

Aplicaciones web dinámicas: PHP y Javascript.



Caso práctico

Juan y **Carlos** tienen definida la estructura de la aplicación web que deben desarrollar. Han llevado a cabo algunas pruebas de programación, y están contentos con los resultados obtenidos. Antes de seguir avanzando, deciden reunir al equipo de BK Programación para mostrarles los progresos realizados y el plan de desarrollo previsto.



Durante la presentación de la aplicación web, todo el equipo se muestra entusiasmado con el aspecto que está tomando el proyecto. **Ada**, la directora, les felicita por el trabajo que han llevado a cabo hasta el momento. **Ana**, que tiene experiencia como diseñadora gráfica, se ofrece a ayudarles con el aspecto visual del interfaz web. Pero de todas las opiniones, hay una que les llama la atención. **María**, que se encarga del mantenimiento de servidores y sitios web, les pregunta si **integrar en su aplicación algún tipo de código cliente**.

de su estructura para muy diversas funciones.

Juan y **Carlos** se miran, y saben que ambos están pensando lo mismo. Habrá que hacer un último esfuerzo e informarse sobre el tema. Si deciden que resulta interesante, aún están a tiempo de incorporarlo en su proyecto.

1.1.- Páginas web dinámicas.

La inclusión de código en páginas web para su ejecución por parte del navegador tiene ciertas limitaciones:

- ✓ Cuando ejecutas código PHP en un servidor, es normalmente el programador el que tiene el control sobre el entorno de ejecución. Al cliente llegan únicamente etiquetas en lenguaje HTML o XHTML. Sin embargo, cuando programas código para que se ejecute en un cliente web, no tienes siquiera la certeza de que el navegador del usuario soporte la ejecución del código que recibe. Existen ciertos sistemas, como dispositivos móviles o navegadores integrados en hardware específico, que no permiten la ejecución de código de cliente.
- ✓ El código que se ejecuta en el navegador está normalmente limitado a ser ejecutado en un entorno controlado, que no permite, por ejemplo, la lectura o escritura de ficheros en el ordenador del usuario. De esta forma se restringen los efectos negativos que pueda causar un guión y se favorece la confianza del usuario en este tipo de código.

Pese a estas limitaciones, la ejecución de código en el navegador encaja perfectamente con cierto tipo de tareas como:

- ✓ Comprobar y/o procesar los datos que introduce el usuario en los formularios, como paso previo a su envío al servidor web.
- ✓ Gestionar diferentes marcos y/o ventanas del navegador.
- ✓ Modificar de forma dinámica los elementos que componen la página web, ajustando sus propiedades o estilos en respuesta a la interacción del usuario.

El código JavaScript de una página se puede ejecutar en respuesta a eventos generados por el navegador. Por ejemplo, utilizando el evento onsubmit podemos llamar a una función validar_email para validar una dirección de correo introducida por el usuario cuando se intenta enviar el formulario:

```
<form action="usuario.php" method="get" name="datos_usuario"
onsubmit="return validar_email()">
<input type="text" id="email" /><br /><input type='submit' name='enviar' value='Enviar' />
</form>
```

Para la función que realiza la validación básica de una dirección de email puedes utilizar, por ejemplo, el siguiente código:

```
function validar_email() {  
    valor = document.getElementById("email").value;  
    pos_arroba = valor.indexOf("@");  
    pos_punto = valor.lastIndexOf(".");  
    if (pos_arroba < 1 || pos_punto < pos_arroba+2  
    || pos_punto+2>=valor.length) {  
        alert("Dirección de correo no válida.");  
        return false;  
    }  
    return true;  
}
```



Las páginas web que se aprovechan de las capacidades de ejecución de código en el cliente para cambiar su apariencia, o su funcionamiento, se conocen como páginas web dinámicas. Se llama **HTML dinámico** (DHTML) al conjunto de técnicas que emplean HTML, el modelo de objetos del documento web  modelo de objetos del documento web (DOM), hojas de estilo CSS y lenguaje ejecutado en el navegador para crear sitios webs dinámicos.

1.2.- El lenguaje JavaScript.

El lenguaje de guiones que se utiliza mayoritariamente hoy en día para la programación de clientes webes JavaScript. Su sintaxis está basada en la del lenguaje C, parecida a la que conocemos del lenguaje PHP. Aunque su utilización principal es incorporarlo a páginas web, también puedes encontrar **JavaScript** en otros lugares como en documentos **PDF**, o para definir la funcionalidad de extensiones de escritorio o de algunas aplicaciones ( widgets).

Si bien, la gran mayoría de navegadores web soportan código en lenguaje JavaScript, debes tener en cuenta que:

- ✓ La ejecución de JavaScript en el navegador puede haber sido deshabilitada por el usuario.
- ✓ La implementación de JavaScript puede variar de un navegador a otro. Lo mismo sucede con el interface de programación que usa JavaScript para acceder a la estructura de las páginas web: el DOM. Por este motivo, es conveniente que verifiques la funcionalidad del código en diversos navegadores antes de publicarlo como parte de tu sitio web.

Se conoce como motor JavaScript a la parte del navegador encargada de interpretar y ejecutar el código JavaScript que forma parte de las páginas web. Los motores JavaScript que se incluyen en los navegadores han experimentado una importante mejora de rendimiento en los últimos tiempos. Existen pruebas específicas destinadas a medir la velocidad de ejecución de distintos motores JavaScript.

Velocidad de ejecución de distintos motores JavaScript.

Aunque no vamos a aprender en esta unidad a programar en JavaScript, deberías saber cómo depurar el código que vamos a utilizar. Es conveniente que manejes un depurador para cada navegador que utilices, pero para esta unidad llegará con la **extensión** Firebug para el navegador Firefox.

Extensión Firebug.



Para ver cómo instalar y utilizar Firebug, puedes consultar el siguiente video tutorial.

Autoevaluación

¿Cuál es una de las principales desventajas de la programación del cliente web?

- Que no es posible asegurar que el navegador vaya a ejecutar el código.
- Que el navegador puede no ser capaz de mostrar correctamente la página al confundir el código con las etiquetas HTML / XHTML.

1.3.- Comunicación asíncrona con el servidor web: AJAX.

Una de las principales causas de la evolución de JavaScript en los últimos tiempos es, sin duda, la tecnología **AJAX**. Como ya vimos en la primera unidad, el término AJAX hace referencia a la posibilidad de una página web de establecer una comunicación con un servidor web y recibir una respuesta sin necesidad de que el navegador recargue la página.

