

# Introducción

# JavaScript en servidor

## Qué es nodejs

- NodeJS es un intérprete de JavaScript que se ejecuta en servidor.
- Está basado en el motor de JavaScript que utiliza Google Chrome (V8), escrito en C++

## Características principales

- El tener el mismo lenguaje en cliente y servidor
  - Permite a cualquier persona desarrollar en backend o en frontend
  - Permite reusar código o incluso mover código de cliente a servidor o al revés
- Está orientado a eventos y utiliza un modelo asíncrono (propio de JavaScript).
- Al contrario que en el navegador, encontramos muchas llamadas asíncronas:
  - Llamadas a APIs
  - Lectura y escritura de ficheros
  - Ejecución de cálculos en el servidor
  - ....
- Llamadas síncronas en servidor serían fatales:
  - ¡Bloquearíamos las conexiones al servidor hasta que acabase la instrucción bloqueante!
  - Al ser asíncrono podremos tener muchas sesiones concurrentes
- Es monohilo
  - Utiliza un solo procesador
  - Si queremos usar toda la potencia de la CPU, tendremos que levantar varias instancias de node y utilizar un balanceador de carga ([por ejemplo con pm2](#))

## Desventajas

- Trabajar con código asíncrono hace que a veces el código no sea excesivamente legible

- Imagina que guardamos un registro de los accesos de los usuarios a nuestra app:

```
trackUser = function(userId) {  
  users.findOne({userId: userId}, function(err, user) {  
    var logIn = {userName: user.name, when: new Date};  
    logIns.insert(logIn, function(err, done) {  
      console.log(' wrote log-in! ');  
    });  
  });  
};
```

- Tenemos 3 funciones anidadas en una simple operación.
- Esto es lo que se conoce como [callback hell](#)

# Evitar el callback en el navegador

- Mediante el [uso de promesas](#)
- Se trata de escribir código asíncrono con un estilo síncrono.
- Opciones más actuales:
  - Generators / Yields (ES6)
  - Async / Await (ES7)
  - El soporte de ES6 en node es limitado (--harmony) y también en el navegador => TRANSPIERS (babel)
- Ver [comparativa de métodos asíncronos](#)

# Hola Mundo en node

```
console.log ("Hola Mundo");
```

- Editamos un fichero en JavaScript, *holaMundo.js*:
- Lo ejecutamos mediante *node holaMundo.js*
- Si escribimos *node* sin más, podemos acceder a la consola de node, un intérprete de JavaScript, igual que el que tenemos en el navegador

# npm

- Es el gestor de paquetes de node
  - Propongo hacer dos prácticas para coger la dinámica del uso de npm y sus librerías y de trabajar con node:
    - Crear una librería en node.js
    - Crear una api rest mediante node.js
- 

Revision #1

Created 19 April 2023 17:37:05 by molombo

Updated 19 April 2023 17:39:07 by molombo