Introducción a Node.js

¿Qué es Node.js?

Node.js es un entorno de programación que corre en el servidor basado en Javascript.

Como sabemos, Javascript es un lenguaje de programación ejecutado en el servidor por diferentes máquinas virtuales.

Node.js utiliza la máquina virtual de Google Chrome: V8 para su ejecución; es decir, tiene las mismas funcionalidades de Javascript que posee en el navegador de Google.

Lo primero a tener en cuenta cuando trabajamos en Node.js es que es un entorno que nos permite realizar aplicaciones en tiempo real.

Esto no lo permite Javascript al ser un lenguaje asíncrono, pero si tenemos un solo proceso síncrono en nuestro código, toda nuestra aplicación será síncrona.

Node.js es capaz de hacer todo lo que deseemos, pero en algunos puntos no se la llevara muy bien, siempre debemos enfocarnos en que su principal función es realizar aplicaciones en tiempo real.

¿Que es NPM?

NPM (Node Package Modules) es el gestor de paquetes en Node.js por excelencia. Es de fácil uso, pensado para programadores.

En NPM tenemos diversos comandos, el principal es “install” con el cual podemos instalar paquetes de manera global o local.

Entre lo que podemos hacer está:

- Actualizar nuestro archivo package.json y/o instalar las dependencias que contiene este.
- Buscar paquetes
- En que ya viene incluido en la instalación de las últimas versiones de Node.js.

Su estructura es:

npm <comando>

Para mas informacion consulta la pagina de npm: [**http://npmjs.org**](http://npmjs.org/)

¿Qúe es el archivo package.json?

Package.json será el archivo que identificara nuestro proyecto en este guardaremos el nombre de la aplicación, la versión y las dependencias de la siguiente forma.

{
    "name": "nombre de la app",
    "version": "0.0.1",
    "dependencies":
        "express": "\*"
}

En dependencies colocarás el nombre de la dependencia y la versión de la misma para que luego cuando pases tu proyecto a otro colaborador.

Él sólo tendrá que ejecutar en terminal:

npm install

Y todas las dependencias se instalarán automágicamente.

Ahora cada vez que instales una nueva dependencia deberás ejecutar el siguiente comando para que dicho archivo se actualice:

npm install  <paquete> --save

Y cuando desees instalar un paquete de manera global, solo tendrás que hacer:

npm install -g <paquete>

Iniciando con node.js

Para iniciar con node.js depues de haber creado nuestro package.json debemos crear el archivo Javascript que hara el papel del servidor, en este caso lo llamaremos server.js, pero el nombre no importa.

En este archivo importamos todas las dependencias  a usar y colocaremos toda la lógica que irá del lado del servidor, un ejemplo de este puede ser:

**var** express = **require**('express');
**var** server = express();

server.**get**('/', **function** (req, res) {
           res.send('Hello World');
});

server.listen(3000);
console.log('Servidor corriendo en [**http://127.0.0.1:3000')**](http://127.0.0.1:3000')/);

Con este servidor le estamos diciendo a node que al recibir un request en la  url raíz del proyecto, devuelva un hola mundo; pero veamos de manera detallada para que sirve cada parte de nuestro código.

**Require()**→ Con require(), le decimos a nuestro server que librerías Node vamos a cargar; librerías que previamente ya están en las dependencias de nuestro proyecto.

En nuestro servidor debemos definir qué hacer con las peticiones que llegan de los usuarios (GET - POST), en este caso usamos la ayuda de “Express.js”; si no se define qué hacer con las peticiones tendríamos un server corriendo que no hace nada.

**.get()** →  Le decimos al server que hacer cuando recibe una petición GET en el root de nuestra aplicación, por medio de funciones que siempre van a recibir como parámetros una petición del cliente (req) y una respuesta del servidor (res).

**res.send()**→ Le decimos que haga un response con la cadena “hola mundo”.

**server.listen(xxxx)**→Le decimos al servidor el puerto en el que debe escuchar peticiones.