

Tema 5: AJAX

- **Asynchronous JavaScript And XML**
- [http://en.wikipedia.org/wiki/Ajax_\(programming\)](http://en.wikipedia.org/wiki/Ajax_(programming))
- <http://es.wikipedia.org/wiki/AJAX>
- <http://www.w3.org/TR/XMLHttpRequest/>
- <http://www.w3schools.com/ajax/>

Objetivos

- Introducir los conceptos básicos relacionados con el modelo de desarrollo AJAX.
- Establecer la relación entre conceptos y herramientas presentados en la asignatura y AJAX.
- Introducir aspectos que permitan profundizar en esta opción y evaluarla como mejora en el desarrollo de sitios Web.

¿Qué es AJAX?

- AJAX es el acrónimo de **A**synchronous **J**avaScript **A**nd **X**ML
- AJAX es una técnica de desarrollo (más) para crear aplicaciones web interactivas
- AJAX se basa en el uso de JavaScript para enviar y recibir datos entre el navegador y el servidor
- La idea en que se fundamenta AJAX consiste en evitar la recarga completa de una página web cada vez que el usuario realiza un cambio
 - Puede aumentar la velocidad y, en consecuencia, la interactividad y la amabilidad del interfaz de usuario
- Con AJAX se utiliza el objeto **XMLHttpRequest** para enviar datos al servidor y XML como formato para recibir datos del servidor
 - En proceso de estandarización por el W3C (draft 30-1-2014)

AJAX

- ◆ Las aplicaciones que siguen el modelo AJAX se ejecutan en el lado del cliente. Usan transferencia de datos asíncrona entre el cliente y el servidor web, permitiendo a las páginas web solicitar pequeños fragmentos de información al servidor, en lugar de páginas completas
 - ◆ Aumenta la velocidad de las aplicaciones web
- ◆ Es tecnología del lado cliente, independiente del software del servidor web.
- ◆ AJAX está basado en JavaScript , XML , HTML y CSS
 - ◆ Una aproximación de una cierta profundidad hace conveniente un conocimiento de los DOM de HTML y XML, XSLT y SOAP

XMLHttpRequest

■ Constructor

```
var r = new XMLHttpRequest();
```

(para IE5 e IE6)

```
var r = new ActiveXObject("Microsoft.XMLHTTP");
```

XMLHttpRequest

■ Propiedades

■ **readyState**: contiene el *estado* del objeto


























(0:sin inicializar, 1:cargando, 2:cargado-headers, 3: interactuando, 4:completado)

Dottoro Web Reference

WordPress Theme & Editor

Browse by Name

Table of Contents

0	    	The current object is not initialized (the <u>open</u> method has not been called yet).
1	    	The request is opened, but the <u>send</u> method has not been called yet.
2	    	The request is sent but no data has been received yet.
3	    	A part of the data has been received, but it is not yet available.
4	    	All data is available.

Default: this property has no default value.

XMLHttpRequest

■ Propiedades

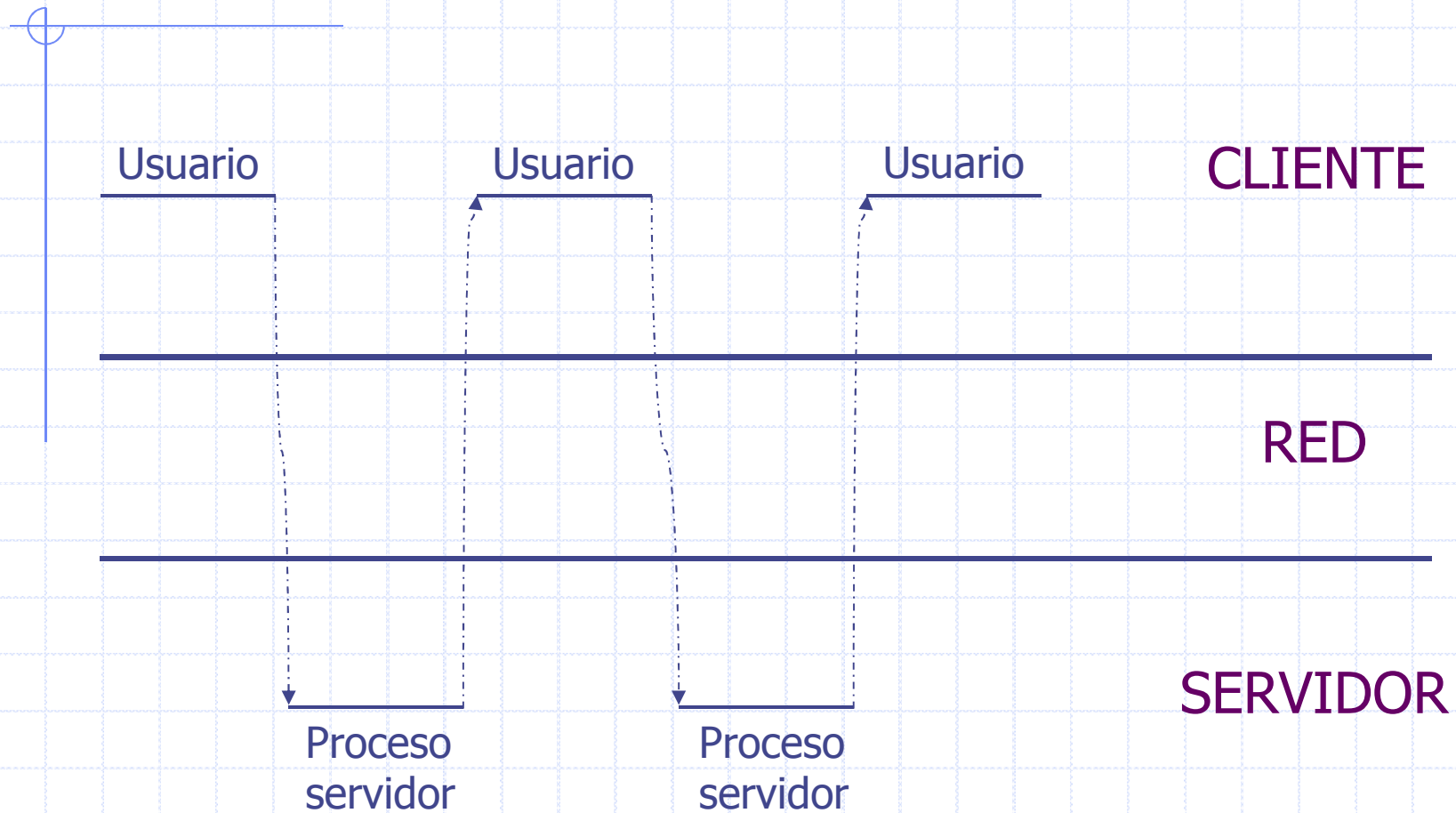
- **Status:** un entero ... (p.e.: 200)
- **statusText** : un string con el texto del status (p.e.: 200 OK)
- **responseText**: la respuesta devuelta como un *string* de caracteres
- **responseXml**: la respuesta devuelta como un objeto de tipo documento XML. Será accesible a través de los métodos y propiedades del DOM.

