

JS 总结

从入门到工作：JS 全解

版权声明

本内容版权属杭州饥人谷教育科技有限公司（简称饥人谷）所有。

任何媒体、网站或个人未经本网协议授权不得转载、链接、转贴，或以其他方式复制、发布和发表。

已获得饥人谷授权的媒体、网站或个人在使用时须注明「资料来源：饥人谷」。

对于违反者，饥人谷将依法追究其责任。

联系方式

如果你想要购买本课程

请微信联系 [xiedaimala02](#) 或 [xiedaimala03](#)

如果你发现有人盗用本课程

请微信联系 [xiedaimala02](#) 或 [xiedaimala03](#)

现在我们学了什么

回忆一下关键词

知识点

- 基本概念

- ✓ 内存、变量、数据类型、对象

- 控制语句

- ✓ if...else...
- ✓ for...

- 对象

- ✓ 原型、原型链
- ✓ 对象分类
- ✓ new 一个新对象
- ✓ 构造函数
- ✓ this 的隐式传递和显式传递

难点

- JS 三座大山

- ✓ 原型
- ✓ this
- ✓ AJAX

- 我们已经遇到了前两座

- ✓ 我还是没理解
- ✓ 没理解就对了
- ✓ 你理解了那它还是三座大山吗
- ✓ 前端门槛就在这
- ✓ 那怎么办?
- ✓ 很简单，反复学反复理解

最重要的知识

把其他知识都忘掉

第一个重要知识：JS 公式

对象.__proto__ === 其构造函数.prototype

JS 唯一公式，如果不会就套公式

第二个重要知识：根公理

Object.prototype 是所有对象的(直接或间接)原型
加了一个直接或间接，所谓公理就是规定好的

第三个重要知识：函数公理

所有函数都是由 Function 构造的

任何函数.__proto__ === Function.prototype

任意函数有 Object / Array / Function

JS 公式、根公理、函数公理

基于这三个知识和基础知识

可以推出 JS 世界

拨乱反正

我希望你的大脑现在是乱的，这样我才能拨乱反正

乱一

- XXX 的原型

- ✓ {name:'frank'} 的原型
- ✓ [1,2,3] 的原型
- ✓ Object 的原型

- 解读

- ✓ Object 的原型是 Object.__proto__: 对
- ✓ Object 的原型是 Object.prototype: 错

- 错在哪

- ✓ 「的原型」等价于「.__proto__」
- ✓ 中文的「原型」无法区分 __proto__ 和 prototype
- ✓ 所以我们只能约定，原型默认表示 __proto__
- ✓ 只不过 __proto__ 正好等于某个函数的 prototype

乱二

- 我觉得老师你矛盾了

- ✓ [1,2,3] 的原型是 Array.prototype
- ✓ 你有说 Object.prototype 是所有对象的原型
- ✓ 那为什么 Object.prototype 不是 [1,2,3] 的原型

- 错在哪

- ✓ 原型分两种：直接原型和间接原型
- ✓ 对于普通对象来说，Object.prototype 是直接原型
- ✓ 对于数组、函数来说，Object.prototype 是间接原型

乱三

- Object.prototype 不是根对象

- 理由

- ✓ Object.prototype 是所有对象的原型
- ✓ Object 是 Function 构造出来的
- ✓ 所以，Function 构造了 Object.prototype
- ✓ 推论，Function 才是万物之源啊！

- 错在哪

- ✓ Object.prototype 和 Object.prototype 对象的区别
- ✓ 对象里面从来都不会包含另一个对象
- ✓ 接下来我们要把 JS 世界的建造顺序理清楚

我们再构建一次 JS 世界

神说，要有光

JS 世界的构造顺序

1. 创建根对象 #101(toString), 根对象没有名字
2. 创建函数的原型 #208(call /apply), 原型 __p 为 #101
3. 创建数组的原型 #404(push/pop), 原型 __p 为 #101
4. 创建 Function #342, 原型 __p 为 #208
5. 用 Function.prototype 存储函数的原型, 等于 #208
6. 此时发现 Function 的 __proto__ 和 prototype 都是 #208
7. 用 Function 创建 Object
8. 用 Object.prototype 存储对象的原型, 等于 #101
9. 用 Function 创建 Array
10. 用 Array.prototype 存储数组的原型, 等于 #404
11. 创建 window 对象
12. 用 window 的 'Object' 'Array' 属性将 7 和 9 中的函数命名
13. 记住一点, JS 创建一个对象时, 不会给这个对象名字的

