# 7.- Utilización de mecanismos de comunicación asíncrona (AJAX-AsynchronousJavascript and XML):

a) Mecanismos de comunicación asíncrona.

b) Modificación dinámica del documento utilizando comunicación asíncrona.

c) Formatos para el envío y recepción de información. XML y JSON (Java Script Object Notation).

d) Notificaciones.

e) Librerías de actualización dinámica.

# Cuestiones previas.

Para poder utilizar AJAX en nuestro equipo y poder probar las cosas deberemos realizar unos pasos previos, que son:

1.-) Instalar un servidor Web, Apache, Internet InformationServices.

2.-) Instalar PHP.

Para realizar estas acciones podemos podemos instalar un programa que nos permite instalar un servidor Web local, como son: XAMPP, WAMPServer (windows), LAMP (Linux), MAMP(Mac),...

También podemos instalar manualmente Apache o Internet InformationServices.

## a) Mecanismos de comunicación asíncrona.

AJAX es el acrónimo de Asynchronous JavaScript XML que se puede traducir como "JavaScript asíncrono + XML.

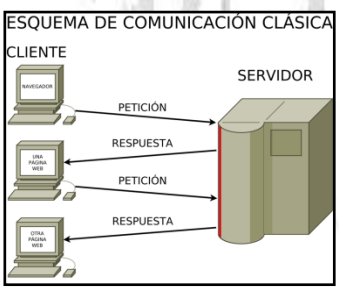
Asíncrono nos indica que cuando se hace una petición, el cliente no se queda a la espera de una respuesta, sino que puede seguir realizando otras cosas.

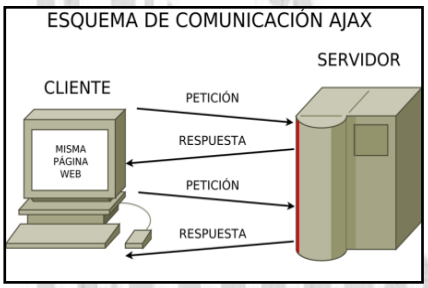
Síncrono nos indica que cuando se hace una petición, el cliente se queda esperando, sin hacer nada, hasta recibir una respuesta,

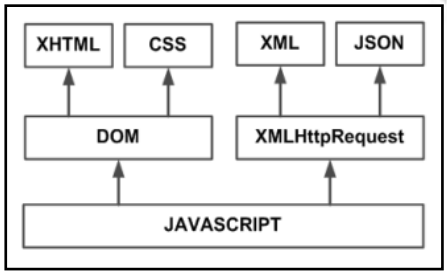
AJAX hace referencia a un conjunto de técnicas que nos permiten enviar y recibir información del servidor desde un documento HTML sin tener que volver a cargarlo.

Tecnologías que forman AJAX:

- XML y CSS para la presentación basada en estándares.
- DOM para la manipulación dinámica.
- XML, XSLT y JSONpara el intercambio y manipulación de información.
- El objeto XMLHttpRequest para el intercambio asíncrono de información.
- JavaScript para unir todos los componentes anteriores.




AJAX nos permite:

- La posibilidad de hacer peticiones al servidor sin tener que volver a cargar la página.
- La posibilidad de analizar y trabajar documentos XML.

Para realizar una comunicación deberemos seguir los siguientes pasos:

1.-) Declarar un objeto **XMLHttpRequest**

**var*nombre-objeto* = new XMLHttpRequest();**

```
001  var peticion_http=newXMLHttpRequest();
```

en algunos casos existe la posibilidad de que los navegadores no soporten el objeto XMLHttpRequest, en cuyo caso deberemos utilizar un objeto ActiveXObject con lo cual deberemos poner

```
001  if(window.XMLHttpRequest){
002      peticion_http=newXMLHttpRequest();
003  }else if(window.ActiveXObject){
004      peticion_http=newActiveXObject("Microsoft.XMLHTTP");
005  }
```

o incluso pueden soportar el objeto ActiveX, que necesitamos, y pondremos:

```
001  if(window.XMLHttpRequest){
002      peticion_http=newXMLHttpRequest();
003  }else if(window.ActiveXObject){
004      try{
005          peticion_http=newActiveXObject("Microsoft.XMLHTTP");
006      }catch(e){
007      }
008  }
```

2.-) Indicar qué función se va a ejecutar cuando se produzca el evento **readystatechange**, añadir oyentes para el evento, para lo cual podemos poner:
***nombre-objeto*.onreadystatechange = *nombre_funcion*;**

```
001  peticion_http.onreadystatechange= mostrar;
```

o también se puede poner
***nombre-objeto*.onreadystatechange = function() { *cuerpo-función* };**

o también se puede poner
***nombre-objeto*.addEventListener("readystatechange",*nombre-función*);**

```
001  if(document.addEventListener)
002      peticion_http.addEventListener("readystatechange",mostrar)
003  else if(document.attachEvent)
004      peticion_http.attachEvent("onreadystatechange",mostrar);
```

3.-) Abrir vía de comunicación con el servidor a través del método **open**.
***nombre-objeto*.open("GET"|"POST","*fichero*" [,*asincrona*])**
con el primer parámetro se indica cómo se envían los datos al servidor, con GET se pasan con el fichero o url y con POST se envían los datos en el cuerpo de la petición. Con asíncrona se indica si la conexión va a ser asíncrona, con el valor true y con el valor false la conexión será síncrona; si no se pone se indica que la conexión es asíncrona. Cuando se solicite un fichero vamos a enviar los datos al servidor mediante get ya que cuando no se envían parámetros vamos a utilizar get.

Ejemplos se solicita que se cargue un fichero de texto, que puede ser texto plano o bien texto html.

```
001 peticion_http.open("GET","http://localhost/holamundo.txt",true);
```

Ejemplo se llama a un programa en php sin ningún parámetro

```
001 peticion_http.open("GET","ajax_00.php",true);
```

Ejemplo se llama a un programa en php y se realiza paso de parámetros mediante get.

```
001 peticion_http.open("GET","ajax.php?nombre=felix&apellido=bayon",true);
```

Ejemplo se llama a un programa en php y el paso de parámetros se va a realizar mediante post, en cuyo caso los parámetros se envían al ejecutar el método send.

```
001 peticion_http.open("POST","ajax.php",true);
```

4.-) Si son necesarios deberemos definir los encabezados, que se utilizan para pasar información al programa del servidor. Para ellos utilizaremos el método **setRequestHeader**.
***nombre-objeto*.setRequestHeader("*nombre*","*valor*");**
Ejemplo cabecera que debemos poner cuando se quieren enviar datos al servidor en formato XML.

```
001 peticion_http.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
```

Ejemplo cabecera que debemos poner cuando se quieren enviar datos al servidor en formato JSON.

```
001 peticion_http.setRequestHeader("Content-Type","application/json");
```

5.-) Enviar petición con el método **send**, si no se envían parámetros se puede dejar sin poner parámetros, aunque se recomienda en este caso pasar null y si deseamos pasar parámetros con POST entonces deberemos poner una cadena del tipo "nombre=valor&nombre-2=valor-2".
***nombre-objeto*.send([*parámetros*|null])**

Ejemplo cuando el paso de parámetros se establece con get

```
001 peticion_http.send(null);
```

Ejemplo cuando el paso de parámetros se establece con post se envían en este punto los parámetros

```
001 peticion_http.send("nombre=felix&apellido=bayon ");
```

6.-) En la función que se encarga de manejar el evento readystatechange tendremos que detectar que se ha recibido la información y luego manipular la misma a través de las propiedades **responseText** o **responseXML** o bien los

métodos **getResponseHeader("encabezado")** o **getAllResponseHeaders()**. Debemos detectar que el **readyState** vale 4 y el **status** vale 200.
Se puede utilizar el objeto o bien evento.target.

Detectamos los valores de estados en las dos propiedades y sacamos un mensaje con el valor devuelto

```
001 function muestra(){
002     if(peticion_http.readyState==4){
003         if(peticion_http.status==200){
004             alert(peticion_http.responseText);
005         }
006     }
007 }
```

o bien detectamos los valores de estados en las dos propiedades y sacamos un mensaje con el valor devuelto

```
001 function muestra(evento){
002     if(evento.target.readyState==4){
003         if(evento.target.status==200){
004             alert(evento.target.responseText);
005         }
006     }
007 }
```

El objeto XMLHttpRequest posee además los siguientes métodos:
**abort()** cancela la petición realiza anteriormente.
**getAllResponseHeaders()** devuelve una cadena con todos los encabezados de la respuesta.
**getResponseHeader("encabezado")** devuelve una cadena con el valor del encabezado indicado.

Ejemplo 1: Vamos a cargar el contenido de un fichero (segovia.txt), que contiene código html en un párrafo (div). En lugar de llamarse segovia.txt también podría llamarse segovia.html

*ajax-01-2.js*

```
001 window.onload=descargaArchivo;
002 var peticion_http;
003 function descargaArchivo(){
004     if(window.XMLHttpRequest){
005         peticion_http=newXMLHttpRequest();
006     }else if(window.ActiveXObject){
007         peticion_http=newActiveXObject("Microsoft.XMLHTTP");
008     }
009     peticion_http.onreadystatechange=muestraContenido;
010     peticion_http.open('GET','segovia.txt',true);
011    peticion_http.send(null);
012 }
013 function muestraContenido(){
014     if(peticion_http.readyState==4){
015         if(peticion_http.status==200){
016             var ele_div=document.getElementById("primero");
017             ele_div.innerHTML=peticion_http.responseText;
018         }
019     }
020 }
```

*ajax-01-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
```

```
003     <head>
004         <title>Hola Mundo con AJAX</title>
005         <meta charset="utf-8"/>
006         <meta name="author" value="Félix Ángel Muñoz Bayón"/>
007         <script src="ja/ajax-01-2.js" type="text/javascript"></script>
008         <script type="text/javascript">
009         </script>
010     </head>
011     <body>
012         <nav>
013         </nav>
014         <header>
015         </header>
016         <main>
017             <section>
018                 <article>
019                     <div>
020                         <div id="primero">
021                         </div>
022                     </div>
023                 </article>
024             </section>
025         </main>
026         <footer>
027         </footer>
028         <aside>
029         </aside>
030     </body>
031 </html>
```

Ejemplo 2: vamos a cargar un fichero con contenido html. Cuando pulsemos en los distintos hiperenlaces vamos a cargar diferentes ficheros.

*ajax-02-3.js*

```
001 var peticion_http;
002 if (document.addEventListener)
003     window.addEventListener("load",iniciar)
004 else if (document.attachEvent)
005     window.attachEvent("onload",iniciar);
006 function iniciar(){
007     descargaArchivo("holamundo.txt");
008 }
009 function descargaArchivo(fichero){
010     if(window.XMLHttpRequest){
011         peticion_http=newXMLHttpRequest();
012     }else if(window.ActiveXObject){
013         peticion_http=newActiveXObject("Microsoft.XMLHTTP");
014     }
015     if (document.addEventListener)
016         peticion_http.addEventListener("readystatechange",muestraContenido)
017     else if (document.attachEvent)
018         peticion_http.attachEvent("onreadystatechange",muestraContenido);
019     peticion_http.open('GET',fichero,true);
020     peticion_http.send(null);
021 }
022 function muestraContenido(){
023     if(peticion_http.readyState==4){
024         if(peticion_http.status==200){
025             var ele_div=document.getElementById("primero");
026             ele_div.innerHTML=peticion_http.responseText;
027         }
028     }
029 }
```

*ajax-02-3.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
```

```
003     <head>
004         <meta charset="utf-8"/>
005         <title>Hola Mundo con AJAX</title>
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <script src="js/ajax-02-3.js" type="text/javascript"></script>
008         <script type="text/javascript">
009         </script>
010     </head>
011     <body>
012         <nav>
013         </nav>
014         <header>
015         </header>
016         <main>
017             <section>
018                 <article>
019                     <div>
020                         <a href="javascript:descargaArchivo('holamundo.txt');">
Hola Mundo</a><br/>
021                         <a href="javascript:descargaArchivo('segovia.txt');">
Segovia</a><br/>
022                         <a href="javascript:descargaArchivo('madrid.txt');">
Madrid</a><br/>
023                         <div id="primero">
024                         </div>
025                     </div>
026                 </article>
027             </section>
028         </main>
029         <footer>
030         </footer>
031         <aside>
032         </aside>
033     </body>
034 </html>
```