XMLHttpRequest

- **Evento**
 - **Onreadystatechange:** se invoca con cada cambio de *estado*
- **Métodos**
 - **open**(*método*, *url*, [*sincronía* [,*nombreusuario* [,*clave*]]])
 - ◆ *método*: tipo de solicitud (GET, POST o HEAD)*
 - ◆ *url*: localización del recurso al que se invoca
 - ◆ *sincronía*: true (asíncrono) / false (síncrono)
 - se para o se continua con el procesamiento del script
 - ◆ También pueden añadirse nombre de usuario y clave, en los casos que sean necesarios
 - **send**("string") se usa, con el método POST, para enviar información,
 - **abort()** cancela la actividad con el servidor

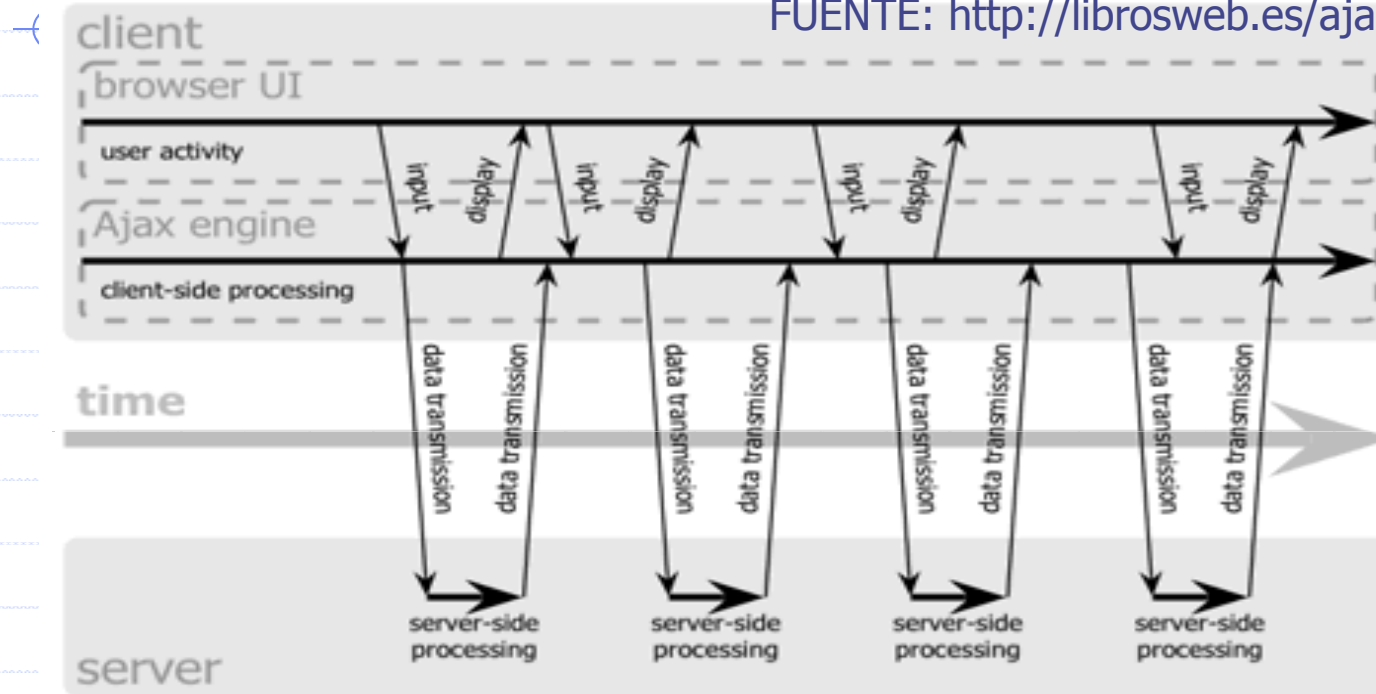
(*) **http 1.1** añade 5 tipos: *OPTIONS*, *PUT*, *DELETE*, *TRACE* y *CONNECT*

HttpRequest (síncrono)



XmlHttpRequest (asíncrono)

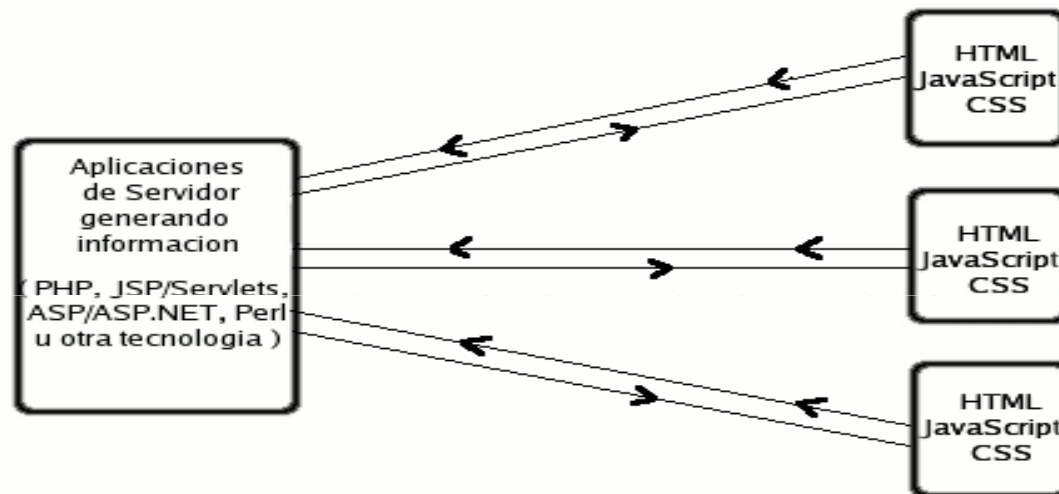
FUENTE: <http://librosweb.es/ajax/capitulo1.html>



