XMLHttpRequest / AJAX

Jorge Bastida Pérez
jbastida@ikusnet.com
neo@art-xtreme.com

Índice

0. Licencia

1.Presentación

1.1 Breve Historia

Desde hace un tiempo la palabra AJAX es la palabra de moda en el mundo del desarrollo de aplicaciones web.

El termino “Ajax” fue acuñado por Jesse James Garret[1] en su articulo “Ajax: A New Approach to Web Applications”, el cual recomiendo su lectura y que a lo largo de esta presentación citaré numerosas veces.
A modo de introducción paso a citar unas líneas brevemente.

“Dejando de lado esto, los diseñadores de interacción Web no pueden evitar sentirse envidiosos de nuestros colegas que crean software de escritorio. Las aplicaciones de escritorio tienen una riqueza y respuesta que parecía fuera del alcance en Internet. La misma simplicidad que ha permitido la rápida proliferación de la Web también crea una brecha entre las experiencias que podemos proveer y las experiencias que los usuarios pueden lograr de las aplicaciones de escritorio.”

1.2 ¿Qué es Ajax ?

Mucha gente me lapidaría por esto, pero ajax en sí no es nada, incluso emplear el termino ajax para definir el uso de XMLHttprequest[2] puede llegar a tratarse de una tautología.

Ajax no es una tecnología , no es un nuevo lenguaje de programación ni nada similar, Ajax es realmente el cúmulo de muchas tecnologías que por méritos propios han llegado a donde están.

AJAX (Asynchronous Javascript and XML) es una técnica de desarrollo web que genera aplicaciones web interactivas combinando:

- XHTML y CSS para la presentación de información
- Document Object Model (DOM[3]) para visualizar dinámicamente e interactuar con la información presentada
- XML, XSLT para intercambiar y manipular datos
- XMLHttpRequest para recuperar datos asíncronamente
- Javascript como nexo de unión de todas estas tecnologías

- No requiere plugins o capacidades específicas de ciertos navegadores.

1 **Jesse James Garrett(wikipedia.org)** is an information architect and founder of Adaptive Path, an information architecture and user experience firm. His work is widely known among information architects, who commonly refer to him simply as jjg

2 **XMLHttprequest (wikipedia.org)** The object was originally invented by Microsoft, used since Internet Explorer 5.0 as an ActiveX object, which is hence accessible via JavaScript, VBScript, or other scripting languages supported by the browser. Mozilla contributors then implemented a compatible native version in Mozilla 1.0. This was later followed by Apple since Safari 1.2 and Opera Software since Opera 8.0.

3 **DOM (wikipedia.org)** Document Object Model (DOM) is a form of representation of structured documents as an object-oriented model. DOM is the official World Wide Web Consortium (W3C) standard for representing structured documents in a platform- and language-neutral manner. The DOM is also the basis for a wide range of application programming interfaces, some of which are standardized by the W3C.

1.2 Comunicación síncrona vs. Asíncrona

En el desarrollo clásico tanto de aplicaciones como sitios web la comunicación con el usuario es síncrona respecto de las interacciones del usuario, es decir:

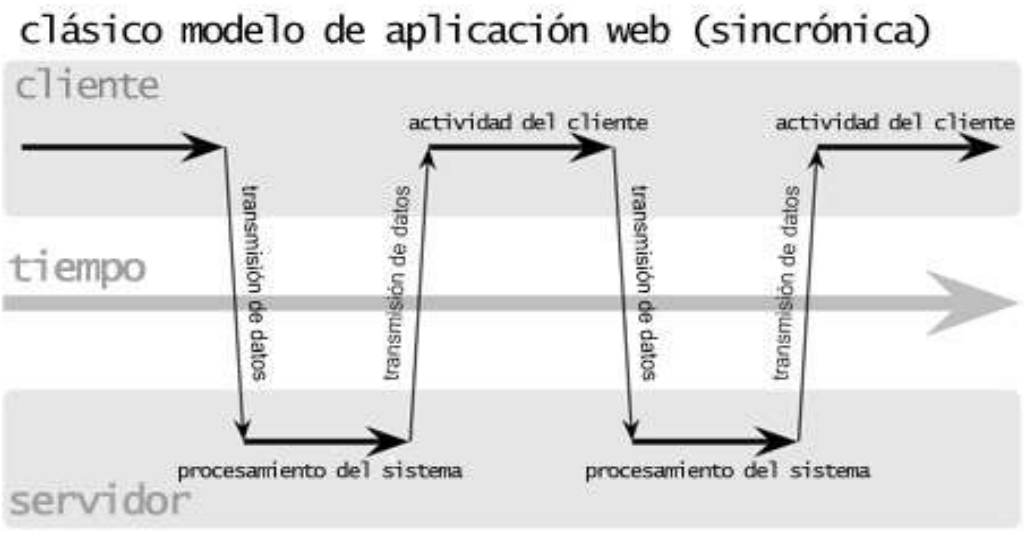
1. El usuario realiza una petición al servidor.
2. El servidor envía la página solicitada.
3. El usuario comienza a “leer” la página.
4. Llega un momento dado en el cual el usuario desea cambiar de página y mediante formularios o links realiza una petición al servidor volviendo al paso número 1.

