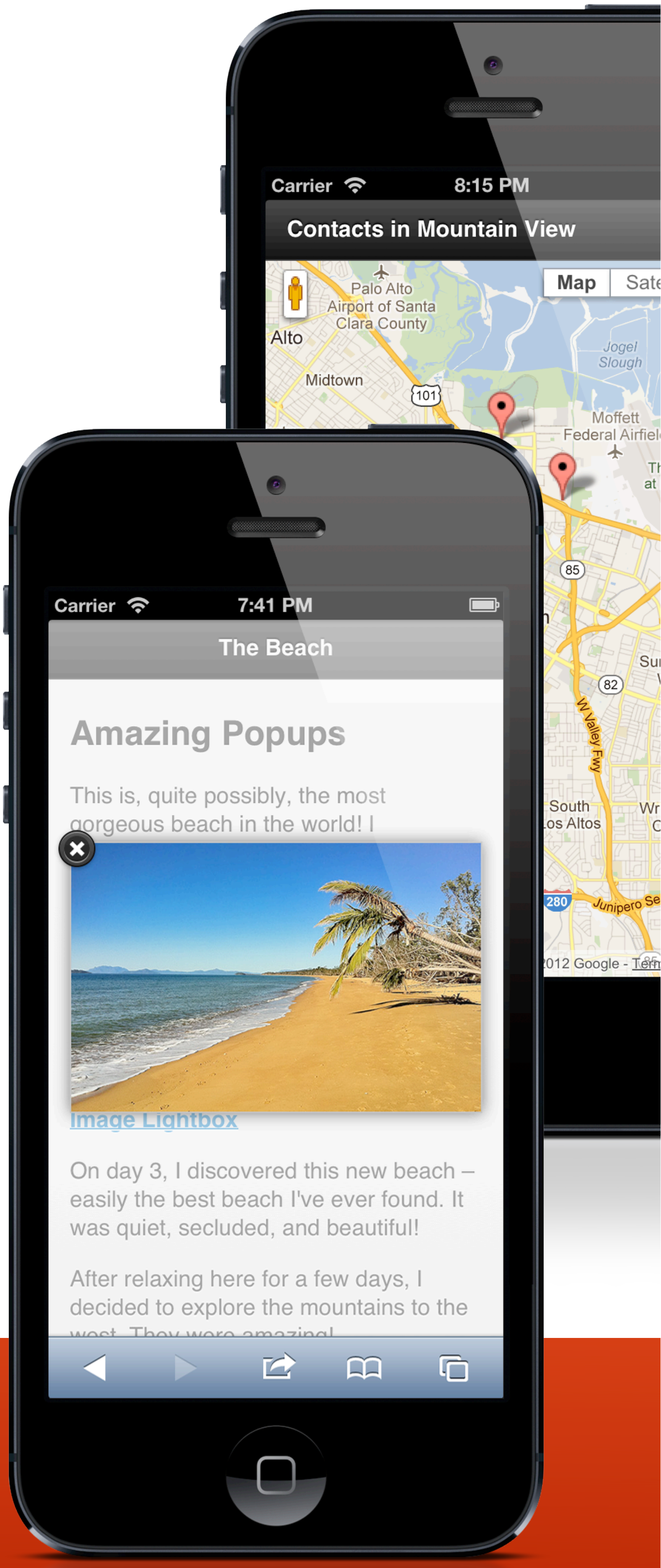
Quickly learn to
create great
mobile web apps!

Master Mobile Web Apps with jQuery Mobile

Third Edition



Amazing Popups

This is, quite possibly, the most gorgeous beach in the world! I

Image Lightbox

On day 3, I discovered this new beach – easily the best beach I've ever found. It was quiet, secluded, and beautiful!

After relaxing here for a few days, I decided to explore the mountains to the west. They were amazing!

Matt Doyle

Elated Books

Master Mobile Web Apps with jQuery Mobile

Third Edition

Matt Doyle

Elated Communications

www.elated.com

Master Mobile Web Apps with jQuery Mobile (Third Edition)

by Matt Doyle

Published by
Elated Communications
PO Box 3313
Robertson
NSW 2577
Australia

www.elated.com

ISBN: 978-0-9873115-0-4

Publishing History:

August 2011:	First Edition.
December 2011:	Second Edition.
October 2012:	Third Edition.

Elated and the Elated logo are trademarks of Elated Communications. All other trademarks mentioned in the book are the property of their respective owners. Rather than using a trademark symbol with every trademarked name in the book, the names are used merely in an editorial fashion and to the benefit of the trademark owner. No intention of infringement of the trademark is intended.