AJAX utiliza el objeto XMLHttpRequest, creado originariamente por Microsoft como parte de su librería MSXML, y que hoy en día se ha incorporado de forma nativa a todos los navegadores actuales.

AJAX

Pese al nombre del objeto, y a que la letra X de las siglas AJAX hace referencia a XML, la información que se transmite de forma asíncrona entre el navegador y el servidor web no es necesario que se encuentre en formato XML.

Entre las tareas que puedes llevar a cabo gracias a AJAX están:

- ✓ Actualizar el contenido de una página web sin necesidad de recargarla.
- ✓ Pedir y recibir información desde un servidor web manteniendo la página cargada en el navegador.
- ✓ Enviar información de la página a un servidor web en segundo plano.

Dependiendo del navegador del usuario, y de si utiliza una versión antigua o moderna, tendrás que usar un método u otro para crear el objeto XMLHttpRequest:

```
// Distintas formas para crear el objeto  
// XMLHttpRequest según el navegador  
xmlhttp = new ActiveXObject("Msxml2.XMLHTTP");  
xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");  
xmlhttp = new XMLHttpRequest();
```

Es recomendable ver [este ejemplo](#) de cómo comprobar si se están usando navegadores antiguos (como IE5 y IE6) y crear el objeto de manera acorde.

Afortunadamente, muchas de las librerías JavaScript de las que hablábamos antes soportan también AJAX, y utilizan un código adaptado según el navegador del usuario. Si utilizas una de estas librerías podrás ahorrarte muchos quebraderos de cabeza al programar. Por ejemplo, si utilizas jQuery podrías utilizar AJAX para enviar en segundo plano el email validado en el ejemplo anterior. Simplemente tendrías que incluir en el HTML la librería jQuery.

```
// Utilizamos la versión de jQuery disponible en las CDN de Google
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js"></script>
```

Y tras el código de validación, debes ejecutar la función ajax incluida en la librería jQuery:

```
function validar_email() {
    valor = document.getElementById("email").value;

    // Aquí iría el código de validación

    $.ajax({
        type: "POST", url: "email.php", data: "email=" + valor,
        statusCode: {
            404: function() { alert('Página no encontrada'); }
        },
        success: function(result) { alert( "Resultado: " + result ); }
    });

    return false;
}
```

La función anterior envía la dirección de email introducida mediante POST a la página email.php, y muestra un mensaje con el resultado obtenido.



Recomendación

Para trabajar con una nueva versión actualizada de jQuery se puede descargar desde Google o Microsoft cómo se indica [aquí](#). Si se descarga desde Google o Microsoft tienes la ventaja de quitar carga a tu servidor web y aprovechar sus capacidades de CDN (Content Delivery Networks) para servir el contenido desde el servidor más cercano al cliente. Además, es posible que el cliente ya lo tenga en su caché si desde otra web ha usado este mismo jQuery con la consiguiente mejora de velocidad. Aunque para desarrollar es mejor guardar una copia en local porque siempre será más rápido.

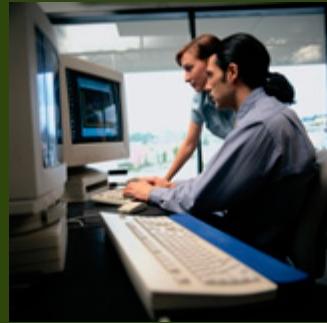
2.- PHP y JavaScript.



Caso práctico

Pasados unos días, **Juan** y **Carlos** se vuelven a reunir y deciden que en el diseño de su aplicación existen muchos lugares en los que podrían aprovechar las capacidades que ofrece la programación del navegador web, concretamente el lenguaje JavaScript.

Sin embargo, ambos siguen sin tener claro cómo pueden integrar el código del cliente web en una aplicación programada en lenguaje PHP. Animados por **María**, que les orienta sobre el rumbo que deben tomar, preparan una serie de pruebas de programación que les puedan orientar sobre el tema.



Si sabes programar aplicaciones que se ejecuten en un servidor web (con un lenguaje como PHP) y en el navegador del usuario (con JavaScript/jQuery), tienes en tu mano las herramientas necesarias para construir aplicaciones web completas. Sin embargo, es necesario que antes de comenzar tengas claras las funcionalidades que soporta cada una de estas tecnologías de desarrollo, y cómo puedes hacer para utilizar ambas a la vez.

Llevándolo a los extremos, podrías hacer aplicaciones en PHP que utilicen programación del cliente simplemente para tareas sencillas como verificar campos en los formularios antes de enviarlos. O, por el contrario, sería también posible programar aplicaciones completas que se ejecuten en el navegador del usuario, dejando el lenguaje del servidor para proporcionar ciertas funcionalidades como almacenamiento en bases de datos.

Una alternativa no es necesariamente mejor que la otra. Es necesario analizar de forma independiente la lógica de cada aplicación, de manera que no se malgasten los recursos del servidor realizando tareas que podrían trasladarse al cliente web. En ocasiones también es necesario comprobar los tiempos de carga de las páginas y el tamaño de las mismas. Puede ser preferible utilizar AJAX para, por ejemplo, enviar nuevos registros al servidor, si el esfuerzo que invertimos redonda en un interfaz de usuario más ágil y usable. En cualquier caso, la consistencia y robustez de la aplicación no debe verse afectada.

Si decides unir en una aplicación programación del cliente web con programación del servidor web, habrá ocasiones en que necesites comunicar ambos lenguajes. En el ejemplo anterior ya has comprobado cómo puedes hacer para pasar un valor o una variable JavaScript desde el navegador web a un guión en PHP: enviándolo como parámetro POST o GET en una petición de nueva página, bien sea al cargarla en el navegador o utilizando AJAX en una comunicación asíncrona:

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Desarrollo Web</title>
  </head>
  <body>
    <script type="text/javascript">
      <?php
        // Creamos en la página un código JavaScript que
        // utiliza la variable PHP "$email"
        $email = "alumno@educacion.es";
        print 'window.open("email.php?email=' . urlencode($email) . ')';
      ?>
    </script>
  </body>
</html>
```

En ambos casos deberás asegurarte de que las cadenas que insertes en las direcciones URL no incluyan caracteres inválidos. Para evitarlo, en PHP puedes usar la función `urlencode`, y en JavaScript `encodeURIComponent`.

Autoevaluación

¿Cómo es posible obtener en PHP el contenido de una variable JavaScript?

- En una petición de nueva página, como parámetro POST o GET.
- Enviando desde el servidor web una solicitud al navegador web.

2.1.- Aplicaciones web con PHP y JavaScript.

Con lo que has visto hasta el momento, seguramente, ya te has hecho una idea de qué tareas son las que con más frecuencia hacen uso de JavaScript. Y si hay una que destaca sobre el resto es, sin duda, la validación de formularios, que vamos a utilizar en los ejemplos que se desarrollan a lo largo de la presente unidad.

Como ya sabes, el mecanismo que utilizan las aplicaciones web para permitir al usuario la entrada de información es el formulario HTML. Los datos que se introducen en un formulario web se envían al servidor utilizando bien el método POST, bien el método GET. Muchos de los campos de los formularios tienen, normalmente, restricciones sobre el contenido que se debe llenar. La validación de un formulario web consiste en comprobar que el contenido introducido en todos los componentes del mismo cumpla estas restricciones.

Si no utilizas código ejecutable en el navegador, la única forma de validar un formulario consiste en enviarlo al servidor web para comprobar si existen errores en los datos. En caso de que así sea, habrá que volver a enviar el formulario al navegador del usuario mostrando las advertencias oportunas.

Obviamente, si utilizas JavaScript en tus aplicaciones, la validación se puede realizar en el cliente web. De esta forma el proceso es mucho más rápido. No es necesario enviar la información al servidor hasta que se haya comprobado que no existen errores de validación.

Sin embargo, aunque parecen claras las ventajas de la validación de formularios en el cliente web, hay ocasiones en las que ésta no es posible. Ya vimos que no siempre podrás asegurar que el navegador que utiliza el usuario tiene capacidad para ejecutar código JavaScript, o incluso puede suceder que se haya deshabilitado por motivos de seguridad.

En los casos que no sea posible asegurar la capacidad de ejecución de código de los clientes, la solución óptima sería utilizar un escenario dual: diseñar las aplicaciones suponiendo que es posible ejecutar JavaScript en los clientes, y al mismo tiempo prever la posibilidad de que no sea así. Por ejemplo, en el caso de la validación del formulario, podrías crear el código JavaScript de validación y, si en algún cliente este código no se puede ejecutar, preparar un código PHP similar que realice la validación en el servidor.

Por ejemplo, supongamos que queremos validar los datos que se introducen en el siguiente formulario web:

Introducción de datos

Nombre:

Contraseña:

Repita la contraseña:

Email:



Si realizas la validación en PHP, podrías generar junto con la página web el texto con las advertencias de validación. Y si el formulario lo quieres validar también utilizando JavaScript, tendrás que crear los mismos textos o similares. Una posibilidad para no repetir el código que introduce esos textos en el cliente y en el servidor, es introducir los textos de validación en las etiquetas HTML de la página web, y utilizar estilos para mostrarlos, o no, según sea oportuno.

Por ejemplo, para realizar la validación del formulario web anterior, puedes crear los siguientes textos en HTML (asociados al nombre, las contraseñas y la dirección de correo respectivamente):

```
<span class='error'>Debe tener más de 3 caracteres.</span>
<span class='error'>Debe ser mayor de 5 caracteres o no coinciden.</span>
<span class='error'>La dirección de email no es válida.</span>
```

El código PHP y JavaScript deberá ocultar cada uno de los textos cuando la validación de su elemento respectivo sea correcta. Por ejemplo, si el nombre tiene más de tres letras, validará correctamente y no se deberá mostrar el primer mensaje.

2.2.- Aplicaciones web con PHP y JavaScript (II).

Para realizar la validación del formulario en el servidor web utilizando PHP, necesitarás utilizar las siguientes funciones o similares:

```
<?php  
function validarNombre($nombre){  
    if(strlen($nombre) < 4) return false;  
    return true;  
}  
  
function validarEmail($email){  
    return preg_match("/^@[a-z0-9]+([_\\.-][a-z0-9]+)*@[([a-z0-9]+([_\\.-][a-z0-9]+)*)+\\.[a-z]{2,}$/i", $email);  
}  
  
function validarPasswords($pass1, $pass2) {  
    return $pass1 == $pass2 && strlen($pass1) > 5;  
}  
  
function validar($nombre, $email, $pass1, $pass2) {  
    return validarNombre($nombre) && validarEmail($email)  
        && validarPasswords($pass1, $pass2);  
}  
?>
```

Fíjate en el uso de la función `preg_match` y de  [expresiones regulares](#) para validar la dirección de correo.

[Función preg_match](#)

Ten en cuenta que las barras invertidas (\) tienen un significado especial dentro de una cadena de PHP (sirven para escapar el siguiente carácter, por ejemplo por si queremos mostrar unas comillas); por ese motivo, la doble barra \\ se convierte en una barra simple \ cuando se interpreta la expresión regular.



Debes conocer

En ocasiones es importante saber construir expresiones regulares en PHP para realizar comparación de cadenas de texto con patrones. Tienes más información en el siguiente enlace.

[Expresiones regulares en PHP.](#)

Con las funciones anteriores, puedes crear código en PHP que oculte los mensajes de validación cuando no sean necesarios:

```
<span id='errorNombre' for='nombre' class='<?php if(!isset($_POST['enviar']) || validarNombre($_POST['nombre'])) echo "oculto"; ?>'>
<span id='errorPassword' for='password' class='<?php if(!isset($_POST['enviar']) || validarPasswords($_POST['password1'], $_POST['password2'])) echo "oculto"; ?>'>
<span id='errorEmail' for='email' class='<?php if(!isset($_POST['enviar']) || validarEmail($_POST['email'])) echo "oculto "; ?>'>
```

El código anterior aplica la clase oculto a los textos de validación que no sea necesario mostrar. En la hoja de estilos correspondientes, deberás definir para esa clase un estilo como `display:none` o `visibility: hidden`.

Autoevaluación

¿Qué significa la siguiente expresión regular: \.[a-z]{2,}?

- Una barra seguida por un carácter cualquier, una letra entre la a y la z, y un número mayor o igual a 2.
- Un punto seguido de 2 o más letras minúsculas.

Introducción de datos

Nombre:
Alumno

Contraseña:
Debe ser menor de 255 caracteres y no contener:

Repita la contraseña:

Email:
La dirección de mail no es valida.



Revisa el código de validación PHP resultante.

 [Código de validación PHP](#) (1.94 KB)

2.3.- Aplicaciones web con PHP y JavaScript (III).

Partiendo de la página PHP anterior, que ya incluye código para validar el formulario web en el servidor, vas a ver cómo puedes hacer para incorporarle código en JavaScript que realice la misma validación en el cliente. De esta forma, si el navegador del usuario soporta JavaScript, se reducirá el procesamiento del servidor y la transferencia de información entre éste y el cliente.

Crearás todo el código necesario en un archivo externo que llamaremos validar.js. Como vamos a utilizar la librería jQuery, tendrás que añadir las dos líneas siguientes a la página anterior:

```
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
<script type="text/javascript" src="validar.js"></script>
```

En el código JavaScript habrá que definir unas funciones de validación similares a las programadas anteriormente en PHP:

```
function validarNombre(){
    if(nombre.val().length < 4) {
        errorNombre.removeClass("oculto");
        return false;
    }
    errorNombre.addClass("oculto");
    return true;
}

function validarEmail(){
    ...
}

function validarPasswords(){
    ...
}

function validar(){
    return validarNombre() & validarEmail() & validarPasswords();
}
```

En JavaScript usamos la función match para validar las direcciones de email. Las expresiones regulares que admite esta función no son exactamente iguales a las que viste anteriormente para la función preg_match de PHP.



Las funciones usan los métodos de jQuery addClass y removeClass para incorporar, y quitar, respectivamente, los textos de validación a la clase oculto, lo mismo que hacía el código PHP anterior. Para seleccionar los elementos a ocultar, puedes utilizar también jQuery. Por ejemplo, para seleccionar el elemento de la página web con identificador nombre, se pone:

```
var nombre = $("#nombre");
```

Como no quieres que el formulario se envíe realmente, captura el evento submit del formulario forzando a que se valide con la función de JavaScript. Si la validación es correcta (return true;), el formulario finalmente se envía.

```
$("#datos").submit(function(){
    if(validar()) return true;
    else return false;
});
```

El código final del ejemplo realiza la validación tanto en el cliente (si éste soporta JavaScript) como en el servidor web (si no soporta JavaScript). Puedes probar a deshabilitar la ejecución de JavaScript en el navegador para comprobar que el formulario se sigue validando.

[Código final.](#) (2.48 KB)

3.- Utilización de AJAX con PHP.



Caso práctico

En los últimos días **Juan** y **Carlos** han logrado grandes avances. Tienen claras las capacidades que les ofrece la programación del cliente web. Han profundizado en la tecnología AJAX. Y saben, incluso, de qué manera pueden integrar en una misma aplicación ambos tipos de programación: la programación del servidor web en lenguaje PHP, y la programación del cliente web en lenguaje JavaScript.



Sólo les falta un detalle: las herramientas. Saben lo qué tienen que hacer, pero antes de ponerse manos a la obra, deben decidir de qué forma hacerlo. **María** les ha informado de que existen multitud de librerías y herramientas que se pueden utilizar para agilizar la programación de aplicaciones en lenguaje JavaScript. Su último paso será decidir qué librerías y herramientas utilizarán para facilitar la programación de la aplicación web, tomando como punto de partida que sería preferible utilizar un único lenguaje en todo el proyecto para evitar la fragmentación del código de la misma.

Como ya sabes, la tecnología AJAX se utiliza desde el cliente web para permitir comunicaciones asíncronas con el servidor web, sin necesidad de recargar la página web que se muestra en el navegador. Se basa en la utilización de código en lenguaje JavaScript. Por tanto, te estarás preguntando, ¿qué tiene que ver la tecnología AJAX con el lenguaje PHP?

Acabamos de ver cómo se pueden crear aplicaciones web que utilicen de forma simultánea la programación del cliente web (con JavaScript) y la programación del servidor web (con PHP). En el procedimiento seguido en el ejemplo anterior, ambas tecnologías coexistían en una misma página, pero su programación era independiente. El código PHP se ejecutaba en el servidor, y en la misma página se incluían también guiones en lenguaje JavaScript que se ejecutaban en el navegador. En nuestro ejemplo solamente existía una relación: si el navegador permitía la ejecución de código JavaScript, se deshabilitaba el envío del formulario y la validación se realizaba en el cliente. En este caso no se llegaba a enviar la página al servidor y no se ejecutaba por tanto el código PHP de validación.

Si vas a usar aplicaciones que utilicen ambos lenguajes, es preferible tener un mecanismo mejor para integrarlos. Afortunadamente existen librerías para PHP que te permiten aprovechar las capacidades de JavaScript utilizando casi exclusivamente código en lenguaje PHP. Estas librerías definen una serie de objetos que puedes utilizar en el código de servidor, y que generan de forma automática código JavaScript en las páginas web que se envían al cliente.

La mayoría de estas librerías añaden a las aplicaciones web funcionalidades de la tecnología AJAX. Esto es: permiten crear páginas PHP que, tras ejecutarse en el servidor, producen páginas web que incorporan código JavaScript con funcionalidades AJAX. El mecanismo de funcionamiento lo puedes observar en el siguiente diagrama.



Muchas de estas librerías suelen apoyarse en librerías JavaScript como jQuery para la ejecución de código en el cliente. En Internet puedes encontrar información sobre librerías PHP con soporte para AJAX.

A continuación vas a aprender a utilizar dos de estas librerías: **Xajax** y **jQuery4PHP**.

Autoevaluación

¿Qué es jQuery?

- Una librería de programación para PHP.
- Una librería de programación para JavaScript.

3.1.- Xajax.

Xajax es una librería PHP de código abierto que permite generar aplicaciones web con tecnología AJAX. Facilita la utilización desde el cliente de funciones existentes en el servidor. Al utilizar los objetos AJAX en el código PHP, las páginas HTML que se obtienen incorporan el código JavaScript necesario para realizar las llamadas al servidor mediante AJAX. En la página web del proyecto tienes información disponible sobre su utilización.

Página web del proyecto

Para poder utilizar AJAX, descárgate la última versión de la librería desde la sección de descargas de su página web. De las carpetas que contienen los ficheros comprimidos, necesitas el contenido de `xajax_core` y `xajax_js`. Cópialas a una ruta de tu servidor web en la que sean accesibles por tus aplicaciones web.

Sección de descargas

En las **páginas PHP en que quieras utilizar AJAX**, deberás incluir la librería escribiendo el siguiente código:

```
require_once("xajax_core/xajax.inc.php");
```

Asegúrate de que la ruta a la librería sea la correcta. Lo siguiente es crear un objeto de la clase `xajax`, indicando como parámetro el script PHP que contiene las funciones a las que se podrán realizar llamadas mediante AJAX. Puedes incluir estas funciones en una página aparte o en la misma página PHP, y en este caso no será necesario indicar ningún parámetro:

Clase xajax

