Broadview®
www.broadview.com.cn

前端工程化
体系设计与实践

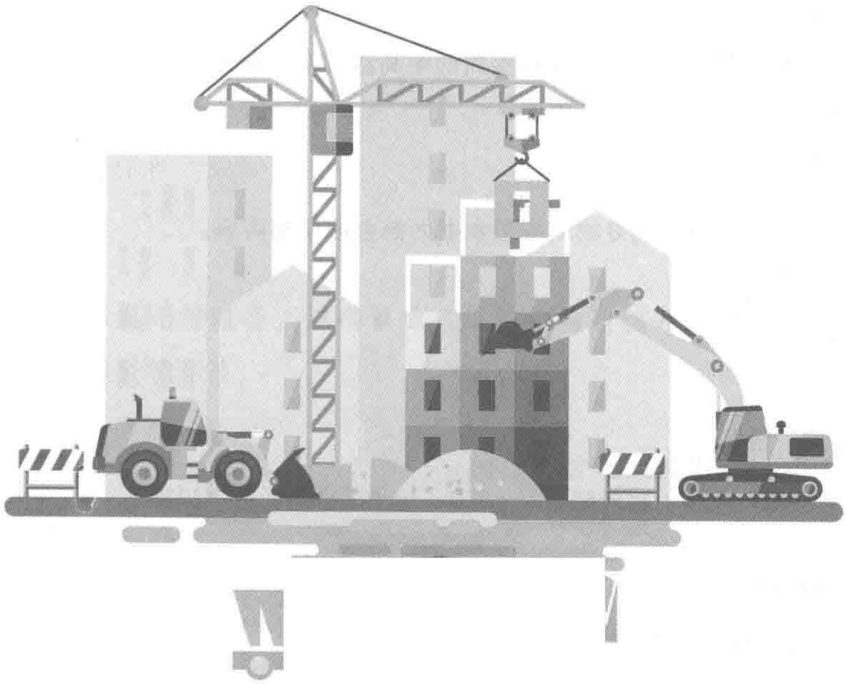
周俊鹏 / 著

前端工程化是前端开发领域非常重要的一环。本书系统、全面地介绍了前端工程体系各个环节的设计要点和实践经验，引导读者深入思考并积极实践。

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
体系设计与实践

· 前端工程化系列 ·



前端工程化
体系设计与实践

周俊鹏 / 著

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

内 容 简 介

前端工程化包含一系列规范和流程，其可提升前端工程师的工作效率，加快Web开发迭代速度，是现在前端开发领域中非常重要的一环。本书系统、全面地介绍了前端工程体系的各个环节，包括设计要点和实践经验。全书分为7章，分别是前端工程简史、脚手架、构建、本地开发服务器、部署、工作流、前端工程化的未来。

本书适合对前端工程化有一定理解和实践的中高级前端工程师阅读，同样适合对前端工程化感兴趣的服务器端开发者以及运维人员阅读。

图书在版编目（CIP）数据

前端工程化：体系设计与实践 / 周俊鹏著. —北京：电子工业出版社，2018.1
（前端工程化系列）
ISBN 978-7-121-33090-2

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(2017)第 286740 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：14　字数：234 千字
版　　次：2018 年 1 月第 1 版
印　　次：2018 年 4 月第 3 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。

推荐序

技术之外

前端工程体系是一种服务，以项目迭代过程中的前端开发为主要服务对象，涉及开发、构建、部署等环节。

——摘自《前端工程化：体系设计与实践》

阿里的玉伯曾经问过我一个问题：前端该不该碰业务？具体一点地说，就是前端要不要了解后端的业务逻辑，甚至将部分这样的逻辑与规则放在前端来处理与实现。我当时思考了片刻，给玉伯的建议是：前端还是不要碰业务逻辑，围绕着交互做就好了。

事实上这个问题的答案有很多，不同的场景下也可以各有权衡，所以上面的答案也并非标准答案。但我在这里提及这件事情的原因是：这个问题的前设、背景与分析过程，是技术无关的。显而易见，我们并没有讨论哪一种框架来解决何种技术问题，又或者在技术上如何做前后端分离。我们是在讨论一个根本上的工程协作问题：谁，该做什么？这个问题的关键点，就是“什么是领域划分的事实依据”。

前端的工程化，事实上还处在一个原始阶段。我们如今之所视，可以一言以蔽之：或在对语言内在功能特性的补充，或在对其外在组织能力的补充。这些种种补充，尽是在工程体系的“工具”这一隅上做的功夫。可以预见的是，在前端工程这个体系上前行，必然面临的问题是过程的优化和方法论的建立。然而如今前端在这