Elated Communications is not associated with any product, service or vendor mentioned in this book.

While they have taken care in the preparation of this book, the author and publisher make no expressed or implied warranty of any kind, and assume no responsibility for errors or omissions. The author and publisher assume no liability to any person or entity with respect to any loss or damage caused or alleged to be caused, directly or indirectly, by the information contained in this book, or by the information and software code associated with this book.

About the Author

Matt Doyle is an experienced technical author and coder who has written two well-received books on Photoshop and PHP.

He has also written articles for Elated.com and SitePoint on a variety of topics, including PHP, CSS, JavaScript and, of course, jQuery Mobile.

Acknowledgements

Several people have helped tremendously with this book. First of all, my wife Cat (http://naturalbloom.com/), who has done a fantastic job helping with the book's production and marketing. Secondly, Todd Parker and other members of the jQuery Mobile team (http://jquerymobile.com/), who have been a great help with getting this book off the ground. Thirdly, Simon Meek (http://www.simonmeek.com/), who gave me a lot of useful ideas and feedback with the book, and who also designed the book's cover.

Last, but certainly not least, I'd like to thank everyone who purchased the first and second editions of this book, and made it such a great success. Your support and feedback have made it all worthwhile, and have made this third edition possible. So thank you!

Table of Contents

Preface

Thank you for buying this book, and welcome to the world of jQuery Mobile and mobile web apps! In this book you'll quickly learn how to build great-looking, easy-to-use mobile web applications using this fantastic framework.

Who This Book Is For

This book is intended for anyone interested in building mobile web applications using the jQuery Mobile framework. You'll need at least a basic knowledge of web technologies including HTML, CSS, JavaScript and web servers.

In addition, some chapters — especially Chapters 11 to 13 — assume you have some experience of the jQuery JavaScript library on which jQuery Mobile is built, including jQuery selectors and events. Chapter 12 also includes a lot of PHP code, so some knowledge of PHP will be helpful when following through the examples. In addition, Chapter 13 shows how to use Xcode on a Mac to build a native iOS app using jQuery Mobile, so you'll find it useful to have at least some familiarity with Mac applications, and you'll need a Mac if you want to work through the example.

That said, even if you've never played with jQuery or PHP before, you'll still be able to gain a lot from this book. One of jQuery Mobile's strengths is that you can often build an entire web app interface using little more than HTML and a bit of CSS.

What's In the Book

In this book you'll explore many areas of jQuery Mobile, including:

- The nature of mobile web apps

- How jQuery Mobile fits into the web app development process
- How to create a basic mobile site using jQuery Mobile
- The details of building mobile user interfaces with jQuery Mobile, including elements such as pages, dialogs, popups, buttons, lists and forms
- jQuery Mobile's theming system, which lets you create your own unique look and feel for your web apps
- The more advanced features of jQuery Mobile's API, including changing default settings and working with events and methods, and
- How to build two complete mobile web apps using various technologies, including jQuery Mobile, JavaScript, PHP, PhoneGap and the Google Maps API.

The Code Zip File

Along with this book, you should have received a `jquery-mobile-book-code.zip` file containing most of the code examples shown in the book.

In the archive, you'll find a list of folders named after the chapters in the book. Each folder contains the code examples for that chapter. For most of the examples, you can simply open the example file in your mobile or desktop browser to see the example in action.

With the Task Tango example app in Chapter 12, you need to install the files on an Apache web server that also has PHP and MySQL installed, as per the instructions in the chapter. You can compile Chapter 13's CityChums app and run it in the iOS Simulator if you have a Mac and Xcode, or you can simply open the app's `index.html` file in a mobile or desktop browser.

The Book Companion Site

This book has a companion site where you can find out about recent updates, look for corrections, and play with the Task Tango demo app. Visit the companion site at:

http://store.elated.com/products/jquery-mobile-book/

You'll also find links to send feedback to the author (always welcome!), as well as to the Elated forums, where you can ask for technical help on the topics covered in the book.

What's New in the Third Edition

The second edition of this book — published in December 2011 — covered jQuery Mobile 1.0. This third edition is fully updated to cover all the new features and changes introduced in jQuery Mobile 1.1 and 1.2, including:

- **Slider improvements**, such as `step` attribute support, a new `data-highlight` attribute, and new `slidestart` and `slidestop` events. See Chapter 7 for details.
- **True fixed toolbars** using the `position: fixed` CSS property, which results in a much smoother user experience. See Chapter 5 for details.
- **Improved transitions** that incorporate a fade for much faster, smoother animations, as well as two new transitions: `turn` and `flow`. See Chapter 3 for details.
- **A new popup widget**, which is both simpler and more flexible than the dialog widget. Great for anything from tooltips to image lightboxes. See Chapter 6 for details.
- **A new loading spinner** that looks great and is easily customizable. See Chapter 11 for details.

