Dynamic Web Development

Lecture 17 – Ajax 2

What is Ajax?

Asynchronous
Javascript
and
XML

A bit misleading, as it doesn't have to be asynchronous, and it doesn't have to involve XML.

The name for a collection of technologies that had existed for some time, but web developers had never combined them in this way.

The term was coined by Jesse James Garrett in:

"Ajax: A New Approach to Web Applications"

http://www.adaptivepath.com/ideas/essays/archives/000385.php

With traditional web page design, if you want something on the page to change, you have to refresh the entire page.

The browser issues an HTTP request.

The server issues an HTTP response.

The new information is displayed in the browser.

The key to Ajax applications, is the use of scripted HTTP requests.

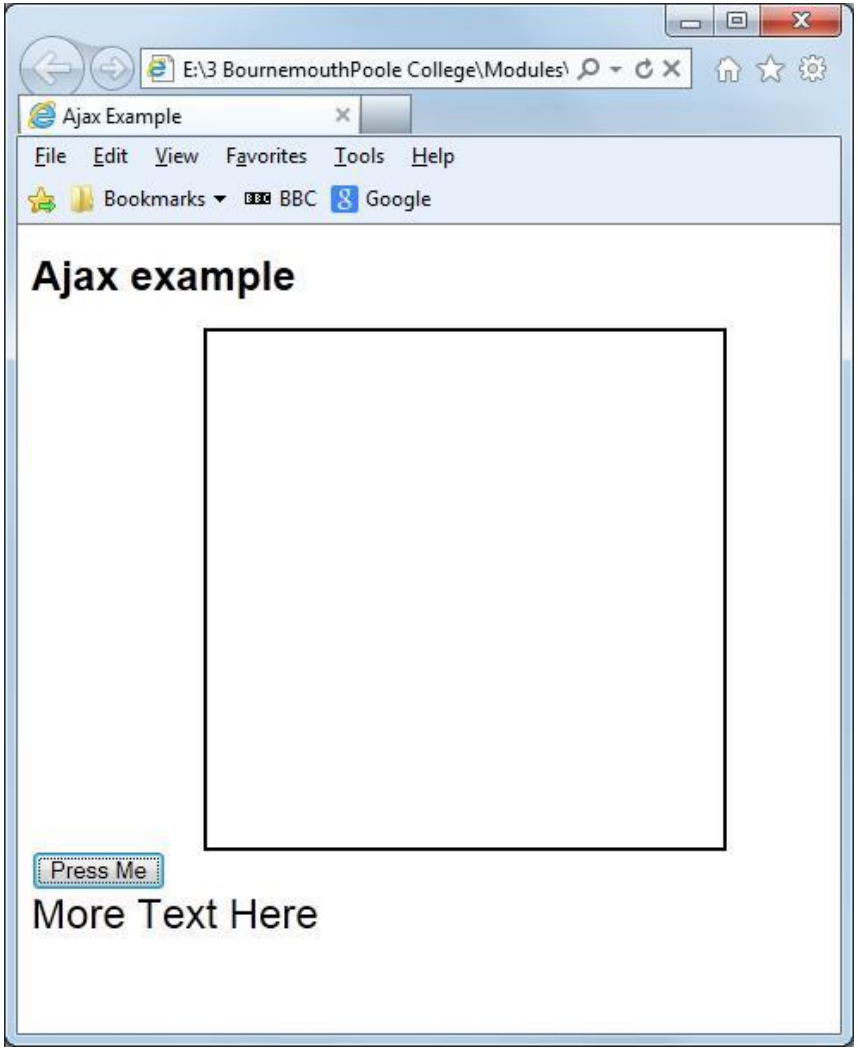
Since the amount of data transferred is often very small, and the browser doesn't have to parse and render an entire web page, response time is improved.

An html file

```
<!DOCTYPE html>

<html>
 <head>
  <title>Ajax Example</title>
  <link type="text/css" rel="stylesheet" href="Ajax03.css"></link>
  <script type="text/javascript" src="jquery-1.7.1.js"></script>
  <script type="text/javascript" src="Ajax03.js"></script>
 </head>

 <body>
   <h1>Ajax example</h1>
   <div id="window"></div>
   <button id="one">Press Me</button>
   <div id="text2">More Text Here</div>
 </body>
</html>
```

CSS file

```
h1
{
font-size: 18pt;
font-family: Arial;
}

div#window
{
margin-left:100px;
width: 300px;
height: 300px;
border: solid 2px black;
}

div#text2
{
font-size: 18pt;
font-family: Arial;
}
```


E:\3 BournemouthPoole College\Modules\
Ajax Example
File Edit View Favorites Tools Help
Bookmarks BBC Google
Ajax example
Press Me
More Text Here

testdata.txt

```
This is a test file containing some text.  I want it
to appear in a document when I click on a button,
without having to refresh the page.  I need to use a
remote http request to get the file. This is called
the Ajax technique.
```

In more complex Ajax applications, this data would be encoded in XML format.