```
$xajax = new xajax();
```

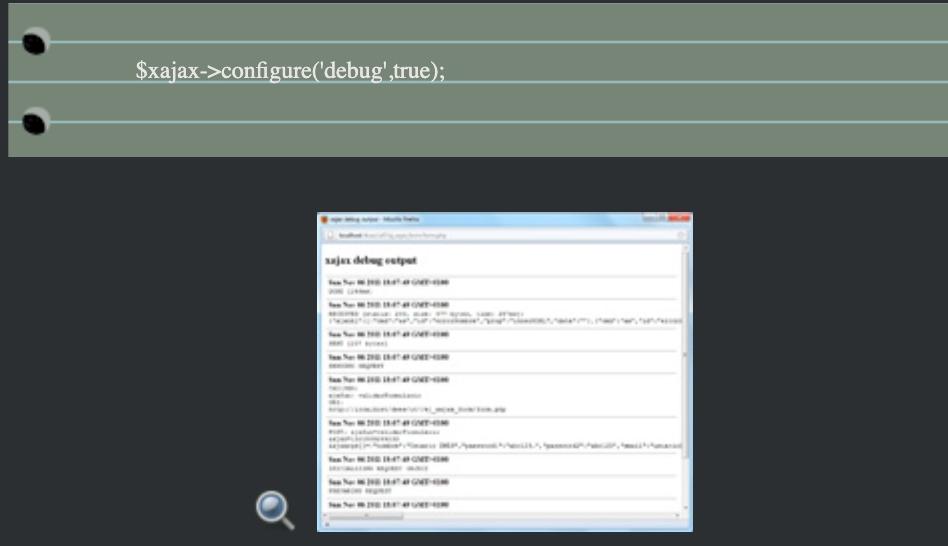
AJAX necesita incluir en la página web que se envía al navegador su propio código JavaScript. Para ello, tienes que incluir en tu código PHP la siguiente llamada al método `printJavascript()` del objeto `$xajax`:

```
$xajax->printJavascript();
```

En caso necesario, también deberás configurar la ruta de acceso a la carpeta `xajax_js` que contiene el código JavaScript de la librería (usa tu propia ruta como segundo parámetro):

```
$xajax->configure('javascript URI','./libs/');
```

Existen otras opciones de configuración de Xajax. Por ejemplo, cuando las cosas no funcionan como deberían, es muy interesante la opción que activa los mensajes de depuración:



Y por último, tienes que utilizar el método register para registrar cada una de las funciones PHP del servidor que estarán disponibles para ser ejecutadas de forma asíncrona desde el navegador:

```
$xajax->register(XAJAX_FUNCTION,"funcion1");
$xajax->register(XAJAX_FUNCTION,"funcion2");
...
```

En el **guión PHP en que definas las funciones** (si no es la misma página que la anterior), tienes que incluir la librería, crear el objeto y registrar las funciones de la misma forma que hiciste antes:

```
require_once("xajax_core/xajax.inc.php");
$xajax = new xajax();
$xajax->register(XAJAX_FUNCTION,"funcion1");
$xajax->register(XAJAX_FUNCTION,"funcion2");
```

Al registrar una función, web crea automáticamente una funciónJavaScript en el documento HTML con su mismo nombre prefijado por "xajax_". En el caso anterior, se crearán las funciones `xajax_funcion1` y `xajax_funcion2`.

Además deberás utilizar el método processRequest, que es el encargado de procesar las llamadas que reciba la página.

```
$xajax->processRequest();
```

Es importante tener en cuenta que la llamada a processRequest debe realizarse antes de que el guión PHP genere ningún tipo de salida.

En cada una de las funciones que definas, podrás instanciar y utilizar un objeto de la clase xajaxResponse para devolver al navegador los comandos resultado del procesamiento:

[Clase xajaxResponse](#)

```
function funcion1($a)
{
    $respuesta = new xajaxResponse();
    ...
    return $respuesta;
}
```

3.2.- Xajax (II).

Vamos a ver un ejemplo de programación que utilice la libreríajax. Partiendo del formulario web que utilizaste anteriormente, veremos el códigoxajax necesario para utilizar AJAX en su validación. La idea es crear una función PHP, que llamaremos validarFormulario, que reciba los datos del formulario y realice su validación. La página web, al pulsar el botón de enviar del formulario, utilizará AJAX para llamar a esta función y mostrar los errores de validación que devuelva.



En este ejemplo, todo el código va a compartir la misma página, incluyendo las funciones que se ejecutarán mediante AJAX. El primer paso es incluir la libreríajax, y definir la función de validación, que aprovechará el código de validación PHP que creaste antes:

```
require_once("xajax_core/xajax.inc.php");

function validarFormulario($valores) {
    $respuesta = new xajaxResponse();
    $error = false;

    if (!validarNombre($valores['nombre'])) {
        $respuesta->assign("errorNombre", "innerHTML",
        "El nombre debe tener más de 3 caracteres.");
        $error = true;
    }
    else $respuesta->clear("errorNombre", "innerHTML");

    // Validamos también las contraseñas y el email
    ...

    if (!$error) $respuesta->alert("Todo correcto.");

    $respuesta->assign("enviar","value","Enviar");
    $respuesta->assign("enviar","disabled",false);
    return $respuesta;
}
```

Como ves, si el nombre no valida, se utiliza el método assign de xajaxResponse para asignar al elemento de id=errorNombre (un span que ya está creado) el mensaje de error correspondiente. Si el nombre es válido, se vacía el contenido de ese elemento utilizando clear (por si se estuviera mostrando algún mensaje de error anterior). Este procedimiento tendrás que realizarlo para los tres elementos que debes validar en el formulario.

Si no se produce ningún error de validación, se muestra un mensaje de información de que todo está correcto. Además, se utiliza assign para volver a habilitar el botón de envío del formulario, y restaurar su etiqueta a Enviar (ahora vemos en qué otro lugar se cambia).

Como ya viste anteriormente, antes de crear el contenido HTML de la página, deberás incluir las siguientes sentencias PHP:

```
$xajax = new xajax();
$xajax->register(XAJAX_FUNCTION,"validarFormulario");
$xajax->processRequest();
```

Autoevaluación

¿En qué páginas debes crear un objeto de la clase xajax?

- En las páginas que vayan a utilizar AJAX para realizar peticiones a otras páginas PHP del servidor
- En las páginas que vayan a utilizar AJAX para realizar peticiones a otras páginas PHP del servidor, y en las páginas del servidor que vayan a recibir dichas peticiones.

3.3.- Xajax (III).

Para ejecutar el código de validación cuando se envíe el formulario deberás crear una función JavaScript y asignarla al evento onsubmit del formulario.

