Universidad Abierta Interamericana

Facultad: Tecnología de Tecnología Informática

Carrera: Licenciatura en Gestión de Tecnología Informática

Materia: Programación Web Avanzada

Docente: Teragni Matías

Alumnos: Zatti Dario
Vello Diego
Brustein Ignacio
Bertero Silvio
Rocca, Eliseo

NodeJS es una tecnología para el desarrollo de aplicaciones de propósito general. Básicamente consiste en una plataforma de ejecución de Javascript, lo que se conoce como un "runtime", sobre la que se pueden ejecutar todo tipo de programas.

Por decirlo de algún modo sencillo, **NodeJS** es el lenguaje Javascript sacado del contexto del navegador. Sus creadores supieron valorar la potencia del lenguaje Javascript y usaron el motor "V8" (el motor Javascript open source del navegador Chrome) para crear una plataforma de ejecución capaz de aprovechar el popular lenguaje para acometer una gran variedad de proyectos.

NodeJS, conocido habitualmente también con la palabra "node" a secas, surge en 2009 como respuesta a algunas necesidades encontradas a la hora de desarrollar sitios web, específicamente el caso de la concurrencia y la velocidad.

NodeJS es una plataforma super-rápida, especialmente diseñada para realizar operaciones de entrada / salida (Input / Output o simplemente I/O en inglés) en redes informáticas por medio de distintos protocolos, apegada a la filosofía UNIX. Es además uno de los actores que ha provocado, junto con HTML5, que Javascript gane gran relevancia en los últimos tiempos, pues ha conseguido llevar al lenguaje a nuevas fronteras como es el trabajo del lado del servidor.

En este artículo pretendemos explicar qué es **Node**, para qué se utiliza, por qué es bueno aprenderlo ya y algunos de los proyectos más relevantes creados con esta tecnología, y que muchos de nosotros conocemos.

¿Qué es NodeJS?

"**Node** Yei es", tal como se pronuncia **NodeJS** en inglés, es básicamente un *framework* para implementar operaciones de entrada y salida, como decíamos anteriormente. Está basado en eventos, *streams* y construido encima del motor de Javascript V8, que es con el que funciona el Javascript de Google Chrome. A lo largo de este artículo daremos más detalles, pero de momento nos interesa abrir la mente a un concepto diferente a lo que podemos conocer, pues NodeJS nos trae una nueva manera de entender Javascript.

Si queremos entender esta plataforma, lo primero que debemos de hacer es desprendernos de varias ideas que los desarrolladores de Javascript hemos cristalizado a lo largo de los años que llevamos usando ese lenguaje. Para empezar, **NodeJS** se programa del lado del servidor, lo que indica que los procesos para el desarrollo de software en "**Node**" se realizan de una manera muy diferente que los de Javascript del lado del cliente.

De entre alguno de los conceptos que cambian al estar **Node.JS** del lado del servidor, está el asunto del "Cross Browser", que indica la necesidad en el lado del cliente de hacer código que se interprete bien en todos los navegadores. Cuando trabajamos con **Node** solamente necesitamos preocuparnos de que el código que escribas se ejecute correctamente en tu servidor. El problema mayor que quizás podamos encontrarnos a la hora de escribir código es hacerlo de calidad, pues con Javascript existe el habitual problema de producir lo que se llama "código espagueti", o código de mala calidad que luego es muy difícil de entender a simple vista y de mantener en el futuro.

Otras de las cosas que deberías tener en cuenta cuando trabajas con **NodeJS**, que veremos con detalle más adelante, son la programación asíncrona y la programación orientada a eventos, con la particularidad que los eventos en esta plataforma son orientados a cosas que suceden del lado del servidor y no del lado del cliente como los que conocemos anteriormente en Javascript "común".

Además, **NodeJS** implementa los protocolos de comunicaciones en redes más habituales, de los usados en Internet, como puede ser el HTTP, DNS, TLS, SSL, etc. Mención especial al protocolo SPDY, fácilmente implementado en **Node**, que ha sido desarrollado mayoritariamente por Google y que pretende modernizar el protocolo HTTP, creando un sistema de comunicaciones que es sensiblemente más rápido que el antiguo HTTP (apuntan un rendimiento 64% superior).

Otro aspecto sobre el que está basada **NodeJS** son los “streams", que son flujos de datos que están entrando en un proceso. Lo veremos con detalle más adelante.

