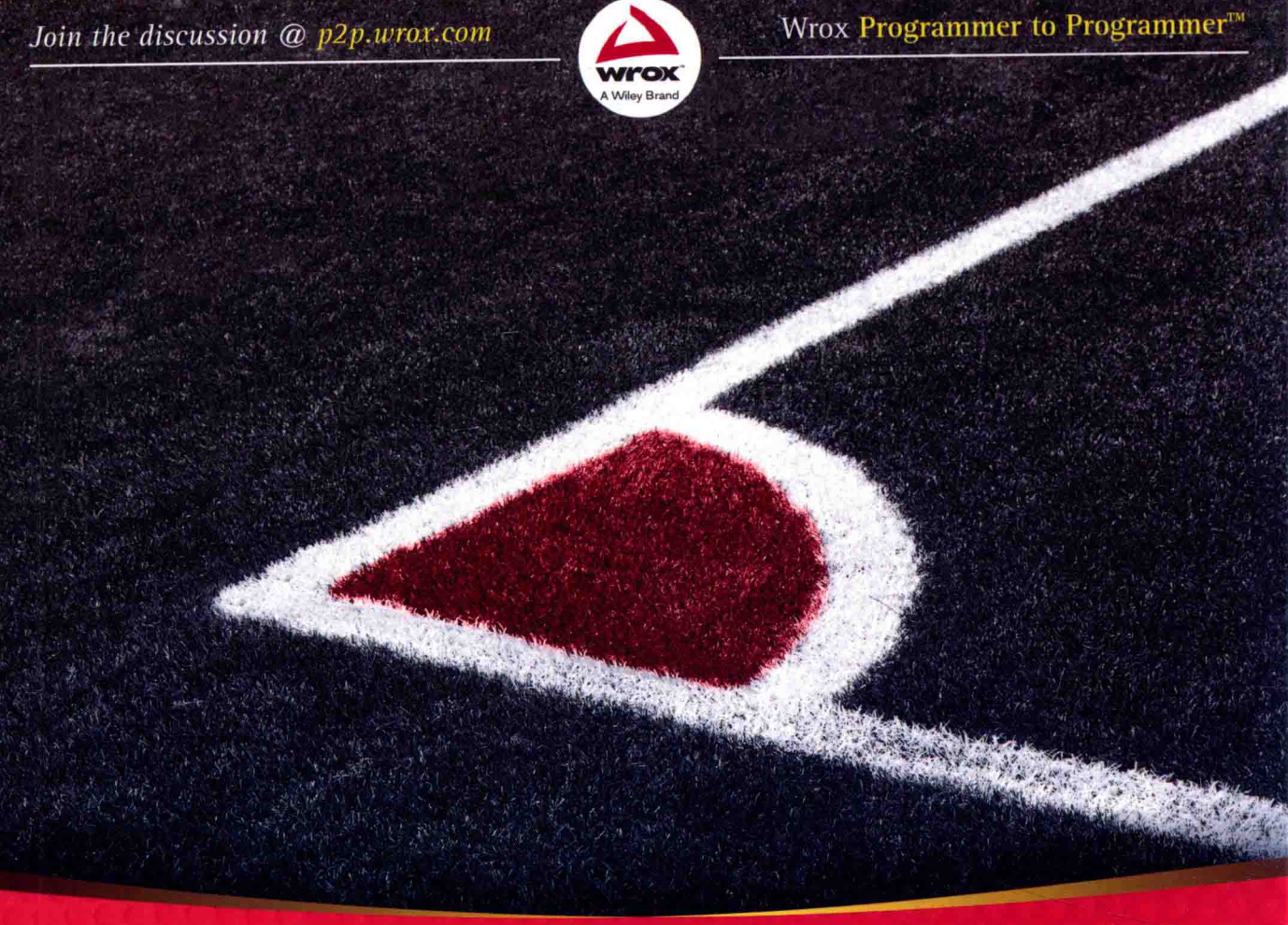

Professional AngularJS

AngularJS 高级编程

[美] Valeri Karpov 著
Diego Netto
王肖峰 译

清华大学出版社

AngularJS 高级编程

[美] Valeri Karpov
Diego Netto 著
王肖峰 译

清华大学出版社
北 京

Valeri Karpov, Diego Netto
Professional AngularJS
EISBN: 978-1-118-83207-3

北京市版权局著作权合同登记号 图字：01-2015-3643

图书在版编目(CIP)数据

AngularJS 高级编程/(美)卡尔波夫(Karpov,V.)，(美)尼托(Netto,D.) 著；王肖峰 译. —北京：清华大学出版社，2016

书名原文：Professional AngularJS

ISBN 978-7-302-42866-4

Ⅰ. ①A… Ⅱ. ①卡… ②尼… ③王… Ⅲ. ①超文本标记语言—程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2016)第 028924 号

