



HTML

CSS

JAVASCRIPT

MORE ▼



JavaScript Objects

[< Previous](#)[Next >](#)

In JavaScript, objects are king. If you understand objects, you understand JavaScript.

In JavaScript, almost "everything" is an object.

- Booleans can be objects (if defined with the **new** keyword)
- Numbers can be objects (if defined with the **new** keyword)
- Strings can be objects (if defined with the **new** keyword)
- Dates are always objects
- Maths are always objects
- Regular expressions are always objects
- Arrays are always objects
- Functions are always objects

- Objects are always objects

All JavaScript values, except primitives, are objects.

JavaScript Primitives

A **primitive value** is a value that has no properties or methods.

A **primitive data type** is data that has a primitive value.

JavaScript defines 5 types of primitive data types:

- `string`
- `number`
- `boolean`
- `null`
- `undefined`

Primitive values are immutable (they are hardcoded and therefore cannot be changed).

if `x = 3.14`, you can change the value of `x`. But you cannot change the value of `3.14`.

| Value | Type | Comment |
|---------|---------|---------------------------|
| "Hello" | string | "Hello" is always "Hello" |
| 3.14 | number | 3.14 is always 3.14 |
| true | boolean | true is always true |
| false | boolean | false is always false |
| | | |

| | | |
|-----------|---------------|-------------------------------|
| null | null (object) | null is always null |
| undefined | undefined | undefined is always undefined |

Objects are Variables

JavaScript variables can contain single values:

Example

```
var person = "John Doe";
```

Try it Yourself »

Objects are variables too. But objects can contain many values.

The values are written as **name : value** pairs (name and value separated by a colon).

Example

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

[Try it Yourself »](#)

A JavaScript object is a collection of **named values**

Object Properties

The named values, in JavaScript objects, are called **properties**.

| Property | Value |
|-----------|-------|
| firstName | John |
| lastName | Doe |
| age | 50 |
| eyeColor | blue |

Objects written as name value pairs are similar to:

- Associative arrays in PHP
- Dictionaries in Python
- Hash tables in C
- Hash maps in Java
- Hashes in Ruby and Perl

Object Methods

Methods are **actions** that can be performed on objects.

Object properties can be both primitive values, other objects, and functions.

An **object method** is an object property containing a **function definition**.

| Property | Value |
|-----------|---|
| firstName | John |
| lastName | Doe |
| age | 50 |
| eyeColor | blue |
| fullName | function() {return this.firstName + " " + this.lastName;} |

JavaScript objects are containers for named values, called properties and methods.

You will learn more about methods in the next chapters.

Creating a JavaScript Object

With JavaScript, you can define and create your own objects.

There are different ways to create new objects:

- Define and create a single object, using an object literal.
- Define and create a single object, with the keyword **new**.
- Define an object constructor, and then create objects of the constructed type.

In ECMAScript 5, an object can also be created with the function `Object.create()`.

Using an Object Literal

This is the easiest way to create a JavaScript Object.

Using an object literal, you both define and create an object in one statement.

An object literal is a list of name:value pairs (like age:50) inside curly braces {}.

The following example creates a new JavaScript object with four properties:

Example

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

Try it Yourself »

Spaces and line breaks are not important. An object definition can span multiple lines:

Example

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

```
    eyeColor: "blue"  
  };  
};
```

Try it Yourself »

Using the JavaScript Keyword new

The following example also creates a new JavaScript object with four properties:

Example

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

Try it Yourself »

The two examples above do exactly the same. There is no need to use `new Object()` .
For simplicity, readability and execution speed, use the first one (the object literal method).

JavaScript Objects are Mutable

Objects are mutable: They are addressed by reference, not by value.

If person is an object, the following statement will not create a copy of person:

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

The object x is **not a copy** of person. It **is** person. Both x and person are the same object.

Any changes to x will also change person, because x and person are the same object.

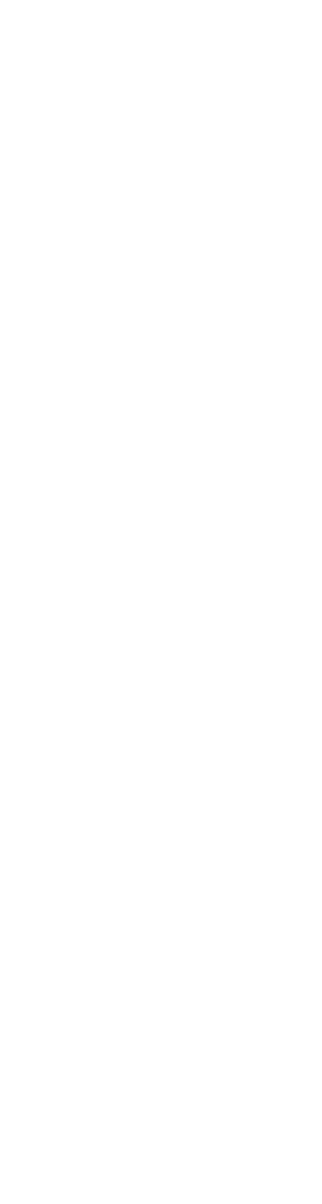
Example

```
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
```

