jQuery Mobile

DEVELOP AND DESIGN

Kris Hadlock

jQuery Mobile

DEVELOP AND DESIGN

Kris Hadlock

jQuery Mobile: Develop and Design
Kris Hadlock

Peachpit Press
1249 Eighth Street
Berkeley, CA 94710
510/524-2178
510/524-2221 (fax)

Find us on the Web at: www.peachpit.com
To report errors, please send a note to: errata@peachpit.com
Peachpit Press is a division of Pearson Education.
Copyright © 2012 by Kris Hadlock

Acquisitions Editor: Michael Nolan
Project Editor: Rebecca Gulick
Development Editor: Robyn G. Thomas
Contributing Writer: Jay Blanchard
Technical Reviewer: Jay Blanchard
Production Coordinator: Myrna Vladic
Compositor: David Van Ness
Copyeditor: Gretchen Dykstra
Proofreader: Patricia Pane
Indexer: Valerie Haynes-Perry
Cover Design: Aren Howell Straiger
Interior Design: Mimi Heft

Notice of Rights

All rights reserved. No part of this book may be reproduced or transmitted in any form by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. For information on getting permission for reprints and excerpts, contact permissions@peachpit.com.

Notice of Liability

The information in this book is distributed on an "As Is" basis, without warranty. While every precaution has been taken in the preparation of the book, neither the author nor Peachpit Press shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the instructions contained in this book or by the computer software and hardware products described in it.

Trademarks

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Peachpit was aware of a trademark claim, the designations appear as requested by the owner of the trademark. All other product names and services identified throughout this book are used in editorial fashion only and for the benefit of such companies with no intention of infringement of the trademark. No such use, or the use of any trade name, is intended to convey endorsement or other affiliation with this book.

ISBN-13: 978-0-321-82041-9
ISBN-10: 0-321-82041-X

9 8 7 6 5 4 3 2 1

Printed and bound in the United States of America

To my wife, Lisa, who carried our first child while I wrote this book.
Only true love can withstand the amount of time that it
takes to write a book while having a new baby.
And to my son, Lucas, words cannot
express the love I feel for you.

ACKNOWLEDGMENTS

There are many people I would like to thank for the opportunity and help they gave before, during, and after this book was being written: Neil Salkind, for helping me navigate the world of publishing and for his support while I was writing. Robyn Thomas for her patience. Jay Blanchard for stepping in when needed and providing excellent technical reviews. Rebecca Gulick for helping to move things along. Michael Nolan for working out the details. All my customers, for understanding how busy I've been. And, of course, Peachpit for giving me the opportunity to write for you.

ABOUT THE AUTHOR

Kris Hadlock has been a web developer and designer since 1996, working on projects for companies such as SPIN Magazine, IKEA, United Airlines, JP Morgan Chase, Canon, and Phoenix Children's Hospital, to name a few. Kris is a featured columnist and writer for numerous websites and design magazines, including Peachpit.com, InformIT.com, IBM developerWorks (*www.ibm.com/developerworks*), and *Practical Web Design* magazine. His other books include *Ajax for Web Application Developers* and *The ActionScript 3.0 Migration Guide*. He is the founder and lead developer-designer of Studio Sedition (www.studiosedition.com), specializing in the fusion of form and function.

CONTENTS

Introducing the Future of Web Development x
Supported jQuery Mobile Platforms xvi

PART I THE FOUNDATION OF JQUERY MOBILE

CHAPTER 1 **UNDERSTANDING JQUERY** **4**
Getting Started 6
jQuery Fundamentals 7
Selecting HTML elements *7*
Managing events and functions *8*
Waiting for documents to be ready *11*
Applying special effects *12*
Using Ajax *14*
Wrapping Up 17

CHAPTER 2 **THE ROLE OF HTML5** **18**
Semantic HTML5 20
Creating an HTML5 template *20*
The viewport Meta Tag 21
Understanding `data-` Attributes 24
Wrapping Up 27

CHAPTER 3 **GETTING STARTED WITH JQUERY MOBILE** **28**
How jQuery Mobile Works 30
Adding the jQuery Mobile framework to your website *30*
Page and toolbar components *32*
Structuring mobile webpages *34*
Wrapping Up 37

PART II UI COMPONENTS

CHAPTER 4 **CREATING MULTIPAGE WEBSITES** **40**

Multipage Template 42
Single-Page Template 45
Hashtags and history *45*
Link Types 47
Preloading and Caching Pages 52
Working with Page Transitions 55
Customizing Loading Messages 58
Wrapping Up 59

CHAPTER 5 **DIALOG WINDOWS AND BUTTONS** **60**

Creating a Basic Dialog Window 62
Working with Buttons 69
Wrapping Up 73

CHAPTER 6 **WORKING WITH TOOLBARS** **74**

Toolbars 76
Header toolbars *76*
Adding buttons *78*
Footer toolbars *82*
Positioning toolbars *83*
Creating Navigation Bars 85
Wrapping Up 87

CHAPTER 7 **LAYOUT OPTIONS** **88**

Grids 90
Grid columns *90*
Grid rows *97*
Collapsible Content 99
Creating accordions *101*
Wrapping Up 103

CHAPTER 8 **WORKING WITH LISTS** **104**

Basic Linked Lists 106
Numbered lists *109*

Nested lists ... *109*
Inset lists ... *111*
Customizing Lists ... 112
Split button lists ... *112*
List dividers ... *114*
Count bubbles, thumbnails, and icons ... *115*
Wrapping Up ... 119

CHAPTER 9 **SEARCH FILTERING** ... **120**
Creating a Search Filter Bar ... 122
Creating Custom Search Filters ... 126
Wrapping Up ... 131

CHAPTER 10 **FORM ELEMENTS** ... **132**
Text Inputs ... 134
Options ... *137*
Methods ... *139*
Events ... *140*
Checkboxes and Radio Buttons ... 141
Select Menus ... 146
Options ... *150*
Methods ... *151*
Sliders ... 152
Options ... *153*
Methods and events ... *154*
Flip Toggle Switches ... 154
Wrapping Up ... 155

CHAPTER 11 **THEMING JQUERY MOBILE** ... 156
Core Color Swatches ... 158
The ThemeRoller ... 160
Theming Components ... 161
Page, toolbar, and button theming ... *161*
Content theming ... *163*
Form and form element theming ... *165*
List ... *166*
Wrapping Up ... 171

PART IV THE MOBILE API

CHAPTER 12 **GLOBAL OPTIONS** ... **174**

Extending the `mobileinit` Event ... 176
Creating Custom Namespaces ... 178
Delaying Page Initialization ... 180
Customizing the `subPageUrlKey` ... 183
Using Active Page and Button Classes ... 184
Enabling and Disabling Ajax Navigation ... 187
Altering the Default Page and Dialog Transitions ... 188
Wrapping Up ... 189

CHAPTER 13 **HOOKING INTO EVENTS** ... **190**

Touch Events ... 192
Orientation Events ... 195
Scroll Events ... 197
Page Transition Events ... 200
Page Initialization and Custom Widget Creation ... 204
Wrapping Up ... 207

CHAPTER 14 **WORKING WITH EXPOSED METHODS** ... **208**

Changing Pages Programmatically ... 210
Loading Pages Silently ... 217
Working with Utility Methods ... 220
Wrapping Up ... 222

PART V JQUERY MOBILE CMS

CHAPTER 15 **INSTALLING A MOBILE WORDPRESS THEME** ... **226**

Getting Started ... 228
Installing WordPress ... *228*
Creating the jQuery Mobile Theme ... 231
Adding Blog Posts and Pages ... 233
Wrapping Up ... 235

CHAPTER 16 **INSTALLING A MOBILE DRUPAL THEME** **236**
Getting Started 238
Installing Drupal *238*
Theming Drupal with jQuery Mobile 242
Installing the jQuery Mobile module *242*
Installing the jQuery Mobile theme *246*
Custom settings *247*
Adding Content 248
Wrapping Up 249

PART VI BEYOND JQUERY MOBILE

CHAPTER 17 **DETECTING MOBILE DEVICES** **252**
Using PHP 254
Identifying the browser *255*
Calling the PHP function *256*
Using JavaScript to Detect Specific Devices 257
Detecting mobile devices with JavaScript *257*
Detecting mobile browser features with jQuery *259*
Wrapping Up 261

CHAPTER 18 **TESTING WITH SIMULATORS** **262**
Exploring Your Testing Options 264
Finding Online Simulators 265
Using Simulators for Testing 268
Testing with online emulators *268*
Using remote labs *269*
Testing with desktop simulators *270*
Crowd testing *271*
Wrapping Up 271

Index 272

INTRODUCING THE FUTURE OF WEB DEVELOPMENT

Smartphone, tablet, and e-reader statistics are showing an unprecedented adoption rate, making the mobile web a very hot topic and requiring a new set of skills from web developers and designers. Mobile device usage is skyrocketing; according to Nielsen's third-quarter 2011 Mobile Media Report, "44 percent of U.S. mobile subscribers now own a smartphone device, compared to 18 percent just two years ago." That's more than double in two years, and "the number of smartphone subscribers using the mobile Internet has grown 45 percent since 2010." As for tablets, in June 2011 AMI-Partners (Access Markets International) forecasted that "tablet adoption among businesses with between 1 and 1,000 employees will grow by 1,000 percent by 2015."

