PACKT PUBLISHING

快速解决移动Web开发的常见问题

HTML5 移动Web开发实战

HTML5 Mobile Development Cookbook

60多种技巧，帮助你构建针对iPhone 5、Android、Windows Phone和Blackberry的快速、响应式HTML移动网站

人民邮电出版社
POSTS & TELECOM PRESS

石川 著
刘旸 刘先宁 译

HTML5 移动Web开发实战

60多种技巧，帮助你构建针对iPhone 5、Android、Windows Phone和Blackberry的快速、响应式HTML移动网站

移动互联网的快速发展，对Web开发提出了新的需求和挑战。HTML5的诞生，为移动Web功能提供了更多的支持和可能性。当前，手机设备发展迅猛、屏幕尺寸各不相同、存储和性能仍有局限性，如何结合HTML5的功能，在多平台上创建高性能、响应式的移动网站，是Web开发者所要面对的首要挑战。

本书提供了应对这一挑战的解决方案。通过阅读本书，你将能够：

- 了解如何有效地利用HTML5最新的移动Web功能，横跨多个移动平台；
- 精通多平台的移动网站设计技术；
- 针对跨平台的开发，使用最佳实践；
- 创建高性能、响应式的移动网站；
- 有效并且高效地调试移动网站；
- 在移动网站中集成Facebook和Twitter；
- 使用HTML5富媒体功能，例如针对移动网站的Canvas和SVG；
- 构建基于HTML5地理位置信息的应用；
- 了解新兴的移动网站技术，如AJAX 2。

本书为那些致力于实现高性能、响应式、跨平台的HTML5移动网站的程序员量身打造，也适合对移动Web开发感兴趣的读者学习参考。

人民邮电出版社－信息技术分社
http://weibo.com/ptpitbooks

分类建议：计算机/程序设计/HTML5
人民邮电出版社网址：www.ptpress.com.cn

本书特色

- 简明手册的风格；
- 精心挑选的任务和问题；
- 高效解决问题的清晰指导；
- 举一反三，将解决方案用于其他情况。

作者简介

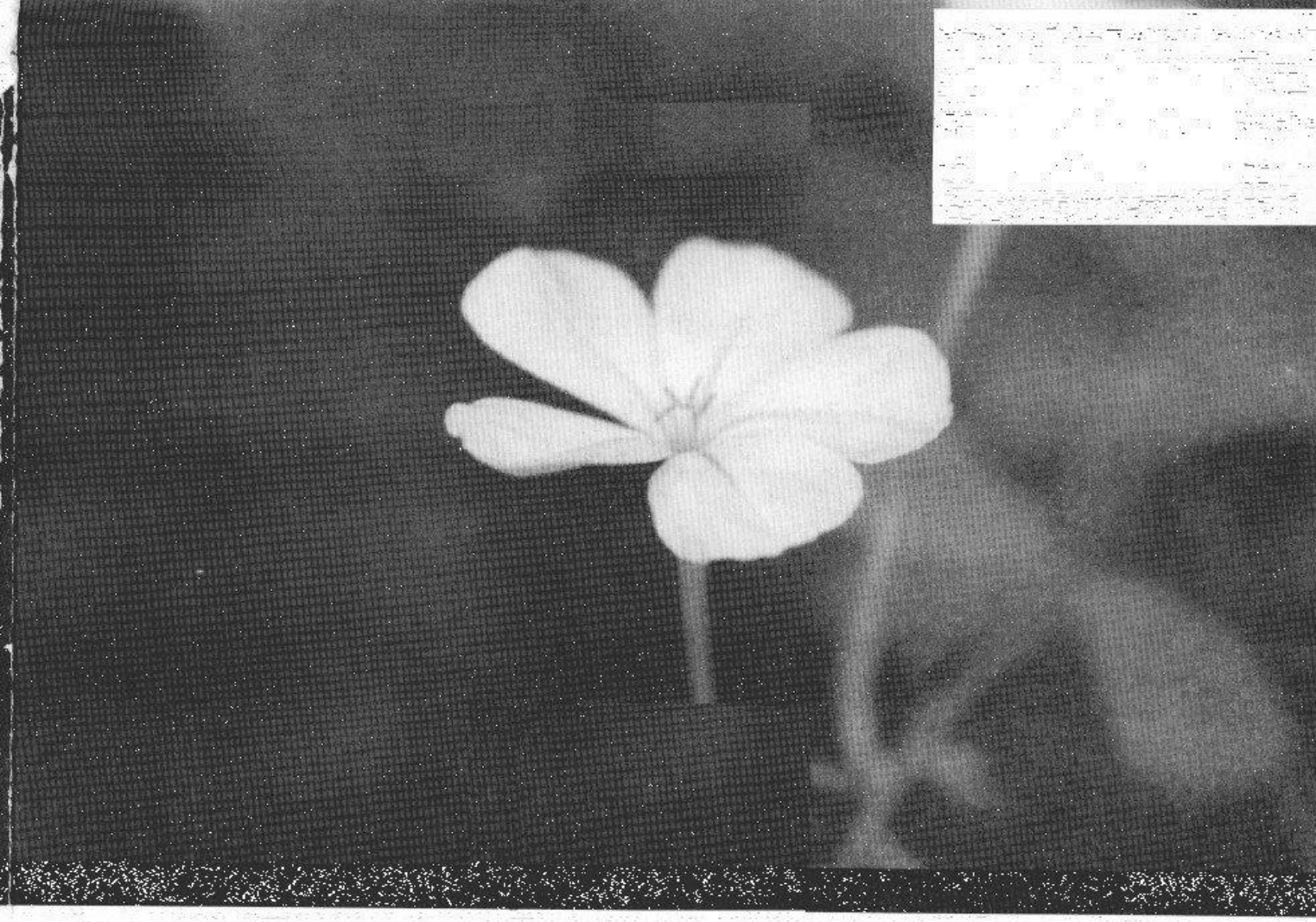
石川有超过5年的Web开发经验，他是HTML5 Boilerplate项目的成员，Mobile Boilerplate（http://h5bp.com/mobile）的主要开发人员，JavaScript Patterns项目（http://shichuan.github.com/javascript-patterns）的所有人。

美术编辑：王建国

ISBN 978-7-115-31328-7

ISBN 978-7-115-31328-7

定价：45.00 元



HTML5 移动Web开发实战

石川 著
刘旸 刘先宁 译

人民邮电出版社
北京

内容提要

移动互联网的快速发展，对 Web 开发提出了新的需求和挑战。HTML5 的诞生，为移动 Web 功能提供了更多的支持和可能性。当前，手机设备发展迅猛、屏幕尺寸各不相同、存储和性能仍有局限性，如何结合 HTML5 的功能，在多平台上创建高性能、响应式的移动网站，是 Web 开发者所要面对的首要挑战。

本书提供了应对这一挑战的解决方案。通过阅读本书，你将了解如何有效地利用最新的 HTML5 的那些针对移动网站的功能，横跨多个移动平台。

全书共分 10 章，从移动 Web、设备端配置和优化，变互、响应式设计、设备访问，调试、性能测试、富媒体等角度出发，包含了 60 多个实用的示倒，详细阐释如何构建快速、响应式的 HTML5 移动网站，适用于 iOS、Android、Windows Phone 和 BlackBerry 等众多主流移动应用平台。

本书作者是 HTML5 Boilerplate 项目的成员，Mobile Boilerplate(http://h5bp．com/mobile)的主要开发人员。本书为那些致力于实现高性能、响应式、跨平台的 HTML5 移动网站的程序员量身打造，也适合对移动 Web 开发感兴趣的读者学习参考。

作者简介

石川有超过 5 年的 Web 开发经验，他是 HTML5 Boilerplate 项目的成员，Mobile Boilerplate（http://h5bp.com/mobile）的主要开发人员，JavaScript Patterns 项目（http://shichuan.github.com/javascript-patterns）的所有人。他现在是居住在美国和中国的独立开发者，你可以在他的个人网站http://www.blog.highub.com上找到更多的信息。他爱好阅读、旅游、美食以及电子技术、Indie 音乐。

我要感谢我的父母和所有家人，他们给予了我无条件的支持。我也要感谢 Jiang Xue，她教会了我许多关于生活的事情，虽然她自己并不知道。

我还要感谢 Boilerplate 团队中的好友——Paul Irish、Divya Manian、Mathias Bynens 和 Nicolas Gallagher，我前公司的上一任 CTO——Chi Tran。他们曾经并且以后一直都是我灵感和理想的来源。

审阅者简介

Dale Cruse，他是波士顿地区的 Web 开发者，《HTML5 Multimedia Development Cookbook》的作者。从 1995 年开始，他就为很多高知名度的客户建立网站，从美国军方到 Bloomingdale's 都是他的客户。他是新英格兰艺术学院的特约讲师，并且现在仍从事着演讲事业，`http://dalejcruse.com` 有他的联系方式。他同时也是一个关于香槟的博客 `http://www.drinksareonme.net/`的作者。

Sarah Soward，在 Bay Area Video Coalition 和 AcademyX 教授开发、设计和 Adobe Creative 套件。除授课外，她也在 BAVC 上开发了有关 HTML5/CSS3、色彩理论、排版、Fireworks 和 Web 设计流程的课程。她长年从事非盈利性组织的艺术总监工作，是《WordPress and Flash Cookbook》的合著者。不再授课以后，她仍然从事各种设计和开发工作，从商务名片到网站、画犀牛、建筑以及击鼓，直到她的双手可以保持自然的节拍。她喜欢忙碌的感觉。

译者简介

刘旸：经历过各种技术和职位，最终发现前端开发才是其归宿，现为 ThoughtWorks 前端程序员，专注于实践优秀的前端代码架构。同时也是 ThoughtWorks UX Group 一员，关注用户体验以及响应式设计、视差滚动等新兴设计元素。热衷于分享、社区活动、结交朋友。博客：zation.me。新浪微博：@Zation。

刘先宁：2008 年毕业于西安交通大学软件工程学院，ThoughtWorks（成都）咨询师，ThoughtWorks Mobile Lab 成员，长期从事软件研发一线工作，热爱互联网，喜欢各种新技术，喜欢分享、讨论。喜欢户外运动。新浪微博：@xianlinbox。

前言

如何在多平台上创建高性能、响应式移动网站？对于程序员来说，由于手机设备快速增加、屏幕尺寸各不相同、性能仍有局限性，这个问题就显得非常重要。本书提供了答案，你会知道如何有效地利用最新的 HTML5 针对移动网站的功能，横跨多个移动平台。

本书会让你了解如何策划、创建、调试和优化移动网站。在引入新兴的移动网站功能时，使用最新的 HTML5 是最好的选择。

本书展望了许多有潜力的移动网站技术和知识：移动用户的交互方式以及创建一个可靠的移动版 HTML 模板；创建高性能、响应式的网站，并且合理地引入独特的移动地理位置功能和富媒体；完美地进行调试，优化性能以及调整服务端。

概要

第 1 章：HTML5 与移动网站，介绍 HTML5 和移动网站基本概念，包括一些模拟器和仿真器。

第 2 章：移动端的配置和优化，讨论针对多种移动设备的配置和优化，例如禁用文字缩放和优化可视宽度。

第 3 章：移动设备的交互方式，讨论了手势事件等移动交互。

第 4 章：构建快速响应式移动互联网站点，介绍了各种构建快速响应式网站的方法。

第 5 章：移动设备访问，讨论了基于地理位置的移动网站和其他 HTML5 针对设备的功能。

第 6 章：移动富媒体，介绍了可用于移动浏览器的 HTML5 富媒体元素。

第 7 章：移动设备调试，讲解了如何在移动设备屏幕尺寸的限制下有效地调试移动网站和应用。

第 8 章：服务器端性能调优，关注移动网站的服务器端性能调优。

第 9 章：移动性能测试，讲解了各种可用于移动性能调优的工具和技术。

第 10 章：拥抱移动互联网特性，讲解了 ECMAScript5 等一些针对移动设备的功能及性能优化。

基础准备

本书大部分内容只需要一个文本编辑器，以及一个移动设备，例如：iPhone、Android、Blackberry 或者其他适合测试的设备。虽然在真实设备上测试是最完美的，但是如果没有也不用担心，本书会详细介绍如何使用仿真器和模拟器。

适合的读者

创建 HTML5 移动网站并且致力于实现高性能、响应式、跨平台的程序员。

约定格式

本书使用多种不同的文字来区分不同的信息，下面是一些示例及解释：

文本中的代码："`geolocation` 是 `navigator` 对象的新属性。"

整段代码：

```
var latitude = position.coords.latitude;
var longitude = position.coords.longitude;
var accuracy = position.coords.accuracy;
```

对于重要的代码段，我们会将相关的部分加粗：

```
var latitude = position.coords.latitude;
var longitude = position.coords.longitude;
var accuracy = position.coords.accuracy;
```

新的术语和**重要文字**会加粗，菜单或对话框之类屏幕上的文字，会这样显示：“单击**捕获选项**对话框中的**开始**按钮来开始捕获。”

重要提示会显示在此框中

提示和技巧会显示在此框中

读者反馈

欢迎读者的任何反馈、看法（哪些喜欢，哪些不喜欢），这非常重要，我们会根据这些写出真正对读者有用的内容。

任何反馈都可以发送邮件至 feedback@packtpub.com，并在标题中包含书名。

如果你希望我们出版某书，请在 www.packtpub.com 的 **SUGGEST A TITLE** 提交，或是发送邮件至 suggest@packtpub.com。

如果你对某个方面有专业的意见，或是希望参与、写作一本书，请参考作者向导：www.packtpub.com/authors

客户支持

在拥有 Packt 图书后，我们还提供多种服务。

下载示例代码

使用你的账号在 http://www.PacktPub.com 下载所有已购买书籍的示例代码，如果你是以其他方式购买的，请访问 http://www.PacktPub.com/support 并注册，代码文件会由邮件发送。

勘误表

虽然我们努力保证内容的准确，但错误是难免的。如果你找到任何错误（文本中的或代码中的）并报告，我们会非常感谢，这样可以避免误导其他读者，也可以改进该书的后

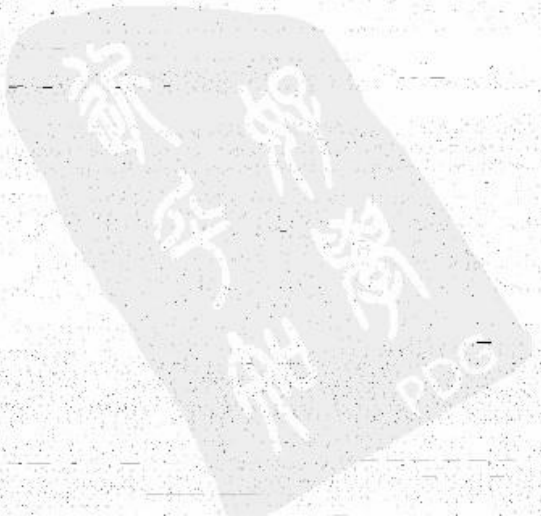
续版本。报告勘误，可以访问 http://www.packtpub.com/support，选择图书，单击 **errata submission form** 连接，输入详细的勘误信息。一旦勘误通过验证，我们将会进行修改并且将勘误显示在网站上或是勘误信息的列表中，任何的勘误都可以在 http://www.packtpub.com/support 通过标题搜索。

版权声明

盗版在互联网中是一个不断发生的问题，Packt 对版权有非常严谨的保护。如果发现任何形式的盗版，请提供地址或网站名，我们将采取维权行动。

问题

如果有关于本书的任何问题，请联系 questions@packtpub.com，我们将会努力为你解决。



图书在版编目（CIP）数据

HTML5移动Web开发实战 / 石川著 ; 刘旸, 刘先宁译
-- 北京 : 人民邮电出版社, 2013.6
ISBN 978-7-115-31328-7

Ⅰ. ①H… Ⅱ. ①石… ②刘… ③刘… 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(2013)第052494号

版权声明

◆ 著　　石　川
译　　刘　旸　刘先宁
责任编辑　陈冀康
责任印制　杨林杰

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京天宇星印刷厂印刷

◆ 开本：800×1000　1/16
印张：14.25
字数：270 千字　　2013 年 6 月第 1 版
印数：1 – 3 000 册　　2013 年 6 月北京第 1 次印刷

著作权合同登记号　图字：01-2012-6490 号

定价：45.00 元

读者服务热线：(010)67132692　印装质量热线：(010)67129223
反盗版热线：(010)67171154

目录

第 1 章
HTML5 与移动网站

本章内容包括：

- 准备好你的移动设备
- 仿真器与模拟器
- 搭建移动开发环境
- 在移动网站中使用 HTML5
- 跨浏览器兼容 HTML5
- 适用于移动设备的设计
- 确定你的核心移动设备
- 定义一个内容策略

1.1 简介

HTML5 以及移动网站都是很有前景的技术，他们都有着相对较短的历史。本章介绍的大部分内容都是比较基础的，可以帮助你迅速、轻松地开始移动端开发。

移动网站以及 HTML5 本身仍然在不断演进，对此你肯定有很多的疑惑。我们会解开这些疑惑并告诉你应该专注在哪些真正重要的事情上。

移动网站的增长非常之快。移动端 Safari 浏览器是当前最常用的 iPhone 应用，它使开发人员可以创建高性能的网页应用并提高用户的浏览体验。移动网站的好处在于，你不需要申请开发者账号就可以运营和维护，你不需要通过任何应用市场的审核就可以发布，你

不需要通过繁琐的审核就可以在任何时候更新。但它也有许多问题，比如，浏览器的不一致；相对于原生应用缺少某些功能和安全性。虽然我们不能解决所有的问题，但是肯定能解决一部分，同时我们还会看到在移动网站的开发过程中的最佳实践。

现在市场上有成千上万的智能手机，但你不需拥有所有的手机来测试你的应用，实际上你只需要不到10个。如果那仍然超出了你的预算，那么两台设备也足够了，剩下的使用模拟器、仿真器去做测试。本书专注于6种最流行的移动设备，特别是iPhone、Android、Windows Phone：

- iOS
- Android
- Windows Mobile
- Blackberry v6.0及以上
- Symbian 60
- Palm webOS

本书还会包括两种跨设备的浏览器：

- Opera Mobile
- Firefox Mobile

在本书讨论的一些问题和技术中，我们也会涉及其他不在此列的浏览器。

1.2 确定网站的适用移动设备

适用浏览器：所有

你不可能给每个移动设备都单独做一个移动网站，没有人有这样的时间和精力。

跨浏览器的移动网站开发是非常困难的，其难点在于如何确定网站的适用范围，John Resig（jQuery Mobile的创始人）在一个描述jQuery Mobile创建经历的PPT（`http://www.slideshare.net/jeresig/testing-mobile-javascript`）中提了三个问题：

- 哪些平台和浏览器是流行的？
- 哪些浏览器可以支持现代脚本？
- 哪些设备和模拟器需要我进行测试？

当你创建一个移动网站，你必须思考类似的问题，但并不一定完全相同，因为不同的网站的目标用户不同。所以你的问题应该是：

- 用户最常用的平台和浏览器是什么？
- 有多少用户使用支持现代脚本的移动设备？
- 应该在哪些设备和模拟器上测试？

1.2.1 用户最常用的平台和浏览器是什么

首先我们来回答第一个问题。在创建移动网站之前，你必须首先确定谁是你的目标用户，他们会使用什么移动设备来访问你的网站。有许多分析工具可以帮助你回答这些问题，例如 Google Analytics。你可以在这里免费注册 Google Analytics 的账号：`http://www.google.com/analytics/`。

使用 Google Analytics 的方法非常简单（大部分开发者对它并不陌生），你需要做的只是从 Google Analytics 网站引入一小段 JavaScript 嵌入到你的网站中。

大部分现代智能手机都支持 JavaScript，所以在移动网站中使用它与在桌面网站中没有不同。

1.2.2 有多少用户使用支持现代脚本的移动设备

我们现在来回答第三个问题。也许你想知道有多少人使用移动浏览器访问你的网站，同时你也想知道有多少人使用完全不支持 JavaScript 的老旧移动浏览器。因为如果使用低端智能手机的人多于使用高端智能手机的人，那么就不值得以 HTML5 作为核心技术了（虽然这种可能性很低）。

所以如果你不仅希望知道有多少人使用智能手机，同时也希望知道有多少人使用老旧的移动电话，Google Analytics 移动版可以派上用场。你可以在这里下载脚本：

```
http://code.google.com/mobile/analytics/download.html#Download_the_Google_Analytics_server_side_package
```

Google Analytics 移动版服务器端程序包当前支持 JSP、ASPX、Perl 以及 PHP，让我们来看一个基于 PHP 的例子。你所需要做的只是修改 **ACCOUNT ID GOES HERE** 为你的 GA 账户 ID，也记得修改“**UA-xx**”为“**MO-xx**”。

遗憾的是，当你使用服务端版本，就不能同时使用 ga.js 这一 JavaScript 跟踪代码库。放弃 JavaScript 版本让人很遗憾，因为 JavaScript 版本提供了很多服务端版本所缺乏的动态跟踪机制：

```
<?php
  //  Copyright 2009 Google Inc. All Rights Reserved.
  $GA_ACCOUNT = "ACCOUNT ID GOES HERE";
  $GA_PIXEL = "ga.php";

  function googleAnalyticsGetImageUrl() {
    global $GA_ACCOUNT, $GA_PIXEL;
    $url = "";
    $url .= $GA_PIXEL . "?";
    $url .= "utmac=" . $GA_ACCOUNT;
    $url .= "&utmn=" . rand(0, 0x7fffffff);

    $referer = $_SERVER["HTTP_REFERER"];
    $query = $_SERVER["QUERY_STRING"];
    $path = $_SERVER["REQUEST_URI"];

    if (empty($referer)) {
      $referer = "-";
    }
    $url .= "&utmr=" . urlencode($referer);

    if (!empty($path)) {
      $url .= "&utmp=" . urlencode($path);
    }

    $url .= "&guid=ON";

    return $url;
  }
?>
```

1.2.3 Google Analytics 的替代品

Google Analytics 不是市场上唯一的移动数据分析服务商，还有其他的服务商提供更有针对性的服务，比如，**PercentMobile** 提供一个帮助分析你的移动用户群和网站价值的移动数据分析服务。你在这里可以找到更多关于该服务的信息：

```
http://percentmobile.com/
```

1.2.4 Google Analytics 的精准度

移动设备报告的位置并不总是精确的，Google Analytics 的 Map Overlay 报告以用户的 IP 地址来确定用户的位置。由于移动设备的 IP 来源于无线网关，它不需要也不知道移动用

户的准确位置，所以些许不准确也是可以接受的。

1.2.5 对服务端读取速度的担忧

基于服务端的处理方式，可能会对服务器造成额外的负担和影响，所以 Google 建议先在某些页面小范围测试一下，确保一切正常之后再在全站部署。

1.3 配置移动开发工具

适用浏览器：所有

在前一节还剩下一个问题没有回答：我应该在哪些设备和模拟器上测试？我们会在这一节回答。

如果你决定了需要支持的核心移动设备，那么现在就可以来看看如何配置了。如果在过多移动设备上测试，那么移动开发成本会很高，虽然我们可以使用移动设备的模拟器和仿真器，但都比不上在真实设备上做测试。现在我们来看看如何最大化测试覆盖率并最小化成本。

1.3.1 准备

我们会首先做一些假设，虽然具体情况可能不同，但思路是一样的。我们假设你的桌面操作系统是 Windows，而大部分用户是通过 iOS、Android、Blackberry 来访问你的网站。

1.3.2 实践

你的目标是最大化覆盖率和最小化成本，虽然所有的设备都有模拟器，但他们并不支持所有的平台。

表 1-1

名　称	兼 容 性
iOS 仿真器	Mac
Android 模拟器	Windows, Mac, Linux
Blackberry 仿真器	Windows

如表 1-1 所示，由于 iOS 模拟器只运行在 Mac 上，如果你使用 Windows 操作系统，那

么最好也是唯一的选择就是购买 iPhone 来测试。对于 Android 和 Blackberry，因为他们都有 Windows 上的模拟器，你可以直接下载模拟器来节约成本。

1.3.3 回顾

1. 列出你的用户的主要移动设备。
2. 了解你用于开发的操作系统。
3. 确定每个设备模拟器对于开发环境的兼容性。

1.3.4 延伸

如果你有资金购买多台运行于不同操作系统的移动设备，那么你可以考虑得更多一些，例如，屏幕尺寸和 DPI。也许你不需要该购买两台高端设备，例如 iPhone4 和 Android Thunderbolt[①]。你可以购买一款低端 Android 设备来测试你的手机在低端设备上的样式。总的来说就是减少你的操作系统、移动设备和模拟器，让他们每个都覆盖尽可能多的使用场景。

各设备模拟器、仿真器下载表

下面的表 1-2 显示了用于网页设计和测试的主流移动设备模拟器。

表 1-2

名　称	类型	兼 容 性	链 接 地 址
iOS	仿真器	Mac	https://developer.apple.com/devcenter/ios/index.action#downloads
Android	模拟器	Mac, Windows, Linux	http://developer.android.com/sdk/index.html
HP webOS	虚拟机	Mac, Windows, Linux	http://developer.palm.com/index.php?option=com_content&view=article&id=1788&Itemid=55
Nokia Symbian	模拟器	Windows	http://www.forum.nokia.com/info/sw.nokia.com/id/ec866fab-4b76-49f6-b5a5-af0631419e9c/S60_All_in_One_SDKs.html
Blackberry	模拟器	Windows	http://us.blackberry.com/developers/resources/simulators.jsp
Windows Mobile 7	模拟器	Windows	http://www.microsoft.com/downloads/en/details.aspx?FamilyID=04704acf-a63a-4f97-952c-8b51b34b00ce

① HTC 的一款高端手机型号——译者注

浏览器的模拟器、仿真器下载表

除移动设备测试工具外，我们还有针对跨平台浏览器的工具，特别是 Opera 和 Firefox，如表 1-3 所示。

表 1-3

名　称	类型	兼 容 性	链 接 地 址
Opera Mobile	模拟器	Mac, Windows, Linux	http://www.opera.com/developer/tools/
Opera Mini	仿真器	Mac, Windows, Linux	http://www.opera.com/developer/tools/http://www.opera.com/mobile/demo/
Firefox for Mobile	仿真器	Mac, Windows, Linux	http://www.mozilla.com/en-US/mobile/download/

远程测试

除仿真器和模拟器之外，还有一些测试平台提供远程连接到真实设备，其中之一就是 DeviceAnywhere，但问题是它并不免费。

`http://www.deviceanywhere.com/`

1.4 BlackBerry 仿真器

适用浏览器：BlackBerry

大部分的移动设备仿真器，都可以很容易地根据其官方网站的介绍来安装和配置，但 BlackBerry 仿真器的工作方式与其他仿真器不同。从 Blackberry 仿真器连接互联网，除了下载仿真器还需要下载安装 **BlackBerry Email and MDS Services Simulator**。

1.4.1 准备

确认你从以下地址下载了仿真器：`http://us.blackberry.com/developers/resources/simulators.jsp`。

1.4.2 实践

首先进入页面：`https://swdownloads.blackberry.com/Downloads/entry.do?code=A8BAA56554F96369AB93E4F3BB068C22&CPID=OTC-SOFTWAREDOWNLOADS&cp=OTC-SOFTWAREDOWNLOADS`，你会看到类似图 1-1 所示的产品列表。

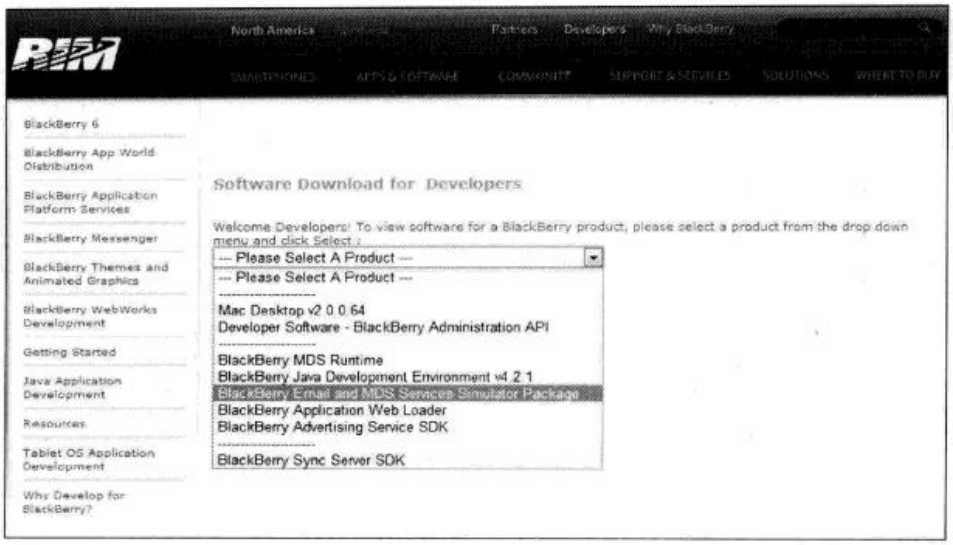


图 1-1

选择 **BlackBerry Email and MDS Services Simulator Package** 后点击 **Next**。

为了可以连接到互联网，下载安装后必须首先启动该服务仿真器，然后再启动 Blackberry 仿真器。

图 1-2 所示就是 Blackberry 仿真器。

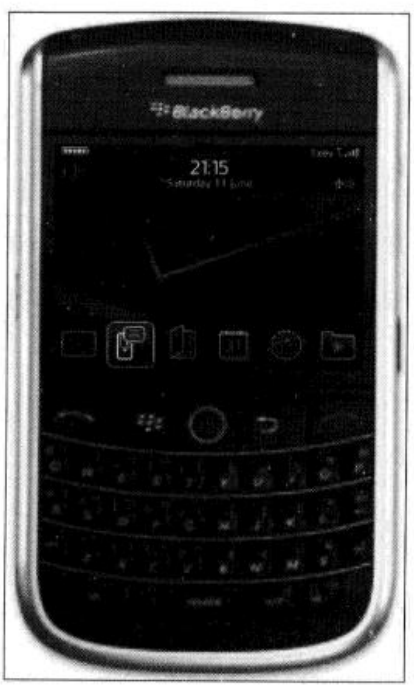


图 1-2

1.5 配置移动开发环境

适用浏览器：所有

在开始移动网站开发之前，我们必须首先配置好开发环境。

1.5.1 准备

1. 配置本地网站服务。对于 Windows、Mac 和 Linux，最容易的方法是使用免费的 XAMPP 软件：

http://www.apachefriends.org/en/index.html

2. 确认你有无线网络。

3. 同时你需要一个移动设备，或者是移动设备的仿真器、模拟器。

4. 确保你的移动设备和电脑在同一无线网络中。

1.5.2 实践

1. 在你的本地服务器根目录中创建一个 HTML 文件并命名为 ch01e1.html，在其中输入如下代码：

```
<html>
  <head>
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
    <header>
    Main Navigation here
    </header>
   body content here
    <footer>
    Footer links here
    </footer>
  </body>
</html>
```

2. 获取你的 IP 地址。如果你使用的是 Windows，你可以在命令行中输入如下命令：

```
ipconfig
```

下载本书的示例

你可以在 http://www.PacktPub.com 下载所有用你的账号购买的 Packt 书籍的示例代码。如果是在其他地方购买的，你可以访问 http://www.PacktPub.com/support 并注册，示例代码会以电子邮件发送给你。

3. 当你获取到你的 IP 地址（如 192.168.1.16），把它输入到移动浏览器的 URL 地址栏，然后你就可以看见页面加载成功并且显示图 1-3 所示的文字。

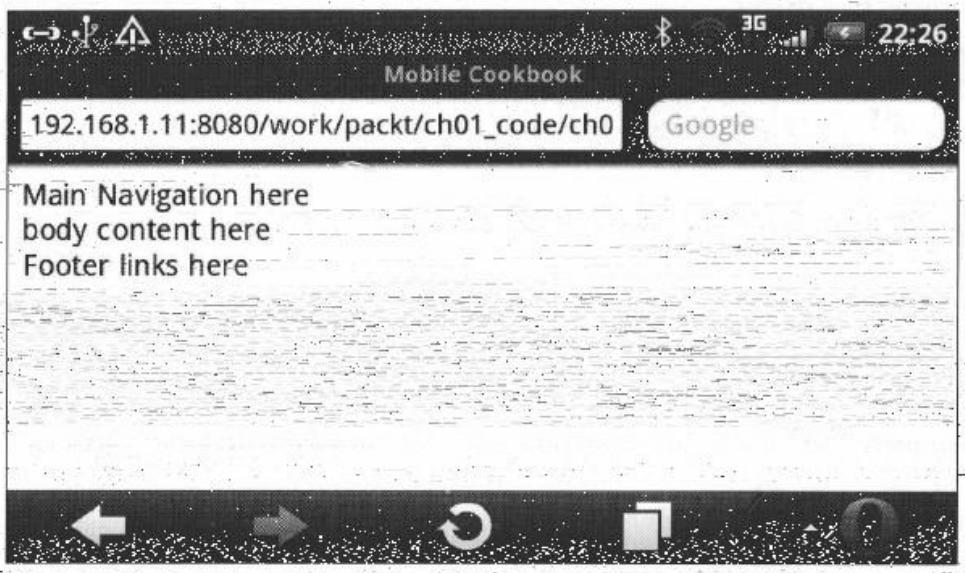


图 1-3

1.5.3 回顾

在同一个局域网中，你的移动设备可以通过 IP 地址访问以桌面电脑作为主机的服务器。

1.5.4 延伸

如果你没有移动设备，你可以使用某个仿真器来测试，但建议至少有一两个真实设备来测试。仿真器可以测试到大部分的情况，但不是所有的。

使用桌面版 Safari 测试

如果你的主要用户使用 iPhone 和移动版 Safari，在桌面电脑上测试可以节约很多时间：打开 Safari，在偏好设置中单击高级栏打开，在菜单栏中显示“开发”菜单，如图 1-4 所示。

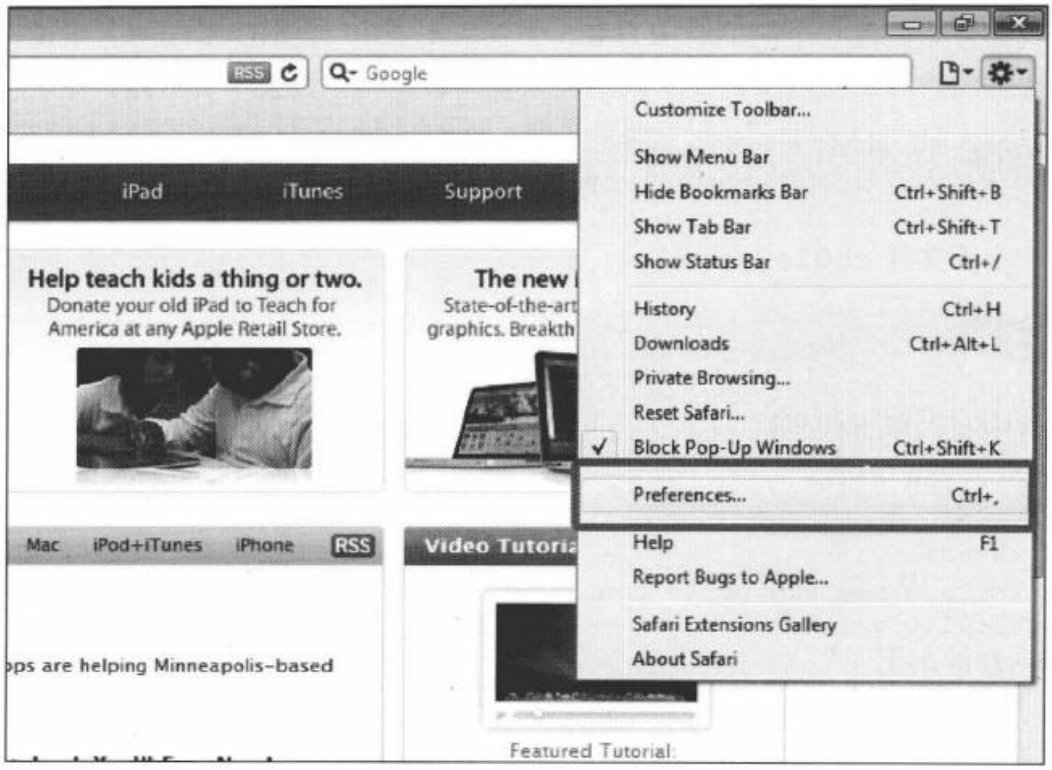


图 1-4

在本页面中选择开发 | 用户代理 | Mobile Safari 3.1.3 – iPhone：

社区搜集的仿真器、模拟器

如果你没有智能手机，也可以使用很多仿真器、模拟器，你可以在 Mobile Boilerplate 项目的 wiki 页面找到：

```
https://github.com/h5bp/mobile-boilerplate/wiki/Mobile-Emulators-&-Simulators
```

Fritman 搜集的仿真器、模拟器

Maximiliano Firtman，一个移动网站开发人员和作家，他维护了一个仿真器列表：

```
http://www.mobilexweb.com/emulators
```

1.6 在移动网站中使用 HTML5

适用浏览器：所有

现在我们开始创建一个简单的 HTML5 页面。对于有 HTML 基础的人来说，HTML5 是很容易理解的。对于有网页开发基础的人来说，移动网站开发不会太难。

1.6.1 准备

创建一个新文件 ch01e2.html

1.6.2 实践

在文件中写入以下代码：

```
<!doctype html>
<html>
  <head>
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
    hello to the HTML5 world!
  </body>
</html>
```

在浏览器中打开文件，你会看到写入的文字。

1.6.3 回顾

HTML5 和其他 HTML 页面的唯一区别就在于我们使用的**文件类型定义**（DTD，Document Type Definition）：<!doctype html>。

Safari 会根据<meta name="viewport" content="width=device- width, initial-scale=1.0">，将页面宽度设为屏幕宽度，并且根据 initial-scale=1 禁用浏览器的缩放。

1.6.4 延伸

下面是 HTML5 的一些历史：HTML5 最初有两个版本的草稿，分别由万维网联盟（W3C，

World Wide Web Consortium）和网页超文本技术工作小组（WHATWG，Web Hypertext Application Technology Working Group）创建。万维网联盟本质上是以民主投票作为决策机制的一个小组，但实际效率非常迟缓。网页超文本技术工作小组是由 lan Hickson（同时也是 Google 的员工）和一组未公开的人员编辑，由于大部分的设计都是 lan 完成的，所以网页超文本技术工作小组的草稿进度快得多。

HTML5 与版本号

为什么 HTML5 会没有版本号？这里有一些原因：

1．浏览器并不会针对 HTML 的某个版本做支持，而是针对某个功能做支持。就是说如果浏览器支持你使用的某个功能，即使你把文档申明为 HTML4，浏览器仍然会按照 HTML5 的标准来显示页面。

2．名字可以很简洁。

移动文档类型

使用 HTML5 文档类型`<!doctype html>`是否是可靠的？答案是文档类型只是用作确认，而非浏览器实际显示。在怪异模式[①]中是否是可靠的？`<!doctype html>`是浏览器按照标准工作所需要的最少的信息，所以使用`<!doctype html>`是非常可靠的。

我们使用`<!doctype html>`而不是`<!DOCTYPE html>`，这是因为 HTML5 不是大小写敏感的，但是出于一致性的考虑，本书中都将使用小写。

学习 HTML5 的免费资源

关于 HTML5 有许多优秀的免费书籍和文章，如果你对 HTML5 不是很熟悉，可以在下面的网站学习：

- HTML5 Doctor: http://html5doctor.com/
- Dive Into HTML5: http://diveintohtml5.org/
- HTML5 Rocks: http://www.html5rocks.com/

如果希望详细了解 HTML5，你可以阅读官方 HTML5 文档。

W3C 版本的文档：

---

① 怪异模式（quirks mode），是指在计算机领域中，一些网页浏览器为了维持对较旧的网页设计的向后兼容性而使用的一种技术。——译者注

http://dev.w3.org/html5/spec/Overview.html

WHATWG 版本的在线标准：

http://www.whatwg.org/specs/web-apps/current-work/multipage/

1.7 跨浏览器 HTML5

适用浏览器：所有

旧浏览器无法识别 HTML5 元素，也无法对这些元素设置样式，但有许多工具可以解决这个问题，例如 Modernizr。

1.7.1 准备

1. Windows Mobile 的自带浏览器无法识别 HTML5 元素，如果没有 Windows Mobile，你可以使用 IE7 来测试，因为他们都是基于相同的浏览器引擎。

2. 在此下载 Modernizr：http://www.modernizr.com/，它由 Faruk Ateş 、Paul Irish 和 Alex Sexton 开发。

1.7.2 实践

1. 新建一个 HTML 文件命名为 ch01e3.html，然后输入以下代码：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
       <style>
    header, footer {display:block;}
  </style>
  </head>
  <body>
    <header>
      Main Navigation here
    </header>
    body content here
    <footer>
      Footer links here
    </footer>
  </body>
</html>
```

2. 现在新建另一个文件命名为 ch01e4.html，引入 Modernizr，如图 1-5 所示。

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <script src="modernizr-1.7.min.js"></script>
  <style>
  header, footer {display:block;}
  </style>
  </head>
  <body>
    <header>
      Main Navigation here
    </header>
    body content here
    <footer>
      Footer links here
    </footer>
  </body>
</html>
```

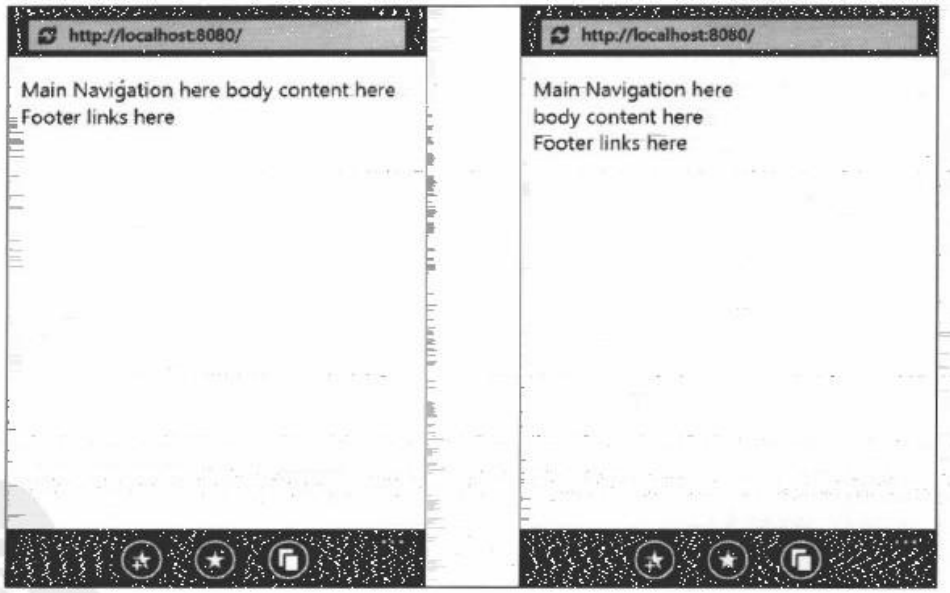


图 1-5

1.7.3 回顾

注意如果要使用 Modernizr，你需要在<head>标签中引入它。下一节还有一些其他的

工具与 Modernizr 类似。

1.7.4 延伸

Modernizr 不是唯一可以帮助我们跨浏览器的库，还有其他两个值得注意：

- **html5shim**，由 Remy Sharp、Jonathan Neal 和社区开发，对打印也同样有效。

  http://code.google.com/p/html5shim/

- **InnerShiv**，由 Joe Bartlett 开发，支持元素的 innerHTML。

  http://jdbartlett.github.com/innershiv/

HTML5 CSS 重置

下面的代码可以清除 HTML5 元素的默认样式：

```
article, aside, canvas, details, figcaption, figure,footer, header,
hgroup, menu, nav, section, summary,time, mark, audio, video {
  margin: 0;
  padding: 0;
  border: 0;
  font-size: 100%;
  font: inherit;
  vertical-align: baseline;
}
```

