Introduction to jQuery Mobile

C. Enrique Ortiz

February 01, 2011

This article provides an introduction to the jQuery Mobile framework. Learn the basics of the framework and how to write a functional mobile web application user interface without writing a single line of JavaScript code. An example guides you through basic pages, navigation, toolbars, list views, form controls, and transition effects.

Introduction

jQuery Mobile, a user interface (UI) framework, lets you write a functional mobile web application without writing a single line of JavaScript code. In this article, learn about the features of this framework, including the basic pages, navigation toolbars, form controls, and transition effects.

To follow along with this article, you will need:

- Previous exposure to HTML
- Understanding of JavaScript fundamentals
- Basic knowledge of jQuery

You can download the jQuery plug-in source code used with this article from the Download table below. Related topics has information on jQuery, JavaScript, DOM, HTML5, and more.

jQuery Mobile

jQuery Mobile is a touch-friendly web UI development framework that lets you develop mobile web applications that work across smartphones and tablets. The jQuery Mobile framework builds on top of jQuery core and provides a number of facilities, including HTML and XML document object model (DOM) traversing and manipulation, handling events, performing server communication using Ajax, as well as animation and image effects for web pages. The mobile framework itself is a separate, additional download of around 12KB (minified and gzipped) from jQuery core, which is around 25KB when minified/gzipped. As with the rest of the jQuery framework, jQuery Mobile is a free, dual-licensed (MIT and GPL) library.

Though jQuery Mobile is still in Alpha, there are some demos and documentation. It is recommended that you review the documentation and demos in Related topics and look at the demo source code in the Download section.
At the time of this writing, the jQuery Mobile framework is an Alpha 2 version (v1.0a2). The code is in draft form and is subject to change. Yet, the existing framework is pretty solid. With an impressive set of components available in the alpha release, jQuery Mobile promises to be a great framework and tool set for developing mobile web applications.

Basic features of jQuery Mobile include:

**General simplicity**  
The framework is simple to use. You can develop pages mainly using markup driven with minimal or no JavaScript.

**Progressive enhancement and graceful degradation**  
While jQuery Mobile leverages the latest HTML5, CSS3, and JavaScript, not all mobile devices provide such support. jQuery Mobile philosophy is to support both high-end and less capable devices, such as those without JavaScript support, and still provide the best possible experience.

**Accessibility**  
jQuery Mobile is designed with accessibility in mind. It has support for Accessible Rich Internet Applications (WAI-ARIA) to help make web pages accessible for visitors with disabilities using assistive technologies.

**Small size**  
The overall size of the jQuery Mobile framework is relatively small at 12KB for the JavaScript library, 6KB for the CSS, plus some icons.

**Theming**  
The framework also provides a theme system that allows you to provide your own application styling.

When used with toolkits such as PhoneGap (see Related topics), which uses web technologies to build stand-alone applications, the jQuery Mobile framework can help simplify your application development.

**Browser support**

We've come a long way with mobile device browser support, but not all mobile devices provide support for HTML5, CSS 3, and JavaScript. This arena is where jQuery Mobile's progressive enhancement and graceful degradation support come into play. As stated, jQuery Mobile supports both high-end and less capable devices, such as those without JavaScript support. Progressive Enhancement consists of the following core principles (source: Wikipedia):

- All basic content should be accessible to all browsers.
- All basic functionality should be accessible to all browsers.
- Enhanced layout is provided by externally linked CSS.
- Enhanced behavior is provided by externally linked JavaScript.
- End user browser preferences are respected.

All basic content should render (as designed) on basic devices, while more advanced platforms and browsers will be progressively enhanced with additional, externally linked JavaScript and CSS.
jQuery Mobile currently provides support for the following mobile platforms:

- Apple® iOS: iPhone, iPod Touch, iPad (all versions)
- Android™: all devices (all versions)
- Blackberry® Torch (version 6)
- Palm™ WebOS Pre, Pixi
- Nokia® N900 (in progress)

See the supported browser matrix on the jQuery Mobile site for more information (see Related topics).

