# \_\_proto\_\_和prototype的区别

\_\_proto\_\_ 主要是安放在一个实际的对象中，用它来产生一个链接，一个原型链，用于寻找方法名或属性，等等。

prototype 是用 new 来创建一个对象时构造 \_\_proto\_\_ 用的。它是构造函数的一个属性。

**一个\_\_proto\_\_的例子：**

var a = {

x: 10,

calculate: function (z) {

return this.x + this.y + z;

}

};

var b = {

y: 20,

\_\_proto\_\_: a

};

var c = {

y: 30,

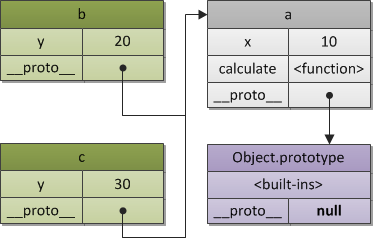
\_\_proto\_\_: a

};

// call the inherited method

b.calculate(30); // 60

c.calculate(40); // 80



注意：ES5 中，规定原型继承需要使用 Object.create() 函数。

var b = Object.create(a, {y: {value: 20}});

var c = Object.create(a, {y: {value: 30}});

**一个prototype的例子：**

// 一种构造函数写法

function Foo(y) {

this.y = y;

}

// 修改 Foo 的 prototype，加入一个成员变量 x

Foo.prototype.x = 10;

// 修改 Foo 的 prototype，加入一个成员函数 calculate

Foo.prototype.calculate = function (z) {

return this.x + this.y + z;

};

// 现在，我们用 Foo 这个原型来创建 b 和 c

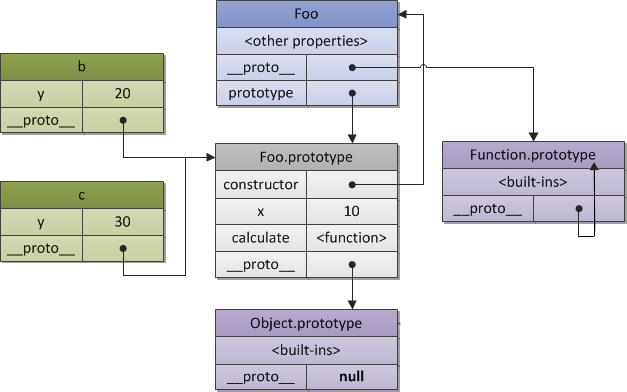
var b = new Foo(20);

var c = new Foo(30);

// 调用原型中的方法，可以得到正确的值

b.calculate(30); // 60

c.calculate(40); // 80



从这张图就可以很容易的看出\_\_proto\_\_和prototype的区别了。

值得一说的是，基于原型（Prototype）的编程其实也是面向对象编程的一种方式。没有 class 化的，直接使用对象。JavaScript就是典型的代表，ES6中的class不过是多了一层语法糖而已。