Desarrollo de Aplicaciones Web

Tema Nº8:NODE JS - FUNDAMENTOS

Indicador de logro Nº8:Aplica fundamentos e intérprete JavaScript del lado del servidor a través de Node.JS en una arquitectura orientada a eventos y basado en el motor V8.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº8:**

NODE JS – FUNDAMENTOS

**Subtema 8.1:**

Node JS

Node.js es una plataforma para ejecutar JavaScript sin necesidad de un navegador, permite a los desarrolladores con experiencia en JavaScript crear aplicaciones que se pueden ejecutar línea de comandos o de lado de servidor.

Internamente Node.js está construido con el motor V8 de Chrome lo que permite soporte de todos los métodos de ECMAScript.

**Diferencias entre Node.js y Browser**

* En Node.js no existen los objetos document y window que son propios del navegador.
* En Node.js se puede usar ES6 de forma nativa, no depende del navegador del cliente.
* En Node.js no se necesita Babel para convertir a versiones anteriores de ECMAScript.
* En Node.js se puede utilizar require() para importar módulos.
* En Node.js existen módulos adicionales cómo fs, http entre otros.

En resumen, podemos crear aplicaciones con JavaScript de manera semejante a otros lenguajes como PHP, Perl, etc.

**Subtema 8.2:**

Como utilizar Node.js

Para usar Node.js debes tener instalado el motor de ejecución, este motor depende de tu sistema operativo y lo puedes descargar en https://nodejs.org/es/download/

Se puede interactuar con el motor de Node.js en línea de comandos.

Para verificar la instalación:



Para ejecutar una aplicación en Node.js se utiliza el nombre del archivo .js:



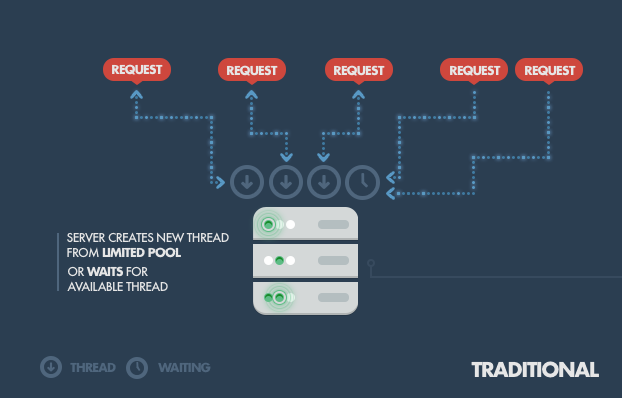
**Subtema 8.3:**

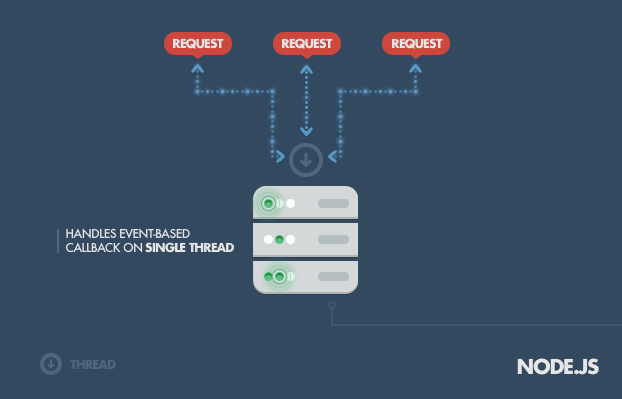
¿Como funciona?

La idea principal de Node.js: uso no-bloqueante, event-driven I/O, permanecer ligero y eficiente en la superficie del uso intensivo de datos en tiempo real de las aplicaciones que se ejecutan en dispositivos distribuidos.

Lo que en realidad significa es que Node.js no es nueva plataforma que dominará el mundo del desarrollo web. Al contrario, se trata de una plataforma que llena una necesidad en particular.

Y este entendimiento es absolutamente esencial. Definitivamente no quieres usar Node.js para operaciones intensivas de CPU; de hecho, utilizándolo para el cálculo pesado anulará casi todas sus ventajas. Donde Node REALMENTE destaca es en la construcción rápida y escalable de aplicaciones de red, debido a que es capaz de manejar un gran número de conexiones simultáneas con alto rendimiento, lo que equivale a una alta escalabilidad.