Ejemplo 3: petición a un programa en php y envío de parámetros mediante get

*ajax-03-2.js*

```
001 var peticion_http;
002 var alumno;
003 var materia;
004 var calificacion;
005 document.addEventListener('readystatechange',inicializar,false);
006 function inicializar(){
007     if (document.readyState=='complete'){
008         alumno=document.getElementById('alumno');
009         materia=document.getElementById('materia');
010         calificacion=document.getElementById('calificacion');
011         if (document.addEventListener){
012             alumno.addEventListener('change',enviarPeticionAJAX);
013             materia.addEventListener('change',enviarPeticionAJAX);
014         } else if (document.attachEvent){
015             alumno.attachEvent('onchange',enviarPeticionAJAX);
016             materia.attachEvent('onchange',enviarPeticionAJAX);
017         }
018
019     }
020 }
021 function enviarPeticionAJAX(evento){
022     if( alumno.value!='' && materia.value!=''){
023         alumno.disabled=true;
024         materia.disabled=true;
025         if (window.XMLHttpRequest){
026             peticion_http=new XMLHttpRequest();
027         }else if(window.ActiveXObject){
028             peticion_http=new ActiveXObject("Microsoft.XMLHTTP");
029         }
030         if (document.addEventListener)
031             peticion_http.addEventListener('readystatechange',gestionarRespuesta)
032         else if (document.attachEvent)
```

```
033             peticion_http.attachEvent('onreadystatechange',gestionarRespuesta);
034         peticion_http.open('GET','ajax-03.php?alumno='+alumno.value+'&materia='+materia.value,true);
035         peticion_http.send(null);
036     }
037 }
038 function gestionarRespuesta(evento){
039     if (evento.target.readyState==4 && evento.target.status==200){
040         alumno.disabled=false;
041         materia.disabled=false;
042         calificacion.value=evento.target.responseText;
043     }
044 }
```

*ajax-03-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <meta charset="utf-8"/>
005         <title>AJAX</title>
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script type="text/javascript" src="js/ajax-03-2.js">
010         </script>
011     </head>
012     <body>
013         <nav>
014         </nav>
015         <header>
016         </header>
017         <main>
018             <section>
019                 <article>
020                     <div>
021                         <label for='alumno'>Alumno: </label>
022                         <select id='alumno'>
023                             <option value='' selected='selected'>--Elija un alumno--</option>
024                             <option>Juan Mateos</option>
025                             <option>Ana Irene Palma</option>
026                         </select>
027                         <label for='materia'>Materia: </label>
028                         <select id='materia'>
029                             <option value='' selected='selected'>--Elija una materia--</option>
030                             <option>Lenguaje</option>
031                             <option>Sociales</option>
032                         </select>
033                         <label for='calificacion'>Calificaci&oacute;n: </label>
034                         <input type='text' readonly='readonly' id='calificacion'/>
035                     </div>
036                 </article>
037             </section>
038         </main>
039         <footer>
040         </footer>
041         <aside>
042         </aside>
043     </body>
044 </html>
```

*ajax-03.php*

```
001 <?php
002     $alumno=$_REQUEST['alumno'];
003     $materia=$_REQUEST['materia'];
004     switch($alumno){
005         case 'Juan Mateos':
006             switch($materia){
007                 case 'Sociales':
008                     echo '7.5';
009                     break;
010                 case 'Lenguaje':
```

```
                        echo '9.5';
                        break;
                }
                break;
            case 'Ana Irene Palma':
                switch($materia){
                    case 'Sociales':
                        echo '8.5';
                        break;
                    case 'Lenguaje':
                        echo '7.5';
                        break;
                }
            break;
    }
?>
```

Ejemplo 4: petición a un programa en php y envío de parámetros mediante post

*ajax-04-2.js*

```
var peticion_http;
var alumno;
var materia;
var calificacion;
if (document.addEventListener)
    document.addEventListener('readystatechange',inicializar)
else if (document.attachEvent)
    document.attachEvent('onreadystatechange',inicializar);
function inicializar(){
    if (document.readyState=='complete'){
        alumno=document.getElementById('alumno');
        materia=document.getElementById('materia');
        calificacion=document.getElementById('calificacion');
        if (document.addEventListener){
            alumno.addEventListener('change',enviarPeticionAJAX);
            materia.addEventListener('change',enviarPeticionAJAX);
        } else if (document.attachEvent){
            alumno.attachEvent('onchange',enviarPeticionAJAX);
            materia.attachEvent('onchange',enviarPeticionAJAX);
        }
    }
}
function enviarPeticionAJAX(evento){
    if(alumno.value!='' && materia.value!=''){
        alumno.disabled=true;
        materia.disabled=true;
        if (window.XMLHttpRequest){
            peticion_http=new XMLHttpRequest();
        }else if(window.ActiveXObject){
            peticion_http=new ActiveXObject("Microsoft.XMLHTTP");
        }
        if (document.addEventListener)
            peticion_http.addEventListener('readystatechange',gestionarRespuesta)
        else if (document.attachEvent)
            peticion_http.attachEvent('onreadystatechange',gestionarRespuesta);

        peticion_http.open('POST','ajax-04.php',true);
        peticion_http.setRequestHeader('Content-Type','application/x-www-form-
urlencoded');
        peticion_http.send('alumno='+alumno.value+'&materia='+materia.value);
    }
}
function gestionarRespuesta(evento){
    if ( evento.target.readyState==4 && evento.target.status==200){
        alumno.disabled=false;
        materia.disabled=false;
        calificacion.value=evento.target.responseText;
    }
}
```

*ajax-04-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <meta charset="utf-8"/>
005         <title>AJAX</title>
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script type="text/javascript" src="js/ajax-04-2.js">
010         </script>
011     </head>
012     <body>
013         <nav>
014         </nav>
015         <header>
016         </header>
017         <main>
018             <section>
019                 <article>
020                     <div>
021                         <label for='alumno'>Alumno: </label>
022                         <select id='alumno'>
023                         <option value='' selected='selected'>--Elija un alumno--
</option>
024                         <option>Juan Mateos</option>
025                         <option>Ana Irene Palma</option>
026                         </select>
027                         <label for='materia'>Materia: </label>
028                         <select id='materia'>
029                         <option value='' selected='selected'>--Elija una materia--
</option>
030                         <option>Lenguaje</option>
031                         <option>Sociales</option>
032                         </select>
033                         <label for='calificacion'>Calificaci&oacute;n: </label>
034                         <input type='text' readonly='readonly' id='calificacion'/>
035                     </div>
036                 </article>
037             </section>
038         </main>
039         <footer>
040         </footer>
041         <aside>
042         </aside>
043     </body>
044 </html>
```

*ajax-04.php*

```
001 <?php
002     $alumno=$_REQUEST['alumno'];
003     $materia=$_REQUEST['materia'];
004     $nota=0:
005     switch($alumno){
006         case 'Juan Mateos':
007             switch($materia){
008                 case 'Sociales':
009                     $nota='7.5';
010                     break;
011                 case 'Lenguaje':
012                     $nota='9.5';
013                     break;
014             }
015             break;
016         case 'Ana Irene Palma':
017             switch($materia){
018                 case 'Sociales':
019                     $nota='8.5';
020                     break;
021                 case 'Lenguaje':
022                     $nota='7.5';
023                     break;
024             }
```

```
025             break;
026         }
027         echo $nota;
028     ?>
```

### b) Modificación dinámica del documento utilizando comunicación asíncrona.

Una vez establecida la comunicación vamos a tener en responseText o bien en responseXML la información devuelta por el servidor y en consecuencia a continuación podemos utilizar el DOM o bien jQuery para modificar el contenido de la página web. Las dos propiedades as podemos aplicar al objeto XMLHttpRequest o bien a la propiedad target del evento.

Podemos obtener el elemento que tiene un determinado identificador con
**var *nombre-variable*=document.getElementById("*identificador*")**
**let *nombre-variable*=document.getElementById("*identificador*")**

```
001 var ele_div=document.getElementById("primero");
```

y a continuación a la propiedad textContent, innerHTML o valuela podemos asignar el valor de responseText.

***nombre-variable*.textContent=*nombre-objeto*.responseText;**
***nombre-variable*.innerHTML=*nombre-objeto*.responseText;**
***nombre-variable*.value=*nombre-objeto*.responseText;**

```
001 ele_div.innerHTML=peticion_http.responseText;
```

```
001 calificacion.value=evento.target.responseText;
```

También se podría hacer todo en un único paso
**document.getElementById("*identificador*").innerHTML=*nombre-objeto*.responseText;**

```
001 document.getElementById("primero").peticion_http.responseText;
```

Si lo hacemos con jQuery deberemos acceder al elemento a través de
**$("*#identificador*").html(*responseText*)**

```
001 $("primero").html(peticion_http.responseText);
```

**$("*#identificador*").val(*responseText*)**

```
001 $("nota").val(peticion_http.responseText);
```

Cuando se desean enviar datos al servidor también podemos utilizar un objeto FormData.

El objeto FormData esta diseñado para enviar conjuntos de claves valores a través de AJAX, utilizando un objeto XMLHttpRequest. Esta pensado para enviar datos de un formulario o cualquier dato que se introduzca mediante el teclado.

Para utilizar un objeto FormData vamos a utilizar
**let *nombre-variable*= new FormData()**;
Que nos permite crear un objeto FornData vacío.

```
001  let datos=newFormData();
```

Otra posibilidad que tenemos es

**let *nombre-variable*= new FormData(*variable-formular*io);**

Que nos crea el objeto FormData con los datos del formulario. En la definición del formulario en HTML deberemos poner el atributo **enctype="multipart/form-data"**.

```
001  letmiformulario=document.getElementById("formulario");
002  let datos=newFormData(miformulario);
```

Sobre el objeto FormData podemos aplicar los siguientes métodos

**append(*clave*,*valor*)**

Nos permite esa clave y ese valor al objeto; el valor puede ser el nombre de un fichero o de un bloq.

```
001  datos.append(valorClave,valorValor);
```

**append(*clave*,*valores, nombre-fichero*)**

Nos permite añadir un fichero o bloq al objeto, en donde clave es el nombre de la variable del objeto, valores son los valores del fichero y nombre del fichero es el nombre del fichero que se va a enviar.

**get(*clave*)**

Nos permite recuperar el valor asociado a esa clave.

```
001  letvalorValor=datos.get(valorClave);
```

**has(*clave*)**

Nos devuelve un valor lógico, que nos indica si existe esa clave en el objeto.

```
001  if(datos.has(valorClave)){
```

**getAll(clave)**

Nos devuelve un array con todos los valores de esa clave, si esa clave no existe devuelve un alista vacía.

```
001  todos=datos.getAll(valorClave);
```

**set(*clave*,*valor*)**

Nos permite modificar un valor asociado a una clave, si existe y si no existe añade esa dupla clave valor al objeto.

```
001  datos.set(valorClave,valorValor);
```

**set(*clave*,*valores, nombre-fichero*)**

Nos permite añadir/modificar un fichero o bloq al objeto, en donde clave es el nombre de la variable del objeto, valores son los valores del fichero y nombre del fichero es el nombre del fichero que se va a enviar.

