「每日一题」你可以不会 class，但是一定要学会 prototype

2020年07月20日 04:33 4282浏览 · 12喜欢 · 1评论

饥人谷编程
粉丝：2.5万 文章：117
+ 关注

今天看到有人问「为什么明知 JS 的 class 是假的，程序员还不厌其烦地实现 class，而不使用原型」。

作为 JS 原型的坚定支持者，我不得不喷一下「JS 里的 class 比 prototype 好」这个观点。

目录

「每日一题」你可以不会 class，但是一定要学会 prototype

饥人谷编程 ＋ 关注

2. prototype 的问题

    1. ES 6 之前大家不想了解 prototype
    2. ES 6 之前 prototype 操作不方便

3. 去学 prototype

逼格

我先来提高一下 prototype 的逼格：

> 「所有」理解 prototype 的 JS 程序员都能轻松理解 JS 里的 class；
> 反过来，理解 class 的 JS 程序员却「有很多」都理解不了 prototype。

这就是为什么早年间很大 JS 大神不得不用 prototype 去模拟一个蹩脚的 klass 给新人用。

我没有把 klass 拼错，因为 class 是保留字，所以大神只能用 klass 来「代替」class。

虽然我认为原型是 JS 的基础知识，但是根据实际情况，原型居然变成了 JS 的高级知识，很多新人直接在入门的时候放弃学习原型，而去学 class。

所以，prototype 绝对能让你有智商优越感 :)

好了，开始正经讨论问题。

JS 里 class 的问题

1 第一个问题，功能缺失。

我是一个实用主义者，如果 class 比 prototype 好用，我绝对会总是使用 class。

但是目前情况是：

1. JS 里 class 的功能还不如 prototype
2. JS 里 class 的功能比 Java 里的 class 更是差了十万八千里

1.1 JS 里 class 的功能还不如 prototype

以 let frank = new Human() 为例。

如果你要给 Human 加一个 x 属性，怎么加?

对于使用原型的人来说，Human 不就是一个函数对象吗，直接加就是了，何必多此一问：

完

对于使用 class 的人来说，经典 OO 理论会使他倾向与把 Human 看做一个 class，所以应该使用这样的语法

然而，这个语法，还没有纳入 ES 标准，所以目前在 JS 里用 class 连个静态字段都定义不了，真可惜。

「每日一题」你可以不会 class，但是一定要学会 prototype

饥人谷编程 + 关注

如果这样写的话 Human 还是类的话，那我就不说什么了。你问问 Java 程序员能不能理解这样不伦不类的 class。

总之，目前你不可能把 JS 的 class 当成真正的类，你还是得把 Human 看成一个函数对象才行。

有些人可能还会说，我等 class 语法升级不就行了。

你都能升级语法了，那我怎么还能说得过你呢，对吧，只要我说 class 哪里有问题你都可以用升级解决。所以我只讨论目前的 ES 语法。

目前使用 class 前你还是需要完全理解 prototype。

1.2 JS 里 class 的功能比 Java 里的 class 更是差了十万八千里

请用 JS 的 class 写出一个抽象类。

对不起，目前 class 做不到，等升级语法吧。

JS 的 class 如何实现 private、public 和 protected。

对不起，目前 class 做不到，等升级语法吧。

可千万别说你可以用其他方法模拟，模拟出个阉割版有意思吗?

有些人可能会说，class 实现不了我认，你 prototype 不也实现不了吗？大家都实现不了，凭什么说我有问题。

这就是一个自相矛盾的地方了，prototype 为什么要去模拟 class 的特性?

在 prototype 体系里，根本就没有类，更遑论抽象类和 private 关键字了。

没有类（class）！

怎么有些人老想着用 JS 模拟 class 呢?

不解决这个问题，用 JS 就会总觉得别扭。

想一想：为什么你非要在一个原型语言里，使用 class 思维来思考问题。
因为你先入为主地觉得 class 比 prototype 好啊……

如果你还没意识自己的问题，恐怕我是真的扳不过来了。

再说一遍：不要把 JS 当做经典面向对象语言（比如 Java）使用。

如果你只学 JS 里的 class 不学 prototype，恐怕最终的后果就是既没有学会经典面向对象，又没有学会原型。

方言称之为半吊子，普通话称之为半壶水。

所以 JS 的 class 到底是个啥?

class 是 prototype 的语法糖而已。

「每日一题」你可以不会 class，但是一定要学会 prototype

饥人谷编程 + 关注

2 第二个问题

如果你说我们不能对 JS 这门动态语言要求太多，class 能部分实现 prototype 的功能，够用就行了，没必要再去学习 prototype 了。

那我依然不同意，因为 class 还有一个更严重的问题：

「只会 class」将使你无法系统理解 JS。
依然以最简单的代码为例

面向对象的解释：Human 类有一个成员方法 sayHi。

只会 class 的同学现在请回答一个问题

Human 的类型是什么?
答案是 'object'（对象），同时 Human 也是一个函数。class 是一个函数，你不觉得奇怪吗?

再问一个问题，console.dir(Human) 你会看到它有以下属性：

请问 Human 为什么有 prototype 属性?
为什么 Human.prototype 有一个 sayHi 属性?

看吧，你始终绕不开原型…… JS 的 class 总会把你引向原型。

我实在是无法理解只会 class 的 JS 开发者要如何理解 JS：

1. 你需要无视每个对象的 __proto__ 属性
2. 你需要无视每个类的 prototype 属性

class 的这个问题才是我反对只学 class 的最关键原因。

无论 JS 的 class 语法如何升级，不管是 ES 6 还是 ES 7、ES 8、ES 9、ES 10、ES 11，都绕不开 prototype。永远都绕不开。因为这是 JS 里对象的本质。

如果绕不开 prototype，你还有什么理由不学 prototype？还有什么理由不好好地、深入地学习 prototype?