**Summary of UI components**

jQuery Mobile provides robust support for different kinds of UI elements. Figure 1 shows a summary of the currently supported UI components.
Toolbars, buttons, list views, tabs, pop-up menus, dialogs, transitions, edit panels, and form elements are supported. Most, if not all, of the UI elements that you need for your mobile web applications are provided.

$.mobile and supported methods and events

The JavaScript object jQuery is also referred to as $. The jQuery Mobile framework extends jQuery core with mobile plug-ins, including the mobile, or $.mobile, which defines several events and methods. Some of the methods exposed by $.mobile are described below.

Table 1. Methods exposed by $.mobile

<table>
<thead>
<tr>
<th>Method</th>
<th>Usage</th>
</tr>
</thead>
</table>

Introduction to jQuery Mobile
$.mobile.changePage

To programmatically change from one page to another. For example, to go to page weblog.php using a slide transition, use

$.mobile.changePage("weblog.php", "slide").

$.mobile.pageLoading

To show or hide the page loading message. For example, to hide the message, use

$.mobile.pageLoading(true).

$.mobile.silentScroll

To scroll to a particular Y position without generating scroll events. For example, to scroll to Y position 50, use

$.mobile.silentScroll(100).

$.mobile.addResolutionBreakpoints

jQuery Mobile already defines some breakpoints for min/max classes. Call this method to add breakpoints. For example, to add min/max class for 800 pixel widths, use

$.mobile.addResolutionBreakpoints(800).

$.mobile.activePage

Refers to the currently active page.

There are several events that you can bind using the bind() or live() method, such as jQuery Mobile initialization, touch events, orientation change, scroll events, page show/hide events, page-initialization events, and animation events.

For example, touch events include tap, taphold, and various swipe events. Scroll events include scrollstart and scrollstop. Page events allow you to receive notifications: prior to a page creation, when a page is created, just before the page is shown or hidden, and when the page is shown and hidden.

Listing 1 shows an example of binding a specific event when jQuery Mobile starts to execute.

**Listing 1. Bind to the mobileinit event**

```javascript
$(document).bind("mobileinit", function(){
    //apply overrides here
});
```

The event above lets you override default values when jQuery Mobile starts. Several settings can be overridden, such as:

- **LoadingMessage** - Sets the default text that appears when a page is loading.
- **defaultTransition** - Sets the default transition for page changes that use Ajax.

There are more configuration parameters that you can override as needed. See the jQuery Mobile documentation (see Related topics), or the source code (see Downloadable resources), for more information.

You can also bind to other events that allow you to create dynamic applications based on touch and page events.

**HTML5 data-* attributes**

jQuery Mobile relies on HTML5 data-* attributes to support the various UI elements, transitions, and page structure. They are silently discarded by browsers that don't support them. Table 2 shows how to use data-* attributes to create UI components.
Table 2. `data-*` attributes for UI components

