Introducción

Tiempo estimado: 15min

Comencemos el curso de fundamentos de Node.js. Tiene como objetivo explicar las bases de Node.js para el desarrollo de aplicaciones bajo esta plataforma. Asume del estudiante conocimientos previos de JavaScript.

Comenzamos la lección introduciendo Node como una plataforma para el desarrollo y la ejecución de aplicaciones escritas con JavaScript. A continuación, describimos el comando node, el núcleo de esta plataforma. Finalmente, presentamos el programa del curso.

Al finalizar la lección, el estudiante sabrá:

- Qué es una plataforma.
- Qué es Node.
- Para qué se puede utilizar Node.
- Para qué se usa el comando node.
- Cuál es el programa del curso.

Introducción

Una plataforma (*platform*) es un entorno de ejecución de software. Node.js, o simplemente Node, es una plataforma para el desarrollo de aplicaciones JavaScript fuera del navegador mediante un modelo asíncrono, dirigido por eventos y no bloqueadora. Se puede utilizar para el desarrollo de aplicaciones webs, servicios webs REST, servidores de aplicaciones, aplicaciones clientes, aplicaciones de escritorio, ejecución de pruebas, monitorización, aplicaciones distribuidas, entre otras muchas cosas.

Fue desarrollado inicialmente por Ryan Dahl, en 2009, y actualmente lo mantiene la Node.js Foundation, entre cuyos miembros encontramos a organizaciones como Google, Groupon, IBM, Intel, Joyent, Microsoft, PayPal, Red Hat, SAP y Yahoo! Está escrito en C/C++ y JavaScript, llevando integrado el motor de JavaScript V8 desarrollado por Google para Chrome. Es gratuito y multiplataforma. Se encuentra disponible para Windows, Linux, Mac o Raspberry Pi. Su sitio web oficial es nodejs.org.

El motor V8 de Google es JIT (*just-in-time*) y muy rápido. JIT significa que analiza el código que tiene que ejecutar y, en vez de interpretarlo, lo convierte a código máquina, ejecutándose entonces desde el código máquina. Inicialmente, la aplicación requiere ese trabajo extra de compilación. Pero una vez compilado, su ejecución es mucho más rápida.

Es utilizado por empresas como Adobe, Amazon, Apple, BMW, Box, citigroup, Dow Jones, eBay, Facebook, GE, General Motors, GitHub, GoDaddy, Groupon, HP, IBM, Intel, Joyent, LinkedIn, Microsoft, MySpace, PayPal, Pearson, Rakuten, SAP, Siemens, Sony, Spotify, Telefónica, The New York Times, Uber, Walmart, Yahoo! y Yammer.

Existe una gran comunidad en creciente expansión. Con billones de descargas mensuales, más de un cuarto de millón de módulos publicados y miles de reuniones de desarrolladores en todo el mundo. Es una de las comunidades más activas, entusiastas y de más rápido crecimiento y aceptación.

Los principales usos de Node son:

- Aplicaciones webs.
- Aplicaciones de escritorio.
- Aplicaciones móviles.
- Aplicaciones de red.
- Aplicaciones intensivas de E/S.
- Aplicaciones de *streaming*.

- Aplicaciones de tiempo real.

Pero ojo, no se recomienda para todo tipo de aplicación. Las intensivas en CPU no suelen ser muy eficientes con Node.

Comando node

El comando node es el intérprete de Node, que permite la ejecución de *scripts* JavaScript. Este intérprete puede funcionar en modo interactivo o *batch*.

Ejecución en modo interactivo

El modo interactivo (*interactive mode*) o REPL (*read-eval-print loop*) permite la ejecución interactiva de proposiciones JavaScript. Cada vez que se introduce manualmente una proposición, ya sea una sentencia o una expresión, Node la ejecuta y muestra su resultado. El intérprete indica que está esperando la entrada de una nueva proposición JavaScript por parte del usuario mediante el prompt, un texto que de manera predeterminada es el carácter >. Si es necesario escribir una proposición en varias líneas, mostrará un *prompt* de segundo nivel, de manera predeterminada, el texto ...

Para entrar en modo interactivo, basta con ejecutar node sin especificar ningún *script* o bien mediante la opción -i. Ejemplo:

```
$ node
> 1+3
4
> "Hello, " + "World!"
'Hello, World!'
> console.log("Hello, World!")
Hello, World!
undefined
> for (var i in [0, 1, 2, 3]) {
...   console.log(i);
... }
0
1
2
3
undefined
> .exit
$
```

Comandos internos del modo interactivo

Un comando (*command*) es una operación con un nombre asociado. Los comandos internos de node comienzan por un punto (.) como, por ejemplo, .help, .exit o .load.

El comando .help muestra una lista de los comandos internos.

El comando .save indica el archivo de historia en el que se debe registrar las proposiciones escritas en el intérprete. Su sintaxis es:

```
.save ruta-de-archivo
```

El comando .exit sale del intérprete.

El comando .load lee el archivo especificado y lo interpreta como si se hubiera escrito directamente mediante la sesión interactiva. Su sintaxis es:

```
.load ruta-de-archivo
```

Combinaciones de teclas del modo interactivo

El intérprete interactivo proporciona varias combinaciones de teclas para realizar determinadas acciones.