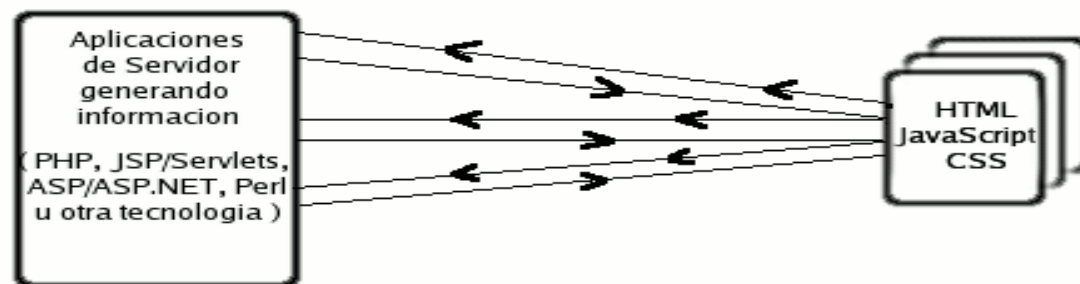
Ajax es una tecnología asíncrona, en el sentido de que los datos adicionales se requieren al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página.

Lo que ve el usuario...

SECUENCIA CLASICA SIN AJAX



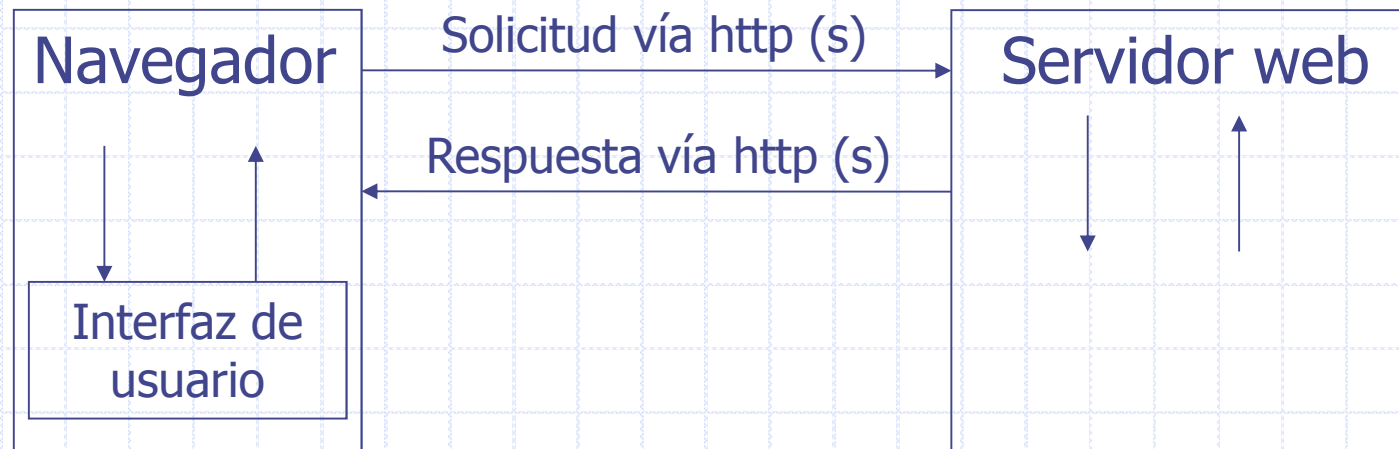
SECUENCIA EMPLEANDO AJAX

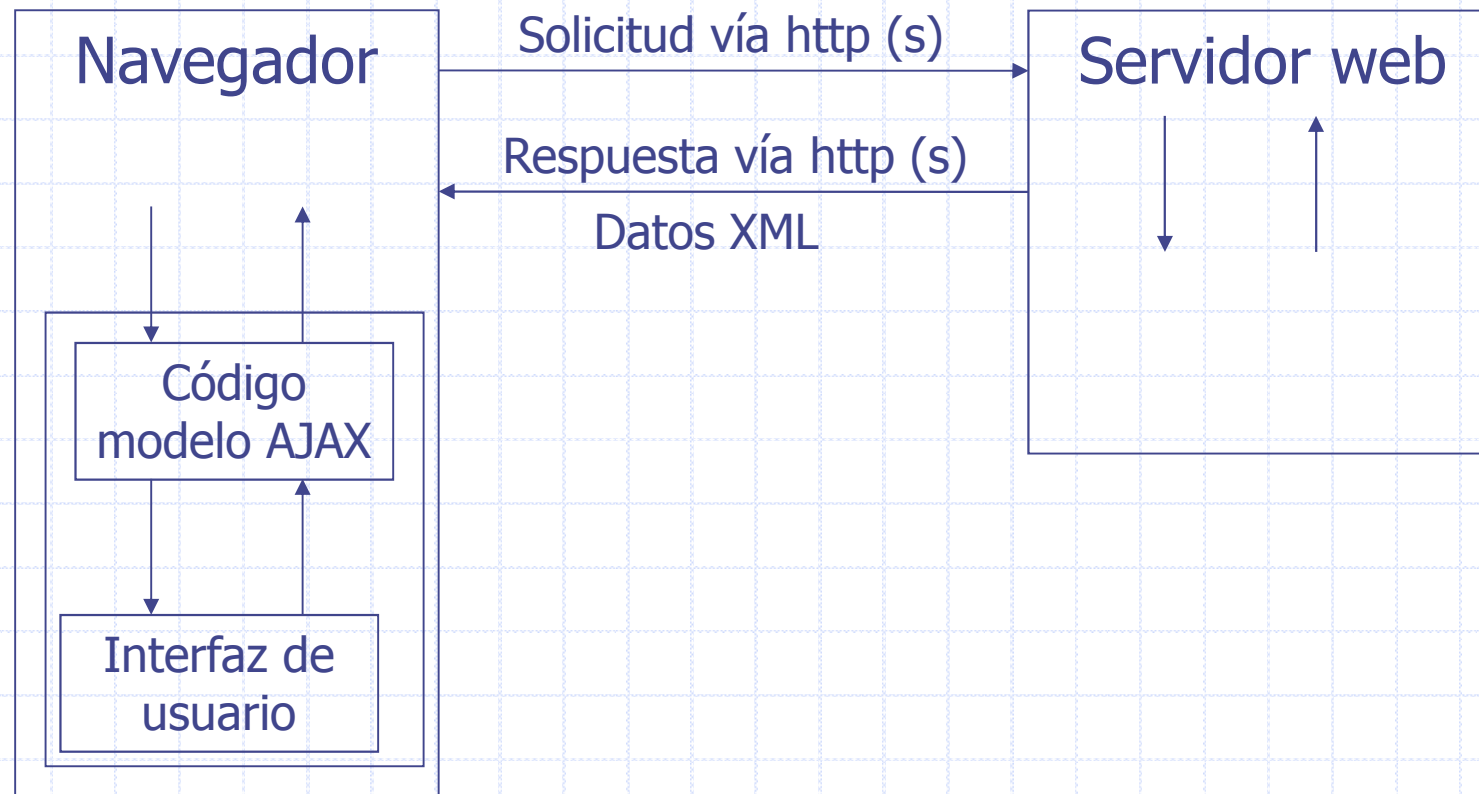


FUENTE:

Título: Introducción a AJAX

Autor: Javier Eguíluz Pérez





AJAX utiliza XML y XmlHttpRequest

- ◆ Las aplicaciones web convencionales utilizan el formulario HTML como forma de envío de la información de entrada al servidor web, el servidor web procesa la información y devuelve una página nueva completa al navegador (y, en última instancia, al usuario)
 - ◆ Este trasiego de información repercute en la velocidad de las aplicaciones web y en la amabilidad de los entornos de usuario
- ◆ Con AJAX las aplicaciones web pueden enviar y recibir datos sin recargar la página completa
 - ◆ Enviando una solicitud HTTP (HTTP Request) y modificando, utilizando JavaScript, sólo partes de la página web
- ◆ Aunque no es preceptivo, la forma preferible de comunicación con el servidor es que el envío de datos se haga utilizando XML

En síntesis

- ◆ Se asocia una función *javascript* a la situación en que se produce un evento determinado
- ◆ La función incluye código que invoca (en muchos casos de forma asíncrona), normalmente, a una página dinámica (asp, php, jsp,...) que devuelve un resultado XML (también puede ser otro formato textual)