Donde se usa NodeJS

Existen varios ejemplos de sitios y empresas que ya están usando **Node** en sitios en producción y algunos casos de éxito que son realmente representativos. Quizás el más comentando sea el de LinkedIn, la plataforma de contacto entre profesionales a modo de red social. Al pasar a **NodeJS**, LindkedIn ha reducido sensiblemente el número de servidores que tenían en funcionamiento para dar servicio a sus usuarios, específicamente de 30 servidores a 3.

Lo que sí queda claro es que **NodeJS** tiene un *footprint* de memoria menor. Es decir, los procesos de **NodeJs** ocupan niveles de memoria sensiblemente menores que los de otros lenguajes, por lo que los requisitos de servidor para atender al mismo número de usuarios son menores. Por aproximar algo, podríamos llegar a tener 1.000 usuarios conectados a la vez y el proceso de **NodeJS** ocuparía solamente 5 MB de memoria. Al final, todo esto se traduce en que empresas grandes pueden tener un ahorro importante en costes de infraestructura.

Otros ejemplos, además de LinkedIn son **eBay**, Microsoft, empresas dedicadas a *hosting* como Nodester o Nodejitsu, redes sociales como Geekli.st, y muchos más.

Por qué Node.JS es una tecnología que se puede usar ya mismo

Node.JS es una plataforma reciente y que ha sufrido muchos cambios a lo largo de su creación. De hecho, en el momento de escribir este artículo aún no se ha presentado la *release* 1.0, por lo que muchos desarrolladores la han tomado en cuenta con cierta distancia. Actualmente se encuentra a disposición la versión 0.8.15.

Inicialmente, es cierto que ha sufrido bastantes modificaciones, un tanto radicales, en su API, lo que ha obligado a diversos profesionales que apostaron por **Node** desde un principio a reciclar sus conocimientos rápidamente y rehacer su código en alguna ocasión. Sin embargo, desde hace tiempo han adquirido el compromiso desde NodeJS a no cambiar el API y continuar con la misma arquitectura, realizando solo cambios a nivel interno.

Esto nos hace entender que es un buen momento para aprender **NodeJS** sin temor a que lo que aprendamos acabe rápidamente en desuso.

Más tecnologías y frameworks basados en NodeJS

No todo termina con **NodeJS**, en la actualidad existen diversos proyectos interesantes que basan su funcionamiento en Node y que nos dan una idea de la madurez que está adquiriendo esta plataforma. Es el caso de proyectos como:

> **Meteor JS:** Un framework Open Source para crear aplicaciones web rápidamente, basado en programación con "Javascript puro" que se ejecuta sobre el motor de Node.JS.
>
> **Grunt:** Un conjunto de herramientas que te ayudan como desarrollador web Javascript. Minifica archivos, los verifica, los organiza, etc. Todo basado en línea de comandos.
>
> **Yeoman:** Otra herramienta, esta vez basada en Grunt, que todavía ofrece más utilidades que ayudan a simplificar diversas tareas en la creación de proyectos, basados en muchas otras librerías y frameworks habituales como Bootstrap, BackboneJS...

Con **NodeJS** se pueden construir aplicaciones web tradicionales, como sitios web. De hecho, existen diversos proyectos de CMS basados en **Node**. Sin embargo, el uso más común de **NodeJS** es el desarrollo de servicios web que devuelven datos en formato JSON, lo que llamamos habitualmente API REST. También con **Node** podemos construir aplicaciones de escritorio multiplataforma, compatibles con Windows Linux y Mac, así como programas de consola. Muchas herramientas del día a día de los desarrolladores están realizadas usando **NodeJS**, desde editores como Atom o VSCode, hasta automatizadores de tareas como Gulp o empaquetadores de assets como Webpack o Rollup.

Node tiene la característica de acometer muchas tareas con poco consumo de recursos, lo que lo hace especialmente interesante para el desarrollo de servicios de alta concurrencia. Puedes aprender mucho más a partir de los contenidos que encontrarás aquí.

Usos de NodeJS

NodeJS es un lenguaje que permite una gran cantidad de aplicaciones, útiles para todo tipo de usuarios, pero sobre todo para desarrolladores.

Lenguaje backend: es una plataforma que permite programación del lado del servidor. Con NodeJS se realiza muy cómodamente servicios web basados en API REST u otras tecnologías como GraphQL. Por supuesto, también es posible desarrollar con Node sitios web tradicionales, incluso existen varios cms basados en esta plataforma.

