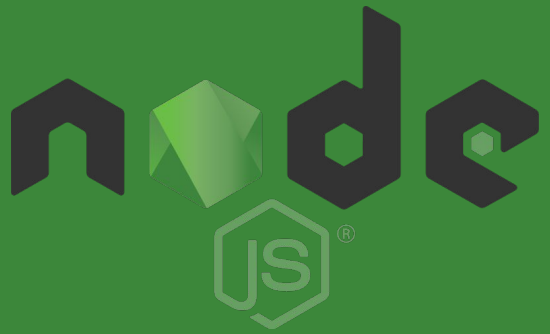
NODE JS

Que es?



¿Qué es Node js?

- Node js **es un entorno de ejecución multiplataforma que permite ejecutar código javascript fuera de un navegador.**
- Node js está basado en **v8**, que es el motor de javascript del navegador Chrome.

- https://nodejs.dev/learn/introduction-to-nodejs

¿Qué es Node js?

- Node js es un **software de código abierto** que **unifica el desarrollo de aplicaciones web en un único lenguaje de programación** (javascript) que se enfoca tanto en scripts del lado del servidor como del lado del cliente.
- Node js **optimiza el rendimiento y la escalabilidad en aplicaciones** web que cuentan con muchas operaciones de I/O (entrada / salida), así como para aplicaciones web en tiempo real.
- Node js es compatible con casi todos los sistemas operativos. Por ejemplo, Windows, Linux, FreeBSD, macOS, OpenBSD son compatibles oficialmente con Node js, mientras que el código fuente proporcionado se modifica para que sea compatible con otros sistemas.

¿Qué NO es Node js?

- Que **NO** es Node js:
  - Una biblioteca
  - Un Framework
  - Un lenguaje de programación

- Tanto una biblioteca como un framework desempeñan un papel vital en el desarrollo de software. Una **biblioteca realiza una operación específica** o bien definida, mientras que un f**ramework proporciona un esqueleto** donde los programadores definen el contenido de la aplicación de la operación.

Características de Node js?

- Node js tiene un modelo de ejecución de un solo subproceso, sin bloqueos y basado en eventos.
  - **Non-blocking**
  - **Event-driven**
  - **Single-thread**

Características de Node js?

- **Non-blocking**: L**os programas pasan la mayor parte de su tiempo esperando que otras cosas sucedan,** por ejemplo, operaciones de I/O como acceso a discos / Bases de datos, solicitudes de red, etc.
  Node.js se ocupa de esto haciendo que estas operaciones no se bloqueen. Esto significa que la ejecución del programa puede continuar mientras ocurren.

- **Todas las API de la biblioteca Node.js son asíncronas, lo que significa que un servidor basado en Node nunca se queda esperando a que una API devuelva datos**. El servidor pasa a la siguiente API después de llamarlo y un mecanismo de notificación de Eventos de Node (callback) ayuda al servidor a obtener una respuesta de la llamada API anterior.

Características de Node js?

- **Event-driven**: La naturaleza impulsada por eventos de Node.js describe cómo se programan las operaciones. En entornos de procedimiento típicos, un programa tiene un punto de entrada que ejecuta un conjunto de comandos hasta la terminación, o entra en un bucle y realiza algún procesamiento en cada iteración. Node.js tiene un bucle de eventos integrado que no está expuesto al desarrollador. Es el trabajo del **event loop** decidir qué pieza de código se va a ejecutar a continuación. Típicamente, esta será una función de callback que está lista para ejecutarse en respuesta a algún otro evento. Por ejemplo, una operación de filesystem puede haberse completado, un timeout puede haber expirado, o una nueva solicitud de red puede haber llegado.

- Este event loop integrado simplifica la programación asincrónica proporcionando un enfoque coherente y evitando la necesidad de aplicaciones para administrar su propia programación.

Características de Node js?

