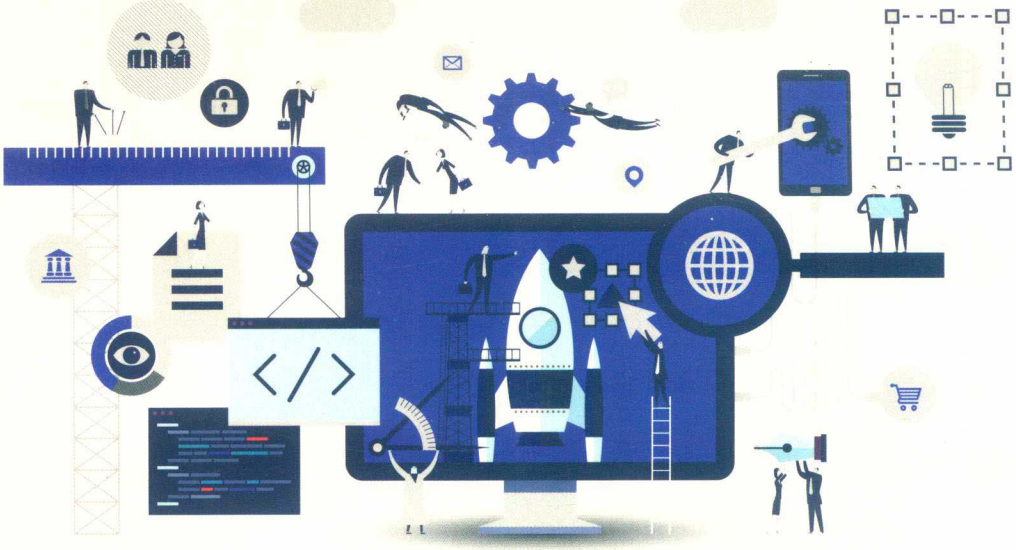
· 前端工程化系列 ·

Broadview®
www.broadview.com.cn



前端技术架构与工程

周俊鹏 / 著

业务是核心出发点 | 架构聚焦于代码 | 工程聚焦于流程

本书将从宏观角度讲解前端技术架构与工程的各项关注点，告诉读者如何实现高可用、高性能、可扩展的前端架构，以及高效、规范、有序的工作流程

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

· 前端工程化系列 ·

前端技术架构与工程

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

明确业务、架构与工程三者之间的关系是研究前端技术架构和工程化的基本前提：业务为核心出发点，架构聚焦于代码，工程聚焦于流程。在此基础之上，本书进一步剖析并明确了架构与工程的子集与超集的关系。本书从架构的角度分析了一个完整 Web 项目在前端以及前后端协作层面需要考虑的各项技术要点和解决方案，在业务需求以及应用质量得到保障的基础之上，进一步从工程的角度分析迭代流程中可能阻碍提高工作效率的关键环节和因素，并讲解了如何通过技术手段提升团队的规范性和生产效率。

本书的大部分内容需要读者对计算机操作系统、浏览器原理以及 Web 前后端工作原理有一定程度的理解。本书适合前端从业经历较丰富并且对前后端协作流程有深度体验的读者，以及对前端技术架构和工程化感兴趣的测试和运维人员阅读使用。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目（CIP）数据

前端技术架构与工程/周俊鹏著. —北京：电子工业出版社，2020.1
（前端工程化系列）
ISBN 978-7-121-38061-7

Ⅰ.①前… Ⅱ.①周… Ⅲ.①网页制作工具 Ⅳ.①TP393.092.2

中国版本图书馆 CIP 数据核字（2019）第 271489 号

责任编辑：付　睿
印　　刷：三河市双峰印刷装订有限公司
装　　订：三河市双峰印刷装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　　印张：15.75　　字数：295 千字
版　　次：2020 年 1 月第 1 版
印　　次：2020 年 6 月第 2 次印刷
定　　价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

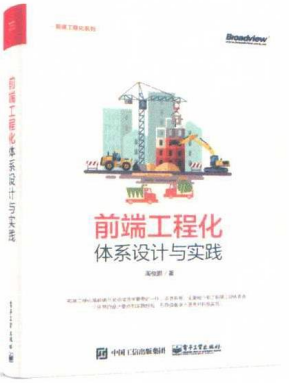
质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

此为试读，需要完整PDF请访问：www.ertongbook.com

周俊鹏

前端工程师，现就职于腾讯，曾就职于优酷、搜狗等互联网企业。拥有多年一线前端开发和架构设计经验，做过大众的Web网站，也做过小众的SVG Charts；做过宏观到跨栈的前端工程化，也做过微观到像素的WebGL编程。目前专注于前端图形编程、工程化和Web应用层架构。