Programas de consola: la aplicación más directa de **NodeJS** sería la creación de programas de consola. Programas que se ejecutan mediante comandos en el terminal. De hecho, la mayoría de las **herramientas frontend** están desarrolladas con esta plataforma de ejecución de Javascript.

Programas de escritorio: también es notable la aportación de Node.js en la creación de programas y aplicaciones que se ejecutan en los ordenadores de cualquier sistema operativo. Aplicaciones habituales como Visual Studio code, Atom, la aplicación de Slack y muchas otras están realizadas con NodeJS y un framework llamado Electron.

Ventajas

- La compilación de Node.js se realiza en tiempo de ejecución, Just InTime (**JIT**), esto trae consigo una mayor optimización a las funciones que más veces sean llamadas.
- Mediante clusters permite tener una **escalabilidad alta**.
- Podemos expandir nuestro código añadiendo módulos de forma fácil gracias al Node Package Manager (**NPM**).
- Un alto rendimiento en proyectos donde necesitemos **ejecución en tiempo real**.
- En startups o equipos pequeños podremos realizar front-end, back-end y hasta una aplicación móvil con un mismo lenguaje.

Desventajas

API Inestable: La API de Node tiene la mala costumbre de cambiar en formas que rompen la compatibilidad hacia atrás de versión en versión, lo que requiere que apliques cambios frecuentes en tu código para mantener todo funcionando en las versiones más actuales. Aun así, se supone que será más estable desde la versión 0.2.0.

Falta de una Librería Estándar: JavaScript es un lenguaje con un buen núcleo, pero con una flaca librería estándar. Cosas que tomarías por hecho en otro lenguaje del lado del servidor simplemente no está disponible.

Falta de Librerías en General: ¿Necesitas una interfaz de bases de datos madura? ¿Un ORM? ¿Una librería de procesamiento de imágenes? ¿Un analizador XML? Como JavaScript no ha gozado de años de popularidad en el lado del servidor, cosas como estas son o demasiado recientes y no probadas intensamente, o todavía están en camino.

Muchas Formas de Programar: La falta inherente de organización de código se puede considerar una gran desventaja. Se nota su efecto claramente cuando el equipo de desarrollo no está muy familiarizado con la programación asíncrona o los patrones de diseño estándar. Simplemente hay demasiadas formas de programar y de obtener código desparejo y difícil de mantener.

No está Probado lo Suficiente: Este punto puede ser susceptible a opiniones subjetivas debido a que es una cuestión bastante abierta. Mientras no tengamos grandes proyectos en producción por varios años, no podremos saber dónde está el problema. No hay algo obvio de lo que debemos cuidarnos, pero es territorio desconocido con respecto al rendimiento, estabilidad, seguridad y mantenibilidad.

¿Qué es una base de datos gráfica neo4j?

Neo4j es un sistema de gestión de base de datos de gráficos desarrollado por **Neo4j**, Inc. Descrito por sus desarrolladores como una base de datos transaccional compatible con ACID con almacenamiento y procesamiento de gráficos nativos, Neo4j es la base de datos de gráficos más popular según el ranking DB-Engines, y la 22ª más popular base de datos en general.

Base de datos gráfica

La información conectada está en todas partes del mundo que nos rodea. Neo4j fue creado para almacenar, manejar y consultar eficientemente datos altamente conectados en su modelo de datos. Con un modelo de datos potente y flexible, puede representar su información estructurada de forma variable en el mundo real sin pérdida de riqueza.

Neo4j es una base de datos de gráficos nativos NoSQL de código abierto que proporciona un backend transaccional compatible con ACID para sus aplicaciones. El desarrollo inicial comenzó en 2003, pero ha estado disponible públicamente desde 2007.

Usos de las bases de datos de gráficos

En informática, una base de datos de gráficos (GDB) es una base de datos que utiliza estructuras de gráficos para consultas semánticas con nodos, bordes y propiedades para representar y almacenar datos.

Las bases de datos de gráficos son parte de las bases de datos NoSQL creadas para abordar las limitaciones de las bases de datos relacionales existentes.

Tipos de base de datos de gráficos

Top 8 software de bases de datos de gráficos

OrientDB.

Neo4j.

ArangoDB.

Amazon Neptune.

FlockDB.

DataStax.

Cassandra

Titán.

Almacenamiento de datos gráficos

Los datos del gráfico se guardan en archivos de la tienda, cada uno de los cuales contiene datos para una parte específica del gráfico, como nodos, relaciones, etiquetas y propiedades.