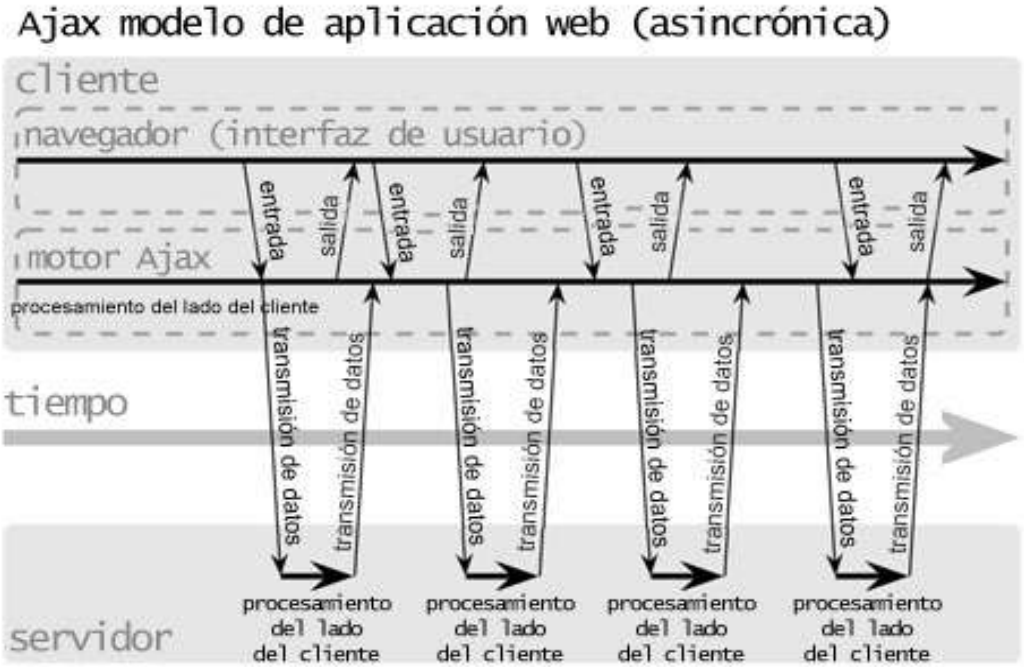
En cambio la comunicación asíncrona no implica sincronismo ante los eventos del usuario, sino que ante cualquier evento del usuario nosotros podemos proceder en consecuencia, es decir:

1. El usuario realiza una petición al servidor
2. El servidor envía la página solicitada
3. El usuario comienza a “leer” la página
4. Llega un momento dado en el cual el usuario quiere cambiar de página o alterar la información que contiene la misma. En ese momento dado la aplicación teniendo definidos los eventos posibles actúa en consecuencia a la interacción que haya podido suceder. Entonces podremos cambiar al usuario de página, o por el contrario modificar la misma para satisfacer al usuario. Estos cambios no implican cambiar de página.

El articulo de Jesse James incluye varias imágenes que hace más fácil la comprensión de esta comunicación asíncrona.(página siguiente)

¿Por qué la interacción del usuario con una aplicación web se interrumpe cada vez que se necesita algo del servidor?





Es por esto que una aplicación ajax termina con la idea “arrancar-frenar-arrancar-frenar”. Cada interacción de un usuario generaría un requerimiento HTTP para poder mostrar la nueva información, pero mediante ajax las peticiones al servidor pueden denegarse al “motor ajax” que es quien procesaría las peticiones contra el servidor, permitiendo o no al usuario interactuar con la página mientras el servidor está procesando la información, evitando de este modo los tiempo de carga en los cuales el usuario ve como su reloj de arena da vueltas mientas su navegador trata de componer la nueva página mientras recibe la información del servidor.

1.2.2 La “magia”

¿Que es “la magia”?, La magia de ajax consiste en la posibilidad de poder ante cualquier evento o la negación del mismo ejecutar código (en este ejemplo php) pudiendo alterar la estructura (DOM) de la página.

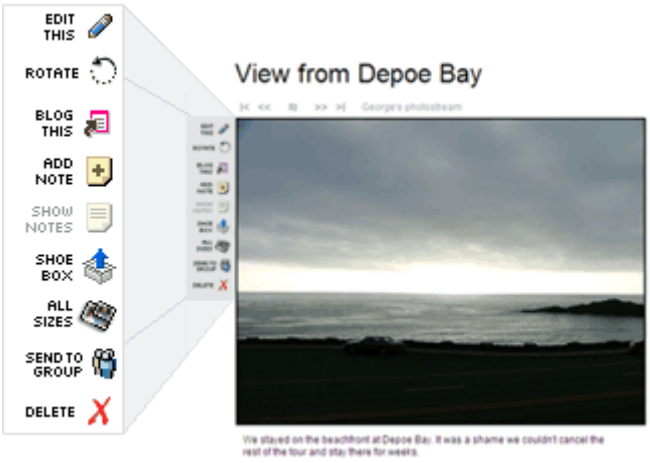
En palabras llanas y para que os hagáis una idea más “visual”...
Podemos ejecutar código php ante cualquier evento que controlaremos con javascript. Por ejemplo... pulse un botón , entre en un espacio , pase cierto tiempo etc...

