AJAX

Powered by

Rogelio Ferreira Escutia

Término AJAX

- El término AJAX se presentó por primera vez en el artículo "Ajax: A New Approach to Web Applications (http://www.adaptivepath.com/publications/essays/archives/000385.php) " publicado por Jesse James Garrett el 18 de Febrero de 2005.

- Hasta ese momento, no existía un término normalizado que hiciera referencia a un nuevo tipo de aplicación web que estaba apareciendo.

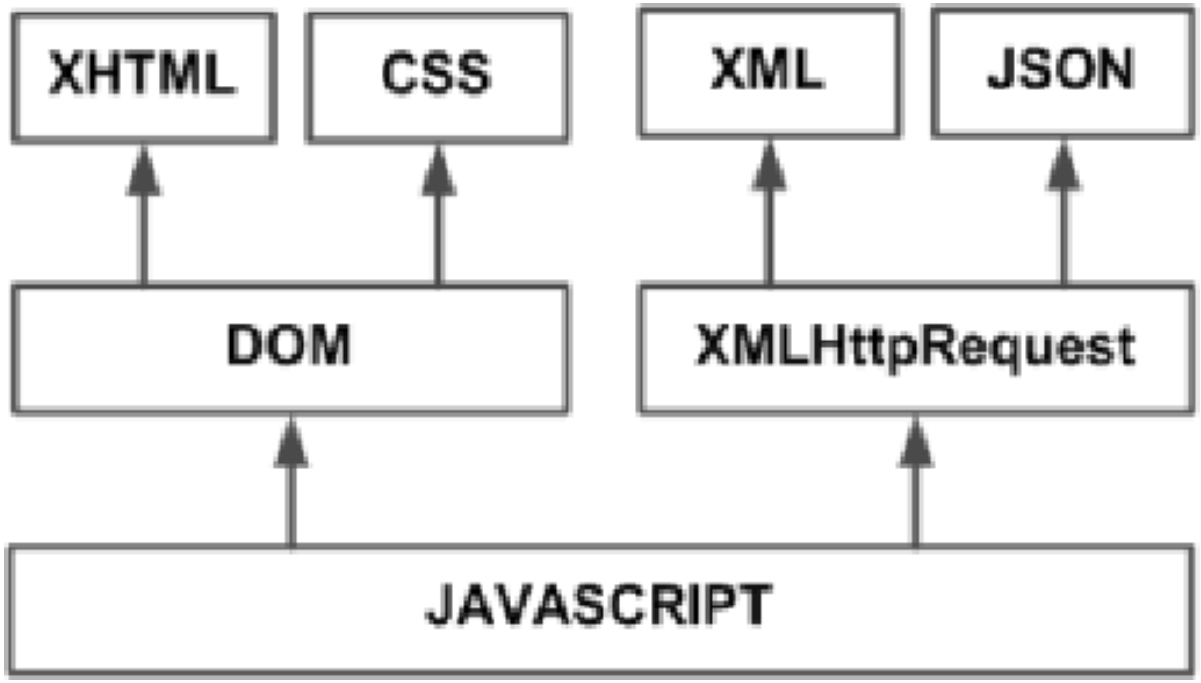
- En realidad, el término AJAX es un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como "JavaScript asíncrono + XML".

Tecnologías AJAX

- **Ajax no es una tecnología en sí mismo, en realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes.**

- **Las tecnologías que forman AJAX son:**
  - **XHTML y CSS, para crear una presentación basada en estándares.**
  - **DOM, para la interacción y manipulación dinámica de la presentación.**
  - **XML, XSLT y JSON, para el intercambio y la manipulación de información.**
  - **XMLHttpRequest, para el intercambio asíncrono de información.**
  - **JavaScript, para unir todas las demás tecnologías.**

