

# Laços

# O que vamos ver hoje?

- O que são Laços (Loops)
- Laços no Javascript:
  - **while**
  - **for**
  - **for... of**

# Laços

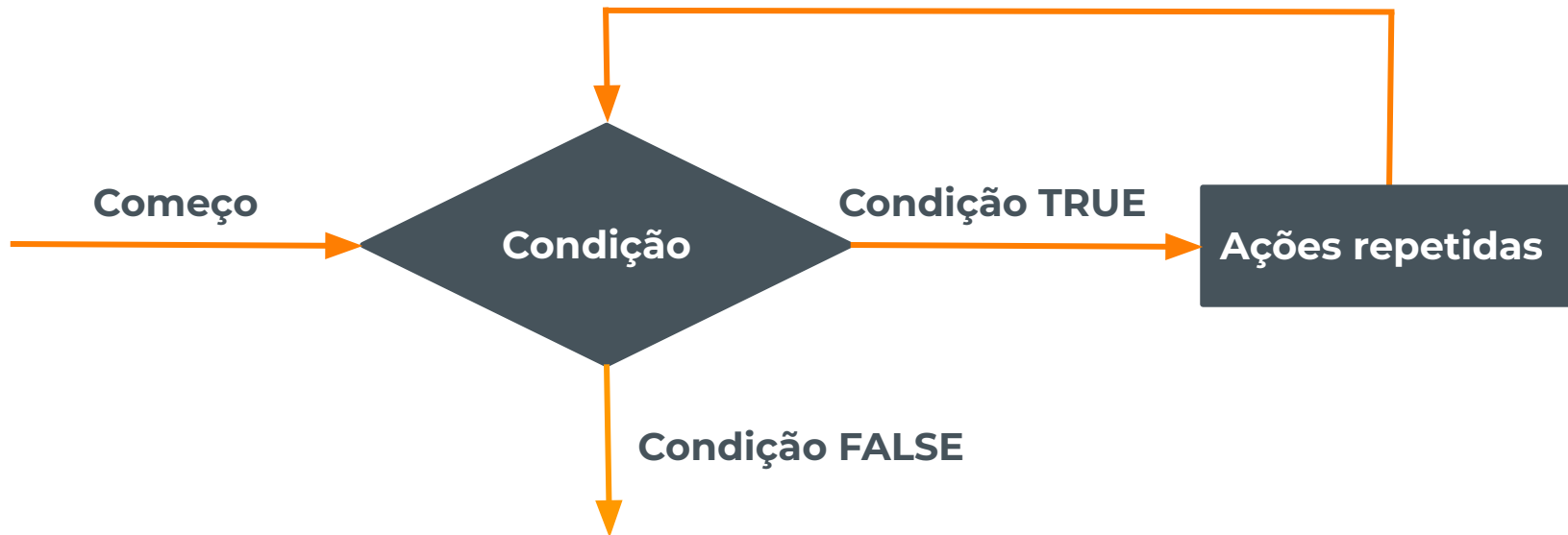
- **Laços** são estruturas de programação que permitem representar **eventos que se repetem**
  - **Aniversário:** todo ano fazemos aniversário
  - **Corrida de bike em um circuito:** os ciclistas percorrem inúmeras vezes o mesmo circuito
  - **Comer:** pegar a comida no garfo; levar até a boca;

# Laços

- Elementos de um laço
  - Deve ter um **começo**;
  - Uma **condição de continuação**;
  - Um **conjunto de ações** para ser repetido;
  - Um **incremento**.

# Laços

- Elementos de um laço



# Laços

- **Exemplo  $\Rightarrow$  Corrida**
  - **Começo:** estouro do alarme de início da corrida
  - **Condição de Continuação:** enquanto não completar X voltas
  - **Ação:** ciclistas pedalam
  - **Incremento:** aumentar uma volta a cada vez que os ciclistas completarem o circuito

# Laços

- **Exemplo** ⇒ **Lista de Transmissão do Zap**
  - **Começo**: primeira pessoa da lista
  - **Condição de Continuação**: enquanto não chegar na última pessoa da lista
  - **Ação**: mandar mensagem
  - **Incremento**: passar para a próxima pessoa da lista de transmissão e mandar a mensagem

# Laços Infinitos



# Loops Infinitos

- **Loop infinito** é um loop que **nunca** acaba.  
Normalmente isto acontece devido a algum **erro** de lógica de programação
- Ele pode acontecer quando:
  - **Esquecemos** de colocar o **incremento** da variável
  - As **condições** de continuação **não** fazem muito **sentido**

# Loops Infinitos

- **O que fazer quando isso acontece:**
  1. Ctrl + C para encerrar a execução do programa
  2. Desligar o computador
  3. Rezar
  4. E se nada der certo...