责任编辑：王 军 韩宏志
装帧设计：孔祥峰
责任校对：成凤进
责任印制：李红英

出版发行：清华大学出版社
网 址：http://www.tup.com.cn，http://www.wqbook.com
地 址：北京清华大学学研大厦 A 座 邮 编：100084
社 总 机：010-62770175 邮 购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：清华大学印刷厂
装 订 者：三河市溧源装订厂
经 销：全国新华书店
开 本：185mm×260mm **印 张**：22.25 **字 数**：528 千字
版 次：2016 年 2 月第 1 版 **印 次**：2016 年 2 月第 1 次印刷
印 数：1~3500
定 价：59.80 元

产品编号：062760-01

译 者 序

随着 JavaScript 社区的快速发展，AngularJS 在 2012 年 6 月发布 1.0 版本时横空出世。该 JavaScript 客户端框架是由 Google 开发的，它可以通过 MVC 框架帮助开发者实现设计优良的 Web 页面和应用。它提供了 MVVM、模块化、双向数据绑定、依赖注入等特性。正是这些特性，使它迅速成为 Web 开发领域的新宠，甚至被誉为 Web 开发世界中最激动人心的创新技术之一。

本书当之无愧地可以被称为 AngularJS 高级编程，除了讲解基础知识之外，它还包含了大量对 AngularJS 特性底层实现的描述，以及许多自动化构建工具和测试工具的应用。其中主要内容包括：

- 智能工作流和构建工具 Yeoman
- 文件依赖管理工具 RequireJS 和 Browserify
- 双向数据绑定、AngularJS 作用域和$digest 循环
- 嵌入指令
- 单页面应用和 Prerender
- 服务的内部实现
- 开源测试运行器 Karma 以及行为驱动开发测试框架 Mocha 和 Jasmine
- 集成 Twitter Bootstrap、Ionic 以及 Moment 和 Mongoose

本书还包含丰富的样例，而且每个章节的内容和样例都独立设计，如果你对特定知识领域感兴趣，可单独阅读某章内容。通过本书的学习，具有一定开发经验的读者可以了解更多AngularJS内幕信息，而初学者通过顺序阅读本书可以学到一套完整的AngularJS开发实践。

我非常高兴清华大学出版社准备引入本书，也非常欣慰自己能负责本书的翻译工作。通过这个过程，不仅可为大家带来一本真正的 AngularJS 开发秘籍，帮助你们快速掌握这门新兴技术，也可以让自己加深对 AngularJS 的理解。

另外，感谢清华大学出版社的编辑们为本书付出的心血。同样感谢妻子对我翻译工作的支持和鼓励。没有你们的支持和鼓励，本书就不可能顺利出版。

对于这本经典之作，译者对本书进行了严格审核，对其中一些具有争议的地方也进行了反复考证，但个人精力有限，难免有疏漏之处，敬请各位读者谅解。如有任何意见或建议，请不吝指正。本书全部章节由王肖峰翻译，参与本次翻译的还有杜欣、高国一、孙其淳、孙绍辰、徐保科、尤大鹏、张立红、邓伟、王蕊、王小红、马宁宁、韩丽威、王耀光。

最后，祝愿各位读者通过阅读本书可以快速掌握 AngularJS 的高级特性，在工作中发挥出它的强大作用。

作 者 简 介

Valeri Karpov是MongoDB的一位NodeJS工程师，他专注于维护流行的Mongoose ODM和其他众多与MongoDB相关的NodeJS模块。此外，他还是BookaLokal网站的一名技术专家、StrongLoop的博主和MEAN栈的命名者。从2010年发布的AngularJS v0.9.4开始，他一直在生产环境中使用AngularJS应用。最近，他还使用AngularJS构建了BookaLokal的移动网站，以及MongoDB内部持续集成框架的一个Web客户端。

