Vue 3移动Web

开发与性能调优实战

吕 鸣 / 著

清華大學出版社
北 京

内 容 简 介

本书旨在向读者介绍如何使用 Vue 3 和其他现代 Web 技术创建高性能的移动 Web 应用程序。本书不仅详细介绍有关移动 Web 和 Vue 3 的技术知识，包括 HTML5、CSS3、Vue 全家桶、构建工具 Vite、移动 Web 屏幕适配等，并讲解如何使用这些技术来创建快速、可靠和可扩展的应用程序，还深入探讨各种性能优化技术，并向读者展示如何使用这些技术来提升 Web 应用程序的性能、可靠性和用户体验。最后通过一个企业级实战项目——仿微信朋友圈系统来全方位讲解移动 Web 和 Vue 3 在企业级项目中的应用实践。

本书既适合有一定前端开发基础的学生、前端开发的从业者以及自由项目开发者，也适合对 Vue 3 感兴趣的、擅于做各种 Vue 3 应用探索、想要深入了解 Vue 3 底层实现的开发者，还可作为高校相关专业的教学用书。

图书在版编目（CIP）数据

Vue 3 移动 Web 开发与性能调优实战/吕鸣著. —北京：清华大学出版社，2023.5
ISBN 978-7-302-63580-2

Ⅰ. ①V… Ⅱ. ①吕… Ⅲ. ①网页制作工具－程序设计 Ⅳ. ①TP393.092.2

中国国家版本馆 CIP 数据核字（2023）第 090913 号

责任编辑：王金柱
封面设计：王　翔
责任校对：闫秀华
责任印制：宋　林

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：010-83470000　　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 装 者：北京同文印刷有限责任公司
经　　销：全国新华书店
开　　本：190mm×260mm　　印　　张：22　　字　　数：594 千字
版　　次：2023 年 6 月第 1 版　　印　　次：2023 年 6 月第 1 次印刷
定　　价：99.00 元

产品编号：100104-01

前　言

移动 Web 和 Vue 3 是当今互联网开发中热门的技术之一。随着移动设备的普及和用户需求的变化，构建高效、响应式的 Web 应用程序已经成为一项非常重要的任务。本书旨在向读者介绍如何使用 Vue 3 和其他现代 Web 技术创建高性能的移动 Web 应用程序。

本书详细介绍有关移动 Web 和 Vue 3 的技术知识，并讲解如何使用这些技术来创建快速、可靠和可扩展的应用程序。此外，本书还深入探讨各种性能优化技术，并向读者展示如何使用这些技术来提升 Web 应用程序的性能、可靠性和用户体验。

本书使用的 Vue 版本是 3.2.28 版本，它是当前 Vue 3 中最稳定的版本。我们通常把 Vue.js 3.0 及其以上的（例如 3.2.4，3.0 等）统称为 Vue 3 版本，而把 Vue.js 2 的一些版本统称为 Vue 2 版本。相较于 Vue 2 来说，Vue 3 在源码实现上有了一定程度的改变，并且在性能和可用性上有了很大的提升。

本书主要内容

- 移动 Web 和 Vue 3 的基础知识：介绍如何创建适用于移动设备的响应式网页和应用程序，以及 Vue 3 的核心概念和语法。
- 通过 Vue 3 构建移动 Web 应用程序：介绍如何使用 Vue 3 构建交互式的移动 Web 应用程序，并提供最佳实践和技巧。
- 性能优化技术：涵盖各种优化技术，如图片优化、代码压缩、资源缓存、懒加载等，以及如何使用这些技术来提升应用程序的性能和用户体验。
- 项目实战：通过一个企业级实战项目——仿微信朋友圈系统来全方位讲解移动 Web 和 Vue 3 在企业级项目中的最佳实践。

本书特色

本书的特色在于它深入覆盖了移动 Web 和 Vue 3 的最佳实践，并介绍了各种性能优化技术，具体特色如下：

- 提供针对移动设备的优化：本书主要关注移动 Web 开发，因此重点介绍如何为移动设备进行优化。本书提供的各种方法和技巧能够帮助读者创建出快速、响应式的移动 Web 应用程序。
- 介绍 Vue 3 的新特性：Vue 3 是 Vue 框架的最新版本，它带来了很多新的特性和改进，例如响应式系统的重构、虚拟 DOM 的改进、Composition API 等。本书将介绍这些新特性，以及如何使用它们来开发更好的 Web 应用程序。
- 实用案例：本书通过实际案例向读者展示如何使用 Vue 3 和性能优化技术构建高效、可靠和可扩展的移动 Web 应用程序。这些案例将涵盖从简单的 Web 应用程序到复杂的企业级应用程序的各个方面。

- 性能优化技术：本书不仅详细介绍各种性能优化技术，例如懒加载、资源压缩、CDN、缓存等，还将介绍如何使用这些技术来提升 Web 应用程序的性能和用户体验。
- 企业级项目开发：本书将提供许多最佳实践和技巧，这些实践和技巧是笔者根据自己的经验和 Vue 社区的经验总结出来的，它们将帮助读者更有效地开发企业级项目并解决各种问题。

综上所述，本书提供的信息和技巧能够帮助读者深入了解移动 Web 和 Vue 3，使读者在设计和开发移动 Web 应用程序时更加自信和有效。本书介绍的各种性能优化技术能让读者创建出快速、高效、可靠且对用户友好的 Web 应用程序。

本书适用对象

本书既适合有一定前端开发基础的学生、前端开发的从业者以及自由项目开发者，也适合对 Vue 3 感兴趣的、擅于做各种 Vue 3 应用探索、想要深入了解 Vue 3 底层实现的开发者。

本书的一些默认环境和依赖说明

本书所包含的源码和项目开发调试环境为 Windows 11 操作系统；编辑器为 Sublime Text 3；调试用的浏览器为 Chrome，版本是 98；在一些案例中会使用到 Node.js，它的版本为 v-14.14.0；建议读者提前进行配置和安装。

配书资源

为方便读者使用本书，本书还提供了案例源码及 PPT 课件。读者可以扫描下方的二维码，按照页面提示把下载链接转发到自己的邮箱进行下载。如果在阅读本书的过程中发现问题，请用电子邮件联系 booksaga@163.com，邮件主题写“Vue 3 移动 Web 开发与性能调优实战”。

最后

“书犹药也，善读之可以医愚”，每本书都是一剂良药，能帮我们解决困惑，带来转机。同样，每门技术的学习都需要从理论到实战，这样才能真正理解并为自己所用。对于每一名前端工程师来说，技术的变化和更新必然会带来持久不断的学习，掌握其中的要领便能从容应对。愿各位读者在学习本书之后都能有所收获，搭上移动互联网这艘大船！

感谢在编写本书时我的家人对我的理解和帮助，尤其是我的妻子以及 3 岁的女儿！

笔　者

2023.3

目　录

第 1 章

移动 Web 开发概述

本章主要介绍移动Web开发的特点、涉及的技术以及相关开发环境的搭建，帮助读者建立概念，为后续学习打下基础。

1.1 移动互联网 Web 开发技术介绍

什么是移动 Web，Web 网页和原生 App 有何区别，移动 Web 开发有什么特点，移动 Web App 是如何工作的，本节将对这些问题逐一进行介绍。

1.1.1 移动 Web 是什么

互联网的发展总是伴随着人们上网设备的更新。在2008年之前，大多数的上网设备还是以台式计算机为主，网上资源也相对较少。那些年比较流行的论坛网站有天涯社区、猫扑社区等，搜索引擎则有百度搜索和搜狗搜索等，新闻资讯类的网站则有四大门户网站——新浪、网易、搜狐和腾讯。这些网站大多数都是以文字加图片的方式展示信息，构成了早期PC（Personal Computer，个人计算机）端网页的内容，并采用提交各类表单进行页面跳转来作为与用户的交互方式。

当时的手机虽然已经很普遍了，但是大多数的功能还是用来接打电话和收发短信，受2G移动网络的限制，使用手机上网或者进行娱乐的应用相对较少，并且其他可供使用的移动互联网软硬件产品和相关业务也较少，大部分的上网应用还是集中在PC端。

