jQuery Mobile 开发入门手册——入门篇

作者：张勇辉 更新日期 2010-11-03

Blog： http://www.uedcool.com

目录

概述

此文档是基于 jQuery Mobile 框架的移动设备 Web 应用开发知识而编制，目的是为了方便开发人员快速的掌握此框架的开发应用，其中包含了框架的基础应用知识和在团队协作开发中的常规约定。

框架特性

JQuery Mobile 以“**Write Less, Do More**”作为目标，为所有的主流移动操作系统平台提供了高度统一的 UI 框架：jQuery 的移动框架可以让你为所有流行的移动平台设计一个高度定制和品牌化的 Web 应用程序，而不必为每个移动设备编写独特的应用程序或操作系统。

jQuery Mobile 目前支持的移动平台有苹果公司的 iOS（iPhone，ipad，iPod Touch），Android，Black Berry OS6.0，惠普 WebOS，Mozilla 的 Fennec 和 Opera Mobile。今后，将增加包括 Windows Mobile，Symbian 和 MeeGo 在内的更多移动平台。

根据 jQuery Mobile 项目网站，目前 jQuery Mobile 的特性包括：

- jQuery 核心——与 jQuery 桌面版一致的 jQuery 核心和语法，以及最小的学习曲线。
- 兼容所有主流的移动平台——iOS、Android、BlackBerry，Palm WebOS、Symbian、Windows Mobile、BaDa、MeeGo 以及所有支持 HTML 的移动平台。
- 轻量级 alpha 版本的 jQuery Mobile 其 JavaScript 大小仅为 12KB，CSS 文件也只有 6KB 大小。
- 标记驱动的配置 jQuery Mobile 采用完全的标记驱动而不需要 JavaScript 的配置。
- 渐进增强 jQuery Mobile 采用完全的渐进增强原则：通过一个全功能的 HTML 网页，和额外的 JavaScript 功能层，提供顶级的在线体验。这意味着即使移动浏览器不支持 JavaScript，基于 jQuery Mobile 的移动应用程序仍能正常的使用。
- 自动初始化通过使用 mobilize()函数自动初始化页面上的所有 jQuery 部件。
- 无障碍 包括 WAI-ARIA 在内的无障碍功能以确保页面能在类似于 VoiceOver 等语音辅助程序和其他辅助技术下正常使用。
- 简单的 API 为用户提供鼠标、触摸和光标焦点简单的输入法支持。
- 强大的主题化框架 jQuery Mobile 提供强大的主题化框架和 UI 接口。

版本约定

为了避免由于版本不统一等引发的问题，在此次撰写中对框架的版本进行了如下约定 ：

jQuery 核心：V 1.50

Mobile 核心：V 1.0 ALPHA 3

初始配置

在<head>中按顺序加入框架的引用，注意加载的顺序:

```
<link rel="stylesheet" type="text/css" href="jquery.mobile-1.0a2.min.css">
<script src="jquery-1.4.4.min.js"></script>
<!--这里加入项目中其他的引用-->
<script src="jquery.mobile-1.0a2.min.js"></script>
```

ps:建议在 meta 中加入'”charset=utf-8”的声明，避免出现乱码和响应方面的问题
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" /> 或者 <meta charset="utf-8" />

