

《SPA设计与架构：理解单页面Web》

图书基本信息

书名：《SPA设计与架构：理解单页面Web应用》

13位ISBN编号：9787121300915

出版时间：2016-11

作者：【美】Emmit A. Scott

页数：304

译者：卢俊祥

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

《SPA设计与架构：理解单页面Web》

内容概要

SPA 开发技术的运用是当今Web 开发领域的热门趋势，但真正全面掌握该技术的开发者并不多。本书详尽阐述单页面Web 应用（SPA）开发技术，从SPA 构建基础入手，通过MV*、模块化编程、路由、模块间通信、服务器端交互等概念的阐述，全面介绍SPA 的设计与架构，帮助读者正确掌握SPA 开发的各方面知识要素。同时，《SPA设计与架构：理解单页面Web应用》中还讨论了SPA 的单元测试及客户端任务自动化，覆盖了从开发到部署的一系列任务，让读者在阅读完《SPA设计与架构：理解单页面Web应用》之后能够打下扎实的SPA 开发基础。

《SPA设计与架构：理解单页面Web应用》的重点是帮助读者正确、全面地掌握SPA 开发概念，这些概念都是通用的。但为了让内容更全面、具体，《SPA设计与架构：理解单页面Web应用》将通过Knockout、Backbone.js 及AngularJS 这三种不同风格的MV* 框架来进行比较性讨论，这是本书的一大特色。同时在涉及具体MV* 框架知识点时，《SPA设计与架构：理解单页面Web应用》中会提供相应介绍。书中示例丰富具体，并提供完整源代码下载。

《SPA设计与架构：理解单页面Web应用》适合前端及对SPA 技术感兴趣的开发者阅读。读者只需掌握JavaScript、HTML 和CSS 基本知识，就可以阅读。

《SPA设计与架构：理解单页面Web》

作者简介

Emmit A. Scott 是一名有17年Web应用构建经验的高级软件工程师和架构师。他为教育、银行和通信领域开发过大型应用程序。他的爱好包括阅读（特别喜欢Jim Butcher的小说）、吉他（想当年他可是一位摇滚乐手）以及尽可能多陪陪孩子。