Let's not forget e-readers, which are becoming very affordable and are more advanced then ever, increasing in shipment volume, as "year-over-year growth was 167%" according to International Data Corporation (IDC). With the introduction of the latest Kindle, mobile Internet access is now becoming a normal experience.

With these increases in adoption rate, there will no doubt be high demand for web developers who can create rich mobile web experiences. The jQuery Mobile framework gives web developers a quick and easy way to create mobile web experiences, making the mobile web space hard to ignore.

WHY jQUERY MOBILE?

As a web developer, you don't have to use the jQuery Mobile framework to create a mobile web experience. So why use it? For starters, the framework is built on the highly respected and widely used jQuery core and jQuery user interface (UI) foundation. It's currently sponsored by companies such as Mozilla, Palm, Adobe, Nokia, BlackBerry, and more. Plus, it works seamlessly across all popular mobile device platforms. The jQuery Mobile team is actively and regularly offering new releases, blogging about new features, and keeping their comprehensive online documentation up to date.

Most web developers and designers agree that browser and cross-platform testing is something they would rather not spend their time on. Imagine all of the devices that could potentially be accessing your mobile website. Then imagine having to test all of those platforms each and every time you build a mobile website—this would be painstaking and incredibly time-consuming. jQuery Mobile gives you this support from the start, as the team prides the framework on its approach to supporting a wide variety of mobile platforms. The framework is built on clean, semantic HTML, which ensures compatibility with a majority of web-enabled devices.

The framework also includes accessibility features, such as WAI-ARIA (Web Accessibility Initiative-Accessible Rich Internet Applications), a technical specification published by the World Wide Web Consortium regarding the increase of accessibility of webpages, which are integrated into the framework to support screen readers, such as VoiceOver on Apple iOS and other assistive technologies. Simply including the jQuery Mobile framework in your website unobtrusively

transforms your code from semantic HTML into a rich, interactive, and accessible mobile experience using jQuery and CSS. As you'll see throughout this book, the jQuery Mobile approach makes mobile web development incredibly easy, quick, and efficient, leaving the platform and browser testing up to the jQuery Mobile team.

jQuery Mobile isn't exclusively for web developers; web designers have access to the jQuery UI, which provides complete design control over mobile web applications. Built-in UI widgets, such as list views, dialogs, toolbars, search mechanisms, and a full set of form elements, are all customizable via the theme framework. Later in the book, you'll also learn about ThemeRoller, which lets you create up to 26 theme swatches using a simple, web-based interface. User experience (UX) designers also get some love, with access to stencils for OmniGraffle and Visio. And, of course, if you want to get geeky with it, the application programming interface (API) is available to web developers. As a web developer, you can configure defaults, handle many different events, and work with several exposed methods and properties.

An emerging community is helping to support the framework with a number of third-party apps and frameworks that you can use to build jQuery Mobile apps. In addition, jQuery Mobile compatible plug-ins and extensions are popping up to help web developers integrate custom widgets and add capabilities to the existing core functionality.

One very important third-party framework is blurring the line between native and mobile web-based development. As an HTML5 app platform, PhoneGap allows you to author native applications using web technologies. With PhoneGap, your web applications can easily be ported into native apps that can do things like retrieve contact information, access cameras, use geolocation, store data, and much more. To learn more about PhoneGap, visit phonegap.com. There's even a section in the jQuery Mobile online documentation about it. With these sorts of possibilities, you no longer need to program multiple versions of a native application. This makes native application development less desirable, because the same application you develop for an iPhone would need to be completely redeveloped for Android, BlackBerry, Windows Mobile, and others. Oh, and don't forget that every new release will need to be updated for every one of these different platforms. The jQuery Mobile framework provides mobile web experiences that rival native application development by giving you instant access to web applications and websites via the web browser, eliminating the need to download and install mobile applications.

WHO THIS BOOK IS FOR

This book is for people who have basic HTML experience and are interested in creating mobile websites using the jQuery Mobile framework.

WHO THIS BOOK IS NOT FOR

This book is not for people who have never created a webpage.

HOW YOU WILL LEARN

In this book, you'll learn by doing. Each chapter includes sample code and descriptions to give you a deep understanding of how things work. You can also find the code samples on the book's website (www.peachpit.com/jquerymobile).

WHAT YOU WILL LEARN

This book will teach you everything from the basics of how to create pages to custom-theming them and developing your own jQuery Mobile content-management system with WordPress and Drupal. By the time you finish this book, you'll be a jQuery Mobile expert.

WRAPPING **UP**

The jQuery Mobile framework is a powerful framework that is supported by mobile industry leaders. It can easily be added to an existing website to create a mobile web experience that is not only touch-friendly, but also supported on a majority of the leading mobile platforms as well as handicap accessible. Design control, page transitions, widget integration, scripting, API access, and much more are all at your fingertips through this framework's easy-to-use features and built-in progressive enhancement techniques.

This page intentionally left blank

SUPPORTED **jQUERY MOBILE** PLATFORMS

The jQuery Mobile framework supports the majority of modern desktop, smartphone, tablet, and e-reader platforms. Rather than spending your time testing multiple devices, you can rest assured that you're offering support for many platforms from the start. This is because the jQuery Mobile platform takes a progressive enhancement approach, which not only brings rich interactive experiences to devices, but also provides support for older browsers and phones. When a browser fails to recognize certain HTML5-specific code, the webpage renders as a simple, yet functional webpage. Users on older phones or browsers are familiar with a limited web experience, therefore this approach still renders an acceptable basic HTML webpage in these cases.

Currently, Android and Apple iOS are the leading mobile operating systems. Android has the largest operating system market share (44.2 percent), while Apple has the largest smartphone market share in the United States (28.6 percent). The Windows, BlackBerry, SymbianOS, and Palm/HP webOS operating systems comprise the remaining majority of smartphone market share (**Figure 1**). All of these platforms/operating systems are supported by the jQuery Mobile framework.

MAJOR PLATFORMS

The following list provides a bit more information on the major platforms that are fully supported by the jQuery Mobile framework.

IOS

iOS is Apple's mobile operating system. Originally developed for the iPhone, it has been extended to support other Apple devices such as the iPod Touch, iPad, and Apple TV. iOS uses Safari as its web browser.

ANDROID

Android is an open-source software project and operating system led by Google. It has been used in a plethora of phones, tablets, and other devices since its release under the Apache license. Android uses Google Chrome as its web browser.

WINDOWS PHONE

Windows Phone is Microsoft's mobile operating system. The system is integrated with third-party and other Microsoft services. Windows Phone uses Internet Explorer Mobile as its web browser.

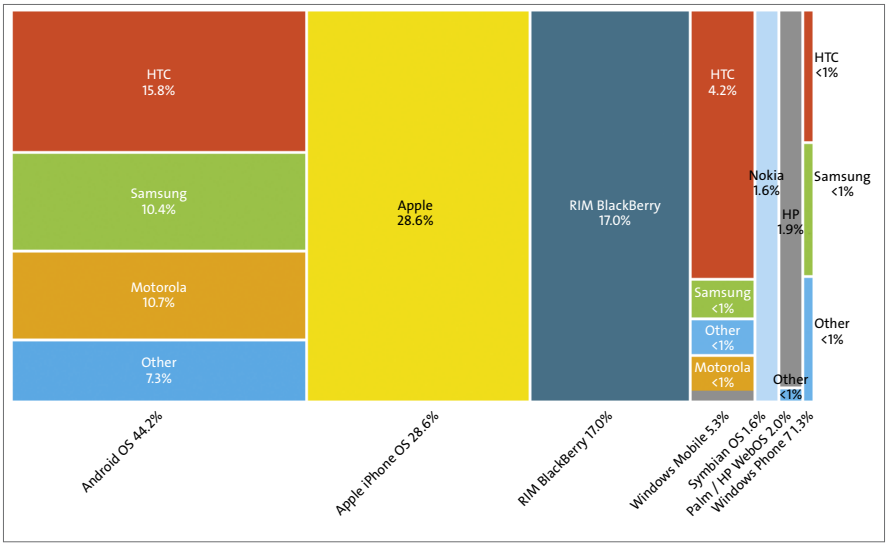


FIGURE 1 Operating system market share as of 2011.

BLACKBERRY

BlackBerry is powered by a proprietary mobile operating system, offered on its own set of smartphones and tablets.

WEBOS

webOS is the open-source operating system used by Palm devices. webOS uses the WebKit layout engine in its web browser, simply named Web.

GRADED SUPPORT

jQuery Mobile uses a three-tier, graded list of the platforms that are supported by the framework. The tiers are A, B, and C. A includes a full experience with the option of Ajax-based page transitions, B includes the same experience minus the Ajax-based page transitions, and C is a basic, yet functional, HTML experience.