页面声明

建议在页面中使用 HTML5 标准的页面声明和标签，因为移动设备浏览器对 HTML5 标准的支持程度要远远优于 PC 设备，因此使用简洁的 HTML5 标准可以更加高效的进行开发，免去了因为声明错误出现的兼容性问题。
HTML5 页面基础元素：

```
<!DOCTYPE HTML>
<html lang="en-US">
<head>
<title>标题 </title>
<meta charset="UTF-8">

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

技术理论

WebKit 和 HTML5

WebKit 是一种浏览器引擎，支撑着 iPhone 内的 Mobile Safari 浏览器以及 Android 内的浏览器背后的技术。WebKit 也在其他的移动环境内有自己的用武之地，但是我们还是将我们

的讨论集中于 iPhone 和 Android 平台。

WebKit 是一个开源项目，其起源可追溯到 K Desktop Environment (KDE)。WebKit 项目催生了面向移动设备的现代 Web 应用程序。虽然设备本身的能力和形态因素都相当重要，但移动用户最热衷的仍然是内容。如果移动用户可用的内容只是 Internet 用户可用内容的一个很小的子集，那么用户体验充其量也只能划分为二等。

我们当中的大多数人都更希望生活是连贯的 — 如果我们在家中的笔记本上访问了一个网站，我们同样希望在火车上旅行时仍然访问到同样的内容。内容是最好的应用程序。不管我们身在何处、在做什么，我们都想要访问到我们的数据。WebKit 让 iPhone 和 Android 平台上可以有丰富的内容。

有一点很值得注意，即 WebKit 还应用在了桌面的 Safari 浏览器内，该浏览器是 Mac OS X 平台默认的浏览器。不管我们讨论的是桌面版本还是 iPhone 或 Android 上的浏览器引擎，WebKit 均优先支持 HTML 和 CSS 特性。实际上，WebKit 还支持尚未被其他浏览器采纳的一些 CSS 样式 — 这些特性正在得到 HTML5 规范的考虑。

HTML5 规范是一个技术草案集，涵盖了各种基于浏览器的技术，包括客户端 SQL 存储、转变、转型、转换等。HTML5 的出现已经有些时间了，虽然尚未完成，但是一旦其特性集因主要浏览器平台支持的加入而逐渐稳定后，Web 应用程序的简陋开端将成为永久的记忆。Web 应用程序开发将成为主导 — 并且不只是在传统的桌面浏览器空间，还将在移动领域。移动将一跃成为首要考虑，而不再是后备之选。

移动 Web 应用程序的考虑

为了访问 Web 开发技术，如今，应用程序开发人员有几个选择。第一，应用程序可严格编写为服务器上的 HTML、CSS 和 JavaScript 文件。当然，HTML 内容可以产生自静态 HTML 文件，也可以从任何的服务器端技术（比如 PHP、 ASP.NET、Java Servlets 等）动态生成。所有这些技术追根到底都可简单地用术语 HTML 指代 — 这不是本文讨论的重点所在 — 并且最为重要的是，受 WebKit-支撑的浏览器能够在移动设备上解析和呈现 HTML。

用户通过在移动设备上（即 iPhone 或 Android）打开浏览器应用程序并输入目标服务器对应的 URL：http://yourcompanyname.com/applicationurl 来访问 Web 应用程序。

特定的某个移动 Web 应用程序总是能找到自己的位置：从一般的 Web 站点到高度特定于平台的移动 Web 应用程序。

一般站点的呈现

WebKit 内的呈现引擎，再配以 iPhone 和 Android 平台上的高度直观的 UI，实际上就使得几乎任何一个基于 HTML 的 Web 站点都能呈现在此设备上。Web 页能被正确呈现，不再

像原来的移动浏览器体验：内容被包裹起来或是根本不显示。当页面加载后，内容通常被完全缩放以便整个页面都可见，尽管内容会被缩放得非常小，甚至不可读，如图 1 所示。不过，页面是可滚动、放大、缩小的，这就提供了对全部内容的访问。默认地，浏览器使用 980 像素宽的视见区或逻辑尺寸。
要想使 Web 页面从一般的页面变成支持移动设备的页面，Web 应用程序可以在几个方面进行修改。

虽然页面可以在 WebKit 中正确呈现，但是，一个以鼠标为中心的设备（比如笔记本或台式机）与一个以触摸为中心的设备（比如一个 iPhone 或 Android 智能手机）还是有区别的。其中主要的一些差异包括 “可单击” 区域的物理大小、“悬浮样式” 的缺少以及完全不同的事件顺序。如下所列的是在设计一个能被移动用户正常查看的 Web 站点时需要注意的一些事情：

•iPhone/Android 浏览器呈现的屏幕是可读的 — 大大好于传统的移动浏览器 — 所以不要急于草草制作您网站的移动版本。
•手指要大过鼠标指针。在设计可单击的导航时要特别注意这一点 — 不要把链接放得相互太靠近，因为用户不太可能单击了一个链接而不触及相邻的链接。
•悬浮样式将不再奏效，因为用手指不能进行用鼠标指针进行的 “悬浮”。
•诸如 mouse-down、mouse-move 等事件在基于触摸的设备上自然大相径庭。这类事件中有一些将被取消，不要指望移动设备上的事件顺序与桌面浏览器上的一样。
让我们来看看要使一个 Web 站点对 iPhone 或 Android 访客具有友好性所面临的最为明显的一个挑战：屏幕大小。我们今天使用的实际移动屏幕尺寸是 320x480。请注意由于用户可能会选择横向查看 Web 内容，所以屏幕大小也可以是 480x320。正如我们在图 1 中看到的，WebKit 将能很好地呈现面向桌面的 Web 页面，但是文本可能会太小以至于若不进行缩放或其他操作就无法有效阅读内容。那么，我们该如何应对这个问题呢？

最为直观也是最不唐突的适合移动用户的方式是通过使用一个特殊的 metatag：viewport。

metatag 是一个放入 HTML 文档的 head 元素内的 HTML 标记。如下是一个使用 viewport 标记的简单例子：<meta name="viewport" content="width=device-width" />。当这个 metatag 被添加到一个 HTML 页面后，我们看到此页面被缩放到更为适合这个移动设备的大小，如图 2 所示。如果浏览器不支持此标记，它会简单地忽略此标记。
为了设置特定的值，将 viewport metatag 的 content 属性设为一个显式的值：*<meta name="viewport" content="width=device-width, initial-scale=1.0 user-scalable=yes" />*。通过改变初始值，屏幕就可以按要求被放大或缩小。将值分别设置在 1.0 和 1.3 之间对于 iPhone 和 Android 平台是比较合适的。viewport metatag 还支持最小和最大伸缩，可用来限制用户对呈现页面的控制力。

自具有 320x480 布局的 iPhone 面世以来，其形态系数就一直没有改变过，而随着来自不同制造商、针对不同用户群的更多设备的出现，Android 则有望具备更多样的物理特点。在开发应用程序并以诸如 Android 这类移动设备为目标时，一定要考虑屏幕尺寸、形态系数以及分辨率方面的潜在多样性。
除了 Android 设备与其他设备之间的这些物理差异之外，经验还表明 Android 的软件还通过设备内置的（on-device）浏览器设置对页面的呈现实施了更多控制。不仅稳定，Android 平

台还很灵活。取决于 SDK 等级和制造商，某个设备上的设置很可能不同于您的开发环境。图 4 显示了取自 Android Emulator V1.6 的浏览器应用程序的设置页面。这个设置屏幕允许用户将一个设备设置为一个预先定义的缩放等级（far、near、medium）或请求此设备自动适应页面。

组件

页面

jQuery Mobile 应用了 HTML5 标准的特性，在结构化的页面中完整的页面结构分为 header、content、footer 这三个主要区域。

在 body 中插入内容块：

