黑客派

[译]Javascript 2018: 你需要知道的和不需要知道的一些东西

作者：zjhch123

原文链接：https://hacpai.com/article/1517060232106

来源网站：黑客派

原文链接JavaScript 2018: Things You Need To Know, and a Few You Can Skip - Michelle Gienw

Javascript已经发展成熟——我们不再是只使用它在网页上制作一些简单的交互，而是更多的使用它建一整套大型的应用。当然了，随着当前Javascript的生态系统不断完善，这会比之前都更复杂。

作为一名职业的软件工程师以及两本与Javascript有关书籍的作者，Ethan Brown花费了很多时间去析目前和Javascript有关的技术栈，并且从中得出了一位现代的“Javascript开发者”该如何去适应这东西。

这里有一些Brown的关于Javascript生态系统的各个方面最前沿或者也是至少会很有用的预测，在201里，机智的开发者可以去熟悉这些东西。有两点需要说明的是：第一，他只是指出了他认为好的观点其中有一些可能是你没有去关注的。对这些知识有一个基本的意识可以让你在发生问题难以抉择时知该去找谁或者该去探索那些技术。

第二，这些观点只是他基于他对于生态系统的经验提出的他的观点，当然可能和你的关注点完全不同Brown说，“这只是我的观点，我们各有各的想法，并且我知道在这其中我可能会有一些疏漏”。

从我们的关注点开始

WebAssembly:

WebAssembly是Javascript的一个子集，它提供了一个针对其他语言的编译器。所以如果你想将C+编译成Javascript，WebAssembly是你正确的选择——它允许将其他语言编译成可以在浏览器环境或者Node.js环境下运行的有趣的应用程序。Brown说，“我觉得它会火，会越来越重要。在2018，一定会花更多的时间与精力去学它。”

Functional Programming(函数式编程):

"这并不是一个新的概念，它已经被Javascript社区中的大多数所应用，但是我觉得在2018这个概念影响力会变得更广，其使用会变得相当关键。"Brown说。”大多数人都批评函数式编程非常难学非难懂，这也是一个非常明显的问题：当我们开始使用函数式编程时，充满迷惑性并且混乱的编码看起非常古怪且难懂”。

Brown建议，“如果你想开始学习函数式编程，你可以看一看Elm或者ClojureScript。但是你也可以句话从今天开始：‘OK，我的所有代码都会都要变成纯函数式的了’。”

Immutability(不可变性):

这点可以和函数式编程关联在一起。“尽管有大多数人刚开始使用不可变性的时候都会想：‘Wow这似乎根本没什么用啊，我创建了所有对象的拷贝，所有的内存占用看起来都没什么必要啊’。”Brown说，“但需要记住的是，你只拷贝了变化的一部分——另一部分结构还是保持不变的。并且，在Javascript中，严格比较的速度更快内存开销更低，许多实用了不可变性的同学都表示他们的程序的运行率更高了。”

更好的是，不可变性为了一些实验性行为提供了天然的保护网——你在知道你不会对原有对象进行修，而是只会创建修改的副本的情况下，这会鼓励你去对一些不熟悉的事物进行尝试。这对一些新手程员也非常好。

单项数据绑定:

“这是前端同学需要关注的事。这是由Elm提出、并且由Facebook的Flux发扬光大的观点。当时Facebook推出了Redux，而现在这个观点同样被Angular和Vue吸收。”Brown说，越来越多的人意识到是一个非常棒的方案，并且2018是一个非常好的时间去掌握它。

单项数据绑定确实让你的应用程序的状态更容易去管理。当你第一次去尝试的时候你会想，天啊，又做这么多工作，这看上去太过分了。同样的，对一些小的应用程序来说也的确是太过分了。但是一旦的应用程序达到了某种量级，这真的能帮助你控制好整个应用，并且不只是你关注的部分。因为当你用单项数据绑定时，你需要考虑到数据在应用程序中的每一层的流动。

计算属性名/属性简写:

"这个东西在ES6中出乎意料的好。但是我没有见到很多人在用它。我认为它可以在很多地方被使用。是一个小而美的语法糖，它允许你动态创建属性名、初始化对象或者简写对象属性值。"，Brown说“我感觉像是我每周都可以发现一些很Cool的使用它的方式。它同样也可以被应用在函数式编程中。果你之前没有见过的话一定要去了解一下——我非常希望看见社区里越来越多的使用它。”

不用太过担心这些

就像Brown所说的那样，在现在，这些领域内的东西我们可以跳过。

面向对象编程:

"在使用Javascript的时候我不常使用传统的面向对象编程。我认为会有更好的方法、更好的范式去实代码复用。所以我不觉得在Javascript中你会为涉及到面向对象编程的知识而感到苦恼。"

Generators:

"在Javascript中这是一个很棒的特性，在许多地方也一定非常有用。但是我认为这里面最重要的一点经被async/await所取代了。我们很兴奋的在Koa.js中使用Generators进行同步语义的异步变成，但现在我们有了async和await并且他们可以表现的更好。其实你觉得在一些很奇怪的案例中使用Generaors会比较合理的话，也不用太在意。"Brown说。

Symbol:

"另外一个非常好的特性，是对Javascript语言的一个非常棒的补充。但是我并没有发现人们有在使用而且每当我尝试去使用Symbol的时候我除了在框架和序列化问题上出现错误意外得不到任何结果。"Bown说。总的来说，他不认为Javascript生态系统内真的需要Symbols，并且这个或许不是一个能很的适应我们正在使用的Javascript的正确的语法特性。他的建议是：保持观望。