<table>
<thead>
<tr>
<th>Component</th>
<th>HTML5 data-* attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header, Footer toolbars</td>
<td><code>&lt;div data-role=&quot;header&quot;&gt;</code> &lt;div data-role=&quot;footer&quot;&gt;</td>
</tr>
<tr>
<td>Content body</td>
<td><code>&lt;div data-role=&quot;content&quot;&gt;</code></td>
</tr>
<tr>
<td>Buttons</td>
<td><code>&lt;a href=&quot;index.html&quot; data-role=&quot;button&quot; data-icon=&quot;info&quot;&gt;Button&lt;/a&gt;</code></td>
</tr>
<tr>
<td>Grouped buttons</td>
<td>`&lt;div data-role=&quot;controlgroup&quot;&gt; &lt;a href=&quot;index.html&quot; data-role=&quot;button&quot;&gt;Yes&lt;/a&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;a href=&quot;index.html&quot; data-role=&quot;button&quot;&gt;No&lt;/a&gt;  &lt;a href=&quot;index.html&quot; data-role=&quot;button&quot;&gt;Hell Yeah&lt;/a&gt; &lt;/div&gt;`</td>
</tr>
<tr>
<td>Inline buttons</td>
<td><code>&lt;div data-inline=&quot;true&quot;&gt; &lt;a href=&quot;index.html&quot; data-role=&quot;button&quot;&gt;Foo&lt;/a&gt;  &lt;a href=&quot;index.html&quot; data-role=&quot;button&quot; data-theme=&quot;b&quot;&gt;Bar&lt;/a&gt; &lt;/div&gt;</code></td>
</tr>
<tr>
<td>Form element (Select menu)</td>
<td><code>&lt;div data-role=&quot;fieldcontain&quot;&gt; &lt;label for=&quot;select-options&quot; class=&quot;select&quot;&gt;Choose an option:&lt;/label&gt;  &lt;select name=&quot;select-options&quot; id=&quot;select-options&quot;&gt;  &lt;option value=&quot;option1&quot;&gt;Option 1&lt;/option&gt;  &lt;option value=&quot;option2&quot;&gt;Option 2&lt;/option&gt;  &lt;option value=&quot;option3&quot;&gt;Option 3&lt;/option&gt;  &lt;/select&gt;  &lt;/div&gt;</code></td>
</tr>
<tr>
<td>Basic List views</td>
<td><code>&lt;ul data-role=&quot;listview&quot;&gt;  &lt;li&gt;&lt;a href=&quot;index.html&quot;&gt;One&lt;/a&gt;&lt;/li&gt;  &lt;li&gt;&lt;a href=&quot;index.html&quot;&gt;Two&lt;/a&gt;&lt;/li&gt;  &lt;li&gt;&lt;a href=&quot;index.html&quot;&gt;Three&lt;/a&gt;&lt;/li&gt;  &lt;/ul&gt;</code></td>
</tr>
<tr>
<td>Dialogs</td>
<td><code>&lt;a href=&quot;foo.html&quot; data-rel=&quot;dialog&quot;&gt;Open dialog&lt;/a&gt;</code></td>
</tr>
<tr>
<td>Transitions</td>
<td><code>&lt;a href=&quot;index.html&quot; data-transition=&quot;pop&quot; data-back=&quot;true&quot;&gt;</code></td>
</tr>
</tbody>
</table>

The jQuery Mobile documentation has a complete list of supported `data-*` syntax.

**Structure of a jQuery Mobile page**

This section discusses the general structure of a jQuery Mobile page. jQuery Mobile has guidelines on the structure of pages themselves. In general, a page structure should have the following sections:

**Header bar**
- Typically contains the page title and Back button

**Content**
- The content of your application

**Footer bar**
- Typically contains navigational elements, copyright information, or whatever you need to add to the footer

Figure 2 shows examples of the different sections.
The header and footer toolbars support fixed and full-screen positioning options. The fixed position makes the toolbars static when scrolling. The full-screen positioning works the same as fixed except the bars are displayed only when clicking on the page (to provide a non-obtrusive, full-content experience). The rest of this section explores the HTML code for a generic page structure.

The definition of an HTML type of document itself is `!DOCTYPE html>`, which defines an HTML5 document type.

The HTML head section loads three important jQuery Mobile components:

- jQuery Core library — The core jQuery library
- jQuery Mobile library — The mobile-specific part of the jQuery framework
- jQuery Mobile CSS — The CSS that defines the core jQuery Mobile UI elements. It defines the transitions and different UI widgets, such as sliders and buttons, and makes heavy use of Webkit transforms and animations.

Listing 2 shows the HTML head section.
Listing 2. HTML head section

```html
<html>
  <head>
    <meta charset="utf-8" />
    <title>Intro to jQuery Mobile</title>
    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.0a2/jquery.mobile-1.0a2.min.css" />
    <script src="http://code.jquery.com/jquery-1.4.4.min.js"></script>
    <script src="http://code.jquery.com/mobile/1.0a2/jquery.mobile-1.0a2.min.js"></script>
  </head>
</html>
```

The next section of the HTML code defines the page itself; see the use of the `data-role="page"` attribute. Listing 3 shows an example.

Listing 3. Defining a page block

```html
<body>
  <div data-role="page" id="page1">
```

Listing 3 defines a page. The `id` attribute is necessary only if local multiple page blocks reside on the same HTML file document, but it's good practice to define a unique ID.

