
IN
INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

IN AULA 03 | JAVASCRIPT

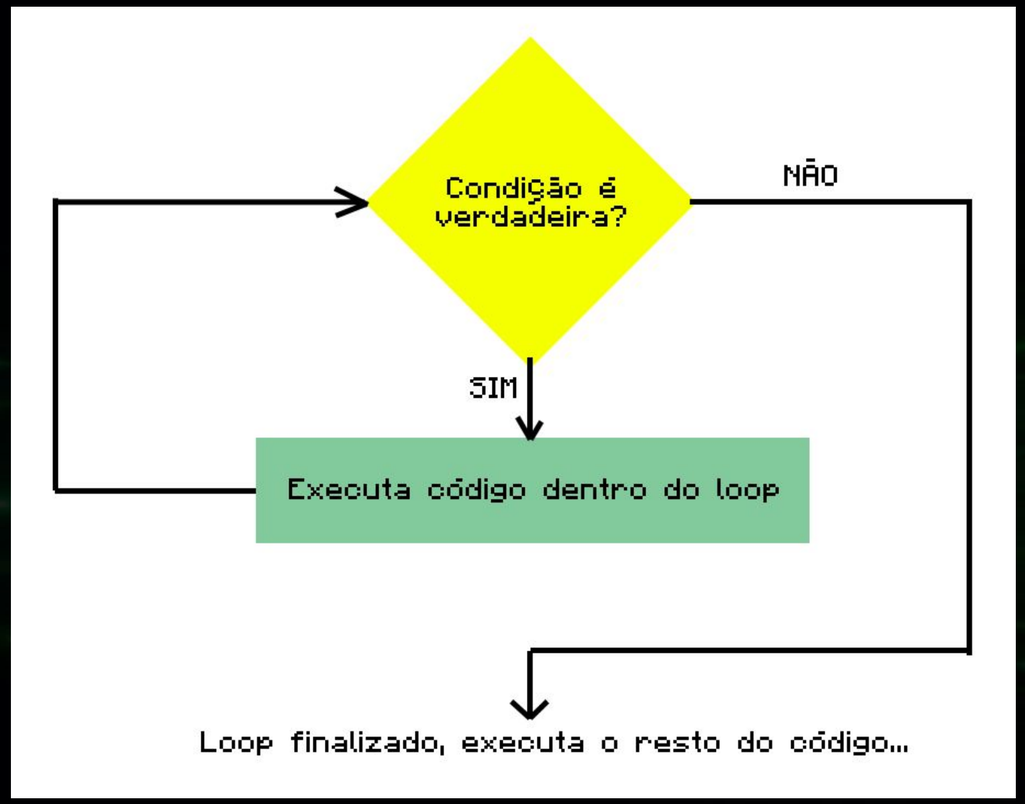
JS - LAÇOS DE REPETIÇÃO I

01 Estrutura de repetição

02 Laço de repetição While

03 Exercício

JS - LAÇOS DE REPETIÇÃO I



Loopings

Um Laço de Repetição, ou loop, é uma estrutura em programação que repete uma sequência de instruções até que uma condição específica seja atendida.

É um recurso que a linguagem de programação fornece para executar as mesmas instruções uma determinada quantidade de vezes.

O JavaScript nos fornece diferentes estruturas de repetição, nesta aula vamos abordar sobre a a estrutura **WHILE**.

JS - LAÇOS DE REPETIÇÃO I

While

O laço de repetição While (Enquanto) segue uma lógica bem simples: Esse laço executará determinada instrução **enquanto** determinada **condição** for verdadeira, a partir do momento em que o teste retornar false, o laço de repetição não será mais executado.

O laço verifica a condição, se for verdadeira, ele executa as instruções que estão dentro das chaves, após isso, ele verifica a condição novamente

O while possui certa semelhança com a instrução if, já que ambos fazem algo baseado em uma condição, a diferença é que o while pode executar aquele código mais de uma vez.