**delete(*clave*)**

Nos permite borrar del objeto el elemento con esa clave

```
001  datos.delete(valorClave);
```

**keys()**

Nos devuelve un iterator que contiene todas las claves del objeto.

```
001  let clave=datos.keys();
```

**values()**

Nos devuelve un iterator que contiene todos los valores del objeto.

```
001  let todo=datos.values();
```

Objetos **iterator** tiene un conjunto de valores

Va a tener un método

**next()**

Que nos permite avanzar en el conjunto de objetos. Este método nos devuelve un objeto con dos propiedades que son:

```
001  let uno=todo.next();
```

- **done**

Devuelve un valor lógico que nos indica si estamos más allá del final. con true estamos más allá del final y con false no.

```
001  while(!uno.done){
```

- **value**

Es el valor del objeto, si done tiene el valor true, esta propiedad tendrá el valor undefined.

```
001  valores.textContent+=uno.value+"\n";
```

Podemos manejar un objeto FormData de forma independiente, como en el siguiente ejemplo

*formdata02.js*

```
001  if (document.addEventListener)
002      window.addEventListener("load",inicio)
003  else if (document.attachEvent)
004      window.attachEvent("onload",inicio);
005  function inicio(){
006      let botonAniadir=document.getElementById("aniadir");
007      let botonLeer=document.getElementById("leer");
008      let botonModificar=document.getElementById("modificar");
009      let botonTodos=document.getElementById("leertodos");
010      let botonBorrar=document.getElementById("borrar");
011      if (document.addEventListener){
012          botonAniadir.addEventListener("click",funcionAniadir);
013          botonLeer.addEventListener("click",funcionLeer);
014          botonModificar.addEventListener("click",funcionModi);
015          botonTodos.addEventListener("click",funcionTodos);
016          botonBorrar.addEventListener("click",funcionBorrar);
017      } else if (document.attachEvent){
018          botonAniadir.attachEvent("onclick",funcionAniadir);
019          botonLeer.attachEvent("onclick",funcionLeer);
020          botonModificar.attachEvent("onclick",funcionModi);
021          botonTodos.attachEvent("onclick",funcionTodos);
022          botonBorrar.attachEvent("onclick",funcionBorrar);
023      }
```

```
024 }
025 var datos=new FormData();
026 function funcionAniadir(){
027     let valorClave=document.getElementById("clave").value.trim();
028     let valorValor=document.getElementById("valor").value.trim();
029     if (valorClave!="" && valorValor!=""){
030         datos.append(valorClave,valorValor);
031         document.getElementById("resultado").textContent="Valores añadidos";
032     } else
033         document.getElementById("resultado").textContent="Clave o valor vacíos";
034 }
035 function funcionLeer(){
036     let valorClave=document.getElementById("clave").value.trim();
037     if (valorClave!="" )
038         if (datos.has(valorClave)){
039             let valorValor=datos.get(valorClave);
040             document.getElementById("valor").value=valorValor;
041             document.getElementById("resultado").textContent="Valor leido";
042         } else
043             document.getElementById("resultado").textContent="No existe un valor
    con esa clave"
044     else
045         document.getElementById("resultado").textContent="Clave vacía";
046 }
047 function funcionModi(){
048     let valorClave=document.getElementById("clave").value.trim();
049     let valorValor=document.getElementById("valor").value.trim();
050     if (valorClave!="" && valorValor!=""){
051         datos.set(valorClave,valorValor);
052         document.getElementById("resultado").textContent="Valor
    modificado/añadido";
053     }
054 }
055 function funcionTodos(){
056     let valorClave=document.getElementById("clave").value.trim();
057     let todos="";
058     let valores;
059     if (valorClave!="" ){
060         todos=datos.getAll(valorClave);
061         valores=document.getElementById("resultado");
062         valores.textContent="";
063         for (let i=0;i<todos.length;i++)
064             valores.textContent+=todos[i]+"\n";
065     } else {
066         let todo=datos.values();
067         valores=document.getElementById("resultado");
068         valores.textContent="";
069         let uno=todo.next();
070         while (!uno.done){
071             valores.textContent+=uno.value+"\n";
072             uno=todo.next();
073         }
074     }
075 }
076 function funcionBorrar(){
077     let valorClave=document.getElementById("clave").value.trim();
078     if (valorClave!="" )
079         if (datos.has(valorClave)){
080             datos.delete(valorClave);
081             document.getElementById("resultado").textContent="Valor borrado";
082         } else
083             document.getElementById("resultado").textContent="No existe un valor
    con esa clave"
084     else
085         document.getElementById("resultado").textContent="Clave vacía";
086 }
```

*formdata02.html*

```
001 <!doctype html>
002 <html lang="es">
003     <head>
```

```
004         <title>FormData 02</title>
005         <meta charset="utf-8" />
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <link href="css/formdata01.css" rel="stylesheet" type="text/css" />
008         <script src="js/formdata02.js" type="text/javascript"></script>
009         <script type="text/javascript">
010
011         </script>
012         <style type="text/css">
013
014         </style>
015     </head>
016     <body>
017         <header>
018
019         </header>
020         <nav>
021
022         </nav>
023         <main>
024             <section>
025                 <article>
026                     <div>
027                         <form name="formulario" id="formu">
028                             <label for="clave">Clave </label>
029                             <input type="text" name="clave" id="clave" /> <br />
030                             <label for="valor">valor </label>
031                             <input type="text" name="valor" id="valor" /> <br />
032                             <label for="resultado">Resultado </label>
033                             <textarea name="resultado" id="resultado" >
034                             </textarea> <br />
035                             <input type="button" name="aniadir" id="aniadir"
    value="Añadir"/> <br />
036                             <input type="button" name="leer" id="leer"
    value="Leer"/> <br />
037                             <input type="button" name="modificar" id="modificar"
    value="modificar"/> <br />
038                             <input type="button" name="leertodos" id="leertodos"
    value="Leer Todos"/> <br />
039                             <input type="button" name="borrar" id="borrar"
    value="Borrar"/> <br />
040                         </form>
041                     </div>
042                 </article>
043             </section>
044         </main>
045         <footer>
046
047         </footer>
048         <aside>
049         </aside>
050     </body>
051 </html>
```

Cuando se utiliza el objeto FormData para enviar datos en una conexión asíncrona lo vamos a enviar mediante post.

Ejemplo de envío de datos a través de un objeto FormData usando ajax.

*formdata/ajax01.js*

```
001 if (document.addEventListener)
002     window.addEventListener("load",inicio)
003 else if (document.attachEvent)
004     window.attachEvent("onload",inicio);
005
006 function inicio(){
007     let boton=document.getElementById("poner");
008     if (document.addEventListener)
009         boton.addEventListener("click",enviar)
010     else if (document.attachEvent)
011         boton.attachEvent("onclick",enviar);
012 }
```

```
var solicitud;
function  enviar(){
    if (window.XMLHttpRequest)
        solicitud=new XMLHttpRequest()
    else if (window.ActiveXObject)
        solicitud=new ActiveXObject("Microsoft.XMLHTTP");
    let nom=document.getElementById("nombre").value;
    let ape=document.getElementById("apellidos").value;
    let datos= new FormData();
    datos.append("nombre",nom);
    datos.append("apellidos",ape);
    if (document.addEventListener)
        solicitud.addEventListener("readystatechange",muestra)
    else if (document.attachEvent)
        solicitud.attachEvent("onreadystatechange",muestra);
    solicitud.open("POST","php/procesar.php");
    solicitud.send(datos);
}
function muestra(){
    if (solicitud.readyState==4)
        if (solicitud.status==200)
            document.getElementById("resultado").value=solicitud.responseText;
}
```

*formdata/ajax01.html*

```
<!doctype html>
<html lang="es">
    <head>
        <title>ajax 01 </title>
        <meta charset="utf-8" />
        <meta name="author" content="Félix Ángel Muñoz Bayón" />
        <script src="js/ajax01.js" type="text/javascript"></script>
        <style type="text/css">

        </style>
    </head>
    <body>
        <header>
        </header>
        <nav>
        </nav>
        <main>
            <section>
                <article>
                    <div>
                        <form name="formulario" id="formulario">
                            <label for="nombre">Nombre</label>
                            <input type="text" id="nombre" name="nombre" /><br />
                            <label for="apellidos">Apellidos</label>
                            <input type="text" id="apellidos" name="apellidos" /><br />
                            <input type="button" id="poner" name="poner" value="Añadir" /><br />
                            <label for="resultado">Apellidos</label>
                            <input type="text" id="resultado" name="resultado" /><br />
                        </form>
                    </div>
                </article>
            </section>
        </main>
        <footer>
        </footer>
        <aside>
        </aside>
    </body>
</html>
```

El programa en PHP va a recibir esos datos como si fuesen uns datros normales pasados, les vamos a recibir con $_POST o $_REQUEST.

*formdata/proccesar.php*

```
<?php
    $nombre=$_POST["nombre"];
    $apellidos=$_POST["apellidos"];
    $total=$nombre . " " . $apellidos;
    echo $total;
?>
```

## c) Formatos para el envío y recepción de información. XML y JSON (Java Script Object Notation).

XML es un lenguaje de etiquetas-

Si queremos enviar los datos de nuestro documento Web a un servidor en formato XML, deberemos crear una cadena cuyo contenido sea el documento XML.

Ejemplo

```
var cadenaXML="<datoalumnos><alumnos>"
cadenaXML+="<alumno>"+alumno.value+"</alumno>"
cadenaXML+="<materia>"+materia.value+"</materia>"
cadenaXML+="</alumnos></datosalumnos>"
```

Deberemos poner una cabecera como la siguiente

```
peticion_http.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
```

Enviaremos los datos mediante POST al servidor.

```
peticion_http.open('POST','ajax_01.php',true);
```

```
peticion_http.send(cadenaXML);
```

Cuando se reciba la respuesta del servidor vamos a utilizar resposeXML y aquí deberemos aplicar el DOM para acceder a los datos que recibimos.

También tenemos la posibilidad de leer un fichero del servidor con datos en XML
Ejemplo de leer un fichero XML, con datos situado en el servidor

*ajax-05-2.js*

```
var peticion_http;
window.onload = descargaArchivo;
function descargaArchivo(){
    // Obtener la instancia del objeto XMLHttpRequest
    if (window.XMLHttpRequest){
        peticion_http =new XMLHttpRequest();
    }else if(window.ActiveXObject){
        peticion_http =new ActiveXObject("Microsoft.XMLHTTP");
    }
    // Preparar la funcion de respuesta
    peticion_http.onreadystatechange = muestraContenido;
    // Realizar peticion HTTP
    peticion_http.open('GET','datos_libros.xml',true);
    peticion_http.send(null);
}
function muestraContenido(){
    if (peticion_http.readyState ==4){
        if (peticion_http.status ==200){
            var todo=peticion_http.responseXML;
            var lineas=todo.getElementsByTagName("Libros");
            var mi_tabla=document.getElementById("tabla");
            var tablita=mi_tabla.getElementsByTagName("tbody").item(0);
```

```
023                 for (var i=0; i< lineas.length;i++){
024                     var titu=lineas.item(i).getElementsByTagName("TITULO");
025                     var auto=lineas.item(i).getElementsByTagName("AUTOR");
026                     var edi=lineas.item(i).getElementsByTagName("EDITORIAL");
027                     var nuevaLinea=document.createElement("tr");
028                     var nuevoDato=document.createElement("td");
029                     var contenido=document.createTextNode(titu.item(0).textContent);
030                     nuevoDato.appendChild(contenido);
031                     var nuevoDauto=document.createElement("td");
032                     var nuevoDedi=document.createElement("td");
033                     var valorDauto=document.createTextNode(auto.item(0).textContent);
034                     var valorDedi=document.createTextNode(edi.item(0).textContent);
035                     nuevoDauto.appendChild(valorDauto);
036                     nuevoDedi.appendChild(valorDedi);
037                     nuevaLinea.appendChild(nuevoDato);
038                     nuevaLinea.appendChild(nuevoDauto);
039                     nuevaLinea.appendChild(nuevoDedi);
040                     tablita.appendChild(nuevaLinea);
041                 }
042             }
043         }
044 }
```

*ajax-05-2.html*

```
001 <!doctype html>
002 <html lang="es">
003     <head>
004         <meta charset="utf-8"/>
005         <title>Leer fichero XML servidor</title>
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script type="text/javascript" src="js/ajax-05-2.js">   </script>
010         <script type="text/javascript">
011         </script>
012     </head>
013     <body>
014         <nav>
015         </nav>
016         <header>
017         </header>
018         <main>
019             <section>
020                 <article>
021                     <div>
022                         <div id="primero">
023                             <table id="tabla">
024                                 <thead>
025                                     <tr>
026                                         <th>Título</th>
027                                         <th>Autor</th>
028                                         <th>Editorial</th>
029                                     </tr>
030                                 </thead>
031                                 <tbody>
032                                 </tbody>
033                             </table>
034                         </div>
035                     </div>
036                 </article>
037             </section>
038         </main>
039         <footer>
040         </footer>
041         <aside>
042         </aside>
043     </body>
044 </html>
```