在2012年左右，随着移动端Android和iOS操作系统的出现，智能手机如雨后春笋般进入我们的生活中，并且出现了微信这种重量级的移动互联网产品业务，伴随着3G和4G移动网络的普及，中国的互联网才真正进入高速移动互联网时代。

简单总结一下，移动Web就是利用移动端浏览器承载的Web网页所呈现出来的程序App，移动Web技术就是将传统的Web开发技术（JavaScript，CSS，HTML）应用在移动端，需要注意的是，移动Web技术与Android、iOS这种原生的技术是不一样的。

1.1.2 Web 网页和原生 App 的区别

随着移动互联网的高速发展，源自用户界面的前端工程师逐渐从软件工程师中独立出来，前端开发技术也逐渐衍生出以下几种分支：

- 原生应用（Native App）开发：这类开发技术是完全使用移动端系统语言编写客户端应用，iOS系统采用Object-C或者Swift语言，Android采用Java语言。采用原生应用（或称为原生App）开发的项目得益于功能强大且丰富的原生接口，可实现较为复杂的交互需求，用户体验好，但灵活性不强，开发成本高。
- Web应用（Web App）开发：这类开发技术也称为移动Web开发或者HTML 5页面开发，是采用HTML+CSS+JavaScript语言开发的。采用Web应用开发的项目由多个前端页面组成，这些页面多采用更新的HTML 5技术，与传统的PC网页不同的是，这些前端页面有更强的适配性和性能要求，并且利用原生应用的WebView组件或者系统自带的浏览器提供应用的壳子，最终形成一个看似是App的应用程序，所以称作Web应用。Web应用中的每个页面都可以单独在移动浏览器里打开，跨平台和可移植性较强，但性能体验和功能性不如原生应用。
- HyBrid App（混合类应用）开发：这类开发技术介于原生应用开发技术和移动Web开发技术之间，是上面两种开发技术的混合版。这类应用整体上看是一个原生应用，功能性和交互性强的部分采用原生的语言开发，另外部分内容会采用WebView组件构成页面容器，采用HTML+CSS+JavaScript语言开发前端页面，同时会提供可定制化的原生应用组件和接口来让前端页面调用，拓宽前端技术的能力，最终组成一个含有原生应用开发技术和移动Web开发技术的混合类应用。

这三类App之间的关系如图1-1所示。当前比较流行的移动互联网产品，例如微信、手机淘宝、豆瓣等都是比较典型的HyBrid App。为了满足更多的动态化需求，挣脱每次发布都需要受到应用市场上架的限制，混合模式也衍生出越来越多的客户端动态化方案，其中包括以前端技术为主的微信小程序、React Native等方案。总之，无论是哪种方案和应用，移动Web开发都是非常重要且必不可少的技术，并且随着5G时代的到来，越来越丰富的移动互联网应用会进入人们的生活，这些应用的技术实现都会用到移动Web开发中。

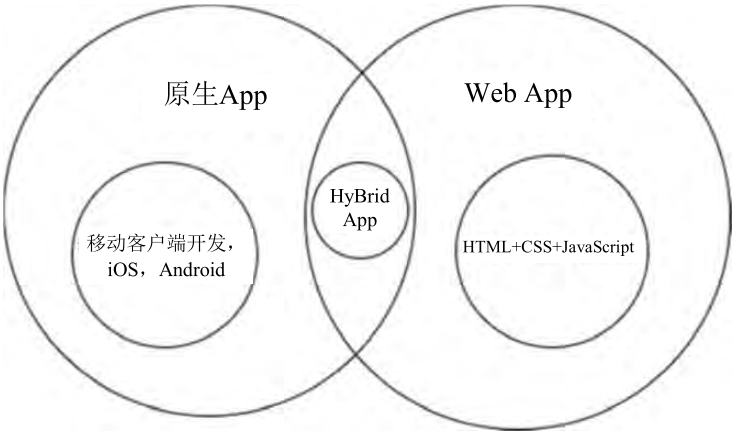

图 1-1　原生 App、Web App 与 HyBrid App 之间的关系

1.1.3　移动 Web 开发的特点

相信大部分读者或多或少都掌握了一些PC端开发技术或具有开发PC端页面的经验，实际上传统的PC端开发和移动Web开发所使用的技术栈基本上是一致的，都是采用HTML+CSS+JavaScript语言来开发的，但是从产品形态、网络环境以及性能要求上来看，它们还是有不少区别的：

- 由于移动设备屏幕较小，而要将原本在PC端的信息内容呈现在移动端，就需要进行优化和精简，因此App界面设计的复杂度比传统PC端要小，这其实降低了一定的开发难度，但是移动端的页面所运行的环境是非常多变的，不同的智能手机或移动设备的屏幕各不相同，有的屏幕大，有的屏幕小，有的采用高清屏，有的采用标清屏，所以在屏幕适配上，移动Web页面有着更高的适配性要求。
- 传统的PC端页面开发始终逃脱不了浏览器兼容性问题，从最开始的IE系列浏览器到当下流行的谷歌Chrome、火狐（Firefox）以及360浏览器等，由于各浏览器厂商的标准不同，导致前端工程师始终要和这些“不标准”斗争，浏览器兼容性问题一直是一个比较令人头疼的问题。好在目前大多数智能手机或移动设备自带的移动端浏览器都采用WebKit[①]内核，统一的标准使得移动Web开发需要处理的浏览器兼容性问题变少了一些。
- 移动设备虽然可以使用WiFi上网，但是不排除在一些关键时刻需要使用3G或4G移动网络上网，这些情况下的网络速度与固网宽带的网络速度相比还是要慢一些，更重要的是这些网络的资费要贵很多，并且当网络信号差时，会有很糟糕的用户体验，所以如何优化移动端在弱网络下的页面性能，提升用户在弱网络下的使用体验，是移动Web开发的一项非常重要的技术。
- 移动设备本身的CPU、内存以及存储设备与PC端相比，差距还是很大的。同样的一个页面在PC端上处理假如需要10毫秒，换到移动设备上可能需要几倍的处理时间，而互联网上的应用响应时间太长会导致大量用户的丢失，所以编写健壮性更强、性能更高效的代码不仅是PC端需要关注的，在移动端更需要关注。

在过去，当一个公司或者企业需要开发一个互联网产品时，首先都会想到PC端，并且以PC端的用户体验为主，如果刚好有移动端的需求，也大多是移植PC端的设计。而现在，这类现象已经悄然发生变化，PC端的业务热度已经降低，以移动端为主的业务理念逐渐成为互联网产品的研发方向，这种新的理念被称为“Mobile First”（移动优先），并延续至今。因此，移动Web开发也就变得更加重要了。

1.1.4　移动 Web App 是如何工作的

移动Web App在本质上就是利用WebView组件（本质是浏览器）提供应用的壳子，最终形成一个看似是App的应用程序。具体来说就是每个原生应用（无论iOS还是Android）都会提供WebView组件，而WebView只需要一个页面的地址就可以进行加载和渲染，同时把导航栏、菜单栏和一些不需要的按钮进行隐藏，这样就构成了一个App，具体工作流程如图1-2所示。

