Hello NodeJS

Paso a paso para comenzar con nodeJS

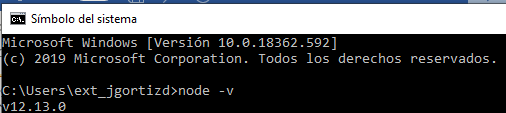
Es necesario tener previamente instalado NodeJS y algún editor de código como Visual Studio Code.

# Verificar la instalación de node

Abrir la línea de comandos de Windows (CMD) y escribir:

node -v

Al presionar Enter debe retornar la versión de NodeJS instalada:

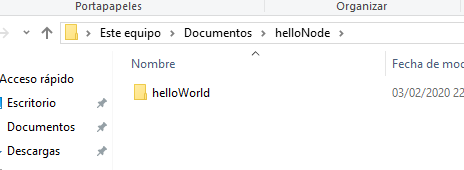


Lo que nos indica que tenemos la versión 12.13 instalada.

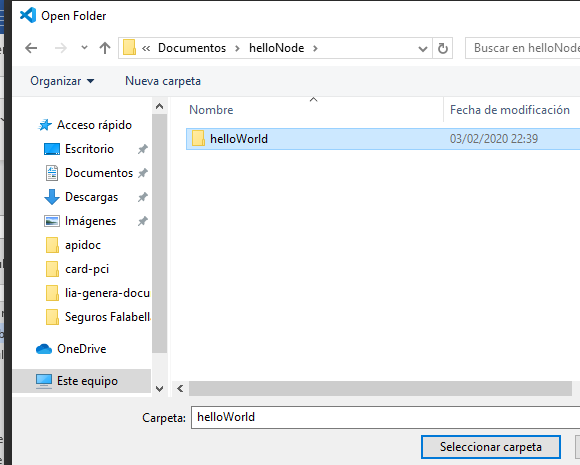
# Preparar ambiente de trabajo

Acá crearemos las carpetas que usaremos para nuestro proyecto, así como abriremos nuestro editor de código:

1. Crear una carpeta llamada helloNode dentro de alguna otra carpeta que tenga permisos de lectura y escritura, como Mis Documentos (En la mayoría de casos).
2. Dentro de helloNode crear otra carpeta llamada helloWorld.



1. Abrir Visual Studio Code. Ubicar el menú File en la parte superior izquierda y luego Open Folder, por último seleccionar la carpeta helloWord y darle en Seleccionar carpeta:



Haciendo un Hola Mundo!

Hacer un hola mundo con node es simple, para ello debemos hacer los siguientes pasos:

1. Hacer clic en File y luego New File en el menú superior lado izquierdo:



1. Dentro del archivo que se creó escribir lo siguiente:

console.log('Hola Mundo!');

1. Guardar el archivo presionando ctrl + S y llamarlo holaMundo.js
2. En el menú superior hacer clic en Terminal y luego en New Terminal y dentro de la nueva terminal escribir:

node holaMundo.js

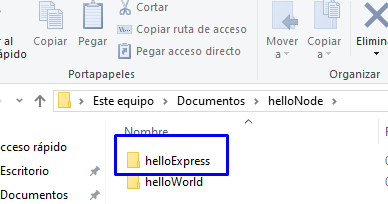
Al presionar enter debe mostar:



# Haciendo un Hola Mundo con Express

Al utilizar express haremos una API que a través del método GET devuelva el mensaje Hola Mundo! En formato JSON.

1. Dentro de la carpeta helloNode crear una carpeta llamada helloExpress.



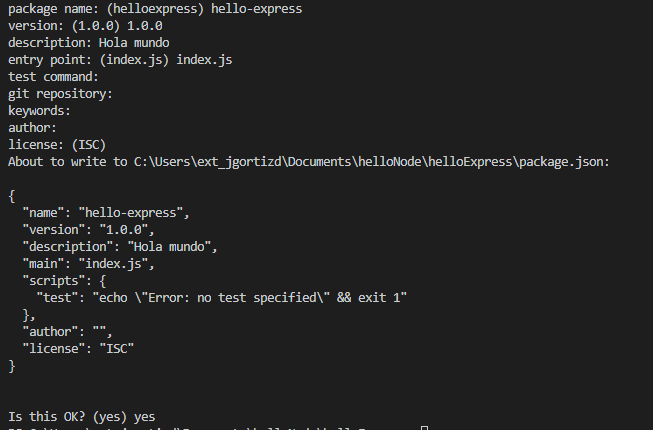
1. Abrir la carpeta desde Visual Studio Basic como se realizó en el ejercicio previo.
2. Abrir una nueva terminal desde el menú de visual studio, allí escribir:

npm init

1. Se van a desplegar una serie de preguntas que nos permitirán inicializar nuestro proyecto con express.

* Package name: hello-express
* Versión: 1.0.0
* Description: Hola mundo
* Entry point: index.js
* Test command: (Vacío)
* Git repository: (Vacío)
* Keywords: (Vacío)
* Author: Tu nombre
* Licence: ISC
* Yes

Debería quedar la línea de comandos como la siguiente imagen:



Una vez finalizado se creó el archivo package.json que nos permitirá iniciar nuestra aplicación utilizando npm y el manejo de paquetes para nuestra aplicación.