使 HTML5 元素在旧版本 IE 中变为块级元素：

下面的代码可以使 HTML5 元素变为块级元素，但注意不是所有的 HTML5 元素都需要显示为块级元素。

下面是 HTML5 中的块级元素：

```
article, aside, details, figcaption, figure,footer, header, hgroup,
menu, nav, section {
  display: block;
}
```

Modernizr

Modernizr 不仅使 HTML5 元素可以被设置样式，它还可以检测 HTML5 各个功能在不同浏览器中的兼容性。你可以在 2.0 版本中自定义下载内容：http://www.modernizr.com/download/。

1.8 移动设计

适用浏览器：所有

桌面网站的设计趋势是固定布局（flxed layout）或流体布局（fluid layout），而在移动网站中我们应该始终使用流体布局，它可以使你的网站适应不同的设备尺寸。

1.8.1 准备

新建两个空白 HTML 文件，命名为 ch01r06_a.html 和 ch01r06_b.html。

1.8.2 实践

1. 在 ch01r06_a.html 中输入以下代码并保存：

```
<!doctype html>
<html>
  <head>
  <meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0">
  <script src="modernizr-1.7.min.js"></script>
  <style>
  body, #main ul, #main li, h1 {
    margin:0; padding:0;
  }
  body {
    background:#FFFFA6;
  }
  #container {
    font-family:Arial;
    width:300px;
    margin:0 auto;
  }
  header, footer {
    display:block;
  }
  #main li{
    list-style:none;
    height:40px;
    background:#29D9C2;
    margin-bottom:0.5em;
    line-height:40px;
```

```
    -moz-border-radius: 15px;
    -webkit-border-radius: 15px;
    border-radius: 15px;
  }
  #main li a {
    color:white;
    text-decoration:none;
    margin-left:1em;
  }
  </style>
  </head>
  <body>
    <div id="container">
      <header>
        <h1>Title here</h1>
      </header>
      <nav id="main">
        <ul>
          <li><a href="#">Home</a></li>
          <li><a href="#">Contact Us</a></li>
          <li><a href="#">Location</a></li>
          <li><a href="#">Product</a></li>
          <li><a href="#">About</a></li>
        </ul>
      </nav>
      <footer>
        Footer links here
      </footer>
    </div>
  </body>
</html>
```

2. 在ch01r06_b.html中输入以下代码并保存：

```
<!doctype html>
<html>
  <head>
  <meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0">
  <script src="modernizr-1.7.min.js"></script>
  <style>
  body, #main ul, #main li, h1 {
    margin:0;
    padding:0;
  }
```

```
body {
  background:#FFFFA6;
}
#container {
  font-family:Arial;
  margin:0 10px;
}
header, footer {
  display:block;
}
#main li{
  list-style:none;
  height:40px;
  background:#29D9C2;
  margin-bottom:0.5em;
  line-height:40px;
  -moz-border-radius: 15px;
  -webkit-border-radius: 15px;
  border-radius: 15px;
}
#main li a {
  color:white;
  text-decoration:none;
  margin-left:1em;
}
</style>
</head>
<body>
  <div id="container">
    <header>
      <h1>Title here</h1>
    </header>
    <nav id="main">
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">Contact Us</a></li>
        <li><a href="#">Location</a></li>
        <li><a href="#">Product</a></li>
        <li><a href="#">About</a></li>
      </ul>
    </nav>
    <footer>
      Footer links here
    </footer>
  </div>
</body>
</html>
```

1.8.3 回顾

两个页面在纵向显示中看起来几乎一样，如图 1-6 所示。

现在旋转你的屏幕，看看发生了什么。

在横向显示中，第一个例子（见图 1-7）两边出现了空白，而第二个例子（见图 1-8）充满了整个屏幕。

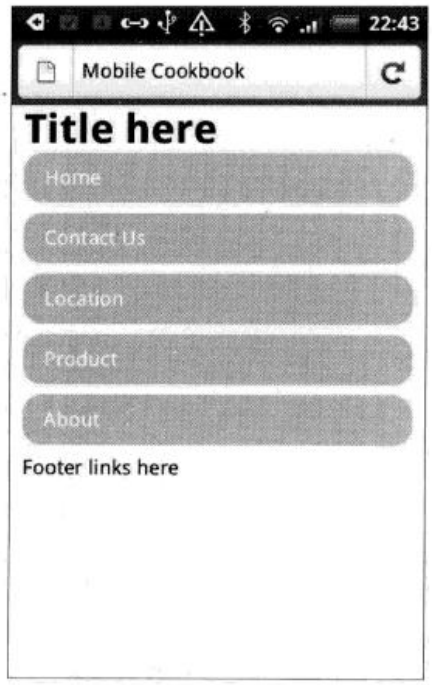


图 1-6

图 1-7

第二个例子有不同的结果。

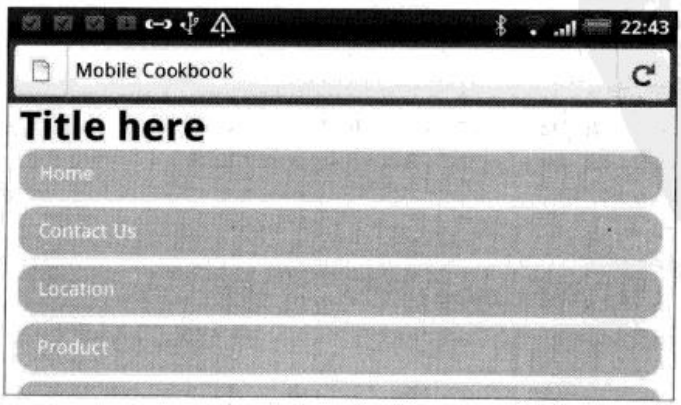


图 1-8

这个页面在固定布局中看起来很奇怪，但在流体布局中正常显示。所以当你在针对移动设备做设计时，始终谨记保持这种灵活性，因为：

- 移动设备有横向和纵向显示。
- 移动设备的屏幕空间很有限，所以需要利用好每个像素。

1.8.4 延伸

CSS 媒介查询（media query）是响应式设计中很重要的部分，它帮助你实现更加灵活的移动设计。

```
<!doctype html>
<html>
  <head>
   <meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0">
   <script src="modernizr-1.7.min.js"></script>
   <style>
   body {
     margin:0;
     padding:0;
     background:#FFFFA6;
   }

   #main section {
     display:block;
     border:5px solid #29D9C2;
     width:60%;
     height:120px;
     margin:5% auto;
   }

   @media screen and (min-width: 480px) {
     #main {
       width:90%;
       margin:0 auto;
     }

     #main > section:first-child {
       margin-right:5%;
     }

     #main section {
       float:left;
```

```
        width:45%;
      }
    }
    </style>
    </head>
    <body>
      <div id="container">

        <div id="main">
          <section id="top-news"></section>
          <section id="sports"></section>
        </div>

      </div>
    </body>
  </html>
```

在窄屏中，这两个 section 会纵向排列（如图 1-9 所示），而在宽屏中会横向排列，这就是 CSS 媒介查询的功能。在示例中，`@media screen and (min-width: 480px){..}`的样式会在 480px 以上宽度的屏幕中启用：

在横向显示中，两块方框并列显示，如图 1-10 所示。

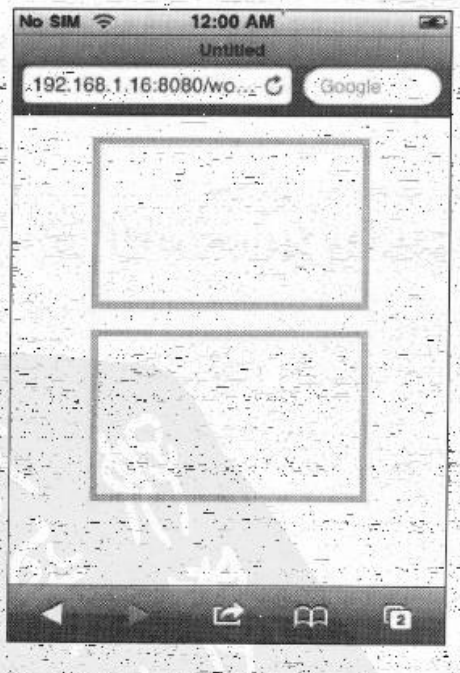


图 1-9

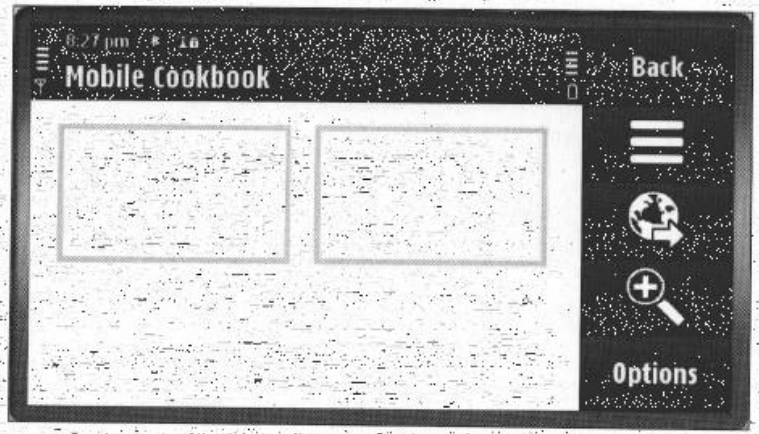


图 1-10

桌面优先的网站

除了创建纯粹的桌面网站或纯粹的移动网站，我们还有其他建站的方式，其中之一就是桌面优先，并在移动设备上优雅降级。

移动优先网站

另一个方式是移动优先，并在桌面设备上优雅降级。

这种方式可以使用以下 CSS：

```
@media only screen and (min-width: 320px) {
  /* Styles */
}
@media only screen and (min-width: 640px) {
  /* Styles */
}
@media only screen and (min-width: 800px) {
  /* Styles */
}
@media only screen and (min-width: 1024px) {
  /* Styles */
}
```

一站方式

另一个方式是只创建一个网站，并兼顾移动和桌面设备，而不是只专注其中一个。

1.9 定义一个内容策略

适用浏览器：所有

通过分析工具搜集到的数据，你可以定义一个内容策略，这对已经有了一个桌面网站的人是非常有用的。

1.9.1 准备

确认你已经把分析工具的 Javascript 代码嵌入到网站中。

1.9.2 实践

1. 进入分析工具，在左边导航栏中点击受众群体 | 移动设备，如图 1-11 所示。

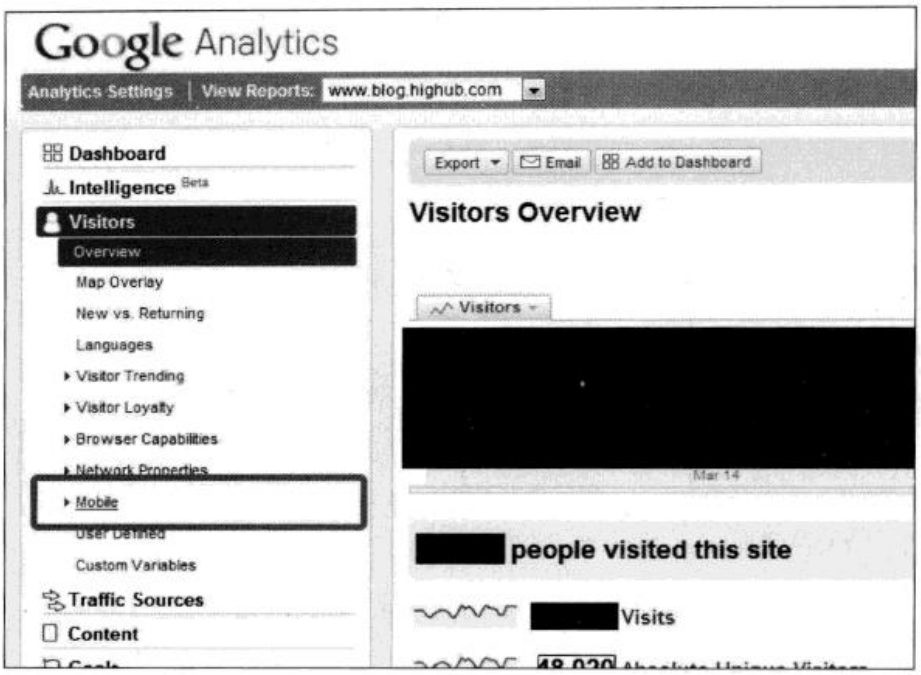


图 1-11

2．现在点击设备，你可以看到人们使用什么设备浏览你的网站，如图 1-12 所示。

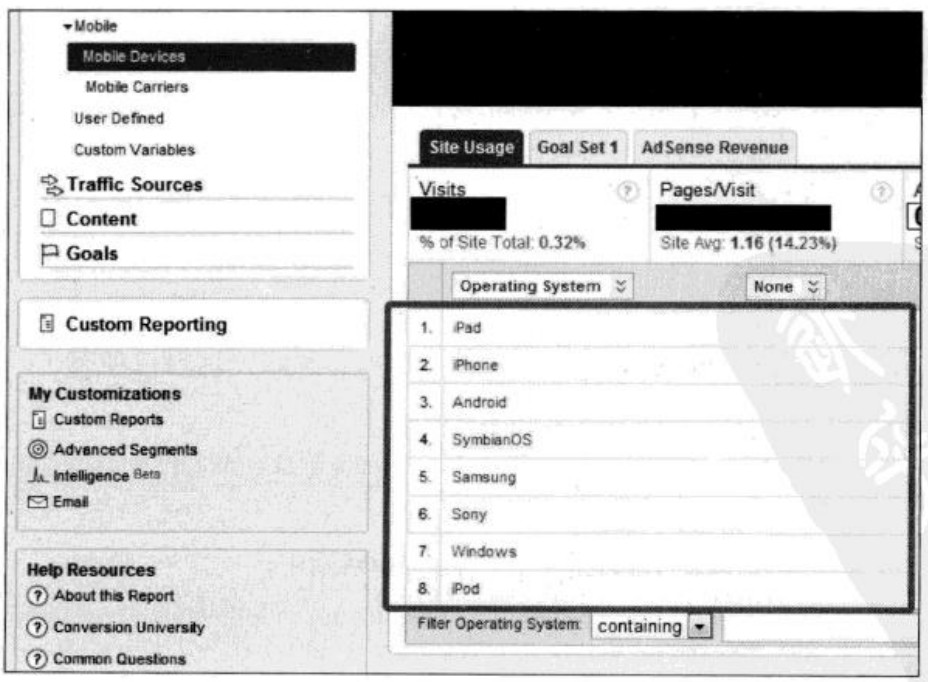


图 1-12

1.9.3 回顾

Google Analytics 可以分析浏览你的网站最常用的移动设备，也可以分析你的网站最受欢迎的部分。

1.9.4 延伸

人们使用移动设备上网的目的与使用桌面设备上网是不同的，例如：对于一个产品销售网站，大部分用户使用移动设备常常查看联系方式、地址和服务，而使用电脑的用户常常搜索产品目录、公司简介和产品描述。Google Analytics 可以分析哪个部分或页面是访问量最大的，除了 Google Analytics，你也可以使用我们早前发现的 PercentMobile，如图 1-13 所示。

图 1-13

浏览器分级

使用分析工具是决定支持哪些浏览器的方法之一，另一种是依据浏览器分级。jQuery Mobile 有一个很好的浏览器支持表：http://jquerymobile.com/gbs/，同时还有一个 jQuery Mobile 的 ppt 讨论了所有移动网站开发策略：http://www.slideshare.net/jeresig/testing- mobile-javascript。

移动设备信息

我与 Jonathan Neal 和其他很多人合作总结了一个移动设备中前端相关的信息表：

https://github.com/h5bp/mobile-boilerplate/wiki/Mobile-Matrices

它包含了当前市面上大部分的智能手机，以及他们的屏幕尺寸、DPI（Dots Per Inch，点每英寸）和操作系统。

第 2 章
移动端的配置和优化

本章内容包括：

- 通过界面图标启动 Web 应用
- 避免文本字体大小重置
- 优化浏览器视口宽度设置
- 修复移动版 Safari 的 re-flow scale 问题
- 在浏览器中调用手机原生应用
- iPhone 下全屏模式启动
- 防止 iOS 在聚焦时自动缩进
- 禁用或限制部分 WebKit 特性

2.1 简介

这个世界有各种各样的移动操作系统，如 iOS、Android、Windows Phone 等；也有很多的设备制造商，如苹果、三星、HTC 等。这样的现状导致了很多让人头疼的跨浏览器问题。但是，我们是程序员，我们乐于接受这样的挑战，能够解决这种棘手的问题本身就是一件极有成就感的事，不是吗？

在本章中，我们首先会告诉读者，在做跨浏览器或针对特定浏览器的配置和优化时，您需要考虑的事项，然后会给读者介绍一些在刚开始开发移动应用时，可能会用到的基本特性，以及一些针对特定浏览器的属性。

2.2 通过界面图标启动 Web 应用

适用设备：iOS、Android、Symbian

现在的智能手机基本上都是触摸屏的，iPhone 的出现改变了我们对移动手机的认知，它把移动设备的所有功能都变成了一个个的应用程序，甚至连打电话和发短信都和应用程序一样，通过点击屏幕上的图标运行。对于基于 HTML 的 Web 应用来说，运行起来就有很大的不同了，用户必须先打开浏览器，然后访问想使用的应用程序站点，这对于用户来说，太过繁琐了。因此，在智能机上，我们是否可以把一个指定的 Web 应用绑定到界面上一个图标呢？这样，用户就能够通过点击界面上的图标来启动对应的 Web 应用程序了。

这貌似听起来很酷，是吧？它的确很酷，但是也有一定的副作用，就是不同浏览器对于触碰图标的反应是不一致的，在本小节中，我们将探索不同浏览器对于点击图标启动的不同反应，从而使点击图标启动 Web 应用的特性能够覆盖到更多的浏览器。

2.2.1 准备

首先，你需要从本章节的源代码目录下载例子需要的图标包，解压并进入图标目录，你会看到如下的图片文件：

- apple-touch-icon.png
- apple-touch-icon-57x57-precomposed.png
- apple-touch-icon-72x72-precomposed.png
- apple-touch-icon-114x114-precomposed.png
- apple-touch-icon-precomposed.png

这些图片是为不同的移动设备准备的，接下来，创建一个名为“ch02r01.html”的 HTML 文件。

2.2.2 实践

在你上一步创建的 HTML 文件中写入如下代码：

```
<!doctype html>
<html>
  <head>
```

```
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="apple-touch-icon-precomposed" sizes="114x114"
href="icons/apple-touch-icon-114x114-precomposed.png">
    <link rel="apple-touch-icon-precomposed" sizes="72x72"
href="icons/apple-touch-icon-72x72-precomposed.png">
    <link rel="apple-touch-icon-precomposed" href="icons/apple-touch-
icon-precomposed.png">
    <link rel="shortcut icon" href="icons/apple-touch-icon.png">
  </head>
  <body>


  </body>
</html>
```

2.2.3 回顾

接下来，我们一步一步地分析一下源代码. 从 iOS 4.2.1 起，iOS 提供了一个新的功能，我们可以通过设置 sizes 属性来为不同的设备提供不同的图标。

```
<link rel="apple-touch-icon-precomposed" sizes="114x114" href="apple-
touch-icon-114x114-precomposed.png">
```

对于 iPhone4 使用的高分辨率 Retina 屏幕，使用“114×114”大小的图标。

```
<link rel="apple-touch-icon-precomposed" sizes="72x72" href="apple-
touch-icon-72x72-precomposed.png">
```

对于 iPad，使用“72×72”大小的图标。对于未使用 Retina 屏幕的 iPhone 和 Android2.1 以上版本的设备，使用“57×57”大小的低分辨率图标。

```
<link rel="apple-touch-icon-precomposed" href="apple-touch-icon-
precomposed.png">
```

对于诺基亚 Symbian 60 的设备来说，一个快捷键图标只是用来告诉移动设备这个图标的位置。

```
<link rel="shortcut icon" href="img/1/apple-touch-icon.png">
```

图 2-1 就是定制的图标在 Android 设备上的显示效果：

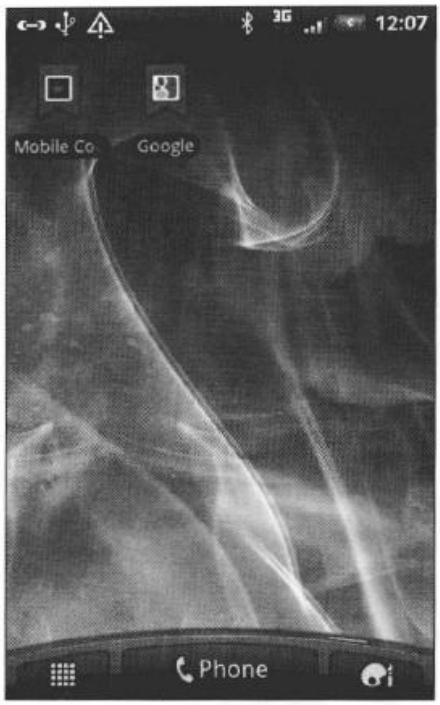


图 2-1

2.2.4 延伸

看完了上面的范例，大家的大脑里是否还有所疑问？比如，rel 属性是否支持多值？如果支持的话，我们就可以把范例中的最后两行合并为：

```
<link rel="shortcut icon apple-touch-icon-precomposed"
href="apple-touch-icon-precomposed.png">
```

其实作者曾尝试过这种写法，但是移动端浏览器根本无法识别这样的写法。你可能还看过有人像下面这么写：

```
<link rel="apple-touch-icon-precomposed" media="screen and
(min-resolution: 150dpi)" href="apple-touch-icon-114x114-
precomposed.png">
```

作者和 Paulk Irish 以及 Divya Manian 一起发起过一个开源项目 Mobile Boilerplate (http://www.h5bp.com/mobile),该项目的目的是为移动设备的前端开发提供一个稳健的基础模板。其实，在 Mobile Boilerplate 项目中，已经囊括了现阶段所有的场景，以及一些未来可能出现的场景，有兴趣的朋友请移步：

```
https://github.com/h5bp/mobile-boilerplate/blob/master/inde
x.html#L21
```

关于触摸式图标的一切

本节中关于触模式图标的大部分观点都来自于 Mathias Bynens。你可以在下面的网址找到他发表的文章“关于触模式图标的一切”：

```
http://mathiasbynens.be/notes/touch-icons
```

苹果触摸式图标的官方文档

下面是苹果公司关于触模式图标的官方文档，你可以从中找到更多关于触模式图标的信息，包含了各种设备和浏览器的信息。

- 苹果公司设备上的触摸式图标

  ```
  http://developer.apple.com/library/safari/#documentation/AppleApplications/Reference/SafariWebContent/ConfiguringWebApplications/ConfiguringWebApplications.html
  ```

- WHATWG 的官网

  ```
  http://www.whatwg.org/specs/web-apps/current-work/multipage/links.html#rel-icon
  ```

定制苹果图标和创建图片

你可以从下面的文章中学习如何创建一个触摸式图标，苹果—定制图标及创建图像指引：

```
http://developer.apple.com/library/ios/#documentation/userexperience/conceptual/mobilehig/IconsImages/IconsImages.html#//apple_ref/doc/uid/TP40006556-CH14-SW11
```

2.2.5 参考

在有了启动图标之后，可以查看 2.7 节，在那一节中，我们还将介绍当从界面点击图标启动程序时，如何以全屏模式启动。

2.3 避免文本字体大小重置

适用设备：iOS、Windows Mobile

在一些移动设备上，比方说 iPhone, Windows Mobile，当用户把手机切换到横屏模式的时候，浏览器会自动地重置文本字体大小。这可能会对 Web 应用开发人员造成困扰，因为开发人员希望能够完全掌控用户界面的设计和浏览器对页面的渲染结果。

2.3.1 准备

创建一个名为“ch02r02.html”的 HTML 文件，并写入如下代码：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
  <style>
    figure, figcaption, header {
      display:block;
      margin:0 auto;
      text-align:center;
    }
  </style>
  </head>
  <body>
    <header>
      HTML5 Logo
    </header>
    <figure>
      <img src="HTML5_Badge_128.png" alt="HTML5 Badge" />
      <figcaption>
        It stands strong and true, resilient and universal as
the markup you write.
        It shines as bright and as bold as the forward-thinking,
dedicated web developers you are.
        It's the standard's standard, a pennant for progress.
        And it certainly doesn't use tables for layout.
      </figcaption>
    </figure>

  </body>
</html>
```

接下来，在 iPhone 的竖屏模式渲染该文件，你会发现，一切都很正常如图 2-2 所示。

接下来，我们把 iPhone 手机切换到横屏模式，该页面的字体大小会突然的放大，显而易见。该页面的字体大小被重置了，这并不是开发者期望的结果，图 2-3 就是该文件在横

屏模式下渲染出来的效果。

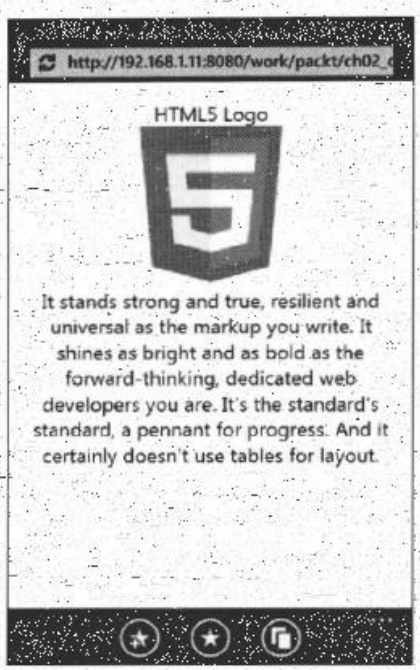


图 2-2

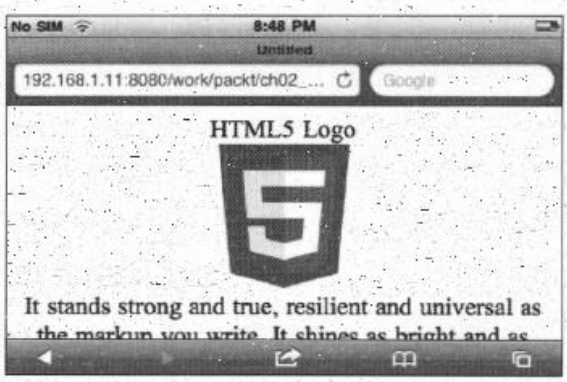


图 2-3

2.3.2 实践

针对这样的问题，我们可以通过如下的方式来解决：

首先，你需要在页面的 CSS 部分加入如下代码：

```
html {
    -webkit-text-size-adjust: none;
}
```

然后重新试一下在横屏模式下渲染页面，如图 2-4 所示，页面的字体大小现在不会重置了：

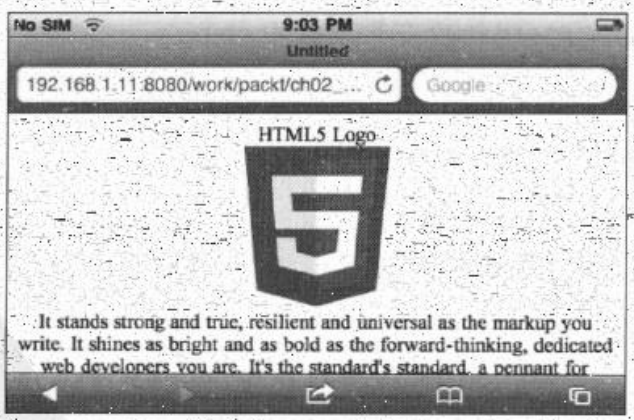


图 2-4

2.3.3 回顾

为了应对页面字体大小重置的问题，你需要在给页面添加一个名为“text-size-adjust”的CSS属性，并且把该属性的值设置为none，作用就是告诉WebKit[①]引擎在渲染该页面的时候不要自动调整文本字体大小。

通过设定text-size-adjust属性值为none可以解决在移动设备上访问web应用的字体大小重置问题，但是，如果在PC桌面访问，或者通过其他的非移动设备的浏览器访问，该设置会导致页面的缩放功能会被禁用。为了防止这种易用性的问题，可以把text-size-adjust的值从none改变为100%。所以，我们可以把上面的例子改进成这样：

```
html {
  -webkit-text-size-adjust: 100%;
}
```

2.3.4 延伸

除了iPhone之外，其他的移动设备同样也有方法设置“text-size-adjust”属性。

Windows Mobile

Windows Mobile IE中“text-size-adjust”属性使用了不同前缀名，他们原本也打算使用webkit作为该属性的前缀，因为这样可以和其他浏览器保持一致，从而可以降低Web开发人员的工作难度，开发人员不需要再去考虑在页面应该添加哪一些特殊前缀的“text-size-adjust”属性来控制文本字体大小缩放的问题。更有趣的是，微软认为，对于这个属性，最常用的情况应该是显示的设置为none，不要重置文本字体大小。

把“text-size-adjust”属性设置为none值带来的问题前面已经提到过，在听取过社区关于这个问题的讨论和反馈后，微软认为最好的方式是只实现以ms为前缀的版本而不是webkit版本。

因此，为了让前面的例子更加完整，你需要对代码做如下改变：

```
html {
  -webkit-text-size-adjust: 100%;
  -ms-text-size-adjust: 100%;
}
```

[①] WebKit是一个开源的浏览器引擎，苹果的safari和Google的Chrome都是基于这个引擎的，与之相对应的有Gecko（Firefox使用）和Trident（MSHTML，IE使用）。——译者注

拥抱未来

另外，我们可以在上面的例子中添加一行没有前缀的“text-size-adjust”属性，以便更好的应对未来变化：

```
html {
  -webkit-text-size-adjust: 100%;
  -ms-text-size-adjust: 100%;
  text-size-adjust: 100%;
}
```

px，em，谁更好？

在 Web 开发领域，关于应该使用 px（像素）还是 em（相对长度单位，相对于当前对象内文本的字体尺寸）的争论不绝于耳，但是，这个问题在移动互联网开发领域，争论并没有那么激烈，Yahoo！的用户接口原本使用的单位是 em，他们这么做的原因是 IE6 不支持 px 级别的缩放。但是，这在移动互联网开发领域，并不是问题，即使在 PC 的浏览器上，也不用太过在意这个问题，因为使用 IE6 的用户已经越来越少了。因此，在大部分场景下，你都可以使用像素来设置字体大小，抛开使用 em 遇到的各种问题和那些烦人的计算。

2.4 优化浏览器视口宽度设置

适用设备：基于浏览器的所有设备

每个移动设备都有自己默认的视口[①]宽度，具体的可以参见附录 X-关于所有移动设备默认视口宽度的列表。如果你不显示的设置它的值，在渲染页面的时候你可能会得不到你想要的效果。比方说，如果不设置 iPhone 的视口宽度，它将会按照 980 像素的宽度渲染页面，如果你的页面设计不是按照这个像素宽度做的，那么出来的效果就会不尽如人意了。

2.4.1 准备

创建一个名为“ch02r03.html”的 HTML 文件。

---

① 什么是视口（viewport）？为了让手机能获得良好的网页浏览体验，Apple 在移动版的 Safari 中引入了 viewport meta 标签，它的作用就是创建一个虚拟的视口（viewport），而这个虚拟窗口的分辨率接近于桌面显示器，Apple 将其默认为 980px，其他浏览器厂商也引入了这个技术，但是虚拟窗口的分辨率默认值就各有不同了。——译者注

2.4.2 实践

我们可以通过如下的方式来优化上面提到的视口宽度的问题：

首先，在刚才创建的 HTML 文件中添加如下代码，在移动设备的浏览器中访问该页面。

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">

  </head>
  <body>
    <header>
      HTML5 Logo
    </header>
    <div id="main">
    <h1>Lorem ipsum</h1>
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
enim ad minim veniam, quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor
in reprehenderit in voluptate velit esse cillum dolore eu fugiat
nulla pariatur. Excepteur sint occaecat cupidatat non proident,
sunt in culpa qui officia deserunt mollit anim id est laborum.
    </div>

  </body>
</html>
```

得到的结果如图 2-5 所示。

如果你在移动设备中实际运行了上面的例子，你会发现，在移动设备上，该页面的字体变得非常小，很难辨认。接下来，把视口的宽度设置为匹配设备宽度。

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">

  </head>
  <body>
    <header>
      HTML5 Logo
```

```
    </header>
    <div id="main">
    <h1>Lorem ipsum</h1>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed
    do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
    enim ad minim veniam, quis nostrud exercitation ullamco laboris
    nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor
    in reprehenderit in voluptate velit esse cillum dolore eu fugiat
    nulla pariatur. Excepteur sint occaecat cupidatat non proident,
    sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
    </div>


    </body>
    </html>
```

再次访问该页面，现在网页内容的宽度和设备的屏幕宽度相匹配，文字内容的易读性大大提高，如图 2-6 所示。

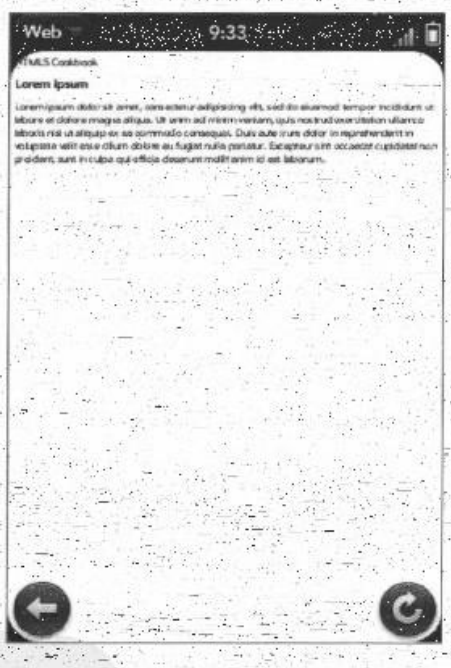


图 2-5

图 2-6

2.4.3 回顾

当我们把视口宽度设为匹配设备宽度时，该属性会告诉浏览器视口的宽度已经等于设备的可视域宽度，不需要缩放页面来适应设备的屏幕大小。因此对于 iPhone 来说，视口的宽度在竖屏模式下是 320px，横屏模式下是 480px。

2.4.4 延伸

有些古老的移动设备浏览器不能识别“viewport”属性，对于这些浏览器，你需要使用如下的代码来处理视口宽度的问题：

```
<meta name="HandheldFriendly" content="true">
```

这行代码主要用来应对老版本的 PalmOS、AvantGo 和黑莓设备。对于微软的 PocketPC，需要使用一个特别的“MobileOptimized”属性。

```
<meta name="MobileOptimized" content="320">
```

因此对于该问题最完整的解决方案的代码如下：

```
<meta name="HandheldFriendly" content="true">
<meta name="MobileOptimized" content="320">
<meta name="viewport" content="width=device-width">
```

关于 Windows Phone 上的 IE 浏览器的视口文档

在 Windows Phone 的 IE 浏览器开发团队的博客上，有一篇关于 Windows Phone 7 上的 IE 浏览器的视口问题的文章，在这篇文章中，作者给出了很多对开发者非常有用的信息，比方说，Windows Phone7 平台上的 IE 浏览器是如何实现“device-width”属性的等。你可以在如下的地址读到这篇文章：http://blogs.msdn.com/b/iemobile/archive/2010/11/22/the-ie-mobile-viewport-on-windows-phone-7.aspx。

关于 Safari 的视口文档

关于 Safari 上 viewport 问题，苹果的开发者库里有一篇关于这个问题的文章，请移步：http://developer.apple.com/library/safari/#documentation/appleapplications/reference/SafariHTMLRef/Articles/MetaTags.html.

关于黑莓的视口文档

这里有一篇在黑莓浏览器上做 Web 开发指南，它上面详细地讲解黑莓操作系统是怎样确定视口的宽度的，请移步：http://docs.blackberry.com/en/developers/deliverables/4305/BlackBerry_Browser-4.6.0-US.pdf。

2.5 修复移动版 Safari 的 re-flow scale 问题

适用设备：iOS 设备

移动版 Safari 有个很烦人的问题：当你从竖屏模式切换到横屏模式的时候，浏览器上的文本字体会突然变大。

在构建 Mobile Boilerplate 项目的时候，我和 Jeremy Keith 关于这个问题争论了很久，对于这个问题的常用解决方案是在 viewport 的 meta 属性中，再增加一些关于缩放的属性：

```
<meta name="viewport" content="width=device-width, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
```

之前，Mobile Boilerplate 也是使用的这种解决方案，但是，Jeremy 指出这个方案虽然解决了字体突然变大的问题，但是同时也引入了一个新的问题：当我们按照上面的方式设置页面之后，当前页面同时也失去了缩放的功能。对于那些视力不太好的用户，可以缩放是一个必要的功能，但是如果不限制缩放功能，其他的用户又会被字体大小突然变化所困扰。因此，在很长一段时间内，这个问题都是一个关于可用性和易用性的问题。我发现有一个方法可以解决这个问题，接下来我将为大家介绍这个方法。

2.5.1 准备

首先创建一个名为“ch02r04.html”的文件，并写入如下的代码：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>

  <div>
  <h1>Lorem ipsum</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
enim ad minim veniam, quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit anim id est laborum.</p>
  </div>
  </body>
</html>
```

在竖屏模式下访问该页面，如图 2-7 所示，渲染效果没有任何问题。

但是，在横屏模式下访问时，渲染效果就不尽如人意了，如图 2-8 所示。

图 2-7

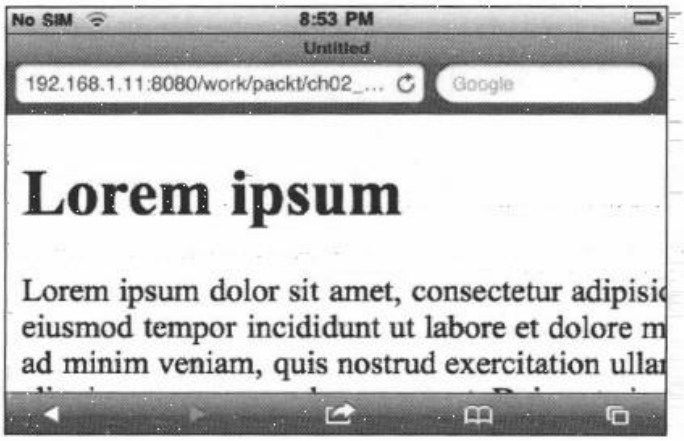


图 2-8

2.5.2 实践

对于上面提到的问题，我们需要做的就是，当用户在页面上使用缩放手势时，动态地把页面上的关于缩进功能的属性重置为默认值。接下来，把下面的代码写到前面的那个HTML文件中：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>


  <div>
  <h1>Lorem ipsum</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed
do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
enim ad minim veniam, quis nostrud exercitation ullamco laboris
nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in
```

```
reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit anim id est laborum.</p>
  </div>
    <script>
      var metas = document.getElementsByTagName('meta');
      var i;
      if (navigator.userAgent.match(/iPhone/i)) {
        for (i=0; i<metas.length; i++) {
          if (metas[i].name == "viewport") {
            metas[i].content = "width=device-width, minimum-scale=1.0,
maximum-scale=1.0";
          }
        }
        document.addEventListener("gesturestart", gestureStart,
false);
      }
      function gestureStart() {
        for (i=0; i<metas.length; i++) {
          if (metas[i].name == "viewport") {
            metas[i].content = "width=device-width, minimum-
scale=0.25, maximum-scale=1.6";
          }
        }
      }
    </script>
  </body>
</html>
```

现在，你不会再遇到横屏竖屏模式切换时遇到的字体变大的问题了，同时，你仍然可以像以前一样使用页面的缩放功能，如图 2-9 所示。

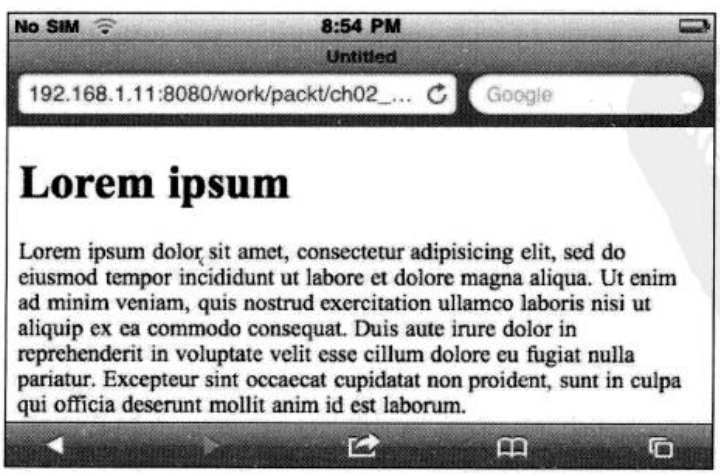


图 2-9

2.5.3 回顾

接下来，我们一步一步来分析一下上面的代码是如何工作的：

首先，我们需要知道页面最小可以缩小到多小，以及最大可以放大到多大。在 iPhone 的官方文档中明确指出，最小可以缩小到原大小的 1/4，最大可以放大到原大小的 1.6 倍，因此，为了能够把缩放的属性值设置为默认值，我们需要加入如下代码：

```
function gestureStart() {
  var metas = document.getElementsByTagName('meta');
  var i;
  for (i=0; i if (metas[i].name == "viewport") {
    metas[i].content = "width=device-width, minimum-
scale=0.25, maximum-scale=1.6";
  }
}
```

然后，我们需要知道在什么时候重置缩放属性的值？这非常容易做到：iPhone 自带了一个手势事件监听器，我们可以用它来监听页面上的手势事件，具体实现代码如下：

```
document.addEventListener("gesturestart", gestureStart, false);
```

最后，我们需要保证，该行为只发生在 iPhone 上，这也很容易做到：