Llegado a esta altura de la presentación tendréis más dudas que al principio pero tratare de enseñaros mas visualmente y con ejemplos en producción al alcance de millones de personas como esta palabras banales son capaces de hacer una ilusión, realidad.

1.3 Ejemplos prácticos en producción

1.3.1 Flickr Yahoo

Flickr es un espacio gratuito donde todo el mundo puede subir sus fotos y exponerlas al resto del mundo. En un principio flickr utilizaba macromedia flash para poder mostrar las fotos , hacerlas rotar , insertar comentarios sobre ellas etc....



Desde hace ya un tiempo incorpora ajax haciendo la navegación mas ligera , accesible e intuitiva. A día de hoy flickr cuenta con más de un millón de miembros, los cuales interactúan gracias a ajax con sus imágenes en tiempo real.
Uno de los mejores detalles que incorpora flickr es la posibilidad de editar el título y descripción (entre otros) en tiempo real sin necesidad de seleccionar la foto, encaminar al usuario a una página de edición donde rellenará un formulario con la nueva información.
Flickr permite pulsando en el título de la imagen convertir el mismo en un formulario, tras editar el título y pulsar el botón de guardar todo resto de formulario desaparece volviendo a la vista normal de la página. En ese momento la nueva información queda guardada en la base de datos para ser consultada por cualquier otro visitante. Este ejemplo demuestra que el peso de la aplicación no tiene porqué (como suponíamos) crecer sustancialmente sino al contrario, produciéndose una reducción considerable en la información intercambiada entre servidor y cliente.

1.3.2 Google Suggest

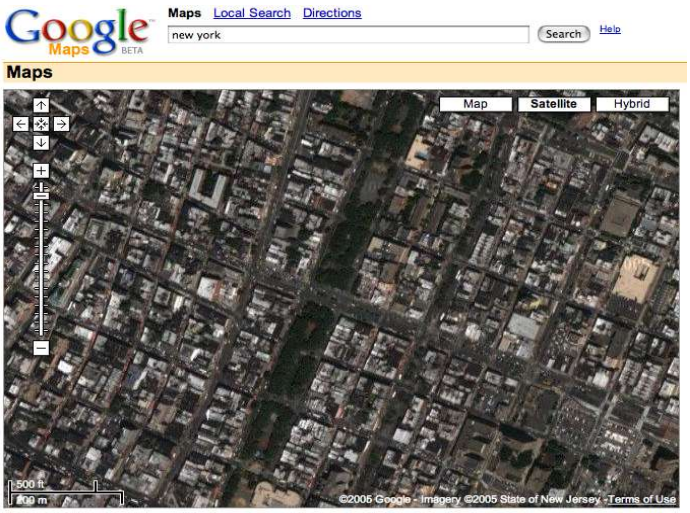
Google, buscador por excelencia demostró otra vez hace más de un año el potencial de la tecnología y de sus servidores. Google Suggest permite comprobar instantáneamente el numero de resultados que una búsqueda va a tener mientras se está escribiendo la misma. ¿Útil?¿Pesado para el servidor? ... si google puede, no es problema de ajax sino de los recursos a emplear.

Este ejemplo demuestra la utilidad de ajax para realizar búsquedas rápidas (ej: Datos personales, números de teléfono) o solucionar problemas a la hora de auto completar campos específicos dentro de formularios.

En la parte práctica copiaremos el ejemplo de google para buscar páginas web haciendo un pequeño código donde buscaremos el número de teléfono de una persona simplemente escribiendo una parte de su nombre o apellido.

1.3.3 Google Maps

Dentro de los innumerables servicios que google ofrece , uno está copando muchos titulares en Internet por su innovación (Plagiada por microsoft hace una semana) mediante la cual podemos consultar mapas y fotos por satélite de gran resolución de cualquier parte del mundo a una velocidad con la que muchas aplicaciones de escritorio no pueden competir.



Ajax es empleado en google maps para poder recargar pequeños cuadros de imagen que forman el plano visible. Por otro lado permite mostrar de una manera mas intuitiva herramientas de análisis , zoom etc...

La herramienta mas útil es el Drag & Drop que nos permite desplazarnos por la imagen arrastrándola en dirección contraria a donde queremos dirigirnos.

1.3.4 Yahoo!

- Oddpost (http://oddpost.com/learnmore)
  - El equipo de webmail Oddpost actualmente están trabajando en rediseñar Yahoo! Mail siguiendo la filosofía AJAX

1.3.4 Gmail

- A la hora de hablar de Gmail, solo quiero hacer un comentario (reflexión personal)... Si gmail fuese de código abierto y se pudiese instalar en cualquier servidor como cliente de webmail... ¿quien usaría Squirremail , neomail o cualquier otro... ?

2.Teoría

2.1 ¿Para qué nos sirve AJAX ?

Ajax nos sirve principalmente para diseñar y programar interfaces de usuario mucho más haya de la web , rompiendo las limitaciones que la “sincronía” supone y abriendo una nueva puerta que nos permitirá desarrollar aplicaciones que en un principio solo podrían concebirse para el escritorio, en aplicaciones web. Este cambio permitirá complementar aplicaciones de escritorio con aplicaciones similares con el plus de la universalidad de Internet permitiendo acceder a información de manera remota pero siempre con una interface similar y un numeroso grupo de utilidades que revalorizarán el producto convirtiendo cualquier desarrollo en algo más que una “página web”.