Fichero xml leido por el programa anterior *datos_libros.xml*

```
<?xmlversion="1.0"standalone="yes"?>
<NewDataSet>
    <Libros>
        <CODIGOLIB>00000001</CODIGOLIB>
        <TITULO>Lenguajes Ensambladores</TITULO>
        <AUTOR>R. Martinez Tomas</AUTOR>
        <EDITORIAL>Paraninfo</EDITORIAL>
        <TEMA>Informatica</TEMA>
        <SUBTEMA>Lenguaje Ensamblador</SUBTEMA>
        <FCH_ALTA>2000-04-25T00:00:00.0000000+02:00</FCH_ALTA>
        <TOTAL_LIBROS>5</TOTAL_LIBROS>
        <DISPONIBLES>3</DISPONIBLES>
        <FCH_PRESTAMO>2004-01-07T00:00:00.0000000+01:00</FCH_PRESTAMO>
        <NUM_PRESTAMOS>19</NUM_PRESTAMOS>
        <TOTAL_MULTAS>0</TOTAL_MULTAS>
        <NUM_MULTAS>0</NUM_MULTAS>
    </Libros>
    <Libros>
        <CODIGOLIB>00000033</CODIGOLIB>
        <TITULO>Por Quien Doblan Las Campanas</TITULO>
        <AUTOR>Ernest Hemingway</AUTOR>
        <EDITORIAL>Seix Barral</EDITORIAL>
        <TEMA>Literatura</TEMA>
        <SUBTEMA>Narrativa</SUBTEMA>
        <FCH_ALTA>2000-07-30T00:00:00.0000000+02:00</FCH_ALTA>
        <TOTAL_LIBROS>8</TOTAL_LIBROS>
        <DISPONIBLES>5</DISPONIBLES>
        <FCH_PRESTAMO>2004-01-07T00:00:00.0000000+01:00</FCH_PRESTAMO>
        <NUM_PRESTAMOS>50</NUM_PRESTAMOS>
        <TOTAL_MULTAS>0</TOTAL_MULTAS>
        <NUM_MULTAS>0</NUM_MULTAS>
    </Libros>
    <Libros>
        <CODIGOLIB>00000192</CODIGOLIB>
        <TITULO>La Casa De Los Espiritus</TITULO>
        <AUTOR>Isabel Allende</AUTOR>
        <EDITORIAL>R.B.A. Editores</EDITORIAL>
        <TEMA>Literatura</TEMA>
        <SUBTEMA>Narrativa</SUBTEMA>
        <FCH_ALTA>2001-11-11T00:00:00.0000000+01:00</FCH_ALTA>
        <TOTAL_LIBROS>3</TOTAL_LIBROS>
        <DISPONIBLES>2</DISPONIBLES>
        <FCH_PRESTAMO>2004-01-21T00:00:00.0000000+01:00</FCH_PRESTAMO>
        <NUM_PRESTAMOS>31</NUM_PRESTAMOS>
        <TOTAL_MULTAS>0</TOTAL_MULTAS>
        <NUM_MULTAS>0</NUM_MULTAS>
    </Libros>
    <Libros>
        <CODIGOLIB>00000350</CODIGOLIB>
        <TITULO>Los Santos Inocentes</TITULO>
        <AUTOR>Miguel Delibes</AUTOR>
        <EDITORIAL>Millenium</EDITORIAL>
        <TEMA>Literatura</TEMA>
        <SUBTEMA>Narrativa</SUBTEMA>
        <FCH_ALTA>2003-01-20T00:00:00.0000000+01:00</FCH_ALTA>
        <TOTAL_LIBROS>8</TOTAL_LIBROS>
        <DISPONIBLES>4</DISPONIBLES>
        <FCH_PRESTAMO>2004-01-16T00:00:00.0000000+01:00</FCH_PRESTAMO>
        <NUM_PRESTAMOS>24</NUM_PRESTAMOS>
        <TOTAL_MULTAS>0</TOTAL_MULTAS>
        <NUM_MULTAS>0</NUM_MULTAS>
    </Libros>
</NewDataSet>
```

## Ejemplo de intercambio de información entre cliente y servidor en XML

*ajax-06-3.js*

```
var conexion;
var alumno;
var materia;
```

```
004 var calificacion;
005 if (document.addEventListener)
006     window.addEventListener("load",principio)
007 else if (document.attachEvent)
008     window.attachEvent("onload", principio);
009
010 function principio(){
011     alumno = document.getElementById('alumno');
012     materia = document.getElementById('materia');
013     calificacion = document.getElementById('calificacion');
014     if (document.addEventListener) {
015         alumno.addEventListener('change', iniciar,false);
016         materia.addEventListener('change', iniciar,false);
017     } else if (document.attachEvent) {
018         alumno.attachEvent('onchange', iniciar,false);
019         materia.attachEvent('onchange', iniciar,false);
020     }
021 }
022 function iniciar(){
023     if (window.XMLHttpRequest){
024         conexion =new XMLHttpRequest();
025     }else if (window.ActiveXObject){
026             try{
027                 conexion =new ActiveXObject("Microsoft.XMLHTTP");
028             }catch(e){
029                 return false;
030             }
031     }
032     if (document.addEventListener)
033         conexion.addEventListener("readystatechange", mostrar)
034     else if (document.attachEvent)
035         conexion.attachEvent("onreadystatechange", mostrar);
036     conexion.open("POST","ajax-06.php",true);
037     var cadena="<datosalumnos><alumno><nombre>"+ alumno.value
    +"</nombre><asignatura>"+materia.value +"</asignatura></alumno></datosalumnos>";
038     conexion.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
039     conexion.setRequestHeader("content-Length",cadena.length);
040     conexion.send(cadena);
041 }
042 function mostrar(){
043     if (conexion.readyState ==4){
044         if (conexion.status ==200){
045             var datos=conexion.responseXML;
046             var tres=datos.getElementsByTagName("nota").item(0);
047             calificacion.value=tres.textContent;
048         }
049     }
050 }
```

*ajax-06-3.html*

```
001 <!doctype html>
002 <html lang="es">
003     <head>
004         <meta charset="utf-8"/>
005         <title>Prueba ajax con XML</title>
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script type="text/javascript" src="js/ajax-06-3.js"> </script>
010         <script type="text/javascript">
011         </script>
012     </head>
013     <body>
014         <nav>
015         </nav>
016         <header>
017         </header>
018         <main>
019             <section>
020                 <article>
021                     <div>
022                         <label for='alumno'>Alumno: </label>
```

```
023                             <select id='alumno'>
024                                 <option value='' selected='selected'>--Elija un alumno--</option>
025                                 <option>Juan Mateos</option>
026                                 <option>Ana Irene Palma</option>
027                             </select>
028                             <label for='materia'>Materia: </label>
029                             <select id='materia'>
030                                 <option value='' selected='selected'>--Elija una materia--</option>
031                                 <option>Lenguaje</option>
032                                 <option>Sociales</option>
033                             </select>
034                             <label for='calificacion'>Calificaci&oacute;n: </label>
035                             <input type='text' readonly='readonly' id='calificacion'/>
036                         </div>
037                     </article>
038                 </section>
039             </main>
040             <footer>
041             </footer>
042             <aside>
043             </aside>
044         </body>
045 </html>
```

*ajax-06.php*

```
001 <?php
002     $dato=fopen('php://input','r');
003     $valor=fgets($dato);
004     $yo=simplexml_load_string($valor);
005     $alumno=$yo->alumno[0]->nombre;
006     $materia=$yo->alumno[0]->asignatura;
007     $nota='0';
008     switch($alumno){
009         case 'Juan Mateos':
010             switch($materia){
011                 case 'Sociales':
012                     $nota='7.5';
013                     break;
014                 case 'Lenguaje':
015                     $nota='9.5';
016                     break;
017             }
018         break;
019         case 'Ana Irene Palma':
020             switch($materia){
021                 case 'Sociales':
022                     $nota='8.5';
023                     break;
024                 case 'Lenguaje':
025                     $nota='7.5';
026                     break;
027                 }
028         break;
029     }
030     header('Content-Type:text/xml');
031 $final='<datosalumnos><alumno><nombre>'.$alumno.'</nombre><asignatura>'.$materia.'</asignatura><nota>'.$nota.'</nota></alumno></datosalumnos>';
032     echo $final;
033 ?>
```

Envió de información mediante JSON.

Vamos a ver un objeto JSON

En este caso tenemos un objeto con dos propiedades que son: **titulo** y **autor** y cada una de esas propiedades con sus respectivos valores: "**El nombre de la rosa**" y "**Umberto Eco**"

```
01 {"titulo": "El nombre de la rosa", "autor": "Umberto Eco"}
```

Otro ejemplo de objetos en notación JSON

En este caso tenemos un array de nombre **libros**, que contiene tres objetos, cada uno de ellos con las propiedades **título** y **autor** y sus respectivos valores.

```
001 {
002 "libros": [
003 { "titulo":"Dime quien soy" , "autor":"Julia Navarro" },
004 { "titulo":"Inferno" , "autor":"Daw Brown" },
005 { "titulo":"Venganza en Sevilla" , "autor":"Matilde Asensi" }
006 ]
007 }
```

Ese mismo ejemplo podría ser también, pero en este caso el array no tiene nombre:

```
001 {
002 [
003 { "titulo":"Dime quien soy" , "autor":"Julia Navarro" },
004 { "titulo":"Inferno" , "autor":"Daw Brown" },
005 { "titulo":"Venganza en Sevilla" , "autor":"Matilde Asensi" }
006 ]
007 }
```

Vamos a crearnos un objeto con los datos que deseamos enviar al programa del servidor, vamos a introducir los datos en el objeto

```
001 var objetoPeticion=new Object();
002 objetoPeticion.alumno=alumno.value;
003 objetoPeticion.materia=materia.value;
```

y luego deberemos convertir ese objeto en un objeto JSON para lo cual vamos a utilizar la función. Podemos utilizar la función directamente al enviar los datos o asignar el objeto JSON a una variable

**JSON.stringify(*objeto*)**

utilizar la función cuando se envían los parámetros

```
001 peticion_http.send(JSON.stringify(objetoPeticion));
```

Asignar el objeto JSON a una variable

```
001 var enviar = JSON.stringify(objetoPeticion);
```

Y este dato es el que vamos a enviar al servidor, mediante post.

Deberemos indicar en las cabeceras que vamos a enviar/recibir datos en formato JSON para lo cual pondremos el método setRequestHeader sobre la petición.

**setRequestHeader("Content-Type", "application/json")**

```
001 peticion_http.setRequestHeader("Content-Type","application/json");
```

Cuando el servidor nos devuelva la respuesta mediante un objeto JSON vamos a tener que convertir el objeto JSON a objeto javascript, para lo cual utilizaremos la función

**JSON.parse(***evento.target.responseText***)**
**JSON.parse(***objeto.responseText***)**

Que nos convierte el objeto JSON que recibe responseText en un objeto de javascript, que ya podemos utilizar.

```
001 objetoRespuesta=JSON.parse(evento.target.responseText);
```

Ejemplo de utilización de JSON para el envío y recepción de datos.

