Fall 2012

Auto Populated jQuery Accordion

By: Max DeAngelis

Purpose:

The purpose of this tutorial is to provide a step-by-step guide for creating a dynamically populated accordion using jQuery UI, PHP, and JSON. This will be a very simple example and the JSON will be minimal.

PHP will be utilized as the server side code, which will respond to the client's (JavaScript) request by returning textual data formatted as JSON. You will be using Ajax to make this request.

Required:

In order to complete this tutorial you will need a couple of things. To run the PHP, server side code, you will need a PHP interpreter like Apache. If you are using a Mac, MAMP is an easy to install App that provides a fully packaged Apache server that can run locally on your system. If you are using Windows you can use WampServer. If you are using Linux simply install Apache (httpd).

Other than the interpreter all you need is your choice of code editor or text editor, and a browser (recommend Firefox or Chrome).

Steps:

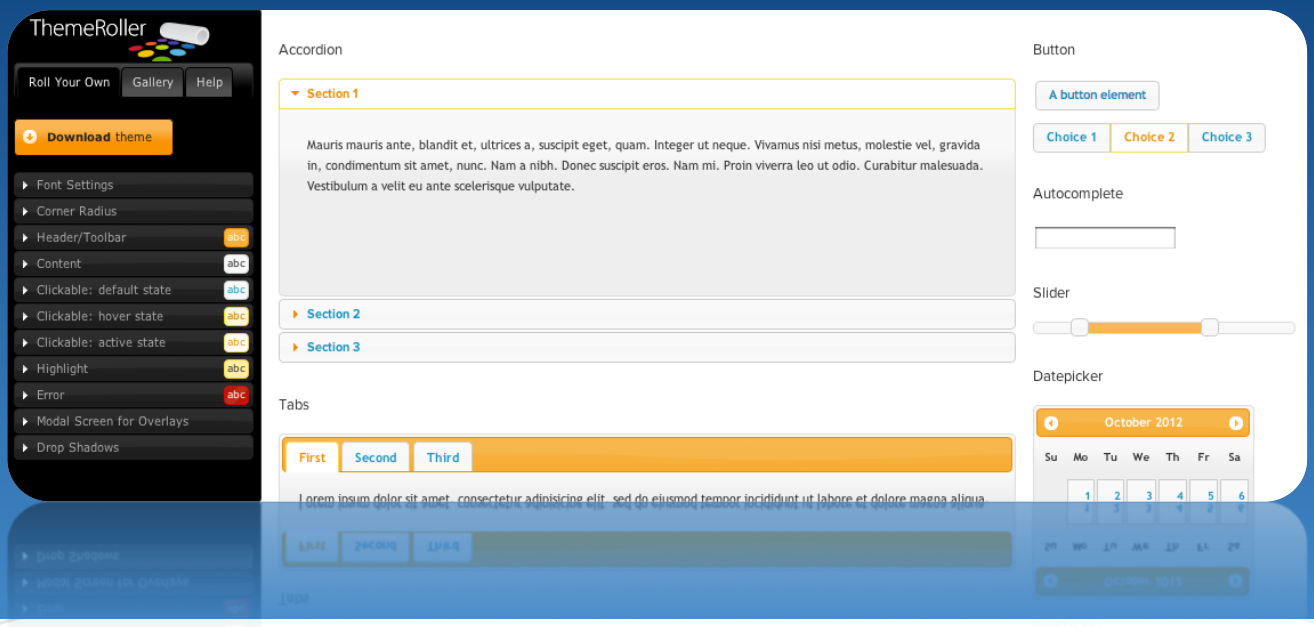
Step 1:

Download jQuery UI:

The first thing that you need to do in order to complete this tutorial is download jQuery UI. You will need to go to http://jqueryui.com/download/ and download the entire package. You can also use the Theme Roller located at the bottom of the page to custom make a theme you would like. The Theme Roller (See image below) allows you to pick the highlight colors for the UI package before you download. This allows you to select colors for highlight, hover, active, inactive, and much more. This alone allows you to add a very nice look to your UI.

The UI collection will download as a zip file, which will need to be extracted to your server's http folder. Once this is done you are ready for step 2.

