




If e Switch

Resumo Instruções If

Olá, pessoal! Hoje vamos abordar as instruções em JavaScript, um tema fundamental para quem está se aventurando no mundo da programação. As instruções são elementos essenciais para controlar o fluxo de um programa e tomar decisões com base em determinadas condições.

Entenderemos o conceito geral das instruções e sua importância no desenvolvimento de software. As instruções são comandos que direcionam o comportamento do programa, permitindo que ele execute diferentes ações dependendo das condições encontradas. Essas instruções são aplicáveis em várias linguagens de programação, e o que vamos aprender aqui se aplica também ao JavaScript.

Primeiramente, vamos focar na instrução “if”, que permite criar ramificações no código de acordo com condições específicas. O “if” é usado para criar uma estrutura condicional, na qual um bloco de código é executado somente se uma determinada condição for verdadeira. A sintaxe básica do “if” consiste em abrir parênteses, colocar a condição a ser testada e fechar os parênteses. Se a condição for verdadeira, o bloco de código delimitado por chaves será executado.

Um exemplo simples para ilustrar o uso do “if” seria o seguinte: suponhamos que temos a variável “a” e queremos verificar se ela  maior que 2. Se isso for verdadeiro, queremos que determinadas ações sejam realizadas. Nesse caso, usamos o “if” da seguinte maneira: `“if (a > 2) { // bloco de código a ser executado }”`. O bloco de código dentro das chaves será executado somente se “a” for de fato maior que 2.


No entanto, é importante lembrar que nem sempre queremos executar apenas um bloco de código em uma situação condicional. Podemos utilizar a estrutura “else if” para testar condições adicionais caso a primeira não seja satisfeita. O “else if” nos permite criar uma ramificação no código, executando um bloco específico se uma nova condição for verdadeira. Essa estrutura pode ser encadeada quantas vezes forem necessárias para atender às diversas possibilidades.

Por exemplo, podemos ter um caso em que precisamos testar se um número é maior que 10. Se for, executamos um bloco de código. Caso contrário, testamos se ele é menor que 5 e, se for, executamos outro bloco de código. Caso nenhuma dessas condições seja verdadeira, podemos utilizar o “else” para executar um bloco padrão.

É importante ressaltar que o uso adequado das instruções “if”, “else if” e “else” permite controlar o fluxo do programa de forma mais precisa e eficiente. Ao entender as possibilidades dessas instruções e como combiná-las, você será capaz de criar lógicas complexas e responder a diferentes cenários.

Mão na massa usando if

Agora, vamos colocar em prática o que aprendemos. Vamos utilizar a IDE para criar um código que ilustre o funcionamento das instruções condicionais.

Criação do arquivo instrução.js: Para começar, crie um novo arquivo chamado instrução.js na sua IDE preferida. 

Utilizando o “if” e “else”: Vamos utilizar a variável país e atribuir o valor “Brasil” a ela. Em seguida, verificaremos se o valor da variável é diferente de “Brasil” utilizando a instrução condicional “if”. Caso seja verdadeira, exibiremos a mensagem “Você é estrangeiro” no console. Caso contrário, exibiremos a mensagem “Você é brasileiro”. Essa é uma forma simples de utilizar o “if” em uma situação real. Observe o código abaixo:

```
var país = "Brasil";

if (país !== "Brasil") {
  console.log("Você é estrangeiro");
} else {
  console.log("Você é brasileiro");
}
```

Utilizando o “else if”: Agora, vamos criar uma nova variável chamada idade e atribuir o valor 22 a ela. Com base nessa variável, realizaremos mais verificações utilizando a instrução “else if”. Primeiro, verificaremos se a idade é menor que 16. Se for, exibiremos a mensagem “Você não vota”. Caso contrário, verificaremos se a idade é menor que 18. Se essa condição for verdadeira, exibiremos a mensagem “O voto é opcional”. Por fim, utilizaremos o “else” para exibir a mensagem “O voto é obrigatório” quando nenhuma das condições anteriores for atendida. Veja:



```
var idade = 22;

if (idade < 16) {
  console.log("Você não vota");
} else if (idade < 18) {
  console.log("O voto é opcional");
} else {
  console.log("O voto é obrigatório");
}
```

Combinação de condições com operadores lógicos: Além das estruturas condicionais apresentadas, podemos combinar condições utilizando os operadores lógicos “e” (&&) e “ou” (||). Por exemplo, se a idade for maior que 65 anos ou menor que 16 anos, podemos exibir a mensagem “O voto é opcional”. Utilizando o operador “ou”, se uma dessas condições for verdadeira, a mensagem será exibida. Observe o código abaixo:

```
if (idade > 65 || idade < 16) {
  console.log("O voto é opcional");
}
```

Utilizar os operadores lógicos “e” e “ou” é uma forma bastante comum de combinar múltiplas condições em uma única instrução “if”.

Agora que colocamos a mão na massa e praticamos o uso das instruções condicionais “if”, “else if” e “else”, além dos operadores lógicos “e” (&&) e “ou” (||), estamos mais preparados para aplicar esses conceitos em nossos programas em JavaScript. Continue praticando e explorando diferentes possibilidades para aprimorar suas habilidades em lógica de programação.

Instruções Switch

Vamos explorar outra instrução muito útil chamada “switch”. Essa instrução pode ser uma alternativa ao “if” e nos permite escrever um código mais elegante e compreensível. Vamos entender como funciona o controle de fluxo com o “switch”.

A sintaxe básica do “switch” é a seguinte: declaramos a instrução com a palavra-chave “switch”, abrimos parênteses e inserimos uma expressão que será avaliada. Em seguida, abrimos uma chave e iniciamos a estrutura dos casos (cases). Cada caso é definido com a palavra-chave “case” seguida de um valor específico e dois pontos. Dentro de cada caso, escrevemos as instruções que serão executadas. É importante mencionar que devemos finalizar cada caso com a palavra-chave “break”, que indica que o código deve sair do “switch” e continuar a execução abaixo.