*ajax-07-3.js*

```
001 var peticion_http;
002 var alumno;
003 var materia;
004 var calificacion;
005 var objetoPeticion=new Object();
006 var objetoRespuesta;
007 if (document.addEventListener)
008     document.addEventListener('readystatechange',inicializar,false)
009 else if (document.attachEvent)
010     document.attachEvent('onreadystatechange',inicializar,false);
011 function inicializar(){
012     if (document.readyState=='complete'){
013         alumno=document.getElementById('alumno');
014         materia=document.getElementById('materia');
015         calificacion=document.getElementById('calificacion');
016         if (document.addEventListener) {
017             alumno.addEventListener('change',enviarPeticionAJAX,false);
018             materia.addEventListener('change',enviarPeticionAJAX,false);
019         } else if (document.attachEvent) {
020             alumno.attachEvent('onchange',enviarPeticionAJAX,false);
021             materia.attachEvent('onchange',enviarPeticionAJAX,false);
022         }
023     }
024 }
025 function enviarPeticionAJAX(evento){
026     if (alumno.value!='' && materia.value!=''){
027         objetoPeticion.alumno=alumno.value;
028         objetoPeticion.materia=materia.value;
029         alumno.disabled=true;
030         materia.disabled=true;
031         if( window.XMLHttpRequest){
032             peticion_http=new XMLHttpRequest();
033         }else if(window.ActiveXObject){
034             peticion_http=new ActiveXObject("Microsoft.XMLHTTP");
035         }
036         if (document.addEventListener)
037 peticion_http.addEventListener('readystatechange',gestionarRespuesta,false)
038         else if (document.attachEvent)
039 peticion_http.attachEvent('onreadystatechange',gestionarRespuesta,false)
040         peticion_http.open('POST','ajax-07.php',true);
041         peticion_http.setRequestHeader("Content-Type","application/json");
042         peticion_http.send(JSON.stringify(objetoPeticion));
043     }else{
044         calificacion.value='';
045     }
046 }
047 function gestionarRespuesta(evento){
048     if (evento.target.readyState==4 && evento.target.status==200){
049         alumno.disabled=false;
050         materia.disabled=false;
051         objetoRespuesta=JSON.parse(evento.target.responseText);
052         calificacion.value=objetoRespuesta.calificacion;
053     }
054 }
```

*ajax-07-3.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <meta charset="utf-8"/>
005         <title>AJAX</title>
006         <meta name="author" content="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
```

```
009         <script type="text/javascript" src="js/ajax-07-3.js"> </script>
010         <script type="text/javascript">
011         </script>
012     </head>
013     <body>
014         <nav>
015         </nav>
016         <header>
017         </header>
018         <main>
019             <section>
020                 <article>
021                     <div>
022                         <form id='formulario'>
023                             <label for='alumno'>Alumno: </label>
024                             <select id='alumno' name='alumno'>
025                                 <option value='' selected='selected'>--Elija un alumno--</option>
026                                 <option>Juan Mateos</option>
027                                 <option>Ana Irene Palma</option>
028                             </select>
029                             <label for='materia'>Materia: </label>
030                             <select id='materia' name='materia'>
031                                 <option value='' selected='selected'>--Elija una materia--</option>
032                                 <option>Lenguaje</option>
033                                 <option>Sociales</option>
034                             </select>
035                             <label for='calificacion'>Calificaci&oacute;n: </label>
036                             <input type='text' readonly='readonly' id='calificacion'/>
037                         </form>
038                     </div>
039                 </article>
040             </section>
041         </main>
042         <footer>
043         </footer>
044         <aside>
045         </aside>
046     </body>
047 </html>
```

*ajax-07.php*

```
001 <?php
002     $entrada=fopen('php://input','r');
003     $datos= fgets($entrada);
004     $datos= json_decode($datos,true);
005 $resultado=(object)array('alumno'=>$datos['alumno'],'materia'=>$datos['materia'],'calificacion'=>0);
006     switch($datos['alumno']){
007         case 'Juan Mateos':
008             switch($datos['materia']){
009                 case 'Sociales':
010                     $resultado->calificacion=7.5;
011                     break;
012                 case 'Lenguaje':
013                     $resultado->calificacion=9.5;
014                     break;
015             }
016         break;
017         case 'Ana Irene Palma':
018             switch($datos['materia']){
019                 case 'Sociales':
020                     $resultado->calificacion=8.5;
021                     break;
022                 case 'Lenguaje':
023                     $resultado->calificacion=7.5;
024                     break;
025             }
026         break;
```

```
027     }
028     $respuesta=json_encode($resultado);
029     echo$respuesta;
030 ?>
```

Otro programa php, que se podría haber llamado

*ajax-07-1.php*

```
001 <?php
002     $entrada=fopen('php://input','r');
003     $datos= fgets($entrada);
004     $datos= json_decode($datos,true);
005     switch($datos['alumno']){
006         case 'Juan Mateos':
007             switch($datos['materia']){
008                 case 'Sociales':
009                     echo '{"calificacion":7.5}';
010                     break;
011                 case 'Lenguaje':
012                     echo '{"calificacion":9.5}';
013                     break;
014             }
015             break;
016         case 'Ana Irene Palma':
017             switch($datos['materia']){
018                 case 'Sociales':
019                     echo '{"calificacion":8.5}';
020                     break;
021                 case 'Lenguaje':
022                     echo '{"calificacion":7.5}';
023                     break;
024             }
025             break;
026     }
027 ?>
```

### d) Notificaciones.

Para detectar las notificaciones que nos envía el servidor las vamos a tener que poner un oyente de eventos y vamos a detectar el cambio de estado para lo cual vamos a poner

***nombre-objeto*.addEventListener('readystatechange', *nombre-función*, false);**

En la función que tendremos más adelante vamos a determinar los valores que pueden tomar dos propiedades del evento, estas propiedades son:

evento.target.stateReady-> estado de la solicitud
evento.target.status -> código de estado
evento.target.statusText -> mensaje de estado

| Valor | Descripción |
|---|---|
| 0 | **Sin inicializar**. Aún no se ha llamado al método open del objeto XMLHttpRequest. |
| 1 | **Abierto**. Se ha llamado al método open del objeto XMLHttpRequest, pero no al método send. |
| 2 | **Enviado**. Se ha llamado al método send del objeto XMLHttpRequest, pero aún no se ha recibido la respuesta del servidor. |
| 3 | **Recibiendo**. Se ha empezado a recibir la respuesta del servidor. |
| 4 | **Completado**. Se ha recibido la respuesta completa del servidor. Tenga en cuenta que esto no quiere decir que la petición se haya tramitado correctamente; por ejemplo, puede ser que el servidor haya contestado con un código 404. Para tener la certeza de que una petición se ha tramitado correctamente tendremos que tener el valor 4 en readyState y el valor 200 en status. |

evento.target.status ->codigo de estado

| Valor | Descripción |
|---|---|
| 100 | **Continuar** .Esta respuesta provisional se utiliza para informar al cliente de que la parte inicial de la solicitud ha sido recibida y aún no ha sido rechazada por el servidor |
| 101 | **Protocolos de conmutación.** El servidor cambiará los protocolos a los que se definen por el campo de cabecera de actualización de la respuesta inmediatamente después de la línea en blanco que termina la respuesta 101 |
| 200 | **OK.** La solicitud ha tenido éxito |
| 201 | **Creado.** La solicitud se ha completado y hay un nuevo recurso que se está creando |
| 202 | **Aceptado**. La solicitud ha sido admitida a trámite, pero el proceso no se ha completado |
| 203 | **Información no autoritativa**. La metainformación devuelta en la entidad cabecera no es el conjunto definitivo disponible en el servidor de origen, pero se obtiene de un local o una copia de terceros. |
| 204 | **Sin contenido**. El servidor ha cumplido la petición, pero no necesita volver un-cuerpo de la entidad, y podría querer volver metainformación actualizada. |
| 205 | **Restablecer contenido**. El servidor ha cumplido con la solicitud y el agente de usuario debe restablecer la vista del documento que provocó la petición que se enviará |
| 206 | **Contenido parcial**. El servidor ha cumplido con la solicitud GET parcial del recurso. |
| 300 | **MultipleChoices**. El recurso solicitado corresponde a cualquiera de un conjunto de representaciones, cada una con su ubicación específica, y agente- información negociación impulsadaestá siendo proporcionado para que el (o agente de usuario) el usuario puede seleccionar una representación preferida y reorientar su solicitar a esa ubicación. |
| 301 | **Moved Permanently**. El recurso solicitado se ha asignado un nuevo URI permanente y todas las referencias futuras a este recurso es conveniente utilizar uno de los URI devueltas. |
| 302 | **Found**. El recurso solicitado reside temporalmente bajo un URI diferente. |
| 303 | **SeeOther**. La respuesta a la solicitud se puede encontrar bajo un URI diferente y debe ser recuperada mediante un método GET en ese recurso |
| 304 | **NotModified**. Si el cliente ha realizado una petición GET condicional y se le permite el acceso, pero el documento no ha sido modificado, el servidor debe responder con este código de estado. |
| 305 | **Uso Proxy**. Debe tener acceso al recurso solicitado a través del proxy propuesta por el campo Ubicación. El campo Ubicación da la URI del proxy. |
| 307 | **Redirección temporal**. El recurso solicitado reside temporalmente bajo un URI diferente. |
| 400 | **BadRequest**. La solicitud no puede ser entendida por el servidor debido a sintaxis incorrecta. El cliente no debe repetir la solicitud sin modificaciones. |
| 401 | **Unauthorized**. La solicitud requiere autenticación de usuario. |
| 403 | **Forbidden**. El servidor ha entendido la petición, pero se niega a cumplirla. |
| 404 | **NotFound**. El servidor no ha encontrado nada que coincida con la URI de solicitud. |
| 405 | **Método no permitido**. El método especificado en la Solicitud-Line no está permitido para el recurso identificado por el Request-URI. |
| 406 | **No aceptable**. El recurso identificado por la petición sólo puede generar entidades de respuesta que tengan características de contenido no aceptables de acuerdo con las cabeceras de aceptación enviadas en la petición. |
| 407 | **Autentificación de poder**. Este código es similar al 401 (Unauthorized), pero indica |

| Valor | Descripción |
|---|---|
| | que el cliente debe autenticarse con el proxy. |
| 408 | **RequestTimeout**. El cliente no produjo una solicitud dentro del plazo que el servidor estaba dispuesto a esperar. |
| 409 | **Conflicto**. La petición no se pudo completar debido a un conflicto con el estado actual del recurso. |
| 410 | **Gone**. El recurso solicitado ya no está disponible en el servidor y no es la dirección de reenvío se conoce. |
| 411 | **Longitud Requerido**. El servidor se niega a aceptar la solicitud sin un Content-longitud definida. |
| 412 | **Requisito Error**. La precondición dada en uno o más de los campos de cabecera de solicitud evaluada como falsa cuando se puso a prueba en el servidor. Este código de respuesta permite al cliente situar precondiciones en la información meta del recurso actual (cabecera de datos de campo) y así evitar que el método solicitado se aplique a un recurso que no sea el previsto. |
| 413 | **Solicitud Entidad demasiado grande**. El servidor se niega a procesar una solicitud debido a que la entidad de solicitud es más grande que el servidor está dispuesto o es capaz de procesar. |
| 414 | **Request-URI demasiado largo**. El servidor se niega a entregar la petición porque la URI de solicitud es más largo que el servidor está dispuesto a interpretar. |
| 415 | **Unsupported Media Type**. El servidor se niega a atender la solicitud porque la entidad de la petición está en un formato no soportado por el recurso solicitado para el método solicitado. |
| 416 | **Rango solicitado no satisfiable**. Un servidor DEBE devolver una respuesta con este código de estado si la solicitud incluye un campo Rango petición-header (sección 14.35), y ninguno de los valores de rango de especificador en este campo solapar la extensión actual del recurso seleccionado, y la solicitud no lo hizo incluir un campo de petición-cabecera If-Range. (Para byte-rangos, esto significa que la primera byte-pos de todos los valores de byte-range-spec fuera mayor que la longitud actual del recurso seleccionado.) |
| 417 | **Expectativa Falló**. La expectativa dado en un campo de petición-header. Esperar no pudo ser cumplido por este servidor, o, si el servidor es un proxy, el servidor tiene pruebas inequívocas de que la solicitud no pudo ser satisfecha por el servidor del próximo salto |
| 500 | **Internal Server Error**. El servidor encontró una condición inesperada que le impidió cumplir con la solicitud. |
| 501 | **No implementado**. El servidor no soporta la funcionalidad requerida para completar la petición. |
| 502 | **Puerta de enlace incorrecta**. El servidor, mientras actúa como pasarela o proxy, recibió una respuesta inválida del servidor upstream que accedió cuando intentaba completar la petición. |
| 503 | **ServiceUnavailable**. El servidor no puede procesar la solicitud debido a una sobrecarga temporal o mantenimiento del servidor. |
| 504 | **Puerta de enlace de tiempo de espera**. El servidor, mientras actúa como pasarela o proxy, no recibió una respuesta a tiempo del servidor upstream especificado por el URI (por ejemplo, HTTP, FTP, LDAP) o algún otro servidor auxiliar (por ejemplo, DNS) que necesitaba para el acceso en el intento de completar la solicitud. |
| 505 | **VersionNotSupported**. El servidor no es compatible, o se niega a apoyar, la versión del protocolo HTTP que se utilizó en el mensaje de petición. |

evento.target.statusText -> mensaje de estado

## e) Librerías de actualización dinámica.

Entre las librerías que podemos utilizar para la actualización dinámica tenemos jquery, a través del cual vamos a poder comunicarnos mediante ajax.

Vamos a ver el acceso a Ajax mediante jquery.

Para realizar estas operaciones tenemos losmétodos::

- load
- get
- post
- ajax

El método load tiene los siguientes formatos:

Si queremos coger el contenido de un fichero que se encuentra en el servidor y cargarlo en un elemento, indicado por el selector deberemos poner

**$(selector).load(fichero);**

*ajax-08-2.js*

