Introduction to JQTouch

When writing HTML5 Mobile Apps there are several Frameworks that have been created to help you along the way. Two of them are Modernizr and JQTouch. We will focus on JQTouch in this document and will cover Modernizr later.

You should recongnize JQTouch having used it in the previous document on Mobile Web Styling. In that document however we focused on only using its existing images and icons in our work. Here we are going to expand our use of the system to make a fully functioning iPhone App. Before you begin you will need the following:

* JQTouch. I have included 2 versions with this document. The first is a simplified version that I have used in the past and we will use in this document. It consists of the theme directory and the jqt directory. The second is the full version of JQTouch (stored in JQTouchFull) available from: <http://jqtouch.com>. I included this because it includes several nice examples of how to use the framework that you can look through when working on your own web app.
* An HTML5 editor. I am using Komodo Edit which can be found here:

<http://www.activestate.com/komodo-edit>. Make sure you click on Komodo Edit and not Komodo IDE (IDE is not free). You may also use Notepad++ or Dreamweaver if you prefer.

You should also keep the following in mind while working through this tutorial: 1) I am focusing on using JQTouch in this document and not on HTML5. You will notice that I fall back to using div tags for everything instead of placing article tags and such. The difference has no effect as all of our styling is still done in CSS. 2) I am using the default JQTouch CSS theme styles. You can change it by including your own if you like but I wanted you to be able to see what the package had available for you.

Now that all of that nonsense is out of the way let’s begin. We are going to complete a tutorial that will produce a simple 5 screen app that allows a person to count calories. Our app should have the following parts:

* A home screen
* An about screen
* A screen that lists all of the dates available to add calories ( we will do 1 week)
* A screen to see the calories of a specific day
* A screen to make a new entry to add a day.

The final result has been included with this document as index.html. We will start with a basic HTML5 setup. Open your web editor and enter the text below:

<!DOCTYPE html>

<html>

<head>

<title>Calorie Counter</title>

</head>

<body>

</body>

</html>

Save the file as counter.html

Now that we have our basic structure, we can begin building our app. We know that we want to use JQTouch and that JQTouch is built on top of JQuery. So, in order for our page to work we have to include both of these items. This will be our next step. By the way, I have renamed jquery to jquery.js. The actual name of the file included with JQTouch is jqtouch-jquery.js. If you download your own version of JQTouch you will need to change the name as I have.

To add JQuery and JQTouch modify the <head> section of counter.html so that it looks like the following:

<head>

<meta charset="utf-8">

<title>Calorie Counter</title>

<meta name="description" content="Calorie Counter">

<meta name="author content="Jonathan Blair">

<link rel="stylesheet" media="screen" href="jqt/jqtouch.css">

<link rel="stylesheet" media="screen" href="theme/jqtouch/theme.css">

<script src="jqt/jquery.js"></script>

<script src="jqt/jqtouch.js"></script>

<script type="text/javascript">

var jQT= $.jQTouch({

icon: 'image.png',

statusBar: 'black'

});

</script>

</head>

Let’s step through the code line by line:

* The first 4 lines are our standard head information we have always included, defining the character set, providing a title and giving some brief information about the document and the author
* The 5th line is a reference to JQTouch’s animation handling style sheet. I have separated out their animation styling from their basic page styling so you can review them. However, I do not recommend doing this if you want to use JQTouch yourself.
* Line 6 is the CSS that styles our page. This is the file you can either modify or replace to make your page look different
* Since JQTouch is built on JQuery, it requires JQuery be loaded before it will work. We do this on line 7.
* Next we load JQTouch itself.
* The final thing that we do is set some properties for how our app will work. Specifically, we set our Web Clip icon (the icon that will be displayed on the desktop of the device for our app). This pic should be 57 pixels square and should be in .png format. We also set the Status Bar color of our app to black. This sets the color of the 20 pixel strip across the top of the screen when the app is in full screen mode (which JQTouch places you in automatically).