XML (Extensible Markup Language) is a way of encoding documents electronically. The standards are laid down by the W3C.

Using jQuery

The jQuery library provides a number of Ajax functions that are essentially wrapper methods.

That is to say, they shield you from some of these inner workings of a scripted HTTP request.

Shorthand methods:

jQuery.load()

jQuery.get()

jQuery.post()

Full method:

jQuery.ajax()

$.load( ) Ajax03.js

```
$(document).ready(function(){
   $("button#one").click(function(){
      $("div#window").load('test1.txt');
   });
});
```

Syntax is:

Wait until document has finished loading {
  When the button is clicked, run this function {
    Load into div#middle the file test1.txt
   }
}

Would this work?

```
$(document).ready(function(){
  $("button#one").click(function(){
    $("div#middle").load('gettext.php');
  });
});
```

and gettext.php contained something like this:

```
<?php
$text1='This is more text that ';
$text2='I want to send.';
echo $text1.$text2;
?>
```

$.get( ) Ajax04.js

Load( ) is very simple.

A more flexible approach is offered by the $.get( ) method.

We can supply the server with a query string – equivalent to passing variables through the url.

We can also specify a function that displays or processes the returned data.

The syntax is as follows:

$.get(url, { key1: value1, key2: value2 }, optionalsuccessfunction)

Ajax04.js

```
$(document).ready(function()
    {
    $("button#one").click(function()
        {
        $.get('getdata4.php',
                { name:"Wilson", age:"50" },
                    function(data)
                        {
                            // Process the data in some way
                            // or, in this case, just display it
                            var element = document.getElementById("window");
                            element.innerHTML = data;
                        }
            )
        });
    });
```

getData4.php

```
<?php
$na = $_GET['name'];
$ag = $_GET['age'];
echo "Hello ".$na. " you are currently ".$ag." years old";
?>
```

http://api.jquery.com/jQuery.get/

jQuery.get()

Categories: Ajax > Shorthand Methods

jQuery.get(url [, data] [, success(data, textStatus, jqXHR)] [, dataType]) *Returns: jqXHR*

Description: *Load data from the server using a HTTP GET request.*

jQuery.get(url [, data] [, success(data, textStatus, jqXHR)] [, dataType]) version added: 1.0

url
Type: String
A string containing the URL to which the request is sent.

data
Type: PlainObject, String
A plain object or string that is sent to the server with the request.

success(data, textStatus, jqXHR)
Type: Function()
A callback function that is executed if the request succeeds.

dataType
Type: String
The type of data expected from the server. Default: Intelligent Guess (xml, json, script, or html).

This is a shorthand Ajax function, which is equivalent to:

```
1 $.ajax({
2   url: url,
3   data: data,
4   success: success,
5   dataType: dataType
6 });
```

Documentation page contains more examples of use of $.get()

json formatted data Ajax05.js

```
$(document).ready(function()
    {
    $("button#one").click(function()
        {
        $.get('getData5.php',
                  function(data)
                      {
                          var element = document.getElementById("window");
                          element.innerHTML = data.name + " " + data.time;
                      },
                  'json'
              )
        });
    });
```

getData5.php

```
<?php
  echo '{"name":"John", "time":"4pm"}';
?>
```

Use of the network tools on Internet Explorer's Developer tools is especially useful when debugging this sort of code.

JSON: JavaScript Object Notation

JSON is syntax for storing and exchanging text information. Much like XML.

JSON is smaller than XML, and faster and easier to parse.

JSON Example – an employees object which is an array of 3 employee records (which are also objects)

```
{
"employees":
    [
        { "firstName":"John" , "lastName":"Doe" },
        { "firstName":"Anna" , "lastName":"Smith" },
        { "firstName":"Peter" , "lastName":"Jones" }
    ]
}
```

Much Like XML

- JSON is plain text
- JSON is "self-describing" (human readable)
- JSON is hierarchical (values within values)
- JSON can be parsed by JavaScript
- JSON data can be transported using AJAX

Much Unlike XML

- No end tag
- Shorter
- Quicker to read and write
- Can be parsed using built-in JavaScript eval()
- Uses arrays
- No reserved words

JSON: JavaScript Object Notation

JSON parsers and JSON libraries exists for many different programming languages.