```
if (navigator.userAgent.match(/iPhone/i)) {
document.addEventListener("gesturestart", gestureStart, false);
}
```

2.5.4 延伸

如果你对前面提到的我和 Jeremy 关于这个问题的讨论有兴趣，你可以在如下的网址读到关于这个讨论的整个过程：http://www.blog.highub.com/mobile-2/a-fix-for-iphone-viewport-scale-bug/。

尽管本章中提出了关于这个问题的一个解决方案，但是该方案并不完美，开发人员还是会遇到各种各样的问题，比如说：

- 一旦用户使用手势缩放，页面的缩放属性值就会被设置为默认值，之后，用户如果再在横屏模式和竖屏模式之间切换时，还是会遇到之前提到的字体突然变大的问题。
- iOS4 用户当第一次使用手势缩放时不起效果，第二次才能正常工作。

微改进版本

Mathias Bynens 对上面的代码进行了改进，你可以在下面的网址看到该版本的代码：https://gist.github.com/901295。

更好的版本

John-David Dalton 有一个更好的版本，代码更少，且能应对更多场景，你可以在下面的网址看到该版本的代码：https://gist.github.com/903131。

jQuery Mobile 版本

来自 jQuery Mobile 项目组的 Scott Jehl 提到，jQuery Mobile 在未来的版本中可能会解决这个问题，现在，你可以从下面的网址了解他们准备如何解决这个问题：https://gist.github.com/1183357。

2.6 在浏览器中启动手机原生应用

使用设备：基于浏览器的所有设备

用户可以在浏览器中启动移动设备的原生应用程序，比方说地图、电话、短信等，具体能够启动哪些应用程序，这取决于该移动设备上哪些原生应用是否允许从浏览器启动。

2.6.1 准备

创建一个名为“ch02r05.html”的 HTML 文件。

2.6.2 实践

下面，我们就来告诉大家如何从浏览器启动一个手机的原生的应用：

首先，把下面的代码写入刚才创建的 HTML 文件中：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

```
  </head>
  <body>
    <header>
      HTML5 Logo
    </header>
  <div>
    <h1>Lorem ipsum</h1>
    <a href="http://maps.google.com/maps?q=cupertino">
Directions</a>
  </div>

  </body>
</html>
```

然后，在 Palm OS 的浏览器中访问该页面，并点击“Directions”连接，该动作会启动你手机上原生的 Google Map 应用，如图 2-10 所示。

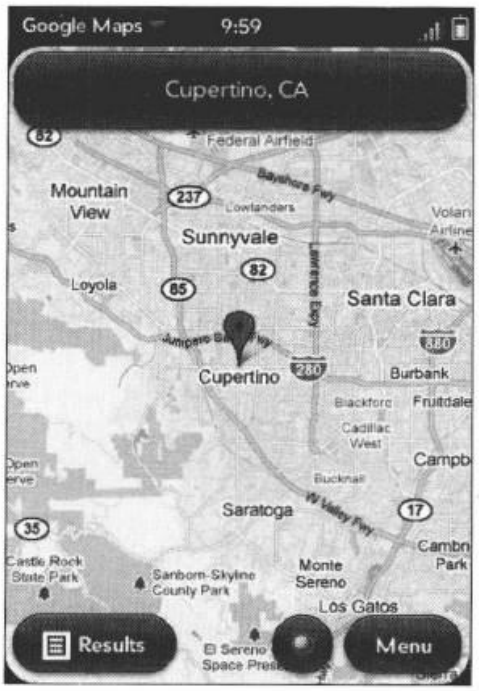


图 2-10

2.6.3 回顾

不像有些 URL Scheme[①]，map 的 URLS 并不是以“maps”前缀来作为 map 标签的标识，

① URL Scheme 是统一资源定位符（URL）的命名结构，通过定义自己的 URL Scheme 或者移动设备原生的 URL Scheme，开发人员可以找到本机文件、应用程序等资源。——译者注

map 的链接和普通的 HTTP 链接一样，但是该链接想要连接的服务器是 Google Maps 的服务器，移动设备的浏览器可以识别该请求是一个地图服务的请求，然后启动移动设备上原生的 Google Maps 服务，并把相应的参数传递给该应用程序。

2.6.4 延伸

除了启动移动设备上的原生应用，你还可以做更多的事情，比方说下面的这个 HTTP 连接将会启动用户设备上的 GoogleMaps 服务，并且告诉用户从 San Francisco 到 Cupertino 的行车路线：

```
<a href="http://maps.google.com/maps?daddr=San+Francisco,+CA&saddr=cupertino">Directions</a>
```

如果，浏览器不能启动移动设备的原生应用会发生什么事呢？没有关系，如果不能启动设备的原生应用程序，那么浏览器将像普通链接一样在浏览器中打开页面，如图 2-11 所示。

图 2-11

移动版 Safari 的 URL Scheme

如果你想了解移动版 Safari 到底支持哪些 URL Scheme，请移步：http://developer.

apple.com/library/safari/#featuredarticles/iPhoneURLScheme_Reference/Introduction/Introduction.html。

黑莓支持的 URL Scheme

如果你想了解黑莓到底支持哪些 URL Scheme，请移步：http://docs.blackberry.com/en/developers/deliverables/18169/。

索爱开发者指南

索爱的开发者们可以从索爱的开发者世界的网站下载到 Web 应用开发者指南，里面会有索爱设备支持的 URL Scheme：http://developer.sonyericsson.com/wportal/devworld/search-downloads?cat=%5B1.706817%2C+1.716594%2C+1.716688%5D&cc=gb&lc=en。

2.7 iPhone 下全屏模式启动

适用设备：iOS 设备

为了让一个 Web 应用看起来更像一个原生应用，iPhone 为 Web 应用开发者提供了很多独有的特性，你可以全屏模式下启动 Web 应用，可以添加一个启动界面，或者添加一个加载进度条之类的，或者为你的 Web 应用定义一个预加载的页面。

2.7.1 准备

从本章的源代码目录下载所有图片，创建一个名为“ch02r06.html”的 HTML 文件。

2.7.2 实践

下面我们就来看看如何能够让一个 Web 应用在 iPhone 上全屏启动。

首先，把下面的代码写入到刚才创建的 HTML 文件中：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <meta name="apple-mobile-web-app-capable" content="yes">
```

```
  <meta name="apple-mobile-web-app-status-bar-style"
content="black">
  <link rel="apple-touch-startup-image" href="img/l/splash.png">
  </head>
  <body>
    <header>
      HTML5 Logo
    </header>
    <div>
      Lorem ipsum
    </div>

  </body>
</html>
```

然后，把该页加载为浏览器的书签，然后通过界面图标启动它，你会发现，你的应用会以全屏的模式启动。

2.7.3 回顾

接下来，我们来分析一下，这段代码是如何做到全屏模式启动的：

```
<meta name="apple-mobile-web-app-capable" content="yes">
```

这段代码的意思是当 Web 应用从界面图标启动时，以全屏模式启动，隐藏浏览器上部的工具栏、地址栏和底部的加载状态栏。

```
<meta name="apple-mobile-web-app-status-bar-style" content="black">
```

这段代码的作用是在浏览器的顶部显示一个状态栏。

```
<link rel="apple-touch-startup-image" href="img/l/splash.png">
```

这段代码作用是，在程序启动、加载的时候，显示一个预加载的界面，告诉用户该程序正在加载。

2.7.4 延伸

iPad 和 iPhone 因为屏幕大小的差异，因此需要不同大小的预加载界面，因此，如果你希望你的 Web 应用可以动态地选择对应的预加载界面，你可以使用如下的 JavaScript 函数做到：

```
var filename = navigator.platform === 'iPad' ? 'h/' : 'l/';
    document.write('<link rel="apple-touch-startup-image" href="/img/'
+ filename + 'splash.png" />' );
```

iOS4.3 全屏模式的问题

iOS4.3 引入了一个新的特性，“JavaScript Nitro Engine”，这个新的特性能够让 Safari 加载页面的性能提升 2 倍，但是，全屏模式下的 Web 应用并不支持这个新特性。部分人在探索为什么苹果公司不把这个新的 Safari 特性和它的 Web 应用整合到一起，另一部分人在质疑这根本就是苹果公司产品的一个 Bug。

关于 Web 应用的 Safari 文档

如果你想查看关于 Safari 的官方文档，请移步：http://developer.apple.com/library/safari/#documentation/appleapplications/Reference/SafariWebContent/ConfiguringWebApplications/ConfiguringWebApplications.html。

Safari 的预加载图片和触摸图标开发指南

如果你对制作和使用预加载图片和触摸图标有兴趣，请移步：http://developer.apple.com/library/safari/#documentation/UserExperience/Conceptual/MobileHIG/IconsImages/IconsImages.html#//apple_ref/doc/uid/TP40006556-CH14。

2.8 防止 iOS 在聚焦时自动缩放

适用设备：iOS 设备

在 JavaScript 的事件 API 中，有一个叫做 onfocus 的事件，当你在 iOS 设备的浏览器上触碰一个表单元素时，这个元素会在屏幕上自动放大。对于一个非响应式设计的应用，或者未对移动设备做优化的应用，这样的缩进方式可以很好地改善用户体验，但是对于那些做了优化的应用，这样的自动缩放就会让用户很困扰了，我们可以通过修改视口 meta 标签的 onfocus 和 onblur 属性的值来禁用这个自动缩放的特性。

2.8.1 准备

创建一个名为“ch02r06_b.html”的 HTML 文件。

2.8.2 实践

下面我们就来讲解如何禁用表单元素自动缩放的功能：

首先，把如下的代码写入刚创建的 HTML 文件：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
  <form>

<label>First name:</label> <input type="text" name="fname" /><br
/>
<label>Last name:</label> <input type="text" name="lname" />
  </form>
  <script>
        var $viewportMeta = $('meta[name="viewport"]');
        $('input, select, textarea').bind('focus blur',
function(event) {
  $viewportMeta.attr('content', 'width=device-width,initial-
scale=1,maximum-scale=' + (event.type == 'blur' ? 10 : 1));
        });
  </script>
  </body>
</html>
```

然后在 iOS 设备上访问该页面，触碰表单的输入框，你会发现，该输入框不会再自动缩放。

2.8.3 回顾

接下来，我们把页面中的 JavaScript 代码拎出来单独分析一下：

```
<script>
        var $viewportMeta = $('meta[name="viewport"]');
        $('input, select, textarea').bind('focus blur',
function(event) {
  $viewportMeta.attr('content', 'width=device-width,initial-
scale=1,maximum-scale=' + (event.type == 'blur' ? 10 : 1));
        });
  </script>
```

这段代码的工作原理是，当浏览器检测到一个 onfocus 事件的发生，动态地把视口的最大缩放值修改为 1，当检测到 onblur 事件时，再动态地把视口的最大缩放值修改为 10。

2.8.4 延伸

你可以在如下的网址找到关于最初讨论这个问题的博客：http://nerd.vasilis.nl/prevent-ios-from-zooming-onfocus/。

本示例的代码已被收录在 Mobile Boilerplate 项目的源代码中：https://github.com/h5bp/mobile-boilerplate/blob/master/js/mylibs/helper.js。

2.9 禁用或限制部分 WebKit 特性

适用设备：使用基于 WebKit 引擎浏览器的设备（Android、iOS）

在移动版浏览器上，经常会遇到很多与具体设备相关的问题，通过使用一些比较少用的 CSS 技术，我们可以很容易地解决这些问题，下面我们就来介绍其中的几种。

2.9.1 准备

创建一个名为“ch02r07.html”的 HTML 文件。

2.9.2 实践

下面是一个关于限制 WebKit 引擎特性的小例子：

首先：把下面的代码写入刚创建的 HTML 文件中：

```
<!doctype html>
<html>
  <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <meta name="apple-mobile-web-app-capable" content="yes">
  <style>
  .nocallout {-webkit-touch-callout: none;}
  #targetarea {width:200px; height:120px; padding-top:80px;
background:#ccc; text-align:center; font-size:20px;}
  </style>
  </head>
  <body>

    <div id="targetarea" class="nocallout">
      <a href="http://www.google.com" target="_blank">Google</a>
```

```
    </div>

  </body>
</html>
```

然后，在移动设备中访问该页面，并点击“Google”连接，结果会如图 2-12 所示。

图 2-12

接着，修改页面代码为下面的这段代码：

```
<!doctype html>
<html>
 <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
 <meta name="apple-mobile-web-app-capable" content="yes">
 <style>
  .nocallout {-webkit-touch-callout: none;}
  #targetarea {width:200px; height:120px; padding-top:80px;
background:#ccc; text-align:center; font-size:20px;}
  </style>
  </head>
  <body>

    <div id="targetarea" class="nocallout">
      <a href="http://www.google.com" target="_blank">Google</a>
    </div>
```

```
    </body>
</html>
```

然后，在浏览器中再次访问该文件，并点击“Google”链接，结果如图 2-13 所示。

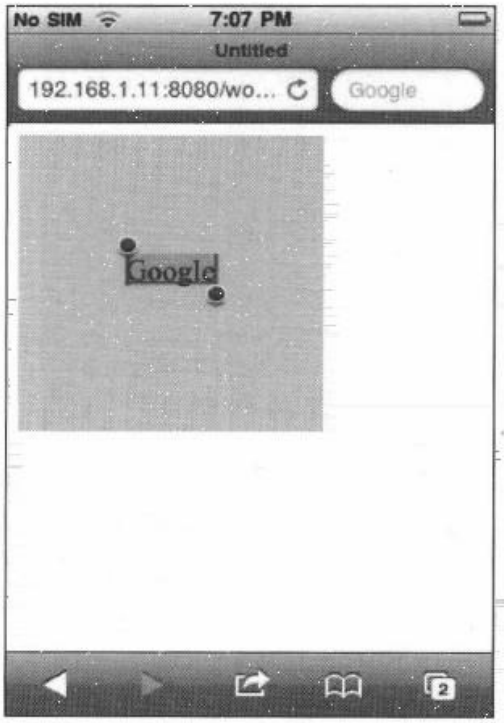


图 2-13

2.9.3 回顾

如果不设置“webkit-touch-callout”属性，当你在移动设备上点击一个链接时，设备会弹出一个对话框，询问你是想要在当前页面打开该链接，在新的页面打开该链接还是想要复制该链接的地址，就像你在上面第一个例子中看到的那样。

如果你想禁用这个特性，你可以把“web-touch-callout”属性的值设为 none，出来的效果就像上面的第二个例子一样，如图 2-13 所示。

2.9.4 延伸

你可能还想要限制应用的复制粘贴功能，这对于一个 web 页面来说也许是有用的，但是对于大多数应用接口来说，并没有多大意义。

```
<style type="text/css">
  .oncopy {
    -webkit-user-select: text;
  }
</style>
```

修改点击之后的背景颜色

你可以使用如下的 CSS 文件 tap 的颜色：

```
<style type="text/css">
    * {
        -webkit-tap-highlight-color: rgba(0,0,0,0);
    }
</style>
```

让文本区内容可编辑

如果你想让某个界面元素的内容可编辑，你可以使用下面的 CSS：

```
textarea.contenteditable {
  -webkit-appearance: none;
}
```

为狭窄的屏幕添加省略号功能

在移动版浏览上，因为移动设备的屏幕普遍比较狭窄，因此当你在显示一个菜单列表项的时候，可能其中的几项的字符长度会超出屏幕大小，你可以通过下面的 CSS 来告诉浏览器，当文本内容长度超出屏幕宽度的时候，以省略号代替：

```
.ellipsis {
  text-overflow: ellipsis;
  overflow: hidden;
  white-space: nowrap;
}
```

第3章
移动设备的交互方式

本章内容如下：

- 利用触控来移动页面元素
- 监测和处理横竖屏切换事件
- 利用手势来旋转页面元素
- 利用滑动来创建图库
- 利用手势进行图片缩放

3.1 简介

移动设备与桌面设备最大的不同在于交互方式，在桌面设备上，我们利用鼠标移动和点击来交互，而在移动设备上，交互来自触控和手势。本章我们将看到触摸屏的特有交互（例如：双指触摸）以及如何利用这些特性创建移动设备上的特有功能。

3.2 利用触控来移动页面元素

适用设备：跨浏览器

在移动设备上，我们利用触控与页面元素交互，因此，我们可以使用手指来移动页面元素。

3.2.1 准备

我们将在本示例中使用 jQuery。首先，创建一个新的 HTML 文件并命名为

ch03r01.html。

3.2.2 实践

在HTML文件中输入以下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <style>
    #square {
      width: 100px;
      height: 100px;
      background:#ccc;
      position:absolute;
    }
    </style>
  </head>
  <body>

    <div id="main">
      <div id="square">
      </div>
    </div>

    <script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
    <script src="http://code.jquery.com/mobile/1.0a4.1/jquery.
mobile-1.0a4.1.min.js"></script>
    <script>
    $('#square').bind('touchmove',function(e){
      e.preventDefault();
      var touch = e.originalEvent.touches[0] || e.originalEvent.
changedTouches[0];
      var elm = $(this).offset();
      var x = touch.pageX - elm.left/2;
      var y = touch.pageY - elm.top/2;
      $(this).css('left', x+'px');
      $(this).css('top', y+'px');
    });
  </script>
  </body>
</html>
```

现在看看它在Opera中的显示，如图3-1所示。

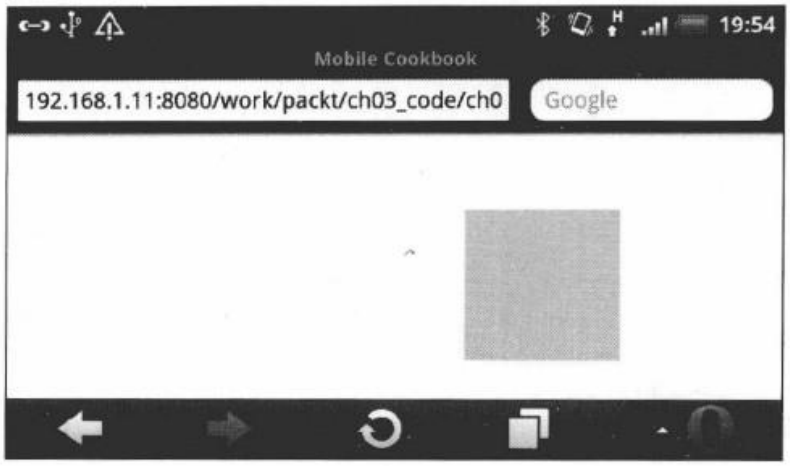


图 3-1

3.2.3 回顾

首先，我们给 div 注册了 touchmove 事件。

利用 touch.pageX 和 touch.pageY 可以获取相对于页面的触摸位置，我们将触摸位置减去 div 高度和宽度的一半，这样 div 的拖动点就在它的中心点上。

```
var x = touch.pageX - elm.left/2;
var y = touch.pageY - elm.top/2;
```

将 x 和 y 的值赋给 div 的 CSS 位置属性，它就可以移动了。

```
$(this).css('left', x+'px');
$(this).css('top', y+'px');
```

3.2.4 延伸

在本示例中，最开始有下面一行代码：

```
var touch = e.originalEvent.touches[0] || e.originalEvent.
changedTouches[0];
```

它的作用在于，移动版 Safari 不允许 event 对象的 touches 和 changedTouches 属性被拷贝给其他对象，我们可以使用 e.originalEvent 来解决这个问题。这里有更多详细的信息：

```
http://www.the-xavi.com/articles/trouble-with-touch-events-jquery
```

jQuery 移动版事件

jQuery 移动版是一系列的组件，这里有移动相关所有事件的详细说明：

```
https://github.com/shichuan/jquery-mobile/blob/master/js/jquery.mobile.event.js
```

Zepto

如果你主要的目标浏览器是基于 WebKit，那么你可以考虑 Zepto，这个轻量级的 jQuery 替代品，这里有关于它的信息：

```
https://github.com/madrobby/zepto
```

Safari 关于移动事件处理的指南

Safari 指南的官方文档：

```
http://developer.apple.com/library/safari/#documentation/appleapplications/
reference/safariwebcontent/HandlingEvents/ HandlingEvents.html
```

3.2.5 参考

- 3.3 节
- 3.4 节
- 3.5 节
- 3.6 节

3.3 检测和处理横竖屏切换事件

适用设备：跨浏览器

对于移动浏览器，如果网站是基于流体布局的，那么在横竖屏切换的时候，网站样式也应该有相应的变化。但是对于富交互的网站，有时也需要以特别的方式来处理横竖屏切换。

3.3.1 准备

创建一个新的 HTML 文件，命名为 `ch03r02.html`。

3.3.2 实践

现在开始利用 HTML 和脚本代码监测和处理横竖屏切换事件。

1. 输入以下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <style>
    html, body {
      padding: none;
      margin: none;
    }
    </style>
    <link rel="stylesheet" href="http://code.jquery.com/
mobile/1.0/jquery.mobile-1.0.min.css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"></
script>
    <script src="http://code.jquery.com/mobile/1.0/jquery.mobile-
1.0.min.js"></script>
  </head>
  <body>
    <div id="a">
    </div>
    <script>
      var metas = document.getElementsByTagName('meta');
      var i;
      if (navigator.userAgent.match(/iPhone/i)) {
        for (i=0; i<metas.length; i++) {
          if (metas[i].name == "viewport") {
            metas[i].content = "width=device-width,
minimum-scale=1.0, maximum-scale=1.0";
          }
        }
        document.addEventListener("gesturestart", gestureStart,
false);
      }
      function gestureStart() {
        for (i=0; i<metas.length; i++) {
```

```
            if (metas[i].name == "viewport") {
              metas[i].content = "width=device-width,
minimum-scale=0.25, maximum-scale=1.6";
            }
          }
        }
      </script>

      <script>
        $(window).bind('orientationchange',function(event){
          updateOrientation(event.orientation);
        })
        function updateOrientation(orientation) {
          $("#a").html("<p>"+orientation.toUpperCase()+"</p>");
        }
      </script>
    </body>
</html>
```

2．现在，在移动浏览器中打开该页面并查看横屏和竖屏中的样式，在竖屏模式，输出的文字为“PORTAIT”模式，如图 3-2 所示。

3．当切换到横屏模式，文字变为“LANDSCAPE”模式，如图 3-3 所示。

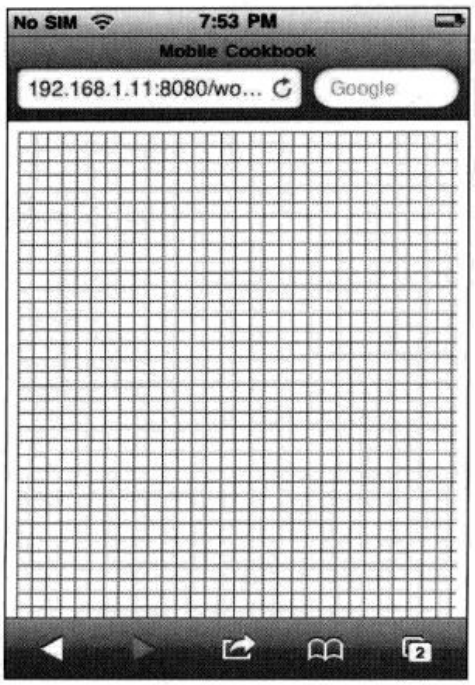


图 3-2

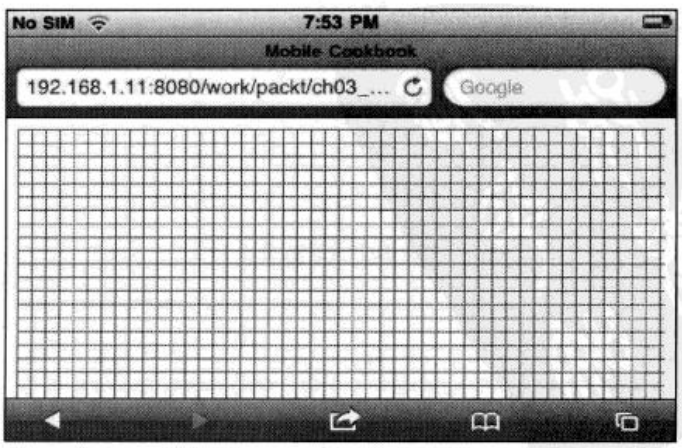


图 3-3

3.3.3 回顾

通过监听 window.onorientationchange 事件，当横竖屏切换事件触发时，orientationchange 会被触发，同时我们得到 event.orientation 作为参数传递给该方法并输出结果。

3.3.4 延伸

有时你需要禁止横竖屏的自动切换，比如开发游戏。对于原生应用这很容易，但对于网页应用非常困难。

现在我们创建一个单页面并锁定为横屏模式，注意这只是一个示例，对于真正的复杂应用或游戏，需要更加复杂的计算和处理。

创建一个文件命名为 ch03r02_b.html，输入以下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <link rel="stylesheet" href="css/style.css">

    <style>
    body {
      font-family: 'Kranky', serif;
      font-size: 36px;
    font-style: normal;
    font-weight: 400;

    word-spacing: 0em;
    line-height: 1.2;
  }
  html {
    background:#F1F2CE;
  }
  html, body, #screen {
    padding:0;
    margin:0;
  }
  #screen {
    text-align:center;
```

```
        -moz-transform:rotate(90deg);
        -webkit-transform:rotate(90deg);
        -o-transform:rotate(90deg);
        -ms-transform:rotate(90deg);
      }
      #screen div {
        padding-top:130px;
      }
      @media screen and (min-width: 321px){
        #screen {
          text-align:center;
          -moz-transform:rotate(0deg);
          -webkit-transform:rotate(0deg);
          -o-transform:rotate(0deg);
          -ms-transform:rotate(0deg);

        }
        #screen div {
          padding-top:70px;
        }
      }
    </style>
  </head>
  <body>
    <div id="screen">
      <div id="loader">enter the game</div>
    </div>
    <script>
      var metas = document.getElementsByTagName('meta');
      var i;
      if (navigator.userAgent.match(/iPhone/i)) {
        for (i=0; i<metas.length; i++) {
          if (metas[i].name == "viewport") {
            metas[i].content = "width=device-width, minimum-scale=1.0,
maximum-scale=1.0";
          }
        }
        document.addEventListener("gesturestart", gestureStart,
false);
      }
      function gestureStart() {
        for (i=0; i<metas.length; i++) {
          if (metas[i].name == "viewport") {
            metas[i].content = "width=device-width,
minimum-scale=0.25, maximum-scale=1.6";
          }
        }
```

```
        }
        window.onorientationchange = function() {
          update();
        }
        function update() {
          switch(window.orientation) {
            case 0:   // Portrait
            case 180: // Upside-down Portrait
              var cWidth = window.innerWidth;
              var cHeight = window.innerHeight;
              document.getElementById("screen").style.width = cHeight-
36+'px';
              document.getElementById("screen").style.height =
cWidth+'px';
              break;
            case -90: // Landscape: turned 90 degrees counter-clockwise
            case 90:  // Landscape: turned 90 degrees clockwise
              var cWidth = window.innerWidth;
              var cHeight = window.innerHeight;
              document.getElementById("screen").style.width = "100%";
              document.getElementById("screen").style.height = "auto";
              break;
          }
        }
        update();
      </script>

    </body>
</html>
```

现在在浏览器中打开该文件，将如图 3-4 所示。在竖屏模式中暗示用户游戏或应用是针对横屏设计的。

转换为横屏模式，则显示正常，如图 3-5 所示。

本例中，我们利用 CSS3 的 transform: rotate，在竖屏模式使显示旋转了 90 度：

```
#screen {
  text-align:center;
  -moz-transform:rotate(90deg);
  -webkit-transform:rotate(90deg);
  -o-transform:rotate(90deg);
  -ms-transform:rotate(90deg);
}
```

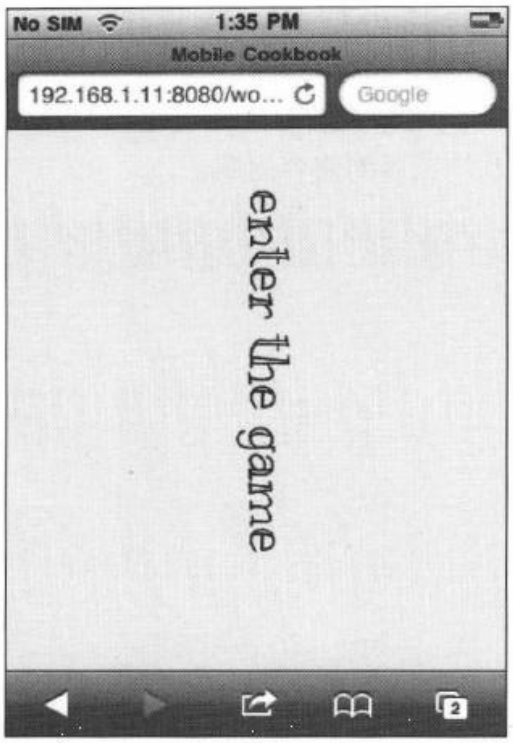


图 3-4

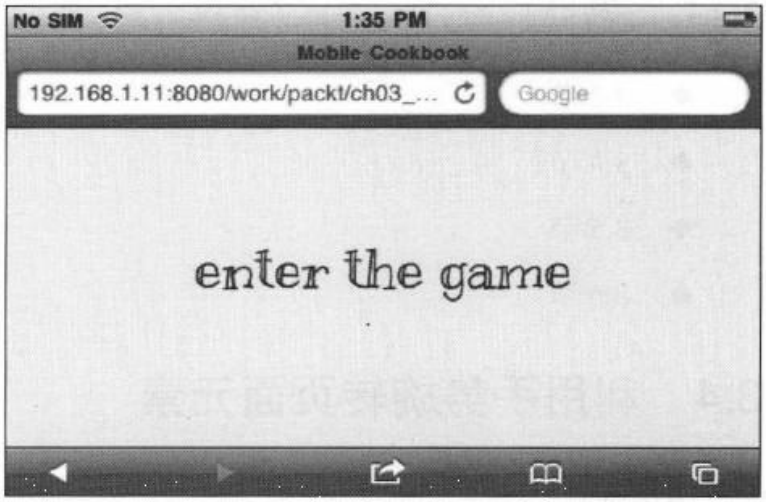


图 3-5

使用 window.orientation 可以检测当前的屏幕模式，它有 4 个值：−90、0、90、180。横屏时的值为−90 或 90，竖屏时的值为 0 或 180。

```
switch(window.orientation) {
  case 0:   // Portrait
  case 180: // Upside-down Portrait
  //...
  break;
  case -90: // Landscape: turned 90 degrees counter-clockwise
  case 90:  // Landscape: turned 90 degrees clockwise
  //...
  break;
}
```

以上代码可以检测屏幕模式。

Safari 的原生支持

以下为官方文档：

http://developer.apple.com/library/safari/#documentation/appleapplications/reference/safariwebcontent/HandlingEvents/ HandlingEvents.html

网站与原生应用

虽然移动网站发展迅速，但请记住对于富交互应用，即使是最慢的原生应用也比 HTML 应用快。如果决定使用 HTML5 来构建应用，你必须注意浏览器的兼容问题。

3.3.5 参考

- 3.2 节
- 3.4 节
- 3.5 节
- 3.6 节

3.4 利用手势旋转页面元素

适用设备：iOS、Android、Symbian

当用户在移动版 Safari 中使用两指触控来旋转，我们可以检测到旋转的角度，因此，我们可以使用两指触控来旋转页面元素。

3.4.1 准备

创建一个 HTML 文件，命名为 `ch03r03.html`。

3.4.2 实践

1. 在 `ch03r03.html` 中输入以下代码，并在移动浏览器中打开：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0">
    <style>
    #main {
      text-align:center;
    }
```

```
    #someElm {
      margin-top:50px;
      margin-left:50px;
      width: 200px;
      height: 200px;
      background:#ccc;
      position:absolute;
    }
    </style>
  </head>
  <body>

    <div id="main">
      <div id="someElm">
      </div>
    </div>


    <script>
    var rotation =0 ;
    var node = document.getElementById('someElm');

    node.ongesturechange = function(e){

      var node = e.target;
      //alert(e.rotation);
      // scale and rotation are relative values,
      // so we wait to change our variables until the gesture ends
      node.style.webkitTransform = "rotate(" + ((rotation +
e.rotation) % 360) + "deg)";
      //alert("rotate(" + ((rotation + e.rotation) % 360) +
"deg)");
    }

    node.ongestureend = function(e){
      // Update the values for the next time a gesture happens
      rotation = (rotation + e.rotation) % 360;
    }
    </script>
  </body>
</html>
```

2. 现在用两指触控旋转灰色区域，结果如图 3-6 所示。

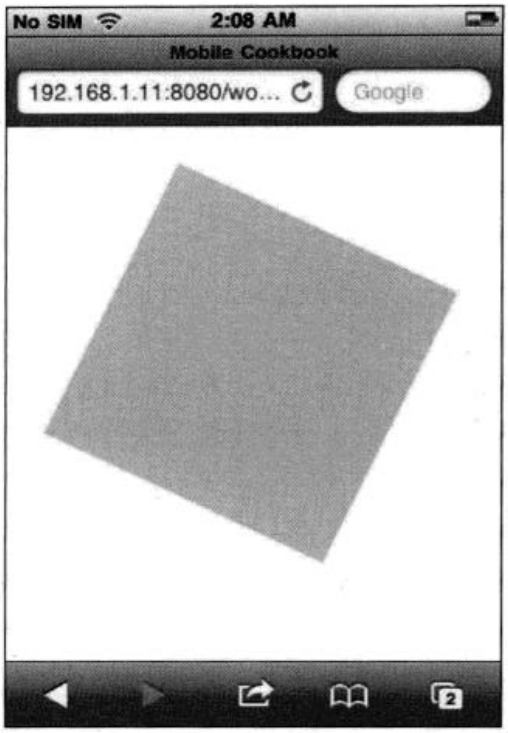


图 3-6

3.4.3 回顾

本例中，我们利用 ongesturechange 事件来旋转页面元素，同时通过 e.target.rotation 的值来获取旋转的角度。

3.4.4 延伸

在代码中对 ongestureend 事件的监听，是为了记录用户上一次的旋转角度，并且作为当前旋转的起始角度。

Safari 事件处理

以下为官方文档地址：

http://developer.apple.com/library/safari/#documentation/appleapplications/reference/safariwebcontent/HandlingEvents/ HandlingEvents.html

CSS3 变换

本例使用了 CSS3 的变换功能，你可以在 WebKit 的博客找到更多关于 WebKit 和 CSS

变换的信息：

http://www.webkit.org/blog/130/css-transforms/

修复缩放 bug 带来的问题

本例使用了 maximum-scale=1.0 来禁止通过手势进行缩放操作，这会导致一些可用性的问题，所以只在创建富交互网页应用时才使用旋转事件，在创建移动网站时不要使用。

3.4.5 参考

- 3.2 节
- 3.3 节
- 3.4 节
- 3.6 节

3.5 利用滑动创建图库

移动设备上最常用的功能之一就是滑动。在照片库中浏览照片时，我们会左右滑动来切换照片。在 Android 设备中，我们会向下滑动来解锁。在移动网站中，同样可以使用滑动。

3.5.1 准备

首先，创建一个 HTML 文件，命名为 ch03r04.html。

3.5.2 实践

1. 输入以下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <style>
      html, body {
```

```
        padding:0;
      margin:10px auto;
    }
    #checkbox {
      border:5px solid #ccc;
      width:30px;
      height:30px;
    }
    #wrapper {
      width:210px;
      height:100px;
      position:relative;
      overflow:hidden;
      margin:0 auto;
    }
    #inner {
      position:absolute;
      width:630px;
    }
    #inner div {
      width:200px;
      height:100px;
      margin:0 5px;
      background:#ccc;
      float:left;
    }
    .full-circle {
       background-color: #ccc;
       height: 10px;
       -moz-border-radius:5px;
       -webkit-border-radius: 5px;
       width: 10px;
       float:left;
       margin:5px;
    }
    .cur {
      background-color: #555;
    }
    #btns {
      width:60px;
      margin:0 auto;
    }
  </style>
</head>
<body>

  <div id="main">
```

```
        <div id="wrapper">
          <div id="inner">
            <div></div>
            <div></div>
            <div></div>
          </div>
        </div>
        <div id="btns">
          <div class="full-circle cur"></div>
          <div class="full-circle"></div>
          <div class="full-circle"></div>
        </div>
      </div>

      <script src="http://code.jquery.com/jquery-1.5.2.min.js"></
script>
      <script src="http://code.jquery.com/mobile/1.0a4.1/jquery.
mobile-1.0a4.1.min.js"></script>
      <script>
      var curNum = 0;
      $('#wrapper').swipeleft(function () {
        $('#inner').animate({
        left: '-=210'
        }, 500, function() {
          // Animation complete.
          curNum +=1;
          $('.full-circle').removeClass('cur');
          $('.full-circle').eq(curNum).addClass('cur');
        });
      });

      $('#wrapper').swiperight(function () {
        $('#inner').animate({
        left: '+=210'
        }, 500, function() {
          // Animation complete.
          curNum -=1;
          $('.full-circle').removeClass('cur');
          $('.full-circle').eq(curNum).addClass('cur');
        });
      });
      </script>
    </body>
  </html>
```

2. 在移动浏览器中打开，并在灰色区域左右滑动，你可以看到它随着手指滑动在左右移动，如图3-7所示。

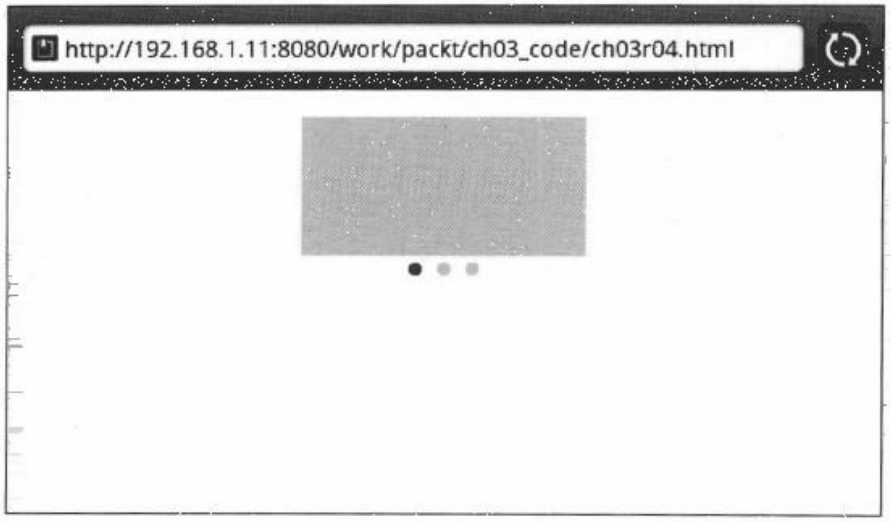


图 3-7

3.5.3 回顾

本例使用了多个 HTML5 技术。首先，我们使用 jQuery Mobile 检测滑动事件，下面的代码会监听手指的左右滑动：

```
$('#wrapper').swipeleft(function () {
});
$('#wrapper').swiperight(function () {
});
```

当滑动事件触发，利用 jQuery 的动画接口 .animate() 来创建移动效果：

```
$('#inner').animate({
      left: '+=210'
      }, 500, function() {
        // Animation complete.
        curNum -=1;
        $('.full-circle').removeClass('cur');
        $('.full-circle').eq(curNum).addClass('cur');
      });
```

3.5.4 延伸

本例中使用了 CSS3 来创建圆形按钮，我们可以仅使用 CSS3 来画圆：

```
.full-circle {
  background-color: #ccc;
  height: 10px;
  -moz-border-radius:5px;
  -webkit-border-radius: 5px;
  border-radius: 5px;
  width: 10px;
}
```

上面的例子定义了一个高、宽 10px，并且带有 5px 圆角的元素样式，我们可以使用类似的几行 CSS 代码，就创建出完美的圆形！

Zepto 框架与滑动事件

我们可以使用 Zepto 框架完成类似的功能，他提供了 swipe、swipeLeft、swipeRight、swipeUp、swipeDown。

YUI 与手势事件

YUI 提供了手势事件来创建滑动效果，下面是连接地址：

http://yuiblog.com/sandbox/yui/3.3.0pr3/examples/event/swipe-gesture.html

深入了解代码

jQuery Mobile 以模块化的方式构建了事件机制，了解 jQuery Mobile 如何实现滑动效果可以访问：

https://github.com/jquery/jquery-mobile/blob/master/js/jquery.mobile. event.js

滑动事件的相关代码在：

```
$.event.special.swipe = {…}
```

横向滑动、纵向滑动的检测和距离临界值都包含在事件的计算中。

3.5.5 参考

- 3.2 节
- 3.3 节
- 3.4 节
- 3.6 节

3.6 利用手势操作图片缩放

在 iPhone 中，我们可以基于缩放检测来放大或缩小页面元素。当缩放手势触发，我们可以获取缩放的信息，并且基于此放大或缩小页面元素。

3.6.1 准备

创建一个 HTML 文件，命名为 ch03r05.html。

3.6.2 实践

输入以下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0, user-scalable=no">
    <style>
      #frame {
        width:100px;
        height:100px;
        background:#ccc;
      }
    </style>
  </head>
  <body>

    <div id="main">
      <div id="frame"></div>
    </div>

    <script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
    <script src="http://code.jquery.com/mobile/1.0a4.1/jquery.mobile-
1.0a4.1.min.js"></script>
    <script>
    var width = 100, height = 100;
    var node = document.getElementById('frame');
    node.ongesturechange = function(e){
```

```
        var node = e.target;
        // scale and rotation are relative values,
        // so we wait to change our variables until the gesture ends
        node.style.width = (width * e.scale) + "px";
        node.style.height = (height * e.scale) + "px";
      }
      node.ongestureend = function(e){
        // Update the values for the next time a gesture happens
        width *= e.scale;
        height *= e.scale;
      }
      </script>
    </body>
  </html>
```

3.6.3 回顾

本例中，我们给被缩放的元素注册了 ongesturechange 事件，缩放的比例根据 e.target.scale 计算：