Diego Netto是一位软件咨询师和开源技术讲师。他拥有众多工程师和企业家的不同称号。作为Los Angeles和Dallas开发商店的所有者，Diego为创业公司和企业创建Web和移动应用。作为IonicFramework Yeoman生成器的维护者，他使用最新的AngularJS和IonicFramework为www.aboatapp.com构建了Prop移动应用，并使用Famo.us/Angular为www.modelrevolt.com构建了移动应用。

技术编辑简介

Stéphane Bégaudeau毕业于南特科学技术学院，目前是法国Obeo公司的Web技术专家和Eclipse模型咨询师。他已经为Eclipse Foundation的几个开源项目做出了贡献，并且还是Acceleo的领导者。他还从事Dart Designer的开发工作，这是Dart编程语言的一个开源工具。

致　谢

苏格拉底曾写道“教育不是填满一个瓶子，而是点燃一束火焰”。本着这种精神，我非常感激我的老师和导师，他们让我的一生充满激情，而不只是把工作当作一个职业。尤其是，我希望感谢普林斯顿大学的 Kernighan 教授和 Tarjan 教授，以及 Bergen County Academies 的 Nevard 博士、Sankaran 博士、Scarpone 和 Nodarse。另外，我希望感谢 Misko Hevery，他是我在 Google 实习时的导师，也是 AngularJS 的原作者，在短短 12 周的时间内，他教给我的软件工程相关的知识比在那个夏天之前所有时间内学到的知识还要多。

—Valeri Karpov

引用成功的企业家 Felix Dennis 说过的一句话：“人不是忙着学习，就是忙着死亡。”这是一个谨慎的提醒，尤其是对于不断发展的软件工程来说，为了保持进步和成功，我们必须终生追求知识。我希望感谢 Virginia Tech 的 Tatar 和 Ribbens 教授，他们让我认识到了这一点。我还希望感谢 Addy Osmani，他帮助我发现了智能工具的重要性，并鼓励我为开源社区做出贡献。特别要感谢我的朋友和合作者 Valeri Karpov，他们早早就让我接触到了 AngularJS。

—Diego Netto

前　言

作为 JavaScript 开发者，现在是一个激动人心的时刻。随着服务器端 JavaScript 开源社区的快速发展(在 2013 年 12 月，NodeJS 包管理器拥有 50 000 个包，而到了 2014 年 10 月这个数字增加了一倍)，下一代客户端框架的流行(例如 AngularJS)，完全基于 JavaScript 构建 Web 工具的公司数量不断增长，对 JavaScript 语言技能的需求也不断增多。现代工具允许我们使用一种语言构建复杂的、基于浏览器客户端的高度并发服务器，甚至是混合的原生移动应用。AngularJS 迅速成为主流的下一代客户端 Web 框架，它允许个人、小团队和大型公司构建和测试基于浏览器的复杂应用。

AngularJS 介绍

随着JavaScript社区的快速发展，AngularJS在2012年6月发布1.0版本时横空出世。尽管它是一个较新的框架，但它在构建应用时提供了强大的特性和优雅的工具，这使它成为许多开发者选择的前端框架。AngularJS最初由Google的测试工程师Misko Hevery开发，他发现现有的工具(例如jQuery)很难构建出需要显示大量复杂数据的浏览器用户界面(User Interface，UI)。Google现在有一个专门的团队用于开发和维护AngularJS以及相关的工具。一些活跃的Google应用也是使用AngularJS开发的，从DoubleClick Digital Marketing Platform到PlayStation 3上的YouTube应用。AngularJS的人气在迅速增长：到2014年10月，Quantcast Top10k网站中有143个都使用了AngularJS，并迅速超过了最接近的对手：KnockoutJS、ReactJS和EmberJS。

