1. ¿Qué son los métodos de HTTP en JavaScript y cuál es su propósito principal?

Los métodos de HTTP en JavaScript son comandos utilizados para indicar la acción que se desea realizar sobre un recurso en un servidor web. Se utilizan principalmente en el contexto de las solicitudes HTTP, que son el medio a través del cual se comunican los clientes (como navegadores web o aplicaciones) y los servidores.

1. Enumera al menos 5 métodos de HTTP en JavaScript y describe brevemente la función de cada uno.

* GET: Solicita datos de un servidor. Utilizado para recuperar información sin modificar el estado del recurso.
* POST: Envía datos al servidor, generalmente para crear un nuevo recurso. Se utiliza comúnmente al enviar formularios.
* PUT: Envía datos al servidor para actualizar un recurso existente. Si el recurso no existe, puede crear uno nuevo.
* DELETE: Elimina un recurso en el servidor.
* PATCH: Similar a PUT, pero se utiliza para realizar una modificación parcial de un recurso.

1. ¿Cuál es la diferencia entre los métodos GET y POST en una solicitud HTTP en JavaScript?

* GET: Ideal para solicitar datos, parámetros en la URL, datos limitados, visibles y almacenables en caché.
* POST: Útil para enviar datos, parámetros en el cuerpo de la solicitud, datos más grandes y complejos, no visibles en la URL y generalmente no almacenados en caché.

1. Explica cómo se puede realizar una solicitud GET en JavaScript utilizando el objeto XMLHttpRequest.

Para realizar una solicitud GET en JavaScript utilizando el objeto XMLHttpRequest, puedes seguir los siguientes pasos:

1. **Crear una instancia del objeto XMLHttpRequest.**
2. **Configurar la solicitud con el método GET y la URL del recurso.**
3. **Definir una función para manejar la respuesta de la solicitud.**
4. **Enviar la solicitud.**
5. ¿Qué es AJAX y cómo se relaciona con los métodos de HTTP en JavaScript?

AJAX, que significa Asynchronous JavaScript and XML, es una técnica de desarrollo web que permite actualizar partes de una página web sin tener que recargar toda la página. Esta técnica permite a las aplicaciones web ser más dinámicas y mejorar la experiencia del usuario al permitir la interacción en tiempo real.

1. Describe el proceso para enviar una solicitud POST en JavaScript utilizando la biblioteca Axios.

Se añade la biblioteca y se utiliza el siguiente código

axios.post('/user', {

firstName: 'Fred',

lastName: 'Flintstone'

})

.then(function (response) {

console.log(response);

})

.catch(function (error) {

console.log(error);

});

Ahí se puede observar que se llenan los valores que se van a crear, después se pide una respuesta y en caso de no obtener respuesta se atrapa el error.

1. ¿Por qué es importante comprender y utilizar los métodos de HTTP en el desarrollo web con JavaScript?

Para tener conectividad entre el front-end y el back-end.

1. ¿Cuál es la importancia de las respuestas de estado en las solicitudes HTTP y cómo se manejan en JavaScript?

Las respuestas son importantes para tener una idea de que le paso a la solicitud hecha, ya que cada una significa algo diferente:

* 100-199 denota una respuesta informativa.
* 200-299 denota una solicitud exitosa.
* 300-399 denota una redirección.
* 400-499 indica un error del cliente.
* 500-599 denota un [error del servidor](https://kinsta.com/es/blog/500-internal-server-error/).

1. ¿Qué papel juegan las promesas en las solicitudes HTTP en JavaScript y por qué son útiles?

Permite que los métodos asíncronos devuelvan valores como los métodos síncronos, en vez de devolver inmediatamente el valor final, el método asíncrono devuelve la promesa de proporcionar el valor en algún momento en el futuro.

Son importantes porque no siempre la respuesta es instantánea.

1. ¿Cuál es la diferencia entre el método fetch y el objeto XMLHttpRequest en términos de realizar solicitudes HTTP en JavaScript?

XMLHttpRequest devuelve los datos como respuesta, mientras que el objeto de respuesta de Fetch contiene información sobre el objeto de respuesta en sí mismo. Esto incluye headers, status code, etc

Referencias

* <https://kinsta.com/es/base-de-conocimiento/javascript-peticion-http/>
* <https://developer.mozilla.org/es/docs/Web/HTTP/Methods>
* <https://developer.mozilla.org/es/docs/Learn/JavaScript/Client-side_web_APIs/Fetching_data>
* <https://axios-http.com/es/docs/post_example>
* <https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Promise>
* <https://www.freecodecamp.org/espanol/news/tutorial-de-fetch-api-en-javascript-con-ejemplos-de-js-fetch-post-y-header>