《前端工程化系列》丛书

本丛书包含两本书：《前端工程化：体系设计与实践》和《前端技术架构与工程》。前者着重讲述辅助性质的工程体系设计和实践过程，而后者则从宏观角度讲解前端技术架构和工程的各项关注点。与前者不同的是，后者对于前端工程服务体系的讲解侧重方法论和指导思想，并未深入具体实现的每一行代码。如果在将此部分理念应用于实践的过程中遇到问题，或许同时参阅两本书能够获取答案。

前　言

在工程化思维进入前端领域的几年内，前端社区一直在试图给前端工程化下一个精确的定义。人们喜欢从历史讲起，然后将视角延伸到时代背景下的宏观技术理论，最后聚焦到自身业务的工程实践。在这个过程中，从第一步过渡到第二步的历程中能够提取出具有普适性的指导思想，宽泛地讲就是规范化、工具化、自动化；而发展到第三步的时候难免会带入一些只适合自身业务特征的方法论和实践模式。一部分声音认为这些狭隘的理论属于功能解决方案，脱离了前端工程化的范畴，但其实这恰恰是工程最基本且最核心的出发点：一切以业务为基准。

在对前端工程化进行讨论和研究之前，一定要摆脱“前端工程化是一种新技术”的错误认识，前端是软件开发的一个细化分支，前端工程化本质上是软件工程理论在前端范畴内的具象实践。方法、工具和过程是软件工程也是前端工程化的三要素，方法面向编码和功能解决方案；工具的根本目的是降低时间成本以提高效率；过程追求高效、有序的工作流程，它是一个抽象的概念，具体到实施中则是方法和工具的综合体。一切编码方案均是为了解决业务的功能需求，在此基础之上以分治和聚合为基本原则设计合理的软件架构，最后进一步规范工作流程和产品发布策略，这便是工程化的理论模型。总结为一句话：以业务为出发点，架构聚焦于代码，工程聚焦于流程。

内容概览

本书所有内容遵循一个基本出发点：业务是架构和工程的核心。

第 1 章围绕上述基本出发点讲解前端自诞生至今在不同时代背景下的意义以及业务特征，进而引申出架构和工程的子集与超集的关系。然后在此基础之上探讨前端工程化在架构以及架构之外的困难之处和核心关注点。

第 2 章在前端单一的编程语言基础上，从技术选型、辅助工具、实现方案以及架构模式的角度思考和探索如何更合理地应用 HTML、JavaScript、CSS。

第 3 章聚焦于编码和架构这些“硬实力”之外的“软实力”——技术规范。除去所有编程领域的一些普适规则（比如技术选型、目录规范和命名规范），由于前端的特殊性，有些技术规范并不仅仅是为了提高代码的可维护性，它们还兼具了架构层面上的设计考虑，比如 JavaScript 在性能与易读性之间的抉择、CSS 的编程范式等。

第 4 章介绍实现前端组件化的 Web Components 技术，以及通过开发合理的工具打造更友好的编码方式。然后从生命周期和宿主环境两个角度介绍前端组件设计模式。

第 5 章描述了两种常见的前后端分离架构模型：SPA 和同构编程。前者是最普遍、实施成本最低、最极端的分离模式，但对 SEO 的弱支持导致其并非适用于所有产品类型；而后者则与前者相反，对 SEO 的良好支持背后是昂贵的实施成本和学习成本，同时对技术选型有一定的限制。探讨两者的综合优劣性时必须结合具体的产品需求。

第 6 章以前端应用的性能评估模型为前提，剖析加载和执行过程中影响性能的各项因素（包括网络、渲染和内存管理）以及对应的优化策略。最后探索综合运用 Web Worker、WebAssembly 甚至 WebGPU，以发挥出浏览器的极限运算能力。

第 7 章将一个完整的迭代流程拆分为开发、测试、部署和发布，然后讨论在传统开发模式下前端如何从开发、测试和运维层面进行工程优化，其中包含高效的工具、合理的规范以及严谨的制度。

第 8 章在第 7 章描述的前端工程服务体系的前提之下，探讨在目前的技术背景下，前端工程化在本地化的基础上进一步演进的方向（DevOps）以及目标（持续交付）。最后，在本书的末尾对继 AJAX 和 Node.js 之后可能引起第三次前端革命的 Serverless 进行了展望。

读者对象

本书内容并非告诉读者如何实现具体的业务需求，所举示例也只是为了辅助理解相关内容背后的思想和理念。换句话说，本书不是教读者怎么编码的，而是从宏观角度讲解了如何实现高可用、高性能、可扩展的软件架构，以及高效、规范、有序的工作流程，所以