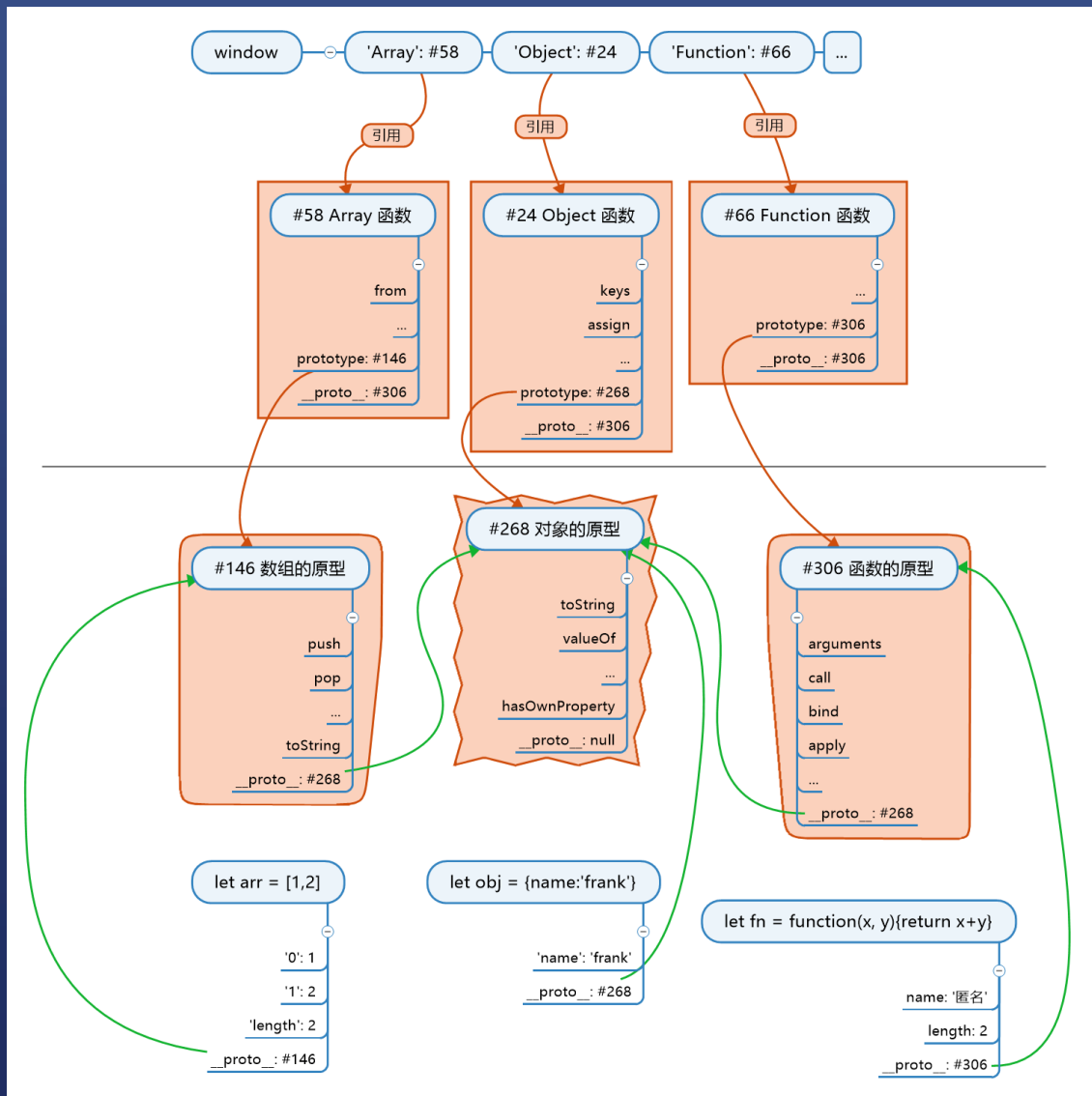
JS 世界的构造顺序（续）

1. 用 `new Object()` 创建 `obj1`
2. `new` 会将 `obj1` 的原型 `__p` 设置为 `Object.prototype`, 也就是 `#101`
3. 用 `new Array()` 创建 `arr1`
4. `new` 会将 `arr1` 的原型 `__p` 设置为 `Array.prototype`, 也就是 `#404`
5. 用 `new Function` 创建 `f1`
6. `new` 会将 `f1` 的原型 `__p` 设置为 `Function.prototype`, 也就是 `#208`

JS 世界的构造顺序（续）

1. 自己定义构造函数 Person，函数里给 this 加属性
2. Person 自动创建 prototype 属性和对应的对象 #502
3. 在 Person.prototype #502 上面加属性
4. 用 new Person() 创建对象 p
5. new 会将 p 的原型 __p 设为 #502

漂亮的图示



总结

- 构造函数

- ✓ 是用来构造对象的
- ✓ 会预先存好对象的原型，原型的原型是根
- ✓ new 的时候将对象的 __p 指向原型

- 对象

- ✓ 所有对象都直接或间接指向根对象
- ✓ 如果对象想要分类，就在原型链上加一环
- ✓ 用构造对象可以加这一环

- 思考

- ✓ 如果加了一环之后，想再加一环怎么办
- ✓ 以后会在「继承」里讲

还有什么没学

- 错误处理 try...catch...
 - ✓ 可以看看[网道教材](#)（也可以不看）
 - ✓ 会在项目中深入理解
- Math / Date / 正则 / JSON
 - ✓ 可以先看教材（也可以不看）
 - ✓ 会在项目中深入理解
- 我觉得
 - ✓ 就算现在讲了，你明天也会忘
 - ✓ 听不如写，写不如做

JS 学习计划

- 第一阶段

- ✓ 了解 JS 语法、特性、对象、数组、函数等

- 第二阶段

- ✓ 了解 AJAX、设计模式、封装、面向对象、MVC

- 第三阶段

- ✓ Vue / React 全家桶
- ✓ 好了，可以工作了

再见

再继续学习 JS 之前，我们先学习算法和数据结构