```
001 function descargaArchivo(fichero){
002     $("#primero").load("http://localhost/ajax08/"+ fichero);
003 }
004 $(function(){descargaArchivo("holamundo.txt");})
```

*ajax-08-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <title>Hola Mundo con AJAX</title>
005         <meta charset="utf-8"/>
006         <meta name="author" value="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script src="http://code.jquery.com/jquery-3.5.1.js"></script>
010         <script src="js/ajax-08-2.js" type="text/javascript"> </script>
011         <script type="text/javascript">
012         </script>
013     </head>
014     <body>
015         <nav>
016         </nav>
017         <header>
018         </header>
019         <main>
020             <section>
021                 <article>
022                     <div>
023                         <a href="javascript:descargaArchivo('holamundo.txt');">
    Hola Mundo</a><br/>
024                         <a href="javascript:descargaArchivo('segovia.html');">
    Segovia</a><br/>
025                         <a href="javascript:descargaArchivo('madrid.html');">
    Madrid</a><br/>
026                         <div id="primero">
027                         </div>
028                     </div>
029                 </article>
030             </section>
031         </main>
032         <footer>
033         </footer>
034         <aside>
035         </aside>
036     </body>
```

```
037 </html>
```

Si queremos cargar el contenido la respuesta del servidor en un elemento del servidor y realizar el paso de parámetros mediante GET usaremos

**$(selector).load(fichero?nombre-1=valor-1&nombre-2=valor2…, función);**

*ajax-09-2.js*

```
001 function llamada(){
002     if($("#alumno").val()!=''&& $("#materia").val()!=''){
003         $("#alumno").disabled =true;
004         $("#materia").disabled =true;
005         var alum=quitarBlanco($("#alumno option:selected").text());
006         var mate=quitarBlanco($("#materia option:selected").text());
007         $("#yo").load("ajax-04.php?alumno="+alum+"&materia="+ mate  ,
008             function(valor){
009                 $("#calificacion").val(valor);
010             });
011     }
012 }
013 $(function(){
014     $("#alumno").on("change",function(){llamada();})
015     $("#materia").change(function(){llamada();});
016 });
017 function quitarBlanco(cadena){
018     cadena=cadena.trim();
019     var datos="";
020     var inicio=0;
021     var fin=cadena.indexOf(" ");
022     while(fin !=-1){
023         datos+=cadena.substring(inicio,fin)+"-";
024         inicio=fin +1;
025         fin=cadena.indexOf(" ", inicio);
026     }
027     datos+=cadena.substring(inicio,cadena.length);
028     return datos;
029 }
```

*ajax-09-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <title>AJAX con jquery</title>
005         <meta charset="utf-8"/>
006         <meta name="author" value="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script src="http://code.jquery.com/jquery-3.5.1.js"></script>
010         <script src="js/ajax-09-2.js" type="text/javascript"> </script>
011         <script type="text/javascript">
012         </script>
013     </head>
014     <body>
015         <nav>
016         </nav>
017         <header>
018         </header>
019         <main>
020             <section>
021                 <article>
022                     <div>
023                         <label for='alumno'>Alumno: </label>
024                         <select id='alumno'>
025                             <option value='' selected='selected'>--Elija un alumno-
-</option>
026                             <option>Juan Mateos</option>
027                             <option>Ana Irene Palma</option>
028                         </select>
029                         <label for='materia'>Materia: </label>
```

```
030                         <select id='materia'>
031                             <option value='' selected='selected'>--Elija una materia--</option>
032                             <option>Lenguaje</option>
033                             <option>Sociales</option>
034                         </select>
035                         <label for='calificacion'>Calificaci&oacute;n: </label>
036                         <input type='text' readonly='readonly' id='calificacion'/>
037                         <div id="yo" hidden></html>
038                     </div>
039                 </article>
040             </section>
041         </main>
042         <footer>
043         </footer>
044         <aside>
045         </aside>
046     </body>
047 </html>
```

En este caso para realizar el paso de parámetros ponemos después del nombre el signo de final de interrogación, el nombre del parámetro el signo igual y el valor del parámetro, si deseamos incluir más parámetros deberemos poner a continuación el signo del ampersan el nombre del parámetro el signo igual y el valor del parámetro y así sucesivamente, tantas veces como parámetros deseamos pasar al programa. Aquí deberemos tener cuidado con el valor del parámetro, ya que si éste tiene espacios en blanco nos va a dar problemas. La forma de solucionar este problema deberemos realizar la siguiente modificación cuando ponemos los parámetros. Deberemos poner

**$.param({ nombre-parámetro-1 : valor-1 , nombre-parametro-2 : valor-2 … })**

Con lo cual el método quedara de la siguiente forma

**$(selector).load(fichero?$.param({ nombre-parámetro-1 : valor-1 , nombre-parametro-2 : valor-2 … }), función);**

*ajax-10-2.js*

```
001 function llamada(){
002     if($("#alumno").val()!=''&& $("#materia").val()!=''){
003         $("#alumno").disabled =true;
004         $("#materia").disabled =true;
005         $("#yo").load("ajax102.php?"+$.param({ alumno:$("#alumno option:selected").text(),materia:$("#materia option:selected").text()}),
006             function(valor){
007                 $("#calificacion").val(valor);
008             });
009     }
010 }
011 $(function(){
012     $("#alumno").on("change",function(){llamada();})
013     $("#materia").change(function(){llamada();});
014 });
```

*ajax-10-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <title>AJAX con jquery</title>
005         <meta charset="utf-8"/>
006         <meta name="author" value="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script src="http://code.jquery.com/jquery-3.5.1.js"></script>
010         <script src="js/ajax-10-2.js" type="text/javascript">  </script>
011         <script type="text/javascript">
```

```
        </script>
    </head>
    <body>
        <nav>
        </nav>
        <header>
        </header>
        <main>
            <section>
                <article>
                    <div>
                        <label for='alumno'>Alumno: </label>
                        <select id='alumno'>
                            <option value='' selected='selected'>--Elija un alumno--</option>
                            <option>Juan Mateos</option>
                            <option>Ana Irene Palma</option>
                        </select>
                        <label for='materia'>Materia: </label>
                        <select id='materia'>
                            <option value='' selected='selected'>--Elija una materia--</option>
                            <option>Lenguaje</option>
                            <option>Sociales</option>
                        </select>
                        <label for='calificacion'>Calificaci&oacute;n: </label>
                        <input type='text' readonly='readonly' id='calificacion'/>
                        <div id="yo" hidden></html>
                    </div>
                </article>
            </section>
        </main>
        <footer>
        </footer>
        <aside>
        </aside>
    </body>
</html>
```

Si queremos cargar el contenido la respuesta del servidor en un elemento del servidor y realizar el paso de parámetros mediante POST usaremos

**$(selector).load(fichero,{ nombre-parámetro-1 : valor-1 , nombre-parametro-2 : valor-2 … }, función);**

*ajax-11-2.js*

```
function llamada(){
    if($("#alumno").val()!=''&& $("#materia").val()!=''){
        $("#alumno").disabled =true;
        $("#materia").disabled =true;
        $("#yo").load("http://localhost/ajax08/ajax102.php",{alumno:$("#alumno option:selected").text(),materia:$("#materia option:selected").text()},
            function(valor){
                $("#calificacion").val(valor);
        });
    }
}
$(function(){
    $("#alumno").on("change",function(){llamada();})
    $("#materia").change(function(){llamada();});
});
```

*ajax-11-2.html*

```
<!DOCTYPE html>
<html lang="es">
    <head>
    <title>AJAX con jquery</title>
    <meta charset="utf-8"/>
    <meta name="author" value="Félix Ángel Muñoz Bayón" />
```

```
007         <style type="text/css">
008         </style>
009     <script src="http://code.jquery.com/jquery-3.5.1.js"></script>
010     <script src="js/ajax-11-2.js" type="text/javascript"> </script>
011     <script type="text/javascript">
012     </script>
013     </head>
014     <body>
015         <nav>
016         </nav>
017         <header>
018         </header>
019         <main>
020             <section>
021                 <article>
022                     <div>
023                         <label for='alumno'>Alumno: </label>
024                         <select id='alumno'>
025                             <option value='' selected='selected'>--Elija un alumno--</option>
026                             <option>Juan Mateos</option>
027                             <option>Ana Irene Palma</option>
028                         </select>
029                         <label for='materia'>Materia: </label>
030                         <select id='materia'>
031                             <option value='' selected='selected'>--Elija una materia--</option>
032                             <option>Lenguaje</option>
033                             <option>Sociales</option>
034                         </select>
035                         <label for='calificacion'>Calificaci&oacute;n: </label>
036                         <input type='text' readonly='readonly' id='calificacion'/>
037                         <div id="yo" hidden>
038                         </div>
039                     </div>
040                 </article>
041             </section>
042         </main>
043         <footer>
044         </footer>
045         <aside>
046         </aside>
047     </body>
048 </html>
```

En los casos anteriores, si queremos realizar ciertas operaciones con el valor devuelto y no mostrarle en un elemento deberemos poner un elemento como oculto atributo hidden en html.

En todos los casos anteriores el método load tiene un parámetro adicional que se corresponde con una función que se ejecutará cuando devuelve el valor el programa del servidor. Esta función puede tener hasta tres parámetros, que se corresponde con el valor devuelto, el valor del estado y el objeto de respuesta.

**.load(***url* **[,***datos***] [,** *función***])**→ equivale a .get.

Formato get en jQuery

**$.get(fichero [, parámetros] [, function-vuelve] [, tipo-dato])**

El tipo de dato puede ser: xml, json, script, html y es el tipo de dato que devuelve el servidor.