2.2 Ventajas / Desventajas

Desventajas

- El cliente necesita un navegador que soporte javascript .Hoy por hoy la mayoría de los navegadores soporta JavaScript. Internet Explorer, Mozilla, Firefox, Safari etc... Para esta desventaja tenemos una “excusa” , no estamos hablando de desarrollar páginas web con ajax, sino aplicaciones web, y como toda aplicación tiene unos requisitos mínimos, y siendo este javascript tampoco se discrimina un espectro muy amplio de usuarios.
- El mal uso de ajax/javascript puede mal emplearse sobrecargando el servidor de peticiones ej: Si hacemos que cada milisegundo haga una consulta con una base de datos... al administrador del sistema le pueden surgir instintos asesinos.

Ventajas

- Nos permite diseñar interfaces muchísimo mas dinámicas acercándonos a "aplicaciones de escritorio".
- La comunicación asíncrona con el servidor nos permite varias cosas que reducen el "peso de la página" / "lineas de código que el cliente tiene que descargarse"
    - Campos de select, si las opciones son muchas... no es necesario transmitirlas en la primera carga de la página, podemos producir la lista de opciones cuando el cliente pulse sobre el des plegable. Otra solución es hacer una búsqueda dinámica.
    - La navegación por la aplicación mediante ajax nos permite evitar al cliente descargar la cabecera de todos los "documentos","pantallas" <html><head><tit..... etc... Podemos cambiar el contenido de cualquier objeto del DOM dinámicamente ante los eventos que controlamos con javascript.
- No requiere plugins o capacidades específicas de ciertos navegadores.
- Las aplicaciones son más interactivas, responden a las interacciones del usuario más rápidamente, al estilo desktop
- Se reduce el tamaño de la información intercambiada
    - Muchas micro-peticiones, pero el flujo de datos global es inferior

2.3 Diferencias en el desarrollo “PRE” ajax / “POST”ajax

Ya sabemos que se puede hacer con ajax, de igual modo conocemos las ventajas que nos aporta un desarrollo de este tipo y las pegas que ello conlleva, pero (como todos os preguntareis... a la hora de programar que diferencias existen haciendo la aplicación síncrona o asíncrona .e
Las diferencias son mínimas por no decir nulas. De cara del programador su trabajo es el mismo, con la diferencia que las “salidas” de su código ha de estar empaquetada

para (generando un xml) ser transmitida al navegador cliente. El programador ha de conocer los identificadores de la estructura de la web para poder inyectar sus respuesta al lugar adecuado.
Estas diferencias podréis verlas mas adelante en la sección práctica de la presentación.

2.4 El cambio “radical” que supone la comunicación asíncrona.

La comunicación asíncrona para bien o para mal supone un cambio radical a la hora de diseñar aplicaciones web. Con la comunicación asíncrona nos libramos de unas barreras insalvables dentro del desarrollo clásico.

Ejemplos:

- **Editar un registro.**
  - **síncrono:** A la hora de actualizar un registro dentro de una aplicación “clásica” encaminamos al usuario a una página(1) donde le indicamos que ha de seleccionar un registro para editar. Una vez transmitida esa información a una página(2) le mostramos la información que disponemos actualmente para ese registro. Tras ser editada esa información le enviamos a otra página (3) donde le indicaremos si la actualización ha sido correcta o si por el contrario se ha dejado algún campo por cumplimentar , pudiendo hacerle repetir los pasos 2 y 3 infinitamente hasta que inserte bien los datos. Hemos utilizado hasta 3 o 4 páginas “vistas” y un mínimo de 3 tiempos de carga
  - **asíncrono:** A la hora de actualizar un registro dentro de una aplicación asíncrona desde la salida de cualquier búsqueda podemos indicarle que pulsando en el botón de herramientas le aparecerán las herramientas de editar y eliminar... pulsará en editar y se le desplegara debajo de su registro un formulario con toda la información de la que disponemos. Mientras escribe los nuevos datos podemos comprobar si son correctos y cuando haya terminado y todo este bien le dejamos pulsar en el botón guardar cambios. En ese momento el formulario desaparece y los cambios quedan guardado en la página. Hemos usado 1 página “vista”. Y un solo tiempo de carga.
- **Insertar un registro**
  - **síncrono:** Si vamos a insertar un registro en la mayoría de los casos queremos comprobar si los datos introducidos por el cliente son correcto o si por el contrario no nos ha puesto la @ en la dirección de email o si su numero de teléfono tiene menos de 9 cifras. En esos casos una vez cumplimentado el formulario página(1) le enviamos a la página(2) donde si los datos son correctos insertará el registro en la BD o le mostrará un mensaje de error sugiriéndole que vuelva atrasa y cambie la información que no pasa el filtro. Hemos utilizando un mínimo de 2 páginas y 2 tiempos de carga.
  - **asíncrono:** En el caso de una aplicación asíncrona podemos controlar la información que esta insertado mientras la escribe o cuando termina de hacerlo. Es por esto que en el caso de que algún dato no nos guste podemos bloquear el formulario o mostrarle una alerta, colorearle el campo en rojo (o lo que queramos). para insertar un registro podemos utilizar 1 o 2 páginas y 1 o 2 tiempos de carga.

