¿Qué es AJAX?

Ajax (Asynchronous JavaScript and XML) se refiere a un conjunto de tecnologías que se utilizan para desarrollar aplicaciones web. Al combinar estas tecnologías, las páginas web parece que son más receptivas puesto que los paquetes pequeños de datos se intercambian con el servidor y las páginas web no se vuelven a cargar cada vez que un usuario realiza un cambio de entrada. Ajax permite que un usuario de la aplicación web interactúe con una página web sin la interrupción que implica volver a cargar la página web. La interacción del sitio web ocurre rápidamente sólo con partes de la página de recarga y renovación.

¿Para qué sirve AJAX?

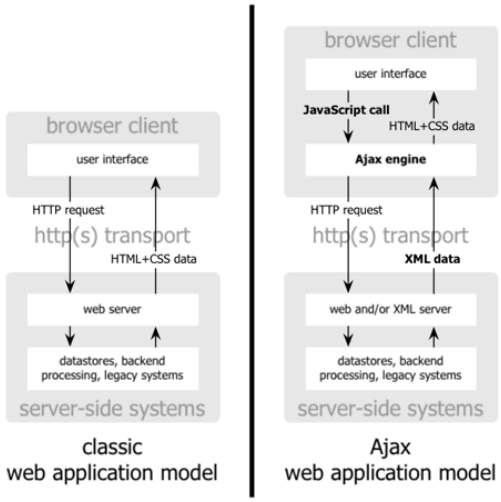
La intención de esta tecnología es la de poder realizar peticiones asíncronas al servidor con JavaScript. Son asíncronas porque ni la interfaz, ni la ejecución de JavaScript se bloquean una vez lanzada la petición. Al contrario, la web sigue ejecutándose hasta que llega la respuesta del servidor, momento en que el código responsable de la petición recupera el control y realiza alguna acción con la información obtenida.

Esta acción suele consistir en actualizar partes del contenido de la página. De esta forma, cuando se actualiza sólo una parte de la página se evita realizar una recargar entera, reduciendo el tiempo de espera del usuario y dando sensación de fluidez. La única forma de lograr algo similar hasta el momento era utilizando iframes y JavaScript, pero era más lento porque obligaba a cargar una página dentro de otra.

Infraestructura de comunicación

En las aplicaciones web tradicionales, las acciones del usuario en la página (presionar 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.

En el siguiente esquema, la imagen de la izquierda muestra el modelo tradicional de las aplicaciones web. La imagen de la derecha muestra el nuevo modelo propuesto por AJAX:

