Ajax Technology in Web Programming

Sergio Luján Mora

Asynchronous JavaScript and XML

AJAX
Ajax Technology in Web Programming

Index

• AJAX
• XMLHttpRequest
• AJAX step by step
• Example
• More information

AJAX

• **Ajax**, shorthand for *Asynchronous JavaScript and XML*
• Web development technique for creating interactive web applications
• The intent is to make web pages feel more responsive by exchanging small amounts of data with the server behind the scenes, so that the entire web page does not have to be reloaded each time the user makes a change
• This is meant to increase the web page's interactivity, speed, and usability
AJAX

- The first known use of the term in public was by Jesse James Garrett in his February 2005 article *Ajax: A New Approach to Web Applications*
- At subsequent talks and seminars Garrett has made the point that Ajax is not an acronym
AJAX

- The Ajax technique uses a combination of:
  - XHTML (or HTML), CSS, for marking up and styling information.
  - The DOM accessed with a client-side scripting language, especially ECMAScript implementations such as JavaScript and JScript, to dynamically display and interact with the information presented.
  - The XMLHttpRequest object to exchange data asynchronously with the web server. In some Ajax frameworks and in certain situations, an IFrame object is used instead of the XMLHttpRequest object to exchange data with the web server.
  - XML is sometimes used as the format for transferring data between the server and client, although any format will work, including preformatted HTML, plain text, JSON and other formats.
- Like DHTML, LAMP, or SPA, Ajax is not a technology in itself, but a term that refers to the use of a group of technologies together.
XMLHttpRequest

- XMLHttpRequest is an API that can be used by JavaScript, JScript, VBScript and other web browser scripting languages to transfer and manipulate XML data to and from a web server using HTTP, establishing an independent connection channel between a web page's Client-Side and Server-Side.

XMLHttpRequest

- The XMLHttpRequest concept was originally developed by Microsoft.
- The Microsoft implementation is called XMLHTTP and, as an ActiveX object, it differs from the published standard in a few small ways. It has been available since Internet Explorer 5.0 and is accessible via JScript, VBScript and other scripting languages supported by IE browsers.
XMLHttpRequest

- The Mozilla project incorporated the first compatible native implementation of XMLHttpRequest in Mozilla 1.0 in 2002.
- This implementation was later followed by Apple since Safari 1.2, Konqueror, Opera Software since Opera 8.0 and iCab since 3.0b352.

XMLHttpRequest

- While this is still a work in progress, its goal is "to document a minimum set of interoperable features based on existing implementations, allowing Web developers to use these features without platform-specific code".
- The draft specification is based upon existing popular implementations, to help improve and ensure interoperability of code across web platforms.
XMLHttpRequest

- abort()
- getAllResponseHeaders()
- getResponseHeader(header)
- open(method, url, async, user, password):
- send(content)
- setRequestHeader(header, value)

open(method, url, async, user, password):
- Initializes an XMLHTTP request.
- Specifies the method, URL, and authentication information for the request.
- After calling this method, you must call send to send the request and data, if any, to the server.
XMLHttpRequest

• `send(content)`:
  – Sends an HTTP request to the server and receives a response.
  – `null` for no data

• Properties:
  – `onreadystatechange`
  – `readyState`
  – `responseText`
  – `responseXML`
  – `status`
  – `statusText`
XMLHttpRequest

- `onreadystatechange`: Function that handles the different events

- `readyState`: The property is read-only
  - It represents the state of the request as an integer
  - The following values are defined:
XMLHttpRequest

- **readyState:**
  - 0 (UNINITIALIZED): The object has been created, but not initialized (the open method has not been called)
  - (1) LOADING: The object has been created, but the send method has not been called.
  - (2) LOADED: The send method has been called, but the status and headers are not yet available.
  - (3) INTERACTIVE: Some data has been received. Calling the `responseText` property at this state to obtain partial results will return an error, because status and response headers are not fully available.
  - (4) COMPLETED: All the data has been received, and the complete data is available in the `responseText` property.

- **responseText:**
  - The property is read-only.
  - This property represents only one of several forms in which the HTTP response can be returned.
XMLHttpRequest

- `responseXML`:
  - The property is read-only.
  - This property represents the parsed response entity body.

AJAX step by step

1. Create XMLHttpRequest object
2. Assign a function to the state change event
3. Send a request to the server
4. On a state change, manage the response
5. On a correct response, process the result and show to the user
Create XMLHttpRequest object

- Depending on the browser:
  - Internet Explorer
    ```javascript
    request = new ActiveXObject("Microsoft.XMLHTTP");
    ```
  - Otros navegadores:
    ```javascript
    request = new XMLHttpRequest();
    ```
- Code adapted for different browsers:
  ```javascript
  if(window.XMLHttpRequest) {
    request = new XMLHttpRequest();
  } else if(window.ActiveXObject) {
    request = new ActiveXObject("Microsoft.XMLHTTP");
  }
  ```

Assign a function to the state change event

- This function will be called automatically, every time the state of the XMLHttpRequest object changes:
  ```javascript
  request.onreadystatechange = nameOfFunction
  ```

**Important**: without “( )”, only the name
Ajax Technology in Web Programming

**Send a request to the server**

- Open the connection, define the method and the type of connection:
  - A synchronous connection (false) blocks the browser until the response is obtained
  - An asynchronous connection (true and default value) executes on the background
  - Important: the URL must belong to the same domain of the current page

```javascript
request.open('GET', 'http://www.ua.es/ajax.jsp', true);
```