- ◆ La información enviada por el servidor es tratada y mostrada (por el código javascript) accediendo a las propiedades del objeto correspondiente de la página HTML

Ejemplo 1: Visualizar el contenido de un fichero de texto ...

```
<body>
<h1>Mostrando datos con AJAX</h1>
<form>
<input type = "button" value = "Mostrar mensaje" onclick =
"pedirDatos()">
</form>
<div id="resultado" style="background-color:#99FF66;">
<p>Aqui aparecera texto</p>
</div>
</body>
</html>
```

Ejemplo 1: Visualizar el contenido de un fichero de texto ...

```
<script language = "javascript">
// (IE7+, Firefox, Chrome, Safari, and Opera)
XMLHttpRequestObject = new XMLHttpRequest();
XMLHttpRequestObject.onreadystatechange = function()
{
    alert (XMLHttpRequestObject.readyState);
    if (XMLHttpRequestObject.readyState==4)
    {var obj = document.getElementById('resultado');
    obj.innerHTML = XMLHttpRequestObject.responseText;}
}
function pedirDatos()
{
    XMLHttpRequestObject.open("GET", 'datos.txt');
    XMLHttpRequestObject.send(null);
}
</script>
```

Ejemplo 2: Manipular documento XML

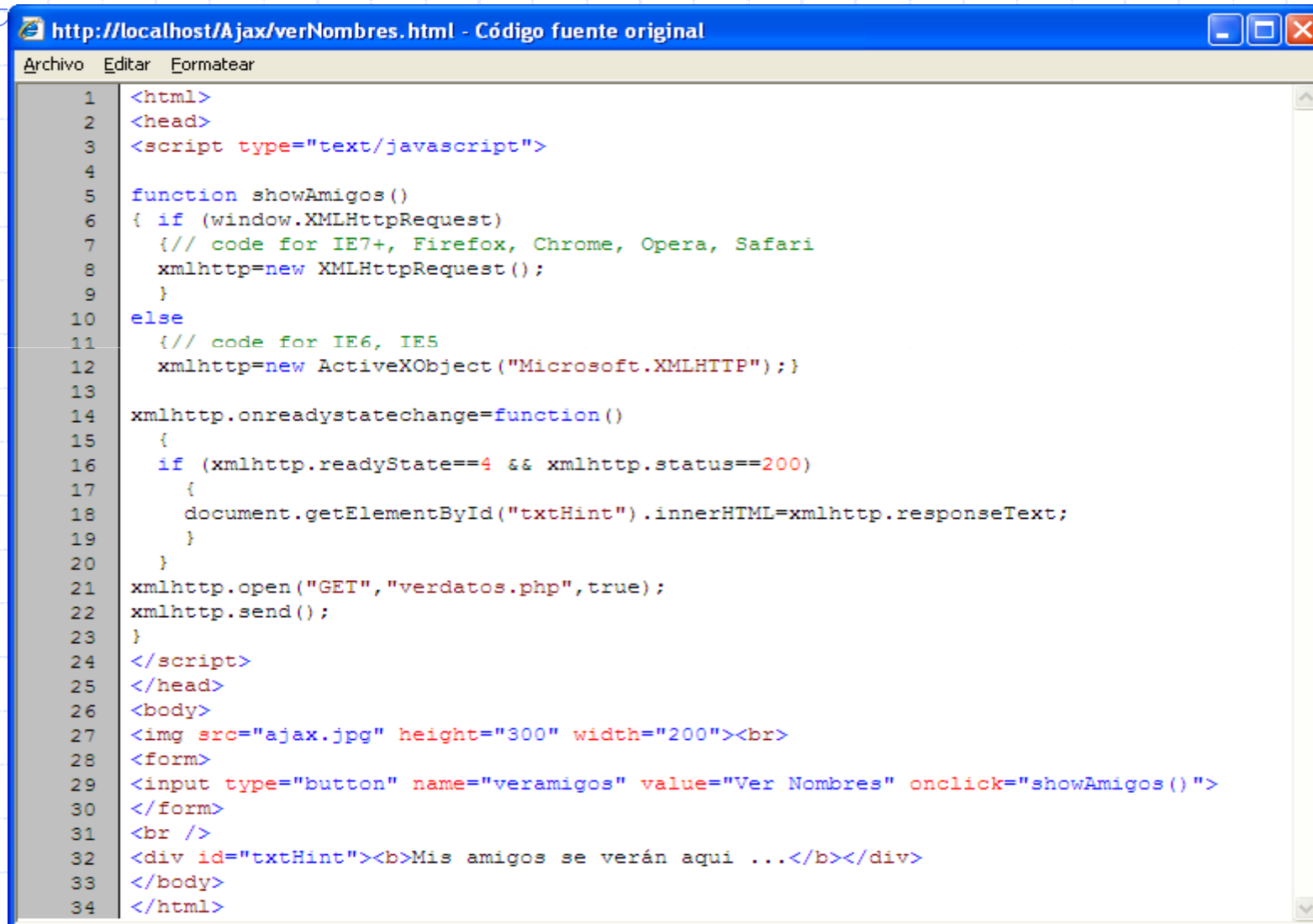
```
<body>
<H1>Mostrando datos con AJAX</H1>
<form>
<input type = "button" value = "Mostrar Pelicula" onclick
    = "pedirDatos()">
</form>
<div id="resultado" style="background-color:#99FF66;">
<p>Aparecera un titulo de pelicula</p>
</div>
</body>
```

