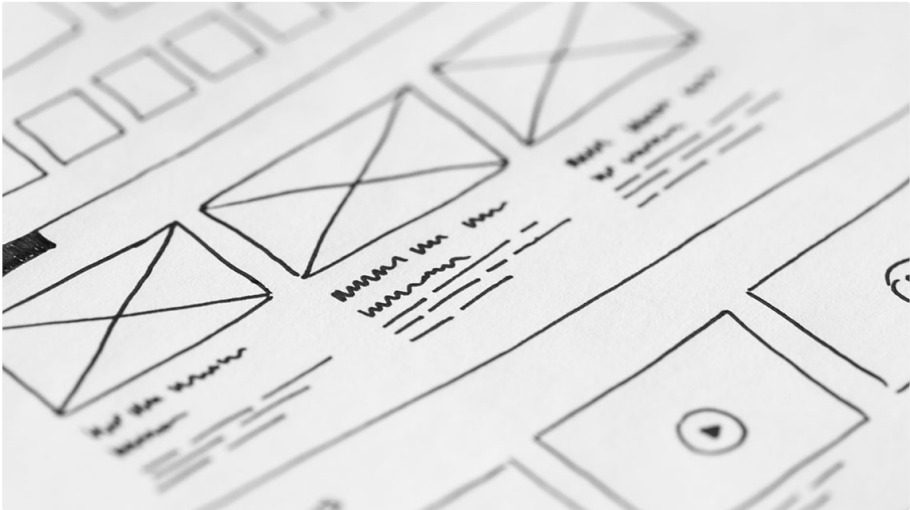


程序员练级攻略（2018）：前端性能优化和框架

2018-08-02 陈皓





程序员练级攻略（2018）：前端性能优化和框架

陈皓

- 00:02 / 13:24

前端性能优化

首先是推荐几本前端性能优化方面的图书。

- [Web Performance in Action](#)，这本书目前国内没有卖的。你可以看电子版，我觉得是一本很不错的书，其中有CSS、图片、字体、JavaScript性能调优等。
- [Designing for Performance](#)，这本在线的电子书很不错，其中讲了很多网页优化的技术和相关的工具，可以让你对整体网页性能优化有所了解。
- [High Performance JavaScript](#)，这本书在国内可以买到，能让你了解如何提升各方面的性能，包括代码的加载、运行、DOM交互、页面生存周期等。雅虎的前端工程师尼古拉斯·扎卡斯（Nicholas C. Zakas）和其他五位JavaScript专家介绍了页面代码加载的最佳方法和编程技巧，来帮助你编写更为高效和快速的代码。你还会了解到构建和部署文件到生产环境的最佳实践，以及有助于定位线上问题的工具。
- [High Performance Web Sites: Essential Knowledge for Front-End Engineers](#)，这本书国内也有卖，翻译版为《高性能网站建设指南：前端工程师技能精髓》。作者给出了14条具体的优化原则，每一条原则都配以范例佐证，并提供了在线支持。

全书内容丰富，主要包括减少HTTP请求、Edge Computing技术、Expires Header技术、gzip组件、CSS和JavaScript最佳实践、主页内联、Domain最小化、JavaScript优化、避免重定向的技巧、删除重复JavaScript的技巧、关闭ETags的技巧、Ajax缓存技术和最小化技术等。
- 除了上面这几本书之外，Google的 [Web Fundamentals](#) 里的 [Performance](#) 这一章节也有很多非常不错的知识和经验。

接下来是一些最佳实践性的文档。

- [Browser Diet](#)，前端权威性能指南（中文版）。这是一群为大型站点工作的专家们建立的一份前端性能的工作指南。
- [PageSpeed Insights Rules](#)，谷歌给的一份性能指南和最佳实践。
- [Best Practices for Speeding Up Your Web Site](#)，雅虎公司给的一份7个分类共35个最佳实践的文档。

接下来，重点推荐一个性能优化的案例学习网站 [WPO Stats](#)。WPO是Web Performance Optimization的缩写，这个网站上有很多很不错的性能优化的案例分享，一定可以帮助你很多。