- Send the additional data:

```javascript
request.send(data or null)
```

Ajax Technology in Web Programming

**On a state change, manage the response**

- The handler is called every time there is a change:
  - 0: UNINITIALIZED
  - 1: LOADING
  - 2: LOADED
  - 3: INTERACTIVE
  - 4: COMPLETED

- Example of handler:

```javascript
if (request.readyState == 4) { // Finished
    if (request.status==200) { // OK
        // Process the result
    }
} else {
    // Not finished
}
```
On a correct response, process the result and show to the user

- The result can be in different formats: plain text, HTML, JSON, XML, etc.
- `responseText` when not structured result as XML:
  ```
  alert(request.responseText);
  ```
- `responseXML` when structured result as XML:
  - Returns an XMLDocument object
  - Use DOM functions

Example

```javascript

function ajaxFunction() {
    var xmlHttp;
    if (window.XMLHttpRequest)
        xmlHttp = new XMLHttpRequest();
    else
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");

    xmlHttp.onreadystatechange=function() {
        if (xmlHttp.readyState == 4) {
            document.myForm.time.value += xmlHttp.responseText + "\n";
        }
    }
    xmlHttp.open("GET", "time.php", true);
    xmlHttp.send(null);
}

</script>
```
Example

```javascript
function ajaxFunction() {
    var xmlHttp;
    if (window.XMLHttpRequest)
        xmlHttp = new XMLHttpRequest();
    else
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");

    xmlHttp.onreadystatechange=function() {
        if(xmlHttp.readyState == 4) {
            document.myForm.time.value += xmlHttp.responseText + "\n";
        }
    }
    xmlHttp.open("GET","time.php",true);
    xmlHttp.send(null);
}
</script>
```
Ajax Technology in Web Programming

Example

```javascript
function ajaxFunction() {
  var xmlHttp;
  if (window.XMLHttpRequest)
    xmlHttp = new XMLHttpRequest();
  else
    xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");

  xmlHttp.onreadystatechange=function() {
    if(xmlHttp.readyState == 4) {
      document.myForm.time.value += xmlHttp.responseText + "\n";
    }
  }
  xmlHttp.open("GET","time.php",true);
  xmlHttp.send(null);
}
</script>
```

Send a request to the server

On a state change, manage the response
Example

```html
<html>
<head>
<title>Ajax example</title>
</head>
<body>
<form name="myForm">
Name: <input type="text" onkeyup="ajaxFunction()" name="username" />
<br />
Time: <textarea name="time" cols="40" rows="10"></textarea>
</form>
</body>
</html>
```

On a correct response, process the result and show to the user
Example

• PHP:

```php
<?php
    header("Expires: -1");
    $str1 = date('h:i:s A');
    sleep(2);
    $str2 = date('h:i:s A');
    echo "$str1 -- $str2";
?>
```

More information

• W3C:
  – The XMLHttpRequest Object (W3C Working Draft 05 April 2006)
  – http://www.w3.org/TR/XMLHttpRequest/

• Microsoft:
  – MSDN: IXMLHTTPRequest
  – MSXML 4.0 SDK
The XMLHttpRequest Object

W3C Working Draft 05 April 2006

This version:
http://www.w3.org/TR/2006/WD-xmlHttpRequest-20060405/

Latest version:
http://www.w3.org/TR/2006/WD-xmlHttpRequest/

Editor:
Anne van Kesteren, Opera Software ASA <annev@opera.com>

Copyright ©2006 W3C (MIT, ERCIM, Keio), All Rights Reserved. W3C trademark and document use rules apply.

Abstract
This specification defines the XMLHttpRequest object, an API that provides some HTTP client functionality.

Status of this Document
This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at:
http://www.w3.org/standards/index-technical BrowserModule/
### XMLHttpRequest Members

The following tables show the properties, methods, and events.

#### Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onreadystatechange</td>
<td>Specifies the event handler to be called when the readystatechange property changes. Read/Write.</td>
</tr>
<tr>
<td>readyState</td>
<td>Represents the state of the request. Read-only.</td>
</tr>
<tr>
<td>responseText</td>
<td>Represents the response text body as a string. Read-only.</td>
</tr>
<tr>
<td>responseXML</td>
<td>Represents the response XML document. Read-only.</td>
</tr>
<tr>
<td>status</td>
<td>Represents the HTTP status code returned by a request. Read-only.</td>
</tr>
</tbody>
</table>

#### Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abort</td>
<td>Closes the current HTTP request.</td>
</tr>
<tr>
<td>getAllResponseHeaders</td>
<td>Retrieves the values of all the HTTP headers.</td>
</tr>
<tr>
<td>getResponseHeader</td>
<td>Retrieves the value of an HTTP header from the response body.</td>
</tr>
<tr>
<td>setRequestHeader</td>
<td>Specifies the name of an HTTP header.</td>
</tr>
<tr>
<td>send</td>
<td>Sends an HTTP request to the server and receives a response.</td>
</tr>
</tbody>
</table>

#### Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>To view reference information for Visual Basic, C/C++, or Script only, click the Language Filter button in the upper left corner of the page.</td>
</tr>
</tbody>
</table>