Cuando estamos escribiendo una proposición de JavaScript y nos damos cuenta que nos hemos equivocado en algún punto, algo más habitual de lo que aceptaremos nunca, podemos cancelar su escritura pulsando Ctrl+C. Si este Ctrl+C lo hacemos dos veces seguidas, se comportará como el comando interno .exit.

La combinación Ctrl+D tiene el mismo objeto que el comando .exit. Finaliza la ejecución del intérprete.

Se puede utilizar el tabulador para autorrellenar variables o palabras claves. Por ejemplo, si escribimos conso y pulsamos la tecla de tabulado, el intérprete buscará en una tabla interna qué variables o palabras claves comienzan por conso. Si no existe más que una, terminará el identificador, añadiendo le. Si existen varias, mostrará una lista con las posible opciones.

Las teclas ↑ y ↓ permiten movernos por la historia de proposiciones introducidas. Al igual que en PowerShell o Bash.

Variable _

Cada vez que escribimos una expresión de JavaScript en el modo interactivo, el intérprete almacena el resultado de su evaluación en la variable _. Veámoslo mediante un ejemplo:

```
> _
undefined
> [1, 2, 3, 4]
[ 1, 2, 3, 4 ]
> _
[ 1, 2, 3, 4 ]
> _.length
4
>
```

Ejecución en modo batch

Cuando lo que se desea es ejecutar un determinado *script*, hay que pasar su nombre como argumento:

```
$ cat script.js
console.log("¡Hola, Mundo!");
$ node script.js
¡Hola, Mundo!
$
```

Ejecución de proposiciones JavaScript

En algunas ocasiones, puede ser útil ejecutar una o más proposiciones JavaScript con Node. Esto puede hacerse con las opciones -e (o --eval) y -p (o --print). La diferencia entre ambas es si mostrar implícitamente el resultado la expresión dada. Veámoslo mediante unos ejemplos:

```
$ node -e "1+2"
$ node -p "1+2"
3
$ node -e "console.log(1+2)"
3
$ node -p "console.log(1+2)"
3
undefined
$
```

Listado de opciones

Para obtener la lista de opciones, se puede utilizar la opción --help.

Información del curso

Este curso es el primero de varios dedicados a Node o a alguno de sus *frameworks* más utilizados.

Tiene como objetivo presentar los fundamentos del desarrollo de aplicaciones con Node y NPM.

Al finalizarlo, el estudiante sabrá:

- Qué es Node.
- Cómo instalar node.
- Cómo usar Node.
- Qué es NPM.

- Cómo instalar y configurar npm.
- Cómo usar NPM.
- Cómo instalar paquetes de NPM.
- Cómo publicar paquetes en el repositorio oficial de NPM.
- Cómo desarrollar paquetes y aplicaciones con Node y NPM.
- Cómo acceder al sistema de ficheros con Node.
- Cómo funciona el modelo asíncrono de Node.
- Cómo usar el modelo de eventos de Node.
- Cómo depurar con el inspector V8.

Conocimientos previos

El estudiante debe tener conocimientos de JavaScript. Recomendándose la especificación ES2015 o una superior.

Plan del curso

El curso tiene una duración aproximada de 9 horas. Se divide en 11 lecciones, cada una de ellas con una parte de teoría y generalmente una de práctica Al finalizar el curso, puede realizar un examen de tipo test, con el que evaluar los conocimientos adquiridos.

El enfoque a seguir es muy sencillo: ir lección a lección; primero hay que leer la teoría y, después, realizar la práctica. Se recomienda encarecidamente que el estudiante realice cada lección, tanto teoría como práctica, en el mismo día, con el menor número de interrupciones a lo largo de su estudio. Al finalizar el curso, se recomienda realizar el examen.

A continuación, se enumera las distintas lecciones y el tiempo estimado para su estudio:

Lección	Teoría	Práctica	Descripción
1 Introducción	15min	20min	Esta lección.
2 Módulos	20min	15min	Cómo desarrollar y utilizar módulos en Node.js.
3 Introducción a NPM	35min	20min	Introducción al sistema de paquetes NPM usado ampliamente en Node.
4 Desarrollo de paquetes	30min	20min	Cómo desarrollar nuestros propios paquetes.
5 Aplicaciones Node	15min	15min	Cómo desarrollar aplicaciones de Node.
6 Publicación de paquetes	15min	20min	Cómo publicar paquetes en el repositorio de NPM.
7 Modelo asíncrono	25min	15min	Describe detalladamente el modelo asíncrono usado por Node.
8 Módulo fs	20min	15min	Describe las funciones más utilizadas del módulo integrado fs para trabajar con el sistema de ficheros.
9 Módulo path	10min	-	Describe cómo trabajar con rutas del sistema de ficheros mediante el módulo integrado path.
10 Modelo de eventos	15min	15min	Describe detalladamente el modelo de eventos utilizado por Node.
11 Depuración de Node.js	30min	45min	Cómo depurar código mediante el inspector V8.
Examen	60min		Evaluación de los conocimientos adquiridos.

Información de publicación

Título Fundamentos de Node.js

Autor Raúl G. González - raulggonzalez@nodemy.com
Primera edición Noviembre de 2016
Versión actual 1.0.0
Versión de Node.js 7.0
Contacto hola@nodemy.com