```
<div data-role="page">
    <div data-role="header">...</div>
    <div data-role="content">...</div>
    <div data-role="footer">...</div>
</div>
```

data-role="page"代表这个 div 是一个 Page，在一个屏幕中只会显示一个 page；
header 是标题，content 是内容块，footer 是页脚

data-role 参数表：

参数	说明
page	页面容器，其内部的 mobile 元素将会继承这个容器上所设置的属性
header	页面标题容器，这个容器内部可以包含文字、返回按钮、功能按钮等元素
footer	页面页脚容器，这个容器内部也可以包含文字、返回按钮、功能按钮等元素
content	页面内容容器，这是一个很宽容的容器，内部可以包含标准的 html 元素和 jQueryMobile 元素
controlgroup	将几个元素设置成一组，一般是几个相同的元素类型
fieldcontain	区域包裹容器，用增加边距和分割线的方式将容器内的元素和容器外的元素明显分隔
navbar	功能导航容器，通俗的讲就是工具条
listview	列表展示容器，类似手机中联系人列表的展示方式
list-divider	列表展示容器的表头，用来展示一组列表的标题，内部不可包含链接
button	按钮，将链接和普通按钮的样式设置成为 jQueryMobile 的风格
none	阻止框架对元素进行渲染，使元素以 html 原生的状态显示，主要用于 form 元素。

完整的页面模版：