Ejemplo 2: Manipular documento XML

```
<script language = "javascript">
XMLHttpRequestObject = new XMLHttpRequest();
XMLHttpRequestObject.onreadystatechange = function()
{
    if (XMLHttpRequestObject.readyState==4)
    {alert (XMLHttpRequestObject.responseText); //visualizar el documento xml como string
    var obj = document.getElementById('resultado');
    var respuesta=XMLHttpRequestObject.responseXML;

    obj.innerHTML=respuesta.getElementsByTagName('titulo')[0].childNodes[0].nodeValue;}
}
function pedirDatos()
{
    XMLHttpRequestObject.open("GET","pelis2.xml");
    XMLHttpRequestObject.send(null);
}
</script>
```

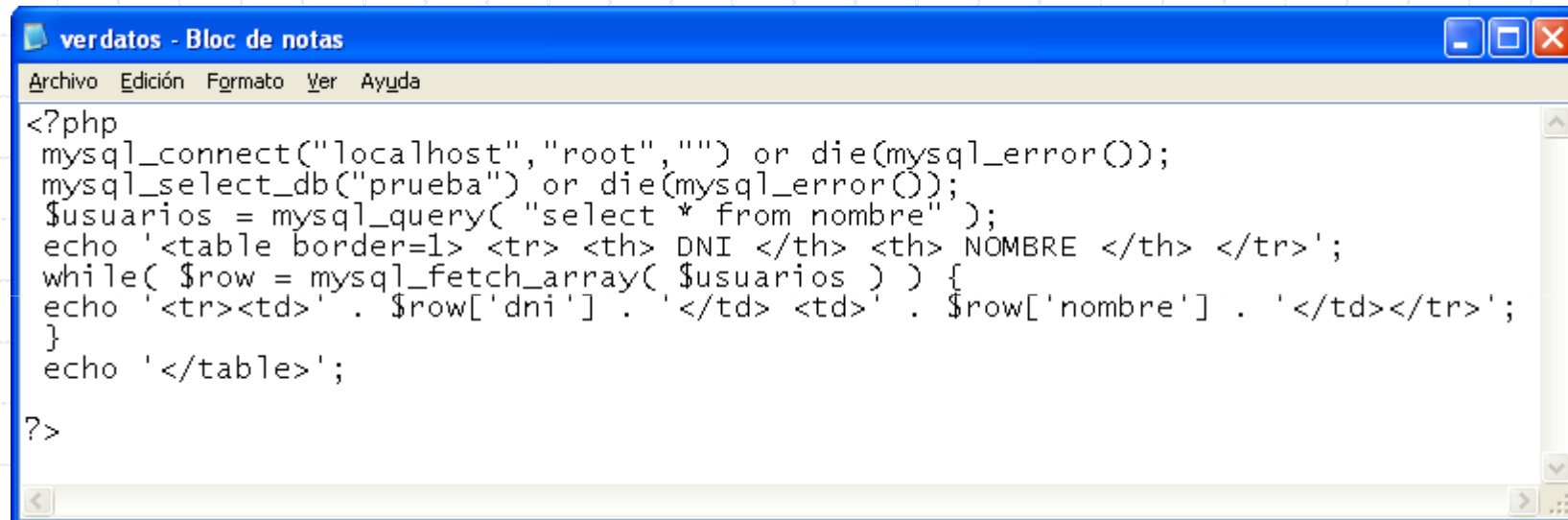
Ejemplo 3: Llamar a un programa PHP que accede a una BD



```
http://localhost/Ajax/verNombres.html - Código fuente original
Archivo  Editar  Formatear

1  <html>
2  <head>
3  <script type="text/javascript">
4
5  function showAmigos()
6  { if (window.XMLHttpRequest)
7    { // code for IE7+, Firefox, Chrome, Opera, Safari
8      xmlhttp=new XMLHttpRequest();
9    }
10   else
11     { // code for IE6, IE5
12       xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");}
13
14   xmlhttp.onreadystatechange=function()
15   {
16     if (xmlhttp.readyState==4 && xmlhttp.status==200)
17     {
18       document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
19     }
20   }
21   xmlhttp.open("GET","verdatos.php",true);
22   xmlhttp.send();
23 }
24 </script>
25 </head>
26 <body>
27 <br>
28 <form>
29 <input type="button" name="veramigos" value="Ver Nombres" onclick="showAmigos()">
30 </form>
31 <br />
32 <div id="txtHint"><b>Mis amigos se verán aqui ...</b></div>
33 </body>
34 </html>
```

Ejemplo 3: Llamar a un programa PHP que accede a una BD



```
<?php
mysql_connect("localhost","root","") or die(mysql_error());
mysql_select_db("prueba") or die(mysql_error());
$usuarios = mysql_query( "select * from nombre" );
echo '<table border=1> <tr> <th> DNI </th> <th> NOMBRE </th> </tr>';
while( $row = mysql_fetch_array( $usuarios ) ) {
echo '<tr><td>' . $row['dni'] . '</td> <td>' . $row['nombre'] . '</td></tr>';
}
echo '</table>';

?>
```