O “switch” verifica se o valor da expressão corresponde a algum dos casos definidos. Se houver correspondência, o código dentro desse caso específico será executado. Podemos ter vários casos dentro do “switch” e, caso o valor da expressão corresponda a um dos casos, somente o código desse caso será executado. No entanto, se não houver correspondência entre o valor da expressão e os casos definidos, podemos utilizar a cláusula “default” para executar um bloco de código padrão.

Vamos partir para a prática!

Suponhamos que a expressão seja “banana”. O “switch” irá verificar cada caso, começando pelo primeiro. No nosso exemplo, não haverá correspondência nos primeiros casos (“laranja” e “maçã”). Porém, quando o “switch” chegar ao caso “banana”, o código dentro desse caso será executado, exibindo a mensagem “banana custa”. Em

seguida, o código sairá do “switch” e continuará a execução normalmente. 

Neste momento, vamos considerar a situação em que a expressão seja “manga”. Nesse caso, o código não entrará nos primeiros casos, mas encontrará correspondência no caso “manga”. O código dentro desse caso será executado, exibindo a mensagem “A manga custa R\$3,00 o quilo”. Vale ressaltar que não há um “break”, o que faz com que o código continue verificando os outros casos do “switch”. Embora isso possa ser útil em alguns cenários, nem sempre é necessário. Se desejarmos sair do “switch” assim que encontrarmos a correspondência adequada, basta adicionar a instrução “break”.

Por fim, vamos considerar o caso em que a expressão seja “abacate”. Nesse caso, como não há um caso correspondente para “abacate”, o bloco de código definido no “default” será executado. A mensagem “Desculpe, estamos sem nenhum abacate” será exibida. Agora que compreendemos a teoria, é hora de colocarmos a mão na massa e praticarmos a utilização do “switch” em exemplos concretos.

Praticando o uso do Switch

Chegou o momento de irmos para a IDE e colocarmos em prática o uso do Switch. Vamos sair da teoria e colocar a mão na massa para entender como essa instrução funciona. Bora lá para a IDE!

Nesta última parte da nossa aula, vamos explorar o Switch, uma instrução extremamente útil e interessante. Com o Switch, podemos lidar com várias condições e substituir sequências longas de “if”, “else if” e “else”. Isso nos permite resolver problemas complexos de maneira mais clara e concisa.

Vou mostrar um exemplo usando o Switch para que vocês possam compreender melhor seu funcionamento. Vamos supor que ter¹ uma variável chamada “diaSemana” e queremos verificar qual é o dia correspondente. Vou definir essa variável como “segunda-feira”. Em seguida, utilizaremos o Switch para verificar o valor do “diaSemana” e tomar a ação apropriada.


Se o valor de “diaSemana” for o primeiro dia da semana, que é o domingo, executaremos um bloco de código correspondente ao caso (case) do Switch. Por exemplo, podemos imprimir a mensagem “Hoje é domingo” utilizando o console.log. Faremos o mesmo para os outros casos, como o segundo dia da semana (segunda-feira) e o terceiro dia da semana (terça-feira).

No entanto, é fundamental lembrar que, para cada caso (case) em que a condição é atendida, devemos incluir a instrução “break” para sair do Switch. Isso evita a execução desnecessária dos casos subsequentes. Portanto, é importante adicionar o “break” após cada bloco de código para interromper a execução do Switch.

Vamos realizar um teste: usarei o valor “1” para a variável “diaSemana” e executarei o código. Ao utilizar esse valor, o programa verificará cada caso e imprimirá a mensagem correspondente. Vocês poderão observar que ele para assim que encontra o caso adequado, não prosseguindo para os outros casos. Isso acontece porque incluímos o “break” em cada caso.

Agora, faremos o teste para o caso “2”. Ao executar o código, ele imprimirá a mensagem “Hoje é segunda-feira”. Novamente, ele para após encontrar o caso adequado e não continua para os demais.

Por fim, consideraremos uma situação em que o valor de “diaSemana” seja “4”. Nesse caso, nenhum dos casos anteriores será atendido.

Portanto, o Switch entrará no “default”, que é executado quando nenhum caso é correspondido. No nosso exemplo, exibiremos  mensagem “Esse dia da semana não existe”. Dessa forma, podemos lidar com situações em que nenhum caso é satisfeito.

Pessoal, o Switch é amplamente utilizado para substituir sequências longas de “if”, “else if” e “else”. Ele simplifica o código e torna-o mais legível. Lembrem-se de incluir o “break” após cada caso e, se necessário, utilizem o “default” para lidar com valores não correspondentes.

Conteúdo Bônus

Recomendo o vídeo “JavaScript Tutorial for Beginners”, disponível no canal do FreeCodeCamp no YouTube. Neste tutorial em vídeo, o FreeCodeCamp oferece uma introdução abrangente ao JavaScript, abordando os conceitos básicos e fundamentais da linguagem de programação. O vídeo é gratuito e oferece uma oportunidade de aprofundar seus conhecimentos em JavaScript de forma didática e acessível. Esse tutorial irá complementar o conteúdo abordado em aula, fornecendo uma visão mais detalhada e prática sobre o uso de instruções condicionais como o “if” e o “switch” em JavaScript.

Referência Bibliográfica

FLANAGAN, David. **JavaScript: O Guia Definitivo**. 6ª Ed. Porto Alegre: Bookman, 2013.

FREEMAN, Eric. **Use a cabeça!: programação JavaScript**. 1ª Ed. São Paulo: Alta Books, 2016.

Ir para exercício