Object.prototype

`Object.prototype` 属性表示 `Object` 的原型对象。

`Object.prototype` 属性的属性特性：

writable	false
enumerable	false
configurable	false

描述

JavaScript中几乎所有的对象都是 `Object` 的实例; 所有的对象都继承了 `Object.prototype`的属性和方法， 它们可以被覆盖（除了以null为原型的对象，如 `Object.create(null)`）。 例如，新的构造函数的原型覆盖原来的构造函数的原型，提供它们自己的 `toString()` 方法.。对象的原型的改变会传播到所有对象上，除非这些属性和方法被其他对原型链更里层的改动所覆盖。

属性

`Object.prototype.constructor`
特定的函数，用于创建一个对象的原型。

`Object.prototype.__proto__` ⚠
指向当对象被实例化的时候，用作原型的对象。

`Object.prototype.__noSuchMethod__` ⚠
当未定义的对象成员被调用作方法的时候，允许定义并执行的函数。

`Object.prototype.__count__` 🗑
用于直接返回用户定义的对象中可数的属性的数量。已被废除。

`Object.prototype.__parent__` 🗑
用于指向对象的内容。已被废除。

方法

`Object.prototype.__defineGetter__()` ⚠👎

关联一个函数到一个属性。访问该函数时，执行该函数并返回其返回值。

Object.prototype.__defineSetter__() ⚠️👎

关联一个函数到一个属性。设置该函数时，执行该修改属性的函数。

Object.prototype.__lookupGetter__() ⚠️👎

返回使用 `__defineGetter__` 定义的方法函数 。

Object.prototype.__lookupSetter__() ⚠️👎

返回使用 `__defineSetter__` 定义的方法函数。

Object.prototype.hasOwnProperty()

返回一个布尔值 ，表示某个对象是否含有指定的属性，而且此属性非原型链继承的。

Object.prototype.isPrototypeOf()

返回一个布尔值，表示指定的对象是否在本对象的原型链中。

Object.prototype.propertyIsEnumerable()

判断指定属性是否可枚举，内部属性设置参见 ECMAScript DontEnum attribute 。

Object.prototype.toSource() ⚠️

返回字符串表示此对象的源代码形式，可以使用此字符串生成一个新的相同的对象。

Object.prototype.toLocaleString()

直接调用 `toString()` 方法。

Object.prototype.toString()

返回对象的字符串表示。

Object.prototype.unwatch() ⚠️

移除对象某个属性的监听。

Object.prototype.valueOf()

返回指定对象的原始值。

Object.prototype.watch() ⚠️

给对象的某个属性增加监听。

Object.prototype.eval() 🗑

在指定对象为上下文情况下执行javascript字符串代码，已经废弃。

例子

由于JavaScript没有子类对象，原型是一种常用的方法，用于为某些表现为对象的函数创建一个“基类”对象。例如：

```
var Person = function() {
  this.canTalk = true;
  this.greet = function() {
    if (this.canTalk) {
      console.log('Hi, I\'m ' + this.name);
    }
  };
};

var Employee = function(name, title) {
  this.name = name;
  this.title = title;
  this.greet = function() {
    if (this.canTalk) {
      console.log("Hi, I'm " + this.name + ", the " + this.title);
    }
  };
};
Employee.prototype = new Person();

var Customer = function(name) {
  this.name = name;
};
Customer.prototype = new Person();

var Mime = function(name) {
  this.name = name;
  this.canTalk = false;
};
Mime.prototype = new Person();

var bob = new Employee('Bob', 'Builder');
var joe = new Customer('Joe');
var rg = new Employee('Red Green', 'Handyman');
var mike = new Customer('Mike');
var mime = new Mime('Mime');
bob.greet();
joe.greet();
rg.greet();
```

```
40 mike.greet();
41 mime.greet();
```

输出：

```
1 Hi, I'm Bob, the Builder
2 Hi, I'm Joe
3 Hi, I'm Red Green, the Handyman
4 Hi, I'm Mike
```

规范

规范	描述	备注
ECMAScript 第一版，在JavaScript 1.0.已实现	标准	初始定义
ECMAScript 5.1 (ECMA-262) Object.prototype	ST Standard	
ECMAScript 2015 (6th Edition, ECMA-262) Object.prototype	ST Standard	

浏览器兼容性

Desktop Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari
基本支持	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)