些大的、根本性的问题上并没有任何触及，甚至连上面这样的“领域划分”问题都没有被认真地讨论过。

这些问题，也都如同开始的那个问题一样，是在技术之外。

所幸作者是意识到了这一点的。他在本书中将“前端工程体系”定义成一种服务，而非一种工程模型。从作者的定义来看，这个体系是可资实用的一种工具——可讨论、可实现，以及可以演化与重构，并遵循这些服务的设计原则、问题场景以及应用的约束。在我看来，这些内容才是书中的闪光点。

除此之外，本书还详细地讨论了其中有关脚手架、构建过程和本地工程化服务等现实中的工程实践所得，并为这些实践构画了一个参考模型。这使得本书提供了大量前端工程师可借鉴、参考并投之于生产实作的最佳实践。我想，作为结果，这些实践的优劣得失尚待时间验证，而作者在这一过程中的分析与观点，也可待业界指正评点。

而我所愿者，亦在读者能与我一道，在技术之外多做一点点观察。

周爱民

2017.11

前言

前端工程师这一岗位最初被独立分化出来专注于网页样式（CSS）的制作，目的是为了令 Web 开发者将更多的精力投入负责的业务逻辑中。然而随着 Web 技术的发展以及 PC、移动智能终端设备性能和功能的提升，用户对于网站的需求也不断增加。市场的需求促进技术的革新，对于前端工程师的要求早已不仅仅是编写 CSS 了。资源的多样性和逻辑的复杂性一度令前端开发工作异常烦琐且难以维护，工作效率的降低直接导致 Web 产品的迭代速度变慢，前端工程化便是在此时代背景下应运而生的。

事实上，前端工程化目前的形态和生态仍然处于非常原始的阶段。每个团队甚至每个人由于存在研究领域（比如业务层和框架层）和业务类型（比如 Google Map 与淘宝）的差异，从而对前端工程化有不同的需求和定位。本书将前端工程化解读为一系列规范和流程的集合，它不是一个框架或者工具，聚焦的不是某个垂直的研究领域或者特殊的业务类型，而是一种可演化、可扩展的服务，服务的目标是解决前端开发以及前后端协作开发过程中的难点和痛点问题，涵盖项目的起始、开发、测试以及部署环节。工具是前端工程化的实现媒介，规范是工程化的指导方针，工作流程是工程化的外在表现形式以及约束规范的载体。

本书通过解析一个 Web 项目迭代过程中前端开发者面临的诸多问题，从工程化的角度给出对应的解决方案，最终将各个环节串联为完整的工作流。希望读者通过阅读本书可以对前端工程化要解决的问题有大致的了解，从而能够对读者自行实现工程化方案有所帮助。

目标读者

本书的主要目标读者是对前端工程化有一定理解和实践的中高级前端工程师，同样适用于对前端工程化感兴趣的服务器端开发者以及运维人员。本书假设读者熟悉 Web 站点的基本工作原理，尤其是前端与服务器端之间的协作流程，并且对 HTTP 协议、异步通信、模块化等知识有深入的理解。

示例代码

本书选取了一个简易的前端工程化解决方案 Boi 作为示例，这并不是一个完整形态的解决方案，但是它的许多理念可以作为论证本书观点的参考。读者可以从 GitHub 上获取其源码：https://github.com/boijs/boi。

内容概览

本书第 1 章以前端工程师从无到有直至发展至今的历程作为后续内容的起始。从历史中我们提炼出前端开发人员在一个 Web 项目迭代周期各个阶段面临的诸多问题，这些问题是前端工程化诞生的催化剂，也是指导工程方案设计的本源。之后，我们会按照 Web 项目从起始到发布的流程分别介绍前端工程化在各个阶段的需求和功能设计，比如脚手架在项目初期减少了重复的体力操作并且降低了业务框架学习成本；构建系统从编程语言、优化和部署 3 个角度解决了前端开发语言内在的缺陷以及由宿主客户端特性引起的开发和生产环境之间的差异性；本地开发服务器提供了前后端并行开发的平台；部署功能权衡速度、协作和安全，把控着 Web 产品上线前的最后一道关卡。最后将这些功能模块合理地串联为完整的工作流，便是前端工程化的完整外在形态。

