

Universidade Federal do Rio Grande do Norte - UFRN

Instituto Metrópole Digital – IMD
JavaScript: Arrays e Strings

Nelson Ion de Oliveira

Agenda

- Arrays
 - Strings
-

Vetores / Arrays

- Assim como em outras linguagens, é possível declarar vetores em JavaScript.
- Uma das formas de se declarar vetores é pela atribuição de colchetes ao nome da variável, por exemplo:

var nomes = [];

- Para atribuir valores, faça:

nomes[i] = “UFRN”;

nomes[i+1] = “IMD”;

Vetores / Arrays

- É possível criar vetores com valores já atribuídos:

`departamentos = ["DCA","DIMAp","CCHLA"];`

- Ou

`var departamentos = ["DCA","DIMAp","CCHLA"];`

Vetores / Arrays

- É possível ainda declarar array utilizando o operador **new**:

```
var variavel = new Array(n)
```

- Porém essa opção não é recomendada.
- A opção sem o **new** é mais rápida e recomendada.

Vetores / Arrays

- O primeiro índice de acesso do array é zero.

```
◀ var departamentos = ["DCA", "DIMAp", "CCHLA"];
▶ undefined
◀ x = departamentos[0]
▶ "DCA"
```

Vetores / Arrays

- Arrays em JavaScript podem ser utilizados como estruturas de dados Pilha e/ou Deque.
- Por esse motivo, é muito fácil inserir ou remover um novo elemento no final ou no inicio de um array.

```
> var nomes = ["IMD", "CIVT", "nPITI"]
> undefined
> nomes.push("DCA")
> 4
> nomes.push("DIMAp")
> 5
> nomes.pop()
> "DIMAp"
> nomes
> Array [ "IMD", "CIVT", "nPITI", "DCA" ]
```

Vetores / Arrays

- Utilize o método **push** para inserir no final.
- Utilize o método **pop** para remover no final.
- Utilize o método **unshift** para inserir no início.
- Utilize o método **shift** para remover no início.

Vetores / Arrays

- Array como uma pilha:

```
◀ var nomes = ["IMD", "CIVT", "nPITI"]
▶ undefined
◀ nomes.push("DCA")
▶ 4
◀ nomes.push("DIMAp")
▶ 5
◀ nomes.pop()
▶ "DIMAp"
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI", "DCA" ]
```

Vetores / Arrays

- Array como um deque:

```
nomes
Array [ "CIVT", "nPITI", "DCA" ]
nomes.unshift("IMD")
4
nomes.unshift("CCHLA")
5
nomes
Array [ "CCHLA", "IMD", "CIVT", "nPITI", "DCA" ]
nomes.shift()
"CCHLA"
nomes.pop
function pop()
nomes
Array [ "IMD", "CIVT", "nPITI", "DCA" ]
nomes.pop()
"DCA"
nomes
Array [ "IMD", "CIVT", "nPITI" ]
```

Vetores / Arrays

- Diferente de algumas linguagens em JavaScript os arrays são acessados **apenas** por índices numéricos.

```
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI" ]
◀ nomes[0]
▶ "IMD"
◀ nomes[1]
▶ "CIVT"
◀ nomes[2]
▶ "nPITI"
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI" ]
```

Vetores / Arrays

- Convertendo Arrays em Strings
- Ao utilizar o método **toString()** em um array, o retorno é uma String com todos os elementos do array separados por vírgulas.

```
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI" ]
◀ nomes.toString()
▶ "IMD,CIVT,nPITI"
```

Vetores / Arrays

- O método **join()** retorna também uma String, semelhante ao método **toString()**, porém no método **join()** é possível informar o separador dos elementos.

```
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI" ]
◀ nomes.toString()
▶ "IMD,CIVT,nPITI"
◀ nomes.join(" / ")
▶ "IMD / CIVT / nPITI"
```

Vetores / Arrays

- É possível ainda ordenar os elementos de um vetor em ordem crescente com o simples uso do método **sort()**.

```
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI" ]
◀ nomes.toString()
▶ "IMD,CIVT,nPITI"
◀ nomes.join(" / ")
▶ "IMD / CIVT / nPITI"
◀ nomes.sort()
▶ Array [ "CIVT", "IMD", "nPITI" ]
```

Vetores / Arrays

- Para ordenar os elementos do vetor em ordem decrescente, use o método **reverse()**.

```
◀ nomes
▶ Array [ "IMD", "CIVT", "nPITI" ]
◀ nomes.toString()
▶ "IMD,CIVT,nPITI"
◀ nomes.join(" / ")
▶ "IMD / CIVT / nPITI"
◀ nomes.sort()
▶ Array [ "CIVT", "IMD", "nPITI" ] [highlighted]
◀ nomes.reverse()
▶ Array [ "nPITI", "IMD", "CIVT" ]
◀ nomes
▶ Array [ "nPITI", "IMD", "CIVT" ]
```

Vetores / Arrays

- Por padrão, a função `sort()` classifica valores do tipo String.
- Isso funciona bem para Strings ("CIVT" vem antes de "nPITI").
- No entanto, se os números são classificados como String, então "2" é maior do que "10", porque "2" é maior do que "10".
- Você pode corrigir isso através de uma função de comparação, veja:

Vetores / Arrays

```
◀ var points = [40, 100, 1, 5, 25, 10]
▶ undefined
◀ points.sort(function(a, b){return a-b})
▶ Array [ 1, 5, 10, 25, 40, 100 ]
```

String

- Propriedade Length:
 - É uma “propriedade” das Strings, e retorna o comprimento da String armazenada na variável.

```
◀ var nome = "Instituto Metrópole Digital"  
▶ undefined  
◀ nome.length  
▶ 27
```

String

- Funções **toLowerCase()** e **toUpperCase()**
 - Retornam a String em caixa baixa e caixa alta respectivamente.

```
◀ var nome = "Instituto Metrópole Digital"
▶ undefined
◀ nome.length
▶ 27
◀ nome.toUpperCase()
▶ "INSTITUTO METRÓPOLE DIGITAL"
◀ nome
▶ "Instituto Metrópole Digital"
◀ nome.toLowerCase()
▶ "instituto metrópole digital"
◀ nome
▶ "Instituto Metrópole Digital"
```

String

- A função `split()` quebra a String com base no divisor indicado.
- Essa função retorna um array com as “partes” da String principal.

```
◀ var nome = "Instituto Metrópole Digital"
▶ undefined
◀ var nomes = nome.split(" ")
▶ undefined
◀ nomes
▶ Array [ "Instituto", "Metrópole", "Digital" ]
◀ var nomes2 = nome.split("i")
▶ undefined
◀ nomes2
▶ Array [ "Inst", "tuto Metrópole D", "g", "tal" ]
```

Dúvidas?



Referências

- **Textos e Exemplos retirados dos sites:**
 - www.w3schools.com/js/
 - www.codecademy.com
-
-