Optional: If you prefer you can rename the jQuery UI, jQuery, and custom-ui files to something easier to remember. This only allows the script includes to be a bit simpler, you only have to include the files once so it does not effect your code much.



Step 2:

Create Simple HTML Page:

In order for jQuery UI to be able to create the Accordion control it needs a to reference on the HTML page. The easiest way to do this is to simply create a basic HTML page with a containing a unique ID.

In step six you will be referring to this unique ID in order to create the Accordion control. See image to the right for an example HTML page.

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Accordion Tutorial Page</title>
    </head>
    <body>
        <h1>Accordion Tutorial Page</h1>
        <!-- Accordion -->
        <div id="accordion">
            <!-- Data Goes Here -->
        </div>
    </body>
</html>
```

Step 3:

Including Proper jQuery Files:

In this step you will need to add the script includes in order to use the jQuery and jQuery UI features.

All you will need to do is add two <script> and one <link> tag to specify the location of jQuery, jQuery UI, and the jQuery UI CSS file. For the source location of these includes you need to specify the locations of the files relative to the location of the HTML file you are currently editing. See image below for an example.

Note: If you create your HTML file within the jQuery UI folder you can simply copy the locations from the image below, but be aware if you renamed the folder containing the CSS file you will need to update that location. In this example the CSS folder is called "custom-theme".

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Accordion Tutorial Page</title>
        <link href="css/custom-theme/jquery-ui-1.9.0.custom.css" rel="stylesheet">
        <script src="js/jquery-1.8.2.js"></script>
        <script src="js/jquery-ui-1.9.0.custom.js"></script>
    </head>
    <body>
        <h1>Accordion Tutorial Page</h1>
        <!-- Accordion -->
        <div id="accordion">
            <!-- Data Goes Here -->
        </div>
    </body>
</html>
```

Step 4:

Create Server Side Controller:

In this step you will be creating a server side controller using PHP. This controller is what returns data to the client which will then be inserted into the accordion control.

In this tutorial, to keep things simple, the data on the server will simply be a hard coded array of sample data. This sample data will then be encoded as JSON and returned to the client. See image below for an example controller. This file should be stored within the root folder of your personal server, so that it can be interpreted. This controller file, like all PHP controllers, should end with a .php extension. In this example it is called tutorial.php.

Note: In a production site this server side controller would be the perfect mechanism to query data from a database or other data source. That data could then be stored in a similarly formatted array and returned to the client as JSON to be displayed in the accordion.

```php
<?php
    if (isset($_POST['action'])){
            $data[0]['title'] = "Header 1";
            $data[0]['content'] = "Some Data 1";
            $data[1]['title'] = "Header 2";
            $data[1]['content'] = "Some Data 2";
            $data[2]['title'] = "Header 3";
            $data[2]['content'] = "Some Data 3";
            //Converting to JSON
            $json = json_encode($data);
            //Echo JSON for client
            echo $json;
    }
?>
```

Step 5:

Making The Connection:

In this step you will need to use a jQuery Ajax call to call the PHP controller you created in the last step in order to retrieve the sample data.

Ajax is a wonderful tool used to communicate with the server. In this tutorial we will put this call inline in the HTML page. This will cause the Ajax call to be executed when the page is done loading, which is fine for this example. In a production site you might want to associate the Ajax call to some other sort of event like a button click.

The image below is a very basic example of the format you should use for the Ajax call. This call will provide the necessary avenue for retrieving the JSON from the server. In the next step you will be parsing the returned JSON to populate the accordion.


```
<script type="text/javascript">
    $(document).ready(function() {
        $.ajax({
            type: 'POST',
            url: '/tutorial.php',
            dataType: "json",
            data: { action: 'true'},
            success: function(output){

                //This is where the JSON will be parsed

            }//End of success
        });
    });
</script>
```

Step 6:

Populating The Data:

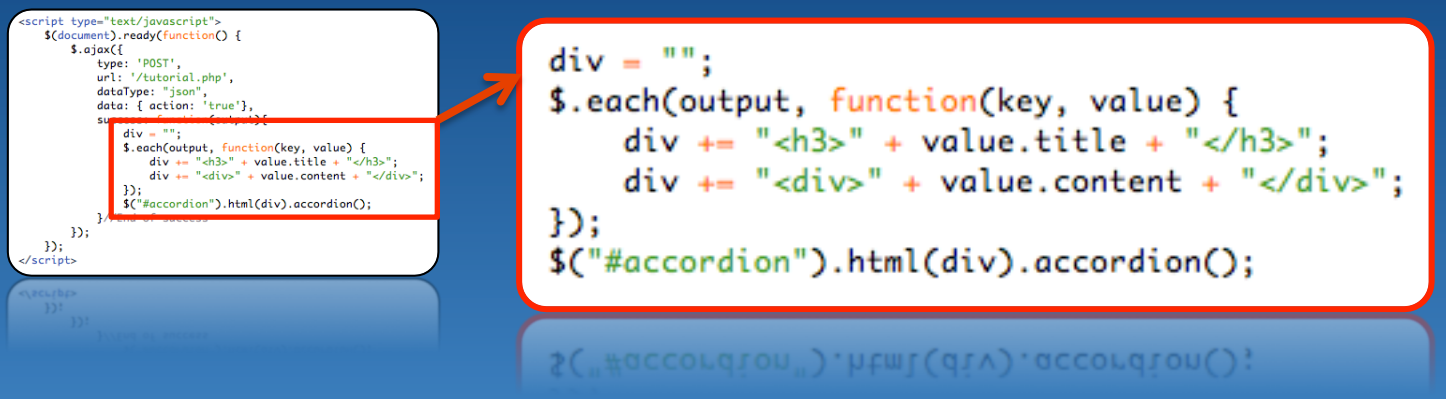
In this step you will need to use an each loop to parse the JSON data that was returned from the data and generate a HTML string that can be inserted into the empty on your HTML page.

JSON acts a lot like an associated array. The each loop in JavaScript allows you to dereference the JSON. In order to reference the individual value you reference it using the key value assigned in the PHP array. For example if you reference value.title while inside of the each loop you would see the value associated with the item with the key named title. In this example this would be the values: “Header 1”, “Header 2”, and “Header 3”. In order to expose the value stored with a key of content you would just specify value.content within the each loop.

Now that you know how to reference the values returned by the server, you will need to construct some HTML within the loop to inject into the empty . The empty div will then be turned into the accordion control. To construct the HTML string, simply concatenate strings and the dereferenced values with a plus sign.

The format needed to create an element of the accordion is first a header (<h3> is default), followed by a containing the content for each element. See the image below for the format needed.

Lastly to create the control you will need to inject the constructed HTML string into the empty using the .html() function, and then activate the control using the .accordion() function.



```
<script type="text/javascript">
    $(document).ready(function() {
        $.ajax({
            type: 'POST',
            url: '/tutorial.php',
            dataType: "json",
            data: { action: 'true'},
            su
                div = "";
                $.each(output, function(key, value) {
                    div += "<h3>" + value.title + "</h3>";
                    div += "<div>" + value.content + "</div>";
                });
                $("#accordion").html(div).accordion();
            }
        });
    });
</script>
```

