# Objetos JavaScript

[❮ Anterior](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/js_validation_api.asp&usg=ALkJrhgyns49qb7VgrjAi8eZDz1LR1-G1A)[Próximo ❯](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/js_properties.asp&usg=ALkJrhhINDG4lvpLk2DfBqsXCnBXaXbjgQ)

Em JavaScript, os objetos são rei. Se você entender objetos, você entende o JavaScript.

Em JavaScript, quase "tudo" é um objeto.

* Os booleanos podem ser objetos (se definidos com a **nova** palavra-chave)
* Os números podem ser objetos (se definidos com a **nova** palavra-chave)
* As strings podem ser objetos (se definido com a **nova** palavra-chave)
* As datas são sempre objetos
* As matemáticas são sempre objetos
* Expressões regulares são sempre objetos
* Arrays são sempre objetos
* As funções são sempre objetos
* Objetos são sempre objetos

Todos os valores de JavaScript, exceto primitivas, são objetos.

## Primitivas de JavaScript

Um **valor primitivo** é um valor que não tem propriedades ou métodos.

Um **tipo de dados primitivo** é um dado que possui um valor primitivo.

O JavaScript define 5 tipos de tipos de dados primitivos:

* corda
* número
* boleano
* nulo
* Indefinido

Os valores primitivos são imutáveis ​​(eles são codificados e, portanto, não podem ser alterados).

se x = 3.14, você pode alterar o valor de x. Mas você não pode alterar o valor de 3.14.

|  |  |  |
| --- | --- | --- |
| **Valor** | **Tipo** | **Comente** |
| "Olá" | corda | "Olá" é sempre "Olá" |
| 3.14 | número | 3.14 é sempre 3.14 |
| verdade | boleano | o verdadeiro é sempre verdadeiro |
| falso | boleano | falso é sempre falso |
| nulo | nulo (objeto) | nulo é sempre nulo |
| Indefinido | Indefinido | indefinido é sempre indefinido |

## Objetos são variáveis ​​contendo variáveis

Variáveis ​​de JavaScript podem conter valores únicos:

### Exemplo

var person = "John Doe";

[Tente você mesmo "](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/tryit.asp%3Ffilename%3Dtryjs_object_variable&usg=ALkJrhgGG4Qa_qEgIsbhnk_c9pwq-xbgSQ)

Os objetos também são variáveis. Mas objetos podem conter muitos valores.

Os valores são escritos como **nome:** pares de **valores** (nome e valor separados por dois pontos).

### Exemplo

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

[Tente você mesmo "](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/tryit.asp%3Ffilename%3Dtryjs_object_object&usg=ALkJrhjDa3GMGJgwA_9iXnEVhJHbmiwUOg)

Um objeto JavaScript é uma coleção de **valores nomeados**

## Propriedades do objeto

Os valores nomeados, em objetos JavaScript, são chamados de **propriedades** .

|  |  |
| --- | --- |
| **Propriedade** | **Valor** |
| primeiro nome | John |
| último nome | Corça |
| era | 50 |
| cor dos olhos | azul |

Os objetos escritos como pares de valores de nome são semelhantes a:

* Arrays associativos em PHP
* Dicionários em Python
* Tabelas Hash em C
* Mapas Hash em Java
* Hashes em Ruby e Perl

## Métodos de objeto

Os métodos são **ações** que podem ser realizadas em objetos.

As propriedades dos objetos podem ser valores primitivos, outros objetos e funções.

Um **método de objeto** é uma propriedade de objeto que contém uma **definição de função** .

|  |  |
| --- | --- |
| **Propriedade** | **Valor** |
| primeiro nome | John |
| último nome | Corça |
| era | 50 |
| cor dos olhos | azul |
| nome completo | function () {return this.firstName + "" + this.lastName;} |

Os objetos JavaScript são recipientes para valores nomeados, chamados propriedades e métodos.

Você aprenderá mais sobre os métodos nos próximos capítulos.

## Criando um objeto JavaScript

Com JavaScript, você pode definir e criar seus próprios objetos.

Existem diferentes maneiras de criar novos objetos:

* Defina e crie um único objeto, usando um objeto literal.
* Defina e crie um único objeto, com a palavra-chave nova.
* Defina um construtor de objetos e, em seguida, crie objetos do tipo construído.

No ECMAScript 5, um objeto também pode ser criado com a função Object.create ().

## Usando um Object Literal

Esta é a maneira mais fácil de criar um objeto JavaScript.

Usando um objeto literal, você define e cria um objeto em uma declaração.

Um objeto literal é uma lista de nomes: pares de valores (como idade: 50) dentro de chaves curvas {}.

O exemplo a seguir cria um novo objeto JavaScript com quatro propriedades:

### Exemplo

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

[Tente você mesmo "](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/tryit.asp%3Ffilename%3Dtryjs_object_create_1&usg=ALkJrhjHqkrKx2R-mc4BiE-3429Nai0FDQ)

Espaços e quebras de linha não são importantes. Uma definição de objeto pode abranger várias linhas:

### Exemplo

var person = {  
    firstName:"John",  
    lastName:"Doe",  
    age:50,  
    eyeColor:"blue"  
};

[Tente você mesmo "](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/tryit.asp%3Ffilename%3Dtryjs_object_create_2&usg=ALkJrhgZiqzRLef8INy0mBuWuQBJ1Sz8Yw)

## Usando a palavra-chave JavaScript nova

O exemplo a seguir também cria um novo objeto JavaScript com quatro propriedades:

### Exemplo

var person = new Object();  
person.firstName = "John";  
person.lastName = "Doe";  
person.age = 50;  
person.eyeColor = "blue";

[Tente você mesmo "](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&ie=UTF8&prev=_t&rurl=translate.google.com.br&sl=en&sp=nmt4&tl=pt-BR&u=https://www.w3schools.com/js/tryit.asp%3Ffilename%3Dtryjs_object_create_new&usg=ALkJrhiTpHR80wdEfWCK6hckR9t1mmHIJQ)

Os dois exemplos acima fazem exatamente o mesmo. Não é necessário usar o novo Objeto ().   
Por simplicidade, legibilidade e velocidade de execução, use o primeiro (o método literal do objeto).

## Objetos JavaScript são Mutable

Os objetos são mutáveis: são abordados por referência, não por valor.

Se a pessoa for um objeto, a seguinte declaração não criará uma cópia da pessoa:

var x = person;  // This will not create a copy of person.

O objeto x **não** é **uma cópia** de pessoa. É **uma** pessoa. Tanto o x quanto a pessoa são o mesmo objeto.

Qualquer alteração em x também mudará de pessoa, porque x e pessoa são o mesmo objeto.

### Exemplo

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"}  
  
var x = person;  
x.age = 10;           // This will change both x.age and person.age