- **Mini form elements** using the new `data-mini="true"` attribute, for when you need to save space. See Chapter 7 for details.
- **An improved flip switch design** that is cleaner and more compact. See Chapter 7 for details.
- **A revamped ThemeRoller tool** for building your own themes, including undo/redo, multiple jQuery Mobile version support, and seamless upgrades. See Chapter 10 for details.
- **A new download builder tool** that lets you include just the jQuery Mobile modules you need for your app. See http://jquerymobile.com/download-builder/ for details.
- **Listview improvements** including collapsible listviews, listview autodividers, an improved read-only listview design, support for the `start` attribute on `ol` listviews, and a new `listviewbeforefilter` event for filtered listviews. See Chapter 8 for details.
- **Form element visual tweaks** so that elements align evenly and consistently.
- **Custom button icons for collapsibles** using the `data-collapsed-icon` and `data-expanded-icon` attributes. See Chapter 9 for details.
- **A cleaner default theme** that looks neater and more consistent.
- **Many more improvements,** including support for the latest jQuery library (1.8.2 at the time of writing); searchable online documentation; larger icon-only buttons in control groups; swipe gesture tweaks; `data-enhance` and `data-ajax` attributes that can exclude entire content containers from enhancement; iOS page zooming fixes; performance improvements; and lots more!

In addition, the CityChums app in Chapter 13 has been fully updated for PhoneGap 2.1, Xcode 4.5 and the new 4-inch display on the iPhone 5, and has also been tweaked to make it run more smoothly.

Over the last couple of years, jQuery Mobile has steadily evolved into a stable, feature-rich mobile web framework. I hope you enjoy reading this book, and using jQuery Mobile to build some fantastic mobile web apps and websites!

Spread the Word

If you enjoy reading this book and find it useful, please help us spread the word about the book. Tell your friends and colleagues about it, mention it on your blog, tweet or facebook it — whatever you can do to help, we really appreciate it! Please link to this URL:

http://store.elated.com/

Thank you!
Matt and the Elated Team

Chapter 1 is omitted from this preview.

2

A Quick Tutorial

Like many things, the easiest way to understand jQuery Mobile is to start using it. To that end, we'll kick things off with a very simple example website, built using jQuery Mobile. This website will have the following components:

- A home page
- A list of products that the visitor can browse through
- A contact form, with a server-side PHP script, that visitors can use to contact the webmaster

While this website is not a "web app" in the common sense of the term — it doesn't have much interactivity, apart from the contact form — it does give you a good idea as to what's possible with jQuery Mobile. In fact, it's perfectly feasible to use jQuery Mobile to build simple mobile websites like this one, as well as more fully-featured web apps.

By the time you've worked through this chapter, you'll:

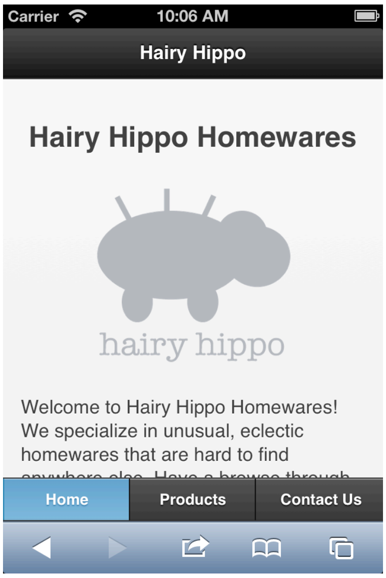
- Know how to install and use jQuery Mobile
- Have a basic understanding of building jQuery Mobile pages, and
- Be familiar with a few of the widgets that jQuery Mobile has to offer.

Creating the Home Page

Let's start by creating the home page for our mobile site. The home page will include:

- The company name ("Hairy Hippo Homewares") and logo
- Some welcome text
- A footer navigation bar with links to the home page, product list, and contact form

Figure 2-1 shows how the finished home page looks.



Welcome to Hairy Hippo Homewares! We specialize in unusual, eclectic homewares that are hard to find

Figure 2-1: The Hairy Hippo home page.

