

Agencia de Aprendizaje a lo largo de la vida

## **Desarrollo Fullstack**







# Les damos la bienvenida

Vamos a comenzar a grabar la clase













### **JAVASCRIPT**

Pedir datos desde el cliente



Agencia de Aprendizaje a lo largo de la vida





## 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 <u>cada actualización de</u> <u>contenido</u> 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.

Agencia de Aprendizaje a lo largo de la vida





### ¿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.

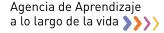
Algunas de estas librerías son axios, node-fetch o superagent dependiendo si la consulta era desde el lado del cliente o del servidor.

#### 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 <u>permite solicitar información a una</u> <u>API Rest,</u> retornando un **objeto JSON** con la información que necesitamos.
- Esa respuesta **posee metadatos**, por eso usamos el método .json() que <u>devuelve</u> <u>otra promesa</u> con **el body de la respuesta** convertido a <u>objeto Javascript</u>.

#### 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**