然后是一些文章和案例。

- [A Simple Performance Comparison of HTTPS, SPDY and HTTP/2](#)，这是一篇比较浏览器的HTTPS、SPDY和HTTP/2性能的文章，除了比较之外，还可以让你了解一些技术细节。
- [7 Tips for Faster HTTP/2 Performance](#)，对于HTTP/2来说，Nginx公司给出的7个增加其性能的小提示。
- [Reducing Slack's memory footprint](#)，Slack团队减少内存使用量的实践。
- [Pinterest: Driving user growth with performance improvements](#)，Pinterest关于性能调优的一些分享，其中包括了前后端的一些性能调优实践。其实也是一些比较通用的玩法，这篇文章主要是想让前端的同学了解一下如何做整体的性能调优。
- [10 JavaScript Performance Boosting Tips](#)，10个提高JavaScript运行效率的小提示，挺有用的。
- [17 Statistics to Sell Web Performance Optimization](#)，这个网页上收集了好些公司的Web性能优化的工程分享，都是非常有价值的。
- [Getting started with the Picture Element](#)，这篇文章讲述了Responsive布局所带来的一些负面的问题。主要是图像适配的问题，其中引出了一篇文章“[Native Responsive Images](#)”，值得一读。

- [Improve Page Load Times With DNS Prefetching](#) , 这篇文章教你一个如何降低DNS解析时间的小技术——DNS prefetching。
- [Jank Busting for Better Rendering Performance](#) , 这是一篇Google I/O上的分享, 关于前端动画渲染性能提升。
- [JavaScript Memory Profiling](#) , 这是一篇谷歌官方教你如何使用Chrome的开发工具来分析JavaScript内存问题的文章。

接下来是一些性能工具。在线性能测试分析工具太多, 这里只推荐比较权威的。

- [PageSpeed](#) , 谷歌有一组PageSpeed工具来帮助你分析和优化网站的性能。Google出品的, 质量相当有保证。
- [YSlow](#) , 雅虎的一个网页分析工具。
- [GTmetrix](#) , 是一个将PageSpeed和YSlow合并起来的一个网页分析工具, 并且加上一些Page load或是其它的一些分析。也是一个很不错的分析工具。
- [Awesome WPO](#) , 在GitHub上的这个Awesome中, 你可以找到更多的性能优化工具和资源。

另外, 中国的网络有各种问题 (你懂的) , 所以, 你不能使用Google共享的JavaScript链接来提速, 你得用中国自己的。你可以到这里看看中国的共享库资源, [Forget Google and Use These Hosted JavaScript Libraries in China](#) 。

前端框架

接下来, 要学习的是Web前端的几大框架。目前而言, 前端社区有三大框架Angular.js、React.js和Vue.js。我认为, React和Vue更为强劲一些, 所以, 我这里只写和React和Vue相关的攻略。关于两者的比较, 网上有好多文章。我这里推荐几篇我觉得还不错的, 供你参考。

- [Angular vs. React vs. Vue: A 2017 comparison](#)
- [React or Vue: Which JavaScript UI Library Should You Be Using?](#)
- [ReactJS vs Angular5 vs Vue.js - What to choose in 2018?](#)

其实, 比较这些框架的优缺点还有利弊并不是要比出个输赢, 而是让你了解一下不同框架的优缺点。我觉得, 这些框架都是可以学习的。而在我们生活工作中具体要用哪个框架, 最好还是要有一些出发点, 比如, 你是为了找份好的工作, 为了快速地搭一个网站, 为了改造一个大规模的前端系统, 还是纯粹地为了学习……

不同的目的会导致不同的决定。我并不希望上述的这些比较会让你进入“二选一”或是“三选一”的境地。我只是想通过这些文章让你知道这些框架的设计思路 and 实现原理, 这些才是让你受益一辈子的事情。

React.js 框架

下面先来学习一下React.js框架。

入门

React学起来并不复杂, 就看 [React 官方教程](#) 和其文档就好了 ([React 的中文教程](#)) 。

然后, 下面的文章会带你了解一下React.js的基本原理。

- [All the fundamental React.js concepts](#) , 这篇文章讲了所有的React.js的基本原理。
- [Learn React Fundamentals and Advanced Patterns](#) , 这篇文章中有几个短视频, 每个视频不超过5分钟, 是学习React的一个很不错的地方。
- [Thinking in React](#) , 这篇文章将引导你完成使用React构建可搜索产品数据表的思考过程。