本书的主要目标读者是有多年一线编码经验、充分理解 Web 整体架构并且具有一定的团队管理和多人协作经验的资深前端开发者和技术经理。

资源链接

本书所有示例的源代码均可以访问 http://www.broadview.com.cn/38061 进行下载。书中提供的额外参考资料也可从上述网站下载，如正文中标有参见“链接 1”“链接 2”等字样时，即可从上述网站下载的“参考资料.pdf”文件中进行查询。

致谢

本书的写作和出版得到了许多同事、朋友和家人的支持和帮助。本书中的很多技术细节得到了我的同事和领导的指正。感谢电子工业出版社的付睿编辑和审校人员对本书的策划和编审，他们是本书出版背后的重要功臣。此外，本书的写作占用了我很多业余时间，感谢我的妻子刘女士在此期间对我的理解、包容和支持。

谨以此书献给我的妻子和父母。

《前端工程化系列》丛书

本书是《前端工程化系列》丛书之一，从宏观角度讲解前端技术架构和工程的各项关注点。与本系列另一本图书《前端工程化：体系设计与实践》不同的是，本书对于前端工程服务体系的讲解侧重方法论和指导思想，并未深入具体实现的每一行代码。如果在将此部分理念应用于实践的过程中遇到问题，或许同时参阅两本书能够获取更全面的答案。

帮助与支持

如果你在阅读本书的过程中有任何问题，可以发送邮件到我的个人邮箱 zjp0432@163.com。

读者服务

微信扫码回复：38061

- 获取博文视点学院 20 元付费内容抵扣券
- 获取本书的配套代码资源
- 获取更多技术专家分享的视频与学习资源
- 加入读者交流群，与更多读者互动

目　录

此为试读，需要完整PDF请访问：www.ertongbook.com

第 1 章
前端工程化

进化本身是生物体与环境之间持续不断的信息交换的具体表现。

——摘自《信息简史》

在桌面软件盛行的 20 世纪 90 年代中期，由Web技术开发的电子商务服务平台Viaweb在当时被认为是另类的、难以成功的。时间最终证明了Paul Graham和Robert Morris[1]眼光的前瞻性，1998 年Viaweb被雅虎以近 5000 万美元的价格收购，成为当时全球最大的电商平台之一：Yahoo! Store。站在当前的时间节点，以现代人的眼光回顾旧时代，你可能会觉得当时的人目光短浅且顽固不化。但是如果剥去时代赋予的知识和眼界回到 1995 年，可能绝大多数人都会站在Viaweb两位创始人的对立面。

想象这样的场景：给你一台装有 Windows 98 操作系统的电脑，使用 IE6 浏览器打开任意 10 个网站，将能够顺利运行的网站个数写下来，然后看结果是否达到了及格线。虽然这个场景在目前没有任何现实意义，因为 Web 技术是随着历史的车轮不断进化的，用老旧的浏览器运行现代的网站是自讨苦吃。而之所以讨论这个场景是为了提醒大家：现代网站具

1 Viaweb 的两位创始人。

有丰富的表现力、流畅的交互操作和便捷的功能的根本原因是，计算机技术和 Web 开发技术的不断进化以及相关生态的逐步完善。如果将这些技术和生态沿着历史的轨迹倒退到二十几年前，你是否会在一个没有 AJAX、没有 CSS3，甚至最高级的浏览器都不如 IE6 的年代，选择使用 Web 开发一个电商平台呢？是否会在一个对于 Web 网站的定位只是将报纸搬上电脑屏幕的时代去讨论 Web 网站的架构和工程化呢？

架构可以渗透到任何一个细微之处，即使再简单不过的一个静态网页也存在架构，只不过过于简单，没有讨论的价值。当业务的复杂度提升到必须由更复杂的技术架构承载时，对架构的研究才有了意义。工程化同样如此。换句话说，任何针对架构和工程的讨论和研究必须以业务为根本出发点。

Web 技术支撑着网站丰富的业务，同时网站的业务类型也受限于 Web 技术的时代局限性。所以，本书讨论的所有关于前端业务、技术、架构和工程的内容，均基于当前时间节点所处时代赋予前端工程师的时代意义。

1.1 前端的时代意义

如果以 2005 年AJAX的诞生[1]为起点，在迄今为止的十几年时间里，针对前端工程师定位的探索和争论从来没有停止过。前端工程师该做什么不该做什么？发展方向是什么？扩展技能是什么？即使在今天（2019 年），这些问题也没有准确的答案。最早一批的专职前端工程师大多是由Web服务端开发者或者UI/UE工程师转变而来的，这两种出身不同的前端工程师分别代表了两个典型的发展方向：Web服务端开发出身的前端工程师普遍偏向于“服务端+前端”，以逻辑见长；UI/UE出身的前端工程师普遍偏向于“设计+前端”，以用户体验见长。

