Lógica de Programação - Aula 4

Santander Coders 2024

Condicional (If / else if/ switch)

IF / else if

Uma estrutura de controle condicional é uma parte fundamental da programação que permite que um programa tome decisões com base em condições específicas. O **if** e o **else if** são exemplos de tais estruturas, que permitem que você execute diferentes blocos de código com base em condições booleanas (**verdadeiras ou falsas**).

A estrutura if

A estrutura if é usada para executar um bloco de código somente se uma condição especificada for verdadeira.

```
const numero = 10;

if (numero > 5) {
  console.log('O número é maior do que 5.');
}
```

Neste exemplo, o código dentro do bloco if só será executado se a condição **numero > 5** for verdadeira.

A estrutura else if

A estrutura **else if** é usada quando você deseja verificar várias condições diferentes sequencialmente. Ela segue um **if** e é executada se a condição do **if** for **falsa** e a condição **else if for verdadeira**. A sintaxe é a seguinte:

```
const numero = 10;

if (numero > 15) {
  console.log('O número é maior do que 15.');
} else if (numero > 5) {
  console.log('O número é maior do que 5, mas não maior do que 15.');
} else {
  console.log('O número é igual ou menor do que 5.');
}
```

Neste exemplo, o código dentro do bloco else if é executado porque a primeira condição (numero > 15) é falsa, mas a segunda condição (numero > 5) é verdadeira.

Resumo if else

- Você pode ter múltiplos blocos else if após um if para verificar várias condições.
- A estrutura else é opcional e é usada para especificar um bloco de código a ser executado se nenhuma das condições anteriores for verdadeira.
- É importante lembrar que apenas um dos blocos condicionais (o primeiro que for verdadeiro) será executado.
- As condições são avaliadas de cima para baixo e, assim que uma condição verdadeira é encontrada, os blocos associados a essa condição são executados e a execução continua após a estrutura condicional.

Essas estruturas são fundamentais para controlar o fluxo de um programa e tomar decisões com base em condições específicas.

Condicional ternário (? :)

Condicional ternário (? :)

O operador condicional ternário, também conhecido como operador ternário, é uma maneira concisa de escrever uma instrução if/else em uma única linha. Ele permite que você tome uma decisão com base em uma condição e retorne um valor dependendo se a condição é verdadeira ou falsa. A sintaxe geral do operador ternário é a seguinte:

```
condição ? valorSeVerdadeiro : valorSeFalso
```

- **condição**: A expressão que será avaliada como verdadeira ou falsa.
- **valorSeVerdadeiro**: O valor retornado se a condição for verdadeira.
- **valorSeFalso**: O valor retornado se a condição for falsa.

```
const idade = 20;
const possoDirigir = idade >= 18 ? "Pode Dirigir!" : "Não Pode Dirigir!";

console.log(possoDirigir)  'Pode Dirigir!'
```

Truthy e Falsy

Valores Verdadeiros (Truthy)

```
if ("Hello") {
    console.log("Esta string é Truthy.");
}

if (42) {
    console.log("Este número é Truthy.");
}

if ({}) {
    console.log("Este objeto vazio é Truthy.");
}

if (function() {}) {
    console.log("Esta função vazia é Truthy.");
}
```

Valores Falsos (Falsy)

```
if (false) {
    console.log("Este valor booleano é Falsy.");
}

if (0) {
    console.log("Este número zero é Falsy.");
}

if (null) {
    console.log("Este valor nulo é Falsy.");
}

if (undefined) {
    console.log("Este valor undefined é Falsy.");
}
```

Vamos a Prática!

- Verificação de Números Pares e Ímpares (If e Ternário)

- Escreva um programa que recebe um número como entrada e verifica se é par ou ímpar. Imprima "É par" se for par e "É ímpar" se for ímpar.

Desafio 2

- Verificação de Nota

- Crie um programa que recebe uma nota como entrada e atribui uma mensagem com base na nota. Use as seguintes regras:

- Se a nota for maior ou igual a 90, imprima "Aprovado com mérito".
- Se a nota for maior ou igual a 70 e menor que 90, imprima "Aprovado".
- Se a nota for menor que 70, imprima "Reprovado".

Desafio 3

- Determinação do Maior Número

- Escreva um programa que receba três números como entrada e determina o maior deles. Imprima o número mais alto.

- Verificação de Triângulo

- Crie um programa que recebe três comprimentos de lados de um triângulo como entrada e determina se eles formam um triângulo equilátero (Todos os lados são iguais), isósceles (Dois lados são iguais) ou escaleno (Se nada é igual). Imprima a classificação do triângulo.