那么AngularJS特别之处在哪里呢？从https://angularjs.org/网站中借用一个对AngularJS特别简洁的描述："写更少的代码，早点去喝啤酒"。AngularJS的核心是一个称为"双向数据绑定"的概念，通过它可将超文本标记语言(Hypertext Markup Language，HTML)和层叠样式表(Cascading Style Sheet，CSS)绑定到JavaScript变量的状态。无论何时变量发生了变化，AngularJS都将更新所有应用了该JavaScript变量的HTML和CSS，如下面的代码所示：

```
<div ng-show="shouldShow" >Hello</div>
```

如果变量 shouldShow 被改为 false，AngularJS 将自动隐藏 div 元素。变量 shouldShow 并没有什么特殊之处：AngularJS 不要求在特殊类型中封装变量；变量 shouldShow 可以是一个普通的 JavaScript 布尔值。

尽管双向绑定是 AngularJS 的基础，但它只是冰山一角。AngularJS 提供了一个优雅的

框架，可以通过一种最大化重用性和测试性的方式来组织客户端 JavaScript。另外，AngularJS 有一组丰富的测试工具，例如 Karma、protractor 和 ngScenario(参见第 9 章)，它们已经做了优化以便用于 AngularJS。AngularJS 专注于可测试的架构和丰富的测试工具，这使它成为关键客户端 JavaScript 的自然选择。它不仅可以使你快速编写复杂的应用，还提供了工具和结构，使应用的测试变得非常容易。事实上，Google 的 DoubleClick 团队将 AngularJS 的“full testing story”引用为将它的数字营销平台迁移到 AngularJS 的 6 个最重要原因之一。下面是对 AngularJS 特点的一些简单概述。

双向数据绑定

在许多较老的客户端 JavaScript 库(例如 jQuery 和 Backbone)中，我们希望自己操作文档对象模型(Document Object Model，DOM)。换句话说，如果希望改变 div 元素的 HTML 内容，需要自己编写必需的 JavaScript。例如：

```
$('div').html('Hello, world!');
```

AngularJS反转了这个模式，使HTML成为如何显示数据的明确来源。双向数据绑定的主要目的是将HTML或CSS属性(例如，div元素的HTML内容或背景颜色)绑定到JavaScript变量的值。当JavaScript变量的值改变时，HTML或CSS属性将随之更新。反之亦然：如果用户在input字段中输入，被绑定的JavaScript变量的值将被更新为用户输入的内容。例如，下面的HTML将问候输入字段中输入的名字。可以在相应章节的样例代码data_binding.html中找到该样例：简单地右击该文件，并在浏览器中打开它——不需要Web服务器或其他依赖！

```
<input type="text" ng-model="user" placeholder="Your Name">
<h3>Hello, {{user}} !</h3>
```

不需要使用 JavaScript！指令 ngModel 和{{}}简写语法将完成所有工作。在这个简单的样例中，AngularJS 体现出的优点非常有限，但在第 1 章构建一个真正的应用时，你将看到数据绑定将极大地简化 JavaScript。多亏了数据绑定，否则我们就很难将 800 行的 jQuery 意大利面条式代码简化成 40 行清晰的、独立于 DOM 的 AngularJS 代码。

DOM 作用域

DOM 作用域是 AngularJS 另一个强大的特性。你可能已经猜到，数据绑定并不是免费的午餐；代码复杂性一定会被转移到某个地方。不过，AngularJS 允许在 DOM 中创建作用域，它的行为类似于 JavaScript 和其他编程语言中的作用域。这将允许我们把 HTML 和 JavaScript 分割成独立的、可重用的块。例如，下面的样例实现的功能与之前样例的功能相同，但使用了两个不同的作用域：一个用于使用英文进行问候，另一个使用的是西班牙语。

```
<div ng-controller="HelloController" >
  <input type="text" ng-model="user" placeholder="Your Name">
  <h3>Hello, {{user}}!</h3>
```

```
</div>

<hr>
<div ng-controller="HelloController" >
  <input type="text" ng-model="user" placeholder="Su Nombre">
  <h3>Hola, {{user}}!</h3>
</div>

<script type="text/javascript"
        src="angular.js">
</script>
<script type="text/javascript">
  functionHelloController($scope) {}
</script>
```

指令ngController是一种创建新作用域的方式，通过它可将相同的代码为不同的目的进行重用。第 4 章包含了对双向数据绑定的全面概述，并对内部实现细节进行了讨论。