提高

学习一个技术最重要的是要学到其中的思想和方法。下面是一些我觉得学习React中最重要的东西。

- 状态, 对于富客户端来说是非常麻烦也是坑最多的地方, 这里有几篇文章你可以一读。
 - [Common React.js mistakes: Unneeded state](#) , React.js编程的常见错误——不必要的状态。
 - [State is an Anti-Pattern](#) , 关于如何做一个不错的组件的思考, 很有帮助。
 - [Why Local Component State is a Trap](#) , 一些关于 “Single state tree” 的想法。
 - [Thinking Statefully](#) , 几个很不错的例子让你对声明式的有状态的技术有更好的理解。
 - 传统上, 解决React的状态问题一般用Redux。在这里推荐 [Tips to learn React + Redux in 2018](#) 。Redux是一个状态粘合组件, 一般来说, 我们会用Redux来做一些数据状态和其上层Component上的同步。这篇教程很不错。
 - 最后是 “State Architecture Patterns in React ” 系列文章, 非常值得一读。
 - [Part 1: A Review](#)
 - [Part 2: The Top-Heavy Architecture, Flux and Performance](#)
 - [Part 3: Articulation Points, zine and An Overall Strategy](#)
 - [Part 4: Purity, Flux-duality and Dataflow](#)
- 函数式编程。从jQuery过来的同学一定非常不习惯React, 而从Java等后端过来的程序员就会很习惯了。所以, 我觉得React就是后端人员开发的, 或者说是做函数式编程的人开发的。对此, 你需要学习一下JavaScript函数式编程的东西。

这里推荐一本免费的电子书 《[Professor Frisby's Mostly Adequate Guide to Functional Programming](#)》, 其中译版为《[JS函数式编程指南中文版](#)》。

下面有几篇文章非常不错。前两篇和函数式编程有关的文章非常值得一读。后三篇是一些比较实用的函数式编程和React结合的文章。

- [Master the JavaScript Interview: What is Functional Programming?](#)
- [The Rise and Fall and Rise of Functional Programming \(Composing Software\)](#)
- [Functional UI and Components as Higher Order Functions](#)
- [Functional JavaScript: Reverse-Engineering the Hype](#)
- [Some Thoughts on Function Components in React](#)

- 设计相关。接下来是学习一些React的设计模式。 [React Pattern](#) 是一个不错的学习React模式的地方。除此之外, 还有如下的一些不错的文章也会对你很有帮助的。
 - [React Higher Order Components in depth](#)
 - [Presentational and Container Components](#)
 - [Controlled and uncontrolled form inputs in React don't have to be complicated](#)

- [Function as Child Components](#)
- [Writing Scalable React Apps with the Component Folder Pattern](#)
- [Reusable Web Application Strategies](#)
- [Characteristics of an Ideal React Architecture](#)

- [实践和经验](#)

还有一些不错的实践和经验。

- [9 things every React.js beginner should know](#)
- [Best practices for building large React applications](#)
- [Clean Code vs. Dirty Code: React Best Practices](#)
- [How to become a more productive React Developer](#)
- [8 Key React Component Decisions](#)

资源列表

最后就是React的资源列表。

- [Awesome React](#)，这是一些React相关资源的列表，很大很全。
- [React/Redux Links](#)，这也是React相关的资源列表，与上面不一样的是，这个列表主要收集了大量的文章，其中讲述了很多React知识和技术，比上面的列表好很多。
- [React Rocks](#)，这个网站主要收集各种React的组件示例，可以让你大开眼界。

Vue.js框架

Vue可能是一个更符合前端工程师习惯的框架。不像React.js那样使用函数式编程方式，是后端程序员思路。

- 通过文章 [“Why 43% of Front-End Developers want to learn Vue.js”](#)，你可以看出其编程方式和React是大相径庭的，符合传统的前端开发的思维方式。
- 通过文章 [Replacing JQuery With Vue.js: No Build Step Necessary](#)，我们可以看到，从JQuery是可以平滑过度到Vue的。
- 另外，我们可以通过 [“10 things I love about Vue”](#)，了解Vue的一些比较优秀的特性。

最令人高兴的是，Vue的作者是我的朋友尤雨溪（Evan You），最近一次对他的采访 [“Vue on 2018 - Interview with Evan You”](#) 当中有很多故事以及对Vue的展望。（注意：Vue是完全由其支持者和用户资助的，这意味着它更接近社区而不受大公司的控制。）