Try it Yourself »

[< Previous](#)

[Next >](#)



COLOR PICKER



HOW TO

Tabs
Dropdowns
Accordions
Side Navigation
Top Navigation
Modal Boxes
Progress Bars
Parallax
Login Form
HTML Includes
Google Maps
Range Sliders
Tooltips
Slideshow
Filter List
Sort List

SHARE



CERTIFICATES

HTML

CSS

JavaScript

SQL

Python

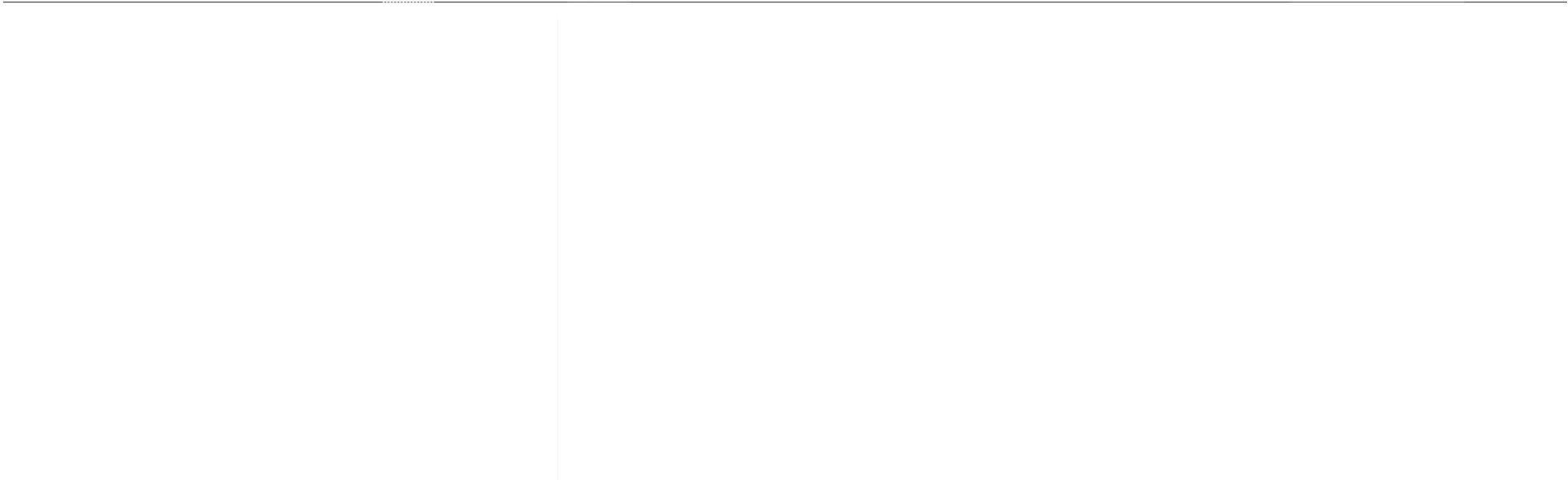
PHP

jQuery

Bootstrap

XML

[Read More »](#)



[REPORT ERROR](#)

[PRINT PAGE](#)

[FORUM](#)

[ABOUT](#)

[Top Tutorials](#)

[Top References](#)

- HTML Tutorial
- CSS Tutorial
- JavaScript Tutorial
- How To Tutorial
- SQL Tutorial
- Python Tutorial
- W3.CSS Tutorial
- Bootstrap Tutorial
- PHP Tutorial
- jQuery Tutorial
- Java Tutorial

Top Examples

- HTML Examples
- CSS Examples
- JavaScript Examples
- How To Examples
- SQL Examples
- Python Examples
- W3.CSS Examples
- Bootstrap Examples
- PHP Examples
- jQuery Examples
- Java Examples
- XML Examples

- HTML Reference
- CSS Reference
- JavaScript Reference
- SQL Reference
- Python Reference
- W3.CSS Reference
- Bootstrap Reference
- PHP Reference
- HTML Colors
- jQuery Reference
- Angular Reference
- Java Reference

Web Certificates

- HTML Certificate
- CSS Certificate
- JavaScript Certificate
- SQL Certificate
- Python Certificate
- jQuery Certificate
- PHP Certificate
- Bootstrap Certificate
- XML Certificate

[Get Certified »](#)

W3Schools is optimized for learning, testing, and training. Examples might be simplified to improve reading and basic understanding. Tutorials, references, and examples are constantly reviewed to avoid errors, but we cannot warrant full correctness of all content. While using this site, you agree to have read and accepted our terms of use, cookie and privacy policy. Copyright 1999-2019 by Refsnes Data. All Rights Reserved.

Powered by W3.CSS.