指令

指令是将HTML和JavaScript功能组装成一个可轻松重用的包的强大工具。AngularJS拥有大量内建指令，例如之前看到的ngController和ngModel，通过他们可在HTML中访问复杂的JavaScript功能。也可以编写自己的自定义指令。AngularJS允许将HTML与指令相关联，因此可以使用指令作为一种重用HTML的方式，以及一种将特定行为绑定到双向数据绑定的方式。编写自定义指令超出了该简介的范围，但第 5 章包含了该主题的全面概述。

模板

在双向数据绑定之上，AngularJS 允许根据 JavaScript 变量的状态替换页面的整个部分。指令 ngInclude 允许有条件地包含模板，根据 JavaScript 的状态在页面中注入 AngularJS HTML 片段。下面的样例演示了一个包含了一个 div 元素的页面，其中包含基于 myTemplate 变量值的不同 HTML。可在相应章节的样例代码的 templates.html 中找到该样例：

```
<div ng-controller="TemplateController">
  <div ng-include="myTemplate">
  </div>
  <br>
  <a ng-click="myTemplate = 'template1';"
     style="cursor: pointer"
     ng-class="{'selected': myTemplate === 'template1' }">
    Display Template 1
  </a>
  <a ng-click="myTemplate = 'template2';"
     style="cursor: pointer"
     ng-class="{'selected': myTemplate === 'template2' }">
    Display Template 2
  </a>
</div>
```

```
<script type="text/javascript"
    src="angular.js">
</script>
<script type="text/javascript">
  functionTemplateController($scope) {
    $scope.myTemplate = 'template1';
  }
</script>
<script type="text/ng-template" id="template1">
  <h1>This is Template 1</h1>
</script>
<script type="text/ng-template" id="template2">
  <h1>This is Template 2</h1>
</script>
```

第 6 章包含了 AngularJS 模板的完整讨论，包括如何使用它们构建单页面应用。

测试和工作流

提供一个框架用于编写可以进行单元测试的代码一直是 AngularJS 从首次发布开始的目标。AngularJS 包含一个优雅、复杂的依赖注入器，而且所有的 AngularJS 组件(控制器、指令、服务和过滤器)都是使用依赖注入器构造的。这将保证在测试时，如果需要我们可以轻松替换代码依赖。另外，AngularJS 团队已经开发了大量强大的测试工具，例如 Karma test runner 和 protractor 以及 ngScenario 集成测试框架。这些工具带来了复杂的多浏览器测试基础设施，这在之前只有大型公司可以实现，现在开发者个人也可以实现。

另外，AngularJS的架构和测试工具可与各种开源JavaScript构建和工作流工具优雅地进行交互，例如Gulp和Grunt。通过使用这些工具，我们可以无缝地执行测试、在测试执行中绑定代码覆盖和linting这样的工具，甚至从头开始搭建新的应用。核心AngularJS只是一个库，但是围绕它的测试和工作流工具使AngularJS生态系统成为一个整体，一种用于构建基于浏览器客户端的创新模式。第9章对可在AngularJS应用中使用的AngularJS测试生态系统和不同类型的测试策略进行了详述。

不适合使用 AngularJS 的场景

与任何其他库一样，AngularJS 非常适用于一些应用，而不太适用于其他应用。在下一节，将学习几个非常适用于 AngularJS 的用例。在本节，将学习 AngularJS 不太适用的一些情形，并了解 AngularJS 的一些限制。

需要支持旧版 Internet Explorer 的应用

AngularJS的这个限制对于一些用户来说可能非常重要，因为它不支持旧版Internet Explorer。AngularJS 1.0.x支持Internet Explorer 6 和 7，但是本书中将学习的版本AngularJS

1.2.x只支持Internet Explorer 8 以及更新版本。此外，AngularJS目前的实验版本AngularJS 1.3.x完全放弃了对Internet Explorer 8 的支持(它们只支持Internet Explorer 9 和更新版本)。如果你的应用需要支持Internet Explorer 7，那么使用AngularJS并不是正确之选。

