Tipos de dados 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_assignment.asp&usg=ALkJrhgDWJbLDQUlWzrFSJF9on8JN8ZhBg)[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_functions.asp&usg=ALkJrhgtjcL0oxo79PlLoWLiEpHdnff8YQ)

Tipos de dados JavaScript

As variáveis ​​de JavaScript podem conter muitos **tipos de dados** : números, strings, objetos e muito mais:

var length = 16;                               // Number  
var lastName = "Johnson";                      // String  
var x = {firstName:"John", lastName:"Doe"};    // Object

O conceito de tipos de dados

Na programação, os tipos de dados são um conceito importante.

Para poder operar em variáveis, é importante saber algo sobre o tipo.

Sem tipos de dados, um computador não pode resolver com segurança isso:

var x = 16 + "Volvo";

Faz algum sentido adicionar "Volvo" a dezesseis? Isso produzirá um erro ou produzirá um resultado?

JavaScript tratará o exemplo acima como:

var x = "16" + "Volvo";

Ao adicionar um número e uma string, o JavaScript tratará o número como uma string.

Exemplo

var x = 16 + "Volvo";

[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_datatypes_addstring&usg=ALkJrhgzuKuWrxjkZ8JvSLa5d4QJLzCgAw)

Exemplo

var x = "Volvo" + 16;

[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_datatypes_addstring2&usg=ALkJrhiHu2-p65X7CiiPf6XBTRwsctj6Rw)

O JavaScript avalia as expressões da esquerda para a direita. Seqüências diferentes podem produzir resultados diferentes:

JavaScript:

var x = 16 + 4 + "Volvo";

Resultado:

20Volvo

[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_datatypes_addstrings_1&usg=ALkJrhhCzHvtmfXb9XqDFBD14vKvFMgdpw)

JavaScript:

var x = "Volvo" + 16 + 4;

Resultado:

Volvo164

[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_datatypes_addstrings_2&usg=ALkJrhgjo02Ma5arbNZXxmzZSYmWxe02hw)

No primeiro exemplo, o JavaScript trata 16 e 4 como números, até chegar a "Volvo".

No segundo exemplo, uma vez que o primeiro operando é uma string, todos os operandos são tratados como strings.

Tipos de JavaScript são dinâmicos.

O JavaScript possui tipos dinâmicos. Isso significa que a mesma variável pode ser usada para armazenar diferentes tipos de dados:

Exemplo

var x;               // Now x is undefined  
var x = 5;           // Now x is a Number  
var x = "John";      // Now x is a String

[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_datatypes_dynamic&usg=ALkJrhhEgSu-Azh0UbihvU32ZtRicErjcQ)

Cordas de JavaScript

Uma string (ou uma string de texto) é uma série de caracteres como "John Doe".

As strings são escritas com aspas. Você pode usar aspas simples ou duplas:

Exemplo

var carName = "Volvo XC60";   // Using double quotes  
var carName = 'Volvo XC60';   // Using single quotes

[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_datatypes_string_quotes&usg=ALkJrhhA2sRl1oz8R7Rs8M9clIHzLytxdg)

Você pode usar aspas dentro de uma seqüência de caracteres, desde que não correspondam às citações que cercam a seqüência de caracteres:

Exemplo

var answer = "It's alright";             // Single quote inside double quotes  
var answer = "He is called 'Johnny'";    // Single quotes inside double quotes  
var answer = 'He is called "Johnny"';    // Double quotes inside single quotes

[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_datatypes_string&usg=ALkJrhinQxdi3VCljZXNQzq2rMUhZUuAcg)

Você aprenderá mais sobre cordas mais adiante neste tutorial.

Números de JavaScript

O JavaScript tem apenas um tipo de números.

Os números podem ser escritos com, ou sem decimais:

Exemplo

var x1 = 34.00;     // Written with decimals  
var x2 = 34;        // Written without decimals

[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_datatypes_numbers&usg=ALkJrhjsH5sR0-_HRlGOZ9CF-ukqoAMX0A)

Os números extra grandes ou extra pequenos podem ser escritos com notação científica (exponencial):

Exemplo

var y = 123e5;      // 12300000  
var z = 123e-5;     // 0.00123

[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_datatypes_numbers_large&usg=ALkJrhj-eVHyAqqfh4266eiUwyz-6SBDog)

Você aprenderá mais sobre números mais tarde neste tutorial.

Booleanos do JavaScript

Os booleanos só podem ter dois valores: verdadeiro ou falso.

Exemplo

var x = 5;  
var y = 5;  
var z = 6;  
(x == y)       // Returns true  
(x == z)       // Returns false

[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_datatypes_bolean&usg=ALkJrhhmdIvNVWRpvRyqDg-7rEnLT9wfWg)

Os booleanos são freqüentemente usados ​​em testes condicionais.

Você aprenderá mais sobre o teste condicional mais adiante neste tutorial.

Arrays de JavaScript

As matrizes de JavaScript são escritas com colchetes.