```
<form id='datos' action="javascript:void(null);" onsubmit="enviarFormulario();">
```

La función JavaScript enviarFormulario utilizará la función xajax_validarFormulario que ha sido creada automáticamente por Xajax al registrar la función correspondiente. Además se encargará de deshabilitar el botón de envío y de cambiar el texto que muestra:

```
xajax.$('enviar').disabled=true;  
xajax.$('enviar').value="Un momento...";  
xajax_validarFormulario (xajax.getFormValues("datos"));
```

Este método de llamar a una función registrada se conoce como asíncrono. Una vez realizada la llamada a la función, el procesamiento del formulario continúa sin esperar a recibir una respuesta. Es por este motivo que se deshabilita el botón de enviar el formulario. Cuando se recibe la respuesta de la función, se producen en el formulario los cambios que se indican en la misma.

Existe en Xajax otro método de realizar llamadas síncronas a una función registrada: el método JavaScript xajax.request:

```
respuesta = xajax.request({xjfun:"validarFormulario"}, {mode:'synchronous', parameters: ["valor1", "valor2", ...]})
```

La función PHP a la que se realiza la llamada, debe recibir tantos parámetros como se pasan. Para indicar el valor que se devuelve, puede usar el método setReturnValue() de la clase xajaxResponse.

```
$respuesta = new xajaxResponse();  
...  
$respuesta->setReturnValue("valorDevuelto");  
return $respuesta;
```

Fíjate que para realizar estas acciones se utiliza el objetoJavaScript xajax. Su método getFormValues se encarga de obtener los datos de un formulario.

[Clase JavaScript xajax](#)

Para acceder a los elementos de una página HTML mediante JavaScript mediante su atributo `id`, se puede usar:

```
objUsuario = document.getElementById("usuario");
```

Y si quieres conocer su valor:

```
valorUsuario = objUsuario.value;
```

Si utilizas Firebug puedes comprobar que los parámetros se envían mediante POST.



Es importante que tengas clara la estructura en clases de la librería Xajax, y sepas diferenciar entre las clases PHP y las clases JavaScript de la misma. Las principales clases son las que has visto en el ejemplo anterior: `xajax` y `xajaxResponse` en cuanto a clases PHP, y `xajax` en cuanto a JavaScript.

3.4.- Xajax (IV).

Esta función, que se encargará de validar el formulario, puedes crearla en el mismo fichero PHP o en un fichero aparte. Deberás incluir también el código necesario para Xajax:

```
<head>
...
<?php $xajax->printJavascript(); ?>
<script type="text/javascript" src="validar.js"></script>
</head>
```

Puedes descargar y comprobar el código completo de la página, incluyendo el código JavaScript y los estilos.

 [Código completo de la página](#) (0.12 MB)

Asegúrate de incluir la librería y ajustar las rutas en el código. Si tienes problemas, puedes utilizar Firebug para comprobar que los cambios que realiza Xajax en el HTML son los adecuados.



Autoevaluación

Utilizando Xajax, ¿cómo debes hacer para que la página web espere por la respuesta de una petición AJAX?

- Llamando a la función JavaScript que crea Xajax cuando registras una función del servidor (su nombre comienza por xajax_).
- Mediante el método request, indicando como parámetro mode:'synchronous'.



Ejercicio resuelto

Partiendo de la aplicación de tienda online que programaste en unidades anteriores, utiliza Xajax para cambiar el mecanismo de login. Se trata de crear una función en PHP de nombre validarLogin, que reciba como parámetros un nombre y contraseña, y que compruebe estas credenciales con la base de datos y devuelva false si no son correctas o true si son válidas. En este caso, deberá encargarse, también, de almacenar el nombre del usuario en una variable de sesión.

[Aplicación de tienda online](#) (9.00 KB)

Modifica la página login.php para que, utilizando Xajax, haga una llamada a la función validarLogin cuando se pulsa el botón Enviar del formulario. Si la función devuelve false, mostrará un mensaje que indique que las credenciales son incorrectas. En caso contrario, cargará la página productos.php.

[Mostrar retroalimentación](#)

3.5.- JQuery4PHP.

La otra librería con la que vas a trabajar se llama **JQuery4PHP**. Su objetivo es proporcionar un interface de programación en PHP que aproveche las capacidades de la librería de JavaScript **jQuery**. Es de código abierto, disponible bajo licencias [MIT](#) y [GPLv2](#), y funciona únicamente con las últimas versiones de PHP (a partir de PHP5).



[jQuery4PHP](#)

Al igual que con XAJAX, para poder utilizarla en tus páginas simplemente descárgate la última versión desde la página de descargas, y extrae del fichero comprimido la carpeta **YepSua**. Cópiala en un lugar accesible por el servidor web y tus aplicaciones.

[Página de descargas](#)

En las **páginas PHP** en que quieras utilizar **jQuery4PHP**, deberás incluir la librería escribiendo el siguiente código:

```
<?php
require_once("YepSua/Labs/RIA/jQuery4PHP/YsJQueryAutoloader.php");
YsJQueryAutoloader::register();
?>
```

Asegúrate de que la ruta a la librería sea la correcta.

El código anterior ejecuta un método estático de la clase **YsJQueryAutoloader** que se encarga de incluir todos los ficheros necesarios para la librería en el código de tu página.

Para acceder a las capacidades de **jQuery4PHP**, deberás utilizar en tu código PHP la clase **YsJQuery**.

Antes viste que con XAJAX necesitabas ejecutar en el servidor el método **printJavascript**, para que incluyese su propio código JavaScript en la página que se envía al cliente. La librería **jQuery4PHP** se apoya completamente en el código JavaScript de la librería **jQuery**. Por tanto, al escribir el código HTML de la página deberás asegurarte de que se incluye dicha librería, poniendo una línea como la siguiente:

```
<head>
    // Utilizamos la versión de jQuery disponible en las CDN de Google
    <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
    ...
</head>
```

En la documentación de la librería, y en los ejemplos que vas a programar a continuación, se utiliza una sintaxis peculiar de programación propia de PHP. Por ejemplo, comprueba el siguiente código:

