AJAX

CS174

Chris Pollett

Nov. 7, 2007.

Outline

- AJAX

Beginning AJAX

- AJAX stands for Asynchronous Javascript and XML.
- Asynchronous means that one does not need to wait for the response to an HTTP request to do something
  - one can use Javascript to have several outstanding HTTP requests and still make things happen in the browser.
  - Further one doesn't have to reload the entire page when the results of a request are returns.
- The XML in AJAX is because the results of a request are usual XML, JSON, or YAML data which is then formatted by some Javascript code.
- The basic key to AJAX is the ability for Javascript to make an HTTP request. This is done with the XMLHttpRequest object. As an example, look at the Javascript in the file:

http://www.cs.sjsu.edu/faculty/pollett/test/dept3.html

Step by Step

- To create an XMLHttpRequest one could simply write in Javascript:

```
request = new XMLHttpRequest()
```

- This works on modern browsers. For older browsers like IE6 you need to do something like:

```
request = new ActiveXObject(‘MSXML2.XMLHTTP’)
```

- For older still use:

```
          request = new ActiveXObject(‘Microsoft.XMLHTTP’)
```

- See the example page for how checking for browsers can be done using a try-catch block.
- At this point if we need to set up HTTP request headers one can use:
  request.setRequestHeader(“name”, “value”)

Step by Step Continued I

- To open a connection back to the server the Javascript came from one can then do:

  request.open(*theHTTPmethod, theURL, theAsync flag*)

  or

  request.open(*theHTTPmethod, theURL, theAsync flag, username, password*)
- For example,

  request.open("GET", "progress.php", true)
- The asynchronous flag says whether or not the Javascript can continue executing (true) or if it must wait for a response (false).
- At this point no data has been sent yet.

Step by Step Continued II

- We next need to set up a callback function which will be called as we get data back from the server:

```
var self = this; /* remember scope of enclosing
                        object */
request.onreadystatechange = function()
{
    switch(request.readyState)
   {
      case 0:// handle uninitialized case
      case 1: // handle open but no send case
      case 2: // handle send but no response case
      case 3: // handle response is being downloaded case
      case 4: // handle response has completed being downloaded case
   }
}
```

Step by Step Continued III

- In case 3 or 4 above, request.responseText or request.responseXML will contain the data that has been returned by the server
- In case 3 or 4 above, request.getAllResponseHeaders() can be used to get the HTTP response headers.
- Sometimes it is useful to start a timer in case 3, using setTimout. If the request is taking to long so that the timer gets called, one can then abort the request using: request.abort.
- Typically, the code in 3 or 4 above would put the response text into some tag in our document. This could be done with:

```
document.getElementById(“myDivTag”).innerHTML = \ request.responseText; /*not really a
     DOM standard but all browsers happy with */
//or we could do…
myDiv = document.getElementById(“myDivTag”)
if(myDiv.firstChild)
{
     myDiv.removeChild(myDiv.firstChild);
}
myDiv.appendChild(document.createTextNode(request.responseText));
```

Step by Step Continued IV

- Once we have set up our callback function we are ready to send data to the server.
- To do this we can do:

```
request.send(null);
/* note: send’s argument can be used if using
   POST method to send the posted data */
```

- That’s it! We just sit back then and wait for our callback to be called.

Example

- We then looked at a couple AJAX examples, such as the dept3.html code mentioned above.

Proxies

- Javascript function is only allowed to make requests back to the server from which it came.
- So if you have a page http://somewhere.com/index.html

  and you would like the Javascript on it to make use of the Yahoo! Rest API, how do you do it?
- You need to use a proxy on your server which passes the request onto Yahoo!
- One example of a PHP script to do such proxy-ing can be found at:

http://developer.yahoo.com/javascript/samples/proxy/php_proxy_simple.txt

- To use such a proxy, you need to have PHP running on your machine.
- You could rename the above file proxy.php set its permissions so that is executable and put it somewhere you know under your document root.
- Then to access the Yahoo! service via the proxy you could do:

http://yourServer/proxy.php?yws_path=urlencodepath

- For example,

http://www.cs.sjsu.edu/faculty/pollett/test/proxy.php?yws_path=NewsSearchService%2FV1%2FnewsSearch%3Fappid%3DYahooDemo%26query%3Dmadonna%26results%3D2%26language%3Den

Example

- Looked at proxy code from Yahoo!
- It is another example of using the PHP curl_init, curl_exec, curl_setopt, curl_close.