有些人可能会说，那我先学会 prototype，但是不用 prototype 只用 class 行不行呢?

「每日一题」你可以不会 class，但是一定要学会 prototype

饥人谷编程 + 关注

但你也要学会 prototype 才行了，然后才能理解 class 呀！

所以本文标题是「你可以不会 class，但是一定要学会 prototype」。

推荐学习路径：

1. 完全理解 JS 的 prototype
2. 熟练使用 prototype
3. 学会 class
4. 熟练使用 class
5. 想用 class 的时候用 class，想用 prototype 的时候用 prototype

你硬要当半吊子只学 class 的话，我也没意见。

3 第三个问题：并不能做到类型安全

class 的第三个问题是，class 带来的好处并不多，除了作为糖能让你少写一些代码外，并没有多少额外的好处。用语法糖我不反对，但是如果只用语法糖不学语法糖背后的原理，那我就反对。

有观点认为 class 能使 JS 更加「类型安全」。

持这种观点的人可能对 JS 有误解，一门「没有编译阶段」的「动态」「弱类型」语言怎么可能类型安全啊……JS 任何对象随时都可以被改得面目全非，用了 class 也无济于事。

想要类型安全去用 TypeScript 吧，class 并不能拯救 JS。

而且类型安全属于另一个问题，可以另开帖子讨论。

另外不要以为 TypeScript 里面的 class 跟 JS 里的 class 一样，虽然 TypeScript 是 JS 的超集，但是有编译阶段的 class 和没编译阶段的 class 就是质的区别。

这就是编译的威力。不是 class 的威力。

prototype 的问题

class 这么多问题，难道 prototype 就是完美的吗？

如果 prototype 那么好，这么会有这么多人喜欢 class 呢？

我先来解释为什么 JS 里 class 的拥趸如此多。

因为最早写 JS 的人，都是从 Java 和 PHP 转行的。

尤其是 Java，Java 里的 class 很好用，所以这批人非常想直接在 JS 里面使用 class。

然而一写才发现 JS 的 class 居然只是一个保留字。

于是他们实在受不了不能写 class 的日子，硬生生用 prototype 模拟出了一个 klass。

嗯，klass 就是一个阉割版的 class。

为何这群人放着 prototype 不用，非要用 klass 呢？

因为 JS 这门语言很烂，他们不想去理解 JS。

由于他们不想去理解 JS，所以也不会去理解 JS 的 prototye 了。

「每日一题」你可以不会 class，但是一定要学会 prototype

饥人谷编程 + 关注

[illegible]

不，没有反转，是你理解错了，我刚刚说得是 JS 的原型比 JS 的 class 好，没说 JS 语言（ES 6 之前）本身有多好。

JS 语言本身（在 ES 6 之前）还是很烂的，为此道格拉斯还专门写了 JS the good parts 这本书来把 JS 为数不多的优点给列出来。

JS 多烂我以后再讨论。但是 JS 的原型绝对是好东西。

正如 JS 之父所说：

> 它的优秀之处并非原创，它的原创之处并不优秀。——JS 之父引用别人的话描述 JS

实际上 JS 原创之处全都很垃圾。这个原型就不是 JS 原创的，而且原型很优秀。

这些程序员不想理解 JS，更不想在 JS 的一堆垃圾特性里面挑出 prototype 来学习，所以宁愿用 klass，因为他们熟悉 class。

所以 prototype 的第一个问题是，它隐藏在 JS 的一堆烂特性中，无法被人发现。

那么第二个问题是什么？

第二个问题是在 ES 6 之前，人们无法很方便地对原型进行操作。

ES 6 之前，你要改变一个对象的原型要怎么做？

你可能会以为这样就可以改变原型：

但是这个代码很可能报错，因为 __proto__ 到目前为止，都是不推荐使用的特性。

ES 6 出了 Object.create 和 Object.setPrototypeOf 才让开发者能安心操作原型。

那么正确的代码是什么呢？

以上三行代码是 JS 程序不懈努力才发现的实现 frank.__proto__ = anotherPrototype 的方法。

现在估计你能理解了「为什么在 ES 6 之前，JS 程序员不喜欢 prototype」，因为理解难度实在有点大啊：

1. 首先你要耐心学会 JS 的各种烂语法，然后走到 prototype 面前
2. 为了兼容性，你需要用特别奇怪的代码来操作 prototype

但是 ES 6 来了之后，这两个问题都解决了呀：

1. JS 的各种烂语法都有对应的修正，如 var 被 let 代替
2. 操作 prototype 很简单了，用 Object.create 就好

所以，现在，你应该去学 prototype 了。

「每日一题」你可以不会 class，但是一定要学会 prototype

饥人谷编程 + 关注

学习过程中推荐看看「方三篇」：

《this 的值到底是什么？一次说清楚》

https://zhuanlan.zhihu.com/p/23804247

《JS 的 new 到底是干什么的？》

《「每日一题」什么是 JS 原型链？》

以及我的一个回答：（想快速理解 prototype 可以看这个回答）

《JS 中 __proto__ 和 prototype 存在的意义是什么？》

https://www.zhihu.com/question/56770432/answer/315342130

• • •

本文作者方应杭，欢迎进群与我探讨技术（加小圆微信：xiedaimala03）

JS 前端 前端学习 前端工作 自学前端