AngularJS 的缺点

PUBLISHED MARCH 26TH 2015 BY QINGNIU

当一项新技术出现的时候，有时让我们着迷，而且让我们觉得它一切皆有可能。

正如我们观察到的，该项技术在某些方面超出我们的预期，我们开始相信我们可以相对容易地在任何地方使用该技术。然而实际经历证明：这简直就是自找苦吃！我们 thoughtworks.com 技术团队使用 AngularJS 去实现 “interactive tech radar”，我作为其中的一员，谈谈我的一些使用体会。

一点儿背景信息

我们有一个 Ruby，Sinatra 和 Plain JavaScript 的技术架构。关于项目开发堆栈的背景信息，你可以在 Andy Robinson 所写的文章中找到。指导原则的其中一项是，代码库的维护是通过最大限度地减少意外的复杂性来实现的。因此，结论就是，我们力求用库，而不是框架。一个具体例子就是选择 Sinatra 而不是 Rails，主要是因为前者作为一个框架，比后者轻得多。

“interactive tech radar” 项目是用 AngularJS 实现的。选择 AngularJS 可以快速开发

出最小可视化原型（MVP）产品，可以实现更丰富的客户端交互。比起我们之前充满文本的静态页面，最终产品的用户体验会有极大的提升。

实战故事

首当其冲的影响出现在我们的构建阶段。AngularJS 的引入，明显增加了测试的时间。我们的测试工具集（RSpec, Capybara, PhantomJS）很适合我们现在开发的程序：多页面应用程序，一个页面里面只有有限的（但会越来越多）JavaScript 交互。AngularJS 的引入，让我们直接面对一个有趣的挑战：我们如何去测试需要跨越多个页面进行 JavaScript 交互的单页面应用？使用 AngularJS 特定的工具像 Protractor，意味着要引入一个新的测试工具集，和当前的测试工具集一起并行工作。维护两个不同的测试工具集，增加了复杂性，而又似乎不会得到什么好处。我们最终使用了 AngularJS 兼容的测试工具集（例如 capybara-angular）。由此开始显现了使用 AngularJS 的一个缺点：不能和非 AngularJS 特定的工具或其它编程库一起使用。

进入第二轮迭代时，更多的功能和错误修正出现在工作列表上。在我们着手解决这些问题时，我们感觉到了由学习曲线陡增所带来的压力。随着我们对 AngularJS 理解的深入，即使较小用户故事也要花费好几天时间。尽管我们对此已有预判，但仍低估了实际所要花费的时间。主要原因是，由于 AngularJS 框架的本质决定的。像任何其它框架一样，它拥有自己独特的世界观。为了获得使用框架的最大好处，使用者必须接受和使用它做事的方式。编程框架总是神奇和美妙的 - 帮你快速实现 MVP，但长远来看，你不得不为此在代码的维护和进化上付出必要的代价。

在工作中，我们经常发现，解决问题的唯一途径就是 AngularJS 提供的方式。JS 库的使用被严格限制了。通常，你只能使用属于其生态系统部分的 JavaScript 工具。例如，基于 jQuery 的库就不能和 AngularJS 构建的页面一起很好的工作。我们还发现，使用 JavaScript 做些简单的事情变得很困难。一个例子就是使用 DOM onready/onload 事件（因为 AngularJS 劫持了这些事件,你不能轻松地使用它们）。另一个例子是，谷歌分析不能直接与 AngularJS 一起工作，你必须使用特定的 AngularJS 库才能使其运行（如 angulartics）。此外，AngularJS 给我们的代码引入了 html，这和我们使用 slim 模板语言的目的一样。出现一组新的工具意味着增加代码库的意外复杂性，这将使代码维护和演化变得更加困难。

我们的业务需求之一，就是让我们的网站对非 JavaScript 用户可用。这意味着我们

必须复制几乎所有交互信息（...破坏了 DRY 原则）所有这些都在一个大的 noscript 标签内！我们看到的另外一个问题是，AngularJS 提供的搜索引擎优化方案并不理想。有些解决办法当然可能更好（见 StackOverflow 上的问题），但它们都需要额外的工作。

未来

对于 Radar 项目来说，我们所能预料到下一个问题将在语言国际化方面。这意味着在 JavaScript 中实现国际化机制，需要复制当前我们在 Ruby 中已实现的国际化。鉴于已有的痛苦经历，我们意识到，随着业务需求越来越复杂，继续使用 AngularJS 进行开发，我们将很难应对。在这一点上，似乎未来可能的方法就是去除 AngularJS，使用 Ruby 页面和纯 JavaScript 来代替。尽管这看起来像是一个巨大的任务，工作量会在短时间内飙升，但如果能有效重用现有的一些 JavaScript 和 HTML 代码，也就是几天的工作而已。

结论

虽然我认为所有我们遇到的小问题都有更简单优雅的解决方案，但它仍然不能改变这一事实，AngularJS 带来了很多意外的复杂性，却没有为它付出而带来的额外的利益。如果我开发一个单页面的 JavaScript 应用程序的话，我很乐意采用和学习 AngularJS，但若将其用作其它方面，我会三思而后行。

作者：*Mircea Moise* 是 *Thoughtworks* 公司的一名软件工程师，热衷 *Ruby &* 敏捷开发。

原文： *AngularJS: The Bad Bits*

感谢： *Jodoo* 帮助审阅并完成校对。