The next couple of code listings show how to define the different header, content, and footer sections of the page. The header bar typically consists of the page title and Back button, as shown in Listing 4.

Listing 4. Page header bar section

```html
  <div data-role="header">
    <h1>Header Bar</h1>
  </div><!-- /header -->
```

In this case, the header bar consists of only an H1 header title text. The bulk of your content goes after the header, as shown below. The example in Listing 5 shows only a simple paragraph, but this is where you would add lists, buttons, forms, and so on.

Listing 5. Page content section

```html
  <div data-role="content">
    <p>Content Section</p>
  </div><!-- /content -->
```

The footer bar is where you typically place navigational elements and copyright information, as shown in Listing 6.

Listing 6. Footer bar section

```html
  <div data-role="footer">
    <h4>Footer Bar</h4>
  </div><!-- /footer -->
```

```html
</body>
</html>
```
The example footer section in Listing 6 is very simple. Adding a navigational bar to the footer bar is not very complicated, as shown in Listing 7.

**Listing 7. Adding a nav bar to the footer section**

```html
<div data-role="footer" class="ui-bar">
  <div data-role="controlgroup" data-type="horizontal">
    <a href="index.html" data-role="button">Today</a>
    <a href="index.html" data-role="button">Tomorrow</a>
    <a href="index.html" data-role="button">Week</a>
    <a href="index.html" data-role="button">No date</a>
  </div>
</div><!-- /footer -->
```

Figure 3 shows the resulting footer bar section with the newly added navigational bar.

**Figure 3. Footer with navigational bar**

![Footer Bar Image](image)

---

**Defining multiple local pages**

The previous example covered a single page. jQuery Mobile also provides support for multiple pages within a single HTML document. The multiple pages are local, internal linked "pages" that you can group together for preloading purposes. The structure of the multipage page is similar to the previous example of a single page, except that it will contain multiple page data-roles. Listing 8 shows an example.

**Listing 8. Defining multiple pages within a single HTML file**

```html
<div data-role="page" id=page1">
  
</div>
```
When referencing a page that is local to the same HTML document, jQuery Mobile automatically deals with the references. When referencing an external page, jQuery Mobile will display a loading spinner. If an error is encountered, the framework will automatically display and handle an error message pop-up.

### Page transitions

jQuery Mobile provides support for CSS-based page transitions (inspired by jQtouch), which are applied when navigating to a new page and back. The transitions include:

- **Slide**
  - Provides a horizontal transition
- **Slideup and Slidedown**
  - Provide transitions up and down the screen
- **Pop**
  - Provides an explode type of transition
- **Fade**
  - Provides a fading transition
- **Flip**
  - Provides a flip transition

You can add page transitions in two different ways:

- Add a data-transition attribute to the link, using `<a href="index.html" data-transition="pop" data-back="true">`.
  - Use the data-transition attribute on static pages.
- Programatically, using `.mobile.changePage("pendingtasks.html", "slideup");`.
  - Use the programmatic approach when working with dynamic pages.

### List views

List views, a fundamental type of UI element, are heavily used in mobile applications. jQuery Mobile supports numerous list views: basic, nested, numbered, and read-only lists; split buttons;
list dividers; count bubbles; thumbnails; icons; search filter bars; inset styled lists; and theming lists.

Listing 9 shows a basic list view. A list view is created by using the `<ul data-role="listview">` data attribute.