正如图1-2所示，掌握好HTML、CSS、JavaScript技术是开发移动Web App的关键，同时结合Vue.js框架可以高效地开发出移动Web App。

① 浏览器内核也称为排版引擎，用来让网页浏览器绘制网页的核心，常见的内核有WebKit、Gecko和Trident，同样的网页在不同的内核中可能会有不同的表现。

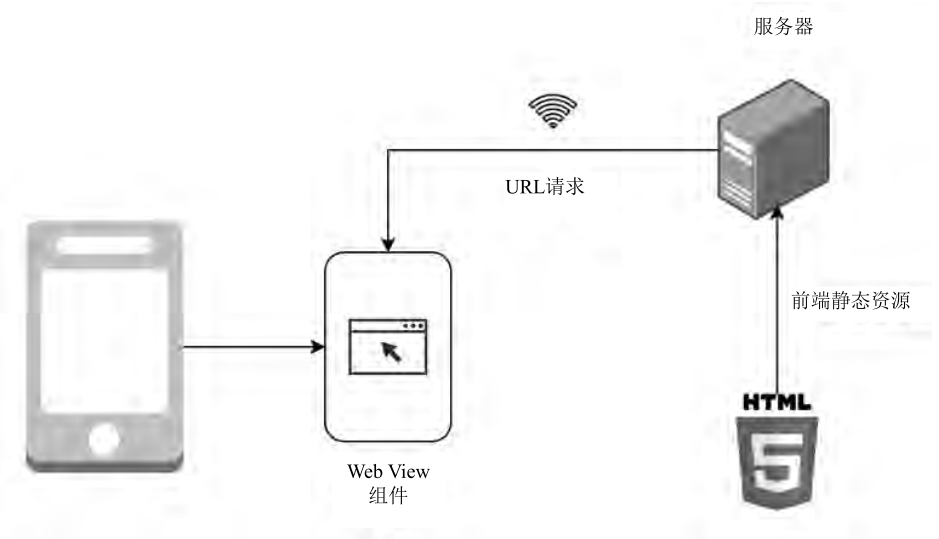

图 1-2　移动 Web App 的运行机制

1.2　移动 Web 与 HTML 5、CSS 3 和 Vue.js 的关系

本节主要介绍移动 Web 与 HTML 5、CSS 3 和 Vue.js 之间的关系。

1.2.1　移动 Web 与 HTML 5 和 CSS 3 的关系

HTML 5（简称H5）技术是定义HTML标准的最新版本，是一个新版本的HTML语言，不仅包含新的标签元素、新的属性和行为，还包含了更加强大的技术集合，涵盖了新的JavaScript Document API（例如Canvas、地理定位等），以及新的CSS版本：CSS 3，新增了如位移、转换和动画的API。

由于HTML 5相关技术是新的标准，因此对于PC端浏览器而言，支持性并不是很好，尤其是一些低版本的IE浏览器，例如IE8以下的浏览器对HTML 5的支持就非常差，所以HTML 5及其相关技术经常用在移动Web端的开发中，采用HTML 5技术开发的页面也有另一个名称，即HTML 5或者H5页面。

那么，新版本的HTML 5与之前的版本相比，到底引入了哪些新的内容呢？与上一个版本相比，HTML 5主要引入的内容如下：

- 语义：能够更恰当地用于描述内容是什么。
- 离线和存储：能够让网页在客户端本地存储数据以及更高效地离线运行。
- 多媒体：使视频（Video）和音频（Audio）成为Web页面中常见的元素。
- Canvas和2D/3D绘图：提供了更多分范围呈现页面元素的选择。
- 设备访问（Device Access）：提供了能够操作原生硬件设备的接口。
- 样式动画效果：使得CSS 3可以创作出更加复杂的前端动画。

随着HTML 5相关技术的引入，JavaScript也更新了版本，提供了一些新的数据结构和API，被称为ECMAScript 6.0（简称ES6），这些内容会在本书后面的章节进行详细讲解。

1.2.2　移动 Web 与 Vue.js 的关系

Vue.js作为当下非常流行的前端框架，本身并不限制在PC端还是移动Web端使用，并且移动Web大多数项目为单页应用，这与Vue.js的应用场景非常契合，举例来说：

- 组件化：Vue.js的组件化功能可以很好地将Web App页面的组件进行抽离和复用，减少重复性代码。
- 页面切换：利用Vue Router可以轻松实现Web App的页面之间的跳转和转场动画等效果。
- UI库：基于Vue.js技术有不少移动端的UI库，例如iView UI、Vant UI等，利用这些UI库可以快速实现页面开发和交互效果。

基于此，采用Vue.js是目前开发移动Web App项目的首选技术方案。

1.3　浏览器安装和代码环境的准备

本节介绍如何安装浏览器以及准备代码环境。

1.3.1　安装 Chrome

作为一本技术研发类的书籍，本书中有很多的代码讲解和演示，建议读者编写并运行这些代码，这样有助于对相关知识点的理解和掌握。

要运行本书中的代码，推荐使用的浏览器为谷歌Chrome，版本为79，如果无法找到指定版本，则尽量使用版本号大于70的版本。

本书中的相关演示代码都是以.html文件的形式承载的，一般情况下双击这些文件即可在浏览器中运行它们，但是也会有一些文件必须通过静态服务器的方式来运行，即采用访问http://localhost/xxx.html的方式来运行，所以读者可以在本地系统安装一个静态资源服务器，推荐使用基于Node.js的轻量级Web服务器——http-server。

1.3.2　安装 Node.js 和 http-server

使用http-server需要安装Node.js。由于本书的演示代码运行和后面的实战项目开发都需要用到Node.js，因此安装Node.js非常必要。Node.js安装起来非常简单，这里我们只讲解Windows平台下的安装步骤。

01 到Node.js官网下载安装包，Node.js官网地址：https://nodejs.org/zh-cn/download/。

02 选择长期支持版，并根据自己计算机系统是32位还是64位选择Windows安装包（.msi）文件，如图1-3所示。

03 下载完成后，双击安装包node-v12.13.1-x64.msi（注意，随着时间的推移，读者下载的最新版安装包可能会比本书中使用的版本要新），进入安装界面，如图1-4所示。

04 在安装界面中依次单击Next按钮，安装位置可自由选择，如图1-5所示。

图 1-3　下载 Node.js 的 Windows 安装包

图 1-4　Node.js 安装（1）

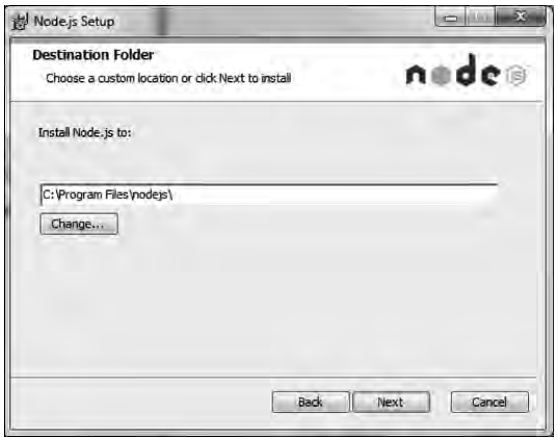

图 1-5　Node.js 安装（2）

05 最后单击Finish按钮完成安装，如图1-6所示。

图 1-6　完成安装

若要检测Node.js是否安装成功，可以使用CMD命令行工具，在Windows桌面上依次单击“开始→运行”命令，再输入CMD命令来启动这个工具。然后在“命令提示符”窗口输入node -v命令来查看Node.js版本号，如图1-7所示。

如果控制台成功地输出了Node.js的版本号，就表示安装成功。

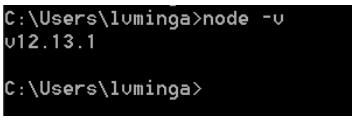

图 1-7　查看 Node.js 的版本号

在完成了Node.js的安装之后，就可以使用它自带的包管理工具NPM来安装http-server了。http-server是一个简单的、零配置的http服务，它的功能强大且使用非常简单，可以用于测试、开发和运行静态页面的服务器。

启动CMD命令行工具，使用包管理工具NPM安装http-server，输入命令如下：

```
npm install spy-debugger -g
```

安装完成之后，打开CMD命令行工具，进入文件所在的目录，执行http-server命令即可开启本地的http服务，如图1-8所示。

图 1-8　运行 http-server 开启本地的 http 服务

在浏览器地址栏输入http://127.0.0.1:8081/xx.html或者http://localhost:8081/xx.html即可访问对应的页面。

关于NPM包管理工具的使用会在“第7章　移动Web开发和调试”中具体讲解。

1.3.3　选择合适的代码编辑器

由于本书中有很多的演示代码，因此提前准备一款合适的代码编辑器是很有必要的，关于代码编辑器的选择，完全可以根据个人的喜好来定，在这里笔者介绍几种比较常用的前端代码编辑器。

