Departamento de Lenguajes y Sistemas Informáticos

Universitat d'Alacant
Universidad de Alicante

Ajax Technology in Web Programming

Sergio Luján Mora

Asynchronous JavaScript and XML

AJAX

Index

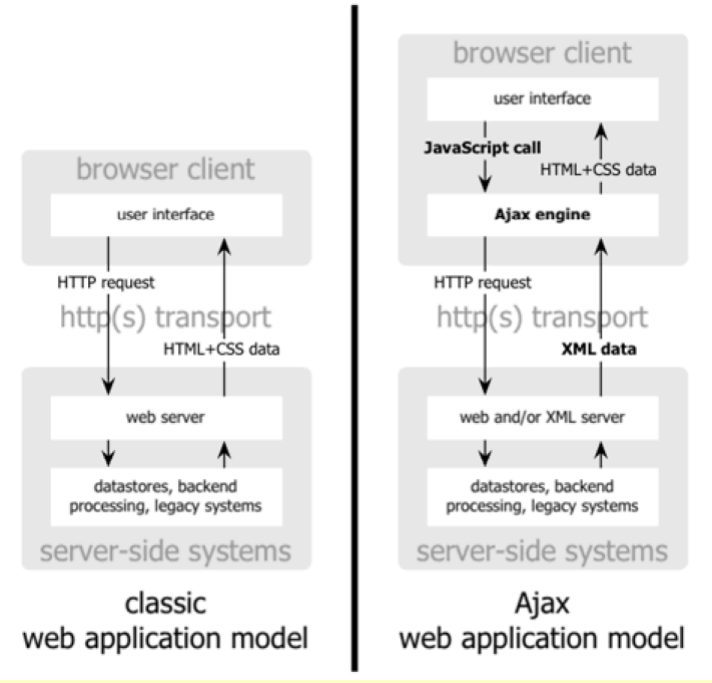
- AJAX
- XMLHttpRequest
- AJAX step by step
- Example
- More information

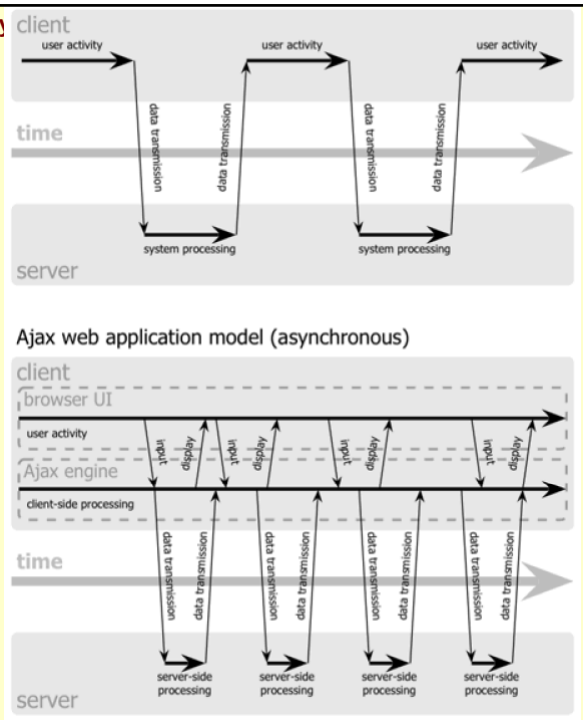
AJAX

- **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

- The World Wide Web Consortium published a Working Draft specification for the XMLHttpRequest object's API on 5 April 2006.
- 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

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

XMLHttpRequest

- `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

XMLHttpRequest

- Properties:
  - `onreadystatechange`
  - `readyState`
  - `responseText`
  - `responseXML`
  - `status`
  - `statusText`

XMLHttpRequest

- `onreadystatechange`:
  - Function than handles the different events

XMLHttpRequest

- `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

XMLHttpRequest

- `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
    ```
    request = new ActiveXObject("Microsoft.XMLHTTP");
    ```
  - Otros navegadores:
    ```
    request = new XMLHttpRequest();
    ```
- Code adapted for different browsers:

```
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:

```
request.onreadystatechange = nameOfFunction
```

Important: without “( )”, only the name

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

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

- Send the additional data:

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


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:

```
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

```
<script type="text/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

```
<script type="text/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>
```

Create XMLHttpRequest object

Example

```
<script type="text/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>
```

Assign a function to the state change event

Example

```
<script type="text/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

Example

```
<script type="text/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>
```

On a state change, manage the response

Example

```
<script type="text/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>
```

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

Ajax Technology in Web Programming

Example

```
<html>
<head>
<title>Ajax example</title>
<!-- script -->
</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>
```