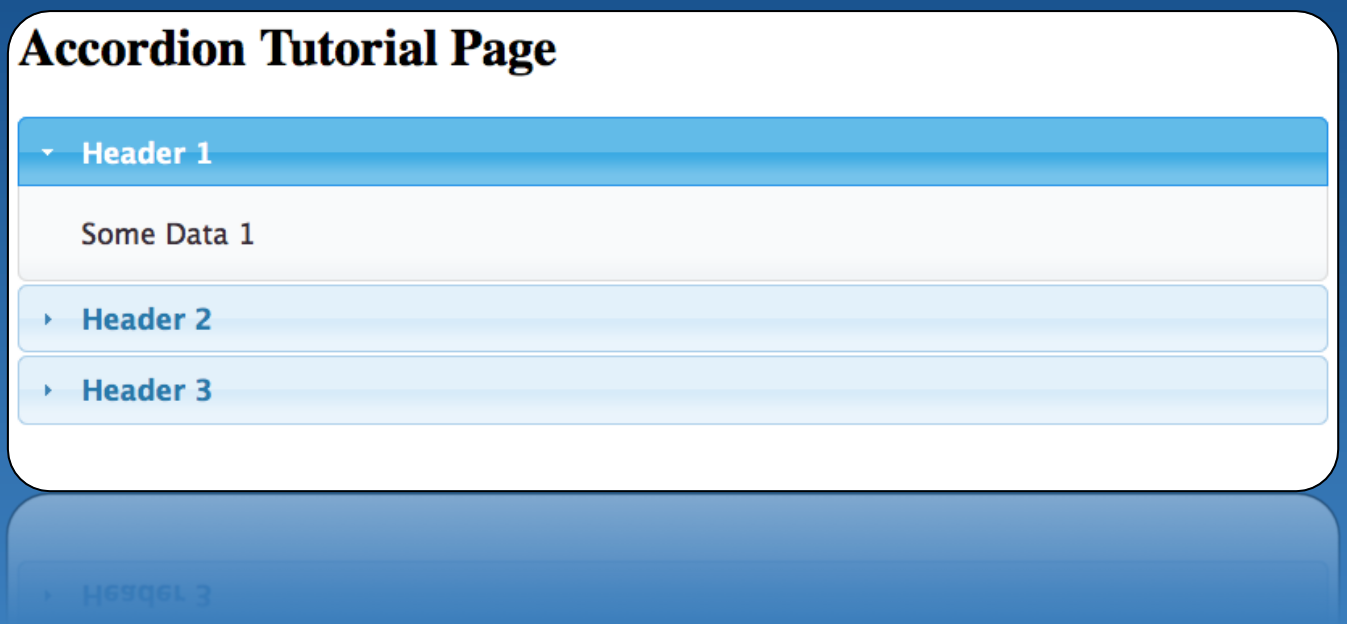
```
div = "";
$.each(output, function(key, value) {
    div += "<h3>" + value.title + "</h3>";
    div += "<div>" + value.content + "</div>";
});
$("#accordion").html(div).accordion();
```

Step 7:

See It In Action:

All there is left to do is open your HTML file in a browser of your choice. To get the full effect you should avoid using Internet Explore, as the border rounding does not work and the appearance will be slightly blocky. However, the accordion will work in all browsers so please use your preferred browser. **Note:** In order for the PHP to be interred your server should be running.

Congratulations you have just created a dynamically generated accordion using jQuery UI. Below you will see an example of how it should be rendered. Please refer to the code examples in the appendix if you have questions or errors.



Appendix:

Index.html:

```
<!doctype html>
    <html>
        <head>
            <meta charset="utf-8">
            <title>Accordion Tutorial Page</title>
            <link href="css/custom-theme/jquery-ui-1.9.0.custom.css" rel="stylesheet">
            <script src="js/jquery-1.8.2.js"></script>
            <script src="js/jquery-ui-1.9.0.custom.js"></script>

            <script type="text/javascript">
                $(document).ready(function() {
                    $.ajax({
                        type: 'POST',
                        url: '/tutorial.php',
                        dataType: "json",
                        data: { action: 'true'},
                        success: function(output){
                            div = "";
                            $.each(output, function(key, value) {
                                div += "<h3>" + value.title + "</h3>";
                                div += "<div>" + value.content + "</div>";
                            });
                            $("#accordion").html(div).accordion();
                        }//End of success
                    });
                });
            </script>

        </head>
        <body>
            <h1>Accordion Tutorial Page</h1>
            <!-- Accordion -->
            <div id="accordion">
                <!-- Data Goes Here -->
            </div>
        </body>
    </html>
```

Tutorial.php:

```php
<?php
    if (isset($_POST['action'])){
            $data[0]['title'] = "Header 1";
            $data[0]['content'] = "Some Data 1";
            $data[1]['title'] = "Header 2";
            $data[1]['content'] = "Some Data 2";
            $data[2]['title'] = "Header 3";
            $data[2]['content'] = "Some Data 3";
            //Converting to JSON
            $json = json_encode($data);
            //Echo JSON for client
            echo $json;
    }
?>
```