- **Single-thread**: La naturaleza de un solo subproceso de Node.js simplemente significa que sólo hay un subproceso de ejecución en cada proceso. Además, cada pieza de código está garantizada para ejecutarse hasta su terminación sin ser interrumpida por otras operaciones. Esto simplifica enormemente el desarrollo y hace que los programas sean más fáciles de razonar. Elimina la posibilidad de una serie de problemas de simultaneidad. Por ejemplo, no es necesario sincronizar/bloquear el acceso al estado de proceso compartido . Un proceso no puede estancarse.

- La programación de un solo subproceso es factible si el hilo nunca se bloquea esperando a que finalice el trabajo de larga duración. De esta forma, este modelo de programación simplificado es posible gracias a la naturaleza no bloqueante de Node.js.

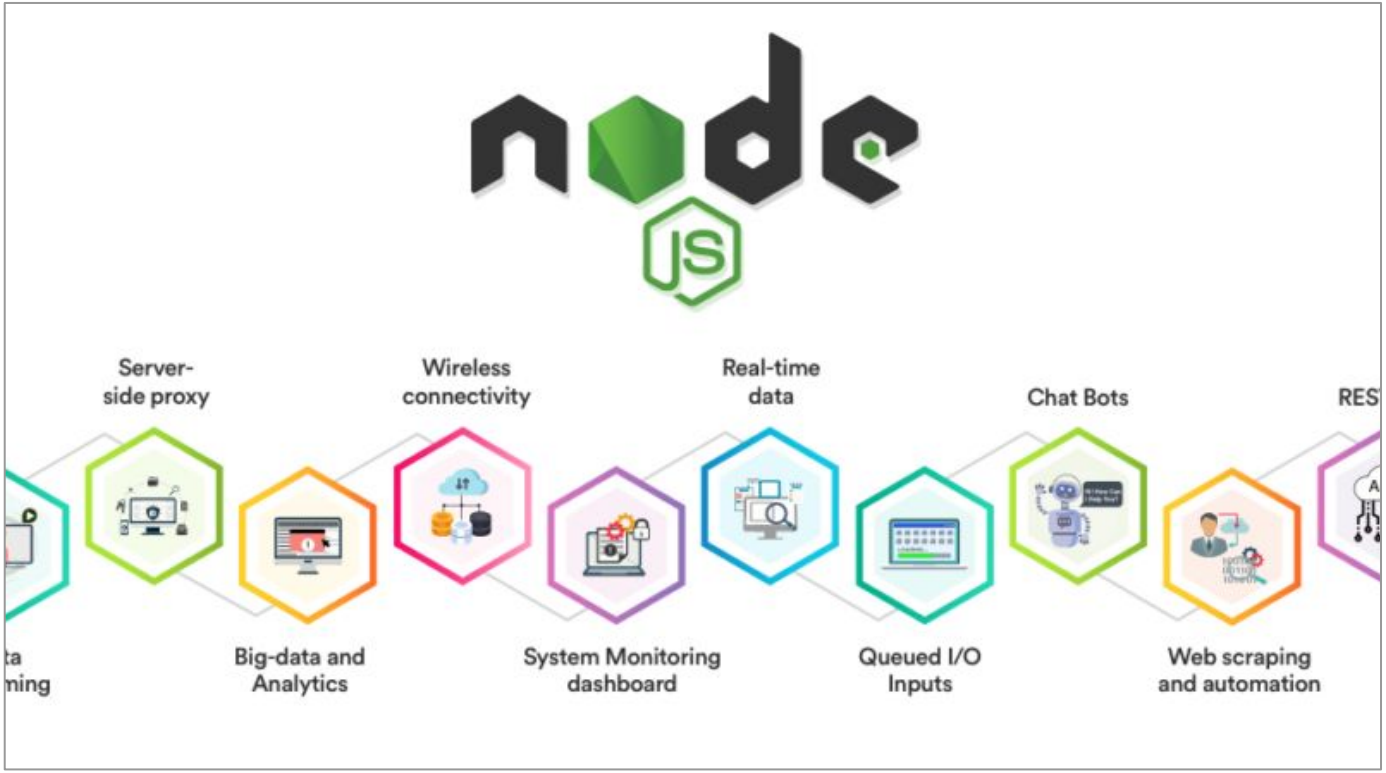
Bloqueante vs. No Bloqueante

- Las tareas que realizan los lenguajes de programación usualmente llevan un tiempo para realizarse. El acceso a disco, base de datos, red, etc. requieren un tiempo para procesarse. Durante el tiempo de procesamiento de determinadas tareas el proceso que las ha iniciado tiene dos opciones:
  - **Bloqueante**: Quedarse en estado "suspendido" esperando hasta que la tarea termine, es decir, hasta que la base de datos devuelva el resultado o el archivo se haya podido abrir o leer.
  - **No Bloqueante**: Liberar el flujo de ejecución, de modo que el proceso que inició la acción puede atender otras necesidades del lenguaje.

- Los lenguajes de programación generalmente se comportan de una u otra manera. Por ejemplo, ASP.Net / PHP son bloqueantes (por default), y Javascript es no bloqueante.

¿Para que se usa Node js?

- Node se utiliza en conjunto con frameworks y bibliotecas para poder realizar proyectos javascript tanto frontend como backend.



¿Cómo se instala Node js?

- Para instalar **Node** hay que descargarlo de https://nodejs.org/ Esta descarga va a instalar NPM junto con **Node**.

- **NPM** (**N**ode **P**ackage **M**anager) es el gestor de paquetes/módulos de Node. El gestor de paquetes nos va a permitir instalar dependencias en nuestro proyecto, por ejemplo, en C# utilizamos NuGet para instalar Dapper. En Node utilizaremos NPM para instalarlo.

- Para saber que version de **NPM** tenemos instalada :

```