Here's the markup for the home page. Save it in a file called **`index.html`** in a folder within your website:

```
<!doctype html>
<html>
<head>
  <title>Hairy Hippo</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1,
maximum-scale=1">
  <link rel="stylesheet" href="http://code.jquery.com/mobile/1.2.0/
jquery.mobile-1.2.0.min.css" />
  <script type="text/javascript" src="http://code.jquery.com/
jquery-1.8.2.min.js"></script>
  <script type="text/javascript" src="http://code.jquery.com/mobile/1.2.0/
jquery.mobile-1.2.0.min.js"></script>
</head>
<body>

  <div data-role="page" id="home">

    <div data-role="header">
      <h1>Hairy Hippo</h1>
    </div>

    <div data-role="content">

      <div style="text-align: center;">
        <h2>Hairy Hippo Homewares</h2>
        <img src="images/logo.png" width="200" alt="Logo">
      </div>

      <p>Welcome to Hairy Hippo Homewares! We specialize in unusual,
eclectic homewares that are hard to find anywhere else. Have a browse
through our products, and have fun!</p>

    </div>

    <div data-role="footer" data-position="fixed" data-id="nav">
      <div data-role="navbar">
        <ul>
          <li><a href="#home" class="ui-btn-active ui-state-persist">Home</
a></li>
          <li><a href="#products">Products</a></li>
          <li><a href="#contact">Contact Us</a></li>
        </ul>
      </div>
    </div>

  </div>
```

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

As you can see, the site's home page is essentially a standard HTML page. The nice thing about jQuery Mobile is that you can create your mobile pages using regular HTML and a few additional attributes — no JavaScript coding required!

There are, however, a few differences compared to a regular HTML document. For one thing, we've included the `viewport meta` tag, as well as the jQuery Mobile theme CSS file, the jQuery script and the jQuery Mobile script, in the document's `head` section.

> *See "Installing jQuery Mobile" in Chapter 1 for more info on using the* `viewport meta` *tag and the jQuery Mobile CSS and JavaScript files.*

What's more, we've wrapped the page content in a special `div` with an attribute of `data-role="page"`, and given it an `id` of `"home"` so that we can refer to it later. This technique lets us create multiple mobile "pages" within a single HTML document, as we'll see in a moment.

Within the `data-role="page" div`, we've created further `div`s for the page header (`data-role="header"`), content area (`data-role="content"`), and page footer (`data-role="footer"`). The header contains the page title; the content area contains the site name, logo and intro text; and the footer contains a navigation bar with links to all three pages of the site.

> *You'll find all the image files for the tutorial, along with the tutorial code files, inside the code zip file that came with the book.*

Let's take a closer look at the footer. We've added `data-position="fixed"` to the footer `div`. This ensures that the footer remains fixed at the bottom of the browser window as the user scrolls the page. Within the footer, we've created a `data-role="navbar" div`. A navbar is a special jQuery Mobile widget that lets you create a row of buttons in a header or footer bar. We've also made the navbar persist across all pages of the site by giving the footer `div` a `data-id="nav"` attribute. This means that the navbar will stay put while new pages slide into view. We'll see how this works later.

Within the navbar, we've created a list of links to the three site pages. jQuery Mobile automatically styles these links as buttons, since they're inside the navbar. The first link has a class of `ui-btn-active`, which gives it a blue highlight, as well as a `ui-state-persist` class, which ensures that the highlight remains when the user returns to the home page.

> *Don't worry too much about the nitty-gritty of the various* `div`*s and attributes in this tutorial. All will be revealed in Part II of the book.*

Creating the Products Page

Now that we've built our site home page, we're ready to create our products page. This will consist of:

- A list of two product categories: "Lamps" and "Kitchen Equipment"
- A list of products — with thumbnail images — inside each category
- The same footer navigation bar as the home page, with the Products item highlighted

Figure 2-2 shows the products page in action.