Os itens da matriz são separados por vírgulas.

O código a seguir declara (cria) uma matriz chamada carros, contendo três itens (nomes do carro):

Exemplo

var cars = ["Saab", "Volvo", "BMW"];

[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_datatypes_array&usg=ALkJrhjuVfUoy3ttfToKyebrodLstNdcug)

Os índices de matrizes são baseados em zero, o que significa que o primeiro item é [0], o segundo é [1], e assim por diante.

Você aprenderá mais sobre arrays mais adiante neste tutorial.

Objetos JavaScript

Objetos de JavaScript são escritos com chaves curly.

As propriedades dos objetos são escritas como pares nome: valor, separados por vírgulas.

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_datatypes_object&usg=ALkJrhiVQwLzFT09LwAYuPr1k7QxFzaouQ)

O objeto (pessoa) no exemplo acima possui 4 propriedades: firstName, lastName, age e eyeColor.

Você aprenderá mais sobre objetos mais tarde neste tutorial.

O tipo de Operador

Você pode usar o **mecanismo typeof** do JavaScript para encontrar o tipo de uma variável JavaScript.

O **tipoof** operador retorna o tipo de uma variável ou uma expressão:

Exemplo

typeof ""                  // Returns "string"  
typeof "John"              // Returns "string"  
typeof "John Doe"          // Returns "string"

[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_datatypes_typeof_string&usg=ALkJrhgwd7Wj6crfnMNsEDfys2tZE20fEA)

Exemplo

typeof 0                   // Returns "number"  
typeof 314                 // Returns "number"  
typeof 3.14                // Returns "number"  
typeof (3)                 // Returns "number"  
typeof (3 + 4)             // Returns "number"

[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_datatypes_typeof_number&usg=ALkJrhgGS5FmAsd_DVD7nkaS7GdAxpHudQ)

Indefinido

Em JavaScript, uma variável sem valor, tem o valor **indefinido** . O tipo de tipo também é **indefinido** .

Exemplo

var car;                // Value is undefined, type is undefined

[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_datatypes_undefined&usg=ALkJrhhHHaR5WJJwt9bPVarnx9U5UDrvUQ)

Qualquer variável pode ser esvaziada, definindo o valor como **indefinido** . O tipo também será **indefinido** .

Exemplo

car = undefined;        // Value is undefined, type is undefined

[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_datatypes_undefined_2&usg=ALkJrhhKliluj0wPU71QqPiuUfm3eJTO2Q)

Valores vazios

Um valor vazio não tem nada a ver com indefinido.

Uma string vazia tem um valor legal e um tipo.

Exemplo

var car = "";              // The value is "", the typeof is "string"

[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_datatypes_empty&usg=ALkJrhjUkEw5TtzR7zjW15y9dwCZkVLBrA)

Nulo

No JavaScript, null é "nada". É suposto ser algo que não existe.

Infelizmente, em JavaScript, o tipo de dados de nulo é um objeto.

Você pode considerar um bug no JavaScript que typeof null é um objeto. Isso deve ser nulo.

Você pode esvaziar um objeto configurando-o como nulo:

Exemplo

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};  
person = null;        // Now value is null, but type is still an object

[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_datatypes_null&usg=ALkJrhi_-EkpgjtHI4UJ7k2KDK-gE8869A)

Você também pode esvaziar um objeto configurando-o para indefinido:

Exemplo

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};  
person = undefined;   // Now both value and type is undefined

[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_datatypes_undefined_1&usg=ALkJrhilsNV2d8ScFTR8P7vIGfJHXbIgJA)

Diferença entre indefinido e nulo

Indeterminado e nulo são de valor igual, mas diferentes no tipo:

typeof undefined           // undefined  
typeof null                // object  
  
null === undefined         // false  
null == undefined          // true

[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_datatypes_undefined_3&usg=ALkJrhhryEBho1yRd59eCh3TargxB-BXvQ)

Dados primitivos

Um valor de dados primitivo é um único valor de dados simples sem propriedades e métodos adicionais.

O **tipo de** operador pode retornar um desses tipos primitivos:

* corda
* número
* boleano
* Indefinido

Exemplo

typeof "John"              // Returns "string"   
typeof 3.14                // Returns "number"  
typeof true                // Returns "boolean"  
typeof false               // Returns "boolean"  
typeof x                   // Returns "undefined" (if x has no value)

[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_datatypes_typeof_primitive&usg=ALkJrhhRnEyJo9SuSk5gUI0fbNILsQzF1w)

Dados complexos

O **tipo de** operador pode retornar um dos dois tipos complexos:

* função
* objeto

O tipoof operador retorna objeto para objetos, arrays e null.

O tipo de operador não retorna objeto para funções.

Exemplo

typeof {name:'John', age:34} // Returns "object"  
typeof [1,2,3,4]             // Returns "object" (not "array", see note below)  
typeof null                  // Returns "object"  
typeof function myFunc(){}   // Returns "function"