

AULA 15 – VARIÁVEIS COMPOSTAS

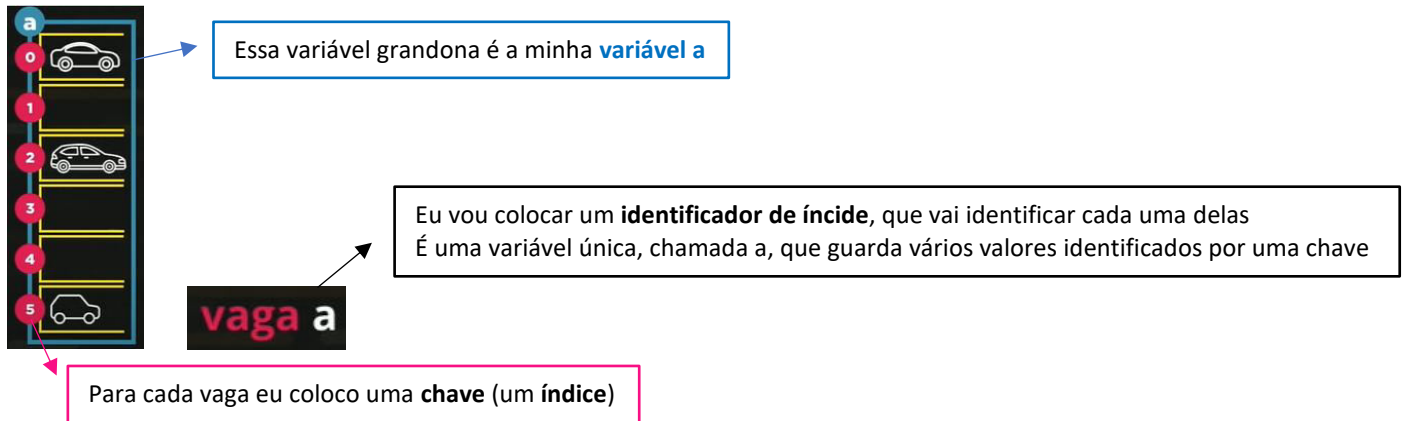
Lembrando:

→ **Variáveis simples** só conseguem armazenar **um valor** por vez

→ **Variáveis compostas** devem ser capazes de armazenar **vários valores** em uma mesma estrutura → ela não perde um valor quando outro é atribuído

Ex: var = 8; var = 7; var = 8 → todos esses valores são guardados

Exemplo do estacionamento:



→ Para fazer declarar isso em JavaScript é muito simples: é só colocar colchetes

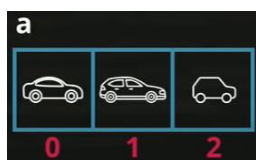
```
vaga a = []
```

→ Como eu posso declarar uma variável desse tipo e já colocar os carros nas vagas?

```
vaga a = [carro, carro, carro]
```

→ Nesse caso, eu declarei uma variável **a** sendo ela dividida em 3 partes → lembrando que nem sempre é 3 partes, é somente um exemplo

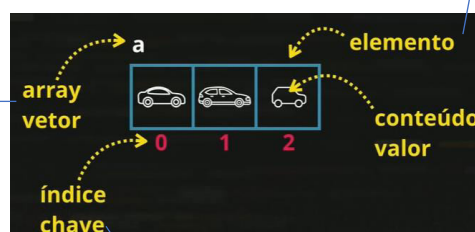
→ Quando eu declarei essa linha com 3 espaços entre os colchetes, automaticamente esses carros vão para essas vagas (que são espaços). Esses espaços recebem uma **identificação** (um número), que chamamos de **chave** ou **índice**



OBS: a primeira vaga sempre é a 0!

→ Nomeando isso:

Essa variável **a** que declaramos é uma variável do tipo composta ou **array (vetor)**



Um vetor é composto de **elementos**. Um elemento de um vetor é um par que agrupa o espaço da memória, o valor colocado dentro dele e o índice

Índice são esses números, que também podemos chamar de **chave**

→ Um array (ou um vetor ou uma variável composta) é uma variável que tem vários elementos. Cada elemento é composto por seu valor e por uma chave de identificação

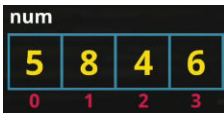
→ Como declarar em JavaScript:

```
let num = [ 5 , 8 , 4 ]
```



→ Para acrescentar valores sem perder os antigos:

```
let num = [ 5 , 8 , 4 ]  
num[3] = 6
```



Aqui eu estou dizendo para o JavaScript: "Coloque o valor 6 na posição 3". Antes a posição 3 não existia, mas o JavaScript percebe isso e cria automaticamente essa posição

OBS: essa é uma maneira automatizada do JavaScript.

