[JavaScript Objects (w3schools.com)](https://www.w3schools.com/js/js_object_definition.asp)

[JS Objects](https://www.w3schools.com/js/js_object_definition.asp)

# JS Objects

[Object Definitions](https://www.w3schools.com/js/js_object_definition.asp" \t "https://www.w3schools.com/js/_top)

[Object Properties](https://www.w3schools.com/js/js_object_properties.asp" \t "https://www.w3schools.com/js/_top)

[Object Methods](https://www.w3schools.com/js/js_object_methods.asp" \t "https://www.w3schools.com/js/_top)

[Object Display](https://www.w3schools.com/js/js_object_display.asp" \t "https://www.w3schools.com/js/_top)

[Object Accessors](https://www.w3schools.com/js/js_object_accessors.asp" \t "https://www.w3schools.com/js/_top)

[Object Constructors](https://www.w3schools.com/js/js_object_constructors.asp" \t "https://www.w3schools.com/js/_top)

[Object Prototypes](https://www.w3schools.com/js/js_object_prototypes.asp" \t "https://www.w3schools.com/js/_top)

[Object Iterables](https://www.w3schools.com/js/js_object_iterables.asp" \t "https://www.w3schools.com/js/_top)

[Object Sets](https://www.w3schools.com/js/js_object_sets.asp" \t "https://www.w3schools.com/js/_top)

[Object Maps](https://www.w3schools.com/js/js_object_maps.asp" \t "https://www.w3schools.com/js/_top)

[Object Reference](https://www.w3schools.com/js/js_object_es5.asp" \t "https://www.w3schools.com/js/_top)

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

JavaScript Objects

You have already learned that JavaScript variables are containers for data values.

This code assigns a ****simple value**** (Fiat) to a ****variable**** named car:

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

This code assigns ****many values**** (Fiat, 500, white) to a ****variable**** named car:

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

JavaScript objects are containers for ****named values****called properties.

Example

In the example above, this refers to the ****person object****.

I.E. ****this.firstName**** means the ****firstName**** property of ****this****.

I.E. ****this.firstName**** means the ****firstName**** property of ****person****.

**Object Definitions**

JavaScript Primitives

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

Immutable(不可改变的)

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

JavaScript variables can also contain many values.

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

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

Object 的值写成多个成对的 **name : value**

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

Object Properties

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

**Object Properties**

Accessing(访问) JavaScript Properties

The JavaScript for...in statement loops through the properties of an object

**Object Methods**

**[Object Display](https://www.w3schools.com/js/js_object_display.asp" \t "https://www.w3schools.com/js/_top)**

Some common solutions to display JavaScript objects are:

* Displaying the Object Properties by name
* Displaying the Object Properties in a Loop
* Displaying the Object using Object.values()
* Displaying the Object using JSON.stringify()

在页面的展示形式有：

**[object Object]**

as a string：

an array

Json

查看对象的内部结构，内部的值

output ，输出

下面说明了对象属性的两种取值方法的区别（适用环境），中括号中的可以是常量也可以是变量，但点的方式只能是常量

You must use **person[x]** in the loop.

**person.x** will not work (Because **x** is a variable).

myString is now a JavaScript string, ready to be displayed:

**Object Accessors**

**Data Quality**

JavaScript can secure(使安全) better data quality when using getters and setters.

Using the lang property, in this example, returns the value of the language property in upper case:

**Why Using Getters and Setters?**

* It gives simpler syntax
* It allows equal syntax for properties and methods
* It can secure better data quality
* It is useful for doing things behind-the-scenes

Object.defineProperty()

The Object.defineProperty() method can also be used to add Getters and Setters:

**Object Constructors**

* Adding a Method to an Object
* Adding a Method to an Object
* Adding a Property to a Constructor
* Adding a Method to a Constructor

**What is this?**

In JavaScript, the this keyword refers to an **object**.

javascript中的this指向一个对象，或者说javascript的this是一个对象。

属性中的this是拥有该属性的对象，

**Which** object depends on how this is being invoked (used or called).

The this keyword refers to different objects depending on how it is used:

|  |
| --- |
| In an object method, this refers to the **object**. |
| Alone, this refers to the **global object**. |
| In a function, this refers to the **global object**. |
| In a function, in strict mode, this is undefined. |
| In an event, this refers to the **element** that received the event. |
| Methods like call(), apply(), and bind() can refer this to **any object**. |

Note

this is not a variable. It is a keyword. You cannot change the value of this.

## About this(\*\*\*\*\*)

In a constructor function this does not have a value. It is a substitute for the new object. The value of this will become the new object when a new object is created.([JavaScript Constructors (w3schools.com)](https://www.w3schools.com/js/js_object_constructors.asp))

由上可知，this是一个替身，一个对象的替身，在没用到时我们不会知道它是谁的替身，只有当我们用到它时，我们才会知道代表着什么

See Also:

[The JavaScript](https://www.w3schools.com/js/js_this.asp)**[this](https://www.w3schools.com/js/js_this.asp)**[Tutorial](https://www.w3schools.com/js/js_this.asp)

[The JavaScript this Keyword](https://www.w3schools.com/js/js_this.asp)

**Object Prototypes**

**Object Iterables(**迭代**)**

iterate

[iterables](javascript:;) 参数

可迭代对象

可迭代的

可遍历

Iterable objects are objects that can be iterated(反复申明) over with for..of.

Technically(技术上；学术上), iterables must implement the Symbol.iterator method.

**Object Sets、**[JS Sets](https://www.w3schools.com/js/js_sets.asp" \t "https://www.w3schools.com/js/_top)

**A JavaScript Set is a collection of unique values.** Each value can only occur once in a Set.

**Essential Set Methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| new Set() | Creates a new Set |
| add() | Adds a new element to the Set |
| delete() | Removes an element from a Set |
| has() | Returns true if a value exists in the Set |
| forEach() | Invokes a callback for each element in the Set |
| values() | Returns an iterator with all the values in a Set |
| **Property** | **Description** |
| size | Returns the number of elements in a Set |

**How to Create a Set**

You can create a JavaScript Set by:

* Passing an Array to new Set()
* Create a new Set and use add() to add values
* Create a new Set and use add() to add variables

**Object Maps、**[JS Maps](https://www.w3schools.com/js/js_maps.asp" \t "https://www.w3schools.com/js/_top)

**Object Reference**

# JS Functions

[Function Definitions](https://www.w3schools.com/js/js_function_definition.asp" \t "https://www.w3schools.com/js/_top)

[Function Parameters](https://www.w3schools.com/js/js_function_parameters.asp" \t "https://www.w3schools.com/js/_top)

[Function Invocation](https://www.w3schools.com/js/js_function_invocation.asp" \t "https://www.w3schools.com/js/_top)

[Function Call](https://www.w3schools.com/js/js_function_call.asp" \t "https://www.w3schools.com/js/_top)

[Function Apply](https://www.w3schools.com/js/js_function_apply.asp" \t "https://www.w3schools.com/js/_top)

[Function Bind](https://www.w3schools.com/js/js_function_bind.asp" \t "https://www.w3schools.com/js/_top)

[Function Closures](https://www.w3schools.com/js/js_function_closures.asp" \t "https://www.w3schools.com/js/_top)