É importante freezar que caso o primeiro teste do laço while retorne false, o bloco de código dentro do while **não** será executado nenhuma vez

```
1  while (condicao) {
2      /*
3      Código que será executado
4      ENQUANTO a condição for verdadeira
5      */
6  }
```

JS - LAÇOS DE REPETIÇÃO I

Exemplo

```
1  let contador = 0 // Variável inicializada com o valor 0
2
3  while (contador < 10) {
4      // O laço se repetirá enquanto o contador for menor do que 10
5      console.log(contador)
6      contador++ // A variável contador está sendo atualizada a cada laço
7  }
8
```

JS - LAÇOS DE REPETIÇÃO I

Looping infinito

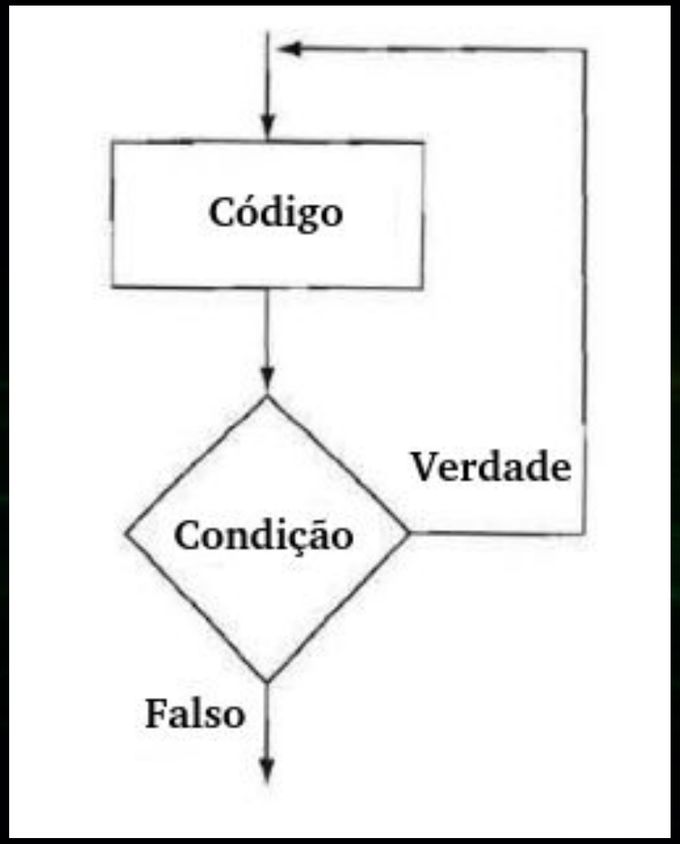
```
let contador = 0

while (contador < 10) {
    console.log(contador)
}

/*
Perceba que nesse programa a variável contador
não é atualizada em nenhum momento, ou seja
ela terá o valor 0 para sempre, portanto,
o while se repetirá pra sempre!
*/
```

Quando estamos trabalhando com laços de repetição devemos nos atentar a não programar um looping infinito!

Um looping infinito é um laço de repetição que se repete para sempre.

Ele consumirá todo o recurso da máquina e irá travar a máquina.

Ou então sempre repetirá as mesmas intruções e nunca conseguimos sair dessas instruções.



JS - LAÇOS DE REPETIÇÃO I

Do While

Além da estrutura While, temos também o **Do While.**

Sabemos que caso a condição retorne false no primeiro teste, as instruções dentro do while não irão ser executadas.

