JavaScript para servidores escalables

Autor: Rafael Anaya Ragel

ÍNDICE

Introducción

Node.js es un entorno en tiempo de ejecución JavaScript, de código abierto y multiplataforma que se usa para ejecutar código JavaScript por parte del servidor. Su primer lanzamiento fue el 27 de mayo de 2009.

JavaScript, históricamente, se usaba en el lado del cliente, a través de un motor en el navegador del usuario que interpretaba el código de los scripts, el cual venía embebido en el HTML de las páginas webs. Node.js permite que el código JavaScript se ejecute de lado del servidor, lo que ejecuta scripts para generar contenido web dinámico antes de que la página sea enviada al cliente.

Historia

Node.js fue originalmente escrito por Ryan Dahl en 2009, cerca de 13 años después del primer entorno de ejecución JavaScript de lado del servidor. Dahl se inspiró en la barra de progreso de la subida de una imagen en Flicker. El navegador no sabía el progreso de la subida del archivo, por lo que tenía que realizar peticiones al servidor web. Dahl quería una forma más fácil de hacerlo.

Dahl expuso el proyecto en la European JSConf el 8 de noviembre de 2009. Node.js combinado con el motor V8 para JavaScript de Google, un bucle de evento y una API E/S a bajo nivel.

En enero de 2010 se introdujo a Node.js un gestor de paquetes llamado npm. Este gestor facilita a los desarrolladores el publicar y compartir código de bibliotecas(módulos) para Node.js, así como su instalación, actualización y desinstalación.

En junio de 2011, Microsoft y Joyent, implementan una versión de Node.js nativa para Windows. Esta fue la primera versión de Node.js para Windows, ya que desde su lanzamiento sólo era compatible con Linux y macOS.

En enero de 2012, Dahl promocionó al creador de npm, Isaac Schlueter, a gestor del proyecto. En enero de 2014, Schlueter anunción a Timothy J. Fontaine como líder del proyecto.

En diciembre de 2014, Fedor Indutny empezó un fork de Node.js llamado io.js, debido al conflicto interno del gobierno de Joyent. Esta alternativa fue creada como una alternativa con gobierno abierto y un comité técnico separado del mismo.

En febrero de 2015, se anunció un acuerdo entre io.js y Node.js para formar una Node.js Foundation neutral. En junio de 2015, las comunidades de io.js y Node.js votaron para trabajar juntos bajo una Node.js Foundation neutral.

En septiembre de 2015, Node.js v0.12 e io.js v3.3 se fusionaron en Node 4.0.

Perspectiva general

Node.js permite la creación de servidores web y herramientas de red usando JavaScript y una colección de módulos(bibliotecas) que manejan varias funcionalidades. Estos módulos están diseñados para reducir la complejidad de escribir aplicaciones de servidor.

Las aplicaciones Node.js pueden correr en servidores Linux, macOS, Windows, NonStop y Unix. Alternativamente, podrán ser escritas dichas aplicaciones en CoffeeScript, Dart, Microsoft TypeScript o cualquier otro lenguaje que pueda compilar JavaScript.

La principal diferencia entre Node.js y PHP es que las funciones PHP bloquean hasta ser completadas, mientras que las funciones de Node.js están diseñadas para ser no-bloqueantes(usan señales callback).

Arquitectura

Node.js trae consigo el paradigma de programación dirigido por eventos a los servidores web, habilitando desarrollo de servidores web rápidos en JavaScript. Los desarrolladores pueden crear servidores altamente escalables sin necesidad de usar threading. Node.js se creó con el propósito en mente de facilitar la concurrencia en el lado del servidor, ya que la mala gestión de esta conlleva un pobre rendimiento.

Node.js se construyó sobre el motor de JavaScript Google V8, ya que era de código abierto bajo la licencia BSD, extremadamente rápido y competente con HTTP, DNS, TCP… Además, JavaScript era un lenguaje ampliamente conocido por la comunidad, por lo que Node.js fue inmediatamente accesible para los desarrolladores.

Soporte

Existen miles de módulos Node.js, la gran mayoría almacenados en npm. Hay una conferencia anual de desarrollo de Node.js, conocida como NodeConf. La comunidad de software libre ha desarrollado frameworks de servidor para acelerar el desarrollo de aplicaciones. Cabe destacar Express.js, framework para Node.js que se considera “de facto” estándar.

Hay cantidad de soporte para edición de Node.js en muchos IDEs, tales como Microsoft Visual Studio, ATOM, Netbeans, etc…

Detalles técnicos

Como ya comentábamos, Node.js es un entorno en tiempo de ejecución JavaScript que procesa las peticiones en un bucle llamado bucle de eventos.