要学习Vue并不难，我认为上官网看文档（[Vue 官方文档（中文版）](#)），照着搞一搞就可以很快上手了。[Vue.js screencasts](#) 是一个很不错的英文视频教程。

另外，推荐 [新手向：Vue 2.0的建议学习顺序](#)，这是Vue作者写的，所以有特殊意义。

Vue的确比较简单，有Web开发经验的人上手也比较快，所以这里也不会像React那样给出很多的资料。下面是一些我觉得还不错的内容，推荐给你。

- [How not to Vue](#)，任何技术都有坑，了解Vue的短板，你就能扬长避短，就能用得更好。
- [Vue.js Component Communication Patterns](#)
- [4 AJAX Patterns For Vue.js Apps](#)
- [How To \(Safely\) Use A JQuery Plugin With Vue.js](#)
- [7 Ways To Define A Component Template in Vue.js](#)
- [Use Any Javascript Library With Vue.js](#)
- [Dynamic and async components made easy with Vue.js](#)

当然，最后一定还有 [Awesome Vue](#)，Vue.js里最为巨大最为优秀的资源列表。

小结

总结一下今天的内容。我先介绍的是前端性能优化方面的内容，推荐了图书、最佳实践性的文档、案例，以及一些在线性能测试分析工具。随后重点讲述了React和Vue两大前端框架，给出了大量的文章、教程和相关资源列表。我认为，React.js使用函数式编程方式，更加符合后端程序员的思路，而Vue是更符合前端工程师习惯的框架。因此，两者比较起来，Vue会更容易上手一些。

下篇文章，我们将讲述前端工程师的一个基本功——UI/UX设计。敬请期待。

下面是《程序员练级攻略（2018）》系列文章的目录（持续更新中）。

- [开篇词](#)
- [入门篇](#)
 - [零基础启蒙](#)
 - [正式入门](#)
- [修养篇](#)
 - [程序员修养](#)
- [专业基础篇](#)
 - [编程语言](#)
 - [理论学科](#)
 - [系统知识](#)
- [软件设计篇](#)
 - [软件设计](#)
- [高手成长篇](#)
 - [Linux系统、内存和网络（系统底层知识）](#)
 - [异步I/O模型和Lock-Free编程（系统底层知识）](#)
 - [Java底层知识](#)
 - [数据库](#)

- [分布式架构入门（分布式架构）](#)
- [分布式架构经典图书和论文（分布式架构）](#)
- [分布式架构工程设计\(分布式架构\)](#)
- [微服务](#)
- [分布式架构工程设计](#)
- [容器化和自动化运维](#)
- [机器学习 and 人工智能](#)
- [前端基础和底层原理（前端方向）](#)
- [前端性能优化和框架（前端方向）](#)
- [UI/UX设计（前端方向）](#)
-



| | |
|---|------------|
| 李奋斗 | 2018-08-02 |
| 皓叔在实际工作中是玩全栈吗？感觉积累了好多知识 | |
| paul.yang | 2018-08-02 |
| 耗哥啥时候出Android iOS练级攻略,爆栈程序员. | |
| 曾经的十字镐 | 2018-08-02 |
| 我现在作为一个交易所的架构师，技术一直停滞不前，英文水平差限制了我的发展，看来需要报一个英语培训班，好好学习了 | |
| 夏洛克的救赎 | 2018-08-02 |
| 大数据呢 | |
| 木子李 | 2018-08-02 |
| 感觉机器学习方面的专栏内容比较少，只写了一篇，有点小遗憾，不知道大数据会不会也有攻略 | |
| 阮超 | 2018-08-02 |
| 大佬，求安卓和IOS的技术学习路线 | |
| oilbeater@alauda | 2018-08-02 |
| 应该推荐一下 typescript，前端工程化是大趋势，后端过来的人表示 angular 的工程化还是最好的 | |
| dancer | 2018-08-02 |
| 如果程序员运动会有十项全能这个项目，我买皓叔夺冠～ | |
| Silence-0°C | 2018-08-02 |
| 为啥没有后端的内容呢？ | |
| ifif | 2018-08-02 |
| 期待已久，终于来了。💎💎 | |
| 白云 | 2018-08-02 |
| React 可以写类型安全的代码，vue 的魔法字符串就不行了 | |
| 修炼 | 2018-08-02 |
| 前端还是别写了。。。。 | |
| 作者回复 | 2018-08-02 |
| 为什么？ | |

