JQuery AJAX Http polling example

Posted At : 26 September 2010 19:05 | Posted By : Shaun McCran
Related Categories: JQuery, Coldfusion, Json, AJAX

I've been using the JQuery post() and get() functions for a while now, and thanks to decent Blog entries from other community members I've got my head around the principles of seamless AJAX http requests and response handling.

This article examines a way of creating a polling AJAX http request. This is a request that will run every *N* seconds based on a value. It will hit a remote service and return a result, and display that result on screen.

View a full demo of an AJAX polling request here.

The remote service CFC

The remote service I am using in this example is a Coldfusion CFC that that returns the date and time. It simply formats the date and time and returns it as a JSON object. You can use the CFJSON CFC or just specify a returnformat value of 'JSON', in the CFC function (depends on your server version).

```
                            <cffunction name="getTime"
                                        access="remote"
                                         output="true"
                                       returntype="void"
                                hint="returns the time to remote calls">

                                  <cfset var response = "Test">
'{"status":"success","data":{"date":"#dateformat(now(), 'dd-mm-yyyy')#","time":"#d

                                <cfoutput>#response#</cfoutput>
                                    </cffunction>
```

Amended

After testing I discovered that the CFC above included extra whitespace that wrecks the JSON response in IE browsers. Instead I am using a cfm template like this.

```
                                    <cfsilent>
                                <cfsetting showDebugOutput=false>
                              <cfsetting enablecfoutputonly="true">
                          <cfprocessingdirective suppresswhitespace="true">
ıs":"success","data":{"date":"#dateformat(now(), 'dd-mm-yyyy')#","time":"#dateForma
                                    </cfprocessingdirective>
                                        </cfsilent>
                                    <cfoutput>#response#</cfoutput>
```

JQuery AJAX http polling request

Include the JQuery libraries from Google, and the polling.js plugin. Nick Riggs has taken the JQuery functions of ajax(); get() and post(); and extended them to create a plugin: http://www.nickriggs.com/posts/simple-ajax-polling-plugin-for-jquery/ .

```
:="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.0/jquery.
'text/javascript" src="http://ajax.googleapis.com/ajax/libs/swfobject/2.2/swfobjec
 <s/cript language="javascript" src="polling.js" type="text/javascript"></script>
```

The code below is one example of how you can use several of the default values to set the type of polling to 'interval' IE continuous polling at a certain number of seconds. the ajaxPoll() method accepts a series of values specifying the url to request, any data to pass to it and how the response should be handled.

I've wrapped this in a click event to show how you could initialise the polling function. I'd quite like to create a stop event as well, but haven't quite got that working. (Hint)

The example uses the JQuery append() method to display the result in a div. I'm incrementing a counter in this routine as well, the counter is all client side to avoid passing data from client to server that you don't need to. Remember keep it as light as possible!

```
                    <s/cript type="text/javascript">
                         $(document).ready(function(){

                         $("#start").click(function(){

                       $.ajaxPollSettings.pollingType = "interval";
                           $.ajaxPollSettings.interval = 5000;

                                   var httpcount = 1;

                                      $.ajaxPoll({
                                 url: "map-service.cfc?method=getTime",
                                            type: "POST",
                                       data: { count: httpcount},
                                           dataType: "json",
                                    successCondition: function(result) {
                                // return result != null; // custom condition goes here.

var responseString = httpcount + '. ' + result.data.date + ' ' + result.da

                                $("#response-container").append(responseString);

                                             httpcount = httpcount + 1;

                                                  },
                                          success: function(result) {
                                             // alert(result.status);
                                           // alert(result.data.date);
                                                    }
                                                });

                                           });

                                  $("#stop").click(function(){

                                 alert('will write stop function here');

                                           });

                                      });
                                   </script>
```

This is the div that shows the response, and the start / stop buttons.

```
<input type="button" value="start" id="start">
 <input type="button" value="stop" id="stop">

          <h2>Ajax responses</h2>
      <div id="response-container"></div>
```

If you use firebug or Charles (web proxy tracking software) you can watch each request firing. Most of the request return a difference of five seconds, but I see the odd delayed one with a slightly longer time.

View a full demo of an AJAX polling request here.

I built this to use in a Google Map application, to poll map points, so look out for that article next.