```
width *= e.scale;
height *= e.scale;
```

3.6.4 延伸

手势事件有许多问题，所以适当的使用是非常重要的。对于两指多点触摸，事件的触发顺序如下：

1. 手指 1 的 touchstart：当手指触摸到屏幕时触发。
2. gesturestart：当第二根手指触摸屏幕时触发。
3. 手指 2 的 touchstart：当第二根时候触摸屏幕时，紧接着 gesturestart 触发。
4. 当前手势的 gesturechange：当两根手指保持触摸并开始移动时触发。
5. gestureend：当第二根手指离开屏幕时触发。
6. 手指 2 的 touchend：当第二根手指离开屏幕时，紧接着 gestureend 触发。
7. 手指 1 的 touchend：当第一根手指离开屏幕时触发。

iOS Safari 关于手势事件的官方文档

以下 iPhone Safari 官方文档详细介绍了手势事件：

http://developer.apple.com/library/safari/#documentation/UserExperience/Reference/GestureEventClassReference/GestureEvent/ GestureEvent.html

YUI 手势事件

雅虎的 YUI 有一套关于跨浏览器手势的解决方案，但仅支持单指事件：

http://developer.yahoo.com/yui/3/event/#gestures

Google 地图与手势事件

两指手势的一个重要使用示例，是移动版 Safari 中的 Google 地图，如图 3-8 所示。

图 3-8

3.6.5 参考

- 3.2 节
- 3.3 节
- 3.4 节
- 3.5 节

第 4 章
构建快速响应式移动互联网站点

本章内容概括：

- 使用 HTML5 语法构建页面
- 使用 CSS3 特性做渐进增强
- 使用响应式设计
- 优化 polyfills 脚本加载速度
- 检测用户客户端
- 使用书签气泡为应用添加桌面快捷方式
- 构建可自动伸缩的文本输入框
- 加速按钮反馈
- 隐藏浏览器地址栏
- 构建移动互联网站点的站点地图

4.1 简介

一般来说，在移动设备上，网络连接的速度比 PC 上要差很多，尤其是当你使用的是速度非常慢的 3G 网络时，网络速度会更加糟糕，即使是使用速度非常快的 WIFI 热点。移动设备上，浏览器的渲染速度也会比 PC 上慢很多。因此，在开发移动互联网站点时，你必须考虑快速响应式设计。

在本章中，我们将介绍 HTML5 的特性。HTML5 是由一系列的技术组合而成，其中包

括：新的语义标签，新的 CSS 规则和属性，新的 JavaScript API，这些技术可以帮助我们构建更加结构化，更加强大的 Web 应用。下面就是 HTML5 带来的 8 大新特性：

- 语义特性
- 离线应用与本地存储特性
- 设备兼容特性
- 连接特性
- 网页多媒体特性
- 3D 图形效果特性
- 性能与集成特性
- CSS3 特性

这些特性并不是专为移动设备而生的，只不过其中的几个特性与移动开发更加相关，而其他的特性则对桌面浏览器和移动浏览器是通用的，接下来我们会一一讨论这些特性，看看如何在移动互联网开发的过程中更好地运用它们。

现在，我们将通过使用最新的语义标签和 CSS3 技术的实际例子，来仔细分析如何使用这些独有的技术，并通过不同的手法，让你的移动互联网站点能覆盖到更多的移动设备浏览器。

4.2 使用 HTML5 语法构建页面

适用设备：基于浏览器的所有设备

HTML5 引入了很多新的标签；这些标签对构建页面结构非常有帮助，且语义清晰。新的语法标签是 HTML5 的一个重要组成部分。

我们就不一一去看所有的 HTML5 语义标签了，这里我们侧重来讲几个最常用的。

4.2.1 准备

首先，创建一个名为“ch04r01.html”的 HTML 文件。假设我们即将创建一个关于音乐的虚构的网站。

4.2.2 实践

在你上一步创建的 HTML 文件中写入如下代码：

```
<!doctype html>
<html>
  <head>
    <title>first.fm</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <style>
    </style>
  </head>
  <body>
    <header>
      <h1>first.fm</h1>
    </header>
    <div id="main">
      <h2>Pages</h2>
        <nav>
          <ul>
            <li class="list"><a href="music.html">Music</a></li>
            <li><a href="radio.html">Radio</a></li>
            <li><a href="events.html">Events</a></li>
            <li><a href="charts.html">Charts</a></li>
            <li><a href="community.html">Community</a></li>
            <li><a href="help.html">Help</a></li>
            <li><a href="about.html">About</a></li>
          </ul>
        </nav>
    </div>
    <footer>
      <small>&copy; 2011 first.fm</small>
    </footer>
  </body>
</html>
```

4.2.3 回顾

header 标签经常用于封装 h1 至 h6 标签，它可以使 h 类型的标签看起来像是整个页面的头部或者一个页面块的头部，header 标签内通常还会使用 title、subtitle 或者 tagline 标签，header 标签用法如下：

```
<header>
</header>
```

nav 标签表示一个文件的导航，该标签一般是用来封装链接的，这些链接可能是链接到其他文档，或者是本文档的某个部分。

并不是页面上的每一个链接都需要被放置到 nav 标签下，它只是用来把重要的链接或者说二级链接归类，最常用的用法就是把 footers 里面的关于该网站的各个链接归于一个组，但是 HTML5 提供了一个更好的标签来处理这个问题——footer。nav 标签的用法如下：

```
<nav>
  <ul class="list">
    <li class="list"><a href="music.html">Music</a></li>
  </ul>
</nav>
```

footer 标签是用来表示一个文档或者一个页面块的底部，footer 标签一般用来放一些关于该页面或者该块的基本信息，比方说版权归属，谁开发的，以及一些相关页面的链接等等。在 footer 标签中的关于联系地址的信息应该放到 address 标签下。footer 标签用法如下：

```
<footer>
  <small>&copy; 2011 first.fm</small>
</footer>
```

small 标签可以用于把一些附属细则变为小号字体，因为这些附属细则并不需要成为页面上的焦点。small 标签不适用于段落或者一长段文字，它主要应用于像版权信息之类的短小文本。

small 标签的用法如下：

```
<small>&copy; 2011 first.fm</small>
```

4.2.4 延伸

HTML5 的新语法不仅仅是增加了一堆新的标签而已，我们除了想要更有意义的标签，还希望整个平台除了标签之外，具有更大的扩展性，这样我们就可以自己添加一些需要的功能，而 HTML5 提供了一种新的机器可读的语法，让数据可以被浏览器、脚本或者机器所理解，这样，开发人员就可以以数据驱动来构建整个程序的界面，这样的界面对用户来说也是更加友好的。HTML5 中实现这种语法的规范有：**RDFa（Resource Description Framework-In-attributs，属性级资源描述框架）、Microdata 和 Microformats。**

RDFa

RDFa 提供了一系列机器可读的 HTML 属性，通过使用 RDFa，开发人员可以避免重复地把一些人类可读的信息转化为机器可读的信息，这里有关于 RDFa 最新版本的特性说明文档：

```
http://www.w3.org/TR/rdfa-in-html/
```

Microdata

Microdata 是通过使用属性来定义一组内嵌的键值对以表达页面内容元素的含义。你可以在这个网站了解更多关于 Microdata 的信息：`http://html5doctor.com/microdata/`。

你可以阅读 W3C 的工作草案来更加深入的了解 Microdata：`http://www.w3.org/TR/microdata/`。

你还可以在这儿读到 W3C 的编辑者草案：`http://dev.w3.org/html5/md/`。

Microformats

Microformats 的设计原则是人读第一，机读第二。目前已经有 34 种 Microformats 规格了。其中一些已经发布，还有一些仍在草稿阶段，你可以从下面这个网站了解更多信息：`http://html5doctor.com/microformats/`

4.2.5 参考

- 1.5 节
- 1.7 节

4.3 使用 CSS3 特性做渐进增强

适用设备：基于浏览器的所有设备

CSS3（图 4-1）通过一系列的样式和特效来增强 Web 应用和互联网站点的设计和用户体验。通过 CSS3，开发人员可以不使用图片创建出很多绚丽的 UI，对于移动互联网开发来说，更少的图片就意味着更快的加载，这是一个提升性能的技巧。目前大多数的智能手机的浏览器都已经支持 CSS3 了，对于那些不支持 CSS3 的移动设备浏览器，我们可以使用一些 polyfills[①]（通过使用一些 Javascript 为那些不支持 HTML5 的浏览器添加上 HTML5 的特性）

图 4-1

[①] polyfills 指的是通过一小段 javascript 代码，把新的标准的 API 功能移植到老版本的浏览器上。——译者注

来实现相应的功能，因此，大家可以放心大胆地开始使用 CSS3 技术。

4.3.1 准备

接下来我们为上一小节创建的 HTML 文件添加一个样式，首先，复制文件“ch04r01.html”，把新文件命名为“ch04r02.html”。

4.3.2 实践

在“ch04r02.html”文件中加入下面的样式规则：

```
<style>
    body {
      margin:0;
      padding:0;
      font-family:Arial;
      background:#ccc;
    }
    header {
      text-shadow: 0 1px #000;
      background: #ff3019; /* Old browsers */
      background: -moz-linear-gradient(top, #ff3019 0%, #cf0404 20%,
#ff3019 100%); /* FF3.6+ */
      background: -webkit-gradient(linear, left top, left
bottom, color-stop(0%,#ff3019), color-stop(20%,#cf0404), color-
stop(100%,#ff3019)); /* Chrome,Safari4+ */
      background: -webkit-linear-gradient(top, #ff3019 0%,#cf0404
20%,#ff3019 100%); /* Chrome10+,Safari5.1+ */
      background: -o-linear-gradient(top, #ff3019 0%,#cf0404
20%,#ff3019 100%); /* Opera11.10+ */
      background: -ms-linear-gradient(top, #ff3019 0%,#cf0404
20%,#ff3019 100%); /* IE10+ */
      filter: progid:DXImageTransform.Microsoft.gradient(
startColorstr='#ff3019', endColorstr='#ff3019',GradientType=0 ); /*
IE6-9 */
      background: linear-gradient(top, #ff3019 0%,#cf0404 20%,#ff3019
100%); /* W3C */
    }
    h1 {
      padding:0.5em 0.2em;
      margin:0;
      font-size: 18px;
      color:white;
    }
```

```
    h2 {
      text-shadow: 0 1px #FFFFFF;
      background: #eeeeee; /* Old browsers */
      background: -moz-linear-gradient(top, #eeeeee 0%, #cccccc 100%);
/* FF3.6+ */
      background: -webkit-gradient(linear, left top, left bottom,
color-stop(0%,#eeeeee), color-stop(100%,#cccccc)); /* Chrome,Safari4+
*/
      background: -webkit-linear-gradient(top, #eeeeee 0%,#cccccc
100%); /* Chrome10+,Safari5.1+ */
      background: -o-linear-gradient(top, #eeeeee 0%,#cccccc 100%); /*
Opera11.10+ */
      background: -ms-linear-gradient(top, #eeeeee 0%,#cccccc 100%);
/* IE10+ */
      filter: progid:DXImageTransform.Microsoft.gradient(
startColorstr='#eeeeee', endColorstr='#cccccc',GradientType=0 ); /*
IE6-9 */
      background: linear-gradient(top, #eeeeee 0%,#cccccc 100%); /*
W3C */
      padding:0.5em 0.2em;
      margin:0;
      font-size: 16px;
      color:#000;
    }
    nav ul {
      border-top:1px solid #fff;
      list-style-type: none;
      padding:0;
      margin:0;
    }
    nav li {
      padding:0.5em 0.2em;
      margin:0;
      background:#AFAFAF;
      border-bottom:1px solid #fff;
    }
    nav li a {
      height:20px;
      display:block;
      text-decoration:none;
      color:white;
    }
    </style>
```

然后在移动设备浏览器上打开该文件，新的页面效果如图 4-2 所示。

图 4-2

4.3.3 回顾

在这个例子中，我们使用了 CSS3 的线性渐变特效来渲染 header 标签。以前，开发人员需要使用 Photoshop 或者 Illustrator[①]才能创建出本例子中展示的线性渐变效果，现在，可以完全通过 CSS 技术创建这样的效果了：

```
background: #eeeeee; /* Old browsers */
      background: -moz-linear-gradient(top, #eeeeee 0%, #cccccc 100%);
/* FF3.6+ */
      background: -webkit-gradient(linear, left top, left bottom,
color-stop(0%,#eeeeee), color-stop(100%,#cccccc)); /* Chrome,Safari4+
*/
      background: -webkit-linear-gradient(top, #eeeeee 0%,#cccccc
100%); /* Chrome10+,Safari5.1+ */
      background: -o-linear-gradient(top, #eeeeee 0%,#cccccc 100%); /*
Opera11.10+ */
      background: -ms-linear-gradient(top, #eeeeee 0%,#cccccc 100%);
/* IE10+ */
```

① Photoshop 和 Illustrator 都是 Adobe 公司开发的图形图像处理软件。——译者注

```
        filter: progid:DXImageTransform.Microsoft.gradient(
startColorstr='#eeeeee', endColorstr='#cccccc',GradientType=0 ); /*
IE6-9 */
        background: linear-gradient(top, #eeeeee 0%,#cccccc 100%); /*
W3C */
```

通过查看前面的每一条 CSS 样式规则，会发现每一个浏览器都有自己独有的线性渐变样式规则实现，例子中有 6 种不同的渐变样式实现就是为了可以兼容更多的浏览器，这时，你可能会想："额的神啊，我需要花这么多时间来考虑兼容多个浏览器吗？"伙计，不用担心，有工具可以帮忙处理这个事情，那就是 **Ultimate CSS Gradient Generator**，如图 4-3 所示。该工具是由 ColorZilla 公司提供的，一个强大的，类似于 PhotoShop 的 CSS 线性渐变效果编辑器。它可以帮助你轻松地创建跨越浏览器的渐变效果样式规则：`http://www.colorzilla.com/gradient-editor/`。

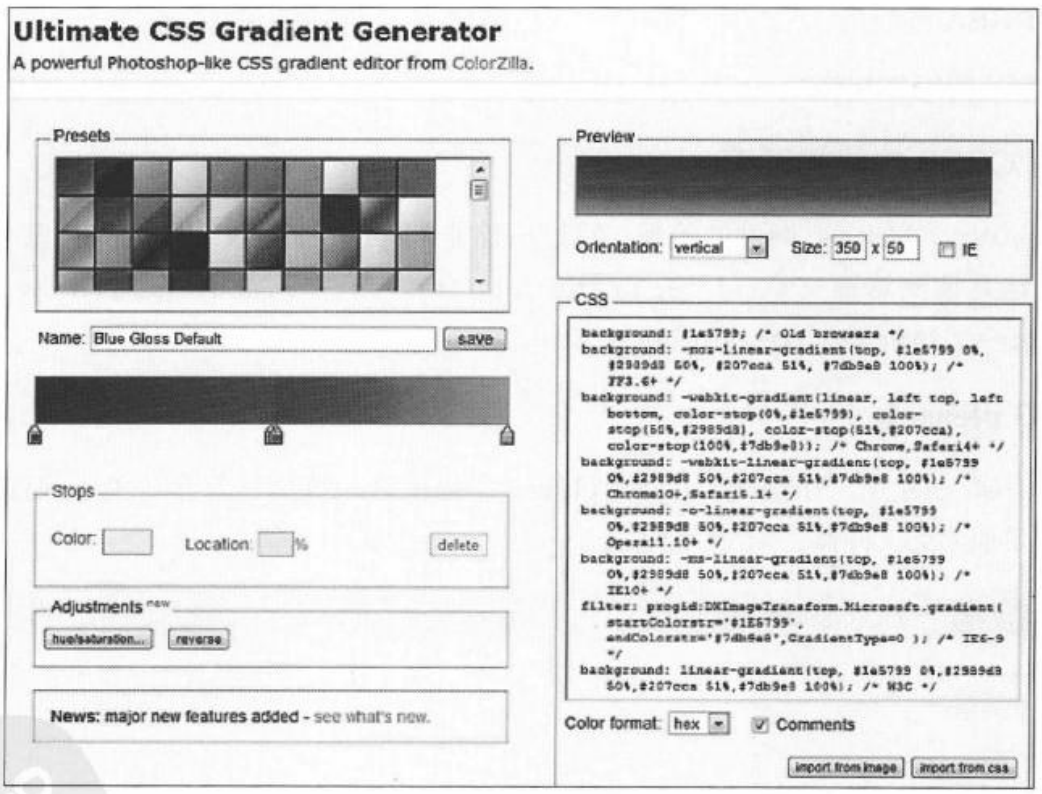


图 4-3

4.3.4 延伸

如果你还想把 IE9 或者其他的 IE 版本也纳入考虑范围的话，CSS3 PIE（`http://css3pie.com/`）可以帮你。

下载 PIE.htc 文件，通过如下的方式把它包含进你的 CSS 样式规则中：

```
-pie-background: linear-gradient(top, #eeeeee 0%,#cccccc 100%);
/*PIE*/
behavior: url(PIE.htc);
```

PIE 支持的特性有：

- 边框圆角
- 阴影效果
- 边框图片
- CSS3 背景
- 线性渐变效果
- RGBA 颜色值
- PIE 的定制属性

理解 CSS3 的线性渐变

Jeffrey Way，Nettuts[①]网站的编辑，写过一篇非常棒的，关于 CSS3 的线性渐变的文章，你可以在这儿读到这篇文章：http://net.tutsplus.com/tutorials/html-css-techniques/quick-tip- understanding-css3-gradients/。

CSS3 please!

Paul Irish 创建了一个名为“CSS3 please!”的项目，该项目包含了关于线性渐变的最新语法和其他 CSS3 特性：http://css3please.com/。

4.3.5 参考

- 1.5 节

4.4 使用响应式设计

适用设备：基于浏览器的所有设备

响应式设计是目前移动开发领域炙手可热的话题，它强调了一个很重要的概念，就是

[①] Nettuts 是一个知名的，为 Web 开发人员和设计人员提供教程和技术文章的网站。——译者注

浏览器应该根据屏幕的大小选择不同的渲染方式。使用响应式设计的移动项目的页面设计可以很容易地移植到桌面浏览器。

为什么我们要使用响应式设计呢？

当我们在针对桌面浏览器的 Web 页面使用了固定的布局时，因为桌面屏幕一般较大，通常会在页面左右两边留下一些空白部分。对于移动浏览器来说，如果原封不动地照搬这样的设计，那么同样会在左右两边留下一些空白。这对于可视区域本来就很受限的移动设备来说太浪费了。在移动设备上，每一个像素都十分宝贵，因此使用好屏幕上的每一个像素对于移动互联网开发来说是十分重要的。响应式设计就是用来解决这一类问题的。

Media Queries 是怎样帮助我们实现响应式设计的？

Media Queries[①]技术主要的能力就是在不改变内容的前提下，根据不同的屏幕大小使用不同的样式。因此，对于同一个 HTML 标签，可能有 2 个不同的样式规则，具体使用哪个来渲染取决于渲染该页面的浏览器的视口大小。

4.4.1 准备

在本小节的例子中，我们将使用一个名为“respond.js”的 HTML5 polyfill 脚本，它是由 jQuery Mobile 项目团队的 Scott Jehl 创建的，位于本章源代码的 ch04_code/js 目录下。

4.4.2 实践

首先，创建一个名为“ch04r03.html”的 HTML 文件，并写入下面的代码：

```
<!doctype html>
<html>
  <head>
    <title>first.fm</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet" href="css/style.css?v=1">
    <script>Modernizr.mq('(min-width:0)') || document.write("<script
src='js/respond.min.js'>\x3C/script>")</script>
  </head>
```

① Media Queries 是 CSS3 中基于 CSS2 media type 的一项扩展，它可以帮助我们获得浏览器的宽和高，设备的宽和高，设备是横向还是竖向以及设备的分辨率等信息。——译者注

```
    <body>
      <header>
        <h1>first.fm</h1>
      </header>
      <div id="main">
        <h2>Pages</h2>
        <nav>
          <ul class="list clearfix">
            <li class="list"><a href="music.html">Music</a></li>
            <li class="list"><a href="radio.html">Radio</a></li>
            <li class="list"><a href="events.html">Events</a></li>
            <li class="list"><a href="charts.html">Charts</a></li>
            <li class="list"><a href="community.html">Community</a></li>
            <li class="list"><a href="help.html">Help</a></li>
            <li class="list"><a href="about.html">About</a></li>
          </ul>
        </nav>
      </div>
      <footer>
        <small>&copy; 2011 first.fm</small>
      </footer>
    </body>
  </html>
```

如果你在移动设备的浏览器上访问该页面，得到的结果和上一小节的效果一样，但是如果你在桌面浏览器上访问该页面，你将会得到如图 4-4 所示的结果。

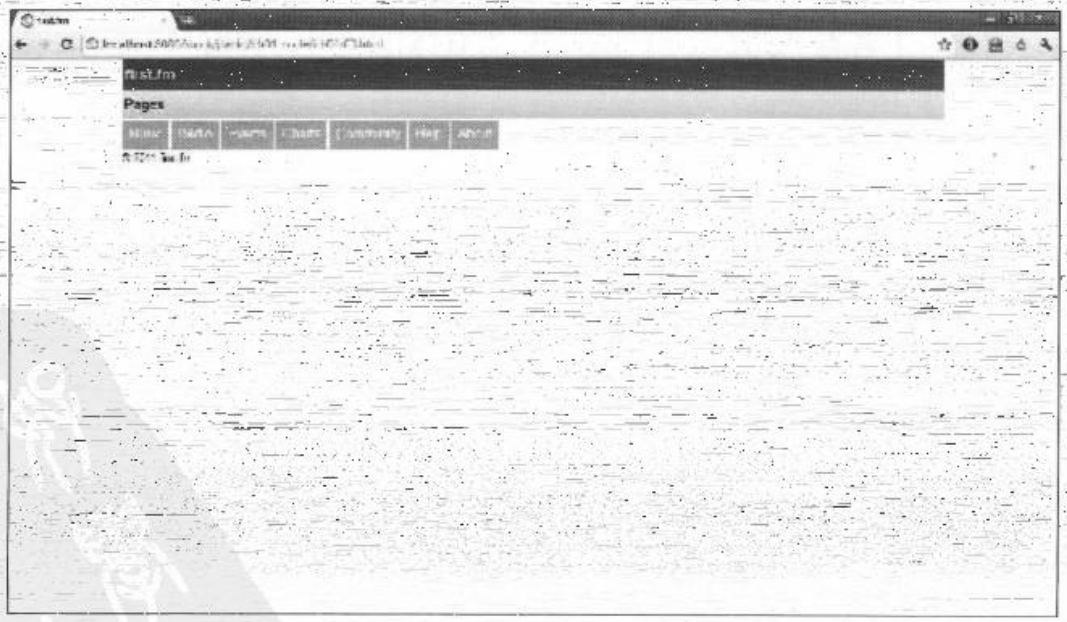


图 4-4

4.4.3 回顾

在 HTML 文件的头部，我们使用了 **Modernizr**[①]来检测当前浏览器是否支持 Media Queries，如果不支持的话，我们将加载 respond.min.js 文件，以支持 Media Queries：

```
<script>Modernizr.mq('(min-width:0)') || document.write("<script
src='js/respond.min.js'>\x3C/script>")</script>
```

在作者写这本书的时候，“respond.js”中的每一个规则都要在末尾加上一个“/*/mediaquery*/”的注释才能工作，这个在“respond.js”未来的版本中可能会改进：

```
@media only screen and (min-width: 800px) {
}/*/mediaquery*/
```

4.4.4 延伸

在 Mobile Boilerplate 项目网站上，有关于 Media Queries 的进一步解析，你可以从这儿了解更多：http://html5boilerplate.com/mobile/。

4.5 优化 polyfills 脚本的加载速度

适用设备：基于浏览器的所有设备

对于任何浏览器来说，加载脚本都是一项很重要且消耗性能的工作，对于移动设备来说更是如此，因为其带宽非常有限。Modernizr 为这个问题提供了一个动态加载的解决方案。

4.5.1 准备

首先创建一个名为“ch03r04.html”的 HTML 文件。

4.5.2 实践

在刚才创建的 HTML 文件中写入如下代码，并在移动设备上访问该文件：

[①] Modernizr 是一个用于检测浏览器功能支持状况的 JavaScript 库，它可以检测 40 多项关于 HTML5 的功能，以及 18 项 CSS3 功能；Media Queries 是 CSS3 中基于 CSS2 media type 的一项扩展，它可以帮助我们获得浏览器的宽和高，设备的宽和高，设备是横向还是竖向以及设备的分辨率等信息。——译者注

```
<!doctype html>
<html>
  <head>
    <title>first.fm</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <script src="js/modernizr.custom.54685.js"></script>
    <style>
    </style>
  </head>
  <body>
    <header>
      <h1>Your Location</h1>
    </header>
    <div id="main">
      Your Geo Location is: <span id="geo"></span>
    </div>

    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.
js"></script>
    <script>
      yepnope({
        test : Modernizr.geolocation,
        nope : ['js/geolocation.js'],
        complete: function () { navigator.geolocation.getCurr
entPosition(function(position) {
          document.getElementById('geo').innerHTML = position.
coords.latitude+", "+position.coords.longitude;
        });
      }
      });
    </script>
  </body>
</html>
```

4.5.3 回顾

在作者写作本书的时候，Modernizr 2.0 还处于 Beta 1 版本，如图 4-5 所示。在这个 Beta 版本中有两个非常吸引人的新特性，其中一个是你可以定制你想要检测的特性，另外一个就是你可以使用 Yepnope.js（也就是 Alex Sexton 和 Ralph Holzmann 写的 Modernizr.load）了，该脚本提供了一个动态的 JavaScript 加载器，你可以在本小节的扩展部分进一步了解该脚本。

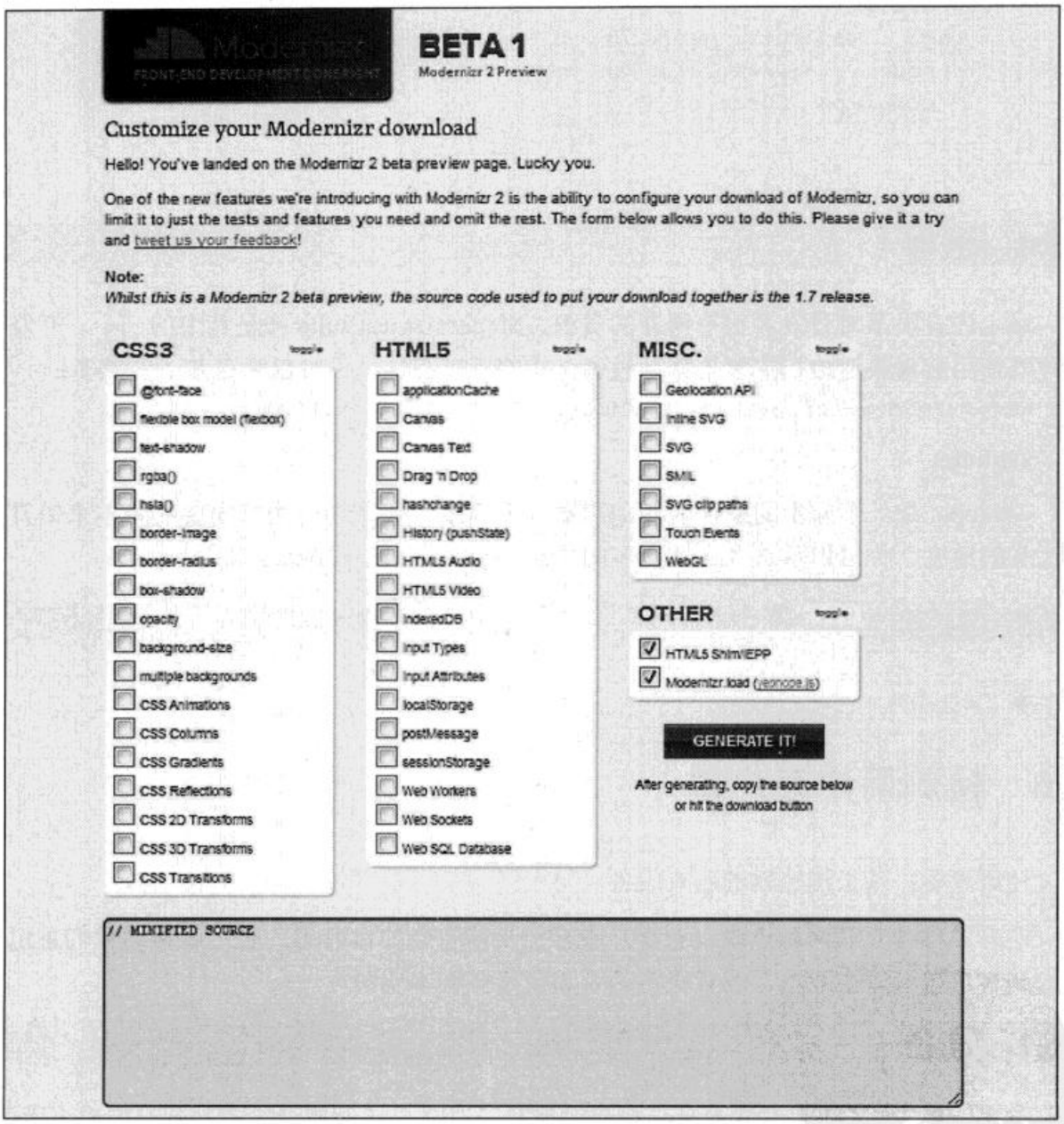


图 4-5

我们可以先通过 Modernizr 来检测某个特性是否已经存在于当前的页面：

```
test : Modernizr.geolocation
```

如果该特性不存在，那我们就是用 yepnope 的加载器加载该特性的 JavaScript 脚本 `geolocation.js`。加载完成之后，我们就可以使用该脚本提供的特性了。

```
yepnope({
  test : Modernizr.geolocation,
    nope : ['js/geolocation.js'],
    complete: function () {
    ...
});
```

4.5.4 延伸

网上还有很多对开发者很有帮助的资源，Modernizr test suite 就是其中的一个，开发人员可以使用它来快速地扫描某个移动设备支持的所有特性。你可以从这儿了解更多：

http://modernizr.github.com/Modernizr/test/index.html

yepnope

yepnope 是一个异步的条件资源加载器。它通过只加载当前用户需要的脚本来提升页面加载的速度。你可以从这儿了解更多的信息：http://yepnopejs.com/。

4.5.5 参考

- 4.3 节

4.6 检测用户客户端

使用设备：基于浏览器的所有设备

在开发移动互联网站点时，有客户端检测功能将会非常有用。它可以帮助我们重定向请求到客户端对应的版本，或者决定是否加载某些脚本或图片。

4.6.1 准备

首先，我们来了解下开发人员是如何实现基于对客户端的侦测重定向到对应的页面的。关于这个问题，目前业界已有很多解决方案，比方说在服务器端通过配置文件重定向，或者在前端通过 JavaScript 重定向。

4.6.2 实践

首先，请从http://detectmobilebrowser.com/网站下载实现重定向功能的JavaScript脚本，这个脚本有很多个版本，本小节中，我们将使用Apache版本的配置，.htaccess。

4.6.3 回顾

打开刚才下载的文件，你可以看到如下的代码：

```
RewriteEngine On
RewriteBase /

RewriteCond %{HTTP_USER_AGENT} android|avantgo|blackberry|blazer|comp
al|
....
|up\.(browser|link)|vodafone|wap|windows\ (ce|phone)|xda|xiino [NC,OR]
RewriteCond %{HTTP_USER_AGENT} ^(1207|6310|6590|
....
|your|zeto|zte\-) [NC]
RewriteRule ^$ http://example.com/mobile [R,L]
```

想要实现把对 PC 浏览器的网址请求重定向到移动版本的网址的功能，你只需要把脚本中的 `http://example.com/mobile` 改为你的网站地址即可。

4.6.4 延伸

客户端检测功能除了用来做重定向功能外，你可以用这个特性来决定是否在客户端加载某些脚本或文件。

在构建 Mobile Boilerplate 项目时，作者使用了 JavaScript 版本的客户端检测脚本来检测客户端环境，从而决定是否渲染内嵌内容：

```
if(!jQuery.browser.mobile) {
...
}
```

使用这个脚本之后，该页面在桌面浏览器上渲染显示的效果如图 4-6 所示。

图 4-6

在移动设备浏览器上，内嵌内容将不显示，具体的显示效果如图 4-7 所示。

图 4-7

移动设备浏览器侦测方法

在 Mobile tuts 网站上有一篇文章详述了很多种移动设备浏览器的侦测方法，请移步：http://mobile.tutsplus.com/tutorials/mobile-web-apps/mobile- browser-detection/。

4.7 使用书签气泡为应用添加桌面快捷方式

适用设备：iOS 设备

在前面的章节中，我们已经介绍过在不同设备上为自己的 Web 应用添加桌面快捷方式的方法。尽管这的确是一个非常酷的特性，可以让 Web 应用看起来更像原生应用，但是这儿有个问题：Web 应用上并没有提示，告诉用户可以为应用添加桌面快捷方式，因此很多用户根本就不知道有这样的一个特性。为了解决这个问题，一些移动开发框架通过使用 CSS 和 JavaScript 提供了一个添加桌面快捷方式的接口。该脚本会在用户访问该应用页面的时候弹出一个小气泡，提醒用户可以为当前应用创建一个桌面快捷方式。

4.7.1 准备

如我们前面所说，很多的开发框架都提供了这个功能，本小节中将选择一个独立的库以便于讲解。Google 发布过一个名为“Mobile Bookmark Bubble”的开源 JavaScript 库来专门提供这个功能，接下来，我们先去下载这个 JavaScript 库：http://code.google.com/p/mobile-bookmark-bubble/。

4.7.2 实践

刚才下载的 JavaScript 库文件中除了 bookmark_bubble.js，还有个文件 sample.js，我们

只需要在自己的页面中引入这两个文件，再访问你的页面的时候，你就会看到页面的效果如图 4-8 所示。

图 4-8

4.7.3 回顾

该 JavaScript 库通过检测 HTML5 的本地记录来确认是否已经弹出过添加快捷方式的提示小气泡，从而可以避免反复的弹出提示，影响用户体验。目前该库的实现只支持移动版的 Safari，因此只有 iPhone 和 iPad 用户才能享受到这个功能。

4.7.4 参考

- 2.7 节

4.8 构建可自动伸缩的文本输入框

适用设备：基于浏览器的所有设备

在移动设备上，像短信等原生应用的输入框都是可以随着用户的输入而自动伸缩的。在 Web 应用中，当你创建完一个文本输入框之后，该文本输入框的大小就固定了。当用户输入的文本超过了文本框的容量的时候，部分文本就不会显示在界面上，导致文本的可读性变差，在本小节中，我们为这个问题提供一个解决方案，看看如何为 Web 应用创建一个可以自动伸缩的文本输入框。

4.8.1 准备

首先，创建一个名为“ch04r05.html”的 HTML 文件，在本小节中，我们还会用到 Mobile Boilerplate 项目的 helper.js 文件，你可以从这儿下载这个文件：https://github.com/h5bp/mobile-boilerplate。

4.8.2 实践

把如下的代码写入刚创建的 HTML 文件：

```
<!doctype html>
<html>
  <head>
    <title>first.fm</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <style>
      #contact {width:220px; height:40px;}
    </style>
  </head>
  <body>
    <header>
      <h1>Contact Form</h1>
    </header>
    <div id="main">
      <p>Type the message to see it autogrow</p>
      <textarea id="contact">
      </textarea>
```

```
    </div>

    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.
  js"></script>
    <script src="//github.com/shichuan/mobile-html5-boilerplate/raw/
  master/js/mylibs/helper.js"></script>
    <script>
      var contact = document.getElementById("contact");
      MBP.autogrow(contact);
    </script>
  </body>
</html>
```

在移动设备浏览器上浏览该文件，效果如图 4-9 所示。

图 4-9

4.8.3 回顾

在这个例子中，我们在代码中添加了一个按键事件监听器，该事件监听器将会通过监测用户的输入来决定是否需要增加文本输入框的高度。我们通过监测文本内容的高度，如果文本内容的高度发生了变化，该脚本会改变文本输入框的 CSS 样式，增加文本输入框的高度。

4.8.4 延伸

文本输入框自动伸缩的概念源自 Google 的代码博客，你可以在如下的网址读到这篇博客： http://googlecode.blogspot.com/2009/07/gmail-for-mobile-html5-

series.html。

4.8.5 参考

- 4.9 节

4.9 加速按钮反馈

适用设备：iOS、Android 设备

在移动设备浏览器上，Web 应用中的按钮的反馈总会比原生应用的按钮反馈慢一点。其实，在移动设备的浏览器上有一个叫做“touchstart”的事件，通过监测该事件来代替按钮点击事件，会让应用更快地得到按钮的反馈。

4.9.1 准备

在本小节的例子中，我们会用到 Mobile Boilerplate 项目中的一个函数来实现加速按钮反馈的功能，首先，创建一个名为“ch04r06.html”的 HTML 文件。

4.9.2 实践

把下面的代码写入刚创建的 HTML 文件中：

```
<!doctype html>
<html>
  <head>
    <title>first.fm</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <style>
      #contact {
        width:220px; height:40px;
      }
    </style>
  </head>
  <body>
    <header>
      <h1>Contact Form</h1>
    </header>
```

```
    <div id="main">
      <textarea id="contact"></textarea><br />
      <button id="btn">INSTANT button!!!</button><br />
      <span id="result"></span>
    </div>
    <footer>
      <small>&copy; 2011 first.fm</small>
    </footer>
    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.5.1/jquery.
  js"></script>
    <script src="//github.com/shichuan/mobile-html5-boilerplate/raw/
  master/js/mylibs/helper.js"></script>
    <script>
      var btn = document.getElementById("btn");
      MBP.fastButton(btn,showForm);
      function showForm() {
        $("#result").html("Thank you for submitting, we will get back
  to you shortly!");
      }
    </script>
  </body>
</html>
```

4.9.3 回顾

下面是关于加速按钮反馈的实现函数的一个摘要，我们将一步一步地分析这个函数是如何工作的。

首先，在文件头部，我们定义了一个入口函数，该函数的用途是为页面添加一个事件监听器，该事件监听器将会监测页面上发生的所有 touchstart 和 click 事件：

```
MBP.fastButton = function (element, handler) {
    this.element = element;
    this.handler = handler;
    if (element.addEventListener) {
      element.addEventListener('touchstart', this, false);
      element.addEventListener('click', this, false);
    }
};

MBP.fastButton.prototype.handleEvent = function(event) {
    switch (event.type) {
        case 'touchstart': this.onTouchStart(event); break;
        case 'touchmove': this.onTouchMove(event); break;
```

```
            case 'touchend': this.onClick(event); break;
            case 'click': this.onClick(event); break;
        }
    };
```

onTouchStart 方法用来监听 touchmove 和 touchend 事件，StopPropagation 函数用来防止事件在监听器中传递，阻止冒泡，即只在当前方法体内寻找当前事件的监听器：

```
MBP.fastButton.prototype.onTouchStart = function(event) {
    event.stopPropagation();
    this.element.addEventListener('touchend', this, false);
    document.body.addEventListener('touchmove', this, false);
    this.startX = event.touches[0].clientX;
    this.startY = event.touches[0].clientY;
    this.element.style.backgroundColor = "rgba(0,0,0,.7)";
};
```

touchmove 方法是用来监测浏览器上是否有拖拽事件的发生，如果用户拖拽超过 10 个像素，我们将会重置页面：

```
MBP.fastButton.prototype.onTouchMove = function(event) {
    if(Math.abs(event.touches[0].clientX - this.startX) > 10 || Math.
abs(event.touches[0].clientY - this.startY) > 10) {
        this.reset();
    }
};
```

下面这段代码用来防止幽灵点击，从而可以响应真正的点击事件：

```
MBP.fastButton.prototype.onClick = function(event) {
    event.stopPropagation();
    this.reset();
    this.handler(event);
    if(event.type == 'touchend') {
        MBP.preventGhostClick(this.startX, this.startY);
    }
    this.element.style.backgroundColor = "";
};

MBP.fastButton.prototype.reset = function() {
    this.element.removeEventListener('touchend', this, false);
    document.body.removeEventListener('touchmove', this, false);
    this.element.style.backgroundColor = "";
};
```

4.9.4 延伸

你可以在 Google 的博客读到更多关于加速按钮反馈的文章，这些文章详细解释了关于加速按钮反馈的理念和背景。请移步：http://code.google.com/mobile/articles/fast_buttons.html。

4.9.5 参考

- 4.8 节

4.10 隐藏浏览器的地址栏

适用设备：iOS、Android 设备

移动设备浏览器的地址栏通常会占很大的界面空间，这对于每个像素就像北京二环内的地皮一样宝贵的移动设备屏幕来说，太浪费了。很多开发人员会在在页面加载之后把地址栏隐藏起来，以增加移动浏览器的可视域。

4.10.1 准备

首先，创建一个名为“ch04r07”的 HTML 文件

4.10.2 实践

在刚才创建的 HTML 文件中写入如下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <style>
      html,body,header,footer{
        padding:0;
        margin:0;
      }
```

```
            header{
              height:40px;
              background:#BFB840;
              display:block;
            }
            #main{
              height:350px;
              background:#F2CB67;
            }
            footer{
              height:40px;
              background:#DB5E31;
              display:block;
            }
          </style>
        </head>
        <body>
          <header>
            header
          </header>
          <div id="main">
            main
          </div>
          <footer>
            footer
          </footer>
          <script src="//ajax.googleapis.com/ajax/libs/jquery/1.5.1/jquery.
      js"></script>
          <script src="//github.com/shichuan/mobile-html5-boilerplate/raw/
      master/js/mylibs/helper.js"></script>
          <script>
            //MBP.hideUrlBar();
          </script>
        </body>
      </html>
```

在移动浏览器中访问该文件，得到效果如图 4-10 所示。

接下来，取消下面这行代码的注释标记，让其生效：

```
MBP.hideUrlBar();
```

在设备中再次访问该界面，你会发现，URL 地址栏被隐藏了，同时在屏幕上能够显示出最下面的 footer 一栏了，如图 4-11 所示。

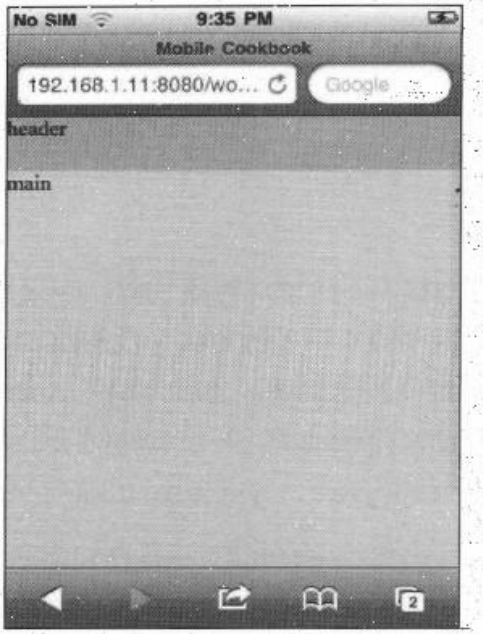


图 4-10

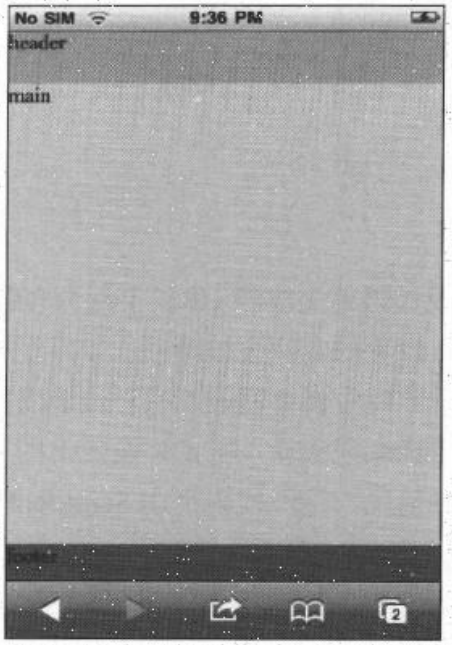


图 4-11

4.10.3 回顾

下面就来分析下“MBP.hideUrlBar()”是如何实现的，其在 Mobile Boilerplate 项目中的源代码如下：

```
MBP.hideUrlBar = function () {
  var win = window,
    doc = win.document;

  // If there's a hash, or addEventListener is undefined, stop here
  if( !location.hash || !win.addEventListener ){

    //scroll to 1
    window.scrollTo( 0, 1 );
    var scrollTop = 1,

    //reset to 0 on bodyready, if needed
    bodycheck = setInterval(function(){
      if( doc.body ){
        clearInterval( bodycheck );
        scrollTop = "scrollTop" in doc.body ? doc.body.scrollTop : 1;
        win.scrollTo( 0, scrollTop === 1 ? 0 : 1 );
      }
    }, 15 );

    win.addEventListener( "load", function(){
```

```
        setTimeout(function(){
          //reset to hide addr bar at onload
          win.scrollTo( 0, scrollTop === 1 ? 0 : 1 );
        }, 0);
      }, false );
  }
};
```