Tecnologías AJAX



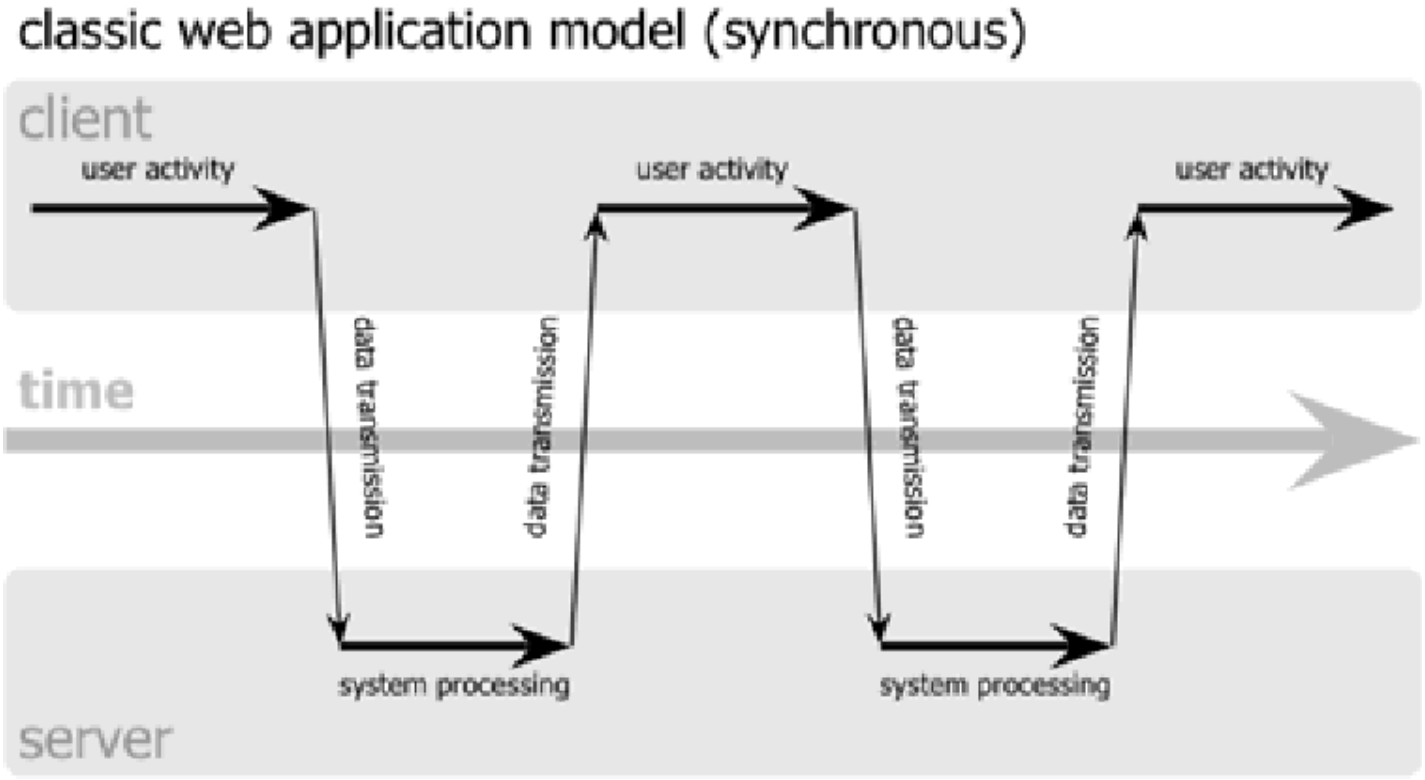
“Introducción a AJAX”, Javier Eguíluz Pérez, http://www.librosweb.es/ajax/, mayo 2010

Modelo Web Clásico

- **En las aplicaciones web tradicionales, las acciones del usuario en la página (pinchar en un botón, seleccionar un valor de una lista, etc.) desencadenan llamadas al servidor. Una vez procesada la petición del usuario, el servidor devuelve una nueva página HTML al navegador del usuario.**

- **Esta técnica tradicional para crear aplicaciones web funciona correctamente, pero no crea una buena sensación al usuario. Al realizar peticiones continuas al servidor, el usuario debe esperar a que se recargue la página con los cambios solicitados. Si la aplicación debe realizar peticiones continuas, su uso se convierte en algo molesto**

Modelo Web Clásico



"Introducción a AJAX", Javier Eguíluz Pérez, http://www.librosweb.es/ajax/, mayo 2010