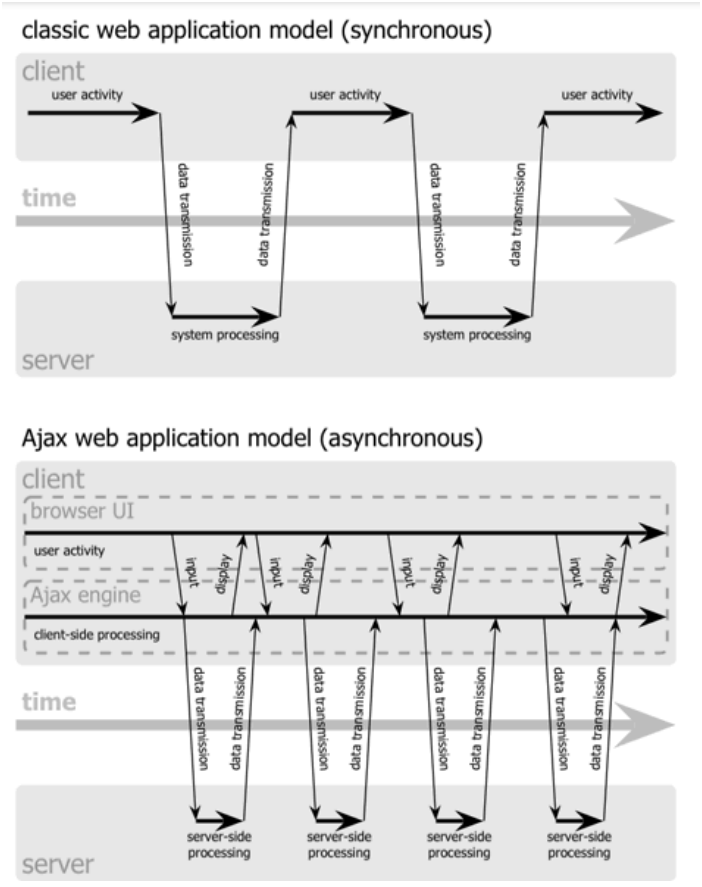


Esta técnica tradicional para crear aplicaciones web funciona correctamente, pero no crea una buena sensación al usuario. Al realizar peticiones continuas al servidor, el usuario debe esperar a que se recargue la página con los cambios solicitados.

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.

El siguiente esquema muestra la diferencia más importante entre una aplicación web tradicional y una aplicación web creada con AJAX. La imagen superior muestra la interacción síncrona propia de las aplicaciones web tradicionales. La imagen inferior muestra la comunicación asíncrona de las aplicaciones creadas con AJAX.



Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript que se realizan al elemento encargado de AJAX. Las peticiones más simples no requieren intervención del servidor, por lo que la respuesta es inmediata. Si la interacción requiere una respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX. En este caso, la interacción del usuario tampoco se ve interrumpida por recargas de página o largas esperas por la respuesta del servidor.

integración en páginas web

Las aplicaciones web Ajax utilizan JavaScript para interactuar con un servidor y actualizar las páginas web sin recargarlas. Los siguientes ocho pasos muestran cómo desarrollar una aplicación web Ajax simple.

1. Una aplicación web Ajax necesita un servidor u otra fuente de datos a la que se pueda acceder mediante una URL para actualizar la página. Para este ejemplo, usaremos el siguiente texto, guardado en un archivo llamado data.txt.

```
Hello, World! This text is loaded using Ajax.
```

2. Luego creamos el HTML, para ello usamos una plantilla HTML5 básica que contiene un elemento h1 con un atributo id. También un botón que activará la funcionalidad Ajax en la página cuando se haga clic en él.

```
<html>
<head>
        <title>An Ajax Web Application</title>
</head>
<body>
<h1 id="page-title"></h1>
<button id="load-data">Click Here to Load the Data</button>
</body>
</html>
```

3. Creamos un bloque de secuencia de comandos en el HTML (justo encima de la etiqueta del cuerpo de cierre) y escribimos un detector de eventos para detectar eventos de clic en el botón.

```
<script>
document.getElementById("load-data").addEventListener("click",function(){

});
</script>
```

4. Utilizamos un objeto XMLHttpRequest dentro de la función de devolución de llamada del detector de eventos para solicitar el archivo de datos.

```
var xmlhttp = new XMLHttpRequest();
xmlhttp.open("GET", "data.txt");
```

5. Para detectar la respuesta, buscamos un cambio en el estado de la respuesta xmlhttp, utilizando el controlador de eventos onreadystatechange. Escribimos una función de devolución de llamada para insertar los datos del archivo de texto en el elemento h1 cuando cambie el estado de la respuesta.

```
xmlhttp.onreadystatechange = function() {
        //Do something here
}
```

6. Utilizamos el método send () para enviar la solicitud.

```
xmlhttp.onreadystatechange = function() {
        document.getElementById("page-title").innerHTML = this.responseText;
}
```

7. La página HTML terminada debería verse así:

```
<html>
<head>
        <title>An Ajax Web Application</title>
</head>
<body>
<h1 id="page-title"></h1>
<button id="load-data">Click Here to Load the Data</button>
<script>
        document.getElementById("load-data").addEventListener("click",function(){
                var xmlhttp = new XMLHttpRequest();
                xmlhttp.open("GET", "data.txt");
                xmlhttp.onreadystatechange = function() {
                        document.getElementById("page-title").innerHTML = this.responseText;
                };
                xmlhttp.send();
        });
</script>
</body>
</html>
```

8. Abrimos la página HTML en un navegador y hacemos clic en el botón. El script producirá la siguiente salida en un navegador:

Funcionalidades

Ajax ofrece una serie de funcionalidades algunas de ellas son

- Incrementar el rendimiento de la página web.
- Soporte para el control de vista de datos y para la representación de plantillas del lado del cliente
- Reducción del consumo de recursos del servidor
- No es necesario volver a cargar toda la página, solo se vuelve a cargar una parte de la página que se requiere para volver a cargar.
- Aparte de obtener el objeto XMLHTTP, todo el procesamiento es el mismo para todos los tipos de navegador, porque se utiliza JavaScript.
- Con ajax, es posible desarrollar aplicaciones web más rápidas e interactivas.
-

Tecnologías de AJAX

Como ya menciono mi compañero AJAX no es una sola tecnología, ni es un lenguaje de programación, es un conjunto de tecnologías que se utilizan para desarrollar aplicaciones web. El sistema generalmente comprende:

- HTML/XHTML: Estas tecnologías se utilizan para mostrar contenido y CSS para la presentación.
- El Modelo de objetos del documento (DOM) para datos de visualización dinámicos y su interacción.
- XML para el intercambio de datos y XSLT para su manipulación. Aunque Muchos desarrolladores han comenzado a reemplazarlo por JSON porque es más similar a JavaScript en su forma.
- El objeto XMLHttpRequest para la comunicación asíncrona.
- Finalmente, JavaScript: el cual se utiliza para unir las tecnologías anteriores y para la validación del lado del cliente.

Entonces, Ajax incorpora estas tecnologías con el objetivo de crear un nuevo enfoque para desarrollar aplicaciones web.

Evolución en la historia

Al principio cada navegador implementaba Ajax de una manera particular, lo que hacía especialmente difícil el uso de esta tecnología, ya que había que escribir código duplicado, o triplicado, para realizar las solicitudes al estilo que requería cada cliente web.

Por ello es muy popular el uso de librerías para realizar Ajax. Una de las librerías más destacadas para Ajax, que hoy en día sigue usándose, es jQuery, la cual dispone de varios métodos para realizar un determinado trabajo con Ajax, que nos aseguran que el código se ejecutará perfectamente en todos los navegadores.

Luego llegó la interfaz estándar de acceso a Ajax fetch, que es propia de los navegadores. Fetch es un API estándar, por lo que no hace falta ninguna librería, ya que forma parte de Javascript.

Sin embargo, aunque fetch nos asegura un código bastante limpio y una interfaz estándar para las conexiones con Ajax, que nos ahorra la necesidad de usar librerías adicionales, sigue siendo bastante popular apoyarse en alguna biblioteca de funciones, ya que nos aportan funcionalidades adicionales para trabajar con Ajax y la consulta a servicios web, evitando tener que escribir mucho código.

Una librería muy usada es Axios, que nos asegura compatibilidad con todo tipo de navegadores y facilita mucho las consultas a APIs REST.

Framework backend para AJAX

Además de las librerías del lado del cliente, como las que mencione anteriormente jQuery o Axios, también existen frameworks backend que permiten trabajar con Ajax, pero sin necesidad de que intervenga código del lado del frontend. Dos ejemplos interesantes de tecnologías backend para hacer Ajax son:

Inertia: Permite crear aplicaciones SPA renderizadas mediante código del cliente, pero sin la complejidad de este tipo de aplicaciones y permitiendo seguir con la arquitectura habitual del backend.

Livewire: Es un paquete para construir aplicaciones dinámicas, con Ajax, sin necesidad de escribir código en Javascript. Simplemente se escribe PHP en el código de los controladores, consiguiendo acceder a las bases de datos y devolver vistas que se mostrarán en el cliente sin recargar la página.

Ventajas

Sin Ajax, los buscadores de Google y Facebook no podrían sugerirnos palabras claves mientras escribimos, Twitter debería recargar la página cada vez que retuiteamos un tweet y quizás no existiría el scroll infinito. Sin embargo, Ajax tiene tanto ventajas como desventajas, algunas ventajas son.