The JSON text format is syntactically identical to the code for creating JavaScript objects.

Because of this similarity, instead of using a parser, a JavaScript program can use the built-in eval() function and execute JSON data to produce native JavaScript objects.

This makes it a lot easier to handle than the equivalent XML file.

JSON Syntax

JSON syntax is a subset of the JavaScript object notation syntax.

- Data is in name/value pairs
- Data is separated by comma
- Curly brackets holds objects
- Square brackets holds arrays

A name/value pair consists of a field name (in double quotes), followed by a colon, followed by a value:

"firstName" : "John"

This is simple to understand, and equals to the JavaScript statement:

firstName = "John"

Values, Objects and Arrays

JSON values can be:

- A number (integer or floating point)
- A string (in double quotes)
- A Boolean (true or false)
- An array (in square brackets)
- An object (in curly brackets)
- null

JSON objects are written inside curly brackets, and can contain multiple name/value pairs

```
{ "firstName":"John" , "lastName":"Doe" }
```

JSON arrays are written inside square brackets, and can contain multiple objects.

```
{
    "employees":
        [
            { "firstName":"John" , "lastName":"Doe" },
            { "firstName":"Anna" , "lastName":"Smith" },
            { "firstName":"Peter" , "lastName":"Jones" }
        ]
}
```

JSON Uses JavaScript Syntax

Because JSON uses JavaScript syntax, no extra software is needed to work with JSON within JavaScript.

With JavaScript you can create an array of objects and assign data to it like this:

```
var employees = [
                  { "firstName":"John" , "lastName":"Doe" },
                  { "firstName":"Anna" , "lastName":"Smith" },
                  { "firstName":"Peter" , "lastName": "Jones" }
                ];
```

The first entry in the object array can be accessed like this: employees[0].lastName;

The returned content will be: Doe

The data can be modified like this: employees[2].firstName = "Jonathan";

Converting a JSON Text to a JavaScript Object

One of the most common use of JSON is to fetch JSON data from a web server (as a file or as an HttpRequest), convert the JSON data to a JavaScript object, and then it uses the data in a web page.

JSON Parser

The eval() function can compile and execute any JavaScript. This represents a potential security problem.

It is safer to use a JSON parser to convert a JSON text to a JavaScript object. A JSON parser will recognize only JSON text and will not compile scripts.

In browsers that provide native JSON support, JSON parsers are also faster.

Native JSON support is included in newer browsers and in the newest ECMAScript (JavaScript) standard.

Retrieving json formatted data

jQuery.getJSON()

Categories: Ajax > Shorthand Methods

jQuery.getJSON(url [, data] [, success(data, textStatus, jqXHR)]) *Returns: jqXHR*

Description: *Load JSON-encoded data from the server using a GET HTTP request.*

jQuery.getJSON(url [, data] [, success(data, textStatus, jqXHR)]) version added: 1.0

url
Type: String
A string containing the URL to which the request is sent.

data
Type: PlainObject
A plain object or string that is sent to the server with the request.

success(data, textStatus, jqXHR)
Type: Function()
A callback function that is executed if the request succeeds.

This is a shorthand Ajax function, which is equivalent to:

```
1 $.ajax({
2   dataType: "json",
3   url: url,
4   data: data,
5   success: success
6 });
```

json formatted data Ajax06.js

```
$(document).ready(function()
    {
    $("button#one").click(function()
        {
        $.getJSON('getData6.php',
                function(data)
                    {
                      var element = document.getElementById("window");
                      element.innerHTML = data.name + " lives in " + data.town;
                    }
                 )
        });
    });
```

getData5.php

```
<?php
  //  Instead of just    echo '{"name":"Alan", "town":"Poole"}';
  // I will demonstrate use of PHP function json_encode

  $employees =  array("name" => "Alan", "town" => "Poole");
  $codedemp = json_encode($employees);
  echo $codedemp;
?>
```

Full List of jQuery AJAX statements

.ajaxComplete()
.ajaxError()
.ajaxSend()
.ajaxStart()
.ajaxStop()
.ajaxSuccess()
jQuery.ajax()
jQuery.ajaxPrefilter()
jQuery.ajaxSetup()
jQuery.ajaxTransport()
jQuery.get()
jQuery.getJSON()
jQuery.getScript()
jQuery.param()
jQuery.post()
.load()
.serialize()
.serializeArray()

The $.ajax() function underlies all Ajax requests sent by jQuery. It is often unnecessary to directly call this function, as several higher-level alternatives like $.get() and .load() are available and are easier to use. If less common options are required, though, $.ajax() can be used more flexibly.