JavaScript ES5 Object Methods

❮ Previous Next ❯

ECMAScript 5 added a lot of new Object Methods to JavaScript.

ES5 New Object Methods

```
// Adding or changing an object property
Object.defineProperty(object, property, descriptor)

// Adding or changing many object properties
Object.defineProperties(object, descriptors)

// Accessing Properties
Object.getOwnPropertyDescriptor(object, property)

// Returns all properties as an array
Object.getOwnPropertyNames(object)

// Returns enumerable properties as an array
Object.keys(object)

// Accessing the prototype
Object.getPrototypeOf(object)

// Prevents adding properties to an object
Object.preventExtensions(object)
// Returns true if properties can be added to an object
Object.isExtensible(object)

// Prevents changes of object properties (not values)
Object.seal(object)
// Returns true if object is sealed
Object.isSealed(object)

// Prevents any changes to an object
Object.freeze(object)
```

Changing a Property Value

Syntax

```
Object.defineProperty(object, property, {value : value})
```

This example changes a property value:

Example

```
var person = {
  firstName: "John",
  lastName : "Doe",
  language : "EN"
};

// Change a property
Object.defineProperty(person, "language", {value : "NO"});
```

Try it Yourself »

Changing Meta Data

ES5 allows the following property meta data to be changed:

```
writable : true      // Property value can be changed
enumerable : true    // Property can be enumerated
configurable : true  // Property can be reconfigured
```

```
writable : false      // Property value can not be changed
enumerable : false    // Property can be not enumerated
configurable : false // Property can be not reconfigured
```

```
// Defining a getter
get: function() { return language }
// Defining a setter
set: function(value) { language = value }
```

This example makes language read-only:

```
Object.defineProperty(person, "language", {writable:false});
```

This example makes language not enumerable:

```
Object.defineProperty(person, "language", {enumerable:false});
```

Listing All Properties

This example list all properties of an object:

Example

```
var person = {
  firstName: "John",
  lastName : "Doe"
  language : "EN"
};

Object.defineProperty(person, "language", {enumerable:false});
Object.getOwnPropertyNames(person);  // Returns an array of properties
```

Try it Yourself »

Listing Enumerable Properties

This example list only the enumerable properties of an object:

Example

```
var person = {
  firstName: "John",
  lastName : "Doe"
  language : "EN"
```

```
Object.defineProperty(person, "language", {enumerable:false});
Object.keys(person);  // Returns an array of enumerable properties
```

Try it Yourself »

Adding a Property

This example adds a new property to an object:

Example

```
// Create an object:
var person = {
  firstName: "John",
  lastName : "Doe",
  language : "EN"
};

// Add a property
Object.defineProperty(person, "year", {value:"2008"});
```

Try it Yourself »

Adding Getters and Setters

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

Example

```
//Create an object
var person = {firstName:"John", lastName:"Doe"};

// Define a getter
Object.defineProperty(person, "fullName", {
  get : function () {return this.firstName + " " + this.lastName;}
});
```

Try it Yourself »

A Counter Example

```
// Define object
var obj = {counter:0};

// Define setters
Object.defineProperty(obj, "reset", {
  get : function () {this.counter = 0;}
});
Object.defineProperty(obj, "increment", {
  get : function () {this.counter++;}
});
Object.defineProperty(obj, "decrement", {
  get : function () {this.counter--;}
});
Object.defineProperty(obj, "add", {
  set : function (value) {this.counter += value;}
});
Object.defineProperty(obj, "subtract", {
  set : function (i) {this.counter -= i;}
});

// Play with the counter:
obj.reset;
obj.add = 5;
obj.subtract = 1;
obj.increment;
obj.decrement;
```

Try it Yourself »

❮ Previous Next ❯