→ Para acrescentar valores explícitos, como por exemplo na última posição:

```
num.push(7)
```



Com isso, o JavaScript vai criar mais um elemento e automaticamente vai decidir que o índice é 4 e vai colocar nele o valor 7

→ Para saber o comprimento do arrays:


```
num.length
```



OBS: length não é um método, mas sim um atributo!

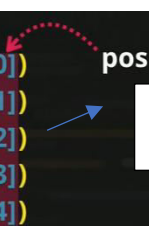
→ Para colocar o vetor em ordem crescente:

```
num.sort()
```



→ Para mostrar o vetor na tela sem a formatação padrão:

```
console.log(num[0])  
console.log(num[1])  
console.log(num[2])  
console.log(num[3])  
console.log(num[4])
```



Observa-se que a única diferença nos códigos são os índices!
Esses números marcados são as minhas chaves, no qual indicam a **posição** de cada vetor

Eu até posso mostrar assim, mas imagina um vetor maior como de tamanho 200 por exemplo...

Para isso:

```
for(let pos=0; pos<num.length; pos++) {
  console.log(num[pos])
}
```

→ Tem uma outra maneira também, que é até mais simples: **for in**

Ele é um for otimizado para variáveis compostas e objetos

Aqui vai meu índice

Aqui vou colocar minha variável composta

```
for(let pos in num) {
  console.log(num[pos])
}
```

Leia-se: "Para cada posição dentro de num, eu vou mostrar o

→ Para buscar o valor de um vetor:

```
num.indexOf(7)
```

num				
4	5	6	7	8
0	1	2	3	4

Ele vai buscar no vetor onde está o valor 7

Essa função vai me retornar a chave onde esse valor se encontra, que no caso é 3

Se por algum acaso eu mando ele buscar um **valor que não existe** no meu vetor, ele vai me **retornar -1**
-1 para o JavaScript significa que ele pesquisou dentro do vetor e não encontrou nenhuma ocorrência

Exemplos no Visual Studio:

```
let num = [6,3,4,7,8]
console.log(`Nosso vetor é o ${num}`)
```

Info: Start process (7:21:03 PM)
Nosso vetor é o 6,3,4,7,8
Info: End process (7:21:03 PM)

OBS: se eu declarar somente a variável, a resposta virá com os colchetes juntos!

```
let num = [6,3,4,7,8]
console.log(num)
```

Info: Start process (7:22:39 PM)
[6, 3, 4, 7, 8]
Info: End process (7:22:40 PM)

```
let num = [6,3,4,7,8]
console.log(`O vetor tem ${num.length} posições`)
```

Info: Start process (7:36:42 PM)
O vetor tem 5 posições
Info: End process (7:36:42 PM)

```
let num = [6,3,4,7,8]
console.log(`O primeiro valor do vetor é ${num[0]}`)
```

Info: Start process (7:38:28 PM)
O primeiro valor do vetor é 6
Info: End process (7:38:28 PM)

```
let num = [6,3,4,7,8]
num.push(1)
num.sort()
console.log(num)
```

Info: Start process (7:41:17 PM)
[1, 3, 4, 6, 7, 8]
Info: End process (7:41:18 PM)

```
let valor = [8,1,7,4,2,9]
for (let pos = 0; pos < valor.length; pos++) {
  console.log(`A posição ${pos} tem valor ${valor[pos]}`)
}
```

```
Info: Start process (8:16:33 PM)
A posição 0 tem valor 8
A posição 1 tem valor 1
A posição 2 tem valor 7
A posição 3 tem valor 4
A posição 4 tem valor 2
A posição 5 tem valor 9
Info: End process (8:16:33 PM)
```

```
let valor = [8,1,7,4,2,9]
for(let pos in valor) {
  console.log(`A posição de ${pos} tem valor ${valor[pos]}`)
}
```

```
Info: Start process (8:24:24 PM)
A posição de 0 tem valor 8
A posição de 1 tem valor 1
A posição de 2 tem valor 7
A posição de 3 tem valor 4
A posição de 4 tem valor 2
A posição de 5 tem valor 9
Info: End process (8:24:24 PM)
```