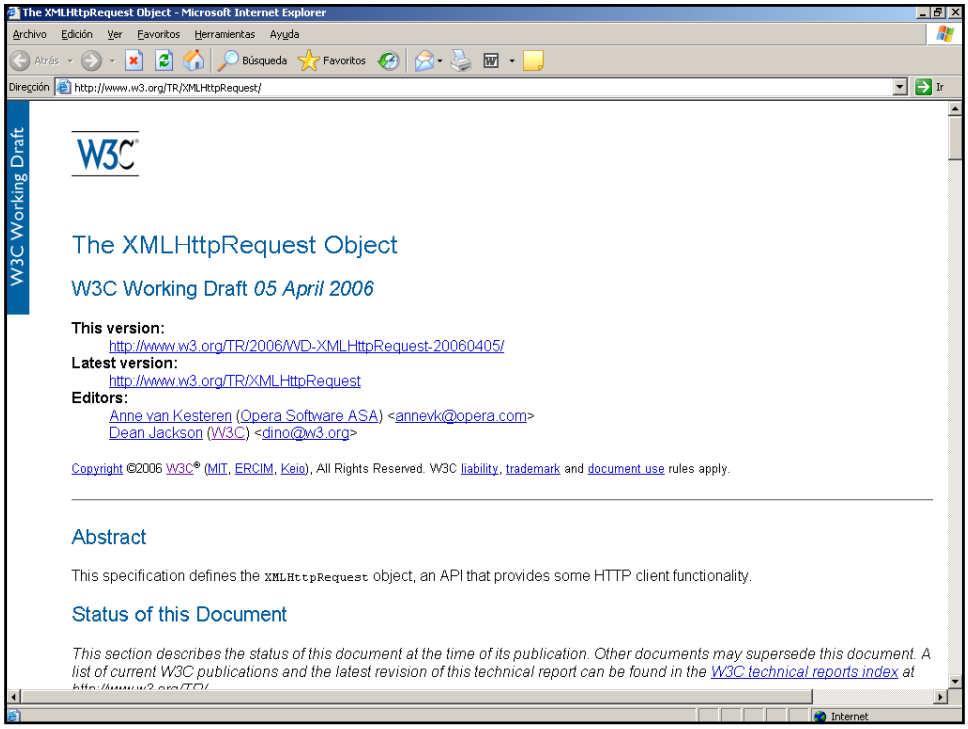
Example

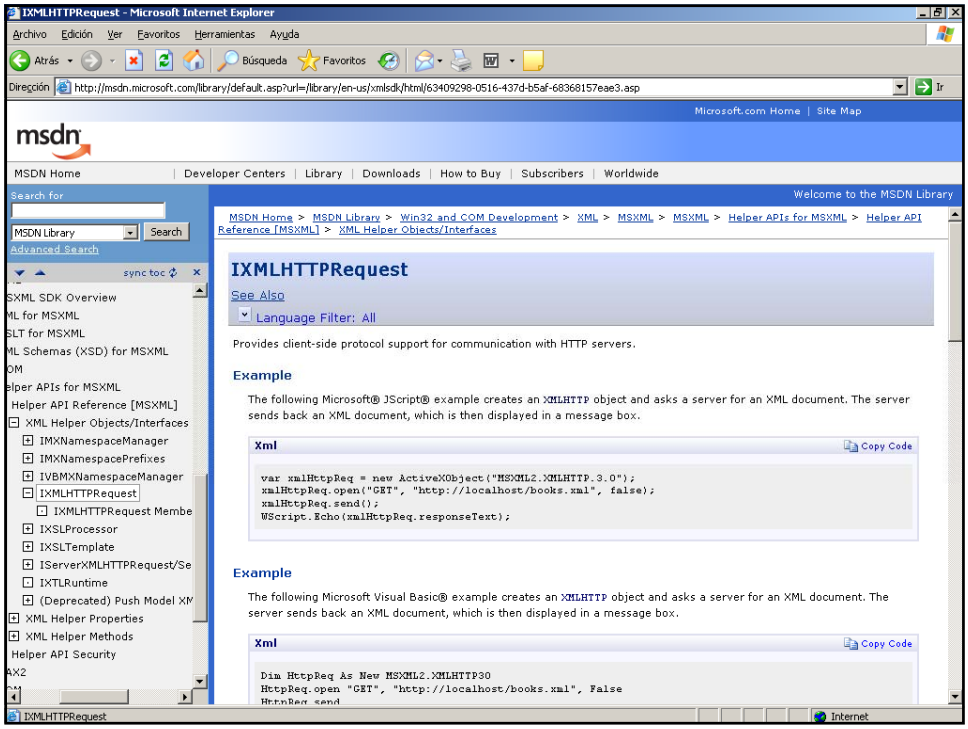
- PHP:

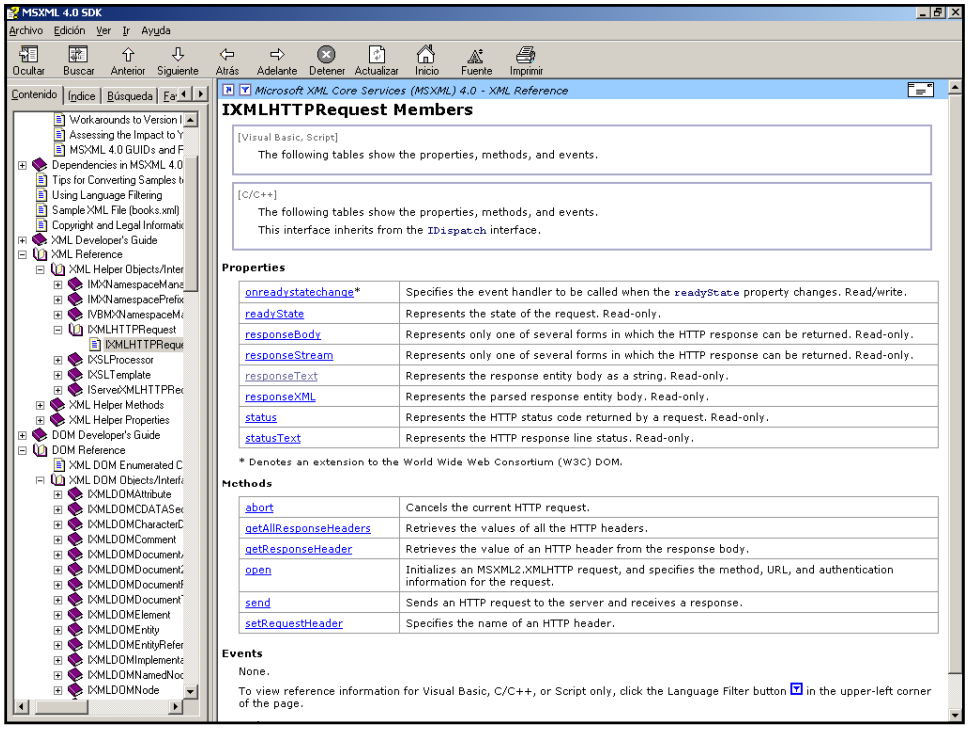
```
<?php
  header("Expires: -1");
  $str1 = date('h:i:s A');
  sleep(2);
  $str2 = date('h:i:s A');
  echo "$str1 -- $str2";
?>
```


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
  - http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsdk/html/xmobjxmlhttprequest.asp


The XMLHttpRequest Object - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Dirección http://www.w3.org/TR/XMLHttpRequest/
W3C Working Draft
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/XMLHttpRequest
Editors:
Anne van Kesteren (Opera Software ASA) <annevk@opera.com>
Dean Jackson (W3C) <dino@w3.org>
Copyright ©2006 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, 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
Internet


IXMLHTTPRequest - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Dirección http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsdk/html/63409298-0516-437d-b5af-68368157eae3.asp
Microsoft.com Home | Site Map
msdn
MSDN Home | Developer Centers | Library | Downloads | How to Buy | Subscribers | Worldwide
Search for
MSDN Library
Search
Advanced Search
Welcome to the MSDN Library
MSDN Home > MSDN Library > Win32 and COM Development > XML > MSXML > MSXML > Helper APIs for MSXML > Helper API Reference [MSXML] > XML Helper Objects/Interfaces
IXMLHTTPRequest
See Also
Language Filter: All
Provides client-side protocol support for communication with HTTP servers.
Example
The following Microsoft® JScript® example creates an XMLHTTP object and asks a server for an XML document. The server sends back an XML document, which is then displayed in a message box.
Xml
Copy Code
var xmlHttpReq = new ActiveXObject("MSXML2.XMLHTTP.3.0");
xmlHttpReq.open("GET", "http://localhost/books.xml", false);
xmlHttpReq.send();
WScript.Echo(xmlHttpReq.responseText);
Example
The following Microsoft Visual Basic® example creates an XMLHTTP object and asks a server for an XML document. The server sends back an XML document, which is then displayed in a message box.
Xml
Copy Code
Dim HttpReq As New MSXML2.XMLHTTP30
HttpReq.open "GET", "http://localhost/books.xml", False
IXMLHTTPRequest
Internet


MSXML 4.0 SDK
Archivo Edición Ver Ir Ayuda
Ocultar Buscar Anterior Siguiente Atrás Adelante Detener Actualizar Inicio Fuente Imprimir
Contenido Índice Búsqueda
Microsoft XML Core Services (MSXML) 4.0 - XML Reference
IXMLHTTPRequest Members
[Visual Basic, Script]
The following tables show the properties, methods, and events.
[C/C++]
The following tables show the properties, methods, and events.
This interface inherits from the IDispatch interface.
Properties
onreadystatechange* Specifies the event handler to be called when the readyState property changes. Read/write.
readyState Represents the state of the request. Read-only.
responseBody Represents only one of several forms in which the HTTP response can be returned. Read-only.
responseStream Represents only one of several forms in which the HTTP response can be returned. Read-only.
responseText Represents the response entity body as a string. Read-only.
responseXML Represents the parsed response entity body. Read-only.
status Represents the HTTP status code returned by a request. Read-only.
statusText Represents the HTTP response line status. Read-only.
* Denotes an extension to the World Wide Web Consortium (W3C) DOM.
Methods
abort Cancels the current HTTP request.
getAllResponseHeaders Retrieves the values of all the HTTP headers.
getResponseHeader Retrieves the value of an HTTP header from the response body.
open Initializes an MSXML2.XMLHTTP request, and specifies the method, URL, and authentication information for the request.
send Sends an HTTP request to the server and receives a response.
setRequestHeader Specifies the name of an HTTP header.
Events
None.
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.