Desarrollo del lado servidor con NodeJS, Express y MongoDB

NodeJS y NPM

Javascript surge en el año 1995 de la mano de Brendan Eich, de Netscape. Como ya sabemos, es un lenguaje orientado a objetos, basado en prototipos y dinámico. Se utiliza del lado cliente, en los navegadores, para darle comportamiento a los diferentes elementos que están en la web, así como también para realizar animaciones y demás.

Su popularidad fue creciendo tanto que, desde el 2012, todos los navegadores modernos lo soportan.

Si bien nació para vivir en los navegadores, poco tiempo después de su creación, se realizó una versión que corría en el lado servidor, también llamada versión de escritorio. Pero recién por 2005 se popularizaron las implementaciones de Javascript para el lado servidor. Nodejs[1] es una de esas implementaciones más populares y con más adeptos en la actualidad.

El motor de Javascript se llama V8 y es el que utilizan Chrome y Nodejs. Nodejs usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. Este modelo contrasta con el tradicional modelo de hilos, donde los servidores web crean un hilo para atender una petición web y se bloquean hasta finalizar la tarea que corresponda. Crear un hilo implica dedicar una parte pequeña de memoria RAM, supongamos 2 MB. Aquí tenemos una primera limitación del esquema tradicional ya que contamos con una limitante de memoria para la atención de peticiones, que pueden ser usuarios concurrentes en nuestro sitio web, por ejemplo. Esto no sucede en NodeJS con lo cual nos permite tener sistemas escalables con mayor facilidad de gestión.

[1] https://nodejs.org/es/about/

Una de las herramientas más importantes para desarrollar aplicaciones con NodeJS es NPM[2]. NPM nos permite gestionar librerías Javascript para incluirlas en nuestro ambiente de desarrollo o proyecto. En su sitio indican que contienen más de 600.000 paquetes, que no son más que bloques de código que nos ayudan a resolver problemas de desarrollo, testing o producción. NPM puede ser utilizado para descargar paquetes manualmente, vía el comando *npm install xxx.* Pero en un proyecto Node, vamos a manejar las dependencias con los paquetes que necesitemos, vía un archivo especial llamado *package.json.* Allí se indicarán los nombres de los paquetes, versiones, descripciones, dependencias tanto de desarrollo como de producción y otras cosas más que iremos explorando. Con ese archivo, NPM puede instalar todas las dependencias de un proyecto y dejar la aplicación lista para su ejecución.

Una vez que terminemos de instalar NodeJS, NPM se habrá instalado *mágicamente.*

Comencemos a utilizarlo.

[2] https://docs.npmjs.com/getting-started/what-is-npm