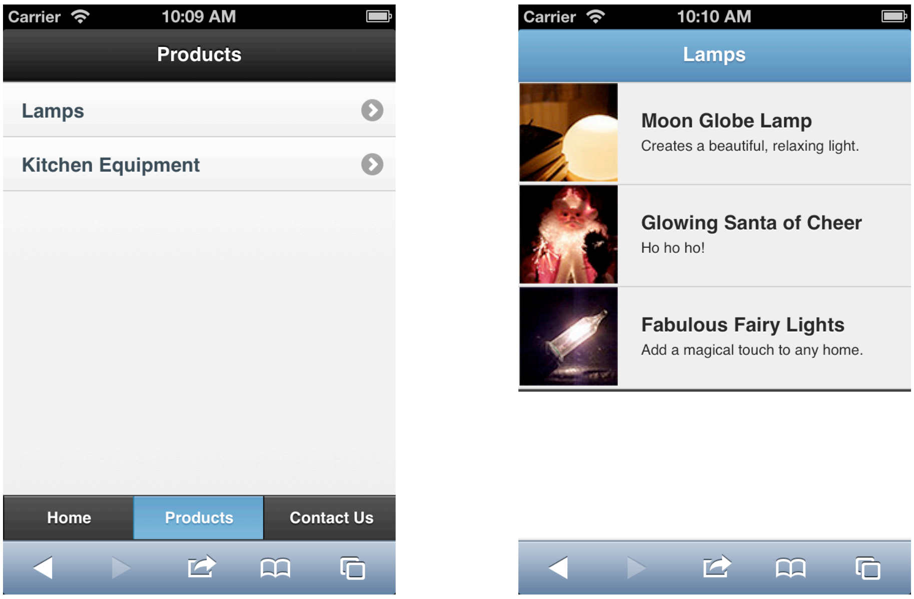


Figure 2-2: The Hairy Hippo products page containing a nested listview (left), and one of the nested lists (right).

jQuery Mobile has a nice feature that lets you embed several mobile "pages" within a single HTML page, just by creating additional `data-role="page"` `div`s. Each `data-role="page" div` should have its own unique `id` attribute so that you can link to it.

So let's add our products page as a new `data-role="page" div` within our existing `index.html` document:

```
  <div data-role="page" id="products">

    <div data-role="header">
      <h1>Products</h1>
    </div>

    <div data-role="content">
      <ul data-role="listview">
        <li>Lamps
          <ul>

            <li>
              <img src="images/products/thumbs/moon-globe-lamp.jpg"
alt="Moon Globe Lamp">
```

```
              <h2>Moon Globe Lamp</h2>
              <p>Creates a beautiful, relaxing light.</p>
            </li>

            <li>
              <img src="images/products/thumbs/glowing-santa.jpg"
alt="Glowing Santa of Cheer">
              <h2>Glowing Santa of Cheer</h2>
              <p>Ho ho ho!</p>
            </li>

            <li>
              <img src="images/products/thumbs/fairy-lights.jpg"
alt="Fabulous Fairy Lights">
              <h2>Fabulous Fairy Lights</h2>
              <p>Add a magical touch to any home.</p>
            </li>

          </ul>
        </li>
        <li>Kitchen Equipment
          <ul>

            <li>
              <img src="images/products/thumbs/milk-pan.jpg" alt="Magic
Milk Pan">
              <h2>Magic Milk Pan</h2>
              <p>Boils milk without boiling over!</p>
            </li>

            <li>
              <img src="images/products/thumbs/cafetiere.jpg" alt="Classy
Cafetière">
              <h2>Classy Cafetière</h2>
              <p>Beautiful, simple, and extra strong.</p>
            </li>

            <li>
              <img src="images/products/thumbs/whisky-glasses.jpg"
alt="Elegance Whisky Glasses">
              <h2>Elegance Whisky Glasses</h2>
              <p>Sample your favorite tipple in style!</p>
            </li>
          </ul>
        </li>
      </ul>
    </div>

    <div data-role="footer" data-position="fixed" data-id="nav">
      <div data-role="navbar">
        <ul>
          <li><a href="#home">Home</a></li>
          <li><a href="#products" class="ui-btn-active ui-state-
persist">Products</a></li>
          <li><a href="#contact">Contact Us</a></li>
        </ul>
      </div>
    </div>

  </div>
```

Let's break the above code down:

- **A new `data-role="page"` `div`.**
  Our whole products page is enclosed in a `div` with the `data-role="page"` attribute. This tells jQuery Mobile that we're creating a new mobile page within the document. We also give the `div` an `id` of `"products"` — this lets us link to the page using the URI `"#products"`.

- **The header.**
  As with the home page, we add a `data-role="header"` `div` that contains our page title.

- **The page content.**
  Again, we add a `data-role="content"` `div` for the page content.

- **The products list.**
  Within the content `div`, we create an unordered list containing our products. We add a `data-role="listview"` attribute to the list — this tells jQuery Mobile to create a special kind of list called a *listview* that has large, easy-to-tap list items. You can also nest lists in a listview, as we've done here: the

product categories are at the top level, while the products themselves are at the next level. jQuery Mobile automatically creates pages for each sublist, allowing the user to easily move around the nested lists.

In Chapter 8 you'll learn all about creating and formatting listviews.

- **The products.**
  Within each sublist, we create list items for the individual products (Moon Globe Lamp, Glowing Santa of Cheer, and so on). Each list item consists of a thumbnail image, the product name as an `h2` heading, and the product description as a paragraph. jQuery Mobile automatically styles the first `img` element in a listview's list item as an 80x80-pixel thumbnail.
- **The footer.**
  Finally, the products page has a `<div data-role="footer">` element, much like the home page. Notice that it has the same `data-id` attribute as the home page's footer; this tells jQuery Mobile that we want a persistent footer that stays put as the user navigates between pages. This time, we've highlighted the Products link instead of the Home link.