前端工程师的定位在不同的年代甚至不同的团队中存在着巨大的差异，即使仅以目前的时间节点为标准也难以给前端工程师一个绝对明确的定义。岗位职责的变化促进了工程体系的演进，所以本书在最后的章节中阐述了一些对前端工程师未来定位的思考，同时探讨了与之对应的前端工程体系的演进形式。

以下是分章节介绍：

- **第 1 章　前端工程简史**　讲述前端工程师的发展史、在团队中的定位，以及前后端分离和前端工程化的进化历程与基本形态。
- **第 2 章　脚手架**　讲述作为前端项目起始阶段取代烦琐人工操作的脚手架必须具备的要素以及本质，通过剖析目前市面上的经典案例讲解实现脚手架过程中需要考虑的要点以及如何集成 Yeoman 到工程化方案中。
- **第 3 章　构建**　讲述构建系统面临的问题以及对应的解决方案。构建是前端工程体系中功能最多、最复杂的模块，也是串联本地开发服务器、部署的关键，是实现工作流的核心模块。
- **第 4 章　本地开发服务器**　讲述如何以 Mock 服务实现前后端并行开发，以及配合动态构建进一步提升前端工程师的开发效率。
- **第 5 章　部署**　讲述部署功能如何权衡速度、协作和安全 3 个重要原则，以及前端静态资源特殊的部署策略。
- **第 6 章　工作流**　讲述如何将既有的功能串联成完整的工作流。以速度见长的本地工作流和注重严谨的云平台工作流，两者各有优劣，适用于不同需求和不同规模的团队。
- **第 7 章　前端工程化的未来**　讲述前端工程师如何选择进阶的方向以便适应未来的变化。前端工程化是服务于前端开发的，前端工程师定位的改变必然会引起工程化方案的调整。本章通过分析未来工程化不变和可变的方面，探讨前端工程化未来的表现形式。

“前端工程化系列”丛书

本书是“前端工程化系列”丛书之一，着重讲述辅助性质的工程体系设计和实践过程。前端工程化可以简单地理解为前端架构与工程体系的综合体，两者相辅相成。本系列丛书的后续作品将从综合的角度深层剖析架构与体系之间的关联及融合，讲述如何从宏观的角度打造合理的前端工程化生态。感兴趣的读者可以关注本系列丛书的相关动态。

联系作者

如果您在阅读过程中有任何问题，可以发送邮件到作者的个人邮箱：zjp0432@163.com。

致谢

感谢我的同事和领导在我创作本书期间给予的建议和支持。特别感谢我曾经的技术领导元亮，在与他共事期间我于前端工程领域的探索和研究得到了充分的空间和资源。

感谢电子工业出版社博文视点的编辑付睿，她在编辑和审校本书期间提出了宝贵的意见。

最后，感谢我的朋友、父母以及妻子刘女士在我创作本书期间给予的空间和支持。

读者服务

轻松注册成为博文视点社区用户（www.broadview.com.cn），扫码直达本书页面。

- 提交勘误：您对书中内容的修改意见可在 提交勘误 处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- 交流互动：在页面下方 读者评论 处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：http://www.broadview.com.cn/33090

目录

第1章

前端工程简史

前端工程化这个概念在近两年被广泛地提及和讨论，究其原因，是前端工程师所负责的客户端功能逻辑在不断复杂化。如果说互联网时代是前端工程师的舞台可能有些夸大其词，但前端工程师绝对撑起了互联网应用开发的“半壁江山”。传统网站、手机应用、桌面应用、微信小程序等，前端工程师已经不是几年前被谑称的“切图仔”了。以往的“写 demo，套模板”模式已经严重拖累了前端开发以及整体团队的开发效率。在这样的时代背景下，前端工程化便应运而生了。

在本章中，我们首先讨论当前市场环境下对前端工程师的技能要求是什么，以此为前提探讨前端开发以及前后端协作开发中有哪些问题需要从工程化的角度解决。随后，沿着前端工程化从无到有的进化历程，了解前端工程化带给前端开发模式的改革和效率的提升，从而总结出前端工程化应有的形态。最后结合作者的经验，讲述如何以 Node.js 为底层平台、以 webpack 为构建体系核心打造一套完整的前端工程解决方案。

本章主要包括以下内容。

- 前端工程师的基本素养。
- Node.js 带给前端的机遇和挑战。
- 前后端分离的必要性和基本原则。
- 前端工程化的进化历程和基本模式。
- 最流行的构建工具之一： webpack。