Existen infinidad de ejemplos, solo tenéis que poneros a programar y comprobareis que ajax es una herramienta que os permitirá hacer de vuestros desarrollos algo más que una página web.

2.2 Alternativas a Ajax.

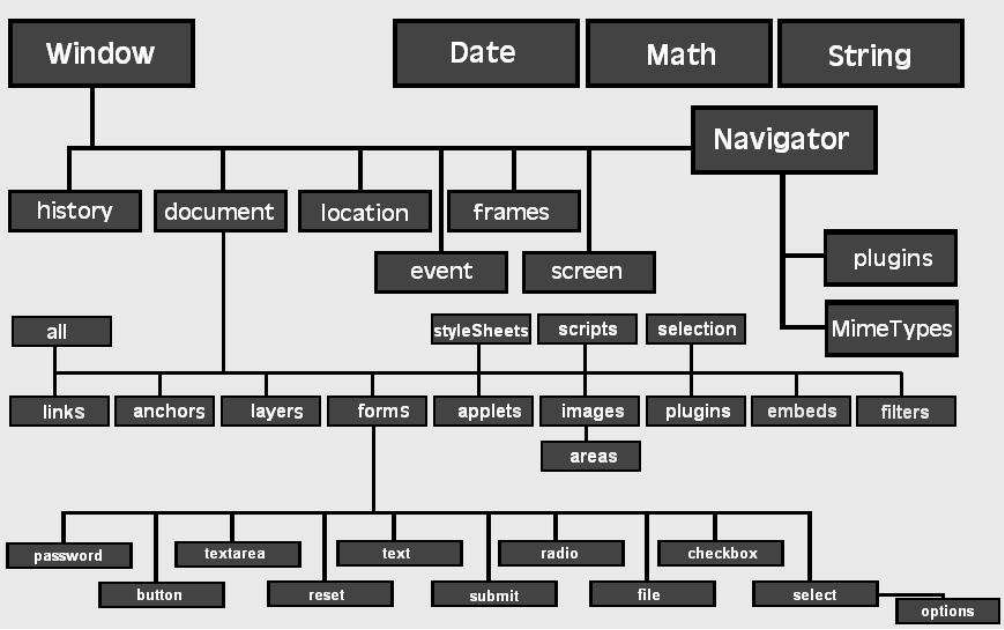
Existen numerosas técnicas que generan una interacción similar a ajax pero todas tienen la desventaja de requerir del lado del cliente plugins / applets para hacerlas funcionar. En cambio ajax no necesita de ningún añadido.

- Java Applets
- FLASH

3. Práctica / teoría

3.1 DOM

- Document Object Model (DOM) es una forma de representar documentos estructurados jerárquicos como un modelo orientado a objetos.
- DOM el el estándar oficial de la **World Wide Web Consortium** (W3C) para representar documentos estructurados en una manera neutral a la plataforma y al lenguaje de programación.
- DOM fue soportado inicialmente por los navegadores web para manipular elementos en un documento HTML. A través de DOM se puede acceder y actualizar dinámicamente el contenido, estructura y estilo de los documentos. Para evitar las incompatibilidades de las diferentes implementaciones de DOM en los diferentes navegadores, el W3C desarrolló la especificación DOM.
- DOM es un árbol que representa al documento visualizado en un navegador
- La funcionalidad del WC3 DOM hace posible la creación de documentos web extremadamente dinámicos.
- Un documento está conformado por varios nodos, cada uno representado un elemento del documento
- Un árbol establece una relación jerárquica entre elementos
    - Por ejemplo, el documento es un nodo que puede contener nodos imágenes, los cuales pueden contener atributos (relación parent-child)



3.2 Programación con framworks de ajax

A la hora de desarrollar aplicaciones con ajax lo más común es desarrollarlas dentro de un framework que te ayude a crear los diferentes objetos XMLHttpRequest dependiendo del navegador, hacer debuggin , etc...

La mayoría de frameworks son muy similares pero cada uno tiene sus ventajas y desventajas. Tras probar muchos he decidido centrarme en uno , su nombre es XAJAX y el administrador del proyecto es J. Max Wilson.

http://xajax.sourceforge.net/

El funcionamiento del framework es muy simple:

- Un archivo hace de servidor para el motor xajax, dentro de este archivo debemos definir todas las funciones que deseemos utilizar ajax. En el pie del documento tenemos que incluir las siguientes lineas.

```
$xajax = new xajax("archivo_servidor","prefijo",debug);

$xajax->registerFunction("funcion_a_registrar");

$xajax->processRequests();
```

Y dentro de las definiciones de funciones como ya comenté anteriormente tenemos que empaquetar nuestras salidas para que el framework nos la convierta en un xml que el navegador del cliente pueda interpretar. Por esto tenemos que añadir al final de las funciones las acciones que deseamos hacer para modificar el DOM de la página.

```
$salida = "El registro se ha insertado correctamente en la Base de Datos";

$respuesta = new xajaxResponse();

$respuesta->addAssign("resultado","innerHTML",$salida);

$respuesta->addAssign("resultado","className","clase_ok");

return $respuesta->getXML();
```

De esta manera cuando ejecutemos este código , se le asignará al objeto con id="resultado" como contenido "El registro se ha insertado correctamente en la Base de Datos" y se le aplicara la clase css "clase_ok".