```
<!DOCTYPE html>
<html>
<head>
    <title>Page Title</title>
    <link rel="stylesheet" href="jquery.mobile-1.0a3.min.css" />
    <script type="text/javascript" src="jquery-1.4.3.min.js"></script>
    <script type="text/javascript" src="jquery.mobile-1.0a3.min.js"></script>
</head>
<body>
<div data-role="page">
    <div data-role="header">
        <h1>Page Title</h1>
    </div><!-- /header -->
    <div data-role="content">
        <p>Page content goes here.</p>
    </div><!-- /content -->
    <div data-role="footer">
        <h4>Page Footer</h4>
    </div><!-- /footer -->
</div><!-- /page -->
</body>
</html>
```

以上是一个完整的页面结构模版代码，在使用过程中可以根据需要来组合。

页面动画：

data-transition 属性可以定义页面切换是的动画效果。

例如：`<a href="index.html" data-transition="pop">I'll pop</a>`

data-transition 参数表：

参数	说明
slide	**从右侧向左滑入页面**
slideup	**从底部向上滑入**
slidedown	**从上向下滑入**
pop	**从中心渐显展开**
fade	**渐显**
flip	**翻转**

备注：如果想要在目标页面中显示后退按钮，可以在链接中加入 data-direction="reverse"属性，这个属性和原来的 data-back="true" 相同，不知道在正式版本中将会保留哪个属性。

模态对话框

模态对话框是一种带有圆角标题栏和关闭按钮的伪浮动层，用于独占事件的应用。任何结构

化的页面都可以用 *data-rel="dialog"*链接的方式实现模态对话框应用。
例如：***Open dialog***
这个页面切换效果同样可以使用标准页面的 **data-transition** 参数效果。建议使用"pop"、"slideup" 和"flip"参数以达到更好的效果。
这个模态对话框会默认生成关闭按钮，用于回到父级页面。在脚本能力较弱的设备上也可以添加一个带有 data-rel="back"的链接来实现关闭按钮。
针对支持脚本的设备可以直接使用 href="#"或者 data-rel="back"来实现关闭。还可以使用内置的"close"方法来关闭模态对话框，例如：*$('.ui-dialog').dialog('close')*。
由于模态对话框是动态显示的临时页面，所以这个页面不会被保存在哈希表内，这就意味着我们讲无法后退到这个页面，例如你在 A 页面中点击一个链接打开 B 对话框,操作完成并关闭对话框,然后跳转到 C 页面,这时候你点击浏览器的后退按钮，这时候将回到 A 页面，而不是 B 页面。

工具条

工具条主要用于"header","footer"等区域，用来支撑和实现页面中业务功能的应用。jQuery Mobile 提供了一个相对完整的解决方案。
工具条分为:标题（header bar）,页脚（footer bar）和导航（nav bar）这三中应用。
其中标题和页脚在页面中有一些不同的应用方式，默认工具条是以嵌入（*inline*）的方式定位的，这种定位方式可以实现最大限度的兼容性，包括在对脚本和 css 兼容性不佳的设备都有很好的优化。
另一种是浮动（fixed）定位的方式，也可以成为“静态“定位，这种定位方式可以让工具条始终保持在屏幕的顶部或者底部。并可以接受点击事件来显示/隐藏工具条，已达到最大化利用屏幕空间的目的。
实现方式：在标题和页脚区域加入 data-position="fixed"属性。

标题容器

标题容器是页面页眉区域的显示控件，主要用来显示标题和主要操作的区域。
结构代码：

