

Agencia de
Aprendizaje
a lo largo
de la vida

Desarrollo Fullstack



Les damos la bienvenida

Vamos a comenzar a grabar la clase

Clase 24

Asincronismo en Javascript

- ▶ ¿Qué es?
- ▶ Call Stack
- ▶ Event Loop
- ▶ Promesas
- ▶ Async/Await

Clase 25

Solicitando info desde Javascript

- ▶ AJAX
- ▶ Fetch
- ▶ Template Strings
- ▶ Local Storage
- ▶ Repaso DOM

Clase 26

Node JS

- ▶ Servidor Web con Node
- ▶ Rutas básicas
- ▶ Método GET
- ▶ Enviar Datos
- ▶ Enviar Archivos

JAVASCRIPT

Pedir datos desde el cliente



Petición HTTP

Como vimos anteriormente, es una de las formas que tiene un navegador para **requerir información a un servidor**.

Tradicionalmente estas peticiones **traen archivos HTML, CSS, JAVASCRIPT, imágenes, entre otros** y frente a cada actualización de contenido debemos pedir un nuevo archivo que **recarga nuestra página** en el navegador.

AJAX

Surgió como una nueva modalidad para solicitar información que luego pueda ser añadida a nuestra página sin necesidad de ser recargada, permitiendo modificar porciones de un sitio.

Sus siglas significan ***Asynchronous Javascript and XML*** dado que originalmente la información se transmitía en formato XML pero hoy en día el formato JSON es mucho más común.

¿Cómo realizar peticiones “AJAX”?

XMLHttpRequest

Método nativo, de todos el más antiguo y complejo de utilizar.

Librerías de terceros

Antes de Fetch, se tenían que instalar librerías de terceros para facilitar el proceso nativo.

Algunas de estas librerías son axios, node-fetch o superagent dependiendo si la consulta era desde el lado del cliente o del servidor.

Fetch

Parte de los métodos nativos de Web API (cliente).

En el servidor, desde la versión de Node +18 se puede usar de forma nativa, mientras que en versiones anteriores era necesario alguna librería (axios, node-fetch).

Actualmente es el más utilizado.

Petición simple con Fetch

- Una petición HTTP es un proceso que no sabemos cuánto puede demorar y si su resultado será exitoso, por eso la función **fetch()** devuelve una promesa.
- Fetch permite solicitar información a una API Rest, retornando un **objeto JSON** con la información que necesitamos.
- Esa respuesta **posee metadatos**, por eso usamos el método **.json()** que devuelve otra promesa con **el body de la respuesta** convertido a objeto Javascript.

Promesa tradicional

```
const baseUrl = "https://rickandmortyapi.com/api";

fetch(baseUrl+'/character')
  .then(res => res.json())
  .then(data => console.log(data))
  .catch(err => console.log(err));
```

async/await

```
const baseUrl = "https://rickandmortyapi.com/api";
const getCharacters = async (url) => {
  const res = await fetch(url+'/character');
  const data = await res.json();

  console.log(data);
}
```


**Ahora que sabemos
consultar datos
externos, veamos
cómo podemos
unirlo a lo que ya
conocemos.**



No te olvides de dar el presente

Recordá:

- **Revisar la Cartelera de Novedades.**
- **Hacer tus consultas en el Foro.**

Todo en el Aula Virtual.

Gracias