🎖️

Instrução for

> Um loop for serve para que possamos percorrer itens, em sua grande maioria o for é uma instrução usada dentro de arrays, mas primeiro vamos ver o básico sobre essa instrução.

O laço de repetição ou for, possui algumas formas (for, for in e for of) e sua sintaxe principal é:

```
for (variavel = 1; condicao; incrementacao) {
  logica em si
}
```

Ele é responsável por criar um loop e executar determinada ação até atender uma certa condição.

Um breve exemplo pode ser:

```
for (let i = 1; i <= 10; i++) {
  console.log(i);
};

/*
Retorna:
1
2
3
...
10
*/
```

Por padrão, a variável é let e seu nome é 'i', para funcionar como um contador em si.

Enquanto 'i' for menor ou igual a 10 ele irá executar a lógica.

Por fim, incrementa 1 a cada vez que realiza com sucesso.

No nosso bloco de execução há apenas um comando simples para aparecer no console o resultado de cada etapa, porém podemos colocar qualquer funcionalidade em si.

Basicamente, você irá definir uma variável e atribuir seu valor inicial, logo em seguida define até quando será executado ("até chegar no número 5" → i <= 5) e por fim incrementa ou decrementa na sua variável inicial.

Como já sabemos, o Javascript é uma linguagem que possui cadeias de dados denominados de Arrays e/ou Objetos, para este próximo exemplo iremos utilizar uma mescla de ambos, portanto, utilizaremos um Array de Objetos:

```
const pessoas = [
  {
    Nome: 'Faustão',
    Idade: 71,
    Profissao: 'Apresentador'
  },
  {
    Nome: 'Elon Musk',
    Idade: 50,
    Profissao: 'Empreendedor'
  },
  {
    Nome: 'Ronaldinho',
    Idade: 41,
    Profissao: 'Futebolista'
  }
];
```

Com o Array criado, já podemos utilizá-lo:

```
for (let i = 0; i < pessoas.length; i++) {
  console.log(pessoas[i].Profissao);
```

```
};

/*
Retorna:

Apresentador
Empreendedor
Futebolista
*/
```

Nesse exemplo, acontece o seguinte:

- 1º : declaramos a variável i e atribuímos 0 como valor;
- 2º : depois falamos que, enquanto i é menor que `pessoas.length` (comprimento do array em si) executará o bloco com a lógica;
- 3º : Por fim, somamos 1 à variável i.
- Já na lógica, colocamos a Profissão das pessoas do Array no console. O [`i`] é a posição do array a ser buscada, por exemplo:

```
// Primeira execução:
for (let i = 0; i < pessoas.length; i++) {
  console.log(pessoas[i].Profissao);
};
/* i = 0, então:

console.log(pessoas[0].Profissao);
// Retorna: Apresentador
*/

// Segunda execução:
for (let i = 0; i < pessoas.length; i++) {
  console.log(pessoas[i].Profissao);
};
/* i = 1, então:

console.log(pessoas[1].Profissao);
// Retorna: Empreendedor
*/

// Terceira execução:
for (let i = 0; i < pessoas.length; i++) {
  console.log(pessoas[i].Profissao);
};
/* i = 2, então:

console.log(pessoas[2].Profissao);
// Retorna: Futebolista

// Essa é a última execução, pois pessoas.length = 3
// logo, i não é menor e sim igual
```

For in →

O For in iterage sobre todas as propriedades enumeráveis do objeto em questão. Ou seja, se usa quando estamos em busca de uma posição específica dentro do Array ou Objeto. A posição retornada é do tipo string:

```
const idades = [
  '11',
  '22',
  '33'
];

// For in:
for (i in idades) {
  console.log(idades[i]);
};
/* Retorna:
  11
  22
  33
*/

// O mesmo que:
for (let i = 0; i < idades.length; i++) {
  console.log(idades[i]);
};
```

```
/* Retorna:
  11
  22
  33
*/
```

Note que não há diferença no consumo das propriedades, conseguimos rodar o Array de itens da mesma forma tanto com um tipo de laço quanto com outro.

For of →

É a maneira mais simples de consumirmos um Array, basicamente definimos uma variável, e para cada item dentro deste Array traremos sua estrutura por completo.

```
const idades = [
  '11',
  '22',
  '33'
];

// For of:
for (idade of idades) {
  console.log(idade);
};

// O mesmo que:
for (let i = 0; i < idades.length; i++) {
  console.log(idades[i]);
  };
```

Podemos também iterar uma string:

```
let string = 'Clóvis';

for (valor of string) {
  console.log(valor);
};

/*
  C
  l
  ó
  v
  i
  s
*/
```

Assim como iteramos um array, podemos iterar um objeto. Possuímos 2 métodos principais: usando o for-in e usando o for-of. No primeiro caso podemos utilizar o seguinte exemplo:

```
var pessoa = {
  nome: "Clóvis",
  idade: 30,
  vendedor: true
};

for (var informacao in pessoa) {
  if (pessoa.hasOwnProperty(informacao)) {
    console.log(informacao + ": " + pessoa[informacao]);
  }
}

/* Retorna:
nome: Clóvis
idade: 30
vendedor: true
*/
```

Não entendeu muito bem? Vamos ver o passo a passo agora:

- 1 → Criamos um objeto chamado `pessoa`, inserimos algumas propriedades e seus valores.
- 2 → Criamos um for in para percorrer todas suas propriedades;

- 3 → Usamos um if para verificar o retorno do método hasOwnProperty();
- 4 → Esse método nos retorna um valor booleano, sendo indicado se o objeto possui a propriedade especificada como uma propriedade definida no próprio objeto em questão;
- 5 → Usamos a variável `informacao` como contador para facilitar a compreensão, mas o nome mais comum para variáveis desse tipo é '`i`'.
- 6 → Caso tenha determinada propriedade, a mesma e seu respectivo valor aparecerão no console;

Como vimos, conseguimos utilizar o for-in para realizar a iteração de um objeto, mas esse não é o único modo. O segundo meio possível é utilizando o for-of:

```
var pessoa = {
  nome: "Clóvis",
  idade: 30,
  vendedor: true
};

for (let [propriedade, valor] of Object.entries(pessoa)) {
  console.log(`${propriedade}: ${valor}`);
}

/* Retorna:
nome: Clóvis
idade: 30
vendedor: true
*/
```

Nesse exemplo temos como resultado o mesmo que no modo anterior.

Explicação da segunda possibilidade:

- 1 → Criamos um objeto chamado `pessoa`, inserimos algumas propriedades e seus valores.
- 2 → Utilizamos o for-of, que para cada item dentro desse objeto traremos sua estrutura por completo.
- 3 → O método Object.entries() retorna uma matriz de [key, value], pares de propriedades com chave de string enumeráveis (no exemplo é [propriedade, valor] para facilitar sua compreensão).
- 4 → Após inserirmos o objeto a ser "visto" pelo Object.entries(), mostramos no console o resultado.

Bibliografias:

- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/for...in#veja_também;
- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/for...of#iterando_sobre_um_jsxref(array);
- https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Statements/for;
- https://stackoverflow.com/questions/684672/how-do-i-loop-through-or-enumerate-a-javascript-object;