- Visual Studio Code：简称VS Code，它是目前使用人数最多的代码编辑器，在2015年由微软发布，是一款比较年轻的代码编辑器，它的功能强大并且支持插件扩展，但占用内存相对较多，适合有一定前端基础的开发人员使用。
- Sublime Text：目前最新版本是Sublime Text 3，它的界面美观，体积小，运行起来非常快，也不会占用大量内存，在功能上稍逊色VS Code，但同样支持插件扩展，适合新手使用。
- WebStorm：功能强大，集成度高，想要的功能几乎都有，被誉为最智能的JavaScript代码编辑器，但体积大，占内存多，并且是一款收费的软件，相对于前面两款代码编辑器，因为增加了使用成本，所以使用人数相对少一些。
- Dreamweaver：中文名称为“梦想编织者”，是老牌子了，伴随前端而生，见证了前端的发展，内置浏览器可以实时预览是这款代码编辑器的一大特色，但其他的功能已经相对落后，现在使用的人并不多，本书不推荐使用。

在本书的实战项目中，会使用Sublime Text 3代码编辑器来编写代码。

1.4 小 结

本章主要包含移动Web开发技术概述和阅读本书的一些前置环境准备工作这两部分内容。第一部分内容包括在移动互联网的大环境下前端技术的主要分支、移动Web开发技术与PC端Web开发技术以及HTML 5的区别和联系。第二部分内容包括了Chrome浏览器和Node.js、http-server的安装与使用，以及如何选择一款合适的代码编辑器。本章虽然内容不多，却包含了移动Web开发的整体入门知识和相关的概念，为读者顺利学习本书后续的内容打下一个良好的基础。

1.5 练 习

（1）什么是HyBrid App？
（2）移动Web开发和PC端Web开发有何区别？
（3）通过本章内容的学习，谈谈你对移动Web开发技术的发展趋势、就业前景的理解。

第 2 章

HTML 5 语义化标签和属性

标签语义化简单来说就是让标签有含义，标题用<hx>标签（<h1>、<h2>等），列表用<ul>标签，这样我们一眼就可以看出网页中的每行源代码要展示哪些内容。

在HTML 5之前，一般是使用 或标签来实现大多数的网页元素，这对于整个HTML文件来说过于单一了。随着HTML 5的到来，引入一些新的标签，例如<header>、<footer>、<nav>和<section>等，这些标签更加语义化，使页面有良好的结构，无论是谁都能够看懂这块内容是什么，并且有利于搜索引擎的搜索。HTML 5这些语义化新标签的优点总结如下：

- HTML结构清晰。
- 代码可读性较好。
- 无障碍阅读。
- 搜索引擎可以根据标签的语义确定上下文和权重问题。
- 移动设备能够更完美地展现网页。
- 便于团队维护和开发。

HTML 5除了新增一些语义化标签外，同时也引入了相关的语义化属性，例如给<input>标签增加了很多实用的属性。接下来就来一一介绍这些内容。

2.1 DOCTYPE 声明

DOCTYPE声明在代码中对应的就是<!DOCTYPE>，它位于HTML文件的最前面，在<html>标签之前。这里讲解<!DOCTYPE>声明主要是为了和HTML 5版本之前的声明进行对比。

<!DOCTYPE>声明不是HTML标签，它的作用是告知Web浏览界面应该使用哪个HTML版本。在HTML 5之前的HTML 4.0.1版本，有三种设置<!DOCTYPE>声明的方式，分别说明如下：

（1）严格标准模式（HTML 4 Strict），声明的代码如下：

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4
/strict.dtd">
```

（2）近似标准模式（HTML 4 Transitional），声明的代码如下：

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org
/TR/html4/loose.dtd">
```

（3）近似标准框架模式（HTML 4 Frameset），声明的代码如下：

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

这些声明的代码都采用固定的写法，并无项目的关联性，使用时直接设置即可。

HTML 5版本的<!DOCTYPE>声明就简单多了，只有一种版本，对应的声明代码如下：

```
    <!DOCTYPE html>
```

在完成<!DOCTYPE>声明之后，在大多数情况下就要对网页的语言和编码进行设置。在网页中声明语言与编码方式是很重要的，如果网页文件没有正确地声明编码方式，那么浏览器会根据网络浏览者计算机上的设置来显示编码。我们有时浏览一些网站时会看到一些网页变成了乱码，通常就是因为没有正确地声明编码方式导致。

在HTML 4.0.1版本中，通常采用<meta>标签的方式来声明语言和编码方式，代码如下：

```
    <meta http-equiv="Content-Type" content="text/html;charset=UTF-8" >
```

在HTML 5中，可以使用对<meta>标签直接追加charset属性的方式来指定字符的编码方式，代码如下：

```
    <meta charset="UTF-8">
```

同时，在<html>标签中使用lang属性来设置语言，代码如下：

```
    <html lang="zh-CN">...</html>
```

需要说明的是，在<!DOCTYPE>声明和<meta>标签中设置的属性都是不区分字母大小写的，例如可以将UTF-8换成utf-8，<!DOCTYPE html>换成<!doctype html>。

接下来，创建一个新的HTML 5页面，并添加上<!DOCTYPE html>声明和语言及编码方式的设置，如示例代码2-1-1所示。

示例代码 2-1-1　第一个 HTML 5 页面

```
    <!DOCTYPE html>
    <html lang="zh-CN">
    <head>
      <meta charset="UTF-8">
      <title>HTML 5</title>
    </head>
    <body>

    </body>
    </html>
```

上面代码是完整的HTML 5代码，可以直接在浏览器中运行，后续有关标签和相关属性的讲解会以此为基础。

2.2　<header>标签

<header>标签（也可称为<header>元素）是HTML 5引入的新标签之一，如果翻译成中文，那么可以理解成头部内容或者页眉内容。顾名思义，我们可以将网页最开始的部分内容放在<header>

标签里面来显示，可以在示例代码2-1-1的<body>里新增<header>标签，如示例代码2-2-1所示。

示例代码 2-2-1　<header>标签

```
<header>
  <h1>I am header</h1>
  <p>header content</p>
</header>
```

<header>标签在样式上和 是一致的，都属于区块级元素，只是在语义上有所区别，所以在规范上来说：

- <header>标签应该作为一个容器，负责HTML页面顶部内容的显示，它可以有很多子元素。
- 在一个HTML页面中，某些业务逻辑情况下可以定义多个<header>标签，数量不受限制。
- 尽量不要把<header>标签放在<footer>标签中或者另一个<header>标签内部。

2.3　<footer>标签

同<header>标签对应的是<footer>标签，正所谓“一头一尾”。<footer>标签也是HTML 5引入的新标签之一，如果翻译成中文，那么可以理解成底部内容或者是页脚内容。顾名思义，我们可以将网页结尾的部分内容放在<footer>标签里面来显示。接下来新增<footer>标签，如示例代码2-3-1所示。

示例代码 2-3-1　<footer>标签

```
<footer>
  <p>Posted by: 移动Web开发实战</p>
  <p>Contact information: <a href="mailto:someone@example.com"> someone@example.com
</a>.</p>
</footer>
```

<footer>标签在样式上和 标签没有什么区别，但是在使用时需要注意语义化和规范：

- 根据语义化的规范，<footer>标签大多数包括多个子元素，有网站的所有者信息、备案信息、姓名、文件的创建日期以及联系信息等。
- 在一个HTML页面中，某些业务逻辑情况下可以定义多个<footer>标签，并且在每个<section>标签中都可以有一个<footer>标签，这一点不受限制。
- 尽量不要把<footer>标签放在<header>标签中或者另一个<footer>标签内部。

<footer>标签通常是在页面底部，与之搭配的CSS样式可以采用fixed定位，代码如下：

```
footer {
  position:absolute;
  bottom:0;
  width:100%;
  height:100px;
  background-color: #ffc0cb;
}
```