4. práctica

4.1 Conocimientos Básicos

4.1.1 CSS

A la hora de trabajar con ajax es importante el uso de estilos CSS ya que nos permitirán cambiar el estilo de cualquier objeto de manera dinámica. Estos cambios de estilo son la escala más visual a la que el usuario llegará.

Por otro lado, los clientes están acostumbrados a cambiar continuamente de página y es por esto que cuando se enfrentan a una aplicación dinámica con gmail , no saben exactamente si los cambios se han guardado, han enviado el email o han borrado la imagen, es por esto que es **MUY IMPORTANTE** hace ver al cliente que se han producido cambios alterando colores , mostrando alertas , mensajes etc...

La definición de estilos css se suele hacer en archivos independientes al documento html, y invocamos al archivo desde el head del documento html.

```
<link href="estilos.css" type=text/css rel=stylesheet>
```

Dentro de este archivo principalmente tendremos 3 tipos de estilos

Estilo para objetos con class="noticia"
*ej: Blau Blau Blau *

```
.noticia{
        color:red;
        font-size:18px;
        border:#343434 solid 1px
}
```

Estilo para objetos con id="top"
*ej: Bienvenidos a casa *

```
#top{
        background-color:#444;
        font-size:10px;
}
```

Estilo para etiquetas
ej: <h2>Hola !!!!</h2>

```
h2{
        background-color:#222;
        font-size:20px;
}
```

4.1.2 JavaScript

- Es un lenguaje de scripting orientado a objetos
- Es utilizado especialmente para permitir la programación en navegadores
- La versión estándar de JavaScript, denominada ECMAScript va por la versión 1.5
- Java y Javascript son dos lenguajes totalmente diferentes

- El JavaScript de un página puede manipular y recuperar su contenido a través de la interfaz DOM
- El mayor uso de JavaScript es para crear aplicaciones web dinámicas
- JavaScript también es usado por:
  - Adobe Acrobat
  - La plataforma Mozilla con XUL

- Cada elemento en un árbol DOM tiene una serie de métodos que permiten manipularlo
- Dado el siguiente fragmento XHTML:

  **
- Podemos recuperar una referencia al mismo mediante:

  mi_imagen= document.getElementById("imagen")
- Los siguientes métodos serán disponibles a x:

  getAttribute() -> devuelve el valor del atributo pasado

  valor = mi_imagen.getAttribute("src") // devuelve logo.png

  setAttribute() -> asigna un valor a un atributo

  mi_imagen.setAttribute("align","right")

  removeAttribute() -> borra el atributo pasado

  mi_imagen.removeAttribute("align")

- De igual modo tenemos muchas más herramientas..
  - getElementsByTagName
  - createElement
  - appendChild
  - replaceChild(nuevo,viejo)
  - removeChild(hijo)
  - insertBefore(new,referencia)
  - hasChildNodes
  - etc...

4.2 Primeros pasos

Tras 14 folios de palabras, ha llegado el momento de convertir toda esa filosofía en algo palpable (aunque sea digitalmente).

Ejemplo #0 :: ejemplos/ejemplo0/index.php

```
##############################################
<html xmlns="http://www.w3.org/1999/xhtml"  xml:lang="es">
        <head>
        <title>Collection Manager</title>
        <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
        <link href="estilos.css" rel="stylesheet" type="text/css"/>
        <script language="JavaScript" src="ejemplo0.js"></script>
```

```
	<?php
		include_once("server0.server.php"); #servidor para XAJAX
		$xajax->printJavascript();
	?>
	</head>
	<body>
		<div id="resultado"></div>
		<input type="submit" onclick="xajax_mostrar()">
	</body>
</html>
```

##

Esta es un simple página con un boton que al ser pulsado llama a una funcion llamara xajax_mostrar ,esta funcion está definida en php y gracias a xajax podemos invocarla desde el JavaScript o como en este caso ante un evento concreto

#server0.server.php###

```
<?php
require("xajax.inc.php");

function mostrar(){
	$respuesta = new xajaxResponse();
	$respuesta->addAssign("resultado","innerHTML","Genial !!! Sabes pulsar botones");
	return $respuesta->getXML();
}

$xajax = new xajax("server0.server.php","xajax_",false);
$xajax->registerFunction("mostrar");
$xajax->processRequests();
?>
```

##

En este archivo definimos la función mostrar. Al final de la función (en este caso... la funcion solo se compone de esto) creamos un objeto en $respuesta que será la salida xml que mandaremos al navegador. con registerFunction(“mostrar”) , registramos/hacemos publica / añadimos la funcion mostrar definida en php a la “galería” de funciones que podremos invocar desde el php.
Al crear la respuesta con xajax("server0.server.php","xajax_",false) , le indicamos que las funciones están ene l archivo server0.server.php (en este caso es el mismo) , le ponemos a todas esas funciones el prefijo “xajax_” y dejamos el debug en false.