# Loops Infinitos



# while (enquanto)

# while

- **while** ("enquanto") é a estrutura mais básica de criação de loops

```
while(condicao) {  
    // ENQUANTO a condicao for verdadeira  
    // as linhas de código dentro deste bloco  
    // serão executadas  
  
    // assim que a condicao ficar falsa  
    // o LOOP/Laço vai parar  
}
```

# while

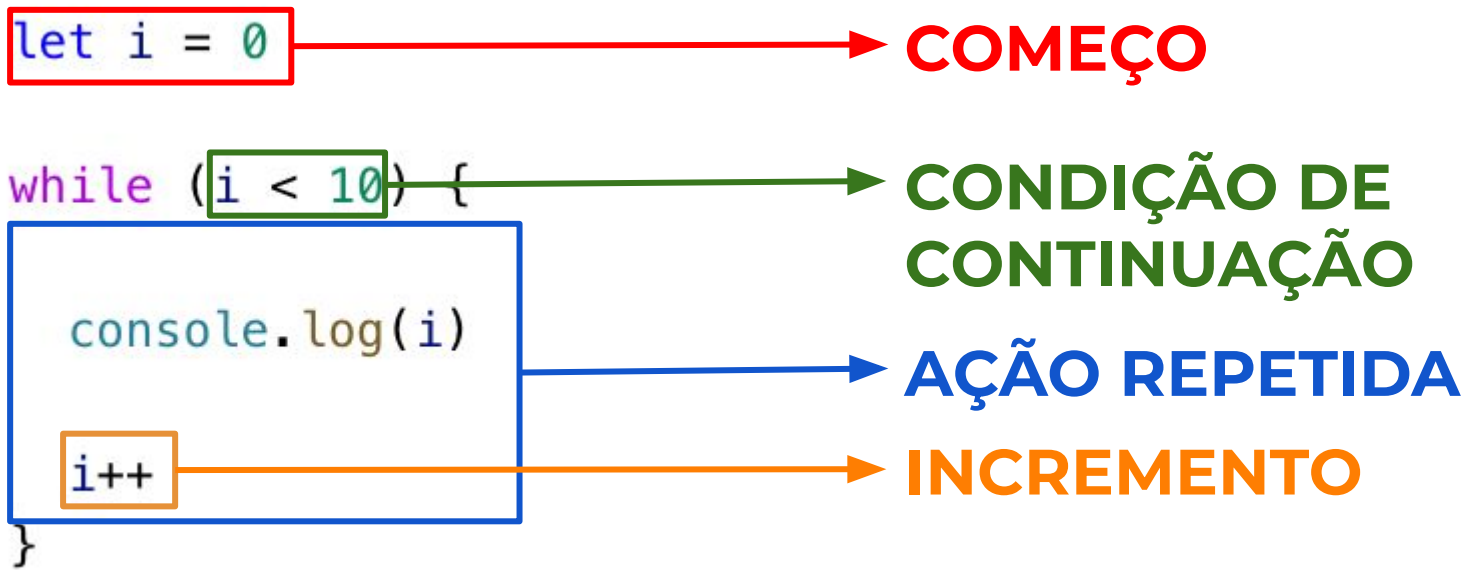
- Exemplo 1 - Imprimindo alguns números

```
let i = 0

while (i < 10) {
    console.log(i)
    i++
}
```

# while

- Exemplo 1 - Imprimindo alguns números



# while

- Exemplo 1 - Imprimindo alguns números

```
let i = 0
```

```
while (i < 10) {
```

```
  console.log(i)
```

```
  i++
```

```
}
```

só podemos colocar  
valores verdadeiros  
aqui

todos valores são  
true exceto **false**, **0**,  
**-0**, **""**, **null**,  
**undefined** e **NaN**



# while

- Exemplo 2: "Vou comer até 100 coxinhas"

```
let estomago = 0;

while (estomago < 100) {
  console.log("Quero comer mais coxinhas");
  estomago = estomago + 10;
}
```

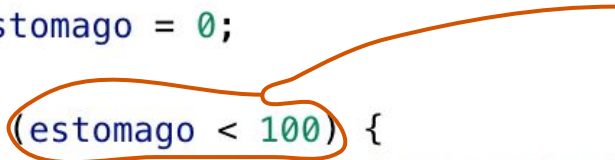
Vamos ver na prática! 

# while

- Exemplo 2: "Vou comer até 100 coxinhas"

```
let estomago = 0;  
  
while (estomago < 100) {  
    console.log("Quero comer mais coxinhas");  
    estomago = estomago + 10;  
}
```

0 < 100 = false  
101 < 100 = true



Vamos ver na prática!



