



Node.js & MongoDB

Instructor

Mario Romero

Scrum Master / Full-stack Dev

- twitter: @maaroarg
- email: maaroarg@gmail.com
- github: maaroarg

Clase 1 - Agenda

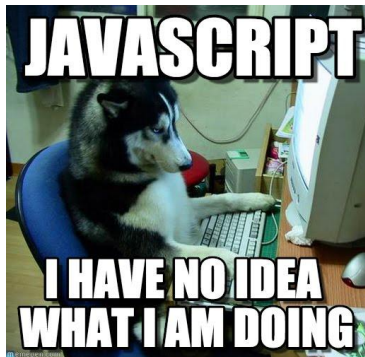
- Qué es Node.js
- Cómo se come? Quién lo come?
- Qué problema resuelve?
- Repaso JS
 - Scope de variables: var, let, const
 - ES5 vs ES6... Fight!
 - Objects
 - Arrays

Qué es Node.js?

Es Javascript del lado del servidor.

Podemos explicarlo un poco más complejo...

“Node.js® es un entorno de ejecución para JavaScript construido con el [motor de JavaScript V8 de Chrome](#). Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, [npm](#), es el ecosistema más grande de librerías de código abierto en el mundo.”



Fuente: <https://nodejs.org/es/>

Cómo se come?

Podemos instalar el entorno de Node.js en Windows, Mac o Linux. Incluso tenemos el código fuente para portarlo y compilarlo en cualquier plataforma. Está programado en una combinación de C++ y Javascript.

Podemos decir que Node.js es...

- Un archivo ejecutable (CLI - command line interface)
- Un servidor web programable (Nuestro propio Apache)
- Un entorno de ejecución de código JS en el server

Quién lo come?

	<p>Lograron disminuir los tiempos de desarrollo y mejoraron notablemente el mantenimiento al <u>migrar sus aplicativos Java a Node.js.</u></p>
	<p>Que el servicio de streaming más grande del mundo use <u>Node habla de su performance.</u> Les funciona tan bien para la parte de interacción de las apps con los usuarios que incluso están migrando capas de acceso a datos.</p>
	<p><u>Tres factores fueron decisivos para su adopción:</u> procesa mucha información en poco tiempo, los errores se pueden fixear on the fly y su comunidad open-source lo mejora todo el tiempo.</p>

Pero entonces...

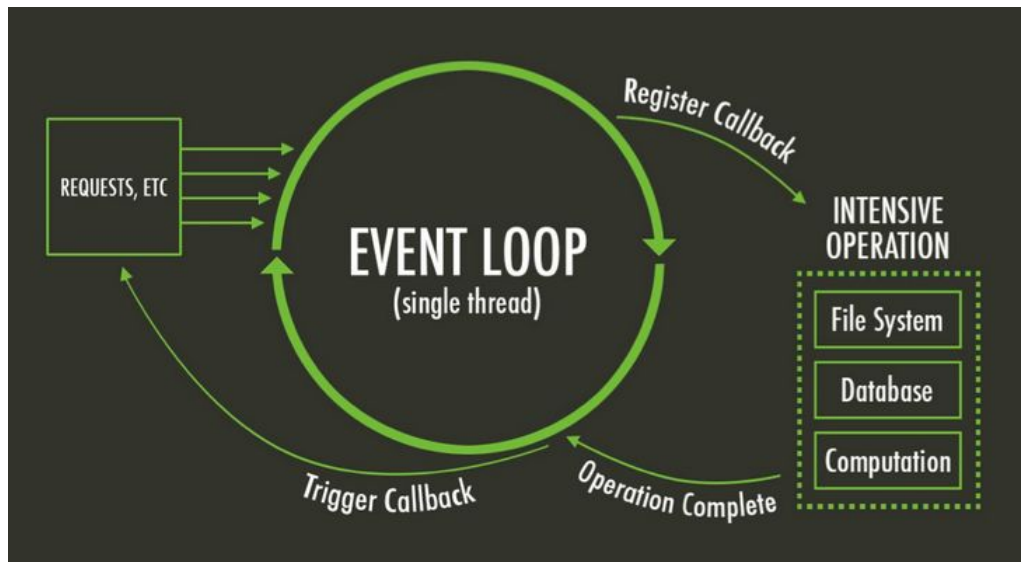


Esto quiere decir que tengo que migrar **TOD**O a Node.js?

WARNING! Node.js no es la mejor solución para todos los casos. Funciona muy bien cuando tenemos alta concurrencia con E/S (Llamadas APIs, peticiones a DB, acceso a disco) pero no escala bien con uso intensivo de CPU...

No podemos martillar tornillos (Bueno, no debemos).



Qué problema resuelve Node.js?



Cada petición a un server tradicional dispara un nuevo hilo acompañado de potencialmente 2MB de memoria. Si nuestro servidor dispone de 8GB de RAM, podríamos atender hasta 4000 potenciales clientes concurrentes.

En lugar de generar un nuevo hilo por cada conexión, Node.js **atiende el pedido dentro del mismo hilo de ejecución del proceso**. Gracias a la arquitectura no bloqueante, podemos soportar muchas más conexiones concurrentes con menos recursos que una arquitectura multihilo.

Node.js vs Browsers

	
<p>No existe el objeto “window” porque no tiene ninguna ventana donde dibujar nada. El objeto global se llama... “global”. Las variables creadas con “var” fuera de un bloque son locales al módulo.</p>	<p>El objeto window es el objeto global. Las variables creadas con “var” fuera un bloque son globales por default.</p>
<p>No existe el objeto “location” porque no tiene que referenciar ninguna url.</p>	<p>El objeto location representa la url actual.</p>
<p>No existe “document” porque no hay nada que renderizar</p>	<p>El objeto “document” representa al DOM de una página web.</p>