Finalmente mandara al navegador este xml.

```
<?xml versión=”1.0” encoding=”UTF-8” ?>
<xajax>
	<update action=”assign”>
		<target attribute=”innerHTML”>resultado</target>
	<data><![CDATA[Genial !!! Sabes pulsar botones]]></data>
	</update>
</xajax>
```

Que se encargara de modificar el DOM. Cambiando el contenido del div resultado por ...
“Genial !!! Sabes pulsar botones”.

Este es el ejemplo mas sencillo que se puede hacer con xajax, a partir de ahora os mostrare algunos intento de emular aplicaciones más complejas y populares utilizando estas técnicas.

4.3 Copiando Flickr

Como comente en el punto 1.3.1 Flickr aporta nuevas técnicas a la hora de editar información de manera dinámica en la página.

Este ejemplo consistirá en un titulo en el cual al pasar por encima se iluminara indicándonos que si pulsamos se convertirá en un área editable, pulsaremos y se convertiría en un formulario con 2 botones (guardar y cancelar) si pulsamos en guardar se guardara la información en la base de datos y si pulsamos en cancelar recuperaremos la información que teníamos en un principio.

Podéis ver el código fuente en ejemplos/ejemplo1/

Este ejemplo demostramos como esa "magia" que tiene flickr puede ser reproducida fácilmente. Por otro lado este ejemplo me sirve para mostraros como convertir una aplicación en el horror de un administrador de sistemas. La aplicación (ejemplo1) actualiza cada segundo la lista de las ciudades del mundo esperando encontrar cambios en la base de datos.

Suponiendo que tenemos 50 visitas online tendríamos 50 consultas a la base de datos por segundo lo que hace **3600 consultas en un minuto**. Por ello en el ejemplo1B hemos implementado de diferente manera el refresco, haciéndose únicamente cuando

el usuario inserta un nuevo registro. De este modo (si no se trata de una aplicación colaborativa) conseguimos el mismo efecto pero con una carga infinitamente menor.

4.4 Copiando Google Suggest

En el punto 1.3.2 comente que Google Suggest nos indicaba el numero de respuesta que nuestra búsqueda tendría mientras escribíamos la susodicha búsqueda.

En este punto vamos a reproducir Google Suggest contra una Base de Datos nuestra que contiene Nombre/Apellido y numero de teléfono.

Este ejemplo se puede hacer bien, mal y peor. Nosotros vamos a intentar que nuestra aplicación no se coma todos los recursos de la maquina haciendo cientos y cientos de consultas a la BD.

Para eso nuestro objetivo es controlar que el usuario ha terminado de escribir la búsqueda y en ese momento ejecutar la búsqueda de la cadena de texto.

¿Y cómo controlamos que ha terminado de escribir?.

La respuesta es sencilla, cuando comience a escribir iniciamos un intervalo que a los 300ms ejecute la búsqueda, y establecemos una salida al intervalo en caso de que pulse otra tecla (el cliente sigue escribiendo) , momento en el cual se inicia de nuevo el intervalo de 300ms. Si esta escribiendo a una velocidad normal cuando termine de escribir “mágica mente/pasaran 300ms sin escribir” se ejecutara la búsqueda y le aparecerán en la parte los resultado de la búsqueda.

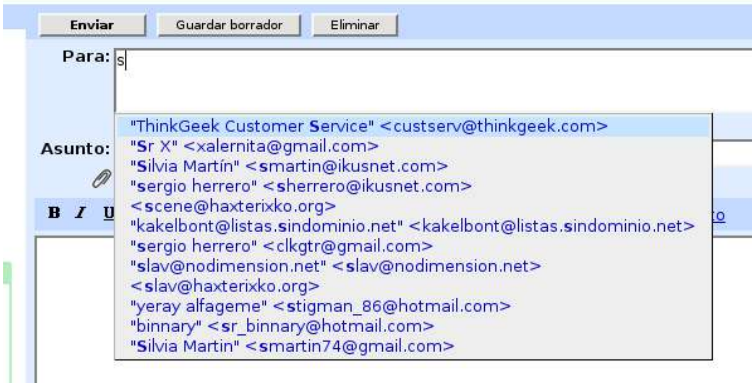
Podéis ver el código fuente en ejemplos/ejemplo2/

El resultado es infinitamente mejorable, pero queda claro que google no hace “magia” y que cualquiera de nosotros podemos integrar utilidades “igual de potentes” que las suyas.

Se me ocurre útil este ejemplo para hacer auto completar campos dentro de formularios, o para permitir realizar búsquedas rápidas y muy simples como pueden ser , insertando el nombre o apellido de una persona mostrar a la derecha su numero de teléfono o dirección o cualquier otro dato que nos resulte útil tener a mano.

4.4 Copiando Gmail

Como todos sabréis google desde hace ya un tiempo ofrece un servicio gratuito de correo electrónico en el que se puede almacenar hasta ,2,5 gb de correo electrónico. Fuera parte del gran tamaño de la bandeja de correo su servicio incorpora una interface muy similar a los clientes de correo de escritorio. Libreta de direcciones automática navegación por pestañas etc...



Una de las herramientas mas útiles es la agenda de contactos. gracias a ella mientras escribimos la dirección de correo electrónico o parte de su nombre nos sugiere una lista de personas de nuestra lista de correo a quienes enviar el correo electrónico.