1.1 前端工程师的基本素养

在讨论前端工程化之前，首先需要弄清楚什么是前端工程师。前端工程师是被人误解的工作很简单的“切图仔”，还是包揽客户端和中间层的“大前端”呢？招聘市场上有大量的公司对前端工程师求贤若渴，但同时求职市场上也有大量的前端工程师在“求职若渴”。造成这种两难局面的原因是，用人单位与求职者对前端工程师的技能需求以及定位存在差异。

应该怎么定位前端工程师这个岗位？下面让我们从前端的发展历史中找出答案。

1.1.1 前端工程师的发展历史

1990 年，Tim Berners Lee 发明了世界上第一个网页浏览器 WorldWideWeb。1995 年，Brendan Eich 只用了 10 天便完成了第 1 版网页脚本语言（也就是目前我们所熟知的 JavaScript）的设计。在网络条件与计算机设备比较落后的年代，网页基本是静态的。对网页脚本语言功能的最初设想仅仅是能够在浏览器中完成一些简单的校验，比如表单验证。所以网页脚本语言的特点是：功能简单、语法简洁、易学习、易部署。那个年代的 Web 应用是重服务器端、轻客户端的模式，Web 开发人员以服务器端开发为主，同时兼顾浏览器端，没有所谓的前端工程师。

2005 年，AJAX 技术的问世令静态的网页“动”了起来，异步请求和局部刷新彻底改变了网页的交互模式。同时，网络速度与个人计算机的普及给网站带来了更多用户，用户对网站的需求也越来越多。需求与技术的同步增长让早期的重服务器端、轻客户端的天平向客户端有所倾斜，也就是从那个时候开始出现了第一批专职的前端工程师。这批前端工程师相对于服务器端工程师的优势主要体现在对交互与 UI 的敏感度和专业度上。所以第一批前端工程师中有很大一部分是设计师出身，导致前端工程师们有了一个很不相称的称谓：美工。但不可否认的是，第一批前端工程师主要负责的是 CSS 与 HTML 的开发，虽然有了 AJAX 技术，但受限于 JavaScript 引擎的性能，浏览器端的功能逻辑仍然十分简单。

2008 年，Google 推出了全新的 JavaScript 引擎 V8，采用 JIT（实时编译）技术解释编译 JavaScript 代码，大大提高了 JavaScript 的运行性能。随后，Netscape 公司的 SpiderMonkey 和苹果公司的 JavaScriptCore 也紧随 V8，加入了 JavaScript 引擎的性能追逐战。JavaScript 引擎性能的提升让许多早期不能在浏览器端实现的功能得以实现，浏览器能够承载几千行甚至几万行的逻辑，Web 应用服务器端与客户端的天平再次向客户端一方发生倾斜。业内开始提倡 REST（Representational State Transfer，具象状态传输）风格的 Web 服务 API 与 SPA（Single Page Application，单页应用）风格的客户端。前端工程师承担起了客户端的交互、UI 和逻辑的开发，工作职责进一步加重。

2009 年，Node.js 的问世在前端界引发了轩然大波。Node.js 将 JavaScript 语言带到了服务器端开发领域，截止到目前，业内已经有很多公司将 Node.js 应用到企业级产品中。虽然 Node.js 仍然没有像 PHP、Java 等传统服务器端语言一样普及，但由它引发的“大前端”模式已经在 Web 开发领域中蔓延。Node.js 对前端生态的促进，以及对同构开发的支持是 PHP、Java 等语言远不能比及的，本书所探讨的前端工程方案便是以 Node.js 为底层平台实现的。“大前端”模式下的前端工程师跨越了之前浏览器与服务器端之间看似难以逾越的鸿沟，踏入了 Web 服务器端开发领域。

1.1.2　前端工程师的技能栈

从最初的重交互/UI，轻 JavaScript 的开发模式，到交互、UI、逻辑一把抓，再到“大前端”的服务器端、客户端全掌控，前端工程师的工作内容和工作职责不断扩宽。从前端工程师的发展历史中，我们可以总结出前端工程师的技能栈。

- **硬技能**：HTML/CSS/JavaScript。这 3 项是前端工程师从蛮荒年代发展至今从未脱离的核心技术。
- **软技能**：用户体验。用户体验是 Web 产品吸引用户的第一道菜，也是前端工程师工作产出的重点。
- **扩展技能**：Node.js。并非特指 Node.js 本身，而是 Node.js 所代表的 Web 服务器端知识。即使你不是一个“大前端”，了解 Web 产品的运行原理也