliar no switch.

Neste caso, queremos pedir o valor da variável fruta.



# {código}

```
let fruta = 'wefwef';
```

```
switch
```

```
  case 'banana':  
    console.log('Uma fruta amarela');  
    break;
```

Este case é **falso**, portanto seu código não é executado.

```
  case 'laranja':  
    console.log('Bem ácida!');  
    break;  
  default:  
    console.log('Qual fruta é?');  
    break;
```

```
}
```

# {código}

```
let fruta = 'wefwef';  
switch  
  case 'banana':  
    console.log('Uma fruta amarela');  
    break;  
  case 'laranja':  
    console.log('Bem ácida!');  
    break;  
  default:  
    console.log('Qual fruta é?');  
    break;  
}
```

Este case também é **falso**, portanto, seu código não é executado.

# {código}

```
let fruta = 'wefwef';  
switch  
  case 'banana':  
    console.log('Uma fruta amarela');  
    break;  
  case 'laranja':  
    console.log('Bem ácida!');  
    break;  
  default:  
    console.log('Qual fruta é?');  
}
```

Caso nenhum dos cases seja verdadeiro, o código é executado dentro do bloco **default**.



DigitalHouse>  
Coding School