将网站的所有者信息、备案信息等放在<footer>中并置于页面底部，这是很多网站的标配。

2.4　<section>标签

<section>标签是HTML 5引入的另一个语义化标签，作用是对页面上的内容进行分块。这里的分块主要是按照功能来分，比如一个新闻消息的列表展示页可分为国际版块、娱乐版块、体育版块，等等。每一个板块都可以使用<section>来划分，而每个版块都需要有自己的标题和内容并且相对独立。

例如图2-1所示的场景页面，我们使用<section>标签进行划分。<section>标签也没有特殊的样式，使用起来和 标签是一样的，如示例代码2-4-1所示。

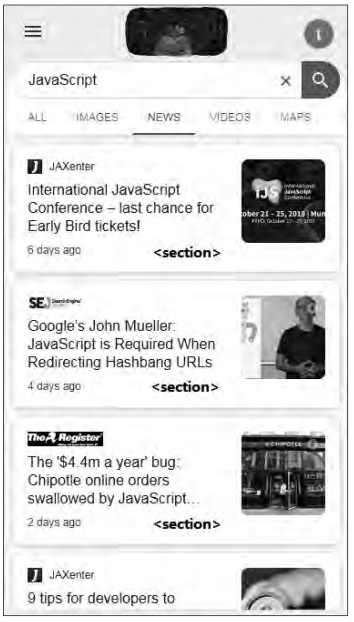

图 2-1　<section>使用场景

示例代码 2-4-1　<section>标签

```
<section>
  <h3>Title</h3>
  <p>section info</p>
  <img src="test.png" />
</section>
```

下面总结一下<section>标签的使用场景和规范：

- 使用<section>标签时，里面的内容一般要搭配标题和正文等，例如<h1>~<h6>或者<p>标签。
- 每个<section>标签都是一个独立的模块，这些独立模块内部不应该再嵌套<section>标签，但是多个<section>标签可以并列使用。
- <section>标签不应当作一个容器元素，它的语义化更强一些，当无法找到使用<section>标签的充分理由时，尽量不要使用。

2.5　<nav>标签

<nav>标签是HTML 5引入的另一个语义化标签，用于表示HTML页面中的导航，可以是页面与页面之间的导航，也可以是页内段与段之间的导航。<nav>所代表的导航一般是位于页面顶部的横向导航，或者是面包屑导航，如图2-2所示。

图 2-2　<nav>导航

由于导航的性质，大部分导航内部都由一个列表组成，也称之为导航列表。<nav>内部可以用<ul>或者<ol>来实现导航元素的布局，如示例代码2-5-1所示。

示例代码 2-5-1　<nav>标签

```
<nav>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">About</a></li>
    <li><a href="#">Contact</a></li>
  </ul>
</nav>
```

下面总结一下<nav>标签的使用场景和规范：

- <nav>标签中一般会放一些<a>标签链接元素来实现单击导航的效果，但是并不是所有的链接都必须使用<nav>标签，它只用来将一些功能性强的链接放入导航栏。
- 一个网页也可能含有多个<nav>标签，例如一个是网站内页面之间的导航列表，另一个是本页面内段与段之间的导航列表。
- 对于移动Web的页面，<nav>标签也可以放置在页面底部来代表页面内的导航，例如微信App底部的“微信”“通讯录”“发现”和“我的”4个导航链接。

2.6　<aside>标签

HTML 5的<aside>标签用来表示与当前页面内容相关的部分内容，通常用于显示侧边栏或者补充的内容，如目录、索引等。在一些场景下，可以将它理解成一个侧边的导航栏，如图2-3所示。

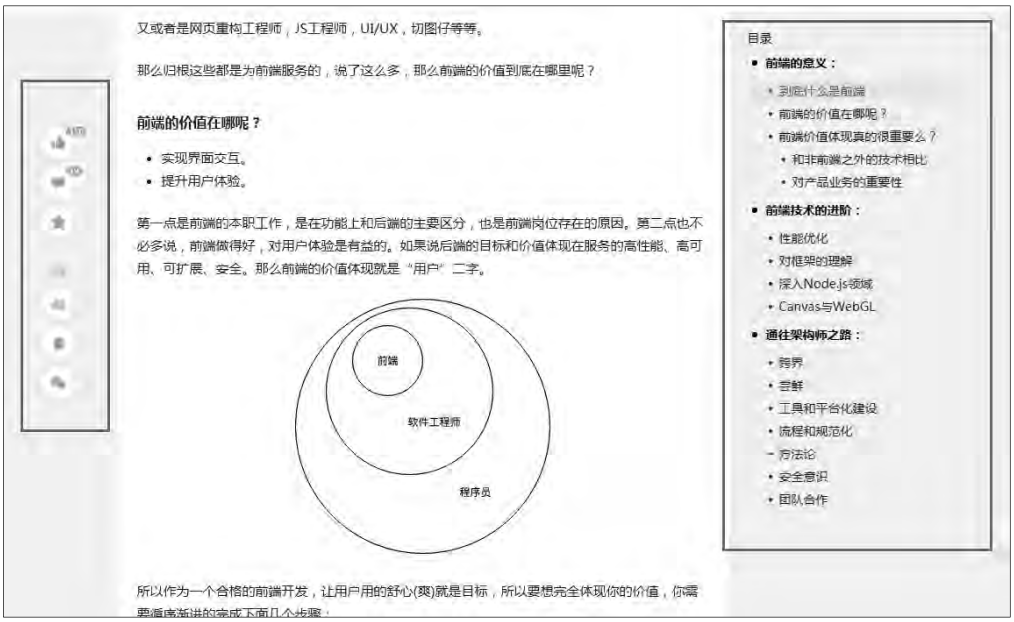

图 2-3　<aside>侧边栏

如果采用<aside>标签来实现侧边栏，与一般的 标签在样式上没有区别，如示例代码2-6-1所示。

示例代码 2-6-1　<aside>标签

```
<aside>
  <h2>标题1</h2>
  <ul>
    <li>目录1</li>
    <li>目录2</li>
  </ul>
  <h2>标题2</h2>
  <ul>
    <li>目录1</li>
    <li>目录2</li>
  </ul>
</aside>
```

<aside>标签也可以作为<section>标签中独立模块的一部分，用来表示主要内容的附属信息，其中的内容可以是与当前文章有关的资料、名词解释等，如示例代码2-6-2所示。

示例代码 2-6-2　<aside>标签和<section>标签

```
<section>
  <h1>文章的标题</h1>
  <p>文章的正文</p>
  <aside>文章相关的资料、名词解释等</aside>
</section>
```

下面总结一下<aside>标签的使用场景和规范：

- <aside>标签就像它的名字一样，在页面的一侧，其中的内容可以是友情链接、博客中的其他文章列表、广告单元等。
- <aside>标签也可以和<section>标签搭配使用，作为单个独立模块的附加信息来显示。

2.7　语义化标签总结

前面几个小节介绍了HTML 5引入的一些新的语义化标签，其中包括用来呈现页面头部的<header>标签和页面尾部的<footer>标签、作为独立模块显示的<section>标签、页面导航<nav>标签和侧边栏<aside>标签，这几个标签在页面中的具体用法如图2-4所示。

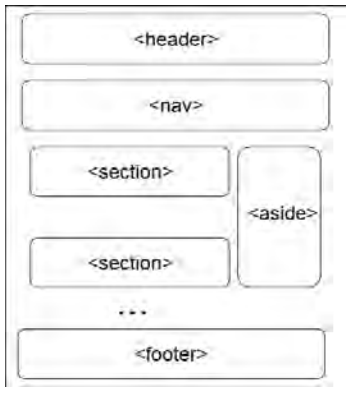

图 2-4　语义化标签总结

HTML 5之所以引入这些新的语义化标签，主要是为了让HTML代码更加规范和语义化。每当我们开始编写一个前端页面时，首先在心里应该有一个思路：如何将页面进行功能划分，划分之后如何按照每个模块的展示内容来选择合适的HTML 5语义化标签。