这段代码首先检测 URL 中是否有锚点，如果有则我们不应该运行该代码，因为原本页面根据锚点设置的定位会被覆盖；如果没有锚点，我们会稍等片刻以确定页面没有滚动，这种自动滚动常常发生在页面刷新后，浏览器根据刷新前的定位自动滚动，之后分别以 Android 设备上的 1 像素纵向滚动，和 iOS 设备上的 0 像素纵向滚动来隐藏地址，本小节的例子中同时处理了这两种情况。这段代码是由 Scott Jehl 提供：https://gist. github.com/1183357。

同样收录在Mobile Boilerplate项目的源代码中:https:// github.com/h5bp/mobile-boilerplate/blob/master/js/mylibs/helper.js。

4.10.4 参考

- 4.8 节

4.11 构建移动互联网站点的站点地图

适用设备：基于浏览器的所有设备

相信很多开发人员都对 Google 的站点地图格式十分熟悉，Google 作为全球最大的搜索引擎，拥有非常庞大的用户群，因此让用户能够通过 Google 搜索到我们的移动互联网站点是非常重要的，Google 为了能给移动互联网站点做更多的搜索引擎优化，推出了移动互联网站点的网站地图格式，建议大家按照该格式来描述自己的 Web 应用的站点地图。

4.11.1 准备

首先，创建一个名为“sitemap.xml”的 XML 文档。

4.11.2 实践

在刚才创建的文件中，写入下面的代码，把其中的 URL 部分替换成你的应用地址：

```
<?xml version="1.0" encoding="UTF-8" ?>
 <urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9"
  xmlns:mobile="http://www.google.com/schemas/sitemap-mobile/1.0">
```

```
    <url>
        <loc>http://mobile.example.com/article100.html</loc>
        <mobile:mobile/>
    </url>
</urlset>
```

所有的网页 URL 地址都必须写在<loc>标签内，同时要注意一定要加上“<mobile:mobile/>”这行代码，否则的话，你的网站可能不能被 Google 的搜索爬虫爬到。

4.11.3 回顾

站点地图[①]都是有自己特殊的格式的，前面提到的 XML 文件就是用来告诉 Google 的搜索引擎该移动站点的各个网页的地址的。通常情况下，如果你的站点使用了 CMS（Content Management System：内容管理系统），那么，你可以使用它为你自动生成所有的需要放到<loc>标签中的 URL。

4.11.4 延伸

移动互联网的站点地图必须使用能在移动设备访问的 URL，仅能在桌面浏览器访问的 URL 是不能用的。

对于那些为移动设备提供了独有版本的网站，你可能想要把移动设备用户对网站的访问自动重定向到移动版站点，比方说把对“example.com”的访问重定向到“m.example.com”，这个可以使用 HTTP 的 301 重定向功能实现。

如果你的移动互联网站点要提供各种类型的内容服务，这个就不是 Google 考虑的范畴了。

Google 和构建移动设备友好的网站

在 Google 的 Webmaster 站点上，有一篇关于如何让你的站点对移动设备更加友好的博客，请移步：http://googlewebmastercentral.blogspot.com/2011/02/making-websites-mobile-friendly.html。

Google 和移动互联网站点索引

在 Google 的 Webmaster 网站上，还有一篇关于如何能让 Google 更好地收罗你的站点的博客，请移步：http://googlewebmastercentral.blogspot.com/2009/11/help-google-index-your-mobile-site.html。

[①] 站点地图：一指面向用户的网站导航，为用户介绍网站的结构、栏目和内容的说明文件，一指面向搜索引擎的包含了你的网站目录结构，以及想被搜索引擎收录的所有网页的 URL 地址的 XML 文件，本小节中，指的是第二个概念。——译者注

第5章
移动设备访问

本章内容：

- 获取位置信息
- 跨浏览器定位
- 根据定位显示地图
- 实时定位
- DeviceOrientation 事件
- 使用 foursquare 的定位

5.1 简介

在 HTML5 的所有功能中，与移动开发关联最大的就是设备访问（Device Access）了。

下面是 W3C HTML5 官方网站关于设备访问的解释（http://www.w3.org/html/logo/）：

从地理信息接口开始，网页应用可以拥有丰富的设备感知的功能和体验。一些基于设备访问的绝佳的创意已经开发出来了，从基于麦克风的音频输入访问，和基于摄像头的视频输入访问，到本地数据（如联系人和事件），甚至是重力感应都包含在内。

你可以在以下地址找到描述和 logo: http://www.w3.org/html/logo/#the-technology。

基于位置的社交网络，例如 foursquare，对于商业模式和人员流动的方式都有深远的影响。Groupon 的新的基于位置的功能，如果发布，将会从根本上改变用户行为习惯和零售

业的运营模式。Google Maps 使用实时的地理信息和 GPRS 来帮助人们及车辆导航。将来还会有更多激动人心的创意是基于设备访问技术的。

本章中，我们会学习地理信息接口、DeviceOrientation 接口、跨浏览器的地址问题以及如何将设备访问与基于位置的常用服务结合起来。

5.2 获取位置信息

适用设备：Android、iOS、webOS、Opera、Firefox

使用地理信息接口，可以获取经度、纬度和当前的位置精度：

- 经纬度：它是地理坐标，单位为十进制度。
- 精度：表示经纬坐标的精度级别，单位为米。

5.2.1 准备

让我们创建一个 HTML 文件，然后获取带精度的经纬度。首先创建 HTML 文件命名为 ch05r01.html。

5.2.2 实践

在 HTML 文件中输入以下代码：

```
<!doctype html>
<html>
  <head>
   <title>Mobile Cookbook</title>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  </head>
  <body>

   <div id="main">
         <div id="someElm">
         </div>
   </div>
   <script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
   <script>
   function getLocation() {
```

```
        navigator.geolocation.getCurrentPosition(showInfo);
      }
      function showInfo(position) {
        var latitude = position.coords.latitude;
        var longitude = position.coords.longitude;
        var accuracy = position.coords.accuracy;
        $('#someElm').html('latitude: '+latitude+'<br />longitude:
'+longitude+'<br />accuracy: '+accuracy);
      }
      getLocation();
       </script>
       </body>
</html>
```

当第一次运行的时候，会出现如图 5-1 所示的提示信息。

地理信息功能是需要用户主动开启的，没有浏览器会自动发送设备的物理位置给服务器。在程序发送设备的地理信息前，浏览器会反复请求许可，同时浏览器也可以记住你的选择以免每次都为同样的网站重复请求许可。

现在选择允许共享位置信息，你会看到地理数据如下显示在屏幕上，如图 5-2 所示。

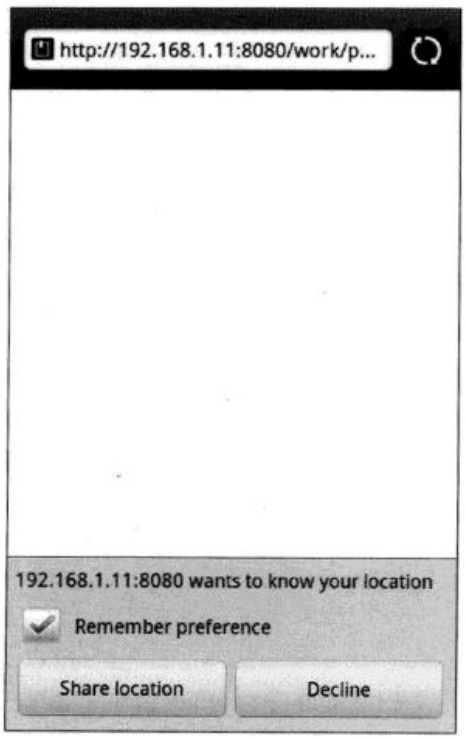


图 5-1

图 5-2

5.2.3 回顾

navigator 对于 JavaScript 程序员来说并不陌生，它通常用于获取浏览器信息：navigator.userAgent。

geolocation 是 navigator 的新属性：navigator.geolocation。

getCurrentPosition 是 navigator.geolocation 的方法，在本例中，我们将函数 showInfo 作为第一个参数传入并执行：

```
navigator.geolocation.getCurrentPosition(showInfo);
```

在 showInfo 函数中，我们返回了 position 变量的三个值，latitude、longitude 和 accuracy：

```
var latitude = position.coords.latitude;
var longitude = position.coords.longitude;
var accuracy = position.coords.accuracy;
```

5.2.4 延伸

是否地理信息接口仅能获取上述提及的属性？理论上是可以获取更多的信息，但是实际上，只有某些浏览器会返回额外的信息。

5.3 跨浏览器定位

适用浏览器：所有

地理信息接口不一定在所有的移动浏览器上都可用，甚至对于支持它的浏览器也一样，浏览器可能有与标准不同的接口。iOS 和 Android 适用标准的接口，已知有不同接口的浏览器是 Blackberry、Nokia 和 Palm，幸运的是我们有 geo-location-javascript——以移动设备为主要对象的地理接口预处理工具，它使没有标准接口的 Blackberry 和 webOS 可以使用标准的接口。

5.3.1 准备

下载本章的资源，创建一个 js 文件夹，将 geo.js 放入。然后创建一个 HTML 文档，命名为 ch05r02.html。

5.3.2 实践

在 HTML 文档中输入以下代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script src="http://code.google.com/apis/gears/gears_init.js"
type="text/javascript" charset="utf-8"></script>
<script src="js/geo.js" type="text/javascript" charset="utf-8"></
script>
</head>
<body>

  <div id="main">
    <div id="someElm">
          </div>
  </div>
  <script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
  <script>
    if(geo_position_js.init()){
      geo_position_js.getCurrentPosition(success_callback,error_callba
ck,{enableHighAccuracy:true,options:5000});
    }
    else{
       $('#someElm').html("Functionality not available");
    }
    function success_callback(p)
    {
       $('#someElm').html('latitude: '+p.coords.latitude+'<br
/>longitude: '+p.coords.longitude+'<br />accuracy: '+p.coords.
accuracy);
    }
    function error_callback(p)
    {
       $('#someElm').html('error='+p.message);
    }
  </script>
</body>
</html>
```

在 Opera 中打开页面，可以看到如图 5-3 所示的结果。

图 5-3

5.3.3 回顾

在 HTML 文档开始，我们引入了 gears_init.js。如果浏览器默认不支持地理信息接口，但安装了 Gears，那么通过 Gear 接口可以获取到地理信息数据；如果浏览器支持地理信息接口，但接口标准不同，那么第二个脚本 geo.js 会使之标准化。

如果 geo_position_js.init()返回 true，那么我们就可以直接或间接的获取到地理信息数据，这样就可以进入下一步，使用 geo_position_js.getCurrentPosition 取代 navigator.geolocation.getCurrentPosition，如下：

```
geo_position_js.getCurrentPosition(showInfo,error_callback,{enableHigh
Accuracy:true,options:5000});
```

5.3.4 延伸

下面是另一个可以辅助获取地理信息的工具。

YQL Geo 库

YQL Geo 库提供了另一种获取地理信息的方式，通过 IP 地址。它是一个轻量级的库，已经在 Yahoo 服务中试用。它可以：

- 通过文本获取地理位置
- 通过经纬度获取位置信息
- 通过某个 URL 获取地理信息
- 通过 IP 地址获取位置

5.4 基于地理信息显示地图

适用浏览器：所有

Google Maps API V3 可以在移动设备上快速加载并且工作良好，尤其当我们关注于高端移动设备的开发，如：iPhone、Android 系统的手机。移动设备的屏幕尺寸小于一般的桌面浏览器，同时对于这些设备还有一些特别的操作方式，如 iPhone 的缩放手势。

5.4.1 准备

让我们创建一个显示于移动设备上的地图。首先创建一个 HTML 文档命名为 ch05r03.html。

5.4.2 实践

输入以下代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
<meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
<script type="text/javascript" src="http://maps.google.com/maps/api/
js?sensor=true"></script>
<script src="http://code.google.com/apis/gears/gears_init.js"></
script>
<script src="js/geo.js"></script>
<style>
html {
  height: auto;
}
body {
  height: auto;
  margin: 0;
  padding: 0;
}
#map_canvas {
  height: auto;
  position: absolute;
```

```
      bottom:0;
      left:0;
      right:0;
      top:0;
    }
    </style>
    </head>
    <body>
      <div id="map_canvas"></div>
      <script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
      <script>
        var initialLocation;
        var siberia = new google.maps.LatLng(60, 105);
        var newyork = new google.maps.LatLng(40.69847032728747,
    -73.9514422416687);
        var browserSupportFlag =  new Boolean();
        var map;
     var infowindow = new google.maps.InfoWindow();

     function initialize() {
       var myOptions = {
         zoom: 12,
         mapTypeId: google.maps.MapTypeId.ROADMAP
       };
       map = new google.maps.Map(document.getElementById("map_canvas"),
myOptions);

       if(geo_position_js.init()){
         browserSupportFlag = true;
         geo_position_js.getCurrentPosition(function(position) {
           initialLocation = new google.maps.LatLng(position.coords.
latitude,position.coords.longitude);
           contentString = "you are here";
           map.setCenter(initialLocation);
           infowindow.setContent(contentString);
           infowindow.setPosition(initialLocation);
           infowindow.open(map);
         });
       }
     }

     function detectBrowser() {
       var useragent = navigator.userAgent;
       var mapdiv = document.getElementById("map_canvas");
```

```
        if (useragent.indexOf('iPhone') != -1 || useragent.
indexOf('Android') != -1) {
          mapdiv.style.width = '100%';
          mapdiv.style.height = '100%';
        } else {
          mapdiv.style.width = '600px';
          mapdiv.style.height = '800px';
        }
      }
      detectBrowser();
      initialize();
    </script>
</body>
</html>
```

在移动浏览器中打开页面，显示如图 5-4 所示。

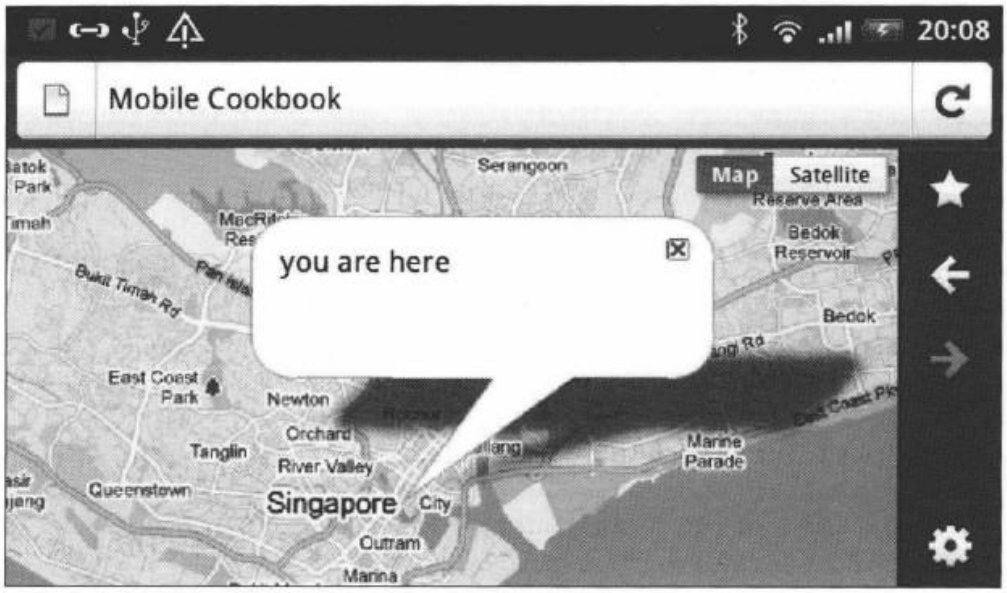


图 5-4

5.4.3 回顾

现在我们分解前面的代码，看看每一部分的作用：

1. iPhone 拥有缩放手势，而 Google Maps API V3 对这一事件也有特殊处理，所以你可以设置以下的 meta 标签来确保用户不能针对 iPhone 浏览器使用缩放手势，Android 版本 1.5（Cupcake）以上的设备同样支持这些变量：

```
<meta name="viewport" content="initial-scale=1.0, user-
scalable=no" />
```

2．将地图放入一个容器 中，并设置容器的宽、高属性为百分之百：

```
mapdiv.style.width = '100%';
mapdiv.style.height = '100%';
```

3．通过 navigator.userAgent 属性来判断手机设备为 iPhone 或 Android：

```
function detectBrowser() {
  var useragent = navigator.userAgent;
  var mapdiv = document.getElementById("map_canvas");

  if (useragent.indexOf('iPhone') != -1 || useragent.
indexOf('Android') != -1 ) {
    mapdiv.style.width = '100%';
    mapdiv.style.height = '100%';
  } else {
    mapdiv.style.width = '600px';
    mapdiv.style.height = '800px';
  }
}
```

4．设置传感器参数：当读取地图接口脚本时，必须设置 sensor=true，应用程序才会利用传感器定位用户的位置。

```
<script type="text/javascript" src="http://maps.google.com/maps/
api/js?sensor=true"></script>
```

使用 Google Maps API，需要声明是否使用传感器（例如 GPS 定位系统）来定位用户的位置，这对于移动设备来说尤其重要。引入地图接口脚本时，应用程序必须在<script>标签设置一个传感器参数，来声明应用程序是否使用传感器设备。

需要注意的是，如果目标设备不支持传感器，仍然需要设置这个参数的值为 false。

5．通过地图接口的 LatLng 方法来获取地理坐标：

```
initialLocation = new google.maps.LatLng(position.coords.
latitude,position.coords.longitude);
```

5.4.4 延伸

你可以从 Google Maps JavaScript API V3 的官方文档中了解更多信息：

```
http://code.google.com/apis/maps/documentation/javascript/
```

HTML5 地理信息教程

在 tuts 的移动栏目中有一篇非常优秀的关于移动地理信息的文章：《HTML5 Apps: Positioning with Geolocation》，你可以在这里阅读：

《HTML5 Apps: Positioning with Geolocation》

http://mobile.tutsplus.com/tutorials/mobile-web-apps/html5-geolocation/

5.5 实时显示地理位置

适用浏览器：所有

除 getCurrentPosition，地理信息接口还有另一个方法 watchPosition，它有两个重要功能：

1. 定义并获取一个监听操作。
2. 异步地进行监听操作。

5.5.1 准备

创建一个 HTML 文档命名为 ch05r04.html。

5.5.2 实践

在 HTML 文档中输入以下代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
<meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
<style>
html {
  height: auto;
}
body {
  height: auto;
  margin: 0;
  padding: 0;
```

```
}
#map_canvas {
  height: auto;
  position: absolute;
  bottom:0;
  left:0;
  right:0;
  top:0;
}
</style>
</head>
<body>
<div id="map_canvas"></div>
<script type="text/javascript" src="http://maps.google.com/maps/api/
js?sensor=true"></script>
<script src="http://code.jquery.com/jquery-1.5.2.min.js"></script>
<script>
var watchProcess = null;
var initialLocation;
var map;
var infowindow = new google.maps.InfoWindow();
var myOptions = {
  zoom: 12,
  mapTypeId: google.maps.MapTypeId.ROADMAP
};

map = new google.maps.Map(document.getElementById("map_canvas"),
myOptions);

navigator.geolocation.getCurrentPosition(function(position) {
  updatePos(position.coords.latitude,position.coords.
longitude,position.coords.accuracy);
});

initiate_watchlocation();

function initiate_watchlocation() {
  if (watchProcess == null) {
    watchProcess = navigator.geolocation.watchPosition(handle_
geolocation_query, handle_errors);
  }
}

function stop_watchlocation() {
  if (watchProcess != null)
```

```
    {
      navigator.geolocation.clearWatch(watchProcess);
      watchProcess = null;
    }
}

function handle_errors(error)
{
  switch(error.code)
  {
  case error.PERMISSION_DENIED: alert("user did not share geolocation
data");
    break;

  case error.POSITION_UNAVAILABLE: alert("could not detect current
position");
    break;

  case error.TIMEOUT: alert("retrieving position timedout");
    break;

  default: alert("unknown error");
    break;
  }
}

function handle_geolocation_query(position) {
  updatePos(position.coords.latitude,position.coords.
longitude,position.coords.accuracy);
}

function updatePos(lat,long,acc) {
  var text = "Latitude: "  + lat  + "<br/>" + "Longitude: " + long +
"<br/>" + "Accuracy: "  + acc  + "m<br/>";
  initialLocation = new google.maps.LatLng(lat,long);
  contentString = text;
  map.setCenter(initialLocation);
  infowindow.setContent(contentString);
  infowindow.setPosition(initialLocation);
  infowindow.open(map);
}
</script>
</body>
</html>
```

显示结果如图5-5所示。

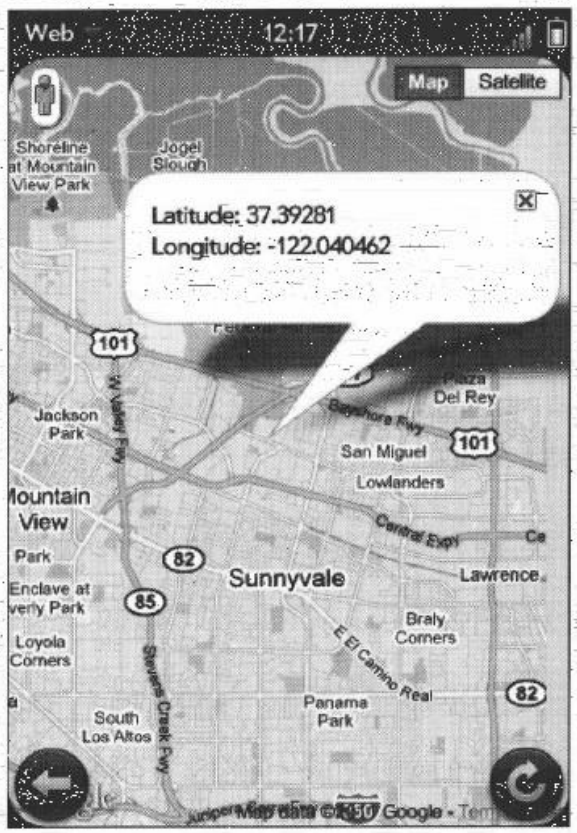


图5-5

5.5.3 回顾

下面的方法初始化了地理位置的监听事件：

```
function initiate_watchlocation() {
  if (watchProcess == null) {
    watchProcess = navigator.geolocation.watchPosition(handle_
geolocation_query, handle_errors);
  }
}
```

navigator.geolocation.watchPosition会根据执行结果返回成功或失败，在成功的回调中，可以解析经度和纬度：

```
navigator.geolocation.watchPosition(handle_geolocation_query,
handle_errors);
```

在监听位置信息的过程中，handle_geolocation_query用于获取当前位置并调

用位置信息的更新函数：

```
function handle_geolocation_query(position) {
  updatePos(position.coords.latitude,position.coords.
longitude,position.coords.accuracy);
}
```

5.6 使用 DeviceOrientation 事件

适用浏览器：iOS

DeviceOrientation 事件是设备访问的重要部分，它包含了设备移动事件和横竖屏切换事件，但只有 iOS 支持这些事件。

5.6.1 准备

创建 HTML 文档命名为 ch05r05.html。

5.6.2 实践

在文档中输入以下代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
<meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
<script type="text/javascript" src="http://maps.google.com/maps/api/
js?sensor=true"></script>
<style>
#no {
    display: none;
}
#ball {
    width: 20px;
    height: 20px;
    border-radius: 10px;

    background-color: red;
    position:absolute;
    top: 0px;
    left: 0px;
}
```

```
</style>
</head>
<body>
  <div id="content">
    <h1>Move the Ball</h1>
    <div id="yes">
        <p>Move your device to move the ball.</p>
        <div id="ball"></div>
    </div>
    <div id="no">
    Your browser does not support Device Orientation and Motion API.
Try this sample with iPhone, iPod or iPad with iOS 4.2+.</div>
  </div>
  <script>
  // Position Variables
  var x = 0;
  var y = 0;

  // Speed - Velocity
  var vx = 0;
  var vy = 0;

  // Acceleration
  var ax = 0;
  var ay = 0;

  var delay = 10;
  var vMultiplier = 0.01;

  if (window.DeviceMotionEvent==undefined) {
    document.getElementById("no").style.display="block";
    document.getElementById("yes").style.display="none";

  } else {
    window.ondevicemotion = function(event) {
      ax = event.accelerationIncludingGravity.x;
      ay = event.accelerationIncludingGravity.y;
    }

    setInterval(function() {
      vy = vy + -(ay);
      vx = vx + ax;

      var ball = document.getElementById("ball");
      y = parseInt(y + vy * vMultiplier);
      x = parseInt(x + vx * vMultiplier);

      if (x<0) { x = 0; vx = 0; }
      if (y<0) { y = 0; vy = 0; }
      if (x>document.documentElement.clientWidth-20) { x = document.
```

```
documentElement.clientWidth-20; vx = 0; }
      if (y>document.documentElement.clientHeight-20) { y = document.
documentElement.clientHeight-20; vy = 0; }

      ball.style.top = y + "px";
      ball.style.left = x + "px";
    }, delay);
  }
  </script>
</body>
</html>
```

5.6.3 回顾

这些代码来自 Maximiliano Firtman（http://www.mobilexweb.com/blog/safari-ios-accelerometer-websockets-html5）。本例中使用了 accelerationIncluding Gravity，它用于获取设备的加速度值，包括了用户的加速度和重力加速度，如图 5-6 所示。

x、y、z 三个值以 m/s^2 作为单位，分别表示了每个轴上的加速度：

```
window.ondevicemotion = function(event) {
  event.accelerationIncludingGravity.x
  event.accelerationIncludingGravity.y
  event.accelerationIncludingGravity.z
}
```

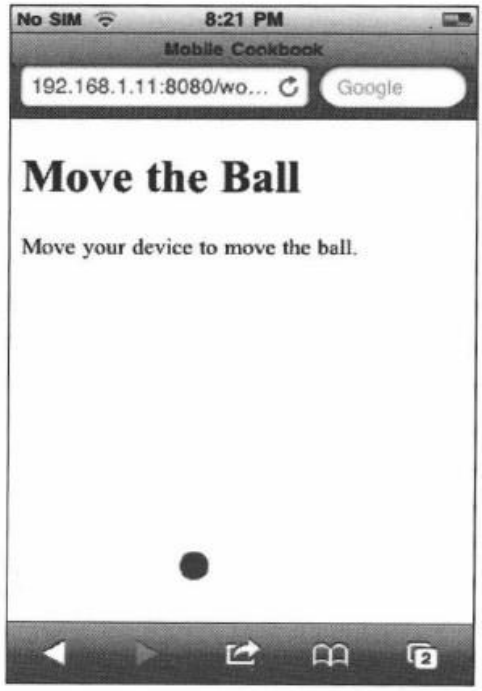


图 5-6

5.6.4 延伸

表 5-1 显示了当前 iOS 设备对于 DeviceOrientationEvent 和 DeviceMotionEvent 的支持：

表 5-1

属　性	说　明	返 回 值	类	支　持
acceleration	用户给予设备的加速度	x、y、z（单位：m/s^2）	DeviceMotion Event	iPhone4、iPod Touch 4G
acceleration IncludingGravity	设备的加速度，包括了用户的加速度和重力加速度	x、y、z（单位：m/s^2）	DeviceMotion Event	iPhone3、iPod Touch 3G
interval	从上一次设备移动，到现在的间距，以毫秒为单位	毫秒	DeviceMotion Event	iPhone3、iPod Touch 3G
rotationRate	设备的转动比率	alpha 、 betahe gamma（值介于 0 到 360 之间）	DeviceMotion Event	iPhone 4、iPod Touch 4G
alpha	设备在 *z* 轴转动的度数	值介于 0 到 360 之间	DeviceOrientation Event	iPhone 4、iPod Touch 4G
beta	设备在 *x* 轴转动的度数	值介于−180 到 180 之间	DeviceOrientation Event	iPhone 4、iPod Touch 4G
gamma	设备在 *y* 轴转动的度数	值介于−90 到 90 之间	DeviceOrientation Event	iPhone 4、iPod Touch 4G

DeviceOrientation 事件详述

http://dev.w3.org/geo/api/spec-source-orientation.html

Safari 网站中的官方指南

DeviceOrientation 事件详述：

https://developer.apple.com/library/safari/#documentation/SafariDOMAdditions/Reference/DeviceMotionEventClassRef/DeviceMotionEvent/DeviceMotionEvent.html

DeviceOrientationEvent 类参考文档：

https://developer.apple.com/library/safari/#documentation/SafariDOMAdditions/Reference/DeviceOrientationEventClassRef/DeviceOrientationEvent/DeviceOrientationEvent.html

5.7 使用 foursquare 的定位

目标浏览器：所有

近几年，基于地理位置的社交网站 foursquare 变得越来越流行，它影响了许多商业模式和用户习惯，用户使用移动网站、移动应用或短信在某地“签到”。

5.7.1 准备

第三方的程序员基于各种编程语言，发布了许多连接 foursquare 接口的库，Marelle 是其中之一，它基于 jQuery 和 coffeescript，但不用担心，实际上都是 JavaScript 语言。

5.7.2 实践

进入 Marelle 的 Github 页面（http://praized.github.com/marelle/）并下载最新版本，它包括了两个例子，一个关于登录，另一个关于签到。

下面是登录代码概略：

```
// Supply your foursquare client id
var FSQUARE_CLIENT_ID = 'FOURSQUARE_CLIENT_ID';
// on DOM ready...
$(function() {
  // setup with your key and a callback function which
  // receives the Marelle Object ( "M" in this example )
  $.Marelle( FSQUARE_CLIENT_ID ).done( function( M ){
    // grab an authentication promise
    var authpromise = M.authenticateVisitor();
    //  handle logged-in visitor
    var authsuccess = function(visitor){
      M.signoutButton( document.body );
      console.log(visitor)
      /*
      I think the single entry point is through the visitor
      */
      venuepromise = visitor.getVenues()
```

```
        // venuepromise.then etc..etc...
      };
      // handle non visitor
      var authfailure = function() {
        M.signinButton( document.body );
      };
      // wait for promise to resolve
      authpromise.then(authsuccess,authfailure)

    }).fail(function(){
      consoloe.log('Marelle could not be loaded.')
    });
  });
```

5.7.3 回顾

以上代码的工作方式：

1. 首先调用 Marelle 的初始化方法$.Marelle(clientID)，它会返回一个 promise 对象：

```
$.Marelle( FSQUARE_CLIENT_ID )
```

2. 然后通过$.Marelle.authenticateVisitor()获取一个认证：

```
authenticateVisitor():
$.Marelle( FSQUARE_CLIENT_ID ).done( function( M ){
    var authpromise = M.authenticateVisitor();
});
```

3. 基于认证结果，authpromise.then()会调用 authsuccess 或 authfailure：

```
authpromise.then(authsuccess,authfailure)
```

4. 如果认证成功，我们在给定的元素中添加“断开连接”按钮：

```
M.signoutButton( document.body );
```

5. 我们可以获取推荐的场馆，或者添加、搜索场馆：

```
venuepromise = visitor.getVenues()
```

6. 如果验证不成功，我们在给定的元素中添加“连接”按钮：

```
M.signinButton( document.body );
```

5.7.4 延伸

下面的地址是 foursquare 的接口列表：

```
https://developer.foursquare.com/docs/libraries.html
```

第 6 章
移动富媒体

本章内容概括：

- 移动设备上播放音频
- 移动设备上播放视频
- 使用离线缓存
- 使用网络存储（Web Storage）
- 使用 Web Workers
- 使用 Session 和 historyAPI 构建类 Flash 的导航

6.1 简介

开发者可以使用 HTML5 构建各种各样的适用于移动设备的富媒体应用。HTML5 为开发者提供了各种各样的方法来构建富媒体应用。只有你想不到的，没有它不能实现的。

在前面的章节中，我们介绍了 HTML5 的语法、CSS3 特性和接入设备归类。在本章中，我们将继续为大家介绍 HTML5 的其他三个部分。

- **多媒体**——越来越多的用户直接在线收听音频和观看视频了，我们将为大家介绍如何把这些多媒体元素添加到移动互联网应用中。
- **离线和存储**——对于移动设备来说，网络连接是非常不稳定的，因此离线使用对移动应用来说非常重要。通过把部分数据缓存在移动设备上，当用户重新访问某个网页的时候就不需要重新从服务端获取数据。

- **性能和集成**——通过使用 Web Workers[①]（仅支持 iOS 和 BlackBerry 设备），我们可以在移动设备浏览器上得到更好的性能。

6.2 移动设备上播放音频

适用设备：iOS、Android、BlackBerry、webOS，以及使用 Opera Mobile 或 Firefox Mobile 的设备

所谓多媒体，主要就是指的音频和视频。在移动设备上播放音频比较麻烦，因为移动设备的浏览器通常只支持一小部分的音频格式，包括 Ogg Vorbis[②]、MP3、WAV。即使是这么一小部分的音频格式，也不是每个浏览器都支持所有的格式。

6.2.1 准备

首先，创建一个名为“ch06r01.html”的 HTML 文件。

6.2.2 实践

在前面创建的 HTML 文件中写入如下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>

    <div id="main">
      <audio src="resources/snake_charmer.mp3" controls preload="auto"
autobuffer>
      </audio>
    </div>

  </body>
</html>
```

[①] Web Workers 是 HTML5 提供的一个 JavaScript 多线程解决方案。——译者注

[②] OGG Vorbis 是一种新的音频压缩格式，类似于 MP3 等现有的音乐格式。但有一点不同的是，它是完全免费、开放和没有专利限制的。OGG Vorbis 有一个很出众的特点，就是支持多声道，随着它的流行，以后用随身听来听 DTS 编码的多声道作品将不会是梦想。——译者注

在移动设备浏览器中渲染该页面，页面上将会展示一个如图 6-1 所示的音频播放器，点击上面的播放按钮，音频将会开始播放。

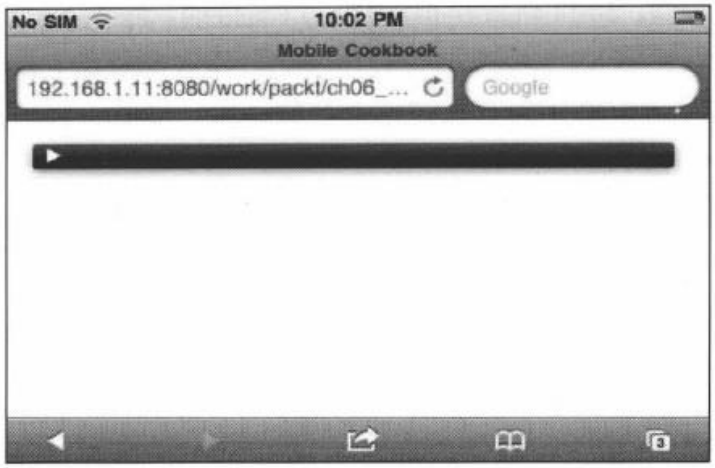


图 6-1

6.2.3 回顾

如示例代码所见，使用 HTML5 为网页添加音频功能相当简单，只需要把你想要播放的音频添加到<audio></audio>标签中就行了。该标签的 controls 属性是告诉浏览器，该音频播放元素需要显示一个控制元素，像“播放”、“暂停”等。autobuffer 属性是告诉浏览器需要缓冲该音频并流式播放。autobuffer 属性是一个布尔值，如果把该属性放置到 audio 标签中，audio 将会自动为音频做缓冲。“preload=auto”的意思就是告诉浏览器在播放音频时自动提前加载。

在移动设备上播放音频最大的问题就是浏览器对音频格式的支持，表 6-1 描述了各个移动设备对音频格式的支持情况。

表 6-1

浏 览 器	OggVorbis	MP3	WAV
Android WebKit	支持	支持	
Opera Mobile		支持	
Firefox Mobile	支持		支持
iOS Safari		支持	支持

正如我们在表 6-1 所示中看到的，各个设备支持的音频格式各不相同，这让想创建一

个跨浏览器的音频播放功能变得十分麻烦。要想解决这个问题，一个方案是使用多个音频轨道，如果浏览器无法识别第一个标签中的音频格式，就让浏览器尝试第二个。在表 6-1 所示中，我们还可以看到，MP3 格式是得到最广泛支持的，除了 Firefox Mobile，其他设备都支持，因此我们可以使用 MP3 格式的音频作为第一个音频标签，然后，再加上一个支持 Firefox Mobile 的 OggVorbis 格式的音频，这样音频播放功能就可以覆盖到我们提到的所有设备了。修改后的代码如下：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  </head>
  <body>

    <div id="main">
      <audio controls preload="auto" autobuffer>
        <source src="resources/snake_charmer.mp3" />
        <source src="resources/snake_charmer.ogg" />
      </audio>
    </div>

  </body>
</html>
```

6.2.4 延伸

看到这里，大家可能会有疑惑：那些不支持 HTML5 的 audio 标签的浏览器应该怎么办呢？一种解决方案是通过使用 polyfills 脚本来支持，但是，坦率来讲，我没看到用 polyfills 脚本来支持 HTML5 的 audio 功能的好处。因为该脚本是通过 Flash 来实现音频播放的，而 Flash Lite 只是被极少数的设备支持，像 Symbian。另一种解决方案是在 audio 标签内，内嵌一个链接，在支持 audio 标签的设备上，会忽略掉该链接标签，直接显示音频播放界面，在不支持 audio 标签的设备上，会显示一个链接。

```
    <div id="main">
      <audio controls preload="auto" autobuffer>
            <a href="resources/snake_charmer.mp3">play or
download here</a>
      </audio>
    </div>
```

如果你在 Windows Phone 中访问该界面，界面将会显示如图 6-2 所示。

点击界面上的链接，设备将打开设备默认的音频播放器开始播放音频，如图 6-3 所示。

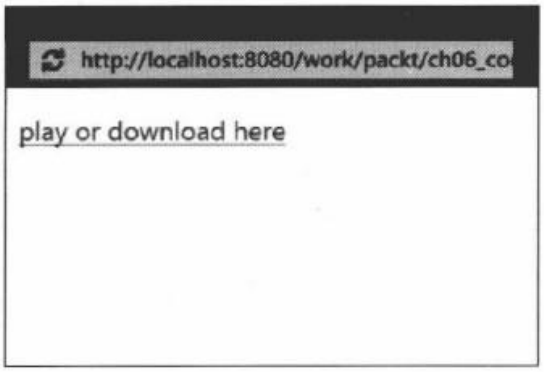


图 6-2

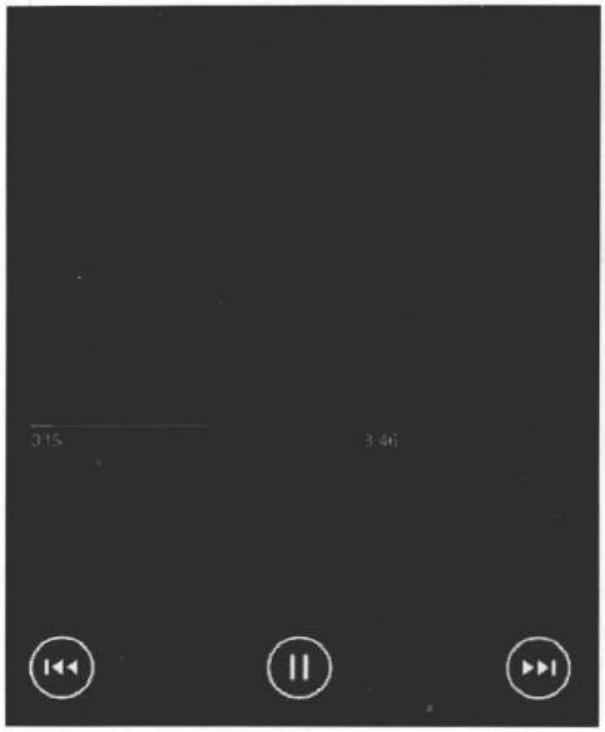


图 6-3

W3C Audio 工作组

当前的 audio 元素还缺乏客户端的 API，W3C 设置了一个音频工作小组（http://www. w3.org/2011/audio/）来专门处理这个问题。该工作组会构建一个新的 API，该 API 将会为音频的交互式应用提供很多新的特性，比方说可以在脚本中直接处理或合成多个音频流。如果读者有兴趣，可以订阅 W3C 的邮件组参与该 API 特性的讨论：`public-audio-request@w3.org`。

6.3 移动设备上播放视频

适用设备：iOS、Android、BlackBerry、webOS，以及使用 Opera Mobile 或 Firefox Mobile 的设备

在桌面领域知名的在线视频网站，比方说 YouTube(http:// www.youtube.com)、Vimeo（http://www.vimeo.com）等，都有为移动设备专门提供一个优化后的版本。对于移动设备

来说，视频流式播放是非常重要的，因为用户观看视频时，他的期望是能够很快地看完一段视频，尤其在观看一些像 YouTube 上的小短片的时候，这些短片只需要很少的时间来缓冲，并且能很快播放完毕。我们如何能在移动设备上做到同样的用户体验呢？接下来我们就用一个例子看看应该怎么做。