Los parámetros que se le pasan al servidor en el formato

**{ nombre-parámetro-1 : valor-1 , nombre-parámetro-2 : valor-2 … }**

También puede ser un array en el formato

**{nombre-array[] : [ lista-valores]}**

*ajax-12-2.js*

```
001 function llamada(){
002     if($("#alumno").val()!=''&& $("#materia").val()!=''){
003         $("#alumno").disabled =true;
004         $("#materia").disabled =true;
005         console.log('alumno:'+$("#alumno
    option:selected").text()+',materia:'+$("#materia option:selected").text());
006         var dato= $.get("ajax-04.php",{alumno:"Juan Mateos",materia:"Lenguaje"},
    muestra);
007     }
008 }
009 function muestra(valor){
010     $("#calificacion").text(valor);
011 }
012 $(function(){
013     $("#alumno").on("change",function(){llamada();})
014     $("#materia").change(function(){llamada();});
015 });
```

*ajax-12-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <title>AJAX con jquery</title>
005         <meta charset="utf-8"/>
006         <meta name="author" value="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script src="http://code.jquery.com/jquery-3.5.1.js"></script>
010         <script src="js/ajax-12-2.js" type="text/javascript"> </script>
011         <script type="text/javascript">
012         </script>
013     </head>
014     <body>
015         <nav>
016         </nav>
017         <header>
018         </header>
019         <main>
020             <section>
021                 <article>
022                     <div>
023                         <label for='alumno'>Alumno: </label>
024                         <select id='alumno'>
025                             <option value='' selected='selected'>--Elija un alumno-
    -</option>
026                             <option>Juan Mateos</option>
027                             <option>Ana Irene Palma</option>
028                         </select>
029                         <label for='materia'>Materia: </label>
030                         <select id='materia'>
031                             <option value='' selected='selected'>--Elija una
    materia--</option>
032                             <option>Lenguaje</option>
033                             <option>Sociales</option>
034                         </select>
035                         <label for='calificacion'>Calificaci&oacute;n: </label>
036                         <input type='text' readonly='readonly' id='calificacion'/>
037                     </div>
038                 </article>
039             </section>
040         </main>
041         <footer>
042         </footer>
```

```
043         <aside>
044         </aside>
045     </body>
046 </html>
```

**$.get(***url* **[,***data***] [,***función***] [,***tipo-dato-devuelto***])**→ realiza una solicitud mediante get, a la url indicada con los datos que se indican y que cuando finaliza con éxito se ejecuta la función, los tipos de datos devueltos pueden tener uno de los siguientes valores: xml, json, script, text, html.

**$.get(***opciones***)**→ realiza una solicitud mediante get utilizando las ocpiones que son un objeto y que tienen las siguientes opciones:

- url: dirección
- data:datos
- success:función
- dataType:tipo-dato-devuelto

Formatopost en jQuery

**$.post(fichero [, parámetros] [, function-vuelve] [, tipo-dato])**

el tipo de dato puede ser: xml, json, script, html y es el tipo de dato que devuelve el servidor.

los parámetros que se le pasan al servidor en el formato

**{ nombre-parámetro-1 : valor-1 , nombre-parámetro-2 : valor-2 …** }

también puede ser un array en el formato

**{nombre-array[] : [ lista-valores]}**

*ajax-13-2.js*

```
001 function llamada(){
002     if($("#alumno").val()!=''&& $("#materia").val()!=''){
003         $("#alumno").disabled =true;
004         $("#materia").disabled =true;
005         var dato= $.post("ajax-04.php",{alumno:"Juan Mateos",materia:"Lenguaje"},
006             function(valor){
007                 $("#calificacion").text(valor);
008     }).always(function(valor){alert("se acabo"+valor+"-");
    $("#calificacion").text(valor);});
009     }
010 }
011 $(function(){
012     $("#alumno").on("change",function(){llamada();})
013     $("#materia").change(function(){llamada();});
014 });
```

*ajax-13-2.html*

```
001 <!DOCTYPE html>
002 <html lang="es">
003     <head>
004         <title>AJAX con jquery</title>
005         <meta charset="utf-8"/>
006         <meta name="author" value="Félix Ángel Muñoz Bayón" />
007         <style type="text/css">
008         </style>
009         <script src="http://code.jquery.com/jquery-3.5.1.js"></script>
010         <script src="js/ajax-13-2.js" type="text/javascript"> </script>
011         <script type="text/javascript">
012         </script>
```

```
013     </head>
014     <body>
015         <nav>
016         </nav>
017         <header>
018         </header>
019         <main>
020             <section>
021                 <article>
022                     <div>
023                         <label for='alumno'>Alumno: </label>
024                         <select id='alumno'>
025                             <option value='' selected='selected'>--Elija un alumno--</option>
026                             <option>Juan Mateos</option>
027                             <option>Ana Irene Palma</option>
028                         </select>
029                         <label for='materia'>Materia: </label>
030                         <select id='materia'>
031                             <option value='' selected='selected'>--Elija una materia--</option>
032                             <option>Lenguaje</option>
033                             <option>Sociales</option>
034                         </select>
035                         <label for='calificacion'>Calificaci&oacute;n: </label>
036                         <input type='text' readonly='readonly' id='calificacion'/>
037                     </div>
038                 </article>
039             </section>
040         </main>
041         <footer>
042         </footer>
043         <aside>
044         </aside>
045     </body>
046 </html>
```

**$.post(***url* **[,***data***] [,***función***] [,***tipo-dato-devuelto***])**→ realiza una solicitud mediante post, a la url indicada con los datos que se indican y que cuando finaliza con éxito se ejecuta la función, los tipos de datos devueltos pueden tener uno de los siguientes valores: xml, json, script, text, html.

**$.post(***opciones***)**→ realiza una solicitud mediante post utilizando las opciones que son un objeto y que tienen las siguientes opciones:

- type: "POST"
- url: dirección
- data:datos
- success:función
- dataType:tipo-dato-devuelto

**$.ajax(***url* **[,** *opciones* **])** → realiza una conexión asíncrona con la url indicada. Las opciones son un conjunto de pares claves valor que nos permiten configurar la conexión.

**$.ajax([***opciones***])**→ realiza la conexión asíncrona con las opciones indicas o bien las establecidas mediante el método **ajaxSetup**.

**$.ajaxSetup(***opciones***)**→ permiten realizar la configuración de una conexión asíncrona
Las claves que tenemos son:

- **url**⇒cadena que contiene la url del fichero al que se quiere acceder en el servidor para establecer la comunicación asíncrona.
- **data**⇒objeto, Datos a enviar al servidor. Se convierte en una cadena de consulta, si no es ya una cadena. Se adjunta a la url para las solicitudes GET. Consulte la opción processData para evitar este procesamiento automático. El

objeto debe ser pares clave / valor. Si el valor es una matriz, jQuery serializa varios valores con la misma clave en función del valor de la configuración tradicional.

- **dataType**⇒ cadena que indica el tipo de datos que espera recibir del servidor, puede tomar uno de los siguientes valores:
  - **xml** recibe un dato xml
  - **html**recibe un valor en html
  - **script** recibe un script de javascript.
  - **json** valor en json, el servidor le envía un objeto json, que se transforma directamente en un objeto javascript, con lo cual es como si recibiéramos un objeto javascript.
  - **jsonp** valor json
  - **text** cadena de texto plano, el servidor le puede enviar un json, que luego deberemos transformar un objeto javascript.
- **method**⇒cadena que indica el tipo de solicitud con uno delos siguientes valores
  - GET, pordefecto
  - POST
  - PUT
- **type**⇒cadena, alias de method
- **headers**⇒Un objeto de pares clave / valor de encabezado adicionales para enviar junto con las solicitudes utilizando el transporte XMLHttpRequest. El encabezado X-Requested-With: XMLHttpRequest siempre se agrega, pero su valor predeterminado de XMLHttpRequest se puede cambiar aquí. Los valores en la configuración de encabezados también se pueden sobrescribir desde dentro de la función beforeSend.
- **success**⇒Una función a la que llamar si la solicitud es satisfactoria. La función se pasa tres argumentos: los datos devueltos desde el servidor, formateados de acuerdo con el parámetro dataType o la función de devolución de llamada dataFilter, si se especifica; una cadena que describe el estado; y el objeto jqXHR (en jQuery 1.4.x, XMLHttpRequest). A partir de jQuery 1.5, la configuración de éxito puede aceptar una serie de funciones. Cada función será llamada a su vez. Este es un evento de Ajax.
- **error⇒**Función que se ejecuta si la conexión falla. La función recibe tres argumentos: el objeto jqXHR (en jQuery 1.4.x, XMLHttpRequest), una cadena que describe el tipo de error que ocurrió y un objeto de excepción opcional, si se produjo uno.
- **complete**⇒ Función que se ejecuta cuando ha finalizado una comunicación, después de las llamadas de éxito y error.La función pasa dos argumentos: el objeto jqXHR (en jQuery 1.4.x, XMLHTTPRequest) y una cadena que categoriza el estado de la solicitud ("success", "notmodified", "nocontent", "error", "timeout", "
- **username**⇒ nombre de usuario de una solicitud
- **password**⇒ contraseña de una solicitud.
- **statusCode**⇒Un objeto de códigos y funciones HTTP numéricos que se llamarán cuando la respuesta tenga el código correspondiente. Por ejemplo, lo siguiente alertará cuando el estado de respuesta sea un 404:Si la solicitud es exitosa, las funciones del código de estado toman los mismos parámetros que la devolución de llamada exitosa; si da como resultado un error (incluida la

redirección 3xx), toman los mismos parámetros que la devolución de llamada de error.

- **accepts**⇒objeto conjunto de clave valor que asigna valor a su tipo MIME, que se envía en el encabezado de la solicitud, este encabezado le dice al servidor que tipo de respuesta aceptará a cambio.
- **async**⇒ valor-lógico indica si la conexión es asíncrona, por defecto, si queremos que sea síncrona deberemos establecer este valor a false.
- **beforeSend**⇒ función que se ejecuta antes de realizar la comunicación asíncrona, se suele utilizar para establecer encabezados personalizados.Los objetos jqXHR y de configuración se pasan como argumentos
- **cache**⇒ valor-lógico indica si las páginas solicitas se almacenan en la cache, por defecto true.
- **contents**⇒ objeto de pares cadenas expresiones regulares que determinan como jquery analizará la respuesta.
- **contentType**⇒ lógico o cadena cuando envía datos al servidor asigne valor a esta opción. Por defecto tiene el valor "application/x-www-form-urlencoded; charset = UTF-8". Si se le asigna el valor falso se indica que no establezca ningún encabezado de tipo contenido. Se puede asignaruno de los valores "application/x-www-form-urlencoded", "multipart/form-data" o "text/plain".
- **context**⇒Este objeto será el contexto de todas las devoluciones de llamada relacionadas con Ajax. De forma predeterminada, el contexto es un objeto que representa la configuración de Ajax utilizada en la Por ejemplo, especificar un elemento DOM como el contexto hará que el contexto para la devolución de llamada completa de una solicitud
- **converters**⇒Un objeto que contiene convertidores de tipo de datos tipo a datos. El valor de cada convertidor es una función que devuelve el valor transformado de la respuesta. Por defecto `{"* text": window.String, "texthtml": true, "textjson": jQuery.parseJSON, "textxml": jQuery.parseXML}`
- **crossDomain**⇒valor-lógico, Si desea forzar una solicitud de dominio cruzado (como JSONP) en el mismo dominio, establezca el valor de dominio cruzado en verdadero. Esto permite, por ejemplo, la redirección del lado del servidor a otro dominio.Pordefecto`falseforsame-domainrequests, true forcross-domainrequests`
- **dataFilter**⇒Una función que se utilizará para manejar los datos de respuesta sin procesar de XMLHttpRequest. Esta es una función de pre-filtrado para sanear la respuesta. Usted debe devolver los datos desinfectados. La función acepta dos argumentos: los datos sin procesar devueltos por el servidor y el parámetro 'dataType'.
- **global**⇒valor-lógico, Ya sea para activar los controladores de eventos Ajax globales para esta solicitud. El defecto es cierto. Establézcalo en falso para evitar que se activen los controladores globales como ajaxStart o ajaxStop. Esto se puede utilizar para controlar varios eventos Ajax.
- **ifModified**⇒valor-lógico, Permita que la solicitud sea exitosa solo si la respuesta ha cambiado desde la última solicitud. Por defecto false.
- **isLocal**⇒valor-lógico, Permite que el entorno actual se reconozca como "local" (por ejemplo, el sistema de archivos), incluso si jQuery no lo reconoce como tal de forma predeterminada.
- **jsonp**⇒cadena o lógico. Reemplace el nombre de la función de devolución de llamada en una solicitud JSONP. Este valor se utilizará en lugar de 'devolución

de llamada' en el 'devolución de llamada =?' parte de la cadena de consulta en la url. Entonces {jsonp: 'onJSONPLoad'} resultaría en 'onJSONPLoad =?' Pasado al servidor. A partir de jQuery 1.5, establecer la opción jsonp en falso evita que jQuery agregue la cadena "?Callback" a la URL o intente usar "=?" para la transformación. En este caso, también debe establecer explícitamente la configuración jsonpCallback. Por ejemplo, {jsonp: false, jsonpCallback: "callbackName"}. Si no confía en el destino de sus solicitudes de Ajax, considere establecer la propiedad jsonp en falso por razones de seguridad.

- **jsonpCallback⇒** cadena o función. Especifique el nombre de la función de devolución de llamada para una solicitud JSONP. Este valor se utilizará en lugar del nombre aleatorio generado automáticamente por jQuery. Es preferible dejar que jQuery genere un nombre único, ya que facilitará la gestión de las solicitudes y proporcionará devoluciones de llamadas y manejo de errores. Es posible que desee especificar la devolución de llamada cuando desee habilitar un mejor almacenamiento en caché de las solicitudes GET. A partir de jQuery 1.5, también puede usar una función para esta configuración, en cuyo caso el valor de jsonpCallback se establece en el valor de retorno de esa función.
- **mimeType**⇒cadena, Un tipo mime para anular el tipo mime XHR.
- **processData**⇒lógico, De forma predeterminada, los datos que se pasan a la opción de datos como un objeto (técnicamente, cualquier otra cosa que no sea una cadena) se procesarán y transformarán en una cadena de consulta, ajustándose al tipo de contenido predeterminado "application/x-www-form-urlencoded" . Si desea enviar un DOMDocument u otros datos no procesados, establezca esta opción en falso. Por defecto true.
- **scriptChatset**⇒Solo se aplica cuando se usa el transporte de "script" (por ejemplo, solicitudes de dominio cruzado con "jsonp" o "script" dataType y tipo "GET"). Establece el atributo charset en la etiqueta de script utilizada en la solicitud. Se utiliza cuando el conjunto de caracteres en la página local no es el mismo que el del script remoto.
- **timeout**⇒número tiempo en milisegundos para que falle la solicitud, con el valor cero tiempo indefinido.
- **traditional**⇒valor-lógico, Establézcalo en verdadero si desea utilizar el estilo tradicional de serialización de parámetros.
- **xhr⇒**Devolución de llamada para crear el objeto XMLHttpRequest. El valor predeterminado es ActiveXObject cuando está disponible (IE), de lo contrario, XMLHttpRequest. Anular para proporcionar su propia implementación para XMLHttpRequest o mejoras a la fábrica.
- **xhrFields⇒** objeto, Un objeto de fieldName-fieldValue pares para establecer en el objeto XHR nativo. Por ejemplo, puede usarlo para establecer withCredentials en true para las solicitudes de dominios cruzados si es necesario.EnjQuery 1.5, la propiedad withCredentials no se propagó al XHR nativo y, por lo tanto, las solicitudes CORS que lo requieren ignorarán este indicador. Por este motivo, le recomendamos que utilice jQuery 1.5.1+ en caso de que requiera su uso.

*ejemplo-jquery-ajax-002.js*

