一、原型(prototype)

- 定义：一个简单的对象，用于实现对象的属性继承。可以简单理解成对象的爹。在Firefox和Chorme中，每个JS对象中都包含一个`__proto__`（非标准）的属性指向该对象的原型，可`obj.__proto__`进行访问
- 实例是通过构造函数（new）创建出来的对象，实例通过`__proto__`指向原型，通过`constructor`指向构造函数
- 以上描述可表示为：`var a = new Object(); a.__proto__ === a.contructor.prototype`

二、原型链(prototype chain)

- 原型链是由原型对象组成，每个对象都有一个`__proto__`属性，指向了创建该对象的构造函数的原型，`__proto__`将对象链接起来组成了原型链。是一个用来实现继承和共享属性的有限的对象链。
- 在 javaScript 中，每个对象都有一个指向它的原型（prototype）对象的内部链接。这个原型对象又有自己的原型，直到某个对象的原型为 null 为止（也就是不再有原型指向），组成这条链的最后一环。这种一级一级的链结构就称为原型链（prototype chain）

```
function A(){};
var a = new A();
// a  ===> a.__proto__(A.prototype) ===> A.prototype.__proto__(Object.prototype) ===>
Object.prototype.__proto__ ===> null
```

三、instanceof函数

定义：instanceof运算时会递归查找L的原型链，即`L.__proto__.__proto__.__proto__.__proto__...`直到找到了或者找到顶层为止。一句话理解instanceof的运算规则为：**`instanceof`检测左侧的`__proto__`原型链上，是否存在右侧的`prototype`原型。**

有一个有趣的现象：

```
Function instanceof Object; // true
Object instanceof Function; // true
```

为何`Object instanceof Function`为true?

- Object, Function, Array等等这些都被称作是构造“函数”（想一下我们可以`var a = new Object()`来创建一个对象，所以`Object`本身就是一个构造函数），他们都是函数。而所有的函数都是构造函数Function的实例。
- 从原型链机制的的角度来说，即`Function.prototype`在`Object`的原型链上。Function这个内置函数的Function.prototype是一个函数a，因为函数同时也是对象，因此这个函数a也定义了apply、call、bind等属性（或者说方法）。而Object的委托目标`Object.__proto__`正是这个函数a，代码表示为：`Function.prototype === Object.__proto__ // true`

阅读 244 发布于 2019-03-01