```
<div data-role="header">
    <h1>Page Title</h1>
</div>
```

为了方便页面的交互在页面切换后会在标题容器的左侧自动生成一个后退按钮，这样可以简化我们的开发复杂程度，但是有些时候我们会因为应用的需求而不需要这个后退按钮，可以在标题容器上添加 data-backbtn="false"属性用来阻止后退按钮的自动创建。
标题容器的左侧和右侧分别可以放置一个按钮，在阻止自动生成的后退按钮后，我们就可以在后退按钮的位置来自定义按钮了。
例如：

```
<div data-role="header" data-position="inline" data-backbtn="false" >
        <a href="index.html" data-icon="delete">Cancel</a>
        <h1>Edit Contact</h1>
        <a href="index.html" data-icon="check">Save</a>
</div>
```

如果需要自定义默认的后退按钮中的文本，可以用 *data-back-btn-text="previous"*属性来实现，或者通过扩展的方式实现：*$.mobile.page.prototype.options.backBtnText = "previous"*。
如果你没有使用标准的结构来创建标题区域，那么框架将不会自动生成默认的按钮。

页脚容器

页脚容器的结构和标题容器的结构基本相同，只要把 data-role 属性的参数设置为”footer”。
例如：

```
<div data-role="footer">
    <h4>Footer content</h4>
</div>
```

与标题容器相比页脚容器有更多的灵活度，它不会想标题容器一样只允许放置两个按钮，并且也不会默认的把按钮放置在左右的顶端，页脚的按钮默认是从左到右依次排列的，并且何以放置更多的按钮。
在页脚容器上只要添加一个 class="ui-bar"就可以将页脚变成一个工具条，你可以不用设置任何的布局样式就可以在其中添加整齐的按钮。
例如：

```
<div data-role="footer" class="ui-bar">
    <a href="index.html" data-role="button" data-icon="delete">Remove</a>
    <a href="index.html" data-role="button" data-icon="plus">Add</a>
    <a href="index.html" data-role="button" data-icon="arrow-u">Up</a>
    <a href="index.html" data-role="button" data-icon="arrow-d">Down</a>
</div>
```

如果我们需要一组链接效果，我们可以这样写：

```
<div data-role="footer" class="ui-bar" data-position="inline">
                <div data-role="controlgroup" data-type="horizontal">
                    <a href="index.html" data-icon="delete">Remove</a>
                    <a href="index.html" data-icon="plus">Add</a>
                    <a href="index.html" data-icon="arrow-u">Up</a>
                    <a href="index.html" data-icon="arrow-d">Down</a>
                </div>
            </div>
```

技巧：通过使用 data-id 属性可以让多个页面使用相同的页脚。

导航

导航容器是一个可以每行容纳最多 5 个按钮的按钮组控件，用一个拥有 data-role="navbar" 属性的 div 来容纳这些按钮。
例子：