**Listing 9. Creating a simple list view**

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Intro to jQuery Mobile</title>
  <link rel="stylesheet" href="http://code.jquery.com/mobile/1.0a2/jquery.mobile-1.0a2.min.css" />
  <script src="http://code.jquery.com/jquery-1.4.4.min.js"></script>
  <script src="http://code.jquery.com/mobile/1.0a2/jquery.mobile-1.0a2.min.js"></script>
</head>
<body>
  <div data-role="page">
    <div data-role="header">
      <h1>Facebook Friends</h1>
    </div><!-- /header -->
    <div data-role="content">
      <p>
        <ul data-role="listview" data-inset="true">
          <li><a href="index.html">Get Friends</a></li>
          <li><a href="index.html">Post to Wall</a></li>
          <li><a href="index.html">Send Message</a></li>
        </ul>
      </p>
    </div><!-- /content -->
    <div data-role="footer">
    </div><!-- /footer -->
  </div><!-- /page -->
</body>
</html>
```

Inside the `<ul data-role="listview"/>` you define common `<li>` list items. This is a perfect example of how jQuery Mobile extends the basic HTML syntax. The result of the code example in Listing 10 is shown in Figure 4.
Forms

Forms are another common web artifact used to post information to a server. jQuery Mobile supports many form UI components: text inputs, search inputs, slider, flip toggle switch, radio buttons, checkboxes, select menus, and theming forms. They can all be created very easily. Listing 10 shows a form with a select menu and a submit button.

Select menus are driven by native `<select name="select-options" id="select-options">`, but jQuery Mobile improves on its "look and feel." The `<div data-role="fieldcontain">` statement groups the different values for visualization purposes. The form itself is defined by native `<form action="..." method="get">`.

Listing 10. A form, select menu, and submit button

```html
<form action="forms-results.php" method="get">
  <fieldset>
    <div data-role="fieldcontain">
      <label for="select-options" class="select">Choose an option:</label>
      <select name="select-options" id="select-options">
        <option value="option1">Option 1</option>
        <option value="option2">Option 2</option>
        <option value="option2">Option 3</option>
      </select>
    </div>
    <button type="submit">Submit</button>
  </fieldset>
</form>
```
Figure 5 shows the form and its content.

**Figure 5. Simple form with selection menu and submit button**

![Simple form with selection menu and submit button](image)

When choosing the selection menu, jQuery Mobile displays the pop-up displayed on the image (on the right in Figure 5) with all of the selection values, which will automatically close after the selection. As long as the `action` and `method` attributes of the form are properly defined, jQuery Mobile takes care of the transitions between the form, the Ajax invocation, and the results page, and displays a spinner as necessary.

**Other UI components**

There are many more UI elements, and variations of elements, to be explored on the jQuery Mobile website and in the documentation (see Related topics). To complement what you learned in this article, it is recommended that you look into: collapsible content, layout grids, theming, and the rest of the list view and forms.

**Conclusion**

This article introduced the extensive jQuery Mobile JavaScript framework. You learned the basics of the framework and how to write functional web pages without having to write a single line of JavaScript code. If you need to manipulate the HTML documents, you can do so with the jQuery core. You explored basic pages and navigation, toolbars, form controls, and transition effects. jQuery Mobile provides numerous methods, events, and properties that you can work with programmatically. Let this inspire you to pursue more information about UI components not covered in this article.
## Downloadable resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source code for this article</td>
<td>jquerymobileexamplecode.zip</td>
<td>3KB</td>
</tr>
</tbody>
</table>
Related topics

- Learn all about the jQuery Mobile framework.
- Explore jQuery Mobile documentation and demos: articles, APIs, and demo code.
- Access all of the jQuery documentation.
- Mobile Graded Browser Support provides a supported browser matrix for jQuery Mobile.
- jQuery: Plugins/Authoring shows how to write your own jQuery plugins.
- The JavaScript Guide explains everything you need to know about using JavaScript.
- The Mozilla Developer Network is a great resource for web developers.
- Read What is the Document Object Model? from the W3C specification document.
- The JavaScript tutorial DOM objects and methods covers all properties, collections, and methods of the W3C DOM.
- Get an overview of WAI-ARIA, the Accessible Rich Internet Applications Suite, which defines a way to make web content and web applications more accessible to people with disabilities.
- Progressive enhancement is a strategy for web design that emphasizes accessibility, semantic HTML markup, and external stylesheet and scripting technologies.
- "JavaScript and the Document Object Model" (developerWorks, Jul 2002) looks at the JavaScript approach to DOM and chronicles the building of a web page to which the user can add notes and edit note content.
- Read the HTML5 Specification from the W3C.
- To listen to interesting interviews and discussions for software developers, check out developerWorks podcasts.
- Get PhoneGap, an open source development framework for building cross-platform mobile apps.

© Copyright IBM Corporation 2011
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)