不需要 JavaScript 服务器 I/O 的应用

AngularJS 是一个极其丰富和强大的库，狂热的用户通常会尝试在所有的应用中使用它。不过，许多情况下使用 AngularJS 都有点大材小用了，而且增加了不必要的复杂性。例如，如果需要在页面中添加一个按钮，当用户单击它时显示或隐藏一个 div 元素，那么使用 AngularJS 并不能帮助你，除非你需要持久化页面 URL 中或者发送到服务器的 div 状态。与此类似，选择使用 AngularJS 编写博客通常是一个糟糕的决定。博客通常以有限的交互方式显示出简单数据，所以使用 AngularJS 通常是不必要的。另外，博客要求与搜索引擎进行良好的集成。如果使用 AngularJS 编写博客的话，还需要完成一些额外的工作(参见第 6 章)，以保证搜索引擎可以高效地爬取博客内容，因为搜索引擎爬虫不执行 JavaScript。

AngularJS 适用的场景

现在我们已经了解了 AngularJS 的一些限制，接下来将学习的是一些 AngularJS 非常适用的用例。

内部数据密集型应用

对于需要在浏览器 UI 中显示复杂数据的应用来说，AngularJS 是一个极其有用的强大工具，例如持续集成框架或产品仪表盘。为这些应用开发 UI 的挑战在于编写重要的 JavaScript，在每次数据改变时正确地渲染数据。通过使用双向数据绑定，我们不必再编写这样的胶水代码，就可以编写出更简洁和易于阅读的 JavaScript。当我们编写第 1 章展示的股票市场仪表盘时，将会看到通过双向数据绑定和指令构建出一个需要显示大量数据的应用是非常简单的。

移动网站

AngularJS为大多数常见的移动浏览器提供了扩展支持(Android、Chrome Mobile、iOS Safari)。而且，在第 6 章你将看到AngularJS有一些强大的动画支持和单页面应用(通过它，我们可以利用浏览器缓存来尽量减少带宽的使用)。这将使我们可以构建快速的、高效模拟原生应用的移动Web应用。另外，通过使用Ionic这样的框架，我们还可以构建混合移动应用：使用JavaScript编写应用(使用AngularJS)，但是通过Android和iPhone应用商店发布。

构建原型

一个在本书中多次出现的主题是使用双向数据绑定在前端 JavaScript 工程和用户界面/用户体验(UI/UX)设计之间创建有效的分离。通过双向数据绑定，前端 JavaScript 工程师可

以公开一个应用编程接口，然后 UI/UX 设计师就可以在 HTML 中访问它们，从而使前端工程师和设计师都能在最佳环境中工作，而不必介入彼此的工作。在快速构建原型浏览器 UI 时，这尤其有用，因为我们可以高效地并行安排任务，使团队平稳地运行。另外，AngularJS 丰富的测试生态系统将使我们可以保证高测试覆盖率，从而确保在展示原型时不会出现明显的问题。

如何使用本书

现在你已经看到了 AngularJS 库变得如此流行的原因，接下来是对本书内容的概述，你将从中了解到从编写初级 AngularJS 到专业级 AngularJS 的所有内容。

可将本书看成学习 AngularJS 的一本“选择个人冒险”的书籍。如果你是一个 AngularJS 初学者，通过顺序阅读本书你将获得大量信息，因为这些章节提供了从头开始学习 AngularJS 的逻辑序列。不过，这些章节和它们的样例基本上被设计为相对独立的。如果你熟悉 AngularJS 并且在寻找拓展自己某个特定领域技能的知识，例如使用测试框架(第 9 章)，那么可以直接跳到合适的章节并忽略中间的章节。某些样例在章节之间是共享的，但每个章节都解释了样例的某块代码(假设你之前从未见过它)。而且，一些章节将引用其他章节的信息，但它们总是提供了所需概念的简略概述。无论你是刚刚开始学习 AngularJS 还是在寻找特定主题，本书都允许你直接跳到最有用的信息(不过，如果你是一个初学者，在跳到其他章节之前应该先阅读第 1 章)。下面对每章内容做一些简单强调。

第 1 章：构建简单的 AngularJS 应用

该章是面向初学者的。将使用 AngularJS 从头开始构建一个股票市场仪表盘应用，并获得对接下来章节所涵盖话题的高级概述。

第 2 章：智能工作流和构建工具

该章将讲解许多用于搭建 AngularJS 应用、自动化工作流、添加外部依赖的开源工具。该章将特别强调流行的搭建工具 Yeoman，它不仅能使我们可以快速启动新的 AngularJS 应用，还提供了管理工作流的强大工具。