A-grade support includes all the major mobile operating systems that were mentioned previously and more, including Apple iOS, Android, Windows Phone, BlackBerry, Palm/HP webOS, Kindle 3, Kindle Fire, and more. B-grade support includes BlackBerry 5, Opera Mini, and Nokia SymbianOS. C-grade support includes BlackBerry 4, Windows Mobile, and all older smartphones and feature phones. This list is always evolving; to see an up-to-date list, visit jquerymobile.com and check out the supported platforms.

This page intentionally left blank

4

CREATING MULTIPAGE WEBSITES

Now that we've covered the basics of structuring mobile webpages, we'll take a deeper look and get a better understanding of the functionality behind them. As mentioned in Chapter 3, "Getting Started with jQuery Mobile," there are two ways to structure webpages for jQuery Mobile: incorporate all the pages in the same file, or create separate files for each page like a typical website. Understanding the different page template types is the foundation for customizing a number of different page-related functionalities. In addition to learning about the page template types, you'll also see how to preload and cache pages, work with different page transitions, and customize loading messages. You'll learn to create custom functionality to take your pages to the next level.

MULTIPAGE TEMPLATE

Internal linking occurs automatically when you have multiple jQuery Mobile pages in the same HTML file. As you've learned, jQuery Mobile pages are defined by adding a `data-role` with a value of page to an HTML element and anything within that page becomes relative to that page. In jQuery Mobile, typical separate webpages are considered *single-page* templates, while webpages that contain multiple pages are considered *multipage* templates. Let's refer back to our multipage template example from the previous chapter (with a few small additions):

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
        → charset=UTF-8">
        <meta name="viewport" content="width=device-width,
        → initial-scale=1">
        <title>Multipage template - jQuery Mobile: Design and
        → Develop</title>
        <link rel="stylesheet" href="http://code.jquery.com/
        → mobile/1.0.1/jquery.mobile-1.0.1.min.css" />
        <script src="http://ajax.googleapis.com/ajax/libs/
        → jquery/1.7.1/jquery.min.js"></script>
        <script src="http://code.jquery.com/mobile/1.0.1/
        → jquery.mobile-1.0.1.min.js"></script>
</script>
        <script type="text/javascript" src="assets/js/ui.js">
</script>
    </head>
<body class="container">
    <div data-role="page" id="page-one" data-title="Page 1">
        <div data-role="header">
            <h1>Page 1</h1>
        </div>
```

```
        <div data-role="content">
            <p>Body copy for page 1</p>
            <p><a href="#page-two">Link to page 2</a></p>
        </div>
        <div data-role="footer">
            Copyright
        </div>
    </div>
    <div data-role="page" id="page-two" data-title="Page 2">
        <div data-role="header">
            <h1>Page 2</h1>
        </div>
        <div data-role="content">
            <p>Body copy for page 2</p>
            <a href="#page-one">Link to page 1</a>
        </div>
        <div data-role="footer">
            Copyright
        </div>
    </div>
</body>
</html>
```

Within this multipage template are two pages defined by `div` elements with custom `id`s. The jQuery Mobile framework shows only one of these pages at a time, and it uses the `data-title` attribute to change the page title dynamically. The page that is shown by default is determined by the order of the source code. In this example, the first page has an `id` of `page-one`, but if the pages in this file were switched, so that the element with an `id` of `page-two` came first, then that would be the default page to load. In other words, the value of the `id` attribute doesn't

FIGURE 4.1 A multipage template with two pages and the transition in between.

determine what page is shown by default; the default page is determined only by the source code order. However, the `id` is used for other important purposes, such as linking pages to one another.

This is where jQuery Mobile pages begin to act like separate webpages. Note that hyperlinks have been added to the original template to link from one page to another. This is somewhat like the toggle functionality that's common in JavaScript development, where `id` values are used to hide and reveal certain HTML elements. The difference here is that jQuery Mobile handles the functionality for you. To link from one page to another you simply need to:

1. Create a hyperlink.
2. Type the pound sign (#).
3. Specify the id value of the page you want to link to.

The end result looks like this:

```
<a href="#page-two">Link to page 2</a>
```

It's similar to creating a page anchor; the difference is that you're referencing the `id` value of another page. Remember that each page needs a unique `id` value. In this case, I've used `page-one` and `page-two`, but you can use something more descriptive and relative to the content of your jQuery Mobile page. The cool thing about the jQuery Mobile framework is that it transitions dynamically from one page to another without requiring you to write an ounce of code. **Figure 4.1** shows an example of this code in both page views and the transition from one to another.

SINGLE-PAGE TEMPLATE

Single-page templates are separate HTML files that act as independent webpages, just like any standard webpage. The main difference is in how jQuery Mobile connects webpages using Ajax. As with multipage templates, the Ajax-based navigation in single-page templates uses hashtags to create a page history. The Ajax-based navigation used by jQuery Mobile is the default, but it can be turned off by setting the `ajaxEnabled` setting to `false` in the configuration. You'll learn more about the configuration settings later in this book.

Ajah is an abbreviation that is sometimes used for Asynchronous JavaScript and HTML. Ajah is essentially Ajax without the XML; the XMLHttpRequest is still used, but HTML is exchanged with the server rather than XML. This is what jQuery Mobile uses as the user browses independent webpages. We briefly covered how the jQuery Mobile framework uses the XMLHttpRequest to load subsequent pages in the previous chapter, but there's a lot to learn and understand about this simple request.

One great thing about the jQuery Mobile framework is how it tracks history: it supports the back and forward buttons! Also, while subsequent pages load, the framework provides a default loading message and transitions between pages. The default page transition is to slide between two pages, and the default loading message is a spinning icon with a "loading..." message. Both options are configurable, as you'll learn in the next chapter. For now, let's see how hashtags and history work in jQuery Mobile.

HASHTAGS AND HISTORY

jQuery Mobile uses hashtags to manage history in single- and multipage templates. The window object's `location.hash` is used to make changes and updates to the history, so the back and forward buttons function as usual, which is uncommon in other Ajax-based systems. Essentially, jQuery Mobile prevents the default functionality of all hyperlinks and uses the hashtag functionality to handle history. Not only is the history updated, the hashtag system also creates a valid URL that can be bookmarked for later reference.

The only issue with jQuery Mobile's hashtag-based navigation is that it doesn't support deep linking. However, there are some workarounds you can use to support this functionality. With a little help from jQuery, you can add a script, like the following, to your webpage and your deep links will function as usual:

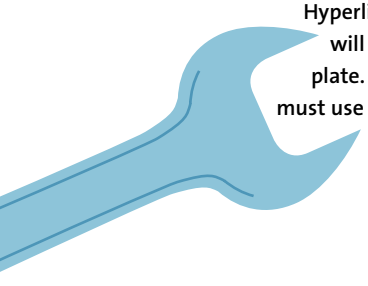
```
$(document).ready(function() {
    $('a[href^="#"]').bind('click vclick', function () {
        location.hash = $(this).attr('href');
        return false;
    });
});
```

In HTML, all hyperlinks that include a pound sign (#) as their first character are identified as anchors. This script uses a regular expression that includes the caret symbol to identify all anchor elements that have an `href` attribute with a value that begins with the pound sign. Once these elements have been selected, you can use the `bind` method to bind a `click` and `vclick` event to the anchor tags and assign an anonymous function handler as the callback.

The callback function sets the window object's `location.hash` to the value of the anchor and returns `false` to prevent the browser from performing the default action associated with clicking the hyperlink.

NOTES: The caret (^) symbol is often used in regular expressions to designate the beginning of a string.

The `vclick` event is an option used in jQuery Mobile by devices that support touch events. The event supports faster page changes and during page transitions it keeps the address bar hidden.



Hyperlinking from single-page to multipage templates with Ajax enabled will load only the first page in the source code of the multipage template. To link to a multipage template from a single-page template, you must use an external hyperlink by using `rel="external"` or `data-ajax="false"`.

LINK TYPES

jQuery Mobile supports standard HTML link types as well as a number of custom link types related to the mobile experience. The following tables offer a list of the supported link types available through the jQuery Mobile framework. Each table shows options categorized based on their end result and/or support of Ajax. **Table 4.1** describes links that support Ajax.

TABLE 4.1 Link types that support Ajax

HYPERLINK MARKUP	DESCRIPTION
`<a href="http://www.jquerymobile.tv">` `→ Hyperlink within same domain </a>`	A standard HTML link that is transformed by the jQuery Mobile framework to use Ajax, include page transitions, and support page history.
`<a href="http://www.jquerymobile.tv"` `→ data-rel="dialog"> Open a dialog</a>`	An option used for dialog windows that is not tracked in page history.
`<a href="http://www.jquerymobile.tv"` `→ data-rel="back">Back button</a>`	This option can be used to navigate back in page history; it's a great option for providing a back button from a page or dialog. The `href` is ignored in A- and B-grade browsers, but is necessary for C-grade browsers.

GRADED **SUPPORT**

As mentioned in the book's introduction, jQuery Mobile uses a three-tier graded list of platforms that are supported by the framework: A, B, and C. A includes a full experience with the option of Ajax-based page transitions; B includes the same experience minus the Ajax-based page transitions; and C is a basic yet functional HTML experience.

A-grade support includes all the major mobile operating systems mentioned previously and others, including Apple iOS, Android, Windows Phone, BlackBerry, Palm WebOS, Kindle 3, and Kindle Fire. B-grade support includes BlackBerry 5, Opera Mini, and Nokia Symbian. C-grade support includes BlackBerry 4, Windows Mobile, and all older smartphones and feature phones. Visit jquerymobile.com to see an up-to-date list of supported platforms.

Table 4.2 describes a list of hyperlinks that disable the Ajax page-transition functionality. These hyperlinks are great for pages on an external domain, pages that open in a new window, linking from single to multipage templates, or linking to pages where you don't want to use Ajax.

Table 4.3 includes link types that stem from a basic HTML hyperlink with the addition of specific attributes.

TABLE 4.2 Link types that disable Ajax

HYPERLINK MARKUP	DESCRIPTION
`<a href="http://www.jquery.com">External hyperlink</a>`	Linking to a page on an external domain automatically disables the Ajax functionality.
`<a href="http://www.jquery.com" rel="external">External hyperlink</a>`	By default this attribute defines a hyperlink as external, which not only disables Ajax, but also removes it from the page hashtag history and refreshes the webpage. This option can be transformed using jQuery to open new windows in a standards-compliant way.
`<a href="http://www.jquery.com" data-ajax="false">Hyperlink disables Ajax</a>`	This option provides a way to define a hyperlink as external, which not only disables Ajax, but also removes it from the page hashtag history and refreshes the webpage.

TABLE 4.3 Miscellaneous link types

HYPERLINK MARKUP	DESCRIPTION
`<a href="tel:15556667777">Phone Number</a>`	This hyperlink initiates a phone call when clicked on some phones.
`<a href="mailto:jdoe@jquerymobile.tv">Email link</a>`	This hyperlink initiates a new email that's prefilled with the specified email address.
`<a href="#">Hyperlink</a>`	This hyperlink returns `false`. It's useful when creating a back button as in Table 4.1.

The jQuery Mobile framework uses many hyperlink attributes to create enhancements to otherwise normal HTML webpages. This is just another reason why jQuery Mobile is more appealing than creating a mobile website from scratch. It lets you focus on what matters, eliminating the need to write core functionality every time you create a new mobile website.

The following examples show a few of the link types covered in the tables.

This webpage offers three examples of the link types that can be used in jQuery Mobile: an internal link, an external link, and a link that disables Ajax:

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
        → charset=UTF-8">
        <meta name="viewport" content="width=device-width,
        → initial-scale=1">
        <title>Single-page template - Page 1 - jQuery Mobile: Design
        → and Develop</title>
        <link rel="stylesheet" href="http://code.jquery.com/
        → mobile/1.0.1/jquery.mobile-1.0.1.min.css" />
        <script src="http://ajax.googleapis.com/ajax/libs/
        → jquery/1.7.1/jquery.min.js"></script>
        <script src="http://code.jquery.com/mobile/1.0.1/
        → jquery.mobile-1.0.1.min.js"></script>
        </script>
    </head>
<body class="container">
    <div data-role="page">
        <div data-role="header">
            <h1>Page 1</h1>
        </div>
        <div data-role="content">
            <p><a href="single-page-2.html">Link to page 2</a></p>
            <p><a href="single-page-2.html" rel="external">
            → External Link to page 2</a></p>
            <p><a href="single-page-2.html" data-ajax="false">
            → Ajax-disabled link to page 2</a></p>
```

```
        </div>
        <div data-role="footer">
            Copyright
        </div>
    </div>
</body>
</html>
```

