# Objects

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).

### Example

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

## Object Properties

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

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

## 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 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:

|  |
| --- |
| const x = person;  // 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

|  |
| --- |
| const person = {   firstName:"John",   lastName:"Doe",   age:50, eyeColor:"blue" }  const x = person; x.age = 10;      // Will change both x.age and person.age |

## JavaScript Properties

Properties are the values associated with a JavaScript object.

A JavaScript object is a collection of unordered properties.

Properties can usually be changed, added, and deleted, but some are read only.