6.3.1 准备

创建一个名为“ch06r02.html”的 HTML 文件。

6.3.2 实践

在刚创建的文件中写入下面代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>

    <div id="main">
      <video id="movie" width="320" height="240" preload controls>
        <source src=" http://diveintohtml5.info/i/test.mp4" />
        <source src=" http://diveintohtml5.info/i/pr6.webm"
type='video/webm; codecs="vp8, vorbis"' />
        <source src=" http://diveintohtml5.info/i/pr6.ogv"
type='video/ogg; codecs="theora, vorbis"' />
        <object width="320" height="240" type="application/x-
shockwave-flash"data=" http://releases.flowplayer.org/swf/flowplayer-
3.2.1.swfflowplayer-3.2.1.swf"> data="flowplayer-3.2.1.swf">
          <param name="movie" value=" http://releases.flowplayer.org/
swf/flowplayer-3.2.1.swf" />
          <param name="allowfullscreen" value="true" />
          <param name="flashvars" value='config={"clip":
{"url":http://diveintohtml5.info/i//test.mp4", "autoPlay":false,
"autoBuffering":true}}' />
          <p>Download video as <a href=" http://diveintohtml5.
info/i/pr6.mp4">MP4</a>, <a href=" http://diveintohtml5.info/i/
pr6.webm">WebM</a>, or <a href=" http://diveintohtml5.info/i/pr6.
ogv">Ogg</a>.</p>
        </object>
      </video>
```

```
        <p>Try this page in Safari 4. Or you can <a href=" http://
diveintohtml5.info/i//test.mp4">download the video</a> instead.</p>
    </div>

  </body>
</html>
```

然后在移动设备浏览器上打开该文件，新的页面效果如图 6-4 所示。

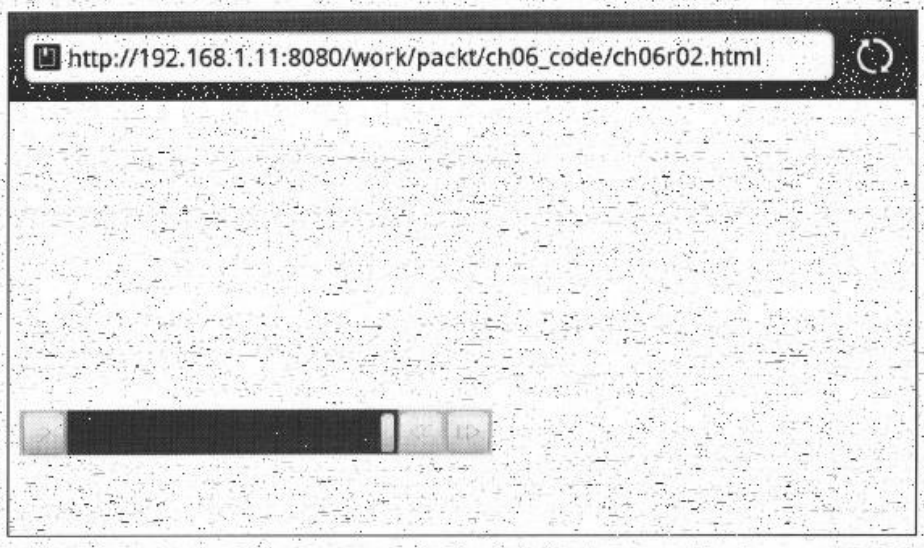


图 6-4

6.3.3 回顾

示例中的部分代码是从 Mark Pilgrim 的“Dive into HTML5”（http://diveintohtml5.info）网站上抄过来的，看了上面例子的代码，你可能会想：“为了弄个视频播放功能，要写那么多代码啊。”，接下来，我们就来看看例子的各部分代码是怎样工作的。iOS 和 Android 设备都支持 H.264(mp4)视频格式，把 webm 和 ogv 格式添加到代码中是为了能够支持桌面浏览器和其他移动设备。

如果你使用了多个<source>元素，iOS 设备只会识别第一个，再加上早期版本的 iOS 只支持 H.264+AAC+MP4 视频格式，因此，你必须把 MP4 格式的元素放最前面，这个问题在 iOS4.0 已经被解决了，但是，在本小节的示例中，我们还是把 MP4 格式的<source>元素放在了第一个：

```
<source src=" http://diveintohtml5.info/i/test.mp4" />
<source src=" http://diveintohtml5.info/i/pr6.webm" type='video/webm;
codecs="vp8, vorbis"' />
<source src=" http://diveintohtml5.info/i/pr6.ogv" type='video/ogg;
codecs="theora, vorbis"' />
```

示例中的关于 Flash fallback 的这段代码是为了支持那些不支持 HTML5 video 标签的设备：

```
<object width="320" height="240" type="application/x-shockwave-
flash"data=" http://releases.flowplayer.org/swf/flowplayer-
3.2.1.swfflowplayer-3.2.1.swf"> data="flowplayer-3.2.1.swf">
<param name="movie" value=" http://releases.flowplayer.org/swf/
flowplayer-3.2.1.swf" />
<param name="allowfullscreen" value="true" />
<param name="flashvars" value='config={"clip": {"url": "resources
http://diveintohtml5.info/i//test.mp4", "autoPlay":false,
"autoBuffering":true}}' />
<p>Download video as <a href="test.mp4">MP4</a>, <a href="test.
webm">WebM</a>, or <a href="test.ogv">Ogg</a>.</p>
</object>
```

6.3.4 延伸

Mark Pilgrim 的“Dive into HTML5”网站上，详细地叙述了在做跨浏览器渲染 video 标签时遇到的种种问题，你可以在这儿读到更多的信息：`http://diveintohtml5.info/video.html`。

Android 在 2.3 版本以前，对 video 标签的支持不是很好；在 Android 的早期版本中，<source>元素中的封装的类型是很让人迷惑的，让浏览器识别一个视频资源的唯一方法就是去除 source 标签上的类型属性，然后你的 H.264+AAC+MP4 格式的视频文件一定要使用.mp4 的扩展名。但是其他类型的视频文件你还是可以使用 type 属性，但是要注意的是 H.264 是 android2.2 支持的唯一视频格式，在 Android 2.3 版本已经修复了上面提到的问题。

另外，controls 属性在 Android 中也不支持。在标签中使用 controls 属性倒是没什么副作用，只是，Android 设备上无法显示出视频控制界面，你需要单独提供脚本来控制视频，至少你需要为你的视频提供一个用户可以点击播放的功能，同样的，这个问题同样也在 Android 2.3 版本已经修复。

6.4 使用离线缓存

适用设备：iOS、Android、webOS，以及使用 Opera Mobile 或 Firefox Mobile 的设备

除了设备接入功能外，离线缓存可以说是移动设备最重要的功能之一，移动用户和桌面用户最大的区别就是移动用户永远都是处于运动状态，他们不像桌面用户，有稳定的网络接入，移动用户访问的应用的时候网络状况随时都在变化，可能从 3G 切换到 WiFi，甚

至出现完全断网的情况，比方说在隧道的时候，离线缓存可以有效的缓解网络连接的问题。表 6-2 所示是各种设备对于离线缓存的支持情况：

表 6-2

设　备	是否支持
iOS	是（3.2 版本以上）
Android	是（2.1 版本以上）
Windows Mobile	否
Blackberry v6.0 及以上	否
Symbian 60	否
Palm webOS	是
Opera Mobile	是
Firefox Mobile	是

6.4.1 准备

创建一个名为“default.appcache”的文本文件。

6.4.2 实践

在刚才创建的文件中写入下面的代码：

```
CACHE MANIFEST
# version 1
img/apple-touch-icon.png
#img/splash.png
NETWORK:
#http://example.com/api/

FALLBACK:
```

接下来，创建一个名为“ch06r03.html”的文件，并写入下面的的代码：

```
<!doctype html>
<html manifest="default.appcache">
<head>
<title>Mobile Cookbook</title>
```

```
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>
  <img src="img/apple-touch-icon.png" alt="Apple Touch Icon" />
</body>
</html>
```

你在移动设备浏览器上访问该HTML文件，然后，切断移动设备的所有网络连接，重新浏览该网页，你会发现，该网页仍然可以加载。

6.4.3 回顾

在CACHE MANIFEST语句下面，我们列出所有的想要做离线缓存的文件，另外，包含CACHE MANIFEST语句的文件会自动做离线缓存，例如下面这个文件：

```
CACHE MANIFEST
# version 1
img/apple-touch-icon.png
#img/splash.png
```

在NETWORK语句下面列出了所有你不想缓存的文件的URL，这些文件每次都会重新加载。举个例子，API调用的返回结果，你肯定不想缓存一个动态API的返回结果。如果你想缓存的API调用拥有相同的前缀，你不需要一个一个地添加这些API调用，你只需要把前缀添加到NETWORK下面就行了，比方说，你不想缓存下面的这些URL：

```
http://example.com/api/?loc=paris
http://example.com/api/?loc=london
```

你不需要一个一个地把它们添加到NETWORK下面，你可以像下面这么处理：

```
NETWORK:
http://example.com/api/
```

在FALLBACK语句后面，我们可以列出所有的网络URL资源的替代选择，当浏览器是离线或者远端服务器不可用的时候，网页可以使用这些替代资源作占位符。

6.4.4 延伸

也许有的同学会问，为什么要用.appcache作为文件的扩展名，而不是.manifest？这个

问题的原因是该扩展名是由 WHATWG 组织推荐的。因此其可以看做是一个标准，使用这个扩展名可以支持更多浏览器。

也许你还会担心，移动设备的浏览器是否能够识别这类扩展名的文件？不用担心，我们可以使用 AddType 来添加解决这个问题。在你的.htaccess 文件中为 MIME 类型[①]中添加上.appcache 和.manifest 类型：

```
AddType text/cache-manifest appcache manifest
```

Appcache Facts

如果你想了解更多关于 Appcache 的信息，可以直接访问 Appcache Facts 网站（http://appcachefacts.info/），该网站上有更多关于 Appcache 很有价值的信息。该网站上还收罗了很多你可以更加深入了解 Appcache 的资源链接：

- Dive Into HTML5 – Let's Take This Offline: (http://diveintohtml5.info/ offline.html)
- Google Code blog – Using AppCache to Launch Offline: (http://googlecode. blogspot.com/2009/04/gmail-for-mobile-html5-series-using.html)
- HTML5 Rocks – A Beginner's Guide to Using the Application Cache: (http://www.html5rocks.com/tutorials/appcache/beginner/)
- MDN Doc Center – Offline resources in Firefox: (https://developer.mozilla. org/en/offline_resources_in_firefox)
- Safari Developer Library – Storing Data on the Client: (http://developer.apple. com/ library/safari/#documentation/appleapplications/reference/ SafariWebContent/Client-SideStorage/Client-SideStorage.html)
- Cache Manifest Validator – Online validator, JSON(P) validation API, and TextMate bundle: (http://manifest-validator.com/)

WHATWG 组织关于 Appcache 的官方描述

如果你想对 Appcache 的规格更加了解，你可以读一下关于 offline 的官方标准表述：

```
http://www.whatwg.org/specs/web-apps/current-work/multipage/offline.html
```

[①] MIME 类型就是设定某种扩展名的文件用一种应用程序来打开的方式类型，当该扩展名文件被访问的时候，浏览器会自动使用指定应用程序来打开。多用于指定一些客户端自定义的文件名，以及一些媒体文件打开方式。——译者注

6.5 使用网络存储（Web Storage）

适用设备：基于浏览器的所有设备

网络存储对于离线应用来说非常有用，尤其是那些抓取内容或者 E-mail 应用，这些应用都不需要实时加载，可以提前加载，其他时间再看。当开发人员在谈论网络存储（Web Storage）的时候，他们通常指的的是本地存储（local storage）部分。本地存储从本质上来说就是一个键值对的存储系统。除了 Web Storage 之外，在 HTML5 中，引入了两个新的存储特性：Indexed Database API 和 Web SQL Database。

下面我们就来看一下三种存储方式的优缺点，如表 6-3 所示。

表 6-3

类　型	优　点	缺　点
Web Storage	简单易用，浏览器支持度高	数据不安全
Indexed Database	类似 NoSQL 的结构化存储	目前浏览器支持度不高，不支持 SQL 查询
Web SQL Database	快； 支持很多 SQL 查询； 新出的大部分移动浏览器都支持；	W3C 还没不打算把该特性变成 HTML 的标准

从表 6-3 所示中，我们可以看出：从浏览器支持度角度来说，Web Storage 的支持度最好的；其次是 Web SQL Database。

Web SQL Database 比 Web Storage 提供了更多更好的特性。在本小节中，我们主要会关注 Web Storage 和 Web SQL Database，暂时就不讲关于 Indexed Database 的东西了。

6.5.1 准备

首先创建一个名为“ch06r04.html”的 HTML 文件。

6.5.2 实践

在刚才创建的 HTML 文件中写入如下代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script src="js/modernizr.custom.54685.js"></script>
</head>
<body>
  <section>
    <p>Values are stored on <code>keyup</code></p>
    <p>Content loaded from previous sessions:</p>
    <div id="previous"></div>
  </section>
  <section>
    <div>
      <label for="local">localStorage:</label>
      <input type="text" name="local" value="" id="local" />
    </div>
```

接下来添加 JavaScript 代码：

```
    <script>

var addEvent = (function () {
      if (document.addEventListener) {
        return function (el, type, fn) {
          if (el && el.nodeName || el === window) {
            el.addEventListener(type, fn, false);
          } else if (el && el.length) {
            for (var i = 0; i < el.length; i++) {
              addEvent(el[i], type, fn);
            }
          }
        };
      } else {
        return function (el, type, fn) {
          if (el && el.nodeName || el === window) {
            el.attachEvent('on' + type, function () { return
fn.call(el, window.event); });
          } else if (el && el.length) {
            for (var i = 0; i < el.length; i++) {
              addEvent(el[i], type, fn);
            }
          }
```

```
        };
      }
    })();
    function getStorage(type) {
      var storage = window[type + 'Storage'],
        delta = 0,
        li = document.createElement('li');

      if (!window[type + 'Storage']) return;

      if (storage.getItem('value')) {
        delta = ((new Date()).getTime() - (new Date()).
setTime(storage.getItem('timestamp'))) / 1000;

        li.innerHTML = type + 'Storage: ' + storage.getItem('value') +
' (last updated: ' + delta + 's ago)';
      } else {
        li.innerHTML = type + 'Storage is empty';
      }

      document.querySelector('#previous').appendChild(li);
    }

    getStorage('local');

    addEvent(document.querySelector('#local'), 'keyup', function () {
      localStorage.setItem('value', this.value);
      localStorage.setItem('timestamp', (new Date()).getTime());
    });

    </script>
```

最后，加上一些结束标签，构成一个完整的 HTML 文档：

```
  </section>
</body>
</html>
```

该本地存储方案甚至可以在海豚浏览器上运行，海豚浏览器是 Samsung 公司制造的移动设备的默认浏览器，所有的 Android 设备都可以安装该浏览器。当使用海豚浏览器渲染页面时，你可以输入任何字符，在书中的例子中，如果你输入了“hello world”，然后点击刷新，页面将会显示如图 6-5 中所示的信息。

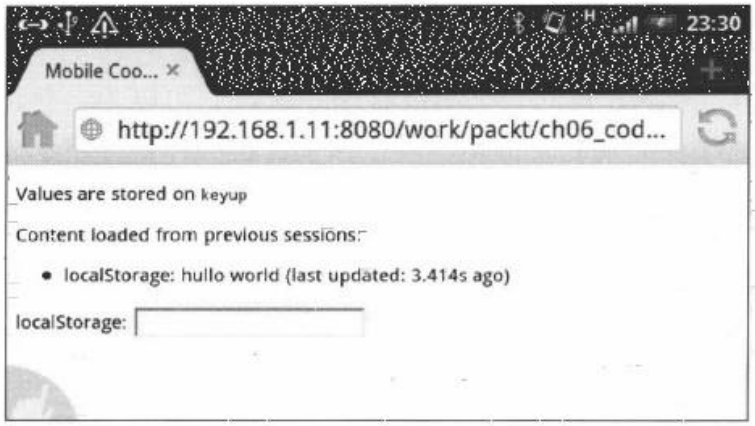


图 6-5

6.5.3 回顾

就像我们前面提到的那样，网络存储就是简单的键值对存储，用户可以通过使用 set 和 get 方法存取值。

使用 setItem 方法存储某个值：

```
localStorage.setItem('value', this.value);
```

使用 getItem 方法取出之前存储的值：

```
storage.getItem('value')
```

想找一个实现网络存储的 polyfill 脚本？ jQueryOffline 就是一个非常好用的离线存储插件。它的 API 接口和 HTML5 的本地存储保持一致，因此，通过使用 jQuery Offline，开发人员就可以在那些不支持本地存储的浏览器使用同样的 API 进行操作。这样就可以在不增加开发人员工作量的前提下，让离线缓存的功能可以覆盖到更多的移动设备。该 polyfill 脚本的实现原理就是：如果浏览器不支持 localStorage，重新向远端的服务器请求一次，你可以在这儿了解其更多的实现细节：http://html5demos.com/database。

6.5.4 延伸

Web SQL Database 可以作为本地存储的替代品，它备受惯于使用 SQL 的开发人员推崇。Remy Sharp 做了一个很好的 Demo 教大家如何使用 Web SQL Database，你可以在这儿看到这个 Demo：http://html5demos.com/database。

网络存储的门面层

网络存储门面层库文件主要作用是让开发人员可以更容易地开发出既支持 HTML5 又

支持 Gears 的离线存储功能。

Gears 是 Google 早期开发的一个离线存储系统，它可以为一些浏览器，像 IE6、IE Mobile4.0.1 等提供离线存储功能。但是，现在该项目已经停止开发了。

你可以从下面的网址了解更多关于网络存储的门面库文件的信息：http://google-opensource.blogspot.com/2009/05/web-storage-portability-layer-common.html。

HTML5 存储类型大战

下面这篇文章详细讲解了关于 HTML5 的三种存储方式：Local Storage、IndexedDB 和 Web SQL 各自的优劣：http://csimms.botonomy.com/2011/05/html5-storage-wars-localstorage- vs-indexeddb-vs- web-sql.html。

6.6 使用 Web Workers

适用设备：iOS5、BlackBerry 以及使用 Opera Mobile 和 Firefox Mobile 的设备

大多数有 Java、Python 或者.Net 背景的程序员，对于多线程和并发编程都不会陌生，JavaScript 曾经因为缺乏高性能的多线程框架而备受诟病，随着 HTML5 的发布，它可以扩展支持并发，从根本上增加前端页面的数据处理能力，JavaScript 也就不再只是一个脚本语言了，随着越来越多使用 JavaScript 写成的精巧的小工具的发布，JavaScript 开始在前端运算中扮演着越来越重要的角色。表 6-4 所示是各个设备对 Web Worker 的支持情况。

表 6-4

设　备	是否支持
iOS	Yes（5.0 以上版本）
Android	No
Windows Mobile	No
BlackBerry	Yes（6.0 以上版本）
Symbian	No
Palm webOS	No
Opera Mobile	Yes
Firefox Mobile	Yes

6.6.1 准备

首先，创建一个名为“math.js”的 JavaScript 文件。

6.6.2 实践

在刚才创建的文件中写入如下代码：

```
/* math.js */

function addNumbers(x,y) {
  return x + y;
}

function minNumbers(x,y) {
  return x - y;
}

/*
  Add an eventlistener to the worker, this will
  be called when the worker receives a message
  from the main page.
*/
this.onmessage = function (event) {
  var data = event.data;

  switch(data.op) {
  case 'mult':
    postMessage(minNumbers(data.x, data.y));
    break;
  case 'add':
    postMessage(addNumbers(data.x, data.y));
    break;
  default:
    postMessage("Wrong operation specified");
  }
};
```

接下来，我们再创建一个名为“ch06r05.html”的 HTML 文件，并写入下面的代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script src="js/modernizr.custom.54685.js"></script>
</head>
<body onload="loadDeals()">
  <input type="text" id="x" value="6" />
  <br />
  <input type="text" id="y" value="3" />
  <br />
  <input type="text" id="output" />
  <br />
  <input type="button" id="minusButton" value="Subtract" />
  <input type="button" id="addButton" value="Add" />
  <script>
  if (Modernizr.webworkers){
     alert('hi');
   }

   /* Create a new worker */
   arithmeticWorker = new Worker("js/math.js");
   /*
     Add an event listener to the worker, this will
     be called whenever the worker posts any message.
   */
   arithmeticWorker.onmessage = function (event) {
     document.getElementById("output").value = event.data;
   };
   /* Register events for buttons */
   document.getElementById("minusButton").onclick = function() {
     /* Get the values to do operation on */
     x = parseFloat(document.getElementById("x").value);
     y = parseFloat(document.getElementById("y").value);
     message = {
       'op' : 'min',
       'x'  : x,
       'y'  : y
     };
     arithmeticWorker.postMessage(message);
   }
   document.getElementById("addButton").onclick = function() {
     /* Get the values to do operation on */
     x = parseFloat(document.getElementById("x").value);
     y = parseFloat(document.getElementById("y").value);
     message = {
```

```
        'op' : 'add',
        'x'  : x,
        'y'  : y
      };

      arithmeticWorker.postMessage(message);
    }
    </script>
  </body>
  </html>
```

然后，我们在移动设备中打开该 HTML 文件，我们可以看到三个文本框和两个按钮用作计算，在图 6-6 所示的截图中，当我们输入 6 和 3，然后点击 Add 按钮，就可以看到 9 显示在第 3 个文本框里。

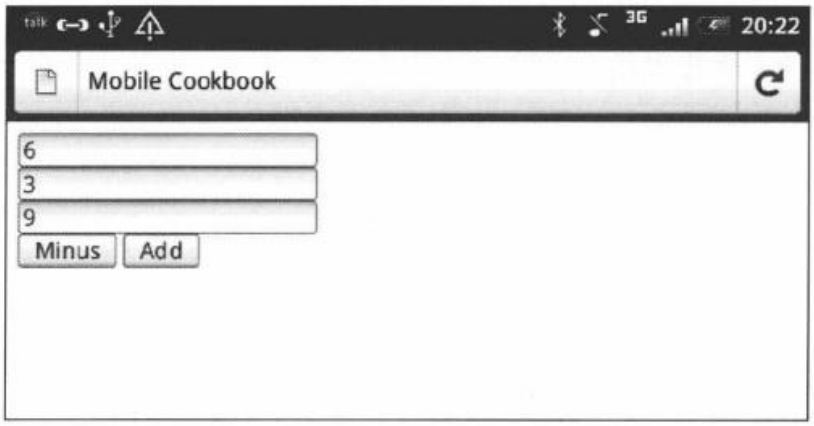


图 6-6

6.6.3 回顾

纵观整个 math.js 的代码，我们可以把它分为 3 个部分：

- 实际的数学函数实现
- 从主线程获取请求事件
- 向主线程回送处理结果

实际的数学函数实现这段代码非常好理解，addNumber 表示加法的函数，minNumber 表示减法：

```
/* math.js */

function addNumbers(x,y) {
```

```
  return x + y;
}

function minNumbers(x,y) {
  return x - y;
}
```

接下来就是 onmessage 函数，该函数的作用是帮助 math.js 从 HTML 文档中获取执行业务逻辑时需要的信息：

```
this.onmessage = function (event) {
  var data = event.data;

  ...
};
```

一旦 math.js 的工作线程从主线程获取到了足够的信息，它就会开始处理自己的业务逻辑，然后把结果通过 postMessage 方法发送回给主线程：

```
switch(data.op) {
  case 'mult':
    postMessage(minNumbers(data.x, data.y));
    break;
  case 'add':
    postMessage(addNumbers(data.x, data.y));
    break;
  default:
    postMessage("Wrong operation specified");
}
```

整个 HTML 文档，我们也可以分成 3 个部分来看：

- 创建一个工作线程
- 调用该线程并发送相应的数据，使其开始工作
- 获取工作线程的结果

在 HTML5 中，创建一个工作线程非常简单，你只需要使用语句“new Worker(“math.js”)”：

```
/* Create a new worker */
  arithmeticWorker = new Worker("js/math.js");
```

至于发送消息给工作进程，可以使用前面讲解“math.js”脚本时介绍过的 postMessage

方法，发送的消息结构可以是一些键值对的集合：

```
message = {
  'op' : 'min',
  'x'  : x,
  'y'  : y
};
arithmeticWorker.postMessage(message);
```

至于获取工作线程的结果，则可以使用前面讲解“math.js”脚本时介绍过的 onMessage 方法：

```
arithmeticWorker.onmessage = function (event) {
    document.getElementById("output").value = event.data;
};
```

6.7 使用 session 和 history API 构建类 Flash 导航效果

适用设备：基于浏览器的所有设备

之前，为了平衡网页导航的平滑性和对 SEO（Search Engine Optimization：搜索引擎优化）的影响，开发人员不得不在 URL 中添加一个“#”标签构建一个假的 URL 来导航网页到指定的位置（译者注：URL 中的#标签只在客户端起作用，把浏览器的可视域导航到#标签指定的段落上，#标签后面的内容不会发送到服务端，对服务端不会有任何影响）。现在，通过使用新的 history API，我们不在需要使用这样黑客式的解决方案了。通过使用 Ajax 和 History API，我们可以动态地更新一个 URL。表 6-5 是各个移动设备平台对 History API 的支持情况：

表 6-5

设备平台	是否支持
iOS	支持(4.2 版本以上)
Android	支持（2.2 版本以上）
Windows Mobile	不支持
BlackBerry	不支持
Symbian	支持（5.2 版本以上）
PalmwebOS	不支持
OperaMobil'e	不支持
Firefox Mobile	支持

6.7.1 准备

创建一个名为“ch06r06.html”的HTML文件。

6.7.2 实践

在刚创建的文件中写入下面的代码：

```
<!doctype html>
<html>
<head>
<title>Mobile Cookbook</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script src="js/modernizr.custom.54685.js"></script>
<style>
section {width:300px; background:#ccc; padding:5px; margin:20px auto;}
html, body, figure {padding:0; margin:0;}
figcaption {display:block;}
</style>
</head>
<body>
  <section id="gallery">
    <p class="photonav"><a id="photoprev" href="ch06r06_b.html">&lt;
Previous</a> <a id="photonext" href="ch06r06_a.html">Next ></a></p>
    <figure id="photo">
      <img id="photoimg" src="http://placekitten.com/300/300"
alt="Fer" width="300" height="300"><br />
      <figcaption>Adagio, 1982</figcaption>
    </figure>
  </section>
  <script src="js/nav.js"></script>
</body>
</html>
```

然后，再创建一个名为“ch06r06_a.html”的文件，并写入如下的代码：

```
<p class="photonav"><a id="photoprev" href="ch06r06_b.html">&lt;
Previous</a> <a id="photonext" href="ch06r06_b.html">Next ></a></p>
<figure id="photo">
  <img id="photoimg" src="http://placekitten.com/300/301" alt="Fer"
width="300" height="300">
  <figcaption>Aida, 1990</figcaption>
</figure>
```

然后，再创建一个名为“ch06r06_b.html”的文件，并写入如下的代码

```
<p class="photonav"><a id="photoprev" href="ch06r06_a.html">&lt;
Previous</a> <a id="photonext" href="ch06r06_a.html">Next ></a></p>
<figure id="photo">
  <img id="photoimg" src="http://placekitten.com/300/299" alt="Fer"
width="300" height="300">
  <figcaption>Air Cat, 2001</figcaption>
</figure>
```

再接着，创建一个 javascript 文件，写入如下的代码，记得把其中的 URL 换成你自己例子的 URL：

```
function supports_history_api() {
  return !!(window.history && history.pushState);
}
function swapPhoto(href) {
  var req = new XMLHttpRequest();
  req.open("GET",
  "http://localhost /work/packt/ch06_code/" +
  href.split("/").pop(),
  false);
  req.send(null);
  if (req.status == 200) {
    document.getElementById("gallery").innerHTML = req.responseText;
    setupHistoryClicks();
    return true;
  }
  return false;
}

function addClicker(link) {
  link.addEventListener("click", function(e) {
    if (swapPhoto(link.href)) {
      history.pushState(null, null, link.href);
      e.preventDefault();
    }
  }, true);
}

function setupHistoryClicks() {
  addClicker(document.getElementById("photonext"));
  addClicker(document.getElementById("photoprev"));
}
```

```
window.onload = function() {
  if (!supports_history_api()) { return; }
  setupHistoryClicks();
  window.setTimeout(function() {
    window.addEventListener("popstate", function(e) {
      swapPhoto(location.pathname);
    }, false);
  }, 1);
}
```

在移动设备浏览器中访问该页面，当你点击“Previous”或者“Next”按钮时，该页面不会刷新，但是，如果你去查看URL，会发现URL已经被更新了，如图6-7所示。

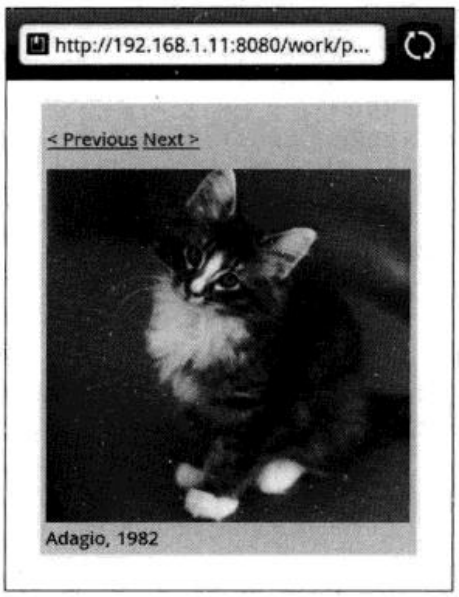


图6-7

6.7.3 回顾

history.pushState语句的作用只是把新的URL地址推送到浏览器的地址栏中：

```
history.pushState(null, null, link.href);
```

真正完成页面导航的是发送到服务端的Ajax请求，因此该页面并不会重新加载，但是URL会被更新，具体的实现细节全在代码中：

```
function swapPhoto(href) {
  var req = new XMLHttpRequest();
  req.open("GET",
  "http://192.168.1.11:8080/work/packt/ch06_code/" +
```

```
    href.split("/").pop(),
    false);
    req.send(null);
    if (req.status == 200) {
      document.getElementById("gallery").innerHTML = req.responseText;
      setupHistoryClicks();
      return true;
    }
    return false;
  }
```

6.7.4 延伸

如果你想更多地了解 history API，你可以到如下的网址看到该 API 的设计规格：http://www.whatwg.org/specs/web-apps/current-work/multipage/history.html。

另外，Mark Pilgrim 也在 Dive into HTML5 网站上对该 API 进行了详细的讲解：http://diveintohtml5.info/history.html。

你还可以看看 Mozilla 的 MDC 文档：https://developer.mozilla.org/en/DOM/Manipulating_ the_browser_history。

Place Kitten

如果你想知道例子中的小猫图片从哪儿来的，它来自于 Place Kitten 网站：http://placekitten.com/。该网站提供了一个简单的服务接口，方便用户获取一些小动物的图片，在设计页面可以用这些图片做图片占位符。

第 7 章
移动设备调试

本章内容：

- 使用 Opera Dragonfly 远程调试
- 使用 weinre 远程调试
- 在移动设备上使用 Firebug
- 使用 JS Console 远程调试
- 配置移动端 Safari 调试

7.1 简介

虽然调试会占据大量的时间，但是对于桌面和移动设备的网站开发都是非常重要的一个方面。在本章，我们会使用某些移动设备的调试工具，让调试工作更加方便、快捷，从而使网站开发效率更高。

7.2 使用 Opera Dragonfly 远程调试

适用浏览器：Opera 移动版

由于相对更小的显示屏幕，使得移动调试与桌面调试变得不同。

7.2.1 准备

1. 确保你在一个 WiFi 网络中。

2. 从 http://www.opera.com/下载最新版 Opera 桌面浏览器。

3. 在移动设备中下载 Opera 移动版。

7.2.2 实践

1. 本书完成前，Opera 的版本是 11.50，当你阅读时某些功能可能已经改变。

2. 在桌面设备中打开 Opera，在下拉菜单中选择 **Page| Developer Tools | Opera Dragonfly**。

3. 你会在页面底部看到调试工具，点击 **Remote debug configuration**，如图 7-1 所示。

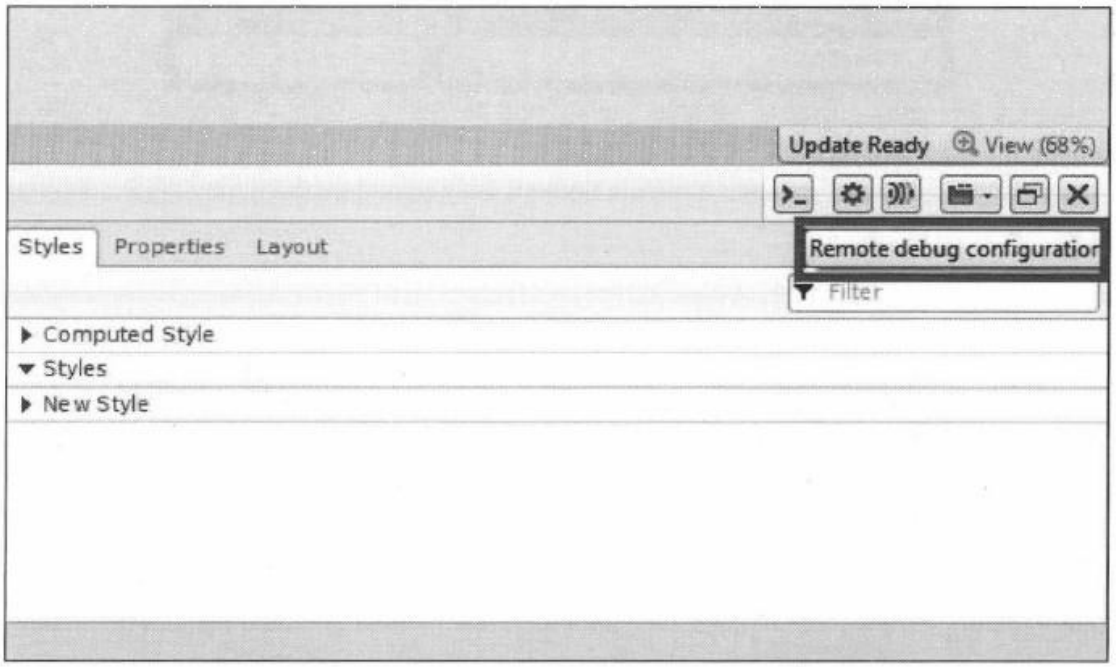


图 7-1

4. 点击 **Remote debug configuration** 后，会出现一个弹出面板，如图 7-2 所示。

5. 在这个面板中会有一个设置端口的文本框和一个 **Apply** 按钮，默认的数字是可用的端口号。

6. 现在打开桌面设备的命令行，输入 ipconfig 后回车，出现的 IPv4 地址就是你的 IP 地址。

7. 在移动设备中打开 Opera 移动版，在 URL 地址栏输入 opera:debug 后，可以看到如图 7-3 所示的页面。

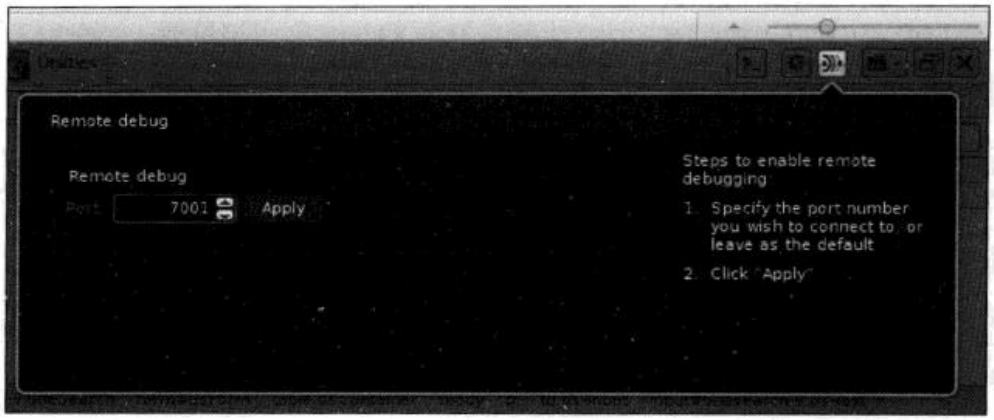


图 7-2

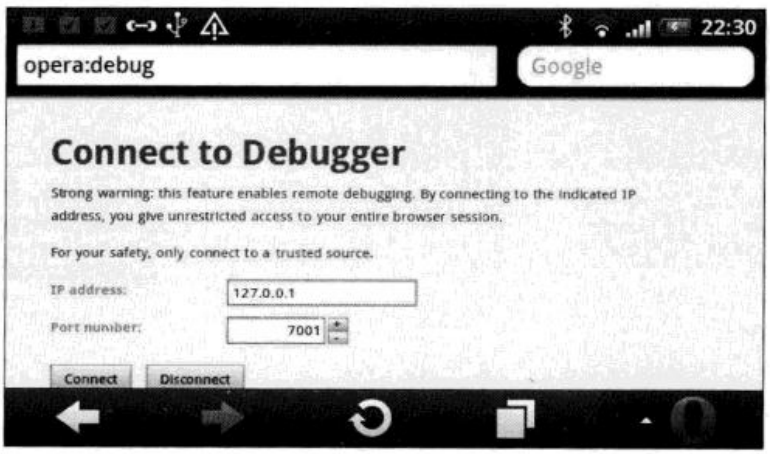


图 7-3

8．输入桌面设备的 IP 地址后点击 **Connect** 后，移动浏览器就可以连接到 Dragonfly，如图 7-4 所示。

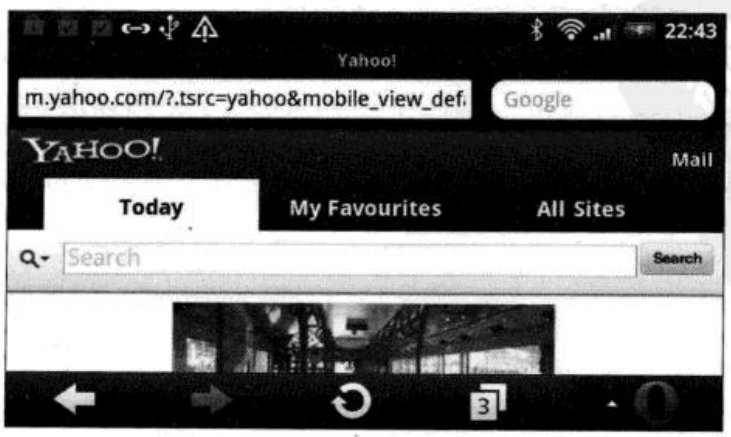


图 7-4

7.2.3 回顾

在 Opera 移动版中打开新的标签，访问 Yahoo.com，回到桌面版点击 **Select the debugging context**，它在右上角的第四个按钮，如图 7-5 所示。从下拉列表中选择 **Yahoo!** 来检查网页！

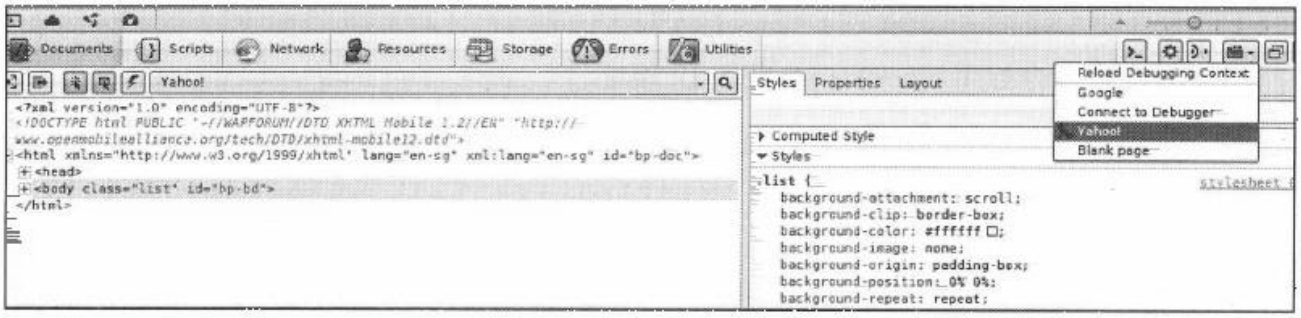


图 7-5

7.2.4 延伸

- 使用 weinre 远程调试
- 使用 JS Console 远程调试

7.3 使用 weinre 远程调试

适用浏览器：iOS、Android、Blackberry、webOS

上一节中，我们看到了如何远程调试 Opera 移动版中的网页，本节中我们会看到如何在其他移动设备中远程调试。**Weinre** 就是一个**网页远程检查工具**。

支持的操作系统如下：

- Android 2.2 浏览器
- Android 2.2、PhoneGap 0.9.2、iOS 4.2
- Safari 移动版
- BlackBerry v6.x 模拟器
- webOS 2.x（无明确版本）

7.3.1 准备

首先，从 weinre 的官方网站下载，这里有两个版本，一个针对 PC，一个针对 Mac：

https://github.com/phonegap/weinre/archives/master

7.3.2 实践

1. 首先在命令行用 ipconfig 获取 IP 地址。

2. 创建一个 HTML 文档命名为 ch07r01.html，输入以下内容，并替换 192.168.1.11 为你的 IP 地址：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">

  </head>
  <body>
    <header>
      <h1>Mobile Cookbook</h1>
    </header>
    <div id="main">
    </div>

    <script src="http://192.168.1.11:8081/target/target-script-
min.js"></script>
  </body>
</html>
```

3. 找到下载好的 weinre.jar 文件地址，在示例中为 C:\ xampp\htdocs\dev\weinre.jar。然后获取 IP 地址，在示例中为：http://192.168.1.11。

4. 现在打开命令行，输入以下命令：

```
java -jar path/to/weinre.jar -httpPort 8081 -boundHost
http://192.168.1.11
```