```
<div data-role="footer">
    <div data-role="navbar">
        <ul>
            <li><a href="a.html" class="ui-btn-active">One</a></li>
            <li><a href="b.html">Two</a></li>
        </ul>
    </div><!-- /navbar -->
</div><!-- /footer -->
```

在默认的按钮上添加 class="ui-btn-active"
如果按钮的数量超过 5 个，导航容器将会自动以合适的数量分配成多行显示。

按钮

你可以将页面中的任何一个链接通过 data-role="button"来声明成为按钮的显示风格。为了风格统一，框架会在页面加载时自动将 form 类的按钮格式化为 jQuery Mobile 风格的按钮，不需要添加 data-role 属性。
框架中包含了一组常用的图标可以用于按钮，用 data-icon 属性中的参数来定义显示不同的图标效果。
例如：`<a href="index.html" data-role="button" data-icon="delete">Delete</a>`

data-icon 原生参数列表

参数	图标

arrow-l	
arrow-r	
arrow-u	
arrow-d	
delete	
plus	
minus	
check	
gear	
refresh	
forward	
back	
grid	
star	
alert	
info	
home	
search	

除了可以默认显示左侧的图标之外，还可以用 data-iconpos 属性来定义图标与文字的位置关系。

data-iconpos 参数列表：

参数	效果

right	**图标在文字的右侧**
top	**图标在文字上面**
bottom	**图标在文字下面**

data-iconpos="notext"属性可以让按钮隐藏文字。

内联样式

在框架中默认情况下按钮是横向独占根据屏幕宽度横向自适应的，但是我们在应用的应用中经常需要在一行中显示多个按钮，这时候我们就需要知道一个新的叫做内联模式的属性了data-inline="true"。

例如：

```
<div data-inline="true">
    <a href="index.html" data-role="button">Cancel</a>
    <a href="index.html" data-role="button" data-theme="b">Save</a>
</div>
```

按钮组

jQuery Mobile 框架可以将几个按钮以组的方式显示，data-role="controlgroup"用以展示按钮间的紧凑关系。例如：

```
<div data-role="controlgroup">
<a href="index.html" data-role="button">Yes</a>
<a href="index.html" data-role="button">No</a>
<a href="index.html" data-role="button">Maybe</a>
</div>
```

如果需要按钮横向排列可以增加 data-type="horizontal"属性。

表单应用

jQuery Mobile 框架为原生的 html 表单元素封装了新的表现形式，对触屏设备的操作进行了优化。在框架的页面中会自动将 form 元素渲染成 jQuery Mobile 风格的元素。

form 元素的使用和默认的 html 方式使用相同，可以同样使用 Post 和 get 方式提交数据，但是需要注意的是元素的 ID 命名问题，在常规的规范中同一个页面中是不允许出现相同的 ID 命名的，在 jQuery Mobile 中由于其允许在同一个 DOM 中存在多个页面，所以建议 form 元素的 ID 命名在整个项目中是唯一的，防止由于 ID 问题引发的错误。

默认情况下框架会自动渲染在标准页面中的 form 元素的风格，一旦成功渲染后，这个控件元素将可以使用 jQuery 中的函数进行操作。

在某些情况下，我们需要使用 html 原生的 form 元素，为了阻止 mobile 框架对该元素的自动渲染，在框架中我们在 data-role 属性中引入了一个控制参数”none”。使用这个属性参数就会让该元素以 html 原生的状态显示。
例如：

```
<select name="foo" id="foo" data-role="none">
    <option value="a" >A</option>
    <option value="b" >B</option>
    <option value="c" >C</option>
</select>
```

列表应用

信息列表是开发应用中使用频率相对比较高的控件，用于数据显示、导航，数据列表等。为了适应不同的信息内容，列表的表现形式也多种多样。
列表的代码结构是以有序和无序列表来实现的，只要在 ul 或 ol 上声明 data-role="listview" 就可以让框架以列表的方式渲染了，例如：

```
<ul data-role="listview" data-theme="g">
    <li><a href="acura.html">Acura</a></li>
    <li><a href="audi.html">Audi</a></li>
    <li><a href="bmw.html">BMW</a></li>
</ul>
```

如果需要在列表里添加数据，则需要在数据加载后执行 refresh()方法对列表进行数据更新。
例如：*$('ul').listview('refresh');*

以上是运用 jQuery Mobile 进行界面构建的基础规则，后续将完善 Ajax 和动态创建页面的技术资料。