There are a ton of other properties you can have JQTouch set for you automatically. Here is a link to the JQTouch documentation on the topic which you should review before moving forward: <https://github.com/senchalabs/jQTouch/wiki/Initialization%20Options>

Now that we have included JQTouch, we can start blocking out our web app inside the <body> tag. The first screens we will want to add will be the Home screen (where the user starts) and the About screen. We do this because we know every app we build will have these screens. Here is the code to create these two screens (which you should place inside the body tag of your counter.html file):

<div id="home">

<div class="toolbar">

<h1>Calorie Counter</h1>

</div>

<ul class="edgetoedge">

<li class="arrow"> <a href="#about">About</a> </li>

</ul>

</div>

<div id="about">

<div class="toolbar">

<h1>About</h1>

<a class="button back" href="#">Back</a>

</div>

<div>

<p>Simple Program to Practice Multiple Screen Apps</p>

</div>

</div>

That’s it. It should be harder than that (and it would be without JQTouch) but it’s not. We can create all of our different screens simply as different sections of the same page. Notice that each section is specified with an id tag. If you remember from our discussions, id tags have the extra ability to be used in navigation and we take advantage of that here. Beyond that it is simply applying the prebuilt JQTouch classes to our sections as needed. The toolbar class creates the nice black grid background for each of our headers at the top of our sections. The edgetoedge class tells the page to run the list all the way across the page and the “button back” class tells the page to place a nice little back button in place of our standard <a> link. More information about JQTouch theming can be found at the github link above.

If you really want to see the effects to JQTouch, comment out the include lines we added to our head section and look at the page without JQTouch’s styling and animation effects.

Now let’s add in the tab that is going to list the dates where the user has entered Calories. Here is the code that should be added below the About div:

<div id="dates">

<div class="toolbar">

<h1>Dates</h1>

<a class="button back" href="#">Back</a>

</div>

<ul class="edgetoedge">

<li class="arrow"><a id="0" href="#date">Today</a></li><li class="arrow"><a id="1" href="#date">Yesterday</a></li><li class="arrow"><a id="2" href="#date">2 Days Ago</a></li>

<li class="arrow"><a id="3" href="#date">3 Days Ago</a></li>

<li class="arrow"><a id="4" href="#date">4 Days Ago</a></li>

<li class="arrow"><a id="5" href="#date">5 Days Ago</a></li>

<li class="arrow"><a id="6" href="#date">6 Days Ago</a></li

</ul>

</div>

There is only one difference here to the code we already had, each <a> link has a specific identifier (id property). This is so we can determine which day’s information to display based on the correct link selected.

Before we can move on we need to update our home screen to have access to our newly created Dates panel. Add the following line *Above* the About list item as shown:

<ul class="edgetoedge">

**<li class="arrow"><a href="#dates">Dates</a></li>**

<li class="arrow"> <a href="#about">About</a> </li>

</ul>

Since we mentioned above that we would need to display the correct information for the selected date, let’s add the code to create the single date panel:

<div id="date">

<div class="toolbar">

<h1>Date</h1>

<a class="button back" href="#">Back</a>

<a class="button slideup" href="#createEntry">+</a>

</div>

<ul class="edgetoedge">

<li id="entryTemplate" class="entry" style="display:none">

<span class="label">Label</span>

<span class="calories">000</span>

<span class="delete">Delete</span>

</li>

</ul>

</div>

There is a little more going on here so let’s look at everything we haven’t seen previously. The first new item is button slideup. We know that button back creates a back button for our screen so button slideup should logically create a button that will slide the new panel up from the bottom instead of from the side as we normally see. You can test this by playing with the button types.

The other new item is style=”display-none”. We have seen this in the past and know that this basically renders the list item invisible. Why would we do this? Because the list item is our template for creating new entries. No reason to list all of these items until the new entry button has been clicked so we hide them.

Since we mentioned the fact that we would need to create a new entry panel we might as well do that now. Add the following below the date div tag created above:

<div id="createEntry">

<div class="toolbar">

<h1>New Entry</h1>

<a class="button cancel" href="#">Cancel</a>

</div>

<form method="post">

<ul class="rounded">

<li><input type="text" placeholder="Food" name="food" id="food" autocapitalize="off" autocorrect="off" autocomplete="off" /></li>

<li><input type="text" placeholder="Calories" name="calories" id="calories" autocapitalize="off" autocorrect="off" autocomplete="off" /></li>

<li><input type="submit" class="submit" name="action" value="Save Entry" /></li>

</ul>

</form>

</div>

The only big change here is the inclusion of a form. We have a form with 2 text boxes and 1 submit button. The type and placeholder properties should be familiar to you from our discussions. The other properties are specific to mobile devices and allow you to turn off or on the autocorrect, autocomplete and autocap features most devices use.

That’s it. If you check your page in Safari you will have a fairly nice looking app with multiple screens. We could expand this to track weight loss by adding another screen that allows a user to enter their starting weight and weekly weight and performing the subtraction for them (btw there are 3500 calories in a pound if you want to try). Currently however, we haven’t stored anything anywhere. We will cover that in the web storage tutorial.