Creating the Contact Form

The last static page of our mobile site is a contact form. This will contain:

- A page heading
- Text fields for the visitor's name and email address
- A multiple `select` menu allowing the visitor to choose their product categories of interest
- A `textarea` field for the visitor's message
- A Send Email button

Figure 2-3 shows the finished page.

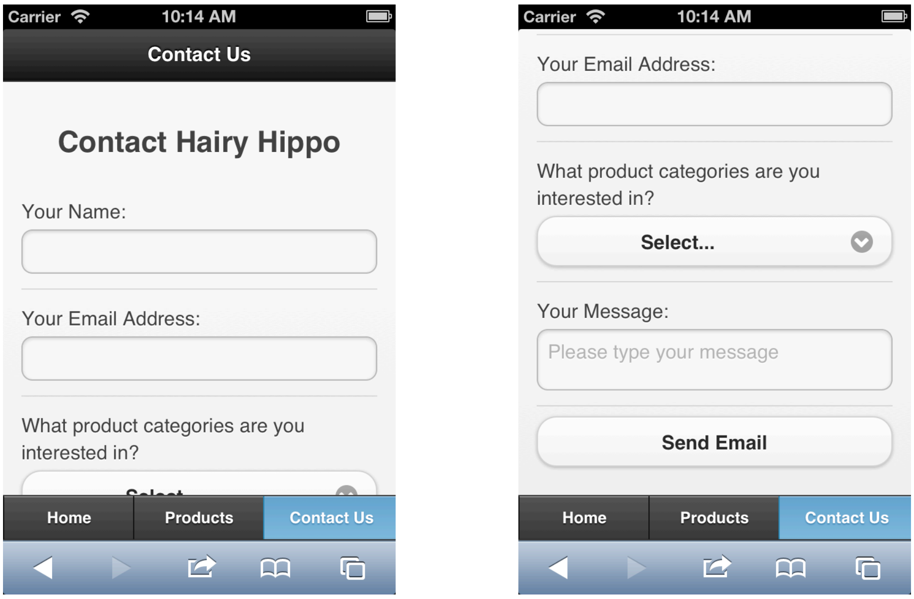


Figure 2-3: The contact form allows visitors to send emails to the site owner. Left: The top of the form; right: the bottom of the form.

As with the other two pages, we add this page as a `data-role="page" div` inside our existing HTML document, `index.html`. Here's the code:

```
<div data-role="page" id="contact">

  <div data-role="header">
    <h1>Contact Us</h1>
  </div>

  <div data-role="content">

    <h2 style="text-align: center;">Contact Hairy Hippo</h2>

    <form action="processForm.php" method="post" data-transition="pop">

      <div data-role="fieldcontain">
        <label for="senderName">Your Name:</label>
        <input type="text" name="senderName" id="senderName">
      </div>
```

```
        <div data-role="fieldcontain">
          <label for="senderEmail">Your Email Address:</label>
          <input type="email" name="senderEmail" id="senderEmail">
        </div>

        <div data-role="fieldcontain">
          <label for="productCategories">What product categories are you 
interested in?</label>
          <select name="productCategories[]" id="productCategories" 
multiple data-native-menu="false">
            <option value="">Select...</option>
            <option value="Lamps">Lamps</option>
            <option value="Kitchen Equipment">Kitchen Equipment</option>
          </select>
        </div>

        <div data-role="fieldcontain">
          <label for="message">Your Message:</label>
          <textarea name="message" id="message" placeholder="Please type 
your message" required="required" cols="80" rows="10" maxlength="10000"></
textarea>
        </div>

        <input type="submit" name="sendMessage" id="sendMessage" 
value="Send Email">

      </form>

    </div>

    <div data-role="footer" data-position="fixed" data-id="nav">
      <div data-role="navbar">
        <ul>
          <li><a href="#home">Home</a></li>
          <li><a href="#products">Products</a></li>
          <li><a href="#contact" class="ui-btn-active ui-state-
persist">Contact Us</a></li>
        </ul>
      </div>
    </div>

  </div>
```

As you can see, the form code looks much like a regular HTML form, with `input`, `select` and `textarea` elements. jQuery Mobile automatically restyles most form elements so that they're easier to use on mobile browsers. We've set the form to use the `post` method, and set the form's handler to be `processForm.php` (we'll create this PHP script next).