当然，有些程序员在网页呈现任何内容都统一使用 标签，我们称这种现象为“标签选择困难症”，虽然以这种方式使用标签并不会影响网页代码的运行，但是当其他程序员来阅读这些代码时，就会感到很乱。

HTML 5语义化标签的意义不仅在于更方便开发人员阅读代码文件和理清代码结构，而且对浏览器而言，能够更清晰地识别网页的结构。同时良好语义化代码的网站对于搜索引擎优化（Search

Engine Optimization，SEO）功能更加友好，能够让搜索引擎清晰地捕捉到网页的主、次模块和内容，所以建议程序员尽可能使用语义化标签来构建HTML页面。

当然，上述所讲解的标签并不是所有的HTML 5新引入的标签，而是与语义化概念关系比较紧密的一些标签。

2.8　HTML 5 其他新增的标签

HTML 5 新增了不少标签，下面介绍几种有代表性的标签。

2.8.1　<progress>标签

HTML 5中的<progress>标签是一个非常实用的标签，它表示一段进度条，可以用在需要显示进度的程序中，例如在需要等待或者加载的场景中使用。该标签有以下两个属性可以进行设置：

- max：该属性描述了<progress>标签所表示的任务一共需要完成多少工作。
- value：该属性用来指定进度条已完成的工作量。

如果没有value属性，那么进度条的进度为“不确定”，也就是说进度条不会显示任何进度，我们无法估计当前的工作会在何时完成（比如在下载一个未知大小的文件时或者请求数据时），此时<progress>元素会呈现出一个动态的效果。

下面使用代码来演示<progress>标签的用法，如示例代码2-8-1所示。

示例代码 2-8-1　<progress>标签

```
设置进度：
<progress value="45" max="100"></progress>
<br>
不设置进度：
<progress></progress>
```

在浏览器中运行后，效果如图2-5所示。

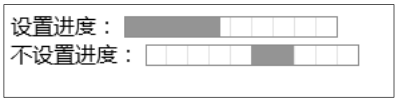

图 2-5　<progress>标签

从图2-5中可以观察到上面的<progress>标签中深色部分表示已完成的量，而下面的<progress>标签的深色部分其实是一个不断左右移动的滑块，用来表示处于等待中的进度条。读者可以在浏览器中运行这个演示代码，就会看到运行时的效果。

在Chrome浏览器中，如果想修改<progress>标签的样式，例如大小和颜色，那么可以使用如下代码：

```
progress::-webkit-progress-bar {         /* 控制进度条背景的样式 */
  height: 10px;
  background-color:#d7d7d7;
}
progress::-webkit-progress-value {     /* 控制进度条值的样式 */
  height: 10px;
  background-color:orange;
}
```

需要注意的是，在上面的代码中采用了指定浏览器的前缀，所以只对Chrome浏览器有效，对于IE或者Firefox浏览器则是无效的。

2.8.2　<picture>标签

继HTML 5新增了许多新的标签之后，在子版本HTML 5.1中（截至2023年，HTML 5共有3个子版本，分别为HTML 5.1、HTML 5.2和HTML 5.3，其中HTML 5.3是最新的版本）又引入了几个“更时尚”的标签，其中就包括<picture>标签。在传统的PC端网页中，显示一张图片大多数会采用<img>标签，但是随着移动互联网的发展，网页越来越多地运行在屏幕大小多变、分辨率不同的移动端设备中，<picture>标签提供了一种新的图片显示方案，可以为当前移动设备选择更加适合的图片。

<picture>标签主要用法是在其内部创建若干个可以设置特性的<source>元素，每个<source>元素可以设置不同的srcset属性，代表不同的图片地址，同时可以设置不同的media属性，代表符合的特定条件。<picture>标签使用方法如示例代码2-8-2所示。

示例代码 2-8-2　<picture>标签

```
<picture>
  <source srcset="large.jpg" media="(min-width: 400px)">
  <source srcset="medium.jpg" media="(min-width: 300px)">
  <img srcset="small.jpg">
</picture>
```

<source>元素的属性及其含义如下：

- srcset：该属性类似<img>标签的src属性，用来设置图片的地址。
- media：该属性也叫作媒体查询，它的结果是一个布尔类型，用来判断是否满足查询条件，当条件成立时便会使用srcset设置的图片来显示。更多关于媒体查询的用法会在“第8章　移动Web屏幕适配”中进行讲解。
- type：该属性为<source>元素的srcset属性设置的图片资源指定一个MIME类型。如果当前设备不支持指定的类型，那么就不会使用该srcset设置的图片。

上面代码的具体含义：当屏幕宽度大于300px且小于400px时，会选用medium.jpg这张图片来显示；当屏幕宽度小于300px时，会选用small.jpg这张图片来显示；其余情况下则选用large.jpg这张图片来显示。在每一个<picture>标签中，都需要有一个<img>标签表示默认图片，当其他的<source>条件都不满足时，就会使用默认图片来显示。

针对不同移动设备加载不同的图片不仅能节约带宽，而且显示效果更好，即便图片差别不大，也可以在细节上提升用户体验。

2.8.3　<dialog>标签

<dialog>标签是在子版本HTML 5.2中引入的标签。<dialog>标签的作用是提供一个弹出的对话框元素，该元素的位置默认为在屏幕上左右居中，同时包括一个黑色的边框。该元素还具有open属性，用来表示显示对话框，但是在大多数情况下需要通过JavaScript来控制。

<dialog>标签的使用如示例代码2-8-3所示。

示例代码 2-8-3　<dialog>标签

```
<dialog id="dialog" open>
  这是一个弹出对话框元素
</dialog>
```

在浏览器中运行后，效果如图2-6所示。

图 2-6　<dialog>标签的使用效果

在上面的代码中，open属性意味着该对话框是可见的。假如没有这个属性，那么这个对话框就会隐藏起来，直到我们使用JavaScript来显示它。

<dialog>标签对应的DOM元素有以下方法可供JavaScript来调用：

- show()和showModal()：这两个方法相同之处都是打开对话框，都会给<dialog>标签添加一个open属性。唯一区别就是show()方法会按照它在DOM中的位置弹出对话框，没有遮罩，而showModal()方法会出现遮罩，并且自动进行按键监控（即按了Esc键，弹出的对话框就会关闭）。在大多数情况下，应使用更智能的showModal()方法。
- close()：关闭对话框，即删除open属性，并且可以携带一个参数作为额外数据，传入的值可以通过DOM对象dialog.returnValue来获取。

<dialog>标签同时提供了两个事件：

- close事件：当关闭弹出的对话框时触发。
- cancel事件：当按下Esc键关闭对话框时触发。

使用JavaScript来操作弹出对话框的隐藏和显示，如示例代码2-8-4所示。

示例代码 2-8-4　使用 JavaScript 操作<dialog>标签

```
<button onclick="openDialog()">打开弹出对话框</button>
<button onclick="closeDialog()">关闭弹出对话框</button>
<dialog id="dialog">这是一个弹出对话框元素</dialog>
<script type="text/javascript">
  // 获取弹出对话框的DOM对象
  var dialog = document.getElementById('dialog')
  // 打开弹出对话框的回调方法
  function openDialog() {
    dialog.showModal()
  }
  // 关闭弹出对话框的回调函数
  function closeDialog() {
    dialog.close()
  }
  dialog.addEventListener('close', function(){
    console.log('弹出对话框被关闭')
  })
</script>
```

标签在实际使用时都会对自定义的样式采用任意CSS样式来重置它的默认样式，但是对于含有遮罩层的弹出对话框，可以采用伪元素的方式去定义遮罩框的样式，代码如下：

```
dialog::backdrop {
  background-color: rgba(41, 107, 255, 0.4);/*定义遮罩框为40%透明度的蓝色*/
}
```

2.9　HTML 5 新增的标签属性

HTML 5新增了不少的标签属性，大多数都是基于<input>标签的属性。作为HTML 5页面中一个与用户交互的重要入口，<input>标签基本上在每个网页的页面中都会被用到。除了新增的<input>标签的属性外，还有新增的<script>标签的async和defer属性（在HTML 4.0.1中提出，在HTML 5中完善），这些属性也是比较重要的。下面我们来一一讲解。