第 3 章：架构

该章提供了构建 AngularJS 组件最佳实践的概述，包括如何在服务、控制器和指令之间传递数据。另外，该章浏览了不同规模应用的目录结构的最佳实践。最后，该章涵盖了两个管理文件依赖的流行工具：RequireJS 和 Browserify。

第 4 章：数据绑定

尽管 AngularJS 数据绑定非常优雅和直观，但是中级 AngularJS 开发者可以通过深入了解数据绑定的实际实现方式而获得进步。该章浏览了如何构建 AngularJS 作用域，以及

$digest 循环的实现细节，从而使我们可以避免常见的数据绑定陷阱。该章还包含了过滤器的概述，包括相关用例和常见的错误。

第 5 章：指令

该章的前半部分提供了如何编写自定义 AngularJS 指令的基本知识，并浏览了指令的各种用例。后半部分则专注于使用诸如嵌入(transclusion)的工具设计更高级的指令。

第 6 章：模板、位置和路由

该章的主要目的是提供如何使用 AngularJS 编写单页面应用的概述，这种应用允许用户在多个“视图”之间进行转换，而不必重新加载页面。为创建单页面应用，该章详述 AngularJS 模板、模板缓存和$location 服务。该章还提供了使用 AngularJS CSS3 动画的概述，以及介绍如何通过使用 Prerender 让单页面应用变得对搜索引擎友好。

第 7 章：服务、工厂和提供者

该章对使用 AngularJS 创建服务的各种不同方法进行了详尽描述。我们还将学习服务在底层是如何工作的，以及如何利用服务的内部实现。

第 8 章：服务器通信

该章将使用基本的服务和拦截器创建一个登录系统。另外，还将学习如何使用 StrongLoop 的 Loopback API 启动一个简单后端，并使用客户端 AngularJS 应用和 Loopback API 集成 Facebook 登录。

第 9 章：测试和调试 AngularJS 应用

该章包含了使用流行的开源测试运行器Karma为AngularJS应用构建单元测试和DOM集成测试(也称为halfway test)的详尽描述。该章还讨论了开源的行为驱动开发(Behavior-Driven Development，BDD)测试框架Mocha和Jasmine，并解释了如何在SauceLab的浏览器云中运行测试。

第 10 章：继续前行

该章包含了对几个流行开源模块的概述，通过它们，AngularJS 可以实现一些令人惊喜的事情。尤其是，将学习如何使用 Angular-UI Bootstrap 集成 Twitter Bootstrap 组件、如何使用 AngularJS 和 Ionic 框架构建混合移动应用，以及如何在 AngularJS 中集成两个流行的开源 JavaScript 模块 Moment 和 Mongoose，还将学习如何结合使用 ECMAScript 6 生成器和 AngularJS 的$http 服务。

如何使用本书的样例代码

本书的每章内容都有自己的样例代码，可以在 http://www.wrox.com/go/proangularjs 的代码下载部分获得。每章的开头都将提醒访问该 URL 以下载样例代码，因此不需要收藏这

个页面。尽管每章都恰当地包含了文本形式的代码，但最好下载每章的样例代码并自己尝试这些样例。另外，也可以访问 www.tupwk.com.cn/downpage，再输入中文书名或中文 ISBN，下载源代码。

本书的样例代码被设计为拥有最小的外部依赖。每章开头都解释了运行样例代码所需的特殊依赖。对于本书的许多样例来说，只需要一个现代浏览器即可(这些样例主要是使用 Google Chrome 37 和 Mozilla Firefox 32 开发的，但是 Internet Explorer 9 和 Safari 6 应该也足够使用)。这些样例都以.html 文件形式存在，可以通过右击文件并使用 file://协议在浏览器中打开该文件。例如，为查看样例代码中的 data_binding.html 样例，如果样例代码在目录/Users/user/Chapter 0 中，那么可以访问 file:///Users/user/Chapter%200/data_binding.html。除非特别指定，否则可以安全地认为可以在浏览器中打开本书样例代码的任意 HTML 文件。