The `data-native-menu="false"` *attribute on the* `select` *element tells jQuery Mobile to pop up its custom overlay menu, instead of the native browser menu, when the user taps the field button. Not only does this menu look great, but it allows multiple selections, even in mobile browsers that don't natively support them. Find out more about select menus in Chapter 7.*

The key difference compared to a regular HTML form is that, when the user submits the form, jQuery Mobile's JavaScript automatically intercepts the submission and instead requests the form via Ajax. When the result page comes back from the server, jQuery Mobile inserts the result page's markup into the current page's DOM and displays it. This all happens automatically, with no JavaScript coding required on your part.

In fact, jQuery Mobile handles most page requests — not just form submissions — using Ajax. This approach has a number of advantages, such as allowing you to create fancy transition effects between pages. In fact, we've created just such a transition for our form by adding a `data-transition="pop"` attribute to the `form` element. This makes the result page appear to "pop" out of the window like a pop-up dialog.

In Chapter 3 you'll look at how to create different transition effects, while Appendix A explores jQuery Mobile's Ajax navigation system in detail.

At the end of our form page, we once again include a page footer, this time highlighting the Contact Us link.

Creating the Contact Form Handler

Nearly done! All that's left to do now is write our form mailer PHP script to handle submissions from our contact form and email the information to the webmaster. The script is fairly standard stuff — it reads the form values, composes and sends the email, and returns a response page to the visitor. Here's the code — save it as `processForm.php` in the same folder as your `index.html` file:

```
<?php

// Define some constants
define( "RECIPIENT_NAME", "John Smith" );
define( "RECIPIENT_EMAIL", "john@example.com" );
define( "EMAIL_SUBJECT", "Visitor Message" );

// Read the form values
$success = false;
$senderName = isset( $_POST['senderName'] ) ? preg_replace( "/[^\.\-\' a-
zA-Z0-9]/", "", $_POST['senderName'] ) : "";
$senderEmail = isset( $_POST['senderEmail'] ) ? preg_replace( "/[^\.\-\_
\@a-zA-Z0-9]/", "", $_POST['senderEmail'] ) : "";

$productCategories = array();

if ( isset( $_POST['productCategories'] ) ) {
  foreach ( $_POST['productCategories'] as $cat ) $productCategories[] =
preg_replace( "/[^\'\-\ a-zA-Z0-9]/", "", $cat );
}

$message = isset( $_POST['message'] ) ? preg_replace( "/(From:|To:|BCC:|
CC:|Subject:|Content-Type:)/", "", $_POST['message'] ) : "";

if ( $productCategories ) {
  $message .= "\n\n---\n\nInterested in product categories:\n\n";
  foreach ( $productCategories as $cat ) $message .= "$cat\n";
```

```
}

// If all values exist, send the email
if ( $senderName && $senderEmail && $message ) {
  $recipient = RECIPIENT_NAME . " <" . RECIPIENT_EMAIL . ">";
  $headers = "From: " . $senderName . " <" . $senderEmail . ">";
  $success = mail( $recipient, EMAIL_SUBJECT, $message, $headers );
}

// Return an appropriate response to the browser

?>
<!doctype html>
<html>
<head>
  <title>Thanks!</title>
  <meta charset="utf-8">
</head>
<body>

  <div data-role="page" id="contactResult">

    <div data-role="header">
      <h1>Hairy Hippo</h1>
    </div>

    <div data-role="content">

<?php if ( $success ) { ?>

      <div style="text-align: center;">
        <h2>Thanks!</h2>
        <img src="images/logo.png" width="200" alt="Logo">
        <p>Thanks for sending your message! We'll get back to you
shortly.</p>
      </div>

<?php } else { ?>

      <div style="text-align: center;">
        <h2>Oops!</h2>
        <p style="color: red">
```

```
          There was a problem sending your message. Please make sure you
fill in all the fields in the form.<br><br>
          <a href="#contact" data-rel="back" data-role="button">Try Again</
a>
        </p>
      </div>
<?php } ?>

    </div>

    <div data-role="footer" data-position="fixed" data-id="nav">
      <div data-role="navbar">
        <ul>
          <li><a href="#home">Home</a></li>
          <li><a href="#products">Products</a></li>
          <li><a href="#contact" class="ui-btn-active ui-state-
persist">Contact Us</a></li>
        </ul>
      </div>
    </div>

  </div>

</body>
</html>
```

> *If you want to try out this example, you'll need to set the* `RECIPIENT_NAME` *and* `RECIPIENT_EMAIL` *constants at the top of this script to your own name and email address.*