- Mejor experiencia de usuario. Ajax permite que las páginas se modifiquen sin tener que volver a cargarse, dándole al usuario la sensación de que los cambios se producen instantáneamente. Este comportamiento es propio de los programas de escritorio a los que la mayoría de los usuarios están más acostumbrados. La experiencia se vuelve mucho más interactiva.
- Optimización de recursos. Al no recargarse la página se reduce el tiempo implicado en cada transacción. También se utiliza menos ancho de banda.
- Alta compatibilidad. Ajax es soportado por casi todas las plataformas Web.

Desventajas

- Problemas de acceso. Normalmente, si un usuario realiza una consulta a una base de datos a través de muchos criterios (por ejemplo, categoría, precio, forma de pago, etc.), la página se recargará con una URL que reflejará los parámetros ingresados. El usuario puede guardar esa URL para volver a acceder a los resultados ya filtrados fácilmente. Pero con Ajax la URL no se modifica ante la consulta, por lo que deberemos volver a ingresar cada

filtro manualmente cuando queramos recuperar los resultados deseados. Existen métodos para modificar este comportamiento, pero agregan dificultad al desarrollo y peso al sitio.

- Problemas de SEO. Los buscadores tienen dificultades al analizar el código escrito en JavaScript. El hecho de que se no se generen nuevas URL elimina un importante factor de posicionamiento.
- Dificultad. Las aplicaciones con Ajax suelen requerir de un mayor tiempo de desarrollo.

Dadas sus ventajas y desventajas, Ajax sólo debería aplicarse en los casos en que una interacción cliente-servidor tradicional no sea capaz de brindar una buena experiencia de usuario.

Limitaciones

Si bien Ajax es una técnica de desarrollo de aplicaciones web que está diseñada para hacer que las páginas web sean más interactivas con un usuario, Ajax tiene algunas limitaciones a considerar antes de desarrollar una aplicación basada en Ajax, algunas de ellas son

- Seguridad y privacidad del usuario: no se abordan todas las preocupaciones. Los problemas relacionados con la seguridad y la privacidad del usuario deben tenerse en cuenta al desarrollar una aplicación Ajax.
- Accesibilidad: debido a que no todos los navegadores admiten JavaScript o XMLHttpRequest, debe asegurarse de proporcionar una forma de hacer que la aplicación web sea accesible para todos los usuarios.
- Marcador y navegación: dado que Ajax se utiliza para cargar de forma asincrónica bits de contenido en una página existente, es posible que parte de la información de la página no corresponda a una página recién cargada. Es posible que el historial del navegador y los marcadores no tengan el comportamiento correcto ya que la URL no se modificó a pesar de que se cambiaron partes de la página.

Aplicaciones basadas en AJAX

El concepto de AJAX ha existido desde hace varios años. Sin embargo, obtuvo un reconocimiento más amplio cuando Google comenzó a incorporar el concepto en GMail y Google Maps en el 2004. Hoy en día, se usa ampliamente en varias aplicaciones web para agilizar el proceso de comunicación del servidor, algunos ejemplos son.

- Sistemas de votación y calificación: ambas operaciones usan AJAX. Una vez que haces clic en el botón de calificación o de votación, el sitio web actualizará el cálculo, pero la página completa permanecerá sin cambios.
- Salas de chat: Algunos sitios web tienen un chat incorporado en su página principal, mediante el cual puedes hablar con un agente de atención al cliente. No tienes que preocuparte si quieres explorar la página al mismo tiempo. AJAX no volverá a cargar la página cada vez que envíes y recibas un mensaje nuevo.
- Notificación de tendencias de Twitter: Twitter ha incorporado AJAX recientemente para sus actualizaciones. Cada vez que se crean nuevos tweets sobre ciertos temas de tendencia, Twitter actualizará las nuevas cifras sin afectar la página principal.

https://desarrolloweb.com/home/ajax#track36

https://www.4rsoluciones.com/blog/ventajas-y-desventajas-de-utilizar-ajax-2/

https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=page-asynchronous-javascript-xml-ajax-overview

https://www.hostinger.mx/tutoriales/que-es-ajax

https://www.sitesbay.com/ajax/ajax-uses-of-ajax

https://jonmircha.com/ajax

https://www.webucator.com/article/how-to-develop-a-web-application-with-ajax/