# Exercício 1

- No nosso sistema, o usuário será solicitado para inserir vários números, um após o outro
- Quando ele digitar o número '0', devemos parar de solicitar novos inputs e imprimir no console a soma de todos os números por ele indicados
- Ex.: Vamos supor que ele coloque: 10, 3, 50, 7, 0.  
O resultado deve ser: 70

# Fixação

- Laços ou Loops são estruturas para representar eventos que se repetem
- while: estrutura de repetição mais básica

**for (para)**

# for

- São bem comuns os laços em que temos a **condição de continuação** atrelada a um **número** que é incrementado.
- O laço **for** é uma maneira que permite **simplificar** a escrita de laços que tenham este comportamento

# for

- O laço **for** é uma maneira que permite **simplificar** a escrita de laços que tenham este comportamento

while	for
<pre>let i = 0  while (i &lt; 10) {    console.log(i)    i++ }</pre>	<pre>for(let i = 0;    i &lt; 10;    i++) {   console.log(i) }</pre>

Vamos ver na prática!



# for

Diagram illustrating the components of a `for` loop:

```
for(let i = 0; i < 10; i++) {  
  console.log(i)  
}
```

The components are labeled as follows:

- COMEÇO** (Start): `let i = 0;`
- CONDIÇÃO DE CONTINUAÇÃO** (Continuation Condition): `i < 10;`
- INCREMENTO** (Increment): `i++`
- AÇÃO REPETIDA** (Repeated Action): `console.log(i)`

Vamos ver na prática! 



# for

- Uma das principais utilidades deste **tipo de estrutura** é para **PERCORRERMOS os valores contidos em um array**. Veja o código abaixo

```
const numeros = [14, 67, 89, 15, 23]

for(let i = 0; i < 5; i++) {
  const elemento = numeros[i]
  console.log(elemento)
}
```

Vamos ver na prática! 

## Exercício 2

- Escreva um código que declare um array com números e imprima na tela qual o maior número dentro dele
- Ex: Para o array [11, 15, 18, 14, 12, 13], a mensagem deve ser: "O maior número é 18"

# Fixação

- O laço **for** é uma maneira que permite **simplificar** a escrita de laços que tenham este comportamento

Programa

**3000 TALENTOS TI**

**for... of... (para... cada...)**

# for... of...

- Uma forma de simplificar a leitura dos elementos do array é utilizando o loop **for...of...**
- O loop for...of **percorre arrays e objetos**, alocando o valor de cada posição do array em uma variável, permitindo executar alguma ação para cada valor distinto.

# for... of...


```
const numeros = [14, 67, 89, 15, 23]

for (let numero of numeros){
  console.log(numero)
}
```

# for... of...

```
const numeros = [14, 67, 89, 15, 23]

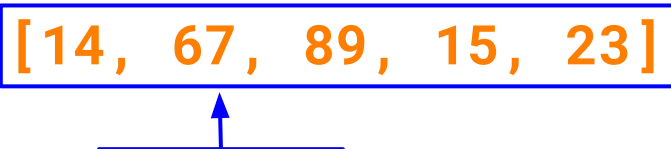
for (let numero of numeros){
  console.log(numero)
}
```



# for... of...

```
const numeros = [14, 67, 89, 15, 23]

for (let numero of numeros) {
  console.log(numero)
}
```





# for... of...

Quantidade de repetições:  
tamanho do array (5)



```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero)  
}
```

# for... of...

## Loop 1

```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero) //14  
}
```

# for... of...

## Loop 2

```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero) //67  
}
```

# for... of...

## Loop 3

```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero) //89  
}
```

# for... of...

## Loop 4

```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero) //15  
}
```

# for... of...

## Loop 5

```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero) //23  
}
```

# for... of...

```
const numeros = [14, 67, 89, 15, 23]
```

```
for (let numero of numeros){  
  console.log(numero)  
}
```

**console.log é apenas um exemplo de ação que podemos estipular para cada valor do array. Outro exemplo é multiplicar cada número por 2.**

**Vamos ver na prática!**



## Exercício 3

- Considere que você tem um array com várias palavras. Nossa tarefa é criar uma função para imprimi-las em uma só mensagem colocando um espaço entre elas.
- Ex: Para este array ["Oi", "sumido", "tudo", "bem?", "Saudades", "kk"], dar a mensagem "Oi sumido tudo bem? Saudades kk"



# Resumo

# Resumo

- **Loops** ou **Laços** são estruturas que permitem representar repetições das mesmas ações
- Um loop deve conter:
  - **Começo**
  - Uma **condição** de **continuação**
  - A **ação** que deve ser repetida
  - Um **incremento** relacionado à condição

# Resumo

- **while**: Realiza uma ação até que a **condição** dentro dos seus parênteses se tornar **falsa**
- **for**: Usado para realizar loops cujas condições estejam diretamente relacionadas a **números** e a um **incremento fixo**
- **for... of...** : Permite **percorrer** os elementos de um **array** de uma forma bem **mais simples de se ler**

# Resumo



- Entrei num loop infinito, o que eu faço?
  - Tente encerrar o código que está rodando:
    - Fechando a aba ou o navegador
    - Usando o gerenciador de tarefas
    - Desligando o PC

# Dúvidas?



Programa  
**3000 TALENTOS TI**  
Obrigado(a)!