Threading

Node.js opera en un único thread con llamadas E/S no-bloqueantes, permitiendo dar soporte a miles de conexiones concurrentes sin contar con el coste de cambio de contexto de hilos. Por este motivo, Node.js es altamente escalable.

Un inconveniente de de este acercamiento de único hilo de ejecución es que Node.js no permite escalado vertical incrementando el número de núcleos CPU de la máquina que está corriendo sin tener que usar otros módulos como cluster, StrongLoop Process Manager o pm2. Igualmente, los desarrolladores pueden incrementar el número de threads en la biblitoeca libuv, los cuales serán distribuidos por el kernel en los distintos cores.

Motor

V8 es el motor de ejecución de JavaScript construido para Google Chrome y de código libre creado por Google en 2008. Está escrito en C++ y compila el código JavaScript a código máquina en lugar de interpretarlo en el navegador en tiempo real.

Node.js usa libuv para manejar eventos asíncronos. Libuv es una biblioteca para eventos asíncronos, multiplataforma y programada en C. Parte de Node.js está programado en C y C++, por lo que estas tecnologías se complementan de buena forma.

Gestión de paquetes

npm es el gestor de paquetes pre-instalado para la plataforma Node.js. Se usa para instalar programas Node.js desde el registro del mismo gestor, organizando la instalación y la gestión de paquetes de terceros. No se debe confundir con CommonJS require(), ya que npm no se usa para cargar código, sino para instalar y gestionar dependencias. En la propia página de Node.js dicen de npm que es la mayor fuente de bibliotecas de código abierto del mundo.

Bucle de eventos

Cada vez que un cliente establece una conexión con Node.js se ejecuta un callback. Sin tener que generar un nuevo hilo de trabajo, cada conexión en el entorno de ejecución de Node.js recibe una pequeña asignación de memoria dinámica. El bucle de eventos finaliza cuando no quedan eventos por atender.

Acerca de

Como hemos repetido varias veces, Node.js es un entorno de ejecución asíncrono y enfocado a eventos para construir servidores altamente escalables. Node es similar en diseño y está influenciado por sistemas como Event Machine de Ruby o Twisted de Python.

Las redes basadas en hilos de ejecución, son relativamente ineficientes y muy difíciles de usar. Casi ninguna función en Node hace directamente E/S, así que los procesos nunca se bloquean, por lo que los usuarios no tienen que preocuparse del bloqueo mutuo.

Aunque Node.js está diseñado sin hilos de ejecución, no significa que no se puedan usar varios cores, ya que existen módulos como cluster, que permiten fácilmente la comunicación entre los mismos.

Gobierno

El proyecto Node.js está conjuntamente gobernado por el TSC (Technical Steering Committee), el cual es responsable de guiar el proyecto. El TSC tiene autoridad sobre el proyecto incluyendo:

- Dirección técnica
- Gobierno de proyecto y proceso (incluyendo esta política)
- Política de contribución
- Hosting del repositorio GitHub
- Pautas de conducta
- Mantener la lista de los colaboradores adicionales

Las invitaciones iniciales al TSC se dieron a las personas individuales que habían sido contribuyentes activos y tenían experiencia considerable en la gestión del proyecto.

Grupos de trabajo

Los grupos de trabajo se crean mediante el Core Technical Committee (CTC). Los grupos actuales de trabajo son:

- Website
- Streams
- Build
- Diagnostics
- i18n
- Evangelism
- Docker
- Addon API
- Benchmarking
- Post-mortem
- Intl
- Documentation
- Testing

Por ejemplo, el grupo Website se encarga de mantener la página pública de Node.js, mientras que, por ejemplo, el grupo Documentation, se encarga de mejorar la documentación de Node.js, tanto de la API como de la página web. Este último grupo trabaja muy unido a otros, para conseguir crear la documentación lo más accesible y disponible posible.

Fundación

La misión de la Node.js Foundation es ayudar a acelerar el desarrollo de Node.js y otros módulos relacionados mediante un modelo de gobierno abierto que alenta la participación y la contribución técnica.

Miembros

La Fundación ofrece en su página web un correo electrónico para solicitar información sobre la membresía y un formulario para la solicitud de la misma.

Hay varias categorías de membresía enfocada a corporaciones. También es posible ser miembro de Node.js como persona individual. Algunas empresas miembro de Node.js que brindan a la Fundación un importante aporte son Joyent, IBM, Intel, Microsoft, RedHat…

Bibliografía

- ➔ https://nodejs.org/en/foundation/members/
- ➔ https://en.wikipedia.org/wiki/Node.js
- ➔ https://www.w3schools.com/nodejs/
- ➔ https://github.com/nodejs/node
- ➔ https://cloud.google.com/nodejs/
- ➔ https://www.digitalocean.com/community/tutorials/how-to-set-up-a-node-js-application-for-production-on-ubuntu-16-04