Bootcamp Backend, Node JS: Lectura 2

Ing. Pedro Rotta

Universidad de Piura - Vida Universitaria

Febrero-2022

¿Qué es Node Js?

Es un entorno de ejecución para javascript. Un entorno de ejecución es un software que permite ejecutar javascript en el sistema operativo.

Esto hace que node permita trabajar con diferentes APIs para obtener objetos enlazados al sistema operativo, como carpetas, archivos, servidores, bases de datos, etc.

Respecto al fundamento de su usabilidad, parte de la premisa que es asíncrono. Es decir su arquitectura permite ejecutar un método de la cadena de código mientras se solicita información de otro método.

Instalación

Para instalar node debemos ir a la página web de nodejs : `https://nodejs.org/`.

Para verificar la versión de node escribimos en el terminal:

```
node --version
```

Diferencia con el navegador

Si se desea correr un programa de JS usando node, debemos saber que no existe el *document object*. Esto es, debido a que son objetos vinculados al navegador, no al sistema operativo.

En el sistema operativo, JS trabaja con otros objetos como archivos o datos, a través de diferentes módulos. Un **módulo** es un archivo que contiene objetos y métodos para su uso en aplicaciones que correran en node.

Módulos

Un módulo es un archivo que permite guardar variables, objetos y métodos para que las aplicaciones usen esta información de manera que no se escriben las mismas líneas de código todas las veces, sino que se pueden reutilizar. En python serían similar a una librería.

Un módulo puede ser importado o generado por nosotros mismos. En primer lugar, veremos cómo crear un módulo.

Supongamos que deseamos tener un módulo con la siguiente función:

```
function log(message){
   console.log(message);
}
```

En el archivo exportamos esta función, añadiéndolo al objeto module como:

```
module.exports.log = log;
```

Objeto module

Para analizar la exportación, debemos analizar qué es el objeto module, al que se hace referencia en el procedimiento anterior. Para analizarlo, corramos un programa con el siguiente comando:

```
console.log(module);
```

Es un objeto de javascript, que se asigna a cada archivo js y que referencia propiedades de cada archivo para ser utilizadas en cualquier otro proyecto o parte de proyecto referido.

En su elemento exports, es en donde se colocan todas las funciones y propiedades que desean volverse públicas del módulo que deseamos exportar.

Privado vs Público

Cuando se desea analizar la exportación de un módulo, es importante tener en cuenta que:

Cada variable o función no declarada en el atributo exports del objeto module, es una variable o método **privado**, debido a que el alcance de ello solo será posible en el propio archivo señala.

Las variables u objetos que están dentro de exports, en el objeto módulo, son variables o métodos **públicos**. Una variable o un método público signiica que puede ser usado en cualquier otro archivo del proyecto

Importando un módulo

Para importar un módulo, utilizamos, dentro del archivo en el que vamos a importar el módulo, la función **require**.

Si el módulo está en la misma carpeta, usamos:

```
require('./module.js')
```

Si el módulo está en una subcarperta, usamos:

```
require('./subcarpeta/module.js')
```

Si el módulo está en una carpeta superior, usamos:

```
require('../carpetasuperior/module.js')
```

Para poder acceder al método o variable que se necesita, se utiliza:

```
module.function();
```

Módulo OS

El módulo os es el módulo que permite trabajar con el sistema operativo. algunos de sus métodos son:

userInfo() Devuelve información sobre el usuario actual. **uptime()** devuelve el tiempo de actividad de sistema. **type()** devuelve el tipo de sistema operativo, **release()** devuelve la versión, **totalmem()** devuelve la memoria totaly **freemem()** devuelve el espacio libre disponible.

Módulo Path

Otro módulo es el módulo asociado con las rutas de la PC. Tiene algunos métodos interesantes, como **sep** para verificar el separador, **join** para unir diferentes rutas, **basename** para saber el nombre del archivo en una ruta larga, **resolve** para unir la ruta raiz con una ruta formada.

Módulo de archivos del sistema

El módulo de archivos de sistema, tiene 2 métodos de interés, **readFile** y **writeFile**

Estos métodos se presentan de manera asíncrona, que quiere decir que necesitan una función de llamada para responder con un valor.

Mientras la función de llamada se ejecuta, el flujo del código permanece normal, hasta que se cumple la función.