译者简介

卢俊祥

译者，书迷；关注Web技术趋势，热衷App开发、Web开发、数据分析、架构设计以及各类编程语言；陈氏太极拳五十六式爱好者；佛禅人生，缘散缘聚。

微博：@2gua

个人网站：<http://www.2gua.info/>

知乎专栏：<https://zhuanlan.zhihu.com/guagua/>

书籍目录

第1部分 基础知识 1

1 单页面应用程序介绍 3

1.1 SPA 简述 4

1.1.1 无须刷新浏览器 7

1.1.2 表现逻辑位于客户端 7

1.1.3 服务器端事务处理 7

1.2 更进一步 8

1.2.1 以Shell 页面开始 8

1.2.2 从传统页面到视图 9

1.2.3 视图的产生 10

1.2.4 实现无刷新的视图切换 11

1.2.5 贯穿动态更新过程的流畅性 12

1.3 SPA 应用相较传统Web 应用的优势 12

1.4 温故知新 13

1.5 优秀SPA 应用的构成 15

1.5.1 组织项目 15

1.5.2 创建可维护的松耦合UI 17

1.5.3 使用JavaScript 模块 18

1.5.4 执行SPA 导航 19

1.5.5 创建视图组成与布局 19

1.5.6 模块通信 20

1.5.7 与服务器端通信 20

1.5.8 执行单元测试 20

1.5.9 客户端自动化技术 20

1.6 小结 21

2 MV* 框架介绍 22

2.1 MV* 概念 24

2.1.1 传统UI 设计模式 25

2.1.2 MV* 和浏览器环境 27

2.2 MV* 基础概念 28

2.2.1 框架 29

2.2.2 我们的MV* 项目 30

2.2.3 模型 32

2.2.4 绑定 36

2.2.5 模板 40

2.2.6 视图 44

2.3 为什么要用MV* 框架 44

2.3.1 关注分离 45

2.3.2 简化日常任务 46

2.3.3 提升生产率 47

2.3.4 标准化 47

2.3.5 可扩展性 48

2.4 框架选择 48

2.5 挑战环节 50

2.6 小结 50

3 JavaScript 模块化 52

3.1 模块概念 53

- 3.1.1 模块模式概念 53
- 3.1.2 模块结构 54
- 3.1.3 揭示模式 55
- 3.2 模块化编程的意义 56
 - 3.2.1 避免命名冲突 56
 - 3.2.2 保护代码完整性 65
 - 3.2.3 隐藏复杂性 67
 - 3.2.4 降低代码改变带来的冲击 68
 - 3.2.5 代码组织 68
 - 3.2.6 模块模式的不足 69
- 3.3 模块模式剖析 69
 - 3.3.1 可访问性控制 69
 - 3.3.2 创建公有API 70
 - 3.3.3 允许全局导入 73
 - 3.3.4 创建模块的命名空间 73
- 3.4 模块加载及依赖管理 74
 - 3.4.1 脚本加载器 74
 - 3.4.2 异步模块定义——AMD 75
 - 3.4.3 通过RequireJS 实践AMD 76
- 3.5 挑战环节 81
- 3.6 小结 81
- 第2部分 核心概念 83
- 4 单页面导航 85
 - 4.1 客户端路由器概念 86
 - 4.1.1 传统导航 86
 - 4.1.2 SPA 导航 86
 - 4.2 路由及其配置 88
 - 4.2.1 路由语法 90
 - 4.2.2 路由配置项 90
 - 4.2.3 路由参数 91
 - 4.2.4 缺省路由 93
 - 4.3 客户端路由器的工作机制 93
 - 4.3.1 片段标识符方式 94
 - 4.3.2 HTML5 历史API 方式 95
 - 4.3.3 使用HTML5 历史API 方式 97
 - 4.4 综合实作：实现SPA 路由 98
 - 4.4.1 教员列表（缺省路由） 99
 - 4.4.2 主要联系人路由 101
 - 4.4.3 教员授课时间（参数化路由） 102
 - 4.5 挑战环节 104
 - 4.6 小结 105
- 5 视图合成与布局 106
 - 5.1 项目介绍 107
 - 5.2 布局设计概念 108
 - 5.2.1 视图 108
 - 5.2.2 Region 109
 - 5.2.3 视图合成 110
 - 5.2.4 嵌套视图 111
 - 5.2.5 路由 112

- 5.3 高级合成与布局的可选方案 113
 - 5.3.1 优点 113
 - 5.3.2 缺点 114
- 5.4 设计应用程序 114
 - 5.4.1 设计基本布局 115
 - 5.4.2 设计基本内容 117
 - 5.4.3 在复杂设计中应用视图管理 122
 - 5.4.4 通过自身状态创建嵌套视图 125
- 5.5 挑战环节 127
- 5.6 小结 128
- 6 模块间交互 129
 - 6.1 模块概念回顾 131
 - 6.1.1 用模块封装代码 131
 - 6.1.2 API 提供对内部功能的访问控制 133
 - 6.1.3 SRP——以单一目的作为设计出发点 134
 - 6.1.4 代码重用——控制项目规模 135
 - 6.2 模块间交互方式 136
 - 6.2.1 通过依赖进行模块间交互 136
 - 6.2.2 依赖方式的优缺点 138
 - 6.2.3 通过发布/订阅模式进行模块间交互 138
 - 6.2.4 发布/订阅模式优缺点 141
 - 6.3 示例项目细节 142
 - 6.3.1 搜索功能 144
 - 6.3.2 显示产品信息 150
 - 6.4 挑战环节 155
 - 6.5 小结 155
- 7 与服务器端通信 156
 - 7.1 示例项目新要求 157
 - 7.2 与服务器端通信综述 158
 - 7.2.1 选择数据类型 158
 - 7.2.2 HTTP 请求方法 159
 - 7.2.3 数据转换 160
 - 7.3 使用MV* 框架 161
 - 7.3.1 请求生成 162
 - 7.3.2 通过回调函数处理结果 165
 - 7.3.3 通过Promise 处理结果 166
 - 7.3.4 Promise 错误处理 170
 - 7.4 RESTful Web 服务调用 172
 - 7.4.1 什么是REST 172
 - 7.4.2 REST 原则 172
 - 7.4.3 MV* 框架的RESTful 支持 174
 - 7.5 示例项目细节 174
 - 7.5.1 配置REST 调用 174
 - 7.5.2 添加产品到购物车 177
 - 7.5.3 查看购物车 179
 - 7.5.4 修改购物车 181
 - 7.5.5 从购物车中移除产品 183
 - 7.6 挑战环节 184
 - 7.7 小结 184