2.9.1　<input>的 type 属性

在HTML 5中，为<input>标签新增了一些type属性值，用来丰富文本框的类型，如示例代码2-9-1所示。

示例代码 2-9-1　<input>的 type 属性

```
<fieldset>
   <legend>HTML 5中新增的input type类型</legend>
    <form>
        邮箱：<input type="email"><br />
        手机号码：<input type="tel"><br />
        网址：<input type="url"><br />
        数字：<input type="number"><br />
        搜索框：<input type="search"><br />
        拖动滑块：<input type="range"><br />
        时间：<input type="time"><br />
        日期：<input type="date"><br />
        几年几月：<input type="month"><br />
        几年几周：<input type="week"><br />
        颜色：<input type="color"><br />
    </form>
</fieldset>
```

上面的代码展示了HTML 5中新增的几种type类型，将这段演示代码在PC端的Chrome浏览器来运行一下（Chrome浏览器使用本书开头指定的版本，即至少70版本以上），即可看到每种type的显示效果，如图2-7所示。

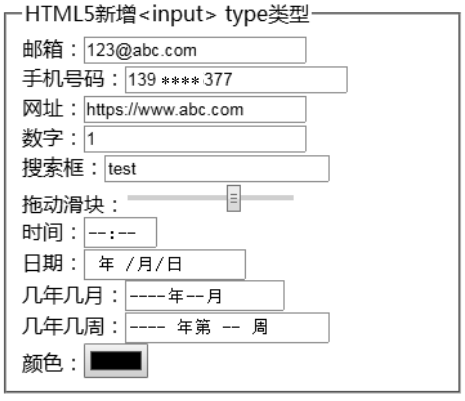

图 2-7　HTML 5 新增<input>的 type 类型

由于移动端有不同的iOS和Android平台，以及不同的WebView内核，因此<input>输入框元素在移动端浏览器中的表现就比较多元化一些，下面列举几个可以明显看出区别的类型。

- 在iOS中使用type="date"，显示的结果如图2-8所示。
- 在iOS中使用input type="tel"，显示的结果如图2-9所示。

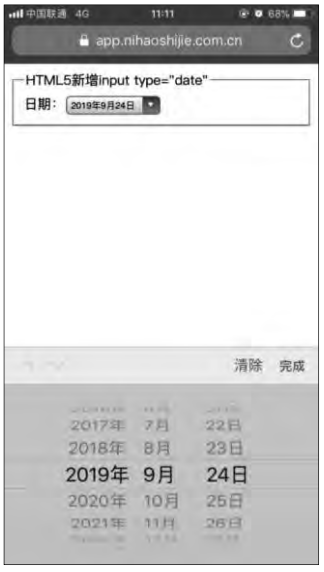

图 2-8　在 iOS 中使用 input type="date"的显示结果

图 2-9　在 iOS 中使用 input type="tel"的显示结果

如果<input>是一个日期的类型，那么它会自动调用手机端的时间选择器；如果<input>是电话类型，那么当调用键盘时，会自动转换成数字键盘，并且无法输入数字之外的字符。关于移动端<input>各种type的效果，笔者建议读者拿起手机来真实体验一下，可以在手机端执行示例代码2-9-1来加深印象。

2.9.2　<input>文件上传功能

在HTML 5之前，可以使用<input type="file">来实现文件或者图片的上传，在HTML 5中使用<input>标签在移动Web端会调用相册面板，在PC端则会打开文件选择窗口，同时HTML 5在<input>标签上扩展了一些属性来丰富上传功能：

- accept：限制上传文件的类型，image/png和image/gif表示只能上传图片类型，并且扩展名是png或gif；image/*表示任何图片类型的文件。当然，accept属性也支持 .xx，表示扩展名标识的限制，例如accept=".pdf,.doc"。
- multiple：设置是否支持同时选择多个文件，选择支持后，files将会得到一个数组。例如在移动Web端调用相册面板时，可以进行多选。
- capture：该属性可以调用系统默认相机、摄像机和录音功能，同时还有其他取值：
  - capture="camera" 表示相机。
  - capture="camcorder" 表示摄像机。
  - capture="microphone" 表示录音。

需要注意的是，在移动Web端给<input>标签设置了capture属性后，当<input>被单击之后，将会直接调用对应的模块，而不会让用户选择。设置了capture属性之后，multiple也将会被忽略。<input>的文件上传如示例代码2-9-2所示。

示例代码 2-9-2　<input>的文件上传

```
    <p>选取多张照片：<input type="file" accept="image/*" multiple="multiple">

    <p>从相机选取图片：<input type="file" accept="image/*" capture="camera"
multiple="multiple">

    <p>从麦克风选取声音：<input type="file" accept="audio/*" capture="microphone">

    <p>从录像机选取（录制）视频：<input type="file" accept="video/*" capture="camcorder">
```

建议读者尝试在手机端体验这段代码的真实效果，在iOS手机端体验的效果如图2-10和图2-11所示。

图 2-10　在 iOS 中使用 multiple 后的效果

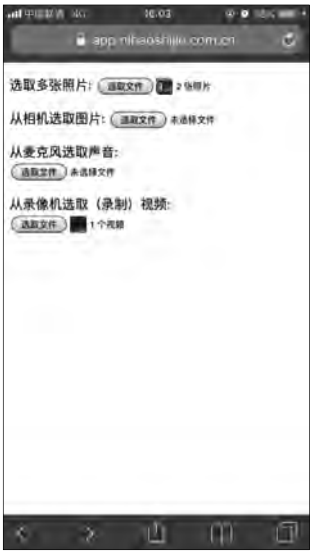

图 2-11　在 iOS 中选择完成后的效果

在真实体验后会发现，capture="microphone"这个属性在移动端的支持度并不是很好，例如iOS 12版本的Safari浏览器就不支持该属性设置的选项。

在获取了相关的文件之后，怎么获取所上传的文件呢？我们可以给<input>绑定一个onchange事件，以便在代码中获取对应的文件数据，如示例代码2-9-3所示。

示例代码 2-9-3　JavaScript 获取<input>数据

```
    <p>选取多张照片：<input type="file" accept="image/*" multiple="multiple" id="uploader">

    <script>
      var recorder = document.getElementById('uploader');
      recorder.addEventListener('change', function(e) {
        var file = e.target.files;
        console.log(file)
        // 这里可以获取文件数据
      });
    </script>
```

2.9.3　<input>其他新增属性

1. autocomplete属性

autocomplete属性规定表单或输入字段是否应该自动完成。在启用自动完成之后，浏览器会基于用户之前的输入值自动填写。在默认情况下，大多数浏览器都启用这项功能。需要注意的是，autocomplete属性适用于这些<input>类型：text、search、url、tel、email、password、datepickers、range以及color，如示例代码2-9-4所示。

示例代码 2-9-4　autocomplete 属性

```
Name: <input type="text" name="name" autocomplete="on"><br />
E-mail: <input type="email" name="email" autocomplete="off"><br />
```

2. autofocus属性

autofocus属性是布尔类型的属性。如果设置该属性，那么当页面加载时<input>元素应该自动获得焦点，如示例代码2-9-5所示。

示例代码 2-9-5　autofocus 属性

```
Name:<input type="text" name="name" autofocus>
```

这里需要注意，对于布尔类型的属性，HTML 5规范规定：元素的布尔类型属性如果有值，就是true，如果没有值，就是false。因此，在声明布尔类型属性时不用赋值，autofocus等同于autofocus="true"或者autofocus="xxx"。

3. min和max属性

min和max属性规定<input>标签的最小值和最大值。min和max属性适用的输入类型：number、range、date、month、time以及week，如示例代码2-9-6所示。