```
001 function iniciar(){
002     $.ajax("http://localhost/ejemplo-jquery-ajax-001.php",{complete:function(resultado,configuracion){
003         $("#respuesta").text(resultado.responseText);
004 }});
005 }
```

*ejemplo-jquery-ajax-003.js*

```
001 function iniciar(){
002     $.ajax({url:"http://localhost/ejemplo-jquery-ajax-001.php",complete:function(resultado,configuracion){
003         $("#respuesta").text(resultado.responseText);
004 }});
005 }
```

*ejemplo-jquery-ajax-004.js*

```
001 function iniciar(){
002     $.ajaxSetup({url:"http://localhost/ejemplo-jquery-ajax-001.php",complete:function(resultado,configuracion){
003         $("#respuesta").text(resultado.responseText);
004 }});
005     $.ajax();
006 }
```

*ejemplo-jquery-ajax-005.js*

```
001 function iniciar(){
002     $.ajax("http://localhost/ejemplo-jquery-ajax-001.php",{complete:mostrar});
003 }
004 function mostrar(resultado,configuracion){
005     $("#respuesta").text(resultado.responseText);
006 }
```

*ejemplo-jquery-ajax-008.js*

```
001 function iniciar(){
002     $.ajax("http://localhost/ejemplo-jquery-ajax-001.php",{success:function(uno,estado,resultado){
003         $("#respuesta").text(resultado.responseText);
004 }});
005 }
```

*ejemplo-jquery-ajax-009.js*

```
001 $.ajax("http://localhost/ejemplos/ejemplo-009/ejemplo-jquery-ajax-009.php",
002 {success:function(uno,estado,resultado){
003 $("#respuesta").text(resultado.responseText);
004 }
005 , data:{"alumno":"JuanMateos","materia":"Sociales"}
006 }
007 );
```

*ejemplo-jquery-ajax-010.js*

```
001 $.ajax("http://localhost/ejemplos/ejemplo-009/ejemplo-jquery-ajax-010.php"
002 ,{success:function(uno,estado,resultado){
003 $("#respuesta").text(resultado.responseText);
004 }
005 , data:{"alumno":"JuanMateos","materia":"Sociales"}
006 ,method:"GET"
007 }
008 );
```

*ejemplo-jquery-ajax-012.js*

```
001 var datos="<datosalumnos><alumno><nombre>Juan Mateos</nombre><asignatura>Sociales</asignatura></alumno></datosalumnos>"
002 $.ajax("http://localhost/ejemplos/ejemplo-009/ejemplo-jquery-ajax-012.php",
003 {success:function(uno,estado,resultado){
004 $(uno).find("datosalumnos").each(function(){
005 $(this).find("alumno").each(function(){
006 $("#respuesta").text($(this).find("nota").text());
007 });
008 });
009 }
010 ,data:datos
011 ,method:"POST"
012 ,dataType:"xml"
```

```
013 }
014 );
```

*ejemplo-jquery-ajax-013.js*

```
    var datos=newObject();
    datos.nombre="Juan Mateos";
    datos.asignatura="Sociales";
    var datitos=JSON.stringify(datos);
    $.ajax("http://localhost/ejemplos/ejemplo-jquery-ajax-013.php",
        { success:function(uno,estado,resultado){
                var datos=JSON.parse(resultado.responseText);
                $("#respuesta").text(datos.calificacion);
            }
            ,data:datitos,method:"GET",dataType:"json"});
            ,method:"GET"
            ,dataType:"json"
        }
    );
```

Con $.ajax tambien se puede utilizar un objeto FormData para enviar datos al servidor, se enviarán mediante post y además deberemos poner los siguientes valores en la configuración **contentType** con el valor **false** y **processData** con el valor **false**.

*formdata/ajax03.js*

```
001 $(window).on("load",inicio)
002 function inicio(){
003     $("#poner").on("click",enviar);
004 }
005 function  enviar(){
006     let nom=$("#nombre").val();
007     let ape=$("#apellidos").val();
008     let datos= new FormData();
009     datos.append("nombre",nom);
010     datos.append("apellidos",ape);
011     $.ajax("php/procesar.php",{ method:"POST",
012                                 data:datos,
013                                 complete:muestra,
014                                 contentType: false,
015                                 processData: false
016                                 })
017 }
018 function muestra(resul){
019     $("#resultado").val(resul.responseText);
020 }
```

**$.ajaxComplete(*función*)**→Cada vez que se completa una solicitud Ajax, jQuery activa el evento ajaxComplete. Todos los controladores que se han registrado con el método .ajaxComplete () se ejecutan en este momento. Se invocan todos los controladores ajaxComplete, independientemente de qué solicitud de Ajax se completó. Si debe diferenciar entre las solicitudes, use los parámetros pasados al controlador. Cada vez que se ejecuta un controlador ajaxComplete, se pasa el objeto de evento, el objeto XMLHttpRequest y el objeto de configuración que se usó en la creación de la solicitud. Por ejemplo, puede restringir la devolución de llamada a solo manejar eventos relacionados con una URL en particular:

*ejemplo-jquery-ajax-001.js*

```
001 function iniciar(){
002     $.ajax("http://localhost/ejemplo-jquery-ajax-001.php");
003     $(document).ajaxComplete(function(evento,resultado,configuracion){
004         $("#respuesta").text(resultado.responseText);
005 });
006 }
```

*ejemplo-jquery-ajax-006.js*

```
001 function iniciar(){
002     $.ajax("http://localhost/ejemplo-jquery-ajax-001.php");
```

```
003     $(document).ajaxComplete(mostrar);
004 }
005 function mostrar(evento,resultado,configuracion){
006     $("#respuesta").text(resultado.responseText);
007 }
```

**$.ajaxSuccess(*función*)**→ función que se va a ejecutar cuando una solicitud tiene éxito.

ejemplo-jquery-ajax-007,js

```
001 function iniciar(){
002     $.ajax("http://localhost/ejemplo-jquery-ajax-001.php");
003     $(document).ajaxSuccess(function(evento,resultado,configuracion){
004         $("#respuesta").text(resultado.responseText);
005 });
006 }
```

**$.ajaxError(*función*)**→ Cada vez que una solicitud Ajax se completa con un error, jQuery activa el evento ajaxError. Todos y cada uno de los controladores que se han registrado con el método .ajaxError () se ejecutan en este momento. Nota: este controlador no se llama para secuencias de comandos de dominio cruzado y JSONP de dominio cruzado. Se invocan todos los controladores ajaxError, independientemente de qué solicitud de Ajax se completó. Para diferenciar entre las solicitudes, use los parámetros pasados al controlador. Cada vez que se ejecuta un controlador ajaxError, se pasa el objeto de evento, el objeto jqXHR (antes de jQuery 1.5, el objeto XHR) y el objeto de configuración que se utilizó en la creación de la solicitud. Cuando se produce un error de HTTP, el cuarto argumento (thrownError) recibe la parte textual del estado de HTTP, como "No encontrado" o "Error interno del servidor". Por ejemplo, para restringir la devolución de llamada de error a solo manejar eventos relacionados con una URL particular:

**$.ajaxSend(*función*)**→ cada vez que se encía una solicitud se va a ejecutar la función, que tiene los tres parámetros estándar.

**$.ajaxStart(*función*)**→ cada vez que se envía una solicitud y no hay ninguna otra solicitud pendiente, se ejecuta la función, que no tiene parámetros.

**$.ajaxStop(*función*)**→ cada vez que se termina una solicitud y no hay ninguna otra solicitud pendiente, se ejecuta la función, que no tiene parámetros.

**$.getJSON(*url* [,*datos*] [, *función*])**→ obtiene un dato en formato json, realiza una solicitud con get.

*ejemplo-jquery-ajax-020.js*

```
001 function iniciar(){
002     $.getJSON("libro.json",function(uno,estado,resultado){
003 var datos=JSON.parse(resultado.responseText);
004         $("#respuesta").text(datos.titulo+" "+datos.autor);
005 })
006 }
```

*ejemplo-jquery-ajax-021.js*

```
001 function iniciar(){
002 $.getJSON({url:"libro.json",success:function(uno,estado,resultado){
003 var datos=JSON.parse(resultado.responseText);
004         $("#respuesta").text(datos.titulo+" "+datos.autor);
005 }})
006 }
```

*ejemplo-jquery-ajax-022.js*

```
001 function iniciar(){
002     $.getJSON({url:"ejemplo-jquery-ajax-
```

```
022.php",dataType:"json",success:correcto,error:erroneo})
003 }
004 function correcto(uno,estado,resultado){
005 var datos=JSON.parse(resultado.responseText);
006         $("#respuesta").text(datos.titulo+" "+datos.autor);
007 }
008 functionerroneo(){
009     console.log("erorajaxjson");
010 }
```

**$.getScript( fichero [, función] )**

**$.getScript( fichero [, function ( valor-devuelto, texto-estado , objeto-httpRequest) {cuerpo-función}] )**

Lee un fichero de javascript para luego poder ejecutarlo.

**$.ajaxTransport(tipo-dato, function ( [objeto-configuración-devuelto, objeto-configuración-original, objeto-httpRequest ]){cuerpo-función})**

Crea un objeto que se encarga de la transmisión real de datos.

**$.ajaxPrefilter(tipo-dato, function ( [objeto-configuración-devuelto, objeto-configuración-original, objeto-httpRequest ]){cuerpo-función})**

Para modificar la configuración de la comunicación, antes de realizar la comunicación con ajax.