An internal link requires no special markup and is where the framework will interject and create a page transition between the current page and the page that's being linked to. An external link is one that requires the `rel` attribute, which must be set to a value of `external`. Setting a hyperlink as external disables Ajax, removes it from the page hashtag history, and refreshes the webpage when the link is clicked. The link that disables Ajax requires a `data-ajax` attribute with a value of `false`. This disables Ajax, removes it from the page hashtag history, and refreshes the webpage, just like the previous example, with the main difference being that the `rel="external"` attribute can be used as a standards-compliant way to create hyperlinks that target new windows with a little help from jQuery.

In this next example, there's an internal link and a hyperlink that acts as a back button:

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
        → charset=UTF-8">
        <meta name="viewport" content="width=device-width,
        → initial-scale=1">
        <title>Single-page template - Page 2 - jQuery Mobile: Design
        → and Develop</title>
        <link rel="stylesheet" href="http://code.jquery.com/
        → mobile/1.0.1/jquery.mobile-1.0.1.min.css" />
```

```
        <script src="http://ajax.googleapis.com/ajax/libs/
        → jquery/1.7.1/jquery.min.js"></script>
        <script src="http://code.jquery.com/mobile/1.0.1/
        → jquery.mobile-1.0.1.min.js"></script>
    </head>
<body class="container">
    <div data-role="page">
        <div data-role="header">
            <h1>Page 2</h1>
        </div>
        <div data-role="content">
            <p><a href="single-page.html">Ajax link to page 1</a></p>
            <p><a href="#" data-rel="back">Back button</a></p>
        </div>
        <div data-role="footer">
            Copyright
        </div>
    </div>
</body>
</html>
```

Setting the `data-rel` attribute to back creates a hyperlink that acts like a back button. When this link is clicked, the jQuery Mobile framework automatically links to the previous page in history.

PRELOADING AND CACHING PAGES

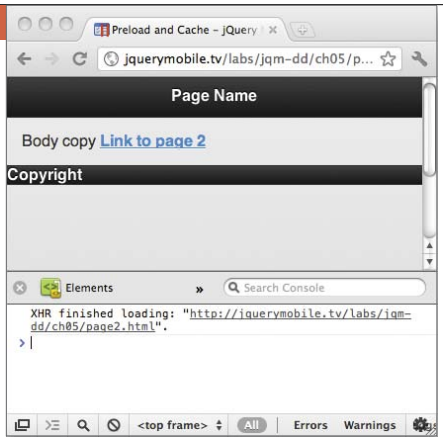


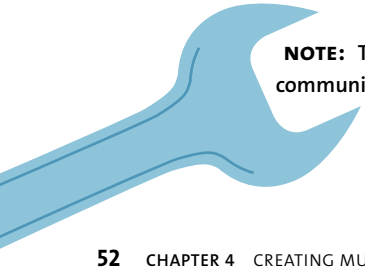
FIGURE 4.2 A page being preloaded by jQuery Mobile using the prefetch attribute.

Page caching is very important when working with anything web-based. Cache refers to hidden storage where files are collected. By default, browsers handle webpage caching by putting the associated files in a local directory on the user's computer so the webpage and its associated assets will load more quickly during subsequent visits. jQuery Mobile takes this concept a step further: the framework allows pages to be cached even before they're linked to or displayed in the browser. Taking advantage of this functionality is as easy as using an HTML attribute named `data-prefetch`. Simply add it to any hyperlink in your webpage, and once the webpage has loaded, the URLs that these hyperlinks point to will be preloaded and cached. The following line of code shows how to add this attribute to a hyperlink to preload and cache the page it's pointing to:

```
<a href="page2.html" data-prefetch>Link to page 2</a>
```

When running this example in a web browser, the request is made immediately upon page load. **Figure 4.2** shows an example of the XMLHttpRequest being made from Chrome when the page loads.

The framework also provides a way to handle page preloading via the application programming interface (API). We'll talk more specifically about the API later in the book, but it's important to know how to access it.



NOTE: The application programming interface (API) allows for communication among multiple pieces of software.

To preload pages programmatically, you first need to access the jQuery Mobile object:

```
$.mobile
```

Once the document object is ready, use the `loadPage` method to pass the URL you want to preload and define a number of properties for the request:

```
$(document).ready(function() {
    $.mobile.loadPage( "page2.html", {
        type: false,
        reloadPage: false,
        type: 'get'
    });
});
```

There are quite a few arguments you can pass as options in the `loadPage` method. **Table 4.4** lists those optional arguments.

TABLE 4.4 `loadPage` optional arguments

ARGUMENT	DESCRIPTION
`data`	Holds an object or string that can be sent with an Ajax page request.
`loadMsgDelay`	Sets the number of milliseconds to delay before showing the load message. The default is 50 milliseconds.
`pageContainer`	Holds the loaded page. The default is the jQuery Mobile page container, but this is customizable.
`reloadPage`	Defines whether or not to reload the page being requested. The default value is `false`.
`role`	When the page is loaded, this defines the `data-role` value that will be applied.
`type`	Specifies whether the request is a get or a post.

In addition to providing control for preloading files, the jQuery Mobile framework provides control for caching pages in the document object model (DOM). Just like the preload option, you can define whether a page should be cached in the DOM via an HTML attribute or the jQuery Mobile API. The following example shows how the attribute can be used in a page tag to cache a webpage:

```
<div data-role="page" data-dom-cache="true">
```

To cache a webpage using the API, you can set all pages to cache by default:

```
$.mobile.page.prototype.options.domCache = true;
```

Or, you can cache a page independently. You would cache a page with an `id` of `my-page` like so:

```
$('#my-page').page(true);
```

When working with the single-page template, jQuery Mobile also manages the pages it preloads so the DOM doesn't get too large. If many pages are kept in the DOM, the browser's memory usage can get out of control and the browser will likely slow down or crash. To manage the memory size, the framework removes pages that are loaded via Ajax from the DOM automatically via the `pagehide` event when the visitor navigates away from them. You'll learn more about this and other events later in the book.

There are a number of benefits to preloading and caching pages. They load quicker and prevent the Ajax loading message from appearing when a visitor tries to access the preloaded page. However, it's important to keep in mind that each preloaded page creates an additional HTTP request, which uses more bandwidth. Therefore, it's important to preload pages only when you think visitors are likely to view a subsequent webpage.

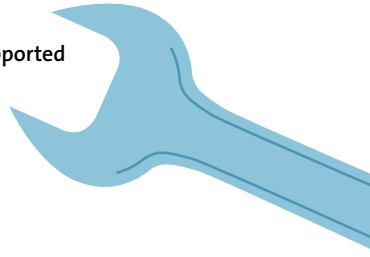
WORKING WITH PAGE TRANSITIONS

A number of page transitions can be used with the jQuery Mobile framework. All the page transitions are CSS-based effects. When using Ajax navigation, the page transitions work between linked pages or form submissions. **Table 4.5** lists the available page transitions in the framework.

TABLE 4.5 Page transitions

METHOD	DESCRIPTION
`slide`	Slides the hyperlinked page in from the right to replace the current page. Slide is the default page transition.
`slideup`	Slides the hyperlinked page up to replace the current page.
`slidedown`	Slides the hyperlinked page down from the top to replace the current page.
`pop`	Zooms the hyperlinked page in from the center of the current page and replaces it.
`fade`	Fades in the hyperlinked page over the current page and replaces it.
`flip`	Creates a 3D effect where the hyperlinked page appears to be on the backside of the current page as it flips and the hyperlinked page comes into view.

NOTE: Use the flip transition with caution, because it is supported only by browsers that support 3D CSS transform rendering.



Each of these page transitions is easy to set up globally as a default using the `defaultPageTransition` property. To set it up properly, it's necessary to bind to the `mobileinit` event, which is accessible through the API via the jQuery Mobile object:

```
$(document).bind("mobileinit", function() {
    $.mobile.defaultPageTransition = 'fade';
});
```

The binding of the event handler needs to be executed before the jQuery Mobile library loads. Therefore, in your HTML file, arrange your JavaScript files in a specific order:

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/
→ jquery.min.js"></script>
<script src="assets/js/custom-jqm-transitions.js"></script>
```

```
<script src="http://code.jquery.com/mobile/1.0.1/
→ jquery.mobile-1.0.1.min.js"></script>
```

They can also be set on a per-link basis to override the default page transition. This option is useful for a number of situations, such as creating pop-up windows that use the pop transition:

```
<a href="popup.html" data-transition="pop">Pop-up</a>
```

Or, form submissions that use the `flip` transition:

```
<form action="" method="get" data-transition="flip">
    <input type="text" name="name">
    <input type="submit" value="Submit">
</form>
```

It's even possible to set the page transition to none to disable the page transition for a particular hyperlink or form submission:

```
<form action="" method="get" data-transition="none">
    <input type="text" name="name">
    <input type="submit" value="Submit">
</form>
```

jQuery also creates a reverse transition by automatically applying the same page transition when the back button is pressed.

The complete example code for the global and individual page transitions looks like the following:

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html;
    → charset=UTF-8">
    <meta name="viewport" content="width=device-width,
    → initial-scale=1">
```

```
    <title>Page Transitions - jQuery Mobile: Design and Develop
    → </title>
    <link rel="stylesheet" href="http://code.jquery.com/
    → mobile/1.0.1/jquery.mobile-1.0.1.min.css" />
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.1/
    → jquery.min.js"></script>
    <script src="assets/js/custom-jqm-transitions.js"></script>
    <script src="http://code.jquery.com/mobile/1.0.1/
    → jquery.mobile-1.0.1.min.js"></script>
</head>
<body class="container">
    <div data-role="page">
        <div data-role="header"><h1>Page Name</h1></div>
        <div data-role="content">
            <p><a href="popup.html" data-transition="pop">Pop-up
            → </a></p>
            <form action="" method="get" data-transition="flip">
                <input type="text" name="name">
                <input type="submit" value="Submit">
            </form>
        </div>
        <div data-role="footer">Copyright</div>
    </div>
</body>
</html>
```

Beginning with the head of the document, you can see that the jQuery library is loaded first, then the custom JavaScript where the `mobileinit` event from our previous example is bound and the default page transition is set. Last, but not least, the jQuery Mobile library is loaded. Within the page there's a hyperlink that has a `pop` transition applied and a form that has a `flip` transition applied.

CUSTOMIZING LOADING MESSAGES

When pages are loading, a default message appears if there's a delay or if the page is not yet preloaded. The jQuery Mobile framework allows you to customize this message and a page-error loading message.

You can customize the loading message through the `loadingMessage` property. Set `loadingMessage` to any custom string by binding to the `mobileinit` event, which jQuery Mobile fires as soon as the document loads.

```
$(document).bind("mobileinit", function() {
    $.mobile.loadingMessage = 'Please wait';
});
```

This code will display a loading message that says "Please wait." The default message used for page loading is "Loading." The property can also be set to `false` to display no message at all:

```
$(document).bind("mobileinit", function() {
    $.mobile.loadingMessage = false;
});
```

When there is an error loading a page, jQuery Mobile displays a message that can also be customized through the API. The default message for page-load errors is "Error Loading Page." You can set the `pageLoadErrorMessage` property to any custom string as the error message:

```
$(document).bind("mobileinit", function() {
    $.mobile.pageLoadErrorMessage = 'There was an error, please try
    → again.';
});
```

WRAPPING **UP**

Working with pages is easy with jQuery Mobile: all you really need to know is basic HTML and a few mobile-related attributes. With most of the heavy lifting being done by the framework, it's easy to focus on the results of the website you're building. Understanding the internal functionality behind how pages work in jQuery Mobile is what begins to set you up for writing custom functionality. Customizing specific functionality in the messaging and behind the scenes helps to personalize your website. The level of customization that jQuery Mobile provides can be very useful in making a website more user-friendly. Visual indicators, like custom loading messages and page transitions, set expectations for visitors and provide them with a frame of reference, so they know when certain things are happening. Preloading and caching improves usability by speeding up page loads and giving visitors what they want when they want it. jQuery Mobile provides fine-grained control to enhance mobile websites in a custom way.

INDEX

$() function, using with HTML elements, 7
^ (caret), using in regular expressions, 46
- (dash)
  using at end of namespaces, 178
  using with custom namespaces, 25
(pound sign), including in hyperlinks, 46

A

<a href> markup, 48–49
accordions, creating, 101–102
activeBtnClass property, 186
activePageClass property, modifying, 184–186
a-e swatches
  examples, 162–163, 167
  explained, 158
  list divider theme, 168
Ajah (Asynchronous JavaScript and HTML), 45
Ajax
  disabling, 48–50
  link types, 47
  overview, 14
  requests and responses, 15–16
  use of, 16
ajax method, using, 16
Ajax navigation, 45
  ajaxEnabled property, 187
  disabling, 187
  enabling, 187
alphabetical keyboard, 135–136
anchors, identifying in HTML, 46
Android SDK, 270
animation methods, availability of, 12–13
API (application programming interface), 52
asynchronous, defined, 14
Asynchronous JavaScript and HTML (Ajah), 45
attributes. *See* data- attributes

B

back button, hyperlink as, 50–51
background gradient, adding, 92–93
bind method
  syntax for, 10
  using with click events, 46
BlackBerry SDK, 270
blog posts
  adding to WordPress, 233–235
  creating in Drupal, 248
  "Hello world!", 232
  listing in WordPress, 234–235
  saving in WordPress, 233
  viewing in WordPress, 233
box model feature, adhering to, 259–260
browser support, determining, 259–260
button classes, using, 184–186
button control groups, separating, 97
buttons. *See also* Save button; split button lists
  activeBtnClass property, 186
  aligning, 81
  checkmark data-icon, 72–73
  converting hyperlinks to, 69
  corners data- attribute, 70
  customizing icons, 72
  data attributes, 70
  data-inline, 69
  data-role, 69
  defining themes for, 158
  float left CSS, 81
  float right CSS, 81
  grouping, 80
  home data-icon, 71–72
  icon data- attribute, 70–71
  iconpos data- attribute, 70, 72
  iconshadow data- attribute, 70
  inline data- attribute, 70
  input elements, 70
  positioning icons, 72
  separating groups, 81
  shadow data- attribute, 70
  theme data- attribute, 70
buttons theming component, 161–162

C

caching webpages, 52–54
  in DOM (document object model), 54
  using API, 54
caret (^), using in regular expressions, 46
CDN (content delivery network), 6
CDN-hosted files
  obtaining, 31
  referencing, 30
changePage method
  setting to step2.php, 215–216
  triggering, 213–214
  using, 210–216
changePage properties
  adding to options argument, 213
  allowSamePageTransition, 212
  changeHash, 212
  data, 212
  dataUrl, 212
  pageContainer, 212
  reloadPage, 212
  reverse, 212
  role, 212
  showLoadMsg, 212
  transition, 212
  type, 212
checkboxes, 141–145
  applying mini, 145
  controlgroup data-role groups, 142
  disable method, 145
  enable method, 145
  events, 145
  horizontal toggle set, 143–144
  legend in fieldset, 142
  questions, 142
  versus radio buttons, 141
  refresh method, 145
  setting theme, 145
  statements, 142
  versus text inputs, 145
checkmark icon, using, 72–73
Chrome Developer Tools, using, 37
clearQueue method, 13–14
click event
  adding to logo widget, 210–211
  applying, 11–12
  example of, 8–10
  using bind method with, 46
CMS (content management system), 227, 237. *See also* Drupal; WordPress
collapsible content area, example of, 164
collapsible content, nesting, 100–101
collapsible content widget. *See also* jQuery Mobile widgets; widgets
  creating accordions, 101–102
  described, 99
  using, 99–100
color blocks, dragging, 160
color swatches, 158–159
content area, separating data-theme from, 165
content delivery network (CDN), 6
content theming, 163–164. *See also* form element theming; theming components
controlgroup, select menu in, 149
copyright notices, including in footers, 82
count bubble
  adding to list item, 115–117
  adding to listview, 169
Cowemo online simulator, 265–266
crowd testing, 271. *See also* testing
custom data
  accessing, 24
  using, 24
custom namespaces, using, 24–26. *See also* namespaces
customizing code, 178

D

dash (-)
  using at end of namespaces, 178
  using with custom namespaces, 25
data- attributes
  adding to image tags, 24
  for buttons, 70

data- attributes *(continued)*
custom namespaces, 24–26
customizing, 178
data-role custom attribute, 25, 32–33
explained, 24
preventing name clashing, 24–25
values, 24
data-collapsed attribute, using, 100
data-divider-theme, using, 168
data-filter attribute, adding, 122–123
data-filter-placeholder attribute, using, 123–124
data-filtertext attribute, adding, 130–131
data-hyperlink attribute, using with changePage method, 211
data-prefetch attribute, using, 52
data-rel attribute
setting to back, 51
setting to dialog, 62–63
data-role attribute
for button, 69–70
button setting, 80
collapsible value, 99
collapsible-set value, 101–102
defining pages, 32
described, 25
header value, 76
list-divider value, 114
listview value, 106
navbar value, 85
data-split-icon attribute, using, 113
data-split-theme, using with listview, 170
data-theme attribute
applying to form elements, 166
described, 25
in listview, 166–167
separating from content area, 165
using with linked lists, 107
data-title attribute, 43
default page transitions, altering, 188–189
defaultDialogTransition property, targeting, 188
defaultPageTransition property, targeting, 188
delay method, 13
Delete button, combining with Save, 80
dequeue method, 13
desktop simulators, testing with, 270. *See also* testing with simulators
detecting devices. *See* device detection
device detection
using JavaScript, 257–260
using PHP, 254–256
dialog transitions
altering, 188–189
pop, 188
dialog windows
creating, 62–68
as external webpages, 64–65
flip transition, 68
max-width, 67
multipage, 62–63
overwriting widths, 67
pop transition, 68
as pop-up windows, 66–68
single-page, 65
slidedown transition, 68
transitions, 68
.ui-dialog classes, 67
dialog-window.html file, 64–65
display, responsive design, 196
displaying content, considering, 23
div.foo jQuery object
binding click event to, 8–9
clicking, 13
<!DOCTYPE> declaration, 20
document.ready event, using with touch events, 193
documents, waiting for loading, 11–12
Drupal. *See also* CMS (content management system); WordPress
adding content, 248–249
blog posts, 248
creating pages, 248–249
custom settings, 247
database configuration, 240
defining front page, 248–249

downloading, 238
installing, 238–241
installing jQuery Mobile theme, 246
jquery directory, 243–244
jQuery Mobile module, 242–245
jQuery Mobile theme, 246
language selection, 239
libraries directory, 243
node ID, 248–249
profile selection, 239
Sequel Pro for Mac OS, 238
site configuration, 240
theming with jQuery Mobile, 242–247

E

email folders, using count bubbles with, 116–117
`emailAddress;` input, 215–216
emulators, testing with, 268
"Error Loading Page" message, 58
events. *See also* `mobileinit` event
binding to elements, 9
customizing, 11
managing, 8–11
orientation, 195–196
page transition, 200–203
scroll, 197–199
touch, 192–194
exposed methods. *See also* methods; utility methods
`changePage`, 210–216
`data-hyperlink` attribute, 211
explained, 209
loading pages, 217–218
external links, using, 49–50

F

`fade` transition, using, 55
`fieldset`, `controlgroup` `data-role`, 148
`filterCallback` option, using with search filters, 126
filtering items, 123
filter.js file, using, 126–128
Firefox, Modify Headers add-on, 269
flip toggle switches, 154–155
making, 154
versus sliders, 155
`flip` transition
using, 55, 57
using with dialog windows, 68
font styling, adding, 92–93
footer component, applying swatch to, 162
footer toolbars
markup, 82
using, 82
form element theming, 165–170. *See also* content theming; theming components
form elements
anchor with `id` of `next-btn`, 213–214
applying `data-theme` to, 166
changing swatches on, 166
checkboxes, 141–145
`emailAddress;` input, 215–216
flip toggle switches, 154–155
radio buttons, 141–145
select menus, 146–151
sliders, 152–154
text inputs, 134–140
functions
managing, 8–11
versus methods, 10

G

gear icons, using with split button lists, 113
global options
active button classes, 184–186
active page classes, 184–186
customizing namespaces, 178–179
customizing `subPageUrlKey`, 183
default page transitions, 188–189
delaying page initialization, 180–182
dialog transitions, 188–189
disabling Ajax navigation, 187
enabling Ajax navigation, 187
extending `mobileinit` event, 176–177

grid columns
adding custom CSS to, 93–94
block-title custom class, 93–94
ui-bar-a class prefix, 92, 98
ui-block class prefix, 90
ui-grid-a class prefix, 90
grid rows, 97–98
grids
columns, 90–97
creating, 90
CSS class, 90