1. Procedemos a crear el archivo index.js indicado como el archivo principal de nuestra aplicación durante el paso anterior, hacemos clic en new file, lo guardamos con ctrl + s y de nombre le colocamos index.js
2. Ahora procedemos a agregar el código de nuestra aplicación al archivo index.js:

* Importamos el framework Express:

var express = require('express');

* A pesar que ya lo importamos a nivel de archivos aún no existe, para ello debemos instalarlo, abriendo la terminal de Visual Studio y escribiendo:

npm install express

* Inmediatamente se comenzará a instalar Express y además se creará automáticamente la carpeta node\_modules que será donde estará guardado así como se editará automaticamente el archivo package.js agregandose dentro de dependencias el nombre de la librería utilizada así como la versión utilizada.

1. Creamos una aplicación de express con la siguiente línea (archivo index.js):

var express = require('express');

var app = express();

1. Ahora publicamos nuestro primer endpoint, el cual será por el método get:

var express = require('express');

var app = express();

app.get('/', function (req, res) {

res.status(200).json({message: 'Hello World!'});

});

Dentro de el se le dice a la app que cuando reciba una solicitud por el método GET en la ruta / llame a una función la cual recibe dos parámetros de entrada, un req que se refiere a request o solicitud, en ella se contiene gran parte de la información sobre la petición que se hizo y también está res que hace referencia a response y es la variable a través de la cual enviaremos una respuesta al usuario que hizo la solicitud.

1. Por último, publicamos nuestra aplicación en el puerto 3000 para que pueda ser consumida desde otras aplicaciones:

var express = require('express');

var app = express();

app.get('/', function (req, res) {

  res.status(200).json({message: 'Hello World!'});

});

app.listen(3000, function () {

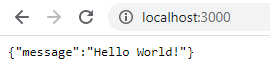
  console.log('Example app listening on port 3000!');

});

1. Para probarla ejecutamos el comando:

node index.js

Luego abrimos algúna ventana de navegador Web e ingresamos a la URL localhost:3000 y nos debe retornar como respuesta:



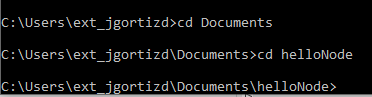
# El generador de aplicaciones de Express

Express para simplificar el desarrollo de aplicaciones y bajar el tiempo de desarrollo provee un gestor de plantillas preconfigurado donde con tan solo un comando podemos tener listo el ejercicio anterior. Para usarlo debemos instalarlo a nivel global, es decir, como dependencia de node que pueda ser utilizado fuera del proyecto. Debemos ejecutar el comando:

npm install -g express-generator

La -g le indica al gestor de paquetes que este en específico no se instalará en un proyecto sino en nuestro computador. Esto nos permitirá usarlo desde cualquier sesión de la linea de comando y en cualquier ubicación con permisos de lectura y escritura.

Una vez termine la instalación, procedemos a probar para ello abrimos la línea de comando de Windows y nos transladamos utilizando los comandos dir y cd hasta la carpeta helloNode



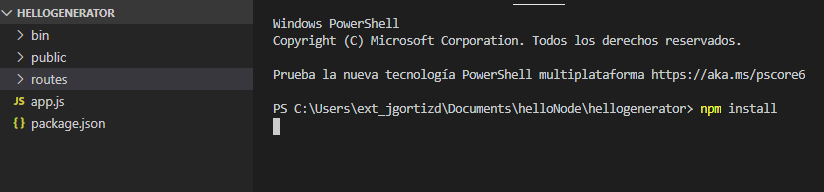
Estando ya en la ruta indicada, ejecutar el comando

express –no-view hellogenerator

Allí se creará una carpeta con una serie de archivos que contienen una aplicación simple de Hola Mundo. Para ejecutarlo debemos abrir la carpeta hellogenerator desde Visual Studio y en la terminal ejecutar el comando:

npm install

Este comando descargará todas las dependencias del proyecto lo cual nos permitirá poder ejecutarlo.



Una vez termine ya podemos iniciar