```
<?php
$jq = YsJQuery::newInstance();
$jq->onClick();
$jq->in('#enviar');
$jq->execute('alert("Has pulsado el botón.")');
$jq->write();
?>
```

Como ves, se crea un nuevo objeto a partir del método estático newInstance, y a continuación se ejecutan ciertos métodos implementados en la clase YsJQuery. Estos métodos definen código JavaScript asociado al evento onClick del botón con id enviar. El código que se ejecutará cuando se pulse el botón se incluye dentro del método execute, y muestra un mensaje en pantalla. El último método (write) es el encargado de generar el código JavaScript en la página.

Ese mismo código se puede escribir de la forma siguiente, que es la que utilizaremos a continuación.

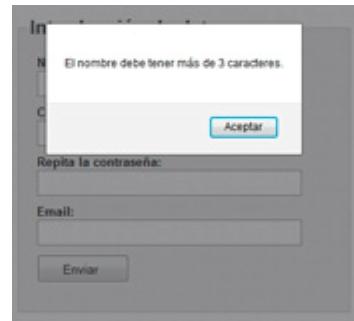
```
<?php
echo
YsJQuery::newInstance()
->onClick()
->in('#enviar')
->execute('alert("Has pulsado el botón.")')
?>
```

Fíjate que no se asigna nombre al objeto que se crea, pues no es necesario nombrarlo si los métodos se ejecutan justo a continuación de su instanciación. Además, se ha sustituido la llamada al método write por un comando echo al comienzo.

3.6.- JQuery4PHP (II).

Para comenzar a ver la librería, utilizaremos una versión simplificada del formulario de introducción de datos con el que has estado trabajando. El objetivo es el mismo, validar los datos introducidos por el usuario; pero en lugar de mostrar los errores de validación integrados en la página web, vas a emplear la función alert de JavaScript.

La página form.php es similar a la que has utilizado anteriormente, añadiéndole código para:



- Incluir y registrar la librería:

```
require_once("YepSua/Labs/RIA/jQuery4PHP/YsJQueryAutoloader.php");
YsJQueryAutoloader::register();
<br />
<br />
```

- Incluir también la librería de JavaScript jQuery:

```
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
```

Además, hay que capturar el evento onClick del formulario, e indicarle que envíe los datos del mismo a una página PHP de validación.

```

<?php
echo
YsJQuery::newInstance()
->onClick()
->in("#enviar")
->execute(
    YsJQuery::getJSON(
        "validar.php",
        YsJQuery::toArray()->in("#datos input"),
        new YsJsFunction(
            if(msg.errorNombre) alert(msg.errorNombre);
            if(msg.errorPassword) alert(msg.errorPassword);
            if(msg.errorEmail) alert(msg.errorEmail);'msg'
        )
    )
);
?>

```

En el código anterior, primero se asocia al evento onClick del botón enviar al código que se encuentra en la llamada a execute. Este código llama al guión validar.php, enviando como parámetros los datos del formulario (los que se han introducido en etiquetas de tipo input) convertidos a array y, una vez recibida la respuesta, ejecuta una función JavaScript. En esta función se usa la función alert de JavaScript para mostrar los errores de validación obtenidos.

Fíjate que para comunicarse con el servidor, se utiliza el método getJSON. Este método utiliza  notación JSON para transmitir la información con el servidor.

Autoevaluación

¿Cuál es la función del método write de la clase YsJQuery?

- Generar el código JavaScript necesario para que la página lleve a cabo las funciones que se han definido.
- Mostrar un texto en la página web generada, de forma similar al echo o print de PHP.

3.7.- JQuery4PHP (III).

Al ejecutar la página PHP que has programado, puedes observar que se genera el siguiente código JavaScript en la misma cuando se envía al navegador:

```
<script type="text/javascript" language="javascript">
/* <![CDATA[ */
jQuery('#enviar').click(function(){
    jQuery.getJSON('validar.php',
        jQuery('#datos input').toArray(),
        function(msg){
            if(msg.errorNombre) alert(msg.errorNombre);
            if(msg.errorPassword) alert(msg.errorPassword);
            if(msg.errorEmail) alert(msg.errorEmail);
        }
    )
})
/* ]]> */
</script>
```

Como ves, todo el código creado utilizando la librería jQuery4PHP se traduce en llamadas a la librería jQuery de JavaScript.

En la página PHP de validación, validar.php, puedes utilizar las mismas funciones de ejemplos anteriores y definir una nueva de nombre validarFormulario:

```
function validarFormulario($valores) {
    $respuesta = array();
    if (!validarNombre($valores['nombre']))
        $respuesta['errorNombre'] =
    "El nombre debe tener más de 3 caracteres.";

    // Validamos también las contraseñas y el email
    ...

    return $respuesta;
}

echo json_encode(validarFormulario($_REQUEST));
<br />
<br />
```



Utilizaremos la función json_encode de PHP para devolver los errores de validación con notación JSON. Revisa el código obtenido y comprueba su funcionamiento.

 [Código obtenido](#) (3.00 KB)

3.8.- JQuery4PHP (IV).

Una de las características que cabe destacar de jQuery4PHP es su extensibilidad. Existen varias extensiones que se integran con la librería y permiten realizar de forma sencilla tareas adicionales a las que soporta el núcleo de la misma.

Por ejemplo, si en la página anterior quisiéramos integrar en etiquetas los mensajes de validación, el código necesario sería más complejo. De hecho, ya has visto que para mostrar los mensajes de alerta has tenido que utilizar código JavaScript mezclado con el código PHP.



Vamos a ver cómo podemos utilizar la **extensión** JqValidate para realizar de forma mucho más sencilla y eficaz la validación del formulario anterior.

Extensión JqValidate

Para poder usar esta extensión en tus páginas has de:

- ✓ Indicar a la librería jQuery4PHP que vas a usar la extensión:

```
YsJQuery::usePlugin(YsJQueryConstant::PLUGIN_JQVALIDATE);
```

- ✓ Incluir el código JavaScript necesario por la extensión, que en este caso concreto se corresponde con la **extensión Validate de jQuery**:

```
<script type="text/javascript" src="http://ajax.aspnetcdn.com/ajax/jquery.validate/1.8.1/jquery.validate.min.js"></script>
```

- ✓ Cargar los mensajes de validación en idioma español (de no hacerlo, se mostrarán en inglés):

```
echo YsJQueryAssets::loadScripts('jq4php-showcase/showcase/jquery4php-assets/js/plugins/bassistance/validate/localiz
```

Como siempre, revisa las rutas del código anterior para ajustarlas a las de tu sistema.

3.9.- JQuery4PHP (V).

Al utilizar la extensión JqValidate para validar los datos introducidos en un formulario, el código de validación que se usará utilizará la extensión jQuery.Validate. No será necesario que programes los algoritmos de validación, sino que indiques qué reglas se deberán aplicar a cada uno de los campos del formulario. Esto se hace utilizando el método `_rules`.

Para indicar las reglas, se pasa como parámetro un array, que contendrá tantos elementos como campos a validar. Para cada uno de estos campos se crea un nuevo array, que contendrá las reglas de dicho campo. Cada una de las reglas se compone en base a los distintos métodos de validación que incorpora la extensión de JavaScript jQuery.Validate.



[Métodos de validación](#)

El código PHP necesario para validar nuestro formulario será:

```
<?php
echo
YsJQuery::newInstance()
->onClick()
->in("#enviar")
->execute(
    YsJQValidate::build()->in("#datos")
    ->_rules(array(
        'nombre' => array('required' => true, 'minlength' => 4),
        'email' => array('required' => true, 'email' => true),
        'password1' => array('required' => true,
        'minlength' => 6, 'equalTo' => '#password2')
    )
)
);
?>
```

Fíjate que la asociación del código con el evento onClick del botón se sigue haciendo como antes.

Si revisas el código JavaScript que se genera en la página HTML, observarás lo siguiente:

```
<script type="text/javascript" language="javascript">
/* <![CDATA[ */
jQuery('#enviar').click(function(){
jQuery('#datos').validate({ "rules": {
"nombre": {"required": true,"minlength": 4},
"email": {"required": true,"email": true},
"password1": {"required": true,"minlength": 6,
"equalTo": '#password2'}}})
})
/* ]]> */
</script>
```

Como ves, todo se sigue traduciendo en llamadas a la librería jQuery de JavaScript. Además, una de las grandes ventajas de este método es que una vez definidas las reglas de validación en PHP, todo el código que se ejecuta para verificarlas es JavaScript. Si el navegador soporta la ejecución de código en lenguaje JavaScript, no es necesario establecer ningún tipo de tráfico de validación con el servidor web.

Revisa el código obtenido y comprueba su funcionamiento.

 [Código obtenido](#) (2.00 KB)

Autoevaluación

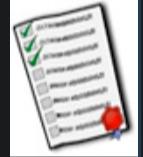
Al emplear la extensión JqValidate de jQuery4PHP, ¿qué código JavaScript deberás incluir en tus páginas?

- El correspondiente a la librería jQuery y a su extensión Validate.
- Únicamente el correspondiente a la libreríajQuery

Anexo.- Licencia de recursos.

Licencias de recursos utilizados en la Unidad de Tra

Recurso (1)	Datos del recurso (1)	Recurso (2)	Da
	Autoría: Stockbyte. Licencia: Uso educativo no comercial para plataformas públicas de Formación Profesional a distancia. Procedencia: CD-DVD Num. SD174.		Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.
	Autoría: Jed Schmidt. Licencia: Creative Commons By. Procedencia: http://www.flickr.com/photos/tr4nslator/5778135533/ .		Autoría: Stockbyte. Licencia: Uso educativo no comercial para plataformas públicas de Formación Profesional a distancia. Procedencia: CD-DVD Num. SD174.
	Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.		Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.
	Autoría: Matthew Batchelder. Licencia: Creative Commons By. Procedencia: http://www.flickr.com/photos/borkweb/285731517/ .		Autoría: Stockbyte. Licencia: Uso educativo no comercial para plataformas públicas de Formación Profesional a distancia. Procedencia: CD-DVD Num. SD174.
	Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.		Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.
	Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.		Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/ . Procedencia: Captura de una página web generada por Firefox.

	<p>Autoría: Omar Yepez. Licencia: GNU General Public License (GPL), MIT License. Procedencia: http://sourceforge.net/projects/jquery4php/.</p>		<p>Autoría: Navegador Firefox. Licencia: Mozilla Public License http://www.mozilla.org/MPL/. Procedencia: Captura de una página web generada por Firefox.</p>
	<p>Autoría: Stocklib. Licencia: Uso educativo no comercial para plataformas públicas de Formación Profesional a distancia. Procedencia: CD-DVD Num. ECD001.</p>		<p>Autoría: yves / Licencia: Dominio público Procedencia: http://www.operacionopen.com/list-by-yves_gutierrez</p>

Condiciones y términos de uso de los materiales

Materiales desarrollados inicialmente por el Ministerio de Educación, Cultura y Deporte y actualizados por el profesorado de la Junta de Andalucía bajo licencia Creative Commons BY-NC-SA.



Antes de cualquier uso leer detenidamente el siguiente [Aviso legal](#)

Histórico de actualizaciones

Versión: 01.00.04

Fecha subida: 20/02/18

Autoría: Jesús Manuel Marín
Navarro

Ubicación: 1.3

Mejora (tipo 1): añadida info sobre descarga de jQuery de CDNs de google y Microsoft. También referencia a cómo usar AJAX con navegadores antiguos IE5 y IE6

Ubicación: 3.0

Mejora (tipo 1): enlace roto: Librerías PHP con soporte para AJAX

Ajustes realizados en la versión

Gloria Ortiz

Mejora: Eliminación del idevice de licencia en la primera página.

Versión: 01.00.03

Fecha subida: 20/03/15

Autoría: Carlos Ríos Ruiz

Ubicación: Todo el contenido

Mejora (tipo 1): Corrección de alguna pequeña errata de escritura

Ubicación: Sección 3.6

Mejora (tipo 1): Cambio de la entrada de glosario Notación JSON por una definición.

Ubicación: Sección 3.3

Mejora (tipo 1): Enlace clase XAJAX lleva a un sitio incorrecto. Modificar para que vaya a la definición de la clase.

Ubicación: Sección 2.2

Mejora (tipo 1): Cambio de la entrada de glosario: expresión regular, por un elemento definición, ya que no funciona el enlace.

Ubicación: Sección 1.2

Mejora (tipo 1): Arreglo enlace a extensión firebug de Firefox e imagen inferior.

Ubicación: Sección 1.2

Mejora (tipo 1): Cambio de la entrada de glosario: Widget web por un elemento definición, ya que no funciona el enlace.

Ubicación: Sección 1.1

Mejora (tipo 1): Cambio de la entrada de glosario: DOM: modelo de objetos del documento web por un elemento definición, ya que no funciona el enlace.

Ubicación: Sección 1.

Mejora (tipo 1): Entrada de glosario: sección CDATA la cambio por definición, ya que el enlace al glosario no funcionaba.

Versión: 01.00.02

Fecha subida:
19/03/15

**Autoría: Manuel Alberto Domínguez
Vega**

Mejora: Pequeñas erratas, enlaces rotos, glosario en formato nuevo y credenciales en imágenes

(S)

Versión: 01.00.01

Fecha subida: 20/04/14

Autoría: Sonia Amate Garrido

Mejora: No especificada.

Versión: 01.00.00

Fecha subida: 20/04/14

Autoría: MECD MECD

Mejora: No especificada.