described, 90
rows, 97–98
toolbar layouts, 95–97
two-column layout, 90–91
ui-grid-c class, 97

H

hashtags, managing history with, 45–46
header component, applying swatch to, 162
header element, adding Save button to, 78–80
header toolbars
CSS classes, 77
grouping buttons, 80
heading element, 77
markup, 77
navbars in, 85–86
with page title, 76
ui-title class, 77
without page title, 76
"Hello world!" blog post, accessing, 232
history, managing with hashtags, 45–46
home data icon, using, 71–72
home page, setting in WordPress, 235
HTML elements. *See also* semantic HTML
accessing by ID, 7
.class selector, 8
scope, 9
selecting, 7–8
targeting, 8
$(this), 9–10, 12
HTML files
anchors, 46
single-page templates, 45
using on webpages, 35
HTML5, data attributes. *See* data- attributes
HTML5 template. *See also* multipage template; single-page template
adding framework to, 31
creating, 20
<!DOCTYPE> declaration, 20
viewport meta tag, 22
hyperlinks
applying transition effects to, 189
converting to buttons, 69
markup, 48
markup for Ajax, 47
pound sign (#) in, 46

I

icons
adding to list items, 118–119
customizing, 72
including on buttons, 71
id value, referencing, 44
image tags, adding data- attributes to, 24
input elements, using buttons with, 70
inset lists, creating, 111
installing
Drupal, 238–241
JQuery Mobile module, 242–245
WordPress, 228–230
internal links, using, 49–51

J

JavaScript
device detection, 257–260
match() method, 258
navigator object, 257
regular expressions, 257–259
switch statement, 258–259
user agent information, 257–258
JavaScript Object Notation (JSON), 213

jqm-wp database, creating, 228
jQuery framework
  cross-browser compatibility, 7
  described, 5
  downloading, 6
  selecting HTML elements, 7–8
`jquery()` function, using with HTML elements, 7
jQuery Mobile framework
  adding to HTML5 template, 31
  adding to websites, 30–31
  A-grade support, 47
  B-grade support, 47
  C-grade support, 47
  downloading packages, 30
  `ns` option, 25
  obtaining, 30
  referencing CDN-hosted files, 30
  `.support` method, 259–260
  version 1.1.0, 135
jQuery Mobile module
  configuring settings, 247
  downloading, 242–245
  jquery.custom directory, 244
  variables in settings.php, 244–245
jQuery Mobile theme
  custom settings, 247
  installing for Drupal, 246
jQuery Mobile widgets, 183. *See also* collapsible content widget; widgets
`jquerymobiletv-` namespace, customizing, 178–179
JSON (JavaScript Object Notation), 213

K

Keynote DeviceAnywhere remote lab, 269

L

`landscape` value, using with `orientation` property, 195
lazy loading effect, creating with scroll events, 197–199
link types, 48
  for Ajax, 47
  disabling Ajax, 47, 49–50
  external, 49–50
  internal, 49–51
linked lists
  `data-theme` attribute, 107
  described, 106
  inset, 111
  `listview` value, 107
  nested, 109–110
  numbered, 109
  pointer cursor, 107–108
  `ui-btn` class, 107–108
  `ui-btn-icon-right` class, 108
  `ui-listview` class, 107
list dividers
  adding themes to, 168
  creating, 114
  using, 116–117
list items, filtering, 130–131
lists
  count bubbles, 115–119
  customizing, 112–119
  icons, 118–119
  linked, 106–111
  search filter bars, 122–123
  split button, 112–113
  thumbnails, 117–118
`listview`
  adding `id` for, 126–128
  count bubble theme, 169
  `data-divider-theme`, 168
  `data-split-theme`, 170
  `data-theme` in, 166–167
  with “e” swatch `data-theme`, 167
  using with search filter text, 125–128
`listview` theme
  customizing, 167
  overriding with list items, 167
loading messages, customizing, 58
loading webpages, 197–199
`loadingMessage` property, setting, 58

loadPage method
markup, 219
url argument, 218
using, 53, 217–220
loadPage method arguments
data, 53
loadMsgDelay, 53
pageContainer, 53
reloadPage, 53
role, 53
type, 53
loadPage properties
data, 218
loadMsgDelay, 218
pageContainer, 218
reloadPage, 218
role, 218
type, 218
logo widget
creating, 204–205
height option, 204–205
image option, 204–205
width option, 204–205

M

messages
loading, 58
suppressing display of, 58
methods versus functions, 10. *See also* exposed methods; utility methods
minification, benefit of, 6
Mob4Hire crowd testing, 271
mobile devices, list of, 254
$.mobile object
activePageClass property, 184
extending, 180
loadPage method, 217–220
mobileinit event, 177
subpageUrlKey property, 183
$.mobile property, ns, 177
mobile simulators
finding online, 265–267
Mobilizer, 270
online, 265–267
using, 264
mobile web, developing for, 264
mobile webpages. *See also* webpages
Chrome Developer Tools, 37
page data-role attribute, 35–36
structuring, 34–37
using separate HTML files, 35
XMLHttpRequest, 36–37
Mobile Websites 4U simulator, 267
mobile_detection.php file, 256
$.mobile.base utility method, 220
mobile-device detection
using JavaScript, 257–260
using PHP, 254–256
$.mobile.activepage property, 221
$.mobile.fixedToolbars.hide utility method, 220
$.mobile.fixedToolbars.show utility method, 220
$.mobile.hidePageLoadingMsg utility method, 220
mobileinit event. *See also* events
$.mobile object, 177
anonymous callback function, 176
custom JavaScript handler, 176
extending, 176–177
overriding global options, 177
overriding properties, 177
setting properties, 177
updating properties, 177
using with search filter text, 124–125
$.mobile.path.isAbsoluteUrl utility method, 220
$.mobile.path.isRelativeUrl utility method, 220
$.mobile.path.isSameDomain utility method, 220
$.mobile.path.makePathAbsolute utility method, 220
$.mobile.path.makeUrlAbsolute utility method, 220

$.mobile.path.parseUrl utility method, 220
authority property, 221
directory property, 221
domain property, 221
filename property, 221
hash property, 221
host property, 221
hostname property, 221
href property, 221
hrefNoHash property, 221
hrefNoSearch property, 221
password property, 221
pathname property, 221
port property, 221
protocol property, 221
search property, 221
username property, 221
mobilephoneemulator.com
layout, 265
menu and choices, 265
user agent string, 265
$.mobile.showPageLoadingMsg utility method, 220
$.mobile.silentScroll utility method, 220
Mobilizer
preview tool, 21
simulator, 270
Mozilla Firefox, Modify Headers add-on, 269
multipage template, 34–35, 42–43. *See also* HTML5 template; single-page template
data-title attribute, 43
linking from single-page template, 46
using with dialog window, 62–63

N

name collisions, preventing, 178
namespaces. *See* custom namespaces
- (dash) at end of, 178
CSS selectors, 179
customizing, 24–26, 178–179
described, 178
navbar value, using with data-role, 85
navbars
with button icons and logo in header, 86
in header toolbars, 85
and logos in headers, 86
with wrapping buttons, 85
navigation bars
creating, 85–86
wrapping, 85
nested lists, creating, 109–110
ns option
$.mobile property, using, 177
using with data- attributes, 25
number patterns, using, 135
numbered lists, creating, 109
numeric keyboard, 134

O

online emulators, testing with, 268
online simulators
Cowemo, 265–266
finding, 265–267
Mobile Websites 4U, 267
Opera Mini, 266–267
on/off functionality, adding, 154–155
Opera Mini online simulator, 266–267
organizing information, 100
orientation property
landscape value, 195
portrait value, 195
orientationchange event, firing, 195–196

P

Paca Mobile Center remote lab, 269
page classes, using, 184–186
page components, 33–34, 161
page data-role attribute, 35–36
page initialization
autoInitializePage property, 180, 182
delaying, 180–182
events, 206
pagecreate event, 204–206

page initialization *(continued)*
pagecreatebefore event, 204–206
pageinit event, 204–206
page theming component, 161–162
page titles, including in header toolbars, 76
page transitions. *See also* webpages
attaching event handlers, 200–201
binding event handler, 55–56
code sample, 56–57
defaultPageTransition property, 55
delegate method, 201–202
disabling, 56
events, 200–203
fade method, 55
flip method, 55, 57
live method, 201
on method, 200–201
mobileinit event, 57–58
overriding default, 56
pagebeforehide, 200–203
pagebeforeshow, 200–203
pagehide, 200–203
pageshow, 200–203
pop method, 55
reverse, 56
setting up, 55
slide method, 55
slidedown method, 55
slideup method, 55
page2.html file
loading, 219
loading silently, 218
pagebeforehide event, 200–203
pagebeforeshow event, 200–203
pagecreate event, 204–206
pagecreatebefore event, 204–206
pagehide event, 200–203
pageinit event
stage in initialization process, 204–206
using with search filters, 126–128
pageLoadErrorMessage property, setting, 58
pages. *See also* mobile webpages
activeBtnClass property, 186
adding to WordPress, 233–235
altering default transitions, 188–189
caching, 52–54
changing programmatically, 210–216
creating in Drupal, 248–249
horizontal display, 195–196
linking, 44
loading, 197–199
loading silently, 217–220
loading without displaying, 217–220
preloading, 52–54
referencing id value, 44
as separate pages, 44
single-page template, 42
slide default transition, 188
.support properties, 260
ui-page-active element, 184
user agent information, 254
vertical display, 195–196
viewing in WordPress, 234
waiting for loading, 11–12
pageshow event, 200–203
Perfecto Mobile remote lab, 269
PHP
browser identification, 255–256
/i used with regular expressions, 254–255
mobile detection function, 256
mobile_detect() function, 254
OR (|) regular expression, 254–255
preg_match() function, 255
regular expressions, 255
testing search strings, 255
URL redirection, 256
user agent information, 254
pointer cursor, adding to linked lists, 107–108
pop transition
altering, 188
using, 55
using with dialog windows, 68
portrait value, using with orientation property, 195
pound sign (#), including in hyperlinks, 46
preloading pages, 52–54

Q

queue methods
  clearQueue, 13–14
  delay, 13
  dequeue, 13
  queue, 13
  using, 14