C:\temp>npm -v
8.5.0
```

- Para actualizar a la última versión de NPM (-g es global):

```
C:\temp>npm install npm@latest -g
```

¿Cómo se instala Node js?

- **NPM** realiza un seguimiento de los módulos instalados en un proyecto con el archivo **`package.json`**, que se encuentra en el directorio de un proyecto y contiene lo siguiente:
  - Todos los módulos necesarios para un proyecto y sus versiones instaladas.
  - Todos los metadatos de un proyecto, como el autor y la licencia, entre otros.
  - Secuencias de comandos que se pueden ejecutar para automatizar tareas del proyecto.

Comandos de NPM

- Para instalar un paquete (por ejemplo axios):

```
C:\temp>npm install axios
```

- Para listar los paquetes locales / [globales] instalados:

```
C:\temp>npm ls [-g]
```

- Para listar donde se encuentran los módulos locales / [globales]:

```
C:\temp>npm root [-g]
```

- Para listar los paquetes instalados y sus dependencias (el 8 se puede cambiar, es el nivel de profundidad):

```
C:\temp>npm ls --depth 8
```

- Para verificar si hay paquetes desactualizados:

```
C:\temp>npm outdated
```

- Para actualizar un paquete a una versión más reciente:

```
C:\temp>npm update axios
```

- Para desinstalar un paquete:

```
C:\temp>npm unistall axios
```

¿Que es un módulo de Node js?

- Básicamente un módulo e**s lo mismo que una dependencia de C#. Es código de terceros que podemos importar y utilizar en nuestro código para resolver algún problema determinado, por ejemplo, la conexión a base de datos**. Todos los Paquetes públicos se pueden encontrar en https://www.npmjs.com/
- Los módulos se instalan en la carpeta `node_modules` dentro del proyecto.
- Los módulos son básicamente bibliotecas en JavaScript.

¿Qué tipos de módulo existen?

- La diferencia que existen entre los módulos y las dependencias de C# es que Node nos permite crear módulos internos dentro de nuestro proyecto o bien, crear módulos privados que no formen parte de mi proyecto, pero que no estan públicos para cualquier usuario.
- Tipos de módulos:
    - Públicos
    - Privados
    - Internos

¿Quienes utilizan Node js?

- Estas son algunas de la empresas que utilizan Node js como parte de la solución en sus plataformas.
  - **Uber**
  - **Netflix**
  - **Wikipedia**
  - **LinkedIn**
  - **PayPal**
  - **Flipboard**
  - **Trello**