



Laços de repetição em Javascript

Laços de repetição

Os laços de repetição são utilizados para executar um bloco de código várias vezes, de acordo com uma condição específica. Temos dois tipos de loop. O for e o while. Cada um com suas características!

O for, utilizamos quando sabemos a quantidade de vezes que determinada ação deve ser repetida, enquanto o while, utilizamos quando sabemos que determinada ação deve ser repetida, mas não sabemos a quantidade de vezes!

Laço de repetição while

O laço `while` executa um bloco de código enquanto uma condição especificada for verdadeira.



```
1  while (condição) {  
2      // código a ser repetido  
3  }
```

Podemos usar o `while` de duas formas: ou com condição, ou usando o `while (True)`. Vamos ver primeiro com a condição!

Laço de repetição while

```
1 //criamos uma variável contador que começa com o valor 1
2 let contador = 1;
3 //enquanto contador for menor ou igual a 5
4 while (contador <= 5) {
5     //mostre o valor de contador no console
6     console.log("Contagem:", contador);
7     //adicione mais um ao valor de contador
8     contador++;
9 }
10 //Resultado:
11 //1 2 3 4 5
```

Observe que, quando temos um while com condição, precisamos sempre ter algo que faça a condição se tornar falsa. Nesse caso, o valor de contador é alterado a cada loop!

Testando conhecimentos

Escreva um programa que imprima os números de 1 a 10 utilizando um laço while.

Laço de repetição while – while (true)


Observe os exemplos anteriores. Sabíamos quantas vezes a ação deveria se repetir, certo? De certa forma limitamos para que isso acontecesse determinada quantidade de vezes. Mas vamos pensar no seguinte exemplo:

Um jogo de advinhação. O usuário precisa advinhar um número, e a cada vez que ele errar, daremos uma dica: O número secreto é maior do que o que você sugeriu, ou menor.

Não temos como saber quantas vezes o usuário vai errar, até acertar, certo?

Laço de repetição while – while (true)

São em situações como essa que utilizamos o while (true). O laço while (true) cria um loop infinito, pois a condição sempre será verdadeira.



```
1  while(true){  
2      //código a ser executado  
3  }
```

Mas se não tem condição, como saímos desse laço?

Break e Continue

O `break` e o `continue` são palavras-chave em JavaScript que podem ser utilizadas dentro de laços de repetição, como o `while`, para controlar o fluxo de execução do código.

A palavra-chave `break` é utilizada para interromper a execução de um loop imediatamente, independentemente da condição de controle do loop. Quando o JavaScript encontra uma instrução `break`, ele sai do loop e continua a execução do código fora do loop.

Break e Continue

Quando utilizado com um loop `while (true)`, o `break` é comumente usado para sair de um loop infinito quando uma condição específica é atendida.

```
//começamos nosso loop, com while true
while (true) {
  //pedimos para o usuário digitar sair quando quiser para o
  //loop
  let entrada = prompt("Digite 'sair' para encerrar o loop:");
  //Se a palavra que o usuário digitar for 'sair'
  if (entrada === "sair") {
    break; // Termina o loop
  }
  //se o usuário não digitar 'sair',
  //a condição no if não é verdadeira,
  //logo ele não executa o bloco de código ali dentro
  //e o loop continua, até bater no break, ou seja
  //até a condição do if (não do laço) ser verdadeira
  console.log("Você digitou:", entrada);
}
```

Break e Continue

A palavra-chave `continue` é utilizada para interromper a iteração atual de um loop e avançar para a próxima iteração, ignorando o restante do código dentro do bloco de repetição.

Quando utilizado com um loop `while (true)`, o `continue` pode ser útil para ignorar certas iterações do loop com base em alguma condição.

```
//criamos uma variável 0
let numero = 0;
//criamos nosso while true
while (true) {
    //aumentamos o valor da variável numero em 1
    numero++;
    //se o valor guardado em numero for par
    if (numero % 2 === 0) {
        continue; // Pula para a próxima iteração.
        //ou seja, toda vez que esse if for verdadeiro,
        //quando bater no continue, ele vai ignorar todo o código
        //abaixo daqui, e rodar o laço novamente
    }
    console.log("Número ímpar:", numero);
    if (numero >= 10) {
        break; // Termina o loop quando o número atingir 10
    }
}

// Número ímpar: 1
// Número ímpar: 3
// Número ímpar: 5
// Número ímpar: 7
// Número ímpar: 9
// Número ímpar: 11
```

Break e Continue

Vale ressaltar que tanto o break quanto o continue também funcionam para o laço de repetição for.