R

radio buttons, 141–145
  applying mini, 145
  versus checkboxes, 141
  controlgroup data-role groups, 142
  disable method, 145
  enable method, 145
  events, 145
  horizontal toggle set, 143–144
  legend in fieldset, 142
  questions, 142
  refresh method, 145
  setting theme, 145
  statements, 142
  versus text inputs, 145
ready event, using with documents, 11–12
regular expressions
  in JavaScript, 257–259
  learning about, 255
  library of, 255
  testing, 255
  websites, 255
remote labs
  Keynote DeviceAnywhere, 269
  Paca Mobile Center, 269
  Perfecto Mobile, 269
responsive design, considering, 196
Roberts, Francis, 232

S

Save button. *See also* buttons
  adding to header element, 78–80
  combining with Delete, 80
scripts, including, 6
scroll events
  lazy loading effect, 197–199
  scrollstart, 197
  scrollstop, 197
SDKs (software development kits)
  Android, 270
  benefits, 270
  BlackBerry, 270
  XCode, 270
search filter bar, creating, 122–125
search filter text
  changing, 124
  custom formatted listview, 125–128
  mobileinit event, 124–125, 130
  updating, 124
search filters
  custom callback function, 129–130
  customizing, 126–131
  defaultSearch function, 128–129
  filterCallback option, 126
  id for listview, 126–128
  logging text, 129
  pageinit event, 126–128
  testing with searchValue, 129
select menus, 146–151
  action sequence, 146
  controlgroup in, 149
  corners option, 150
  in fieldcontain, 147
  formatting, 147
  grouping, 148–149
  horizontally grouped, 149
  icon option, 150
  iconpos option, 150
  iconshadow option, 150
  initSelector option, 150
  inline option, 150
  methods, 151
  mini option, 150
  nativeMenu option, 150
  open method, 151
  options, 150
  overlayTheme option, 150

select menus *(continued)*
preventFocusZoom option, 150
shadow option, 150
theme option, 150
semantic HTML, 20. *See also* HTML elements
Sequel Pro for Mac OS
downloading, 228
using with Drupal, 238
using with WordPress, 228
simulators, 265. *See also* testing with simulators
finding online, 265–267
Mobilizer, 270
online, 265–267
using, 264
single-page template. *See also* HTML5 template; multipage template
Ajax-based navigation, 45
described, 45
dialog windows, 65
hashtags, 45–46
history, 45–46
linking to multipage template, 46
preloading pages, 54
slide transition
altering, 188
using, 55
slidedown transition
using, 55
using with dialog windows, 68
sliders, 152–154
creating, 152
disable method, 154
disabled option, 153
enable method, 154
events, 154
fill highlight, 153
versus flip toggle switches, 155
highlight option, 153
methods, 154
options, 153
refresh method, 154
trackTheme option, 153
slideup transition, using, 55
special effects, applying, 12–14
split button lists. *See also* buttons
changing default, 113
creating, 112–113
with gear icons, 113
spoofing user agent strings, 269
step2.php page, emailAddress input, 215–216
Submit button, 70
subpages, referencing, 183
subPageUrlKey
customizing, 183
ui-page parameter, 183
.support method
ajax feature, 260
cors feature, 260
detecting browser features with, 259–260
opacity feature, 260
properties list, 260
submitBubbles feature, 260
swatches. *See also* ThemeRoller tool
a-e, 158–159, 162–163
applying to footer component, 162
applying to header component, 162
changing on form elements, 166
copying, 159
customizing, 158, 160
defaults, 158
updating, 158
using, 158–160
swipe event, described, 192
swipe event properties
durationThreshold, 192
horizontalDistanceThreshold, 192
scrollSupressionThreshold, 192
verticalDistanceThreshold, 192
swipeleft touch event, 192
swiperight touch event, 192

T

tap touch event, 192
taphold touch event, 192–193
telephone keypad, 134, 136

testing. *See also* crowd testing
on desktop, 264
with desktop simulators, 270
with online emulators, 268
testing with simulators. *See also* desktop simulators; simulators
online emulators, 268
options, 264
pros and cons, 263
remote labs, 269
text inputs, 134–140
alphabetical keyboard, 135–136
`bind` method for events, 140
versus checkboxes, 145
`clearSearchButtonText` option, 139
disabled element, 139
events, 140
`initSelector` option, 137
methods, 139
number patterns, 134–135
numeric keyboard, 134
options, 137–139
`preventFocusZoom` option, 138–139
versus radio buttons, 145
regular and `mini`, 138
setting element types, 134
telephone keypad, 134
`theme` option, 137
`type` set to number, 134
`textinput` element
`disable` method, 139
`enable` method, 139
theme option, 138
using, 137
theme dividers, `data-divider-theme`, 168
theme forms, creating, 165–170
theme swatches
a-e, 158–159, 162–163
applying to footer component, 162
applying to header component, 162
changing on form elements, 166
copying, 159
customizing, 158, 160
defaults, 158
updating, 158
using, 158–160
ThemeRoller tool, 160. *See also* swatches
themes
adding to WordPress, 231–232
defining, 158
jQuery Mobile for Drupal, 246
theming components. *See also* content theming; form element theming
buttons, 161–162
content, 163–164
lists, 166–170
page, 161–162
toolbar, 161–162
`$(this)` element, using, 9–10, 12
thumbnails, adding to list items, 117–118
toggle switch, interaction sequence, 155
toolbar components, 33–34
toolbar theming component, 161–162
toolbars
fixed, 83
footers, 82
fullscreen, 83–84
headers, 76–77
positioning, 83–84
updating position of, 84
touch events
binding, 193–194
`document.ready` event, 193
`swipe`, 192
`swipeleft`, 192
`swiperight`, 192
`tap`, 192
`taphold`, 192–193
`.test-tap-hold` class, 193–194
transitions effects, applying to hyperlinks, 189
true/false functionality, adding, 154–155

U

`ui-bar-a` class prefix, using, 92, 98
`ui-block` class prefix, using, 90
`ui-btn-icon-right` class, 108

.ui-dialog classes, using, 67
ui-grid prefix, using, 90–91
ui-grid-a class prefix, using, 90
ui-grid-c class prefix, using, 97
ui-li-count class, using, 115
ui-li-icon class, using, 118–119
ui-listview class, using, 107
ui-page-active element, 184
ui-title class, using with header toolbars, 77
url argument, using with loadPage method, 218
user agent information, obtaining, 254
user agent strings, spoofing, 269
user profile, modifying, 78–80
utility methods. *See also* exposed methods; methods
  $.mobile.activepage property, 221
  jqmData, 220
  jqmRemoveData, 220
  $.mobile.base, 220
  $.mobile.fixedToolbars.hide, 220
  $.mobile.fixedToolbars.show, 220
  $.mobile.hidePageLoadingMsg, 220
  $.mobile.path.isAbsoluteUrl, 220
  $.mobile.path.isRelativeUrl, 220
  $.mobile.path.isSameDomain, 220
  $.mobile.path.makePathAbsolute, 220
  $.mobile.path.makeUrlAbsolute, 220
  $.mobile.path.parseUrl, 220–221
  $.mobile.showPageLoadingMsg, 220
  $.mobile.silentScroll, 220
  using, 220–222

V

vclick event, using bind method with, 46
versions, minified versus source, 6
viewport meta tag
  adding properties to, 22
  described, 21
  height property, 23
  including in HTML5 templates, 22
  initial-scale property, 22–23
  maximum-scale property, 23
  minimum-scale property, 23
  setting display with, 22
  user-scalable property, 23
  width property, 22–23

W

web hosts, choosing for WordPress, 228
webpages. *See also* mobile webpages; page transitions
  activeBtnClass property, 186
  adding to WordPress, 233–235
  altering default transitions, 188–189
  caching, 52–54
  changing programmatically, 210–216
  creating in Drupal, 248–249
  horizontal display, 195–196
  linking, 44
  loading, 197–199
  loading silently, 217–220
  loading without displaying, 217–220
  preloading, 52–54
  referencing id value, 44
  as separate pages, 44
  single-page template, 42
  slide default transition, 188
  .support properties, 260
  ui-page-active element, 184
  user agent information, 254
  vertical display, 195–196
  viewing in WordPress, 234
  waiting for loading, 11–12
websites
  Cowemo online simulator, 265–266
  Firefox Modify Headers add-on, 269
  jQuery Mobile framework, 30
  Keynote DeviceAnywhere, 269
  Mob4Hire crowd testing, 271
  Mobile Websites 4U simulator, 267
  mobilephoneemulator.com, 265
  Mobilizer preview tool, 21

Mobilizer simulator, 270
Mozilla Firefox Modify Headers add-on, 269
online simulators, 265–267
Opera Mini online simulator, 266–267
Paca Mobile Center, 269
Perfecto Mobile, 269
regular expressions, 255
Sequel Pro for Mac OS, 228
simulators, 265–267
World Wide Web Consortium, 269
widgets, customizing, 204–206. *See also* collapsible content widget; jQuery Mobile widgets
WordPress. *See also* CMS (content management system) Drupal
adding themes to, 231–232
blog posts, 233–235
changing themes, 232
creating webpages, 233–235
creating websites, 228
data storage, 228
database configuration variables, 229
database connection, 229
database login data, 229
`DB_HOST` variable, 229
`DB_NAME` variable, 229
`DB_PASSWORD` variable, 229
`DB_USER` variable, 229
described, 227
"Hello world!" blog post, 232
installing, 228–230
jqm-wp database, 228
`$jqtheme` value, 232
jQuery Mobile theme, 231
listing blog posts in, 234–235
saving blog posts, 233
Sequel Pro for Mac OS, 228
setting home page, 235
viewing blog posts, 233
viewing webpages, 234
web hosts, 228
wp-config.php file, 229
World Wide Web Consortium, 269
wp-config.php file, contents of, 229
WWW box model, adhering to, 259–260

X

XCode SDK, 270
XMLHttpRequest, using with webpages, 36–37

Y

yes/no functionality, adding, 154–155

Unlimited online access to all Peachpit, Adobe Press, Apple Training and New Riders videos and books, as well as content from other leading publishers including: O'Reilly Media, Focal Press, Sams, Que, Total Training, John Wiley & Sons, Course Technology PTR, Class on Demand, VTC and more.

No time commitment or contract required! Sign up for one month or a year. All for $19.99 a month

SIGN UP TODAY

peachpit.com/creativeedge

creative edge