5. 访问 http://192.168.1.11:8081/，如果配置成功，会出现如图 7-6 所示的页面。

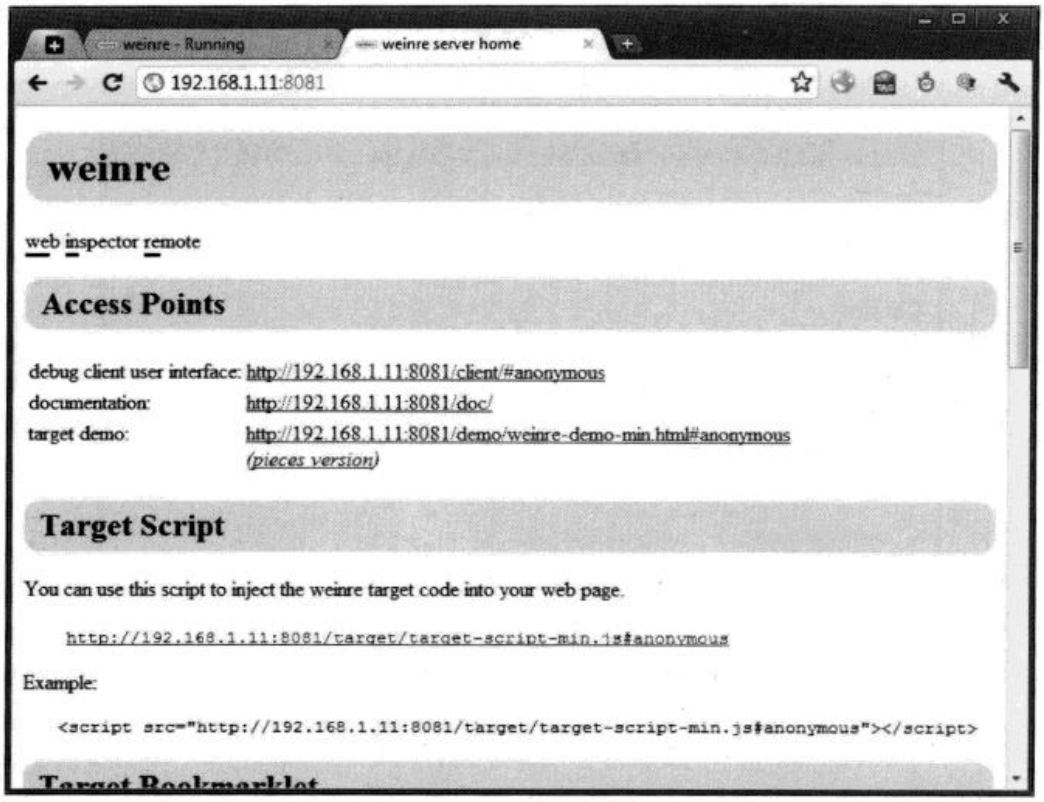


图 7-6

7.3.3 回顾

现在使用移动设备访问刚才创建的示例页面，如图 7-7 所示。

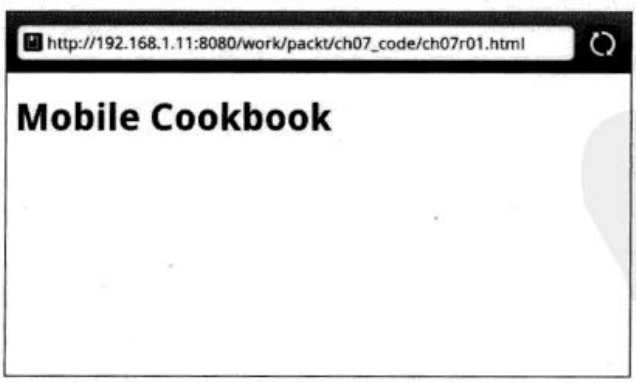


图 7-7

回到桌面设备，点击 **Debug client user interface**，不要打开一个新的标签，打开一个新的窗口。

可以看到类似图 7-8 所示的页面。

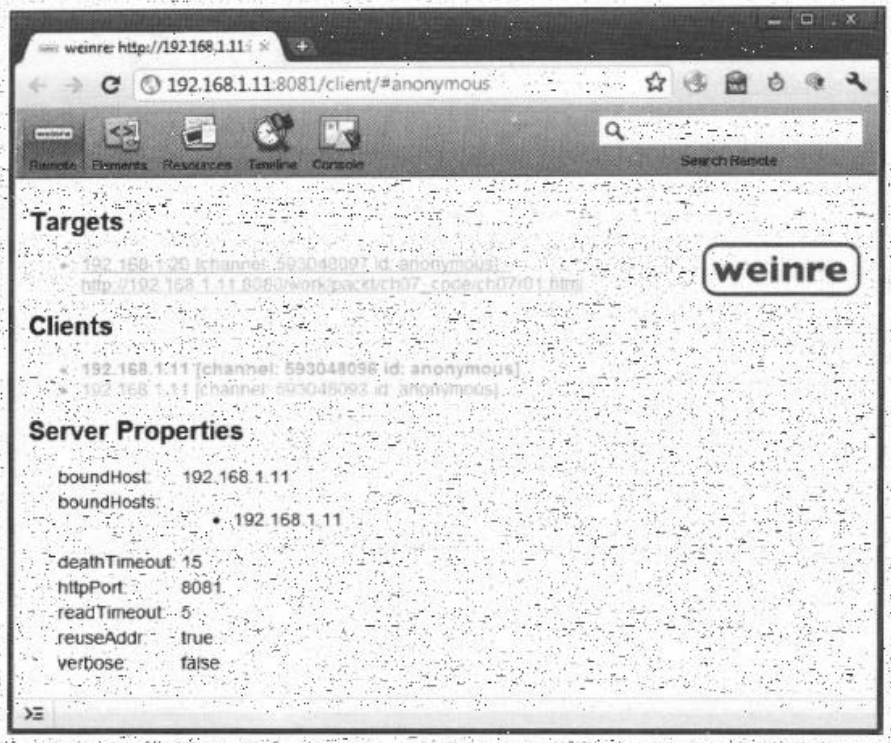


图 7-8

点击 **Elements** 后，你可以检查所有的页面元素（如图 7-9 所示）。

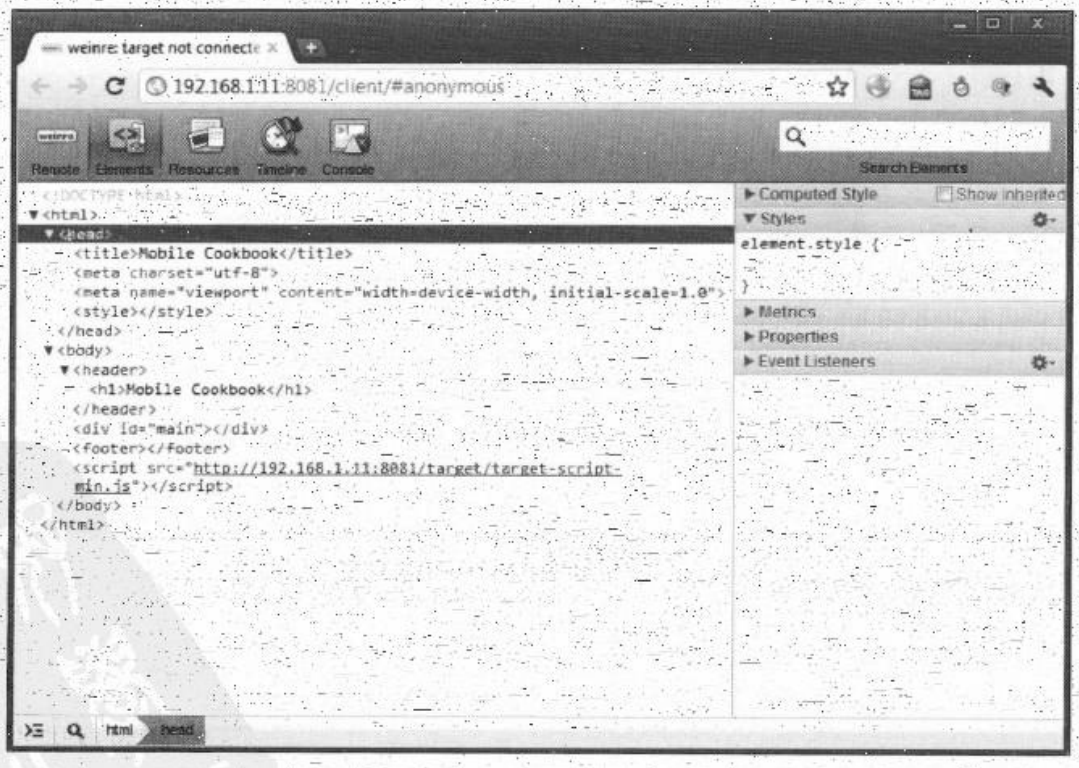


图 7-9

7.4 在移动设备上使用 Firebug

适用浏览器：所有

很多人在 Firefox 和 Chrome 中使用 Firebug，而 Firebug Lite 可以运行在任何支持 JavaScript 的浏览器上。本节我们会看到如何使用 Firebug 来调试。

7.4.1 准备

创建一个 HTML 文档，命名为 ch07r02.html。

7.4.2 实践

1. 在 HTML 中输入以下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">

  </head>
  <body>

    <div id="main">
    </div>

    <script type="text/javascript" src="https://getfirebug.com/
firebug-lite.js"></script>
  </body>
</html>
```

2. 在移动浏览器中打开，如图 7-10 所示。

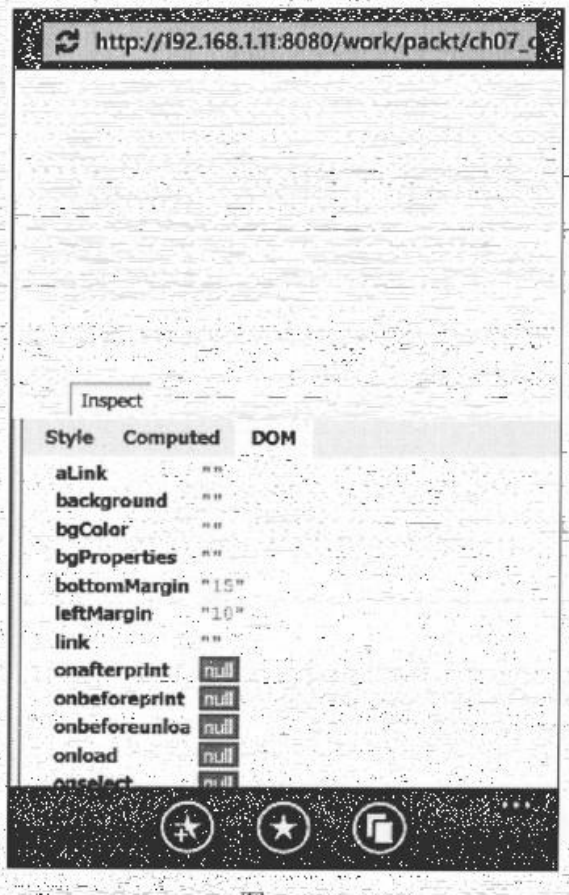


图 7-10

7.4.3 回顾

Firebug Lite 是 Firebug 的 JavaScript 版本，以下代码会从 Firebug 网站读取并加载 Firebug Lite。

```
<script type="text/javascript" src="https://getfirebug.com/firebug-
lite.js"></script>
```

你也可以下载脚本文件放在本地。

你可以检查 HTML、CSS、JavaScript，控制台对于 JavaScript 输入非常有用。

对于 Firebug Lit，有 4 个不同的版本：

- 稳定版
- 调试版
- 测试版

◆ 开发者版

本例中使用的是稳定版，其他版本会在延伸中讲解。

7.4.4 延伸

除了使用在线和本地版本，也可以添加书签，虽然不是所有浏览器都通用。下面是使用说明：

1. 点击 `http://getfirebug.com/firebuglite` 网页中右边的链接。

2. 它在移动浏览器的 URL 后面加入了`#javascript:(function...`。

3. 在 Safari 中保存本页为书签。

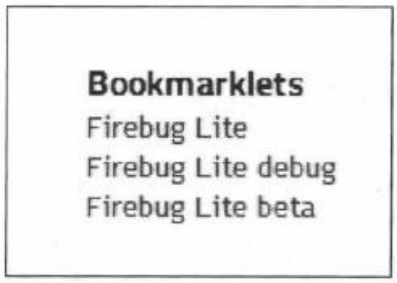


图 7-11

4. 修改书签的名字为 Firebug Lite、Firebug Lite debug 或者 Firebug Lite beta。

5. 保存书签后，打开书签栏，选择 **Firebug Lite**，点击 **Edit**。删除 URL 和#，仅留下 `javascript:(function...`。

6. 现在如果你打开任何网页并点击 **Firebug Lite 书签**，一个 Firebug 调试工具会在页面右下角出现。

调试版

调试版使用与稳定版相同的版本，不同在于预设值，使它更容易针对 Firebug Lite 自己进行调试。

测试版

测试版拥有新功能和问题修复，它应该是非常稳定的（已知还没有版本回归），但它可能包含了一些 bug 或者新功能还未完成。

开发者版

开发者版用于创建和测试新的想法，一旦代码上传到代码仓库，你就可以获取最新的代码，同时会比其他版更加频繁地获得更新。需要注意的一件事情是，开发者版非常不稳定，并且初始加载时很慢。

7.5 使用 JS Console 远程调试

适用浏览器：所有

如果你只需要远程调试 JavaScript，*Remy Sharp* 有一个非常好的工具叫做 JavaScript Console，它对于移动设备调试相当有效。

7.5.1 准备

访问 http://jsconsole.com/后会看到如图 7-12 所示的页面。

图 7-12

7.5.2 实践

1．在页面中输入:listen 后，可以看到如下信息：

Creating connection ...

Connected to "65C1F9F1-6A57-46C0-96BB-35C5B515331F"

2. 后面会有一行类似以下代码的 JavaScript：

<script src="http://jsconsole.com/remote.js?65C1F9F1-6A57-46C0-96BB-35C5B515331F"></script>

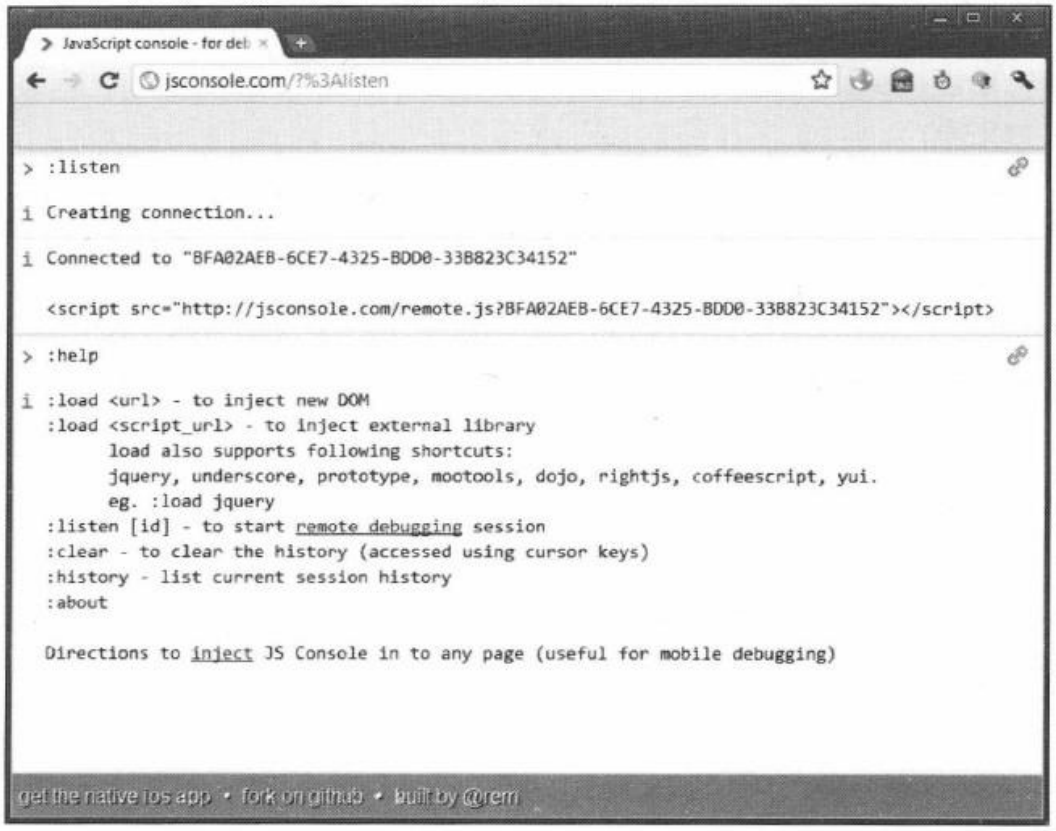


图 7-13

3. 创建一个 HTML 页面，命名为 ch07r04.html，输入以下代码，替换<script>...</script>为你从 jsconsole.com 获取的脚本：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
    >
```

```
    <div id="main">
    </div>

    <script src="http://jsconsole.com/remote.js?04926BFB-44AB-
4979-BAE9-F4A4FA7CE22C"></script>
    <script>
    for (var i=0; i<10; i++) {
      console.log('testing '+i);
    }
    </script>
  </body>
</html>
```

4. 现在如果我们在移动设备上打开页面，我们可以看到桌面中的网页有如下的记录显示：

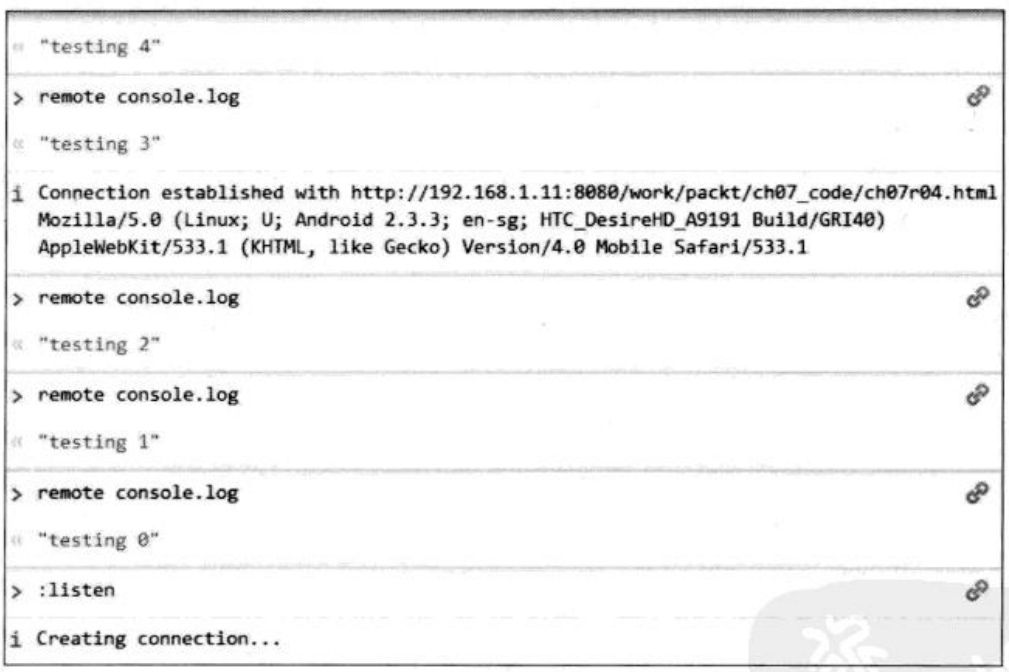


图 7-14

7.5.3 回顾

在下面的循环中，我们使用 console.log 来输出信息：

```
<script>
for (var i=0; i<10; i++) {
  console.log('testing '+i);
}
</script>
```

任何在你的网站上调用 `console.log` 的输出结果，都会显示在监听了你的网站的 jsconsole 中。同样的，如果你在 jsconsole 中运行命令，这些代码都会被注入到你的网站中并将结果返回给 jsconsole。

7.5.4 延伸

整个 JavaScript Console 都是开源的，如果你想知道它是如何工作的可以访问：`https://github.com/remy/jsconsole`。

JS Console iOS 应用

JS Console iOS 版，同样由 *Remy Sharp* 开发，用于没有网络的浏览器的环境中，调试你的 JavaScript。

简单的 iOS 模拟器示例

以下的视频由 *Remy Sharp* 制作，展示了如何在 iOS 中使用 jsconsole.com 远程调试 JavaScript，以及如何接收 log 并发送任意的命令：

`http://www.youtube.com/watch?v=Y219Ziuipvc&feature=player_embedded`

在任意设备上远程调试 JavaScript

在以下的视频中，Remy Sharp 记录了如何使用 jsconsole.com 在任何浏览器或设备上远程调试：

`http://www.youtube.com/watch?v=DSH392Gxaho&feature=player_embedded`

7.6 设置移动 Safari 调试器

适用浏览器：iOS

在 iOS 移动 Safari 中，有一个内建的调试器。

7.6.1 准备

使用 iPhone 进入主屏。

7.6.2 实践

1. 找到并打开 **Setting**，如图 7-15 所示。

2. 选择 **Safari**，如图 7-16 所示。

图 7-15

图 7-16

3. 滑动到底部找到开发者选项，如图 7-17 所示。

4. 调试器默认是 **OFF** 的，如图 7-18 所示。

图 7-17

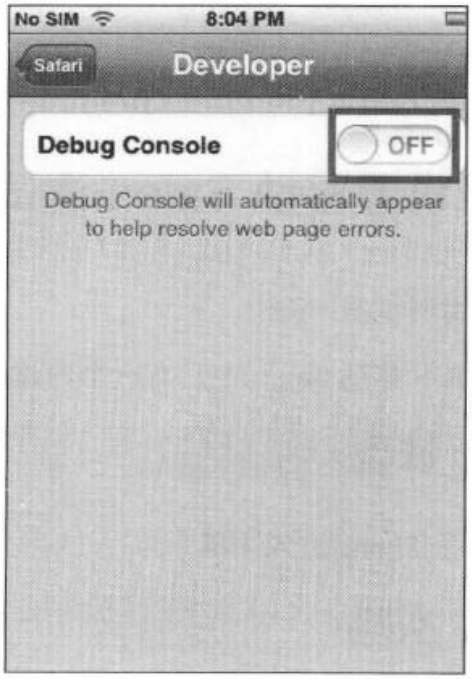


图 7-18

5. 现在我们**开启调试器**，如图 7-19 所示。

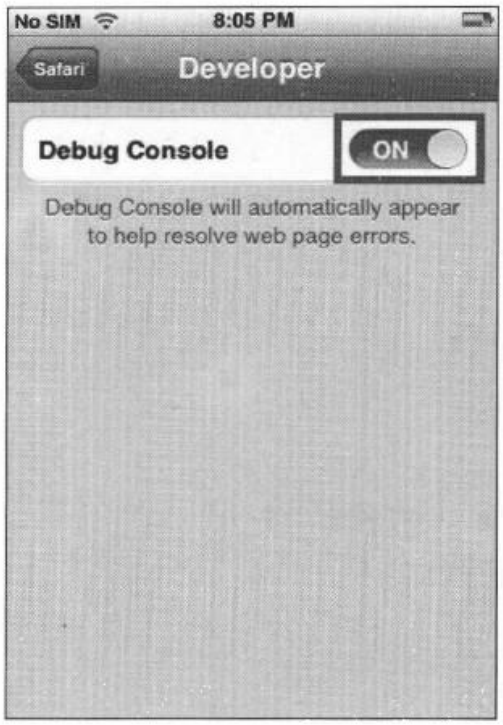


图 7-19

6. 在 Safari 的页面顶部，找到调试器的概况，它就在 URL 栏下面，如图 7-20 所示。

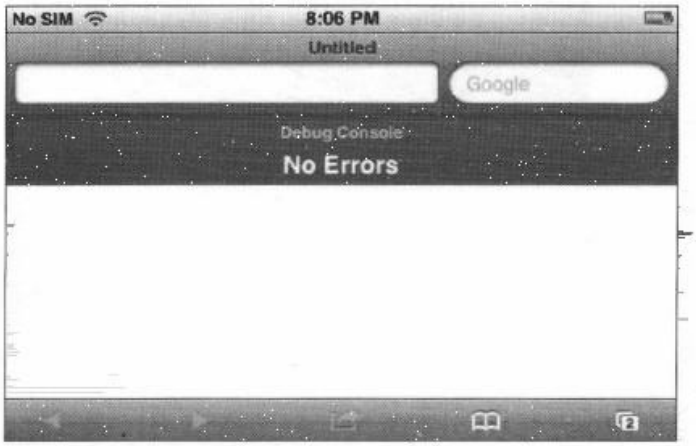


图 7-20

7. 点击概况来查看页面错误的详细信息。

8. 创建一个 HTML 文档命名为 `ch07r05.html`，输入以下代码：

```
<!doctype html>
<html>
   <head>
     <title>Mobile Cookbook</title>
     <meta charset="utf-8">
     <meta name="viewport" content="width=device-width,
initial-scale=1.0">

   </head>
   <body>

     <div id="main">
     </div>

     <script>
     for (var i=0; i<3; i++) {
       console.log('testing '+i);
     }
     </script>
   </body>
</html>
```

9. 在浏览器中打开后，我们可以看到图 7-21 所示的结果。

图 7-21

7.6.3 回顾

当你点击调试工具，它会显示如图 7-22 所示的详细信息。

图 7-22

以下代码就是用于创建这些调试信息的：

```
<script>
for (var i=0; i<3; i++) {
  console.log('testing '+i);
}
</script>
```

第 8 章
服务器端性能调优

本章内容包括：

- 防止移动设备转码
- 添加移动设备支持的 MIME 类型
- 正确显示 cache manifest 文件
- 在头文件中设置未来过期时间
- 使用 Gzip 压缩
- 移除 ETags

8.1 简介

对于 Web 应用来说，服务器端的性能、配置等对应用页面的加载速率有直接的影响，一个适当的服务器端配置，可以大幅提高客户端的应用加载速度。

在本章中，我们将为大家介绍一些移动 Web 应用的服务器端配置技巧。这些技巧能够提高移动 Web 应用的加载速度和用户体验。其中的部分技巧仅适用于移动 Web 应用，部分技巧对桌面 Web 应用也同样适用。

在互联网上有很多关于服务器端调优的建议或实践，其中的一些实践可能比较难理解，在本章中，我们将从这些实践中挑出几个最好的实践来为大家介绍如何从服务器端优化一个移动 Web 应用。

8.2 防止移动设备转码（Mobile Transcoding）

适用设备：基于浏览器的所有设备

许多移动运营商会使用代理或者自适应引擎来改变你请求的网页的内容，在很多移动设备上自带的浏览器或者安装的其他浏览器都有使用内置转码器来重构整个页面的结构和压缩页面内容。这就是我们常说的**移动设备转码（Mobile Transcoding）**。如果你希望运营商或者浏览器不要做移动设备转码，你需要在 Http 请求的头文件中增加一些内容告诉服务器和浏览器不要做转码。

8.2.1 准备

在 Apache 服务器上，通常使用 .htaccess 配置文件来管理相关目录下的网页配置。还有一种做法是通过编辑 httpd.conf 文件来管理这些配置，某些服务器托管公司不允许客户访问安装 Apache 的服务器的根目录。因此，在本小节的例子中，我们会使用.htaccesss 配置文件来实现防止移动设备转码的功能。另外，.htaccess 文件使用起来更为方便，如果使用 http.conf 文件的话，每一次改动都需要重启服务器。接下来，创建或者打开已经存在的.htaccess 文件。

8.2.2 实践

在前面创建的.htaccess 文件中写入如下代码：

```
<FilesMatch "\.(php|cgi|pl)$">
Header append Cache-Control "no-transform"
Header append Vary "User-Agent, Accept"
</FilesMatch>
```

把.htaccess 文件和你想要控制转码的页面放置到同一个文件目录下。通过这个配置，以后再有移动设备访问该服务器的时候，该服务器就不会再做转码了。

8.2.3 回顾

FilesMatch 标签中的正则表达式是用来指定对那些文件运用标签内的规则，例子中就只会对 CGI 和 PHP 文件使用该规则，其他文件不受影响。

```
<FilesMatch "\.(php|cgi|pl)$">
```

Apache 提供了可以控制和修改请求和响应头文件的模块 mod_headers，在启用该模块之后，我们就可以在 FilesMatch 标签中，添加如下的规则，为对应的请求和响应添加一个信息，告诉服务器该文件不需要做移动设备转码：

```
Header append Cache-Control "no-transform"
```

8.2.4 延伸

下面的这些资源可以帮助大家更深地了解移动设备转码：

Microsoft Internet Information Server（IIS）

如果你使用的是微软的 IIS 服务器的话，你可以使用 IIS 自带的 UI 界面配置，具体的做法请移步：

http://mobiforge.com/developing/story/setting-http-headers-advise-transcoding-proxies

响应式重组

下面的这篇文章将为大家介绍一些关于网络运营商为移动设备做内容转码对 Web 应用的影响：

http://mobiforge.com/developing/blog/responsible-reformatting

MBP-Mobile Boilerplate

本小节示例中的代码已收录在 Mobile Boilerplate 项目的源代码中：

https://github.com/h5bp/mobile-boilerplate/blob/master/.htaccess

8.3 添加移动设备支持的 MIME 类型

适用设备：BlackBerry、Symbian

BlackBerry 和 Nokia 的浏览器支持一些特殊的数据类型，这些数据类型可能无法被服务器识别。在本小节中，我们将为大家介绍其中的几种 MIME 类型，并介绍如何把这几种 MIME 类型的配置添加到服务器配置文件中，这样服务器就不会识别不了这些数据类型了。

8.3.1 准备

同样的，该配置的实现我们还是通过.htaccess 文件，创建.htaccess 文件。

8.3.2 实践

在刚创建的.htaccess 文件中写入下面代码：

```
AddType application/x-bb-appworld bbaw
AddType text/vnd.rim.location.xloc xloc
AddType text/x-vcard                    vcf
AddType application/octet-stream sisx
AddType application/vnd.symbian.install sis
```

把.htaccess 文件和需要使用该规则的页面放置到同一个文件目录下。

8.3.3 回顾

我们通过使用 AddTyp 指令让服务器可以识别一些特殊的 MIME 类型，如表 8-1 所示。

表 8-1

指　令	描　述
AddType application/x-bb-appworld bbaw	包含应用在 BlackBerry App World 的应用 ID 的文本文件
AddType text/vnd.rim.location.xloc xloc	包含黑莓设备地图位置信息的文件
AddType text/x-vcard vcf	电子名片的标准格式文件
AddType application/octet-stream sisx	Nokia Symbian 平台下的文件类型
AddType application/vnd.symbian.install sis	Nokia Symbian 平台下的文件类型

8.3.4 延伸

如果你想了解更多黑莓设备支持的文件类型，请访问：

http://docs.blackberry.com/en/developers/deliverables/18169/index.jsp?name=Feature+and+Technical+Overview+-+BlackBerry+Browser6.0&language=English&userType=21&category=BlackBerry+Browser&subCategory=

8.4 正确显示 cache manifest 文件

适用设备：基于浏览器的所有设备

我们在讲第 6 章：“移动富媒体” 的时候讲过，cache manifest 文件主要是用来做离线

应用缓存功能。但是，cache manifest 文件的扩展名可能不能被服务器识别，下面我们就来看看如何让服务器可以正确地识别该文件扩展名。

8.4.1 准备

创建成打开一个已经存在的.htaccess 文件。

8.4.2 实践

在其中加入下面的代码：

```
AddType text/cache-manifest appcache manifest
```

把新的.htaccess 文件和需要使用该规则的页面放置到同一个文件目录下。

8.4.3 回顾

cache manifest 文件的扩展名可能是.appcache，也可能是.manifest。通常使用 AddType 命令把这两种扩展名都添加到 text/cache-manifest 下，这样，就可以保证，不管你使用哪种扩展名都可以被服务器正确识别。

MBP-Mobile Boilerplate

本小节的例子中的代码同样收录在 Mobile Boilerplate 项目源代码中：
https://github.com/h5bp/mobile-boilerplate/blob/master/.htaccess#L75

8.5 在头文件设置未来过期时间

适用设备：基于浏览器的所有设备

为某些文件设置一个未来过期时间可以有效提升网站的性能，因为其能避免很多不必要的 HTTP 请求。尤其是对于那些富媒体应用服务站点，它们有很多的资源文件需要加载，通过为一些文件设置未来过期时间，这些文件未到过期时间之前，客户可以直接从本地加载，无需向服务器请求，从而显著提升应用加载速度。下面的例子中，我们将为大家演示如何为不同的文件类型，基于不同文件的作用，加上不同的未来过期时间。

8.5.1 准备

同样的，打开之前创建的.htaccess 文件。

8.5.2 实践

在刚才创建的 HTML 文件中写入如下代码：

```
<IfModule mod_expires.c>
  ExpiresActive on
  ExpiresDefault                                "access plus 1 month"
  ExpiresByType text/cache-manifest             "access plus 0 seconds"
  ExpiresByType text/html                       "access plus 0 seconds"
  ExpiresByType text/xml                        "access plus 0 seconds"
  ExpiresByType application/xml                 "access plus 0 seconds"
  ExpiresByType application/json                "access plus 0 seconds"
  ExpiresByType application/rss+xml             "access plus 1 hour"
  ExpiresByType image/x-icon                    "access plus 1 week"
  ExpiresByType image/gif                       "access plus 1 month"
  ExpiresByType image/png                       "access plus 1 month"
  ExpiresByType image/jpg                       "access plus 1 month"
  ExpiresByType image/jpeg                      "access plus 1 month"
  ExpiresByType video/ogg                       "access plus 1 month"
  ExpiresByType audio/ogg                       "access plus 1 month"
  ExpiresByType video/mp4                       "access plus 1 month"
  ExpiresByType video/webm                      "access plus 1 month"
  ExpiresByType text/x-component                "access plus 1 month"
  ExpiresByType font/truetype                   "access plus 1 month"
  ExpiresByType font/opentype                   "access plus 1 month"
  ExpiresByType application/x-font-woff         "access plus 1 month"
  ExpiresByType image/svg+xml                   "access plus 1 month"
  ExpiresByType application/vnd.ms-fontobject "access plus 1 month"
  ExpiresByType text/css                        "access plus 1 year"
  ExpiresByType application/javascript          "access plus 1 year"
  ExpiresByType text/javascript                 "access plus 1 year"
  <IfModule mod_headers.c>
    Header append Cache-Control "public"
  </IfModule>
</IfModule>
```

然后，把.htaccess 文件和需要使用该规则的页面放置到同一个文件目录下。

8.5.3 回顾

下面，我就来为大家一一介绍，示例中的代码是如何工作的：

1. 默认文件的过期时间为 1 个月：

```
ExpiresDefault                              "access plus 1 month"
```

2. 在 FireFox 3.6 下，缓存配置文件 cache.appcahe 需要每次都刷新：

```
ExpiresByType text/cache-manifest           "access plus 0 seconds"
```

3. 请求想访问的 HTML 文件不应该缓存：

```
ExpiresByType text/html                     "access plus 0 seconds"
```

4. 请求的数据文件每次也都需要刷新：

```
ExpiresByType text/xml                      "access plus 0 seconds"
ExpiresByType application/xml               "access plus 0 seconds"
ExpiresByType application/json              "access plus 0 seconds"
```

5. RSS 资源文件更新频率比正常的 API 数据频率要低很多：

```
ExpiresByType application/rss+xml           "access plus 1 hour"
```

6. 网站的图标更改的概率很低，因此把其过期时间设置为一周比较恰当：

```
ExpiresByType image/x-icon                  "access plus 1 week"
```

7. 对于那些大文件，像图片、视频或者音频等，我们可以把过期时间设置久一点：

```
ExpiresByType image/gif                     "access plus 1 month"
...
ExpiresByType video/webm                    "access plus 1 month"
```

8. 如果你的应用使用了 HTML5 的 CSS3PIE polyfill 脚本，缓存 HTC 文件也很有必要：

```
ExpiresByType text/x-component              "access plus 1 month"
```

9. 为网页字体设置一个较长的过期的时间也可接受：

```
ExpiresByType font/truetype                 "access plus 1 month"
...
ExpiresByType application/vnd.ms-fontobject "access plus 1 month"
```

10. 对于 CSS 和 JavaScript 脚本，我们可以把过期时间设置到 1 年之后：

```
ExpiresByType text/css                      "access plus 1 year"
ExpiresByType application/javascript        "access plus 1 year"
ExpiresByType text/javascript               "access plus 1 year"
```

8.5.4 延伸

有很多的设置未来过期时间的配置都假设开发人员会使用不同的请求参数来对资源文件做版本控制：

```
<script src="script_034543.js" ></script>
```

另外，我们还需考虑一些代理服务器可能缓存某些文件时未使用版本参数导致缓存不命中的情况：

http://www.stevesouders.com/blog/2008/08/23/revving- ilenames-ont-use-querystring/

添加过期或者缓存控制头文件

在雅虎开发者论坛上，有一篇关于资源过期规则讲得非常好的文章：
http://developer.yahoo.com/performance/rules.html#expires

Mobile Boilerplate

本小节示例中的代码同样收录在 Mobile Boilerplate 项目的源码库中：
https://github.com/h5bp/mobile-boilerplate/blob/master/.htaccess#L142

8.6 使用 Gzip 压缩

适用设备：基于浏览器的所有设备

如何减少 HTTP 请求和响应在网络中传输的时间，前端开发人员在其中扮演着非常重要角色。使用 Gzip 压缩请求和响应文件可以减小需传递的数据的大小，从而减少传输时间。

Gzip 可以显著地减小响应的大小，通常能够减少 70%左右，另外，大部分的浏览器都已经支持 Gzip 了。

对于大多服务器来说，默认情况下只会压缩指定类型的文件，因此，我们需要显式的配置服务器以支持更多的文件类型，如 HTML、XML、JSON 等。

8.6.1 准备

创建或打开一个已经存在的.htaccess 文件。

8.6.2 实践

在刚才创建的文件中写入如下代码：

```
<IfModule mod_deflate.c>
  <IfModule mod_setenvif.c>
    <IfModule mod_headers.c>
      SetEnvIfNoCase ^(Accept-EncodXng|X-cept-
Encoding|X{15}|~{15}|-{15})$ ^((gzip|deflate)\s,?\s(gzip|deflate)?
|X{4,13}|~{4,13}|-{4,13})$ HAVE_Accept-Encoding
      RequestHeader append Accept-Encoding "gzip,deflate" env=HAVE_
Accept-Encoding
    </IfModule>
  </IfModule>

  <IfModule filter_module>
    FilterDeclare   COMPRESS
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $text/html
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $text/css
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $text/
javascript
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $text/plain
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $text/xml
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $text/x-
component
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
javascript
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
json
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
xml
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
x-javascript
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
xhtml+xml
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
rss+xml
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
atom+xml
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
vnd.ms-fontobject
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $image/svg+xml
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $application/
x-font-ttf
    FilterProvider  COMPRESS  DEFLATE resp=Content-Type $font/opentype
```

```
    FilterChain     COMPRESS
    FilterProtocol  COMPRESS  DEFLATE change=yes;byteranges=no
  </IfModule>

  <IfModule !mod_filter.c>
    AddOutputFilterByType DEFLATE text/html text/plain text/css
application/json
    AddOutputFilterByType DEFLATE text/javascript application/
javascript application/x-javascript
    AddOutputFilterByType DEFLATE text/xml application/xml text/x-
component
    AddOutputFilterByType DEFLATE application/xhtml+xml application/
rss+xml application/atom+xml
    AddOutputFilterByType DEFLATE image/svg+xml application/vnd.ms-
fontobject application/x-font-ttf font/opentype
  </IfModule>
</IfModule>
```

然后，把.htaccess 文件和需要使用该规则的页面放置到同一个文件目录下。

8.6.3 回顾

下面的这段代码是为了能够识别被损坏的但是包含浏览器支持的编码类型的头文件，通过 mod_setenvif 模块来支持做正则表达式的匹配，以匹配破坏后的头文件格式。匹配上之后，设置一个环境变量指明被损坏的头文件中包含浏览器支持的编码类型：

```
SetEnvIfNoCase ^(Accept-EncodXng|X-cept-Encoding|X{15}|~{15}|-{15})$
^((gzip|deflate)\s,?\s(gzip|deflate)?|X{4,13}|~{4,13}|-{4,13})$ HAVE_
Accept-Encoding
```

修改请求头文件以支持压缩就比较容易理解了：

```
RequestHeader append Accept-Encoding "gzip,deflate" env=HAVE_Accept-
Encoding
```

下面是关于压缩哪些文件类型的配置，有 HTML、TXT、CSS、JavaScript、JSON、XML、HTC：

```
<IfModule filter_module>
  FilterDeclare   COMPRESS
  ...
  FilterProtocol  COMPRESS  DEFLATE change=yes;byteranges=no
</IfModule>
```

为了支持 Apache 2.1 以前的版本，需添加如下代码：

```
<IfModule !mod_filter.c>
    AddOutputFilterByType DEFLATE text/html text/plain text/css
application/json
    AddOutputFilterByType DEFLATE text/javascript application/
javascript application/x-javascript
    AddOutputFilterByType DEFLATE text/xml application/xml text/x-
component
    AddOutputFilterByType DEFLATE application/xhtml+xml application/
rss+xml application/atom+xml
    AddOutputFilterByType DEFLATE image/svg+xml application/vnd.ms-
fontobject application/x-font-ttf font/opentype
</IfModule>
```

8.6.4 延伸

这儿需要注意的一点是图片和 PDF 文档不需要做 gzip 压缩，因为他们默认已经压缩过了，压缩它们只是空耗 CPU 而已，甚至有可能会增加文件的大小。

主动使用 Gzip 压缩

在雅虎开发者论坛上，Marcel Duran 发表了一篇关于 gzipping 的文章，里面谈了很多关于在服务端配置 gzipping 的最新研究：

http://developer.yahoo.com/blogs/ydn/posts/2010/12/pushing-beyond-gzipping/

8.7 移除 ETags

适用设备：基于浏览器的所有设备

ETags 指的是资源标签（Entity tags[①]），在互联网中，一个资源表示的是一个网络组件，比方说一个 CSS 文件或者一个 JavaScript 文件等等。一个资源标签的作用就是标明一个网络组件的某个特定版本。关于 ETags 的配置，在雅虎开发者论坛上这篇文章讲得很好：高性能网站：规则 13-配置 Etags：（(http://developer.yahoo.com/blogs/ydn/posts/2007/07/high_ performanc_11/)。

如果你的应用后台使用了多个服务器托管，在服务器和客户端的数据传递网络中，通过 ETag 验证机制来减少数据传输的策略可能不会那么有效，因为每一台服务器会为同一个资源生成不同的 ETag，因此服务器可能错误的判断某个文件被更新了，导致很多不必要重

[①] 译者注：Entity Tags (ETags) 是 HTTP/1.1 的新特性。在 Conditional GET Requests 中起作用，是一个标记文件特定版本的标记，是除了最后修改时间以外，判断组件是否变化的另一种方式。

新请求。解决这个问题的最好方法就是移除资源文件的 ETag。

8.7.1 准备

创建或打开一个已存在.htaccess 文件。

8.7.2 实践

在刚才创建的文件中写入如下代码：

```
<IfModule mod_headers.c>
  Header unset Etag
</IfModule>

FileETag None
```

然后，把.htaccess 文件和需要使用该规则的页面放置到同一个文件目录下。

8.7.3 回顾

首先，使用 unset 指令，告诉服务器不要给头文件添加 ETag：

```
<IfModule mod_headers.c>
  Header unset Etag
</IfModule>
```

然后，使用 FileETag None 语句确保所有的请求、响应都不包含 ETag：

```
FileETag None
```

8.7.4 延伸

下面为大家介绍一些关于 ETag 的资源。

同步 IIS 服务器上的 ETag 的值

如果你的应用是运行在 IIS 服务器上的，那么，你可以通过同步所有服务器上同一个资源的 ETag 值来解决我们前面提到的 ETag 验证机制导致的问题。在多个服务器构建的 Web Farm[①]中，我们可以使用 Mdutil.exe 获取其中某个服务器的 ETag 值，然后，在其他的