Ejemplo 4: Llamar a un programa PHP con paso de parámetros (GET)

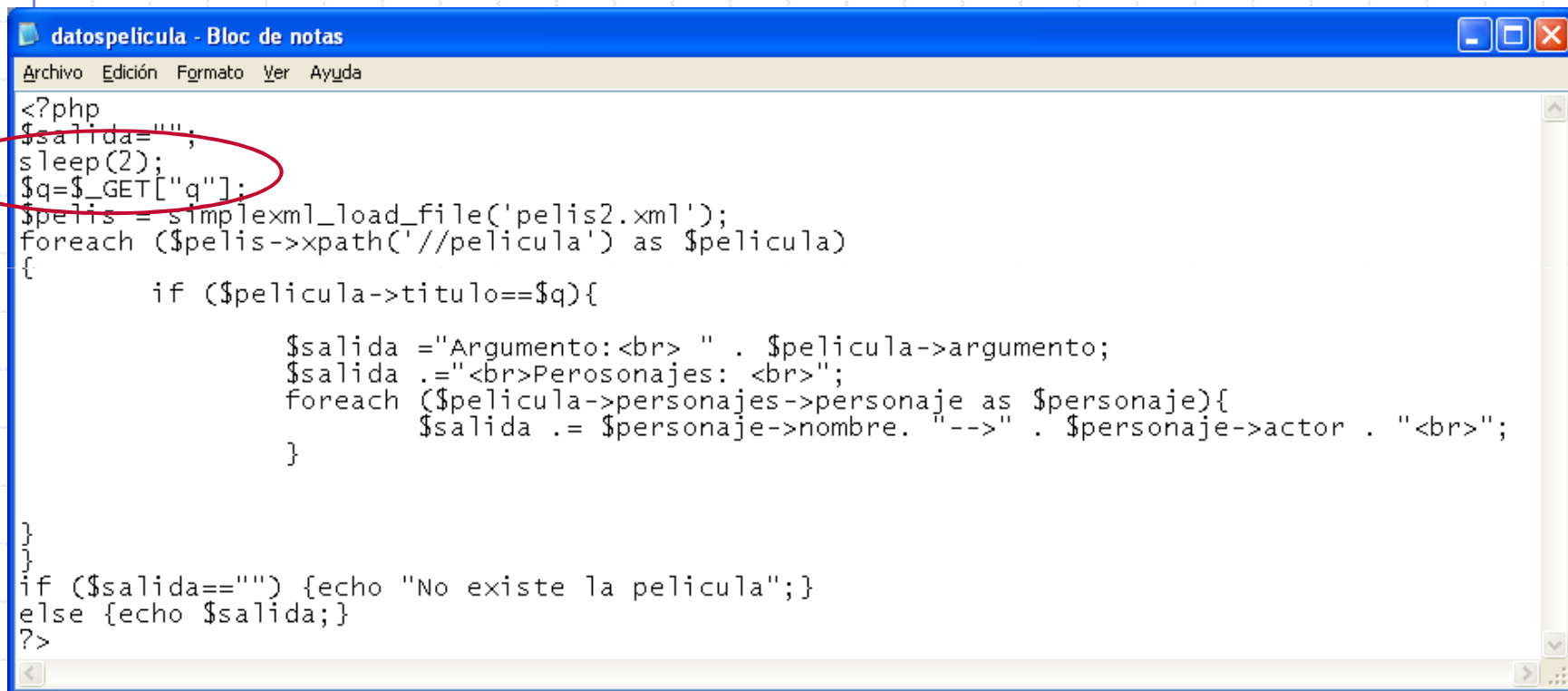
```
<body>
<br>
<form>
Seleccionar un título:
<select name="pelis" onchange="showPeli(this.value)">
<option value="">.....</option>
<option value="La vida de los otros">La vida de los otros</option>
<option value="Milenium I">Milenium I</option>
</select>
</form>
<div id="txtHint"><b>Los datos de la pelicula se mostrarán aquí ...</b></div>
</body>
```


Ejemplo 4: Llamar a un programa PHP con paso de parámetros (GET)

```
<script>
xmlhttp = new XMLHttpRequest();
xmlhttp.onreadystatechange=function()
{
    if (xmlhttp.readyState==4 && xmlhttp.status==200)
        {document.getElementById("txtHint").innerHTML=xmlhttp.responseText; }
}
function showPeli(str)
{
    if(str=="")
    {
        document.getElementById("txtHint").innerHTML="";
        return;
    }
    xmlhttp.open("GET","datos pelicula.php?q="+str,true);
    xmlhttp.send();
}
</script>
```

AJAX

Ejemplo 4: Llamar a un programa PHP con paso de parámetros (GET)



