目录

[我是怎么悟出原型的 2](#_Toc61198633)

# 我是怎么悟出原型的

原型我会用，那是因为我会服务端开发，我做的本身就是面向对象的开发。但是有很多地方我还是不明白。比如：function person(){};var lisi = new person();

为啥person.\_\_proto\_\_ === Function.prototype？

为啥person.prototype.\_\_proto\_\_ === Object.prototype？

为啥person.prototype.constructor === person？

为啥lisi没有prototype？

后来有一天我突然想到了“原型”两字，原型在生活中就经常使用啊，比如《天不藏奸》里的大坏蛋江啸的原型就是张君，众泰SR7就是以保时捷卡宴为原型仿制的。那么原型是不是就可以理解为模仿、克隆（括号里的这一段可以不看，只是原型设计模型的题外话，设计模型中的“原型模式”就是克隆，核心思想就是“克隆远比创造来的快”，创建一个对象的开销非常大，如果构造函数中有非常复杂的计算，每次实例化一个对象的时候都要重复这种复杂的计算，克隆的话只要第一次计算，后面直接复制就行，可以省略无数次的重复计算）。想象一下：早期国外只要出一个新鲜玩意，淘宝可能会比原型产品更早的出现山寨版？人家花了很多时间弄出一个产品，你一天或几天时间就复制出来了，这不就是复制原型吗？

所以这里要注意一点，模仿和复制原型是不是要将原型的属性也复制过来呀？然后可以再加点自己的属性，就可以形成自己的产品。

举个例子：保时捷发布了卡宴。

function kayan(){}

kayan.prototype = Object.assign(kayan.prototype, {

name: ‘卡宴’, //名称

speed: ‘230’, //最高时速

//……..更多的属性

});

众泰看到后马上抄袭发布了sr7：

function sr7() {}

sr7.prototype = Object.create(kayan.prototype, {

name: ‘sr7’

});

众泰的第一台sr7原型车出来了，完全仿着kayan抄袭，属性完全一样，只是改了个名字。然后众泰就开始按照原型车开始生产了。new sr7();

过了一段时间之后，大家就开始开骂了，说众泰无耻，完全抄袭，众泰就想了个办法，我在我的第一台原型车上新增一些功能，再把名字改一下，尽量和卡宴做出差异化，于是众泰就在sr7的基础上做了个升级，叫做sr8：

function sr8() {}

sr8.prototype = Object.create(sr7.prototype, {

name: ‘sr8’,

led: true, //加led灯

autoHold: true, //自动泊车

//…….更多属性

});

然后开始生产…… new sr8();

有一天产品经理问车间人员：sr8原型车的原型车在哪？（sr.prototype.\_\_proto\_\_）。

车间人员一脸懵逼，你们当初没说sr8原型车的原型车编号啊（工厂车太多，都是按编号管理的，上哪找去？）？

产品经理发现自己的失误了，然后马上对sr8进行升级，做sr9。

function sr9() {}

sr9.prototype = Object.assign(sr9.prototype, {

\_\_proto\_\_: sr8.prototype //sr9的\_\_proto\_\_（想象成编码）指向sr8的原型

});

然后就开始sr9的生产了。var car1 = new sr9();……

好了，这里咱们想象一下：

car1已经是按照sr9原型车生成出来的具体车辆了，car1有prototype吗？

car1.\_\_proto\_\_ === ?（这里可以理解为car1的编码指向谁？）

sr9.prototype.\_\_proto\_\_ === ?（这里可以理解为sr9的原型车的编码指向谁？）

通过car1.\_\_proto\_\_找到sr9.prototype，通过sr9.prototype.\_\_proto\_\_能找到sr8.prototype，是不是可以通过\_\_proto\_\_这个属性一条一条的往上溯源？一条一条的\_\_proto\_\_连接起来是不是跟链条似的？这就是原型链。

最后有个难以理解的地方，就是sr9和sr9.prototype的关系，sr9是啥？由于js语言的特殊性，ES5的面向对象的表达很傻叉（所以才有了ES6的class），所以这一块理解就比较难。

按照js来解释的话，sr9是封装的对象，也是构造函数，对象包含一个叫prototype（原型）的属性，这个属性的constructor又指向包含这个原型的构造函数，这里有点绕，做个比喻：

有一座庙(function temple(){})，庙里住着（包含）一个和尚（prototype），你问和尚，你住哪啊？那和尚的容身之所是哪啊？是谁包含这个和尚啊？是庙（temple）。然后又来了，庙里住着一个和尚，和尚的容身之所是庙。用代码表示：

function temple(){}

temple.prototype.constructor === temple;

temple.prototype.constrcutor.prototype === ?

temple.prototype.constructor.prototype.constructor === ?

最后的感悟总结：大家都说艺术来源于生活，我觉得其实编程也来源于生活，因为面向对象不就是要贴近现实和生活吗？所以很多事物要从现实角度，贴近生活的东西来理解（将对象具体化），这样就会容易很多，比如“原型”两字。

**请大家有感悟就输出出来，输出感悟既能帮助大家，更能加深自己的理解，不要害怕自己理解错误。**