本书的样例代码不要求使用特殊的集成开发环境(IDE)。尝试样例代码时使用文本编辑器(例如 vim 和 SublimeText)即可。如果喜欢，也可以使用诸如 WebStorm 的 IDE，但对于本书的样例来说，使用 IDE 的好处非常有限。

本书涵盖的许多概念都要求使用Web服务器。为使整个过程尽可能轻量级，本书将使用NodeJS和NodeJS包管理器npm来启动Web服务器。另外，你将在本书中学到许多工具，例如Grunt、Prerender和Yeoman，通过npm都可以轻松安装它们。为安装NodeJS，你应该访问http://nodejs.org/download，并按照对应平台的指令进行安装。NodeJS非常易于安装，并且几乎支持所有常见的桌面操作系统(包括Windows)；而且，npm被自动包含在了NodeJS中。不过，本书要求使用NodeJS的大部分样例都假定你正在使用bash shell。Linux和OS X用户可以使用它们的默认终端。在Windows中，如果你希望运行命令行指令，那么应使用git bash(http://msysgit.github.io)，这是Windows的一个bash终端(记住，NodeJS并未正式支持Cygwin，所以不推荐使用它)。每章都将演示如何安装额外的依赖，并在必要时提醒你安装NodeJS。

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但是错误总是难免的，如果你在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

请给 wkservice@vip.163.com 发电子邮件，我们就会检查你的反馈信息，如果是正确的，我们将在本书的后续版本中采用。

要在网站上找到本书英文版的勘误表，可以登录 http://www.wrox.com/WileyCDA，通过 Search 工具或书名列表查找本书，然后在本书的细目页面上，点击 Errata 链接。在这个页面上可以查看到 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表，网址是 www.wrox.com/WileyCDA/id_105077.html。

p2p.wrox.com

要与作者和同行讨论，请加入 p2p.wrox.com 上的 P2P 论坛。这个论坛是一个基于 Web 的系统，便于你张贴与 Wrox 图书相关的消息和相关技术，与其他读者和技术用户交流心得。该论坛提供了订阅功能，当论坛上有新的消息时，它可以给你传送感兴趣的论题。Wrox 作者、编辑和其他业界专家和读者都会到这个论坛上来探讨问题。

在 http://p2p.wrox.com 上，有许多不同的论坛，它们不仅有助于阅读本书，还有助于开发自己的应用程序。要加入论坛，可以遵循下面的步骤：

(1) 进入 p2p.wrox.com，单击 Register 链接。

(2) 阅读使用协议，并单击 Agree 按钮。

(3) 填写加入该论坛所需要的信息和自己希望提供的其他信息，单击 Submit 按钮。

(4) 你会收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。

注意：

不加入 P2P 也可以阅读论坛上的消息，但要张贴自己的消息，就必须加入该论坛。

加入论坛后，就可以张贴新消息，响应其他用户张贴的消息。可以随时在 Web 上阅读消息。如果要让该网站给自己发送特定论坛中的消息，可以单击论坛列表中该论坛名旁边的 Subscribe to this Forum 图标。

关于使用 Wrox P2P 的更多信息，可阅读 P2P FAQ，了解论坛软件的工作情况以及 P2P 和 Wrox 图书的许多常见问题。要阅读 FAQ，可以在任意 P2P 页面上点击 FAQ 链接。

源代码

在读者学习本书中的示例时，可手动输入所有的代码，也可使用本书附带的源代码文件。本书使用的所有源代码都可从本书合作站点 http://www.wrox.com/ 或 www.tupwk.com.cn/downpage 下载。登录到站点 http://www.wrox.com/，使用 Search 工具或使用书名列表就可以找到本书。接着单击 Download Code 链接，就可以获得所有的源代码。既可选择下载一个大的包含本书所有代码的 ZIP 文件，也可以只下载某个章节中的代码。

注意：

由于许多图书的标题都很类似，因此按 ISBN 搜索是最简单的，本书英文版的 ISBN 是 978-1-118-83207-3。

在下载代码后，只需用解压缩软件对它进行解压缩即可。另外，也可以进入 http://www.wrox.com/dynamic/books/download.aspx 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有代码。

目　录