Las bases de datos de gráficos no nativas no están optimizadas para almacenar gráficos, por lo que los algoritmos utilizados para escribir datos pueden almacenar nodos y relaciones en todo el lugar.

Los archivos de la base de datos **Neo4j** se conservan en el almacenamiento para una durabilidad a largo plazo. Archivos relacionados con datos ubicados en datos / bases de datos / gráfico. db (v3. x +) por defecto en el directorio de datos **Neo4j**.

NoSQL es un DB gráfico

Las bases de datos de gráficos son bases de datos NoSQL que usan el modelo de datos de gráficos compuesto por vértices, que es una entidad como una persona, lugar, objeto o datos relevantes y bordes, que representan la relación entre dos nodos.

Funcionamiento

Las bases de datos gráficas funcionan almacenando las relaciones junto con los datos.

Una base de datos de gráficos no solo almacena las relaciones entre los objetos de forma nativa, haciendo que las consultas sobre las relaciones sean rápidas y fáciles, sino que le permite incluir diferentes tipos de objetos y diferentes tipos de relaciones en el gráfico.

Ventajas de usar una base de datos gráfica

Aquí, discutimos las principales ventajas de usar bases de datos de gráficos desde el punto de vista de la gestión de datos.

- Pensamiento orientado a objetos.
- Actuación.
- Mejor resolución de problemas.
- Actualice datos en tiempo real y consultas de soporte simultáneamente.
- Entorno de esquema flexible en línea.
- Haga que la consulta de ruta recursiva poderosa sea fácilmente accesible,

Facebook usa la base de datos de gráficos

Social Graph de Facebook, la base de datos subyacente a su motor Graph Search presentado ayer, es solo una de las muchas bases de datos de gráficos que se utilizan para datos complejos y conectados. **Neo4j** es otra base de datos gráfica líder que es de código abierto y utilizada por Cisco, Adobe, Squidoo e Intuit.

Cuándo usar base de datos gráfica

Las bases de datos de gráficos son muy buenas para atravesar relaciones entre pequeñas entidades de datos, pero no son ideales para almacenar muchas propiedades en un solo nodo o valores grandes en esas propiedades.

Big Data

La estructura fundamental para las bases de datos de gráficos en Big Data se denomina "relación de nodo". Esta estructura es más útil cuando debe tratar con datos altamente interconectados. Los nodos y las relaciones admiten propiedades, un par clave-valor donde se almacenan los datos. Una de las bases de datos de gráficos más utilizadas es **Neo4J**.

Modelo de datos gráficos

El modelado de datos gráficos es el proceso en el que un usuario describe un dominio arbitrario como un gráfico conectado de nodos y relaciones con propiedades y etiquetas.

Graph Database

El futuro de las tecnologías de bases de datos. La base de datos de gráficos (wiki) es ahora una palabra de moda, ya que la tecnología está creciendo rápidamente y las empresas no pueden darse el lujo de ignorar esto, ya que debido a los inmensos beneficios, esta tecnología ofrece que se predice con razón como el futuro de DBMS (Sistemas de gestión de bases de datos).

Casos de uso:

- **Detección del fraude:**

Neo4j ya trabaja con varias corporaciones en la detección del fraude en sectores como la banca, los seguros o el comercio electrónico. Esta base de datos puede descubrir patrones que con otro tipo de BD sería difícil de detectar.

Las redes de fraude tienen mecanismos para delinquir que no son detectables con el análisis lineal de los datos. Pero con un análisis escalable de las múltiples relaciones entre los datos, esto es mucho más fácil.

- **Recomendaciones en tiempo real y redes sociales:**

Neo4j permite conectar de forma eficaz a las personas con nuestros productos y servicios, en función de la información personal, sus perfiles en redes sociales y su actividad online reciente. En este sentido, las bases de datos orientadas a grafos son interesantes porque son capaces de conectar personas e intereses.

Con esa información, una empresa puede ajustar sus productos y servicios a su público objetivo y personalizar la recomendación en función de los perfiles. Eso es lo que permite que se aumente la precisión comercial y el compromiso del cliente.

- **Gestión de centros de datos:**

Las bases de datos gráficas son el antídoto perfecto ante el crecimiento desbordante de los datos. La gran cantidad de información, dispositivos y usuarios hacen que las tecnologías tradicionales no puedan gestionar tantos datos.

La flexibilidad, rendimiento y escalabilidad de **Neo4j** permite gestionar, monitorizar y optimizar todo tipo de redes físicas y virtuales pese a la gran cantidad de datos.