- 8 单元测试 186
 - 8.1 示例项目说明 187
 - 8.2 什么是单元测试 187
 - 8.2.1 单元测试的好处 188
 - 8.2.2 构建更好的单元测试 189
 - 8.3 传统的单元测试 192
 - 8.3.1 QUnit 起步 193
 - 8.3.2 创建第一个单元测试 196
 - 8.3.3 测试由MV* 对象创建的代码 200
 - 8.3.4 测试对DOM 所做的改变 205
 - 8.3.5 混合使用其他测试框架 206
 - 8.4 挑战环节 208
 - 8.5 小结 208
- 9 客户端任务自动化 209
 - 9.1 Task Runner 的常见用途 210
 - 9.1.1 即时刷新浏览器 210
 - 9.1.2 自动化JavaScript 和CSS 的预处理过程 211
 - 9.1.3 自动化Linter 代码分析 211
 - 9.1.4 持续单元测试 211
 - 9.1.5 文件串接 212
 - 9.1.6 代码压缩 212
 - 9.1.7 持续集成 212
 - 9.2 Task Runner 选择 212
 - 9.3 本章示例项目 213
 - 9.3.1 Gulp.js 介绍 214
 - 9.3.2 创建第一个任务 215
 - 9.3.3 创建代码分析任务 216
 - 9.3.4 创建浏览器刷新任务 218
 - 9.3.5 自动化单元测试 220
 - 9.3.6 创建构建过程 222
 - 9.4 挑战环节 227
 - 9.5 小结 227
- A 员工通讯录示例说明 229
- B XMLHttpRequest API 259
- C 第7 章内容的服务器端设置与总结 266
- D 安装Node.js 与Gulp.js 277

《SPA设计与架构：理解单页面Web》

精彩短评

- 1、英文版应该很早就出了，内容比较表面和落后，缺少深入的技术洞察，单页面程序除了函数命名污染，还有元素ID污染、事件重复监听、监听内存易泄漏等问题，这些难点问题都没有触碰，非常遗憾。
- 2、狂喜，感觉脑海中堵塞的回路都被疏通了，改名叫前端设计与架构也不为过。以ng和backbone为例从宏观角度、思想层面阐述了MV*、模块化、路由、REST、单测、优化的各种框架和理念，突出一个新字，还想再多看好几遍。
- 3、毫无压力的看完了
- 4、这绝对是一本好书，作者很用心，翻译也非常给力；从内容上看，它会将你的视角自然地过渡到比较现代的Web开发方式——MV*单页面应用，适用于后端UI逻辑转向前端的应用场景，前端不再是简单的切图和特效。为了实现这样的方式，作者从模块化、路由、视图、数据源等各方面结合一些流行的技术方案进行了介绍，非常值得一看。
- 5、2015年出的，到现在其实有一点过时吧。花了一天看完，虽然没学过angular，backbone等，但不影响理解，有所收获，本书涉及gulp，单元构建，路由，模块化开发，restful web 服务器等，这些我都有所了解，看了之后，我的整个知识体系更加完整了。如果去年年初出来，绝对是本好书

《SPA设计与架构：理解单页面Web》

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：www.tushu000.com