El funcionamiento es similar al de Google Suggest con la peculiaridad de poder (pulsando encima de la sugerencia) agregarla a la lista de personas a quien va dirigido el email.

Podéis ver el código fuente en ejemplos/ejemplo3/

El principal inconveniente es de diseño. Para mostrar el cuando flotando solamente cuando tengamos algo escrito hay que cambiar de clase al campo donde se muestran los resultados de la búsqueda

Además sería interesante poder añadir mas de un destinatario para el email pulsando en varios. Por el momento no es posible ya que cuando deseas insertar otro has de eliminar lo que tienes escrito.

4.6 Copiando aplicaciones de escritorio.

4.5.1 Crear / Destruir objetos

JavaScript ya nos permitirá crear y destruir objetos del DOM de forma dinámica mediante scripts. Pero incorporando ajax a la receta podemos crear y destruir objetos en funcion de nuestras necesidades de una manera mucho más dinámica. Por ejemplo ante cualquier evento podríamos consultar una base de datos y crear 5 nuevas filas a una tabla para cumplimentar unos datos, o poder “expandir” información pulsando en un boton.

Podéis ver el código fuente en ejemplos/ejemplo4

El código nos permite pulsando en un boton crear un nuevo div hijo del primero que contendrá una serie de números aleatorios.

4.8 Abre tu mente

4.8.1 Que hacer / Que NO hacer con ajax

Ajax puede parecernos útil o no , pero como en todos los aspectos de la vida, de nada se puede abusar. Por muy útil, potente o maravilloso que nos parezca no podemos utilizarlo en el 100% de la página sino para pequeños detalles donde aporte novedades y se diferencie del resto de productos.

Tampoco podemos utilizar ajax si nuestro objetivo es la accesibilidad, los requisitos mínimos son “mínimos” pero no podemos confiar a la suerte que todos nuestros clientes tengan un navegador que soporte javascript. Por esto es necesario controlar las personas que entran a nuestra página, porque de lo contrario se encontraran con una aplicación que simplemente “no hace nada”.

Yo era el primer reacio a utilizar JavaScript, lo odiaba y lo odio pero creo que es la única manera actualmente de desarrollar aplicaciones web siendo estas “lo mas accesible posibles” , desechando flash, DHTML etc...

4.8.2 Limites de Ajax

A mi entender ajax “manda de una patada” los limites de la web muy lejos. Ajax nos permite hacer interfaces tan dinámicas como siempre hemos deseado pero la inaccesibilidad de flash, plugins de java, etc... no nos dejaban llegar.

Por otro lado a nivel “personal” ajax nos permite hacer muchas más cosas a las que estamos acostumbrados. Es por esto y por su “juventud” que es un espacio muy abierto para la innovación. Los limites personales los pone la imaginación de cada uno.

Estos limites “personales” pueden sufrir un placare tecnológico. Nos encantaría poder tener páginas web actualizadas al enano segundo donde todos los visitantes pudiesen intercambiar información y se refrescasen las ventanas de todos ellos mientras suben archivos y editan Bases de datos mientras graban ficheros de configuración. Pero hay que tener presente que del lado servidor, toda esa interactividad supone gastos cuantiosos. Es necesario llegar a un equilibro entre innovación, recursos y accesibilidad de los clientes.

Never Without:

- http://script.aculo.us/visual-effects
- http://www.ajaxian.com/archives/examples/index.html
- http://script.aculo.us/drag-and-drop
- http://www.javascriptkit.com/javaindex.shtml
- http://www.fiftyfoureleven.com/resources/programming/xmlhttprequest/examples
- http://www.phpriot.com/d/articles/php/application-design/google-suggest-ajaxac/page16.html
- http://www.codehelp.co.uk/php/xmlparse1.php
- http://en.wikipedia.org/wiki/AJAX
- http://www.phpbuilder.com/columns/kassemi20050606.php3
- http://www.adaptivepath.com/publications/essays/archives/000385.php
- http://www.ajaxian.com/
- http://www.xml.com/pub/a/2005/02/09/xml-http-request.html
- http://www.fiftyfoureleven.com/resources/programming/xmlhttprequest/examples
- http://dev.fiaminga.com/
- http://www.ajaxpatterns.org/
- http://www.ajaxmatters.com/r/welcome
- http://developer.apple.com/internet/webcontent/xmlhttpreq.html
- http://www.w3schools.com/dom/dom_http.asp
- http://jibbering.com/2002/4/httprequest.html
- http://www.peej.co.uk/articles/rich-user-experience.html
- http://www.jamesdam.com/ajax_login/login.html
- http://www.onlamp.com/pub/a/onlamp/2005/05/19/xmlhttprequest.html
- http://www.15seconds.com/issue/020606.htm
- http://www.informit.com/guides/content.asp?g=xml&seqNum=230&rl=1
- http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php
- http://www.ciberseleccion.com/index.php?topic=102.new
- http://particletree.com/features/smart-validation-with-ajax
- http://blog.joshuaeichorn.com/archives/2005/04/19/ajax-hello-world-with-sajax/
- http://www.xml.com/pub/a/2005/05/11/ajax-error.html
- http://paginaspersonales.deusto.es/dipina/
- http://www.modernmethod.com/sajax/
- http://xajax.sourceforge.net/
- http://ajax.zervaas.com.au/

etc.....