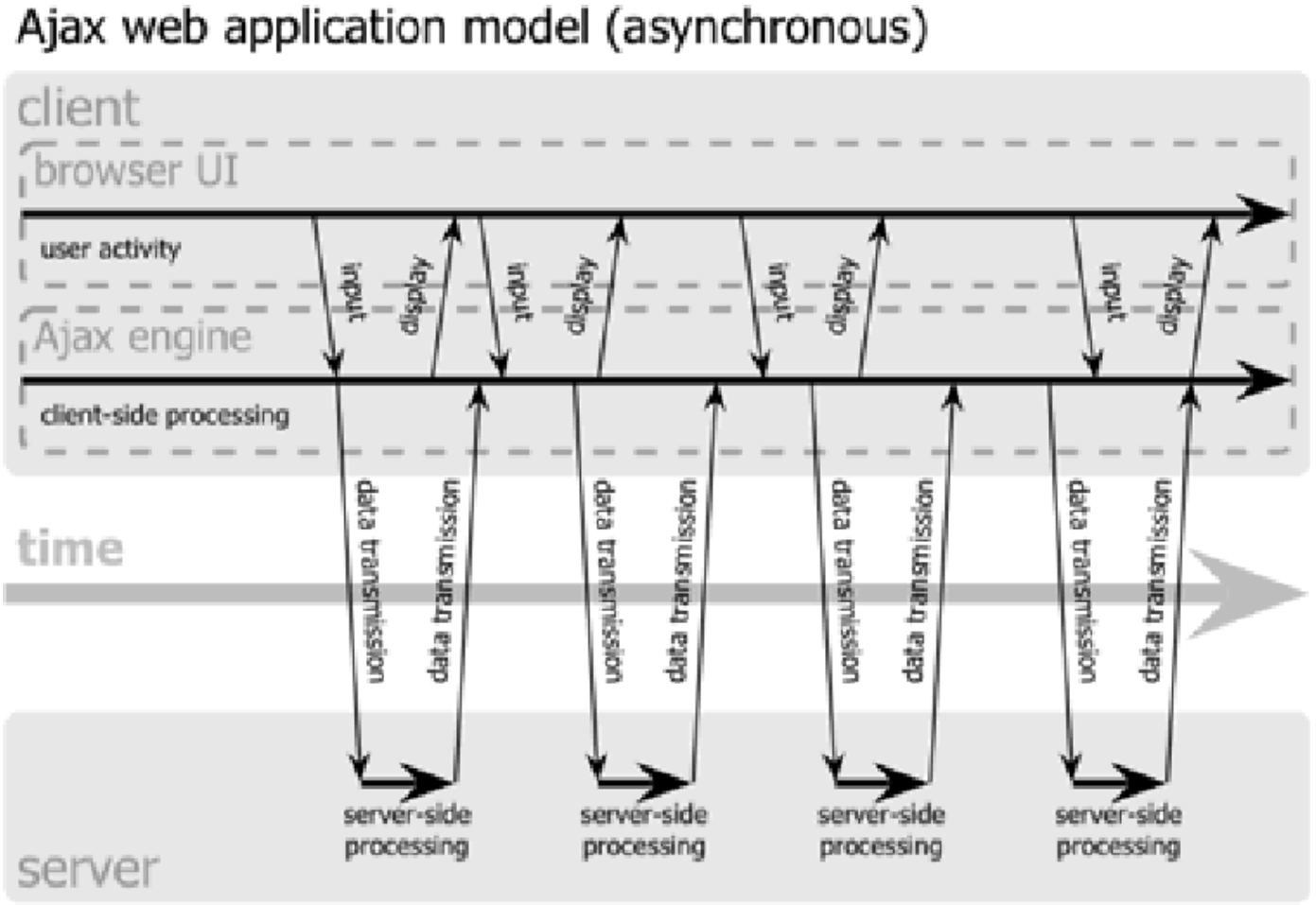
Modelo AJAX

- **AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano.**

- **Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor.**

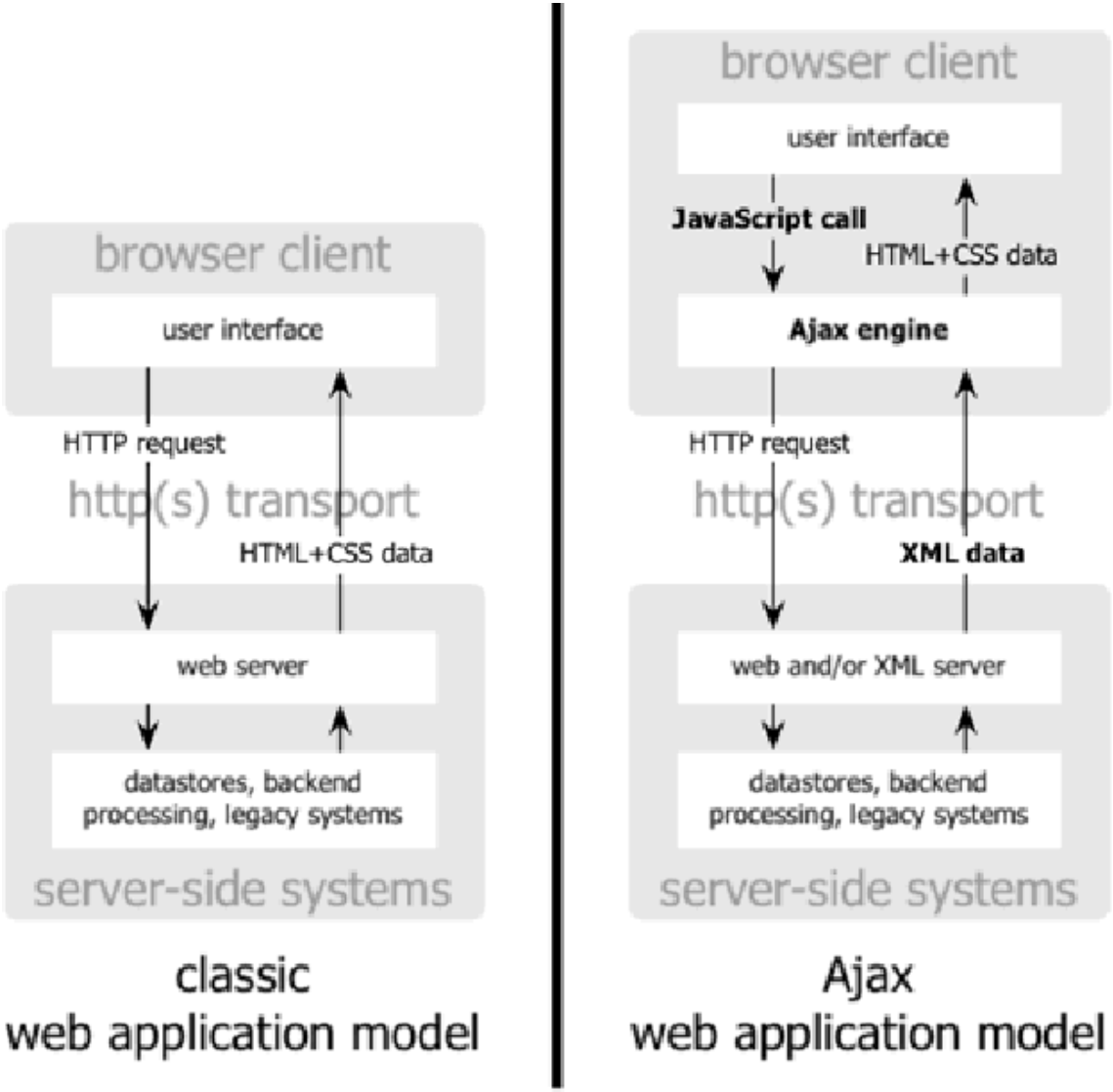
"Introducción a AJAX", Javier Eguíluz Pérez, http://www.librosweb.es/ajax/, mayo 2010

Modelo AJAX



"Introducción a AJAX", Javier Eguíluz Pérez, http://www.librosweb.es/ajax/, mayo 2010

Modelos



"Introducción a AJAX", Javier Eguíluz Pérez, http://www.librosweb.es/ajax/, mayo 2010

Objeto XMLHttp

- **El objeto “XMLHttp” es el corazón de toda aplicación AJAX, dado que es el que técnicamente permite realizar una petición con el servidor en forma asincrónica y sin cambiar de URL.**

1998) Primera versión

Remote Scripting de Microsoft

- **En 1998 Microsoft liberó una tecnología conocida como “Remote Scripting”, que permitía por medio de applet Java, realizar peticiones desde JavaScript al servidor sin que el ususario lo note ni se cambie de URL del sitio web, mientras se hace esperar al usuario.**

“Introducción a AJAX”, Javier Eguíluz Pérez, http://www.librosweb.es/ajax/, mayo 2010

1999) Segunda versión

ActiveX de Microsoft

- **En 1999 Microsoft lanzó el Internet Explorer 5.0 y ahí incorporó, bajo la tecnología ActiveX, un objeto llamado XMLHttp que reemplazaba funcionalmente al applet Java de la anterior implementación de Microsoft.**

- **De esta manera, era posible utilizar esta tecnología anteriormente conocida como "Remote Scripting" sin necesidad de que el usuario tenga instalada una máquina virtual de Java, lo que además aumentaba la velocidad de las aplicaciones de este tipo.**

2004) Tercera versión

XMLHttp de Google

- **En el 2004, algunas empresas lideradas por Google retomaron el concepto de ActiveX y desarrollaron su propia versión para aplicarlo en sitios web, con lo que se pudieron realizar aplicaciones nuevas en HTML como el manejo de mapas.**