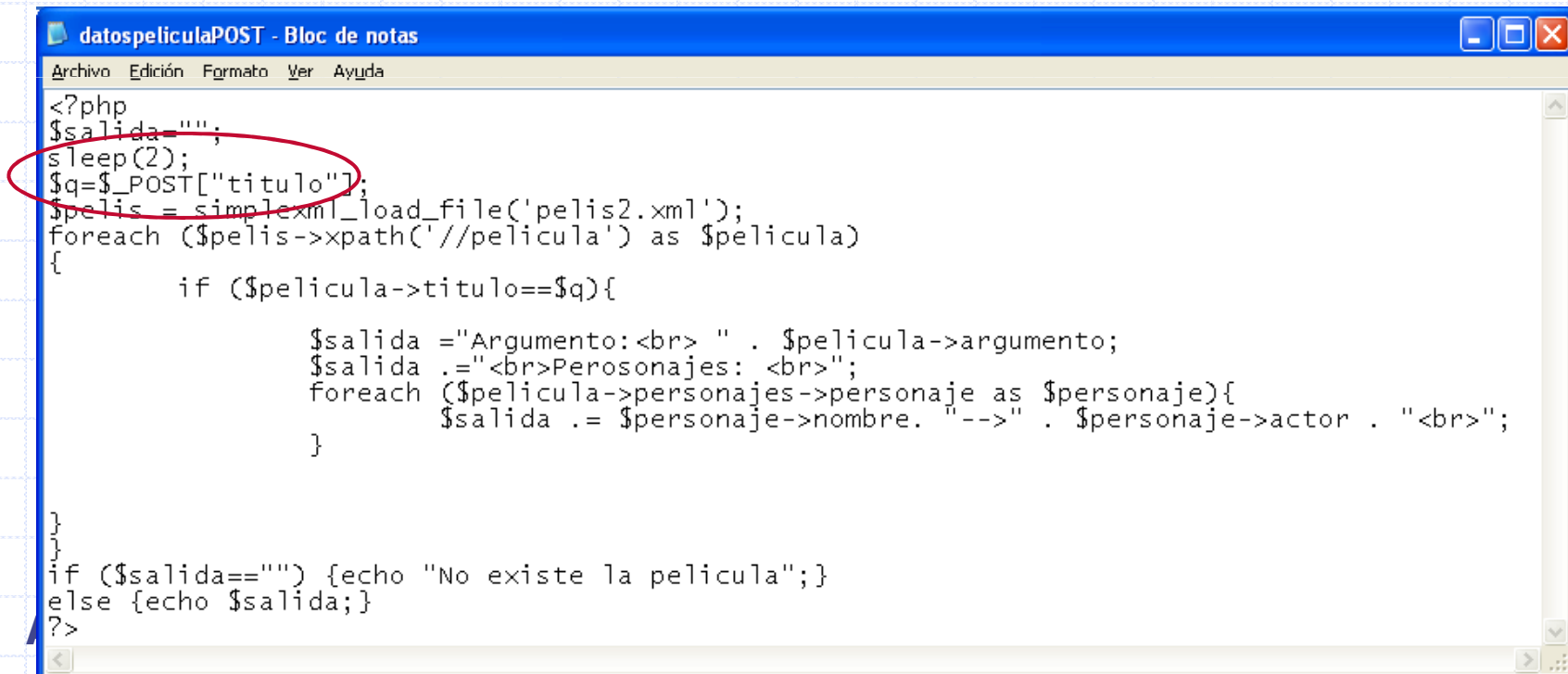
```
<?php
$salida="";
sleep(2);
$q=$_GET["q"];
$pelis = simplexml_load_file('pelis2.xml');
foreach ($pelis->xpath('//pelicula') as $pelicula)
{
    if ($pelicula->titulo==$q){
        $salida ="Argumento:<br> " . $pelicula->argumento;
        $salida .="<br>Personajes: <br>";
        foreach ($pelicula->personajes->personaje as $personaje){
            $salida .= $personaje->nombre. "-->" . $personaje->actor . "<br>";
        }
    }
}
if ($salida=="") {echo "No existe la pelicula";}
else {echo $salida;}
?>
```

Ejemplo 5: Llamar a un programa PHP con paso de parámetros (POST)

```
function showPeli(str)
```

```
.....
```

```
xmlhttp.open("POST","datospeliculaPOST.php",true);  
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");  
xmlhttp.send("titulo="+str);  
}
```



```
datospeliculaPOST - Bloc de notas  
Archivo Edición Formato Ver Ayuda  
<?php  
$salida="";  
sleep(2);  
$q=$_POST["titulo"];  
$pelis = simplexml_load_file('pelis2.xml');  
foreach ($pelis->xpath('//pelicula') as $pelicula)  
{  
    if ($pelicula->titulo==$q){  
        $salida ="Argumento:<br> " . $pelicula->argumento;  
        $salida .="<br>Personajes: <br>";  
        foreach ($pelicula->personajes->personaje as $personaje){  
            $salida .= $personaje->nombre. "-->" . $personaje->actor . "<br>";  
        }  
    }  
}  
if ($salida=="") {echo "No existe la pelicula";}   
else {echo $salida;}  
?>
```

Ejemplo 6: Mostrar mensaje mientras se espera la respuesta ...

```
function showPelicula(str)
{
  if (xmlhttp){
    xmlhttp.onreadystatechange=estadoPeticion;
    xmlhttp.open("GET","datos pelicula.php?q="+str,true);
    xmlhttp.send(null);}
}
```

Ejemplo 6: Mostrar mensaje mientras se espera la respuesta ...

```
function estadoPeticion() {  
  switch(xmlhttp.readyState) { //Según el estado de la petición devolvemos un Texto  
    case 0: document.getElementById('estado').innerHTML =  
      "Sin iniciar..."; break;  
    case 1: document.getElementById('estado').innerHTML =  
      "<b>Cargando...</b>"; break;  
    case 2: document.getElementById('estado').innerHTML =  
      "<b>Cargado...</b>"; break;  
    case 3: document.getElementById('estado').innerHTML =  
      "Interactivo..."; break;  
    case 4: document.getElementById('estado').innerHTML =  
      "<b>Completado...</b>";  
      //Si ya hemos completado la petición, devolvemos además la información.  
      document.getElementById('txtHint').innerHTML=xmlhttp.responseText;break;  
  }  
}
```

Ajax con JQuery-web oficial

Algunas posibilidades (no todas)

Fuente: <http://net.tutsplus.com/tutorials/javascript-ajax/5-ways-to-make-ajax-calls-with-jquery/>

jQuery function	API	GET	POST
load()	load(url, data, callback)	YES	YES
\$.getJSON()	\$.getJSON(url, data, callback)	YES	NO
\$.getScript()	\$.getScript(url, callback)	NO	NO
\$.get()	\$.get(url, data, callback, type)	YES	NO
\$.post()	\$.post(url, data, callback, type)	NO	YES
\$.ajax()	\$.ajax(options)	YES	YES

Ajax con JQuery

load(): Load a piece of html into a container DOM.

\$.getJSON() : Load a JSON with GET method.

\$.getScript(): Load a JavaScript.

\$.get(): Use this if you want to make a GET call and play extensively with the response.

\$.post(): Use this if you want to make a POST call and don't want to load the response to some container DOM.

\$.ajax(): Use this if you need to do something when XHR fails, or you need to specify ajax options (*e.g. cache: false*) on the fly.