Testando conhecimentos

Questão sobre adivinhação de número: Escreva um programa que gera um número aleatório entre 1 e 100. O usuário deve tentar adivinhar o número. Se o palpite estiver correto, exiba uma mensagem de sucesso e encerre o programa. Se o palpite estiver errado, informe ao usuário se o número é maior ou menor e permita que ele faça outra tentativa. Use **while (true)**, **break** e **continue**.

Questão sobre escolha de opção: Escreva um programa que apresenta um menu de opções ao usuário. O programa permite que o usuário escolha uma opção (1, 2 ou 3). Se o usuário escolher uma opção inválida, informe-o e permita que ele tente novamente. Se o usuário escolher uma opção válida, exiba uma mensagem de sucesso e encerre o programa. Use **while (true)**, **break** e **continue**.

Laço de repetição for

Como vimos anteriormente, o for também serve para repetição de ações no código, mas agora, controlando a quantidade de vezes na qual isso pode se repetir!

```
for (inicialização; condição; incremento) {  
    // código a ser repetido  
}
```


Laço de repetição for

Como vimos anteriormente, o for também serve para repetição de ações no código, mas agora, controlando a quantidade de vezes na qual isso pode se repetir!

```
for (let i = 1; i <= 5; i++) {  
  console.log("Número:", i);  
}
```

```
//resultado: Número: 1,  
// Número: 2,  
// Número: 3,  
// Número: 4,  
// Número: 5,
```

**let i = 1 é nosso início;
i<=5 é a nossa condição;
i++ é o nosso incremento.**

**Começaremos do numero 1 e vamos
testar, esse numero é menor que
5? Se sim, vai executar o código
e em seguida, adicionar mais 1 ao
valor de i;**

Testando conhecimentos

Escreva um programa que imprima os números pares de 1 a 10 utilizando um laço for.

Questão sobre soma de números pares: Escreva um programa que solicite ao usuário que insira uma sequência de números inteiros positivos. O programa deve calcular e exibir a soma apenas dos números pares. Utilize um loop **for**, **continue** para pular a adição dos números ímpares e **break** para encerrar o loop quando o usuário inserir um número negativo.

Laço de repetição for of

No JavaScript, a iteração sobre os caracteres de uma string é uma tarefa comum. O loop `for...of` é uma forma conveniente e legível de realizar essa iteração, permitindo acessar cada caractere individualmente.

```
const palavra = "Exemplo";  
//letra é uma variável criada por nós, para poder chamar cada  
//uma das iterações da palavra  
for (let letra of palavra) {  
    console.log(letra);  
}  
  
//E  
//x  
//e  
//m  
//p  
//l  
//o
```


Obtendo o tamanho da string

Além da iteração sobre os caracteres, muitas vezes precisamos saber o tamanho total de uma string. Para isso, podemos utilizar a propriedade `length` de uma string.

```
const palavra = "JavaScript";  
const tamanho = palavra.length;  
  
console.log("A palavra tem", tamanho, "caracteres.");  
//A palavra tem 10 caracteres.
```

Testando conhecimentos

Questão sobre contagem de vogais: Escreva um programa que solicite ao usuário que insira uma palavra e conte quantas vogais (a, e, i, o, u) estão presentes nela. Utilize um loop `for...of` para percorrer os caracteres da palavra e verifique se cada caractere é uma vogal.

Questão sobre validação de senha: Escreva um programa que solicite ao usuário que insira uma senha com pelo menos 6 caracteres, contendo pelo menos uma letra maiúscula e um número. Utilize um loop `for...of` para percorrer os caracteres da senha e verifique se ela atende aos critérios especificados.

Testando conhecimentos

Questão sobre tabuada com for e while: Escreva um programa que solicita ao usuário que insira um número e, em seguida, imprima a tabuada desse número até o 10. Utilize tanto um loop for quanto um loop while para calcular a tabuada.

Questão sobre busca de palavras em uma frase com for e while: Escreva um programa que solicite ao usuário que insira uma frase e uma palavra específica para buscar. Utilize tanto um loop for quanto um loop while para percorrer a frase e contar quantas vezes a palavra aparece.

Segue lá nas redes sociais

No instagram, tem um post novo todo dia, premiações e desafios, além de lives que você não vai encontrar em mais nenhum lugar!

Já no LinkedIn, tem mais conteúdo voltado para a prática, experiências incríveis e muito mais!