- **Google logró que esta tecnología se hiciera conocida y muchos otros desarrolladores quisieron implementarla, dando nacimiento al concepto de AJAX.**

Modo de Uso

- **Para hacer una petición AJAX es necesario realizar los siguientes pasos:**
  - **1) Instanciar el objeto.**
  - **2) Configurar y abrir la petición.**
  - **3) Definir una función de JavaScript que se encargue de administrar la evolución de la petición.**
  - **4) Enviar la petición y los datos al servidor.**
  - **5) En la función definida antes, manipular el estado de la petición, y en el caso correcto, recibir los datos y actuar en consecuencia con ellos.**

1) Instanciar el objeto

Internet Explorer

objeto = new ActiveXObject(“nombreClase”);

IE7+, Firefox, Chrome, Opera, Safari

objeto = new XMLHttpRequestect();

1) Instanciar el objeto

Para cualquier navegador

```
if (window.XMLHttpRequest) {
        // codigo para IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp=new XMLHttpRequest();
} else {
        // codigo para IE6, IE5
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
```

2) Configurar y abrir la petición

Método "open"

- **El método "open" no abre la conexión con el servidor, sino que sólo configura la petición y la deja lista para enviarla y su codificación es:**

peticion.open(GET, "url", true);

4) Enviar la petición y los datos al servidor

Método "send"

- **El método "send" se encarga de enviar una petición al servidor una vez que se configuró el método open. Se pueden enviar datos vía POST o se puede usar el valor "null" si los envíos de datos son vía GET y se incluye el parámetro en la petición a la URL.**

```
peticion.send(null);
```

5) Administración de la petición

Métodos mas utilizados

open	Configura la conexión (no abre la conexión)
send	Enviar una petición al servidor
abort	Abortar un petición en curso
readyState	Administrar el estado de la conexión
status	Devuelve el código HTTP que nos devuelve el servidor
startusText	Devuelve un texto descriptivo del resultado del método "status"
responseText	Devuelve una cadena con el contenido del cuerpo devuelto por el servidor anta la petición
responseXML	Recibe el objeto XML nativo en JavaScript y luego procesarlo por los métodos del DOM
onreadystatechange	Define la función que se ejecutará cuando se produzca un evento

5) Administración de la petición

Método "abort"

- **El método "abort" se encarga de abortar una petición en curso, luego de haber invocado al método "send".**

- **Abortar implica que ya no se nos avisará cuando lleguen los datos y en el navegador se suspenderán todas las acciones que se estaban utilizando para enviar datos al servidor y recibirlos.**

```
peticion.abort();
```

5) Administración de la petición

"readyState"

- **Nos sirve para monitorear el estado de la petición y nos regresa un código numérico entre 0 y 4.**

Código	Estado	Descripción
0	Sin inicializar	El requerimiento sólo fué instanciado
1	Cargando	El requerimiento se configuró (con open) pero todavía no se envió.
2	Cargando	El requerimiento se envió o se está enviando, aunque todavía no tenemos respuesta alguna del servidor
3	Interactivo	El servidor ya respondió la petición, ya tenemos disponibles las cabeceras pero el contenido todavía se está descargando
4	Completo	La petición ya finalizó y el contenido está completo

5) Administración de la petición

"status"

- **La propiedad "status" devuelve el código HTTP que nos devolvió el servidor.**
- **Estos códigos nunca se administraron en la web 1.0 aunque los usuarios los conocen, como cuando se trata de accesar una página que no existe (error 404, recurso no encontrado).**
- **Si recibimos un error del servidor por una petición AJAX, el navegador no mostrará nada al usuario, por lo que es nuestra responsabilidad el procesamiento de estos errores.**

5) Administración de la petición

"status"

- **Algunos de los códigos de "status" mas usados son:**

Código	Descripción
200	La petición se pudo procesar en forma correcta.
404	La URL que invocamos no existe en el servidor.
500	Error interno del servidor. Puede indicarnos que el servidor está saturado o que hay algún error en el script ejecutado en el servidor.
400	La petición enviada al servidor es errónea. Hay algún inconveniente con las cabeceras o con la información POST enviada.
403	No tenemos permiso de acceder al recurso en el servidor.
405	No se acepta el método. Hay un problema al definir los métodos POST ó GET.
414	La URL pedida es muy larga. Puede producirse cuando se envían muchos datos por GET. En este caso, se debe cambiar el método a POST.
503	El servidor está temporalmente no disponible.