**NPM: El Node Package Manager**

Cuando hablamos de Node.js, una cosa que definitivamente no debe omitirse es integrarlo en el apoyo de la gestión de paquetes utilizando la herramienta NPM que viene por defecto con cada instalación de Node.js. La idea de los módulos NPM es muy similar a la de Ruby Gemas: un conjunto de componentes reutilizables disponibles públicamente a través de una fácil instalación a través de un repositorio en línea, con la versión y la dependencia de gestión.

Una lista completa de los paquetes de módulos puede encontrarse en el sitio web de NPM Https://npmjs.org/ o acceder utilizando la herramienta de la CLI de NPM que automáticamente se instala con Node.js. El módulo es un ecosistema abierto a todos, y cualquiera puede publicar su propio módulo que será incluido en el repositorio de NPM. Una breve introducción a la NPM (un poco viejo, pero sigue siendo válido) se puede encontrar en <http://howtonode.org/introduction-to-npm>.

Una de las ventajas de Node.js es que existe un [repositorio de módulos](http://www.senchalabs.org/connect/) que resuelven diferentes tareas y que han sido creados por la comunidad (https://www.npmjs.com/). La lista de módulos adicionales utilizados se registra en el archivo *package.json*, mientras que el contenido de estos componentes se descargan y almacenan en la carpeta *node\_modules*.

Para manejo estos módulos se utiliza comando npm que viene incluido en la instalación de Node.js.

Para verificar la versión de npm:



Para inicializar el archivo *package.json*:



Para instalar un módulo:



Algunos de los más populares hoy en día son módulos de NPM:

* [express](http://expressjs.com/) - Express.js, inspirado en el framework de desarrollo web para Node.js, y el estándar de facto para la mayoría de aplicaciones Node.js de hoy en día.
* [connect](https://github.com/mranney/node_redis) - Connect es un servidor HTTP extensible framework para Node.js, que proporciona una colección de alto rendimiento de plugins conocidos como middleware; sirve como fundamento para expresar.
* [socket.io](https://npmjs.org/package/underscore) y [sockjs](https://github.com/sockjs) - Componente del servidor de los dos componentes de websockets más comúnes en la actualidad.
* [Jade](http://jade-lang.com/) - Uno de los más populares motores de plantillas, inspirados por HAML, un defecto en Express.js.
* [mongo](https://www.npmjs.com/) y [mongojs](https://github.com/gett/mongojs) - mongoDB wrappers para proporcionar la API para bases de datos de objetos MongoDB en Node.js.
* [redis](https://npmjs.org/package/coffee-script) - Redis biblioteca cliente.
* [coffee-script](http://socket.io/) - CoffeeScript compilador que permite a los desarrolladores escribir sus programas Node.js con café.
* [Underscore](https://npmjs.org/package/mongodb) ([lodash](http://lodash.com/), [lazy](https://npmjs.org/package/lazy)) - La biblioteca de utilidades más popular de JavaScript, empaquetados para ser utilizado con Node.js, así como sus dos contrapartes, que prometen mejorar el rendimiento mediante la adopción de un enfoque de aplicación ligeramente diferente.
* [forever](https://npmjs.org/package/forever) - Probablemente la utilidad más común para asegurar que un determinado Node script se ejecuta continuamente. Mantiene su proceso de Node.js en la producción y en el rostro de cualquier fallo inesperado.

**1. MATERIALES**

Para la experiencia a realizar se requiere lo siguiente:

1. EQUIPO

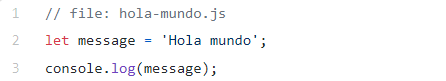
Se requiere tener un equipo de cómputo (PC o laptop).

1. HERRAMIENTAS

* Visual Stude Code
* Node.js
* Npm
* JavaScript

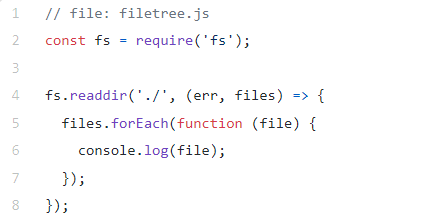
**2. PROCEDIMIENTO**

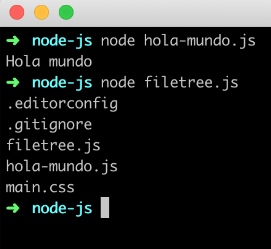
Como primer ejemplo vamos a utilizar la función **console.log** que devuelve un texto a la consola del usuario.