Desafio 5

- Verificação de Ano Bissexto

- Desenvolva um programa que receba um ano como entrada e verifica se ele é bissexto ou não. Um ano bissexto é aquele que é divisível por 4, exceto por anos que são divisíveis por 100, a menos que sejam divisíveis por 400. Imprima "Ano bissexto" ou "Não é um ano bissexto" com base na entrada. (2000, 1996)

Desafio 6

- Verificação de Idade que é permitido dirigir (If e Ternário)

- Crie um programa que verifica a idade de uma pessoa e determina se ela pode dirigir ou não. Se a pessoa tiver 18 anos ou mais, ela pode dirigir; caso contrário, não pode.

switch

Switch

O **switch** é uma estrutura de controle condicional em programação que permite que você selecione um dos vários blocos de código a serem executados com base no valor de uma expressão. É uma alternativa ao uso de múltiplas instruções **if e else if** quando você precisa comparar uma única expressão com vários valores diferentes.

Sintaxe básica do switch

```
switch (expressao) {
  case 1:
    // Código a ser executado se a expressao for igual a valor1
    break;
  case 2:
    // Código a ser executado se a expressao for igual a valor2
    break;
  // Outros casos...
  default:
  // Código a ser executado se a expressao não corresponder a nenhum dos casos anteriores
}
```

- **expressao**: É a expressão que você deseja avaliar.

case valor1: Cada case é um valor específico que a expressão pode assumir. Se a **expressao** for igual a **valor1**, o código associado a esse case será executado.

- **break:** A palavra-chave break é usada para sair do bloco switch após o código de um case ser executado. Isso evita que os blocos subsequentes sejam executados acidentalmente.
- **default:** O bloco default é opcional e é executado se a expressão não corresponder a nenhum dos casos especificados.

Exemplo de uso:

```
let nomeDoDia;
switch (dia) {
  case 1:
    nomeDoDia = 'Domingo';
    break;
  case 2:
    nomeDoDia = 'Segunda-feira';
    break;
  case 3:
    nomeDoDia = 'Terça-feira';
    break;
  case 4:
    nomeDoDia = 'Quarta-feira';
    break;
  case 5:
    nomeDoDia = 'Quinta-feira';
    break;
  case 6:
    nomeDoDia = 'Sexta-feira';
    break;
  case 7:
    nomeDoDia = 'Sábado';
    break;
  default:
    nomeDoDia = 'Dia inválido';
}

console.log('Hoje é ' + nomeDoDia);
```

Vantagens do switch

- O switch é útil quando você precisa comparar uma única expressão com vários valores diferentes, tornando o código mais legível.
- Ele é mais eficiente do que uma série de instruções if e else if em termos de desempenho, especialmente quando há muitos casos.

Limitações do switch

- O switch só pode ser usado para comparar igualdade estrita (o operador ===), o que significa que ele não funciona bem com comparações complexas ou intervalos de valores.

Bora Praticar!

Desafio 7

- Verificação de Números Pares e Ímpares

- Escreva um programa que recebe um número como entrada e verifica se é par ou ímpar. Imprima "É par" se for par e "É ímpar" se for ímpar. (Usar switch Case)

Desafio 8

- Conversão de Notas em Conceitos

- Faça um programa que peça ao usuário para digitar uma letra e verifique se é uma vogal ou uma consoante utilizando o comando switch case. Se o usuário digitar uma vogal (a, e, i, o, u), o programa deve exibir a mensagem "É uma vogal". Se o usuário digitar uma consoante, o programa deve exibir a mensagem "É uma consoante".

Desafio 9

- Determinação de Estação do Ano

- Escreva um programa que recebe o nome de um mês como entrada e utiliza um switch case para determinar a estação do ano correspondente. Use a seguinte correspondência de meses:

- Dezembro, Janeiro, Fevereiro: "Inverno" Março, Abril, Maio: "Primavera" Junho, Julho, Agosto: "Verão" Setembro, Outubro, Novembro: "Outono"

Operadores de Coalescência ?? e ?.

Operadores de Coalescência ?? e ?.

O operador ?? é usado para fornecer um valor padrão quando o valor à esquerda é nulo (null) ou indefinido (undefined). Se o valor à esquerda não for nulo ou indefinido, o operador ?? retornará esse valor. Caso contrário, ele retornará o valor à direita.

```
const valorNaoNulo = "Olá";
const valorNulo = null;

const resultado1 = valorNaoNulo ?? "Valor padrão";
const resultado2 = valorNulo ?? "Valor padrão";

console.log(resultado1); // Output: "Olá"
console.log(resultado2); // Output: "Valor padrão"
```

No primeiro caso, valorNaoNulo é uma string não nula, então o operador ?? retorna o próprio valor da esquerda. No segundo caso, valorNulo é nulo, então o operador ?? retorna o valor padrão "Valor padrão".

Operador de Acesso Opcional "?"

Operador de Acesso Opcional ?

O operador ?. é usado para acessar propriedades de objetos de forma segura, verificando se o objeto ou a propriedade que você está tentando acessar existem. Se o objeto ou a propriedade não existirem, a expressão retornará undefined, em vez de causar um erro.

```
const pessoa = {
    nome: "João",
    endereco: {
        cidade: "São Paulo",
    },
};

const cidade = pessoa.endereco?.cidade;
const cep = pessoa.endereco?.cep;

console.log(cidade); // Output: "São Paulo"
console.log(cep); // Output: undefined
```

Vamos a prática!