示例代码 2-9-6　min 和 max 属性

```
<!--只能输入1980-01-01之前的日期:-->
<input type="date" name="beforeday" max="1979-12-31">
<!--只能输入2000-01-01之后的日期:-->
<input type="date" name="afterday" min="2000-01-02">
<!--只能输入1-5(包括1和5)数字的:-->
<input type="number" name="range" min="1" max="5">
```

4. pattern属性

pattern属性用于检查<input>标签内容值的正则表达式。适用于以下输入类型：text、search、url、tel、email和password。例如，只能包含3个字母的输入内容（无数字或特殊字符），如示例代码2-9-7所示。

示例代码 2-9-7 pattern 属性

```
<input type="text" name="code" pattern="[A-Za-z]{3}" title="Three letter code">
```

5. placeholder属性

placeholder属性用以描述输入字段预期值的提示（样本值或有关格式的简短描述），该提示会在用户输入值之前显示在输入字段中，在输入任何值后自动消失。

在HTML 5之前，实现一个输入框的placeholder需要借助CSS和JavaScript来实现，现在有了HTML 5，一个placeholder属性即可实现效果，减少了重复性的开发工作。目前placeholder属性适用于以下输入类型：text、search、url、tel、email以及password，如示例代码2-9-8所示。

示例代码 2-9-8　placeholder 属性

```
Name:<input type="text" name="name" placeholder="请输入名字">
```

6. required属性

required属性是布尔类型的属性。如果设置这个属性，那么规定在提交表单之前必须填写输入字段。required属性适用于以下输入类型：text、search、url、tel、email、password、number、checkbox、radio和file，如示例代码2-9-9所示。

示例代码 2-9-9　required 属性

```
Username: <input type="text" name="usrname" required>
```

对于所有的内容限制类属性，例如pattern、max、min和required等，如果输入的值非法，那么当此<input>放在表单<form>中作为表单元素提交时，会有错误提示信息，如图2-12所示。或者当鼠标移到非法元素上时，也会有错误提示信息，如图2-13所示。

图 2-12　错误提示信息（1）

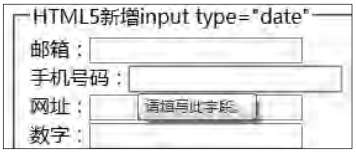

图 2-13　错误提示信息（2）

2.9.4　<script>的 async 和 defer 属性

在讲解<script>的async和defer属性之前，我们首先需要了解一下浏览器渲染页面的原理，如图2-14所示。

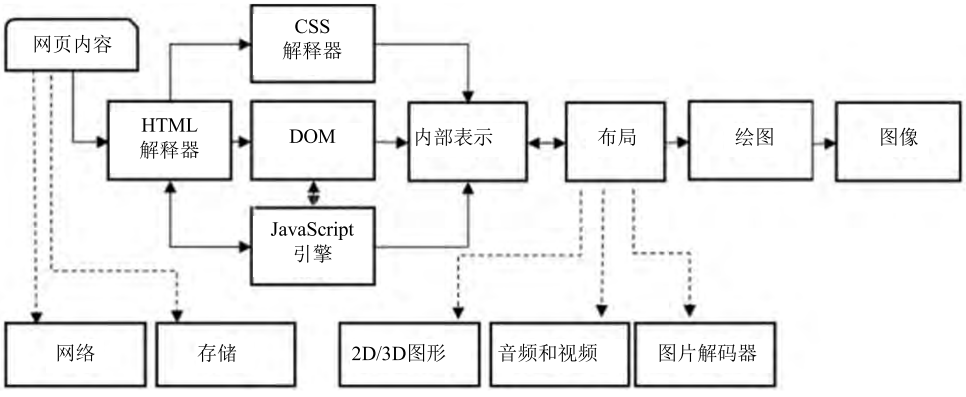

图 2-14　浏览器渲染页面的原理

在图2-14的左半部分，当浏览器获得HTML页面内容进行解析并渲染时，就会出现阻塞问题，下面来详细解释一下：

- 当浏览器获得服务端返回的HTML页面内容时，总是会从上往下解析并渲染页面。
- 一般的HTML页面，一些样式文件（CSS）和脚本文件（JavaScript）会放在头部<head>标签中被导入。
- 当浏览器解析到头部的CSS和JavaScript标签时，如果遇到的是外部链接，就会下载这些资源。
- 暂不提外部CSS资源，这里我们只说JavaScript外部资源，即当浏览器遇到外部的<script src="xx.js">时，就会暂停解析后面的HTML页面内容，先发起请求获取当前页面内容，而后解析获取的页面内容并执行。
- 所以，<script>标签就会阻塞正常的HTML页面内容的解析和渲染，尤其当<script>标签导入的外部内容很大时，这种阻塞问题就更加明显，将会导致HTML页面加载变慢，白屏时间变长。

为了解决<script>阻塞页面解析和渲染的问题，HTML 5引入了<script>标签的defer和async属性。这两个属性都是布尔类型的属性。

1. defer属性

当浏览器遇到设置了引入外部资源(注意只针对外部资源)<script src="xx.js" defer>的标签时，就不再阻止解析，会另外并行去下载对应的文件，当下载完成之后也不会立刻执行，而是等到整个HTML页面解析完成后再执行。如果页面有多个<script src="xx.js" defer />，就会按照定义的顺序执行，这一点很重要，如示例代码2-9-10所示。

示例代码 2-9-10　defer 属性

```
<head>
<script type="text/javascript" src="abc.js" defer></script>
<script type="text/javascript" src="efg.js" defer></script>
</head>
```

2. async属性

async属性和defer属性类似，都用于改变处理脚本的行为。与defer属性相同的是，async属性只适用于外部资源，并告诉浏览器立即下载文件。但与defer属性不同的是，标记为async属性的脚本并不保证按照定义它们的先后顺序执行，如示例代码2-9-11所示。

示例代码 2-9-11　async 属性

```
<head>
<script type="text/javascript" src="abc.js" async></script>
<script type="text/javascript" src="efg.js" async></script>
</head>
```

在上面的代码中，如果efg.js文件比abc.js文件先下载完成，那么efg.js文件会在abc.js文件之前执行，因此确保两者之间互不依赖这一点非常重要。指定async属性的目的是不让页面等待两个脚本文件下载和执行，从而异步加载页面的其他内容，因此建议在指定async属性的脚本内容中不要有修改DOM的逻辑。

同时，如果感觉这两个属性并不需要设置，或者并不需要延迟加载，那么最优的方法就是老老实实将外部资源的<script>放在页面底部，例如在</body>标签上面，这样就不会影响HTML页面的解析和渲染。

2.10　小　　结

就前端技术而言，每一个新标准的诞生，都会带来许多与浏览器兼容性有关的问题，HTML 5也不例外。在日常使用HTML 5相关的新特性时，要尤其注意浏览器的兼容性。目前PC端对于HTML 5的支持度已经很好了，而且样式表现比较统一，但是在移动端，比如在iOS和Android平台上对于同一个特性就可能有不同的表现方式。最稳妥的还是多找几个不同型号的移动设备来进行测试，正所谓“标签由我们定，是否真实跟进就是各大浏览器厂商的问题了”。

本章主要分为三个部分：第一部分主要讲解了HTML 5新引入的一些标签，这些标签包括DOCTYPE声明、<header>标签、<footer>标签、<section>标签、<nav>标签和<aside>标签，并使用了示例代码来演示它们的用法；第二部分主要讲解了HTML 5引入的一些新的标签属性，这些属性主要是和<input>配合使用，其中包括使用最多的type属性、文件上传所用属性以及一些其他的属性，如autocomplete属性、autofocus属性、min和max属性、pattern属性、placeholder属性、required属性，并使用代码来演示它们的用法；第三部分讲解了<script>标签阻塞页面解析和渲染的原因，并给出了解决办法，即对<script>设置async和defer属性。

2.11　练　　习

（1）什么是HTML标签语义化？

（2）<section>标签使用的规范是什么？

（3）HTML 5中布尔类型的属性有什么特点？

（4）HTML 5使用<input>标签上传多张图片对应如何设置？

（5）为什么不建议在<head>标签内使用引入外部资源的<script>标签？