Cómo segundo ejemplo vamos a listar los archivos dentro de una carpeta, para ello usamos el módulo fs de Node.js que permite acceder a los archivos y carpetas del sistema operativo.

Para utilizar el módulo lo importamos con el método **require()**.





Como tercer ejemplo utilizamos el módulo http que permite manejar el protocolo HTTP y por consiguiente crear un servidor Web, en el ejemplo creamos un servidor web en el puerto 3000 y a todas las peticiones le devolvemos el mismo texto.

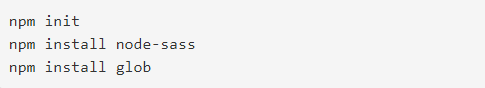


El resultado se puede revisar con un navegador en http://127.0.0.1:3000/

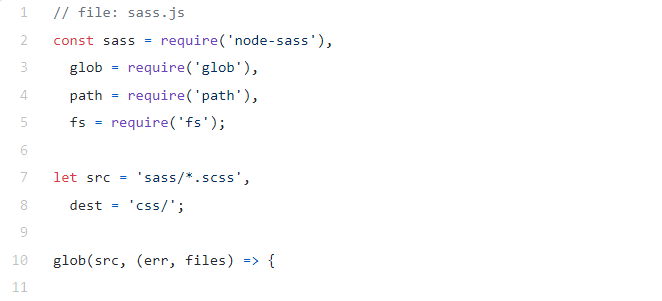
Como siguiente ejemplo vamos a crear un script que compile los archivos Sass de una carpeta, para ello vamos a utilizar dos modulos adicionales.

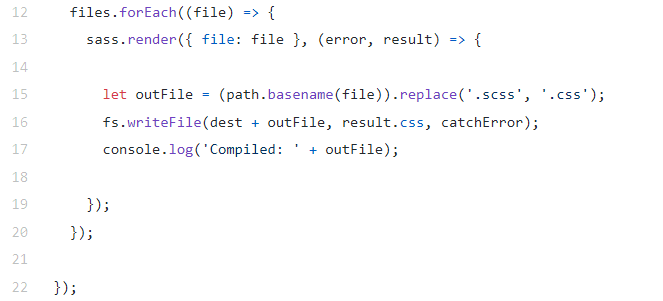
* **nodes-sass** incluye el preprocesador de Sass en Node.js
* **glob** permite buscar archivos de usando patrones.

Inicializamos npm para el proyecto e instalamos los módulos adicionales:

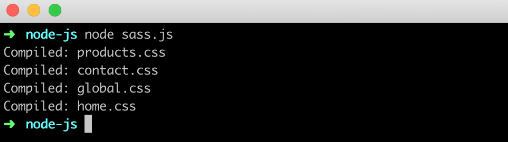


Una vez instalados los módulos creamos el script para compilar Sass: primero se utiliza el módulo **glob** para buscar los archivos con la extensión *.scss*, recorremos uno a uno los archivos utilizando el módulo **path** para obtener el nombre del archivo y renombrarlo con la extensión .css, finalmente utilizamos el módulo **node-sass** para generar los archivos CSS.





Para simplificar el ejemplo hemos obviado verificar si existen errores en los procesos.



**Conclusiones**

Con Node.js se puede ejecutar JavaScript sin necesidad de un navegador aprovechando las bondades de ES6 de forma nativa, además se puede acceder a módulos extras creados por la comunidad que permiten automatizar tareas e incluso crear aplicaciones complejas.

**ACTIVIDAD VIRTUAL:**

1. **CUESTIONARIO TÉCNICO**

Revisa y analiza el tema desarrollado en la presente sesión, luego responde las siguientes preguntas propuestas:

* ¿Qué es y para qué sirve Node JS?
* ¿Cómo funciona el Node JS?
* ¿Es Node JS un lenguaje de programación? De una breve explicación
* Realice una App aplicando todo lo aprendido de esta guía

1. **ENLACES DE REFERENCIA**

* https://www.kodetop.com/tutorial-basico-de-node-js/
* https://www.toptal.com/nodejs/por-que-demonios-usaria-node-js-un-tutorial-caso-por-caso
* https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/introduccion-a-nodejs/

1. **CONCLUSIONES DE LA EXPERIENCIA**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_