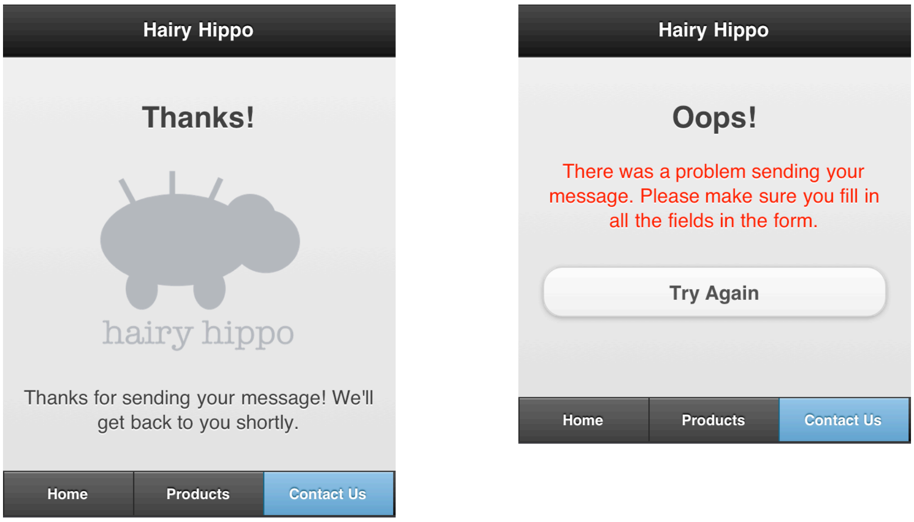
We won't go into the PHP code here, since it's outside the scope of the tutorial. The interesting bit from our point of view is the HTML response page returned by the PHP script, which includes some jQuery Mobile-specific markup. You can see this response page embedded in the PHP script, below the comment "Return an appropriate response to the browser".

First, the response page's content is enclosed in another `data-role="page"` `div`. jQuery Mobile looks for a `data-role="page"` container when it receives the response page, and displays the content that is inside the container.

Within the page content proper, we display either a success message (with logo), or a red failure message, as appropriate. The failure message includes a button (a link with the `data-role="button"` attribute) that the visitor can tap to return to the form. The button includes a `data-rel="back"` attribute, which makes jQuery Mobile emulate the browser's Back button if possible, rather than following the link and adding to the browsing history.

Finally, our response page includes the standard page footer, with the Contact Us button highlighted.

You can see the result of submitting both a valid and an incomplete form in Figure 2-4.



Thanks!

Thanks for sending your message! We'll get back to you shortly.

Oops!

There was a problem sending your message. Please make sure you fill in all the fields in the form.

Figure 2-4: The form handler returns either a success message to the user (left) or a failure message (right) as required.

Testing the Website

To try out the Hairy Hippo site yourself, simply open the `index.html` page in your mobile or desktop browser.

> *The easiest way to test a site in a mobile browser is to upload the site's files to a publicly-accessible web server, then open the site's URL in your mobile device's browser. You can also install the site on your local development web server, then use a wildcard DNS service such as http://xip.io so that your mobile device can find your local server. Alternatively, if you have a mobile device simulator installed on your computer, such as the iOS Simulator that comes with Xcode on the Mac, then you can simply drag the site's* `index.html` *file into the simulator window.*

Once you've opened the site in your browser, you should see the Hairy Hippo home page (Figure 2-1). Try tapping Products to view the product categories (Figure 2-2), then tapping a product category to view the products in the category. Use your Back button to return to the product categories, then tap Contact Us and try sending a message (Figures 2-3 and 2-4).

> *The message sending feature won't work if you're browsing the site directly from your hard drive. If you want the* `processForm.php` *script to send emails, you need to install the site files on a PHP-enabled web server so that the script can run, and set the* `RECIPIENT_NAME` *and* `RECIPIENT_EMAIL` *constants at the top of the PHP script.*

Congratulations — you've just built your first mobile site using jQuery Mobile!

Summary

In this brief introduction to jQuery Mobile, you've seen how to build a simple, yet fully-functional mobile site using nothing but jQuery Mobile, some HTML, and a smattering of PHP. Along the way you've touched on some important concepts of jQuery Mobile, including:

- How to construct mobile-friendly pages using jQuery Mobile
- How to include multiple jQuery Mobile "pages" in a single HTML page
- How page headers, content areas, and footers work
- How to use navbars to create navigation buttons in a page footer
- Some of jQuery Mobile's `data-` attributes, which you can use to control the look and behavior of page elements
- Listviews, which let you create good-looking, easy-to-navigate lists of items
- Creating a form and form handler that work with jQuery Mobile
- jQuery Mobile's Ajax-based page navigation system

Now that you understand the basic process of putting together pages in jQuery Mobile, you're ready to start delving into the nitty-gritty of jQuery Mobile in Part II.