[①] 译者注：Web Farm 就是多个 Web 服务器上部署同一 Web 应用程序来均衡负载。

服务器上同样使用该值做 ETag 值。

关于这个问题，你可以在微软的技术支持网站得到更多的细节信息：

```
http://support.microsoft.com/?id=922733
```

构建高性能网站

Steve Souders 在他的构建高性能 Web 站点的系列文章中，详细地解释过关于 ETag 的配置：

```
http://developer.yahoo.com/blogs/ydn/posts/2007/07/high_performanc_11/
```

David Walsh 的博客

David Walsh 的博客上发表过一篇由 Eric Wendelin 写的文章：如何通过.htaccess 提升你的网站的 YSlow[①]得分，这篇文章中同样提到了本小节我们讲的问题：

```
http://davidwalsh.name/yslow-htaccess
```

MBP – Mobile Boilerplate

同样的，本小节示例中的移除 ETag 的配置代码也收录在 Mobile Boilerplate 项目的源代码中：

```
https://github.com/h5bp/mobile-boilerplate/blob/master/.htaccess#L211-L218
```

[①] 译者注：YSlow 是由雅虎公司开发的，开源的 Web 页面性能分析、测试和调试工具（http://developer.yahoo.com/yslow/）。

第 9 章
移动性能测试

本章内容：

- 使用 Blaze 测试移动设备
- 在线分析移动页面速度
- PCAP 网站性能分析
- 移动版 HTTP Archive
- 使用 Jdrop 存储性能数据

9.1 简介

本章将介绍一些热门的移动性能测试工具。

和移动调试一样，移动性能测试并不像桌面测试那么简单，但是基于无处不在的云计算，我们可以得到解决方案，程序员们创造出了许多方法来解决各种问题。

9.2 使用 Blaze 的移动设备速度测试

适用浏览器：所有

如果你希望方便快捷的测试移动网站的性能，包括加载时间和页面资源信息，那么 Blaze 是一个很好的选择。Mobitest Performance Tool 可以用于了解移动网站性能，它提供了以下测试结果：

- 整体加载时间
- 单独页面资源的加载错误

- 显示视频
- Raw HTTP Archive (HAR) 文件

9.2.1 准备

你只需要登录 `http://www.blaze.io/mobile/`。

9.2.2 实践

图 9-1

在图 9-1 所示的页面有以下表单接受一个 URL，我们先输入 yahoo.com 来测试。

在结果页面的顶部，如图 9-2 所示，我们可以看到网站的平均加载时间、页面大小和速度评级。

图 9-2

图 9-3 显示了网站加载的瀑布图。

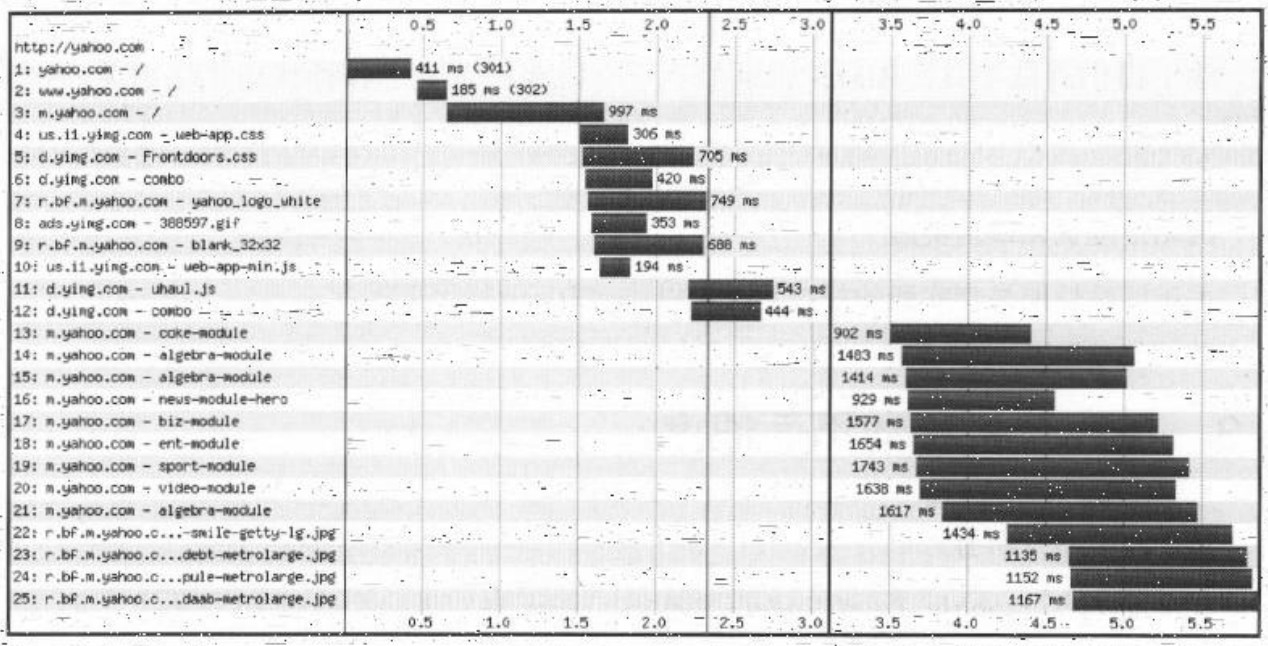


图 9-3

9.2.3 回顾

使用的设备

也许你想知道测试所使用的设备是什么，它是模拟器还是仿真器？所有的测试都运行在真实的定制设备上。

加载时间排名百分比

该工具有一个内部的索引来计算加载时间的排名百分比，它参考并比较了上百个网站的速度。

测试运行的地理位置

测试运行的地理位置在渥太华、加拿大，测试代理使用 WiFi 连接互联网。在本书发布前，测试设备为：iPhone、Nexus 和三星 Galaxy S。

更多信息请访问：

`http://www.blaze.io/mobile/methodology/`

9.2.4 延伸

一个实用的测试工具列表可以在以下链接中找到：

http://www.blaze.io/learn/feo-resources/

Blaze 博客

除了提供测试工具，Blaze 还维护了一个博客，有关于各种移动优化相关的优秀文章：

http://www.blaze.io/blog/

网站性能优化最佳实践

关于网站性能最佳实践的好技巧，可以访问 Blaze 优化页面：

http://www.blaze.io/overview/optimizations/

9.3 在线分析移动页面速度

适用浏览器：所有

如果你对于 Google Page Speed 很熟悉，你应该知道一个在桌面浏览器上测试加载速度的 Chrome 插件，同时也有一个网页版的 Google Page Speed 可以测试移动端的性能。

9.3.1 准备

访问 Google Page Speed：

http://pagespeed.googlelabs.com/

9.3.2 实践

本例中，我们将测试 Google 移动版主页：

1. 输入分析对象的 URL，本例中使用 m.google.com，如图 9-4 所示。

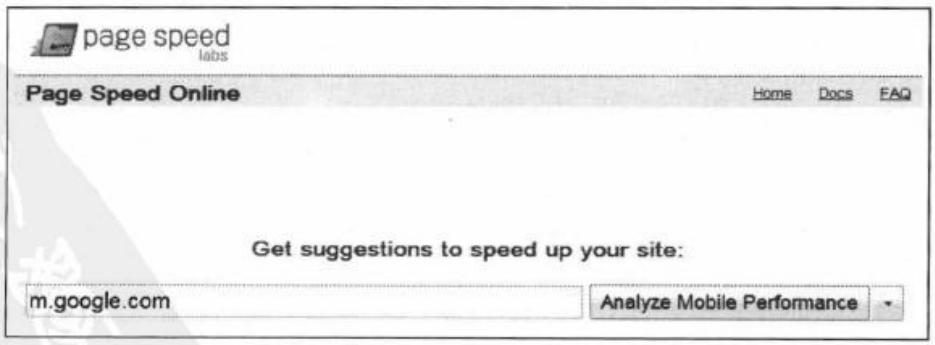


图 9-4

2. 打开输入框旁边的下拉列表，从中选择 Get mobile suggestions，如图 9-5 所示。

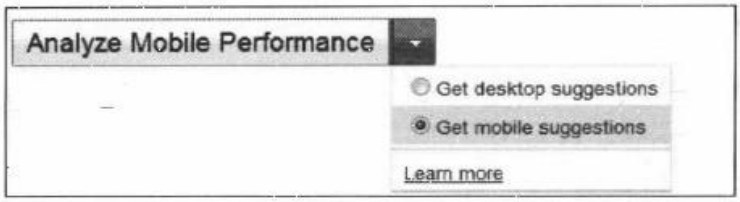


图 9-5

3. 点击 Analyze Mobile Performance 后，页面显示如图 9-6 所示。

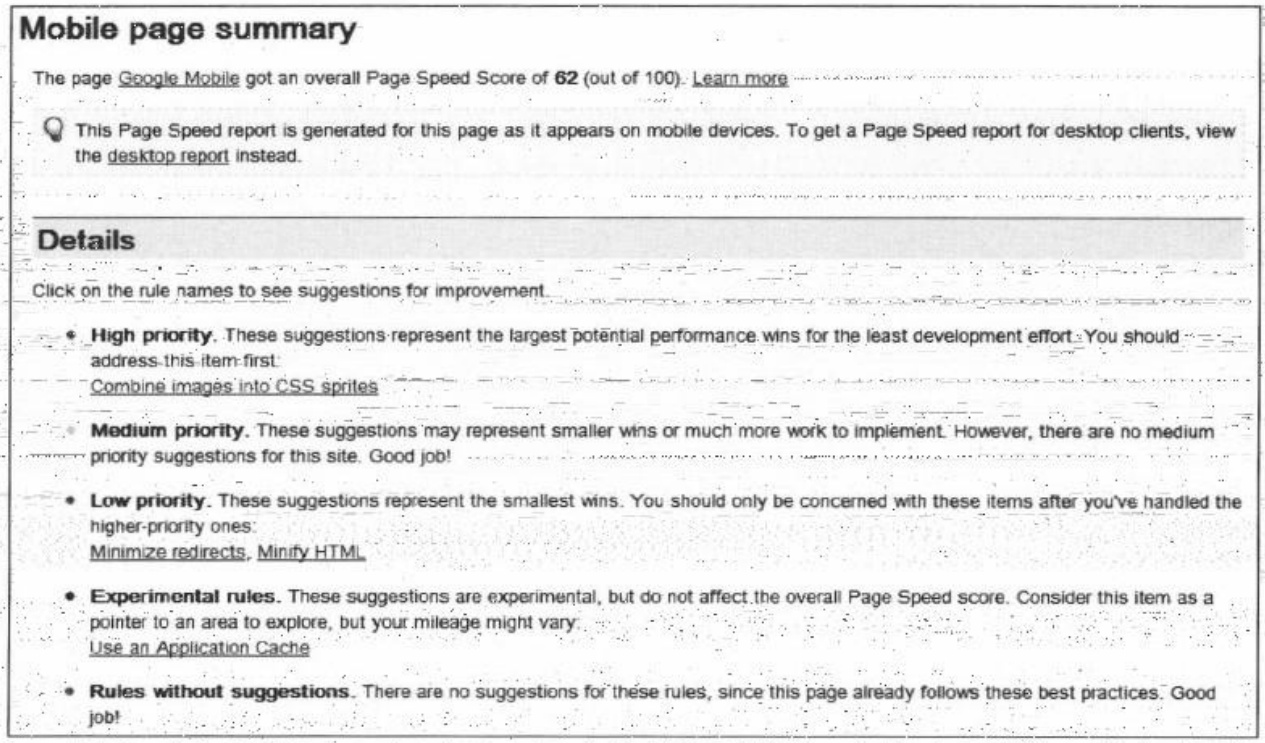


图 9-6

9.3.3 回顾

Page Speed Score 显示了页面加载的速度，本例中的分数为 62（满分 100）。

下面是详细的分析，该分析被分为以下几个部分：

- 高优先级：这些建议代表着用最小的开发成本可能获取到最大的性能提升，应该首先从这些建议着手优化。
- 中优先级：这些建议可能代表着较小的性能提升，或是较大的开发成本。

- 低优先级：这些建议代表着最小的性能提升。
- 尝试性：这些建议都是尝试性的，不会影响 Page Speed Score。
- 无建议：由于页面已经完成了所有最佳实践，所以没有更多的建议，但仍可以打开左边的折叠菜单，查看所有的最佳实践。

9.3.4 延伸

移动性能工具列表：

https://github.com/h5bp/mobile-boilerplate/wiki/Mobile-Performance-Tools

页面速度的重要性

麻省理工的技术学报展示了一些图标和统计数据，关于页面速度的重要性以及它如何影响网站的访问者，该文章提到即使是很小的延迟，也会使人们感到不适并且增大公司开销：

http://www.technologyreview.com/files/54902/GoogleSpeed_charts.pdf

当时间开始计算

Gomez 公司统计了一个对于桌面、移动网站性能期望的全球用户调查表：

http://www.gomez.com/wp-content/downloads/GomezWebSpeedSurvey.pdf

9.4 PCAP 网站性能分析

适用浏览器：所有

PCAP 网站性能分析器针对数据分析提供了很好的控制能力，你可以与移动网站或应用交互并获取更加精确的性能数据，它由 Bryan McQuade 和 Libo Song 创建。

9.4.1 准备

在使用 PCAP 网站性能分析器前，我们需要首先获取移动设备的 PCAP 文件：设置一个私有的 WiFi 网络，将移动设备连接至该网络，截获并分析数据传输。以下是详细说明[①]：

1. 打开控制面板 | 网络和 Internet | 网络和共享中心。

[①] 该说明基于 Window 7 操作系统——译者注。

2. 点击链接设置新的连接或网络。

3. 点击设置无线临时（计算机到计算机）网络。

4. 然后，输入网络名（例如 hot1），勾选保存这个网络。

5. 回到网络和共享中心，点击左边的更改适配器设置。

6. 找到你的局域网，右键点击打开属性 | 共享标签。

7. 启用网络共享。

现在我们必须下载 Wireshark，以便在指定的网络中截获传输数据。我们可以按照以下步骤生成 HAR 文件并存放在本地的电脑上：

1. 下载 Wireshark：`http://www.wireshark.org/download.html`。

2. 打开 Wireshark。

3. 点击 Menu Capture | Options。

4. 在 Options 对话框中，选择你的无线接口，点击 Capture Filter。

5. 在 Capture Filter 对话框中，创建一个新的过滤器（如果你还没有创建过的话），并设置名称为 TCP and UDP port 53 (DNS)，同时设置过滤器字符为 tcp or udp port 53。

6. 选择该过滤器，并关闭对话框。

7. 点击 Capture Options 对话框中的 Start 按钮来开始截获数据传输。

8. 完成后保存截获到的数据。

要将移动设备接入热点，可以将你的移动设备接入某个 WiFi 热点（本例中为“hot1”），现在，在移动设备上访问的任何网站都会被 tcpdump[①]截获。

9.4.2 实践

在性能分析器页面，选择保存的 HAR 文件并点击 Upload，该文件处理完成后，将会显示一个基于瀑布图的详细分析报告，如图 9-3 所示。

①tcpdump 是一个运行在命令行下的嗅探工具。它允许用户拦截和显示发送或收到过网络连接到该计算机的 TCP/IP 和其他数据包。tcpdump 是一个在 BSD 许可下发表的自由软件。——译者摘自维基百科

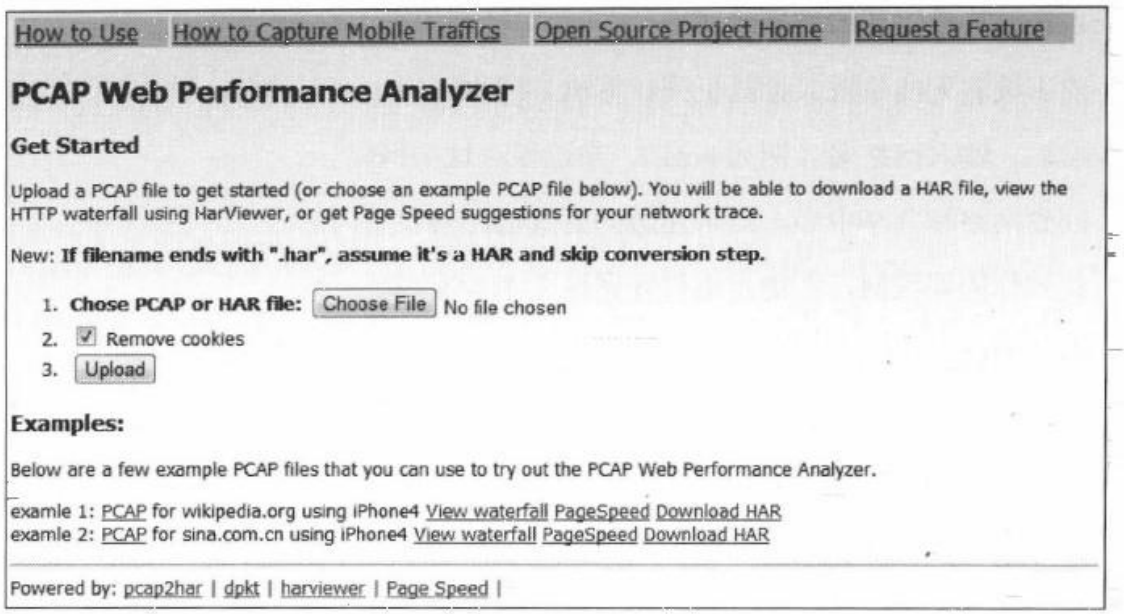


图 9-7

9.4.3 回顾

我们使用了开源文件格式 PCAP 和 HAR，以及开源工具 pcap2har、HAR viewer 和 Page Speed。

9.4.4 延伸

Stoyan Stefanov 维护着一个关于网站和移动网站性能的非常有用的网站：

```
http://calendar.perfplanet.com/2010/mobile-performance-analysis-using-pcapperf/
```

HAR viewer

如同官方所描述的，HAR Viewer 是一个网页应用（PHP+JavaScript），基于 HTTP Archive format（HAR），它可以将 HTTP 跟踪日志图形化。该项目放在 Google Code 上，你可以在此查看代码：`http://code.google.com/p/harviewer/`。

使用 Page Speed 优化你的网站或移动网站

这里有一个关于如何使用 Google 的 Page Speed 的视频，该视频摄制于 2011 年 Google I/O 大会，由 PACPPERF 的创始者主讲：

```
http://www.google.com/events/io/2011/sessions/use-page-speed-to-optimize-your-web-site-for-mobile.html
```

pcap2har

要更多地了解 pcap2har，你可以访问它在 Github 的主页：

`https://github.com/andrewf/pcap2har`

9.5 移动版 HTTP Archive

适用浏览器：所有

移动版 HTTP Archive 追踪网站是如何建立的，它提供了：

- 网站技术的趋势：加载时间、下载量、性能指数。
- 有趣的统计：流行的脚本、图片格式、错误、跳转。
- 网站性能：某个 URL 的截屏、瀑布流图表、HTTP 头文件。

9.5.1 准备

进入 `http://mobile.httparchive.org/`。

9.5.2 实践

点击 Trends 可以看到许多趋势，例如传输量和 HTML、JavaScript、CSS、图片以及 Flash 的请求数，图 9-8 就是 HTML 传输量和请求数的图表：

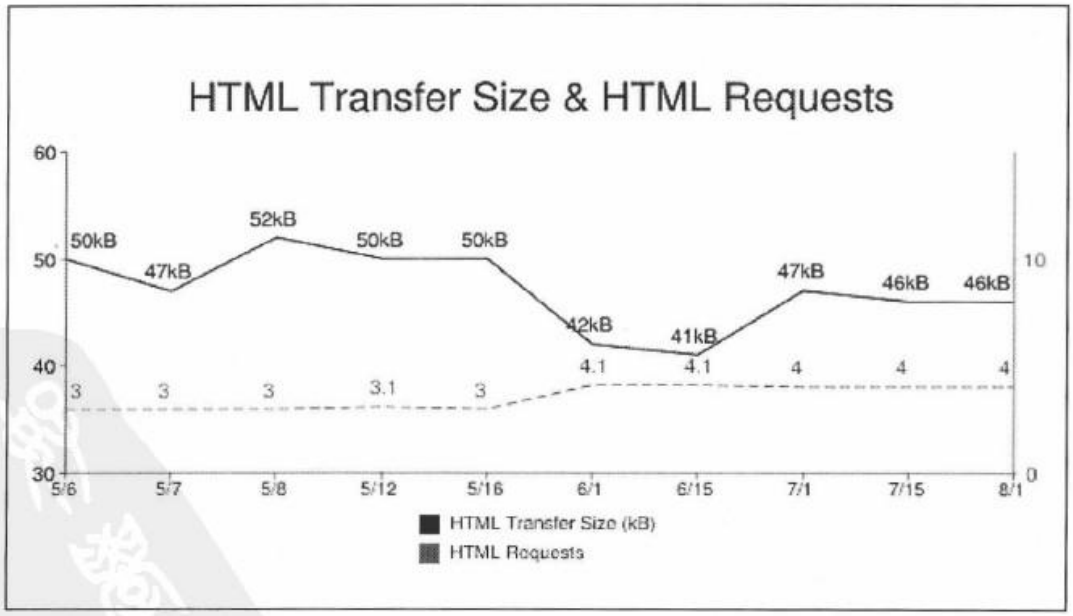


图 9-8

点击 Stats 可以看到许多有趣的统计，从最常用的图片格式到最常用的服务器，从拥有最多 CSS 的页面到拥有最多图片的页面。

图 9-9 显示了最常用的 JS 库：

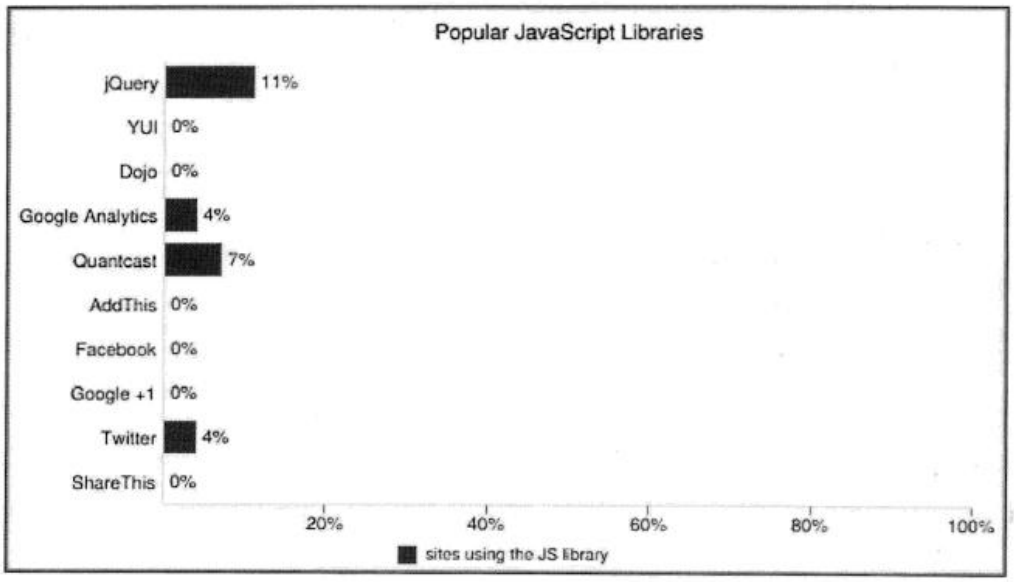


图 9-9

点击 Websites 可以得到某个网站的所有性能信息，包括了加载过程截图、瀑布图、页面速度、请求数、趋势以及 HAR 文件的下载。

9.5.3 回顾

所有列出的网站都是在 Alexa、Fortune、Global 500、Quancast10K 排名靠前的。

URL 列表将用于 WebPagetest.org。

HTTP 瀑布图表使用 JavaScript 根据 HAR 文件生成。

9.5.4 延伸

你可能会问为什么需要记录数据，这样做是因为它对于归档网站性能的历史并从中学习是非常重要的。如同 Steve Souders 所说的，“HTTP Archive 提供了这些记录，它是网站性能信息的永久仓库，包括了页面大小、失败的请求和使用的技术。”（摘自 `http://www.stevesouders.com/blog/2011/03/30/announcing-the-http-archive/`）。

数据有多精确？

如果你想知道数据的精确度，可以阅读 http://mobile. httparchive.org/about.php#accuracy，还介绍了关于度量，特别是时间度量的信息。

测试方法论的局限。

虽然测试结果非常有用，但是它还是有一些需要用户注意的局限：
http://mobile.httparchive.org/about.php#limitations

9.6 使用 Jdrop 存储性能数据

适用浏览器：所有

Jdrop 用于存储移动设备的性能数据，所有的数据都以 JSON 格式存储在云端。

由于移动设备屏幕很小，使得大量的数据难以分析，为了解决这个问题，Jdrop 允许你在大屏幕上分析从移动设备获取的数据。

9.6.1 准备

登录 Jdrop: http://jdrop.org/。

9.6.2 实践

在移动设备中：

1. 登录 Jdrop。
2. 安装 Jdrop 应用。
3. 运行应用并将数据保存至 Jdrop。

桌面电脑或笔记本：

1. 登录 Jdrop。
2. 查看搜集的 JSON 数据。

在移动设备，最简单的入门方式是将 jdrop-example.js（http: // jdrop. org/jdrop-example.js）的内容嵌入书签脚本。另外，你还需要在书签栏添加一个*保存到 Jdrop* 的链接。你可以在这里看到代码和解释：http://jdrop.org/devdocs。

保存至 Jdrop：

以下的代码就是你需要添加至书签的*保存到 Jdrop* 的链接：

```
<a href="javascript:SaveToJdrop('MY APP NAME', myDataObj, '1.1.3',
'1.8 secs')">Save to Jdrop</a>
```

注册你的应用：

注册你的应用需要在 Jdrop 讨论列表发起一个申请（http: // groups . google . com/group/jdrop/topics）。

以下是注册应用所需的信息：

- **应用名**（必要）
- **脚本 URL**（必要）
- **回调方法**（可选）
- **格式**（可选）
- **格式键名**（可选）

本书出版以后一些信息可能已经改变了，你可以进入 http://jdrop.org/devdocs 来检查任何变更。

9.6.3 回顾

你也许想知道为什么在使用 Google 账号登录时，Jdrop 要求 Google 联系人的权限。这是因为使用 OAuth 连接 Google，必须指定某个服务才能验证登陆，Jdrop 实际上不会获取任何联系人信息。创始人正在考虑使用 OpenID 来替换 OAuth 来跳过这一步。

9.6.4 延伸

Jdrop 由 Steve Souders 和 James Pearce 创建。

Steve Souders 对于大多数开发者来说并不陌生，在这里可以看到他出色的成果：

http://stevesouders.com/

James Pearce 是 Sencha 公司的开发者关系总监，你可以在他的网站上找到很多有趣的点子和关于移动的有用的信息：

http://tripleodeon.com/

第 10 章
拥抱移动互联网特性

本章内容包括：

- window.onerror
- 使用 ECMAScript 5 中的新方法
- HTML5 中新的输入类型
- 在 HTML 中内嵌 SVG
- position:fixed
- overflow:scroll

10.1 简介

iOS 5 上的 Safari 引入了一系列的改进，这些改进让 Safari 成为了最高端的移动浏览器之一。其支持很多前沿的 HTML5 特性，支持 ECMAScript 5 标准以及很多专门为移动设备而生的特性。通过合理地使用这些特性，可以帮助开发人员构建功能更加完善、性能更好、用户体验更佳的移动 Web 应用：

- **网页表单（Web forms）**可以帮助开发人员构建更加友好的用户界面，更为便捷地设计出应用原型界面。
- **内嵌 SVG**[1]可以让图像在移动浏览器中无损地放大、缩小。这个特性对响应式设

[1] SVG：可伸缩矢量图形（Scalable Vector Graphics），要用来定义用于网络的、基于矢量的图形，其最大特点是图像在放大或者改变尺寸的情况下，图像的质量不会损失。

计非常有帮助。

- **ECMAScript 5** 允许开发人员更好的控制对象的生命周期，这样我们就可以通过 JavaScript 构建大型的、复杂的应用。
- **专为移动设备而生的属性**：在以前版本的 Safari 上，在 Web 应用中想要做出原生应用的那种滚动效果是非常难的。现在，新版本的 Safari 引入了一些专为移动设备而生的属性，这些属性值可以帮助网络开发人员更容易把 Web 应用的性能和体验做得和原生应用一样。

10.2　window.onerror

适用设备：iOS 5

在 iOS 5 中，新加入了一个事件处理器：`window.onerror`。该事件处理器会处理所有发送给当前窗口的事件。其使用方式如下：

```
window.onerror = funcA;
```

10.2.1　准备

创建一个名为“ch10r01.html”的 HTML 文件。

10.2.2　实践

在刚创建的文件中写入如下代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <style>
    </style>
  </head>
  <body>
    <script>
    window.onerror=function(){
      alert('An error has occurred!')
    }
    </script>
```

```
    <script>
    document.write('hello world'
    </script>
  </body>
</html>
```

在 iOS 5 设备浏览器中打开该文件，你会看到一个警告信息弹出。

10.2.3 回顾

在示例中，因为在语句 document.write 不完整，会发生一个错误：

```
<script>
  document.write('hello world'
</script>
```

如果我们补上闭括号，重新打开该文件，你就不会再看到报错的信息了：

```
<script>
  document.write('hello world');
</script>
```

10.2.4 延伸

浏览器默认对错误消息的处理是什么都不做，示例中的代码是覆盖了浏览器的默认行为：

```
window.onerror = null;
```

浏览器对象模型：

所谓浏览器对象模型（**Browser Object Model**），就是一组用于控制浏览器和设配屏幕的对象。所有的这些对象都可以通过全局对象 window 和 window.screen 获取到，如果你想了解更多的关于浏览器对象模型的信息，请移步：

http://javascript.about.com/od/browserobjectmodel/Browser_Object_Model.htm

10.3 使用 ECMAScript 5 中的新方法

适用设备：iOS 5

ECMAScript 5 是用于替代 ECMAScript 3.1 标准的。ECMAScript 5 在对象的交互方面改进了很多。苹果自从 iOS 4 起，在移动版的 Safari 中引入了很多的新的 ECMAScript 5 特

性，在iOS 5中，对ECMAScript 5的支持度就更高了。

iOS 5新引入的ECMAScript 5对象方法有：

```
Object.seal/Object.isSealed
Object.freeze/Object.isFrozen
Object.preventExtensions/Object.isExtensible
Function.prototype.bind
```

10.3.1 准备

创建一个名为“ch10r02.html”的HTML文件。

10.3.2 实践

在刚创建的HTML文件中写入下面代码，并在iOS 5设备中打开该文件：

```
/*** freeze ***/

var dog = {
  eat: function () {},
  hair: "black"
};
var o = Object.freeze(dog);

// test if dog is frozen
assert(Object.isFrozen(dog) === true);

// can't alter the property
dog.hair = "yellow";

// can't remove property
delete dog.hair;

// can't add new property
dog.height = "0.5m";

/*** seal ***/

var human = {
  eat: function () {},
  hair: "black"
};
human.hair = "blonde";
var o = Object.seal(obj);
// changing property works
```

```
human.hair = "grey";
// can't convert
Object.defineProperty(obj, "hair", { get: function() { return "green";
} });
// silently doesn't add the property
human.height = "1.80m";
// silently doesn't delete the property
delete human.hair;
// detect if an object is sealed
assert(Object.isSealed(human) === true);

/*** preventExtensions ***/

var nonExtensible = { removable: true };
Object.preventExtensions(nonExtensible);
Object.defineProperty(nonExtensible, "new", { value: 8675309 });
// throws a TypeError
assert(Object.isExtensible(nonExtensible) === true);

/*** bind ***/

var x = 9;
var module = {
  x: 81,
  getX: function() { return this.x; }
};

module.getX(); // 81

var getX = module.getX;
getX(); // 9, because in this case, "this" refers to the global object

// create a new function with 'this' bound to module
var boundGetX = getX.bind(module);
boundGetX(); // 81
```

10.3.3 回顾

Freeze

方法作用如其名，freeze 方法的作用就是冻结一个对象，对象被冻结之后就再也不能添加或移除任何内容，甚至不能修改已有的任何内容，该方法把一个对象设置为不可更改的，然后返回被冻结过的对象：

```
// can't alter the property
dog.hair = "yellow";
```

```
// can't remove property
delete dog.hair;

// can't add new property
dog.height = "0.5m";
```

如果你想测试一个对象是不是已经被冻结了，可以使用方法 isFrozen：

```
// test if dog is frozen
assert(Object.isFrozen(dog) === true);

// silently doesn't add the property
human.height = "1.80m";

// silently doesn't delete the property
delete human.hair;
```

如果你 seal 了一个对象，那么该对象就不再新增或者移除属性了，其和 freeze 的区别是：freeze 之后，我们除了不能增加、删除属性外，还不能更改现有属性的值，但是 seal 是可以修改已有属性的值的：

```
// changing property works
human.hair = "grey";

// detect if an object is sealed
assert(Object.isSealed(human) === true);
```

默认情况下，一个对象是可以扩展的，但是如果你使用了 preventExtensions 方法，我们可以防止一个对象扩展，也就是说，该对象不能增加任何新的属性。

```
/*** preventExtensions ***/

var nonExtensible = { removable: true };
Object.preventExtensions(nonExtensible);
Object.defineProperty(nonExtensible, "new", { value: 8675309 });
// throws a TypeError
assert(Object.isExtensible(nonExtensible) === true);
```

Function.prototype.bind

另一个必须介绍的，非常有用的特性就是 bind 方法。简单来说就是通过 this 来指明你想要访问的具体是哪个对象。在我们的示例中，不管函数被如何调用，该函数执行的过程中都会有一个指代当前对象的 this。

下面的这段代码中，有一个全局变量 x，它的值会在 module 对象中被修改：

```
var x = 9;
var module = {
  x: 81,
  getX: function() { return this.x; }
};

module.getX(); // 81
```

当我们在 module 中提取到 getX 方法，在后面的逻辑中，调用到该函数，我们希望它可以使用初始的 module 对象作为 this，但是这个时候，程序运行在主程序上下文中，此时的 this 指代的是全局对象 global，所以，getX 方法会返回 9。

```
var getX = module.getX;
getX(); // 9, 这个时候 this 指代的是全局对象 global
```

通过使用 bind，我们可以创建一个新的方法，把该方法中的 this 绑定到 module 对象上：

```
// create a new function with 'this' bound to module
var boundGetX = getX.bind(module);
boundGetX(); // 81
```

10.3.4 延伸

```
window.onerror = null;
```

MDN（Mozilla Developer Network）上的文档

Object.freeze/Object.isFrozen:

- https://developer.mozilla.org/en/JavaScript/Reference/Global_Objects/Object/freeze
- https://developer.mozilla.org/en/JavaScript/Reference/ Global_Objects/Object/isFrozen

Object.seal/Object.isSealed:

- https://developer.mozilla.org/en/JavaScript/Reference/Global_Objects/Object/seal
- https://developer.mozilla.org/en/JavaScript/Reference/ Global_

Objects/Object/isSealed

preventExtensions/isExtensible:

- https://developer.mozilla.org/en/JavaScript/Reference/Global_Objects/Object/preventExtensions
- https://developer.mozilla.org/en/JavaScript/Reference/ Global_Objects/Object/isExtensible

Function.prototype.bind:

- https://developer.mozilla.org/en/JavaScript/Reference/Global_Objects/Function/bind

10.4 HTML5 中新的输入类型

适用设备： iOS 5

新的输入类型对我们构建 Web 表单有很大帮助，现在 iOS 5 支持的新的输入类型有：date、datetime、month、time、range 等。

10.4.1 准备

创建一个名为“ch10r03.html”的 HTML 文件。

10.4.2 实践

在刚创建的 HTML 文件中写入下面的代码，然后在 iOS 5 设备中访问该文件：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">

  </head>
  <body>
    <input type="date">
    <input type="datetime">
    <input type="month">
    <input type="time">
```

```
    <input type="range">
  </body>
</html>
```

10.4.3 回顾

在 iOS 5 中，date 和 datetime 类型渲染效果如图 10-1 所示。

图 10-1

而 month 和 time 输入类型会被渲染成如图 10-2 所示的样子。

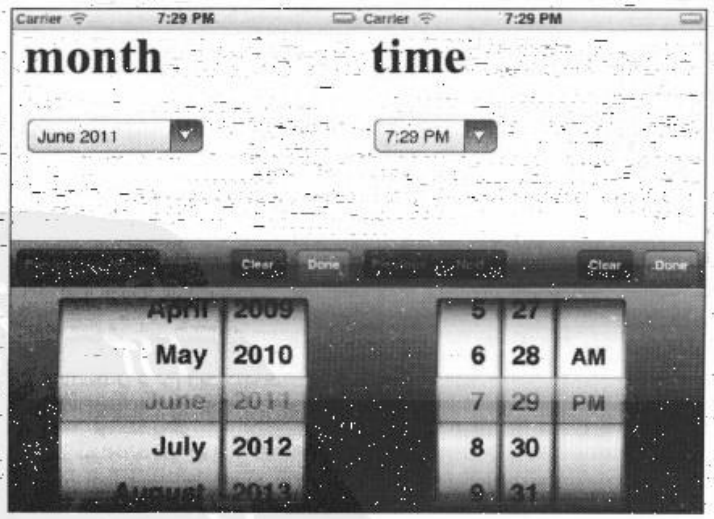


图 10-2

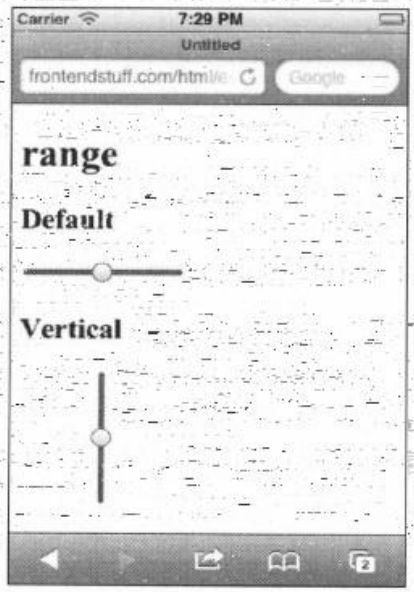


图 10-3

slider 输入类型渲染得到的结果如图 10-3 所示。

10.4.4 延伸

网络上有很多 polyfills 脚本可以让使用这些新的输入类型的网络表单可以跨浏览器运行。html5slider 就是一个为 Firefox 4 及以上版本实现 HTML5 的 range 输入类型的 polyfills 脚本。你可以在这儿了解更多它的信息：

https://github.com/fryn/html5slider

10.5 在 HTML 中内嵌 SVG

适用设备：iOS 5

通过对内嵌可缩放矢量图形（**Scalable Vector Graphics**）的支持，我们就可以在 HTML 页面中直接使用 SVG。

10.5.1 准备

创建一个名为“ch10r04.html”的 HTML 文件。

10.5.2 实践

在刚才创建的 HTML 文件中写入如下代码：

```
<svg width="500" height="220" xmlns="http://www.w3.org/2000/svg"
version="1.1">
  <rect x="2" y="2" width="496" height="216" stroke="#000"
stroke-width="2px" fill="transparent"></rect>
</svg>
```

10.5.3 回顾

HTML 内嵌 SVG 的文件最后会使用“Content-Type: text/xml”的 MIME 类型渲染，因此你可以直接以“.xml”代替“.html”作为文件扩展名。

10.5.4 延伸

在 HTML 页面中内嵌 SVG 代码有很多种方法，像<object>、<embed>、<iframe>等等。如果你还想了解各个浏览器对于 SVG 的支持情况，请移步下面网页中的“Embed SVG

code directly into the HTML”一小节：

http://www.w3schools.com/svg/svg_inhtml.asp

在 HTML 中使用 SVG

Mozilla 的开发者论坛上有很多对前端开发非常有用的文章：

https://developer.mozilla.org/en/SVG_In_HTML_Introduction

10.6 position:fixed

适用设备：iOS 5

position:fixed 的主要作用就是为网页创建一个固定位置的工具栏，iOS 5 已经支持该特性，我们可以在移动网站同样使用该功能了。

10.6.1 准备

创建一个名为“ch10r05.html”的 HTML 文件。

10.6.2 实践

在 iOS 5 之前，Safari 是不支持 position:fixed 的，如果开发人员想要做一个工具栏或者固定位置的头部或底部，开发人员需要类似于黑客的做法去动态地修改控件的位置：

```
<div id="fixedDiv">
</div>
<script>
window.onscroll = function() {
  document.getElementById('fixedDiv').style.top =
      (window.pageYOffset + window.innerHeight - 25) + 'px';
};
</script>
```

随着 iOS 5 的发布，开发人员无需再这么麻烦了，他们只需要简单地为该控件设置一个 CSS 样式规则就行了：

```
<style>
  #fixedDiv { position:fixed; }
</style>
<div id="fixedDiv">
</div>
```

10.6.3 回顾

示例中，我们首先为 window 对象注册上一个 onscroll 事件监听器，当用户滚动页面的时候，我们设置了固定位置的 div 层会固定在页面的底部。

https://developer.mozilla.org/en/SVG_In_HTML_Introduction

10.7 overflow:scroll

适用设备：iOS 5

移动设备浏览器和桌面浏览器在用户交互方面的最大区别之一就是：当用户做滚动事件时，桌面浏览器用户可以使用鼠标中间的滚轮，也可以使用浏览器边上的滚动条；而移动浏览器用户只能使用手指触碰事件。因此，在很长的一段时间里，overflow:scroll 都不被 iOS 平台所支持。但是，从 iOS 5 开始，iOS 平台支持该属性了。

10.7.1 准备

创建一个名为“ch10r06.html”的 HTML 文件。

10.7.2 实践

如果你想使你网页上的某一部分区域可以滚动，请使用下面的代码：

```
<!doctype html>
<html>
  <head>
    <title>Mobile Cookbook</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-
initial-scale=1.0">
    <style>
      div {
        width:200px;
        height:200px;
        margin:0 auto;
        border:1px solid black;
        overflow: scroll;
        -webkit-overflow-scrolling: touch;
      }
```

```
    </style>
  </head>
  <body>
  <div>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
    <p>Lorem Ipsum</p>
     <p>Lorem Ipsum</p>
  </div>
  </body>
</html>
```

10.7.3 回顾

通过把区域的样式文件中 overflow 的值设为 scroll，把-webkit-overflow-scrolling 的值设置为 touch。我们就可以在 Safari 中滚动该区域了，不需要其他任何的额外处理。

10.7.4 延伸

在过去的几年，有很多的开发人员尝试使用一些黑客技巧来模拟原生的用户滚动行为。据说苹果公司曾经开发过但是却未发布过网页框架叫做 **PastryKit**，该框架可以很好地模拟一些原生应用的功能，该框架激励了很多开发人员去做类似的框架，其中几个比较知名的是：

- Sencha touch: http://www.sencha.com/products/touch/
- iScroll: http://cubiq.org/iscroll
- Scrollability: https://github.com/joehewitt/scrollability/