在 PC 时代，“设计+前端”的模式相较占优。当时智能手机尚未普及，绝大多数的线上业务是以 PC 浏览器为载体的，并且 Web 网站的交互逻辑普遍比较简单，以展示为主。前端工程师的主要工作集中于 UI 的静态表现和动画上。设计师和产品经理在 UI 和动画上费尽了脑筋，如果负责开发的前端工程师能够对设计有所理解和掌握便事半功倍了。所以

1 Jesse James Garrett 于 2005 年 2 月发表了文章 *AJAX: A New Approach to Web Applications*，AJAX 在其中被正式命名，并沿用至今。

PC 时代的前端工程师结对编程的伙伴通常是一名 UI 或者 UE 设计师，目标是开发出高度还原设计稿的 UI 和动画。

“PC 时代”是一个比较模糊的概念，并没有很准确的划分界限。一种较普遍的观点是，将 2010 年之前定义为“PC 时代”。这是由于自 2010 年起，Android 和 iOS 系统市场份额急剧上升，智能手机崛起，分流了大量 PC 用户。

之后智能手机崛起，彻底改变了用户的使用习惯和思维方式。今天人们看到一家陌生而特别的小店，第一反应是询问店家有没有专属的 App 或者微信公众号而不是网站。在这样的时代背景下，前端工程师的工作重心也发生了倾斜。历史的前进不仅带来了智能手机，也推动着设备硬件性能和浏览器的进化，同时还有 HTML5。前端工程师可以踏入视频、语音、VR 等新技术领域。从理论上讲，这个时代下的硬件和技术条件能够支撑更精致的 UI 和动画，但是 PC 时代火热的“设计+前端”模式反而逐渐式微，甚至有一段时间出现了设计的“返古潮”，即追求极简的设计风格，颇有些类似于文艺复兴时期建筑风格的“仿古潮”。

14 世纪文艺复兴时期，意大利的建筑风格出现了一股“仿古潮”。这股风潮反对代表着神权的哥特式建筑，学习以古希腊和古罗马为代表的古典建筑风格，核心理念是体现“自然”与“和谐”。

哥特式建筑代表——德国科隆大教堂
——图片引自维基百科

古希腊建筑代表——帕特农神庙
——图片引自维基百科

前端工程师的工作重点逐渐偏向复杂的交互逻辑和架构，同时技能和发展方向也发生了改变。在当前的时间节点，前端工程师的发展方向被重新定义，较普遍的有如下两种：

- 服务端 + Web 前端
- App 前端 + Web 前端

Node.js 是革命性的，语言的亲和性可以令前端工程师以相对较小的成本涉足服务端开发。以此为起点逐渐发展出了“服务端+Web 前端”的所谓“大前端”模式，是目前占比较高的一种趋势。与之形成对比的是“App 前端+Web 前端”，或者也可以称为“泛前端”，代表性技术是 React Native、Flutter 以及后续推出的各种类似技术和框架。前端工程师可以使用熟悉的 JavaScript、CSS、HTML 或者相似的技术（比如开发微信小程序和支付宝小程序所使用的技术）开发原生或混合移动应用。“泛前端”已经脱离了传统意义上的前端范畴，这种模式目前仍然处于探索阶段，尚未成为主流。与其相关的技术也没有发展成熟，未来如何发展尚不可知，但也不失为一个有价值的研究方向。本书所讨论的内容仍然是围绕传统 Web 领域的前端以及“大前端”展开的。

“大前端”模式下的前端工程师负责一部分 Web 服务端的开发工作，但是具体负责哪一部分或者说纵深的程度如何却是众说纷纭。有一种声音提倡所谓的“全栈开发”，即前端工程师负责开发客户端逻辑的同时，将 Web 服务端、数据库管理等工作一并包揽。这是一个美好的愿望，但实际上一个人很难做到从前到后面面俱到。如果仅是类似个人博客这种体量小、架构简单且对稳定性要求不高的网站，“全栈”没有任何问题。但是对于大型 Web 应用程序来说，每一个环节都必须做到尽善尽美。术业有专攻，专业的事情最好交给专业的人去做。另一种声音认为前端工程师可以负责一部分 Web 服务端工作，但是仅限于渲染。比如，使用 Node.js 架设中间渲染层，将前后端模板统一，比较典型的案例是淘宝的 Midway Framework。这种模式是实现前后端分离的一种探索，且目前已经得到业界一定的认可和普及。当然，随着相关技术的迭代和进化，其真实的价值和正误如何还有待评定，但就目前的时间节点来说，“大前端”模式已经成为一种主流趋势。

业务逻辑与交互逻辑

Rockford Lhotka 在 *Expert C# Business Objects* 一书中将应用程序的架构分为 5 层，由

此为试读，需要完整PDF请访问：www.ertongbook.com