Fuente: <http://net.tutsplus.com/tutorials/javascript-ajax/5-ways-to-make-ajax-calls-with-jquery/>


```
<title>Ejemplo1</title>
<script src="jquery-1.8.3.min.js" type="text/javascript"></script>
<script language = "javascript">

function pedirDatos()
{
$("#resultado").load("datos.txt");
}
</script>
</head>
<body>
<H1>Mostrará datos con AJAX</H1>
<form> <input type = "button" value = "Mostrar mensaje" onclick =
"pedirDatos()">
</form>
<div id="resultado" style="background-color:#99FF66;">
<p>Aquí aparecerá texto</p>
</div>
</body>
```

Ejemplo .load

Misma
funcionalidad
que el ejemplo 1

```

<html>
<head>
<script src="jquery-1.8.3.min.js" type="text/javascript"></script>
<script>
function showPeli(str)
{
$('#content').html('<div></div>');
var jqxhr=$.get ("datospelicula.php", {q:str},
function(datos){
    $('#content').fadeIn(1000).html(datos);
});
jqxhr.fail(function(){
    $('#content').fadeIn().html('<p class="error"><strong>
    El servidor parece que no responde</p>');
})
}
</script>
</head>
<body>
<br>
<form>
Seleccionar un título:
<select name="pelis" onchange="showPeli(this.value)">
<option value="">.....</option>
<option value="La vida de los otros">La vida de los otros</option>
<option value="Milenium I">Milenium I</option>
</select>
</form>
<div id="content"><b>Los datos de la pelicula se mostrarán aquí ...</b></div>
</body>
</html>

```

Ejemplo .get

Funcionalidad similar a la del ejemplo 6 pero usando jQuery

```
<html>
<head>
<script src="jquery-1.8.3.min.js" type="text/javascript"></script>
<script>
function ejemploMetodoAjax(str){
$.ajax({
  url: 'datos pelicula.php?q=' + str,
  beforeSend: function() { $('#content').html('<div></div>') },
  success: function(datos) {
    $('#content').fadeIn().html(datos);
  },
  error: function() {
    $('#content').fadeIn().html('<p class="error"><strong>El servidor parece que no responde</p>');
  }
});
}
</script>
</head>
<body>
<br>
<form>
  Seleccionar un título:
  <select name="pelis" onchange="showPeli(this.value)">
    <option value="">.....</option>
    <option value="La vida de los otros">La vida de los otros</option>
    <option value="Milenium I">Milenium I</option>
  </select>
</form>
<div id="content"><b>Los datos de la pelicula se mostrarán aquí ...</b></div>
</body>
</html>
```

AJAX

Ejemplo .ajax

Misma funcionalidad que el ejemplo anterior pero usando el método .ajax de jQuery

Uso del método **.getJSON**
para visualizar fotos
obtenidas del portal *flickr*.

```

<html lang="en">
<head> <meta charset="utf-8">
<title>jQuery.getJSON demo</title>
<style> img {height: 100px; float: left; } </style>
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
</head>
<body>
<div id="images"></div>
<script>
(function()
{var flickerAPI =
"http://api.flickr.com/services/feeds/photos_public.gne?jsoncallback=?";
$.getJSON( flickerAPI, {tags: "paisajes", tagmode: "any", format: "json" })
.done(function( data )
{
$.each( data.items, function( i, item ) {
if ( i == 10 ) {return false;}
else
{
alert (item.title);
$( "<img>" ).attr( "src", item.media.m ).appendTo( "#images" );
}
});
});})();
</script>
</body>
</html>
  
```

Los datos que recibe el cliente mediante .getJSON (.....)

```
{
  "title": "Talk On Travel Pool",
  "link": "http://www.flickr.com/groups/talkontravel/pool/",
  "description": "Travel and vacation photos from around the world, featured on the Talk On Travel blog. Join us
    in sharing your favourite destinations, your top hot spots, and the places to avoid. ",
  "modified": "2009-02-02T11:10:27Z",
  "generator": "http://www.flickr.com/",
  "items": [
    {
      "title": "View from the hotel",
      "link": "http://www.flickr.com/photos/33112458@N08/3081564649/in/pool-998875@N22",
      "media": {"m": "http://farm4.static.flickr.com/3037/3081564649_4a6569750c_m.jpg"},
      "date_taken": "2008-12-04T04:43:03-08:00",
      "description": "<p><a href='http://www.flickr.com/people/33112458@N08/'> Talk On Travel</a>
        has added a photo to the pool:</p> <p><a href='http://
        www.flickr.com/photos/33112458@N08/3081564649/' title='View from the hotel'> <img
        src='http://farm4.static.flickr.com/3037/3081564649_4a6569750c_m.jpg' width='240'
        height='180' alt='View from the hotel' /></a></p> ",
      "published": "2008-12-04T12:43:03Z",
      "author": "nobody@flickr.com (Talk On Travel)",
      "author_id": "33112458@N08",
      "tags": "spain dolphins tenerife canaries lagomera aqualand playadelasamericas junglepark losgigantos
        loscristines talkontravel"
    }
  ]
}
```

Un ejemplo para mostrar noticias (RSS)

- <http://kyleschaeffer.com/development/the-perfect-jquery-ajax-request/>

Problemas con la cache ...

- No todos los navegadores se comportan de la misma forma al hacer una petición al servidor
- Es posible modificar su comportamiento solicitando explícitamente que guarde/no guarde la solicitud en la cache

Problemas con la cache ...

- En PHP podemos indicar que no se almacene el resultado:

```
<?php  
header("Cache-Control: no-store, no-cache, must-revalidate");  
... la aplicación PHP ...  
?>
```

Devolverá una cabecera HTTP Response indicando que este resultado no se almacene en la cache.

Problemas con la cache ...

En la parte cliente, con JavaScript, si no se invoca a un .php (por ejemplo queremos traer un .xml) se puede añadir un parámetro ficticio a la petición que siempre tome valores diferentes:

```
XHR.open("GET","contador.xml?q="+ new Date().getTime());
```

Con jQuery, se puede usar el método [\\$.ajax](#). Este método tiene un parámetro *cache* que según su valor (true,false), se almacena el resultado o no en la cache.

```
$.ajax({  
  url : '/ruta/hacia/mi/archivo/archivo.json',  
  dataType : 'json',  
  data : data,  
  cache : false,  
  success : function(data) {  
  }  
});
```

Herramientas para facilitar el desarrollo basado en AJAX

- ◆ Por supuesto, se puede desarrollar programando directamente en JavaScript, pero no es una forma de trabajar “escalable”*, es necesario acudir a herramientas, como librerías, entornos de desarrollo,...

- ◆ <http://developer.yahoo.com/yui/>

- ◆ /connection

- ◆ <http://www.dojotoolkit.org/>

- ◆ <http://www.mochikit.com/>

- ◆ <http://wiki.script.aculo.us/>

- ◆ Y en el mundo de los productos Ms

- ◆ <http://www.asp.net/ajax/>

* “Aumento rápido y por lo general alarmante de algo...”