

# *Atribuição de um Objeto em JavaScript*

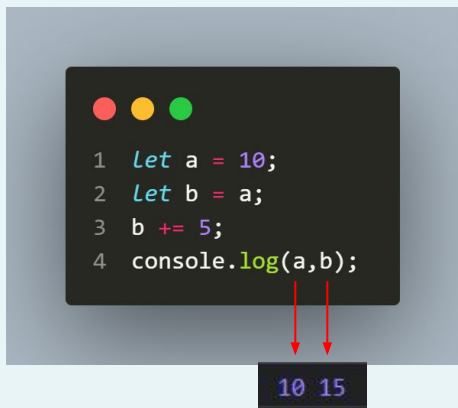
# Armazenamento de Tipos Primitivos em Memória

- As variáveis que são declaradas ou criadas ao longo do código, devem ser armazenadas nos espaços da memória principal do computador durante a execução do script pelo navegador, por exemplo
- Para fins didáticos, o exemplo abaixo simplifica como os valores de duas variáveis do tipo `number` são armazenados na memória principal

endereço de memória	espaço de memória	
679	10	← valor de uma variável <code>x</code>
680	7	← valor de uma variável <code>y</code>
681		

# Atribuição de um Tipo Primitivo em JavaScript

- A atribuição de um tipo de dado primitivo (number, string, boolean, por exemplo) ocorre como exemplo anterior
  - o que explica o resultado apresentado no exemplo abaixo



The image shows a code editor with a dark background and three colored circles (red, yellow, green) at the top left. The code is as follows:

```
1 let a = 10;  
2 let b = a;  
3 b += 5;  
4 console.log(a,b);
```

Below the code, there is a small black box containing the output '10 15'. Two red arrows point from the 'a' and 'b' arguments in the `console.log` statement to the '10' and '15' in the output box, respectively.

*o valor da variável **a** foi atribuído à variável **b**, porém ao incrementar o valor da variável **b**, não alterou o valor da variável **a***

# Armazenamento de Tipos Primitivos em Memória

- No exemplo anterior, vamos supor hipoteticamente para fins didáticos que:
  - `let a = 10` (criou a variável `a` e armazenou o valor `10` no espaço de memória `679`)
  - `let b = a` (criou a variável `b` e armazenou o valor `10` no espaço de memória `680`)
  - `b += 5` (o valor do espaço de memória com endereço `680` foi incrementado em `5`)

endereço de memória	espaço de memória	
679	10	← valor de <code>a</code>
680	10	← valor de <code>b</code>
681		

endereço de memória	espaço de memória	
679	10	← valor de <code>a</code> após <code>b += 5</code>
680	15	← valor de <code>b</code> após <code>b += 5</code>
681		

# Armazenamento de um Objeto em Memória

- Diferente dos tipos primitivos, os objetos instanciados armazenam uma referência (um endereço) para uma outra região da memória principal do computador onde os dados do objeto estão armazenados

endereço de memória	espaço de memória
111	#547 ← referência do objeto x
112	
113	

Região de memória com a referência do objeto x

endereço de memória	espaço de memória
547	dados do objeto x
548	
549	

Região de memória com os dados do objeto x

# Atribuição de um Objeto em JavaScript

- Ao invés de criar uma cópia dos valores a atribuição passa a referência do objeto (endereço de memória)
  - isso faz com que a variável que recebe a atribuição e o objeto que foi atribuído se tornem um mesmo objeto apenas com nomes diferentes

# Atribuição de um Objeto em JavaScript

```
1 class Produto {
2   tipo;
3   valor;
4   constructor(tipo, valor) {
5     this.tipo = tipo;
6     this.valor = valor;
7   }
8   getDados() {
9     return `
10      Tipo: ${this.tipo} R$ ${this.valor}`;
11   }
12 }
13
14 const prod = new Produto('Generico',50);
15 console.log(prod.getDados())
16 const novoprod = prod;
17 novoprod.tipo = 'Computador';
18 novoprod.valor = 900;
19 console.log(prod.getDados())
```

Ao atribuir o objeto *prod* à variável *novoprod*, a variável *novoprod* e *prod* se referem ao mesmo objeto. Por isso, ao alterar o tipo e o valor do objeto *novoprod*, foi alterado também o tipo e o valor do objeto *prod*

Tipo: Generico R\$ 50

Tipo: Computador R\$ 900

# Armazenamento de um Objeto em Memória

- No exemplo anterior, vamos supor de forma simplificada que a referência do objeto *prod* foi armazenada conforme o exemplo abaixo:

endereço de memória	espaço de memória
111	#547 ← referência do objeto <i>prod</i>
112	
113	

Região de memória com a referência do objeto *prod*

endereço de memória	espaço de memória
547	dados do objeto <i>prod</i>
548	
549	

Região de memória com os dados do objeto *prod*



# Armazenamento de um Objeto em Memória

- A atribuição:
  - `const novoprod = prod;`
    - faz com que ambos objetos possuam a mesma referência

endereço de memória	espaço de memória	
111	#547	referência do objeto <i>prod</i>
112	#547	referência do objeto <i>novoprod</i>
113		

Região de memória com as referências dos objetos *prod* e *novoprod*

endereço de memória	espaço de memória
547	dados dos objetos <i>prod</i> e <i>novoprod</i>
548	
549	

Região de memória com os dados dos objetos *prod* e *novoprod*