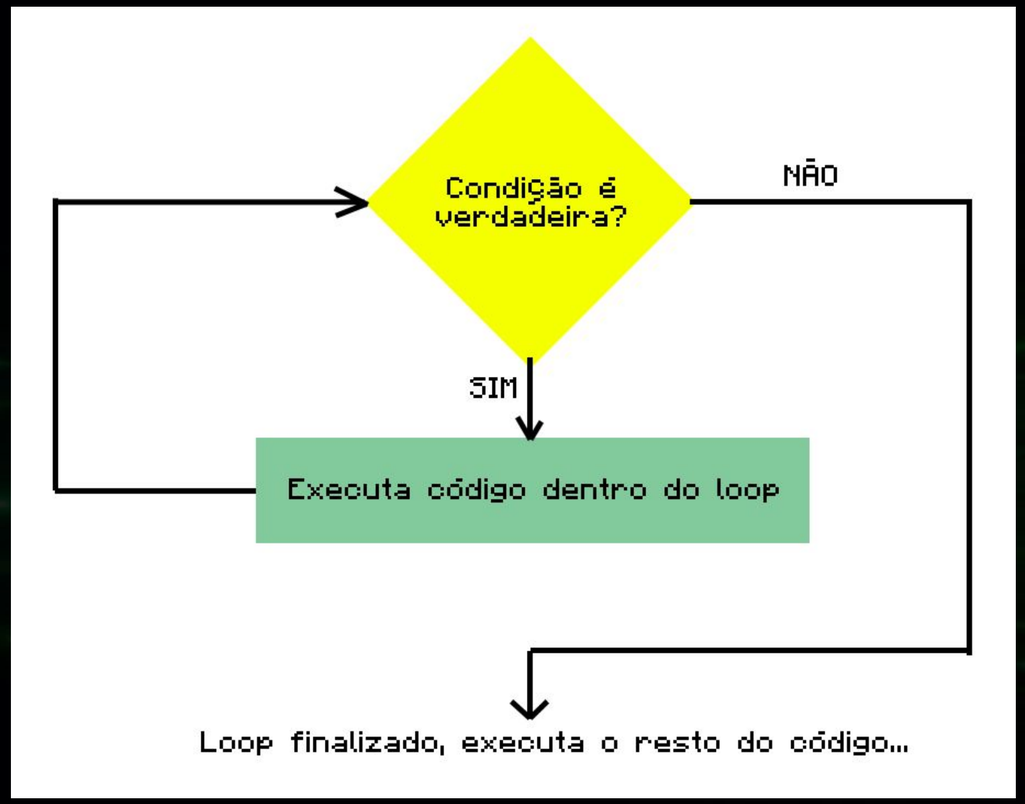
A estrutura **Do While** executa pelo menos uma vez as instruções, mesmo que a condição seja falsa, e após essa primeira execução, o laço trabalha como um laço While normalmente.

JS - LAÇOS DE REPETIÇÃO I

Exemplo

```
1  let contador = 1
2
3  do {
4      console.log(`Essa instrução foi executada ${contador} vez(es)`)
5  } while(contador < 0)
6  /*
7      Perceba que essa condição irá retornar false
8      já que 1 não é menor do que 0.
9      Entretanto, o console.log é executado 1 vez
10 */
```

JS - LAÇOS DE REPETIÇÃO I



Instruções break e continue

Conforme vamos incrementando nossos programas, os laços de repetição vão ficando mais complexos também, diante desse cenário, é importante que tenhamos mais controle do fluxo do programa, o JavaScript nos dá esse controle através das instruções **Break** e **Continue**.

O Break (quebrar) irá quebrar a estrutura de repetição, irá sair dela.

O Continue irá apenas quebrar aquele laço e partir para o próximo.

Os próximos slides detalharão sobre cada im

JS - LAÇOS DE REPETIÇÃO I

```
1  let contador = 0
2
3  while (contador < 10) {
4      console.log(contador)
5      if(contador === 5) {
6          // Quando o contador for igual a 5, o laço será quebrado
7          break
8      }
9      contador++
10 }
11 console.log(`O laço foi quebrado e contador parou com o valor ${contador} )
12
```

```
0
1
2
3
4
5
O laço foi quebrado e contador parou com o valor 5
```

Break

O comando break é usado para sair ou terminar imediatamente o loop.

Ele permite interromper prematuramente a execução do loop e continuar com o código após o loop.

JS - LAÇOS DE REPETIÇÃO I

```
1  let contador = 0
2
3  while (contador < 10) {
4      contador++
5      if(contador === 5) {
6          // Quando o contador for igual a 5, o laço atual será quebrado
7          continue
8      }
9      console.log(contador)
10 }
11 console.log(`O laço finalizou e o contador parou com o valor ${contador}`)
12 // Perceba que não houve o print do valor 5
```

```
1
2
3
4
6
7
8
9
10
O laço finalizou e o contador parou com o valor 10
```

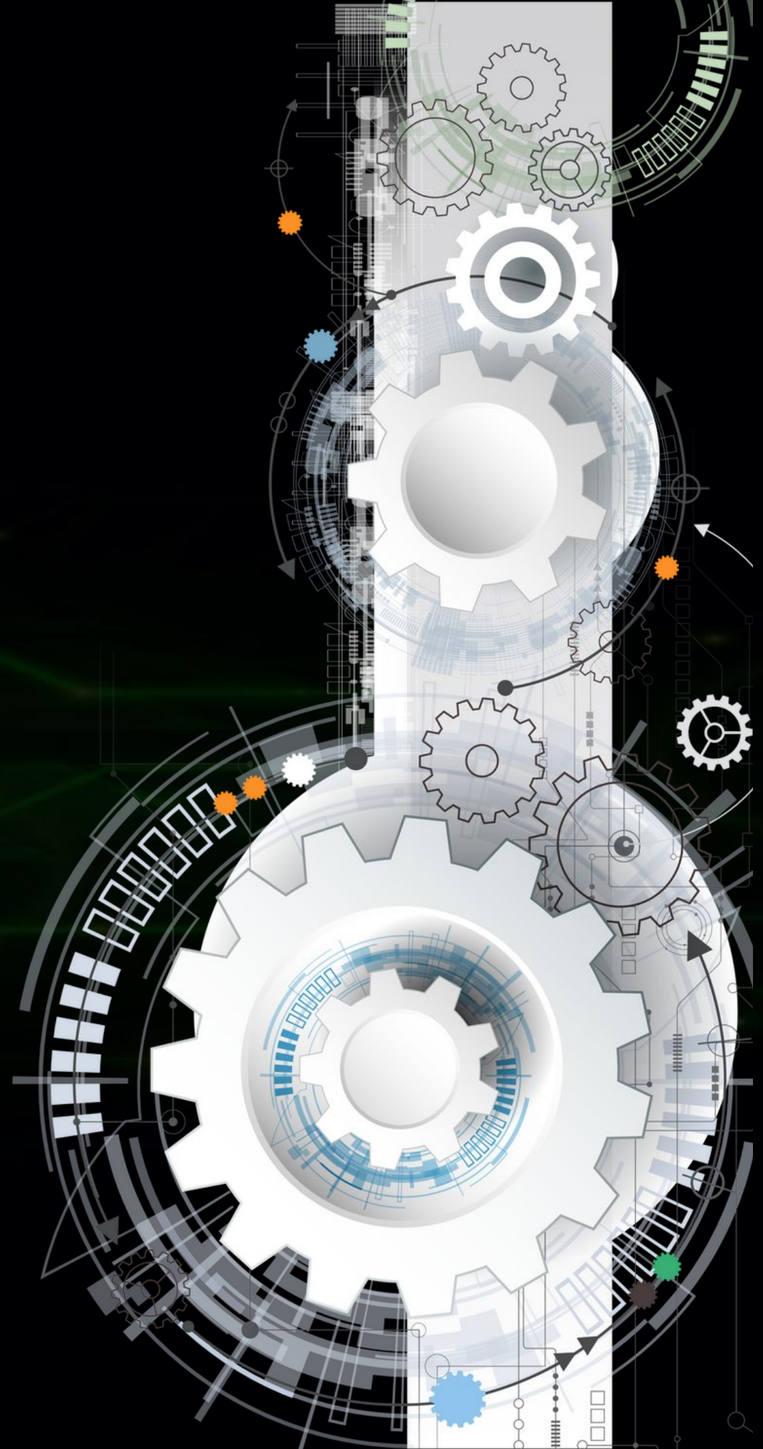
Continue

A intrução **continue** permite quebrar o laço atual e pular para o próximo laço.

Diferente do break, que sai da estrutura, o continue quebra apenas o laço atual, sem sair da estrutura de repetição, e vai para o próximo laço

IN AULA 03 | JAVASCRIPT

JS - LAÇOS DE REPETIÇÃO I



Mão no código

1) Imprima na tela os número de 1 a 10

2) Calcule a soma dos números de 1 a 50

3) Encontre o fatorial de um determinado número

4) Faça um programa que peça uma nota, entre zero e dez. Mostre uma mensagem caso o valor seja inválido e continue pedindo até que o usuário informe um valor válido.

5) Faça um programa que leia um nome de usuário e a sua senha e não aceite a senha igual ao nome do usuário, mostrando uma mensagem de erro e voltando a pedir as informações.

6) Faça um programa que calcule o mostre a média aritmética de N notas

7) Faça um programa que peça para 5 pessoas a sua idade, ao final o programa devera verificar se a média de idade da turma varia entre 0 e 25, 26 e 60 e maior que 60; e então, dizer se a turma é jovem, adulta ou idosa, conforme a média calculada.

IN PARABÉNS! VOCÊ TERMINOU A AULA 03 DO MÓDULO DE JAVASCRIPT