目录

[我是怎么悟出原型的 2](#_Toc26986)

[马俊杰0112晚自习总结 4](#_Toc21462)

[马俊杰学习心得 4](#_Toc1740)

[周宇成学习心得 4](#_Toc13166)

[1.细化 4](#_Toc18539)

[2.检索 5](#_Toc13945)

[3.测验 5](#_Toc11313)

[4.盘点 5](#_Toc30357)

# 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。**

# 马俊杰0112晚自习总结

原型与原型链的理解:

我对于原型与原型链的理解，来源于今天的学习与自己的理解结合，原型的概念就是模仿与复制，非常简单的逻辑，比如一个最简单的例子

在一部电影中人物的塑造就是按照以前真实的人物来模仿的这就是原型的概念，只要能理解原型的概念那么来理解她也就好理解的多，

就是来举例的众泰汽车与保时捷，在保时捷卡宴出来以后，众泰sr3以保时捷卡宴为原型模仿了她，到后来众泰又以新的sr4以sr3为原型加了led灯光，

那么我们是不是可以认为sr4的prototype就是sr3的prototype而sr3的原型又是谁呢？

sr3的的汽车是不是以保时捷卡宴为原型来复制克隆的她呢，那么原型的理解是不是就是以什么为原型来复制她，复制下来的东西，不光可以继承原型的属性，还可以自己

加属性，那么原型链又是什么呢？原型链就相当与一个个查找追根溯源，就还是小汽车的例子，众泰sr4发布时被查出来它有模仿的嫌疑，那么这时候警察要来查找模仿谁了

那么是不是从sr4来向上面查找找到sr4是以sr3为原型创造的，没有问题，但是查到sr3是以保时捷卡宴为原型模仿的，这样的查找就是原型链的查找！

上课注意听课，才会有自己的感悟与心得，师傅领进门，学习靠个人！！！

# 马俊杰学习心得

不管我们做什么，都要去规划她，然后去全力以赴的完成，今天世界上最成功的内些人，有几个人是唯唯诺诺的，等人吩咐以后才做事情，

对待自己的前途与学业，应该清楚的知道自己的未来要变成一个什么样的人，今天你向着这个目标努力了多少，积极学习的人一定是目标非常明确的人

因为他清楚的知道自己想要什么，要得到这个成就需要付出多大的努力，积极学习总比被动学习来的快一点

只要我们把思想开阔了，我认为学习不需要督促，因为一种未来的责任感会让你去自主学习

# 周宇成学习心得

## **1.细化**

##### **学习的东西吸收，用自己的语言理解并转化成自己的东西**

##### **场景模拟理论结合实际**

##### **比喻法：回想一些相同概念的案例或事，便于理解**

## **2.检索**

##### **回忆今天的学习内容，都有哪些知识点，不能讲到最后把前面的忘了**

##### **复习/默写，如果不熟练或忘了，就多复习，熟能生巧。**

##### **间隔法，多样化的学习方式，没事就多看看，别把时间浪费，闲着也是闲着。**

##### **日报和反思-抽查**

##### **今日学会了什么知识点？复习的？新的？分类总结。**

##### **作业是否完成，完成情况，是否有未懂的知识，有没有其他的解决办法？**

##### **今日待改进的地方/不足之处/有没有进步。**

## **3.测验**

##### **互相提问题，测验，看看自己是不是真的掌握了，别到时候因为紧张或者某种问题而犯低级错误。**

##### **提问题 抢答 听写**

##### **日测和周测**

## **4.盘点**

##### **这一周都学会了什么，都有那些不易理解的东西，有哪些落下的知识，掌握了多少？**

##### **学习的方法有没有效，是否更改后更有进步？**

##### **遇到事情有没有第一时间自己解决，都犯了哪些错，有没有拓展自己的知识面(对知识有更深一层的理解)。**

# 我的学习方式

1. 阅读过程中我会用书上的代码去做跑一遍，加深自己的印象，直到直到这个方法是干嘛的，怎么用。
2. 会用之后，我会想，如果换个条件或者参数会是怎样的（通常书中不会提及所有情况），比如：Array的fill方法，第二个参数是开始下标，可以为负数，那我就会想如果负数超过长度会怎样。例：

var arr = [1, 2, 3];

arr.length === 3;   //true

arr.fill(5, -3, 3);

这个很容易理解，3-3 = 0。然后我就会想 -4,-5,-6,-10,-100...这些怎么计算。

3-4 = -1; 3-5 = -2; 3-6 = -3; 3-100 = -97;

结果都超过长度3太多了，像这种超太多的又是怎么计算的？

其实我完全可以去百度查，但这种简单的东西，自己试一下就能试出来了，为什么一定要百度呢，查出来的肯定没有自己试出来的印象深刻。

所以我就自己去试了下，刚开始错了（以为要计算到位正数为止），后来又用了很多个数字去试，最后发现再次计算还是负数的话，直接算0，这样我自己试出来的理解就非常深刻了。

1. 理解之后容易忘记，我会自己去造一些数据，多练几遍，或者结合其他方法写一个小案例。
2. 俗话说“好记性不如烂笔头”，我会在边看书的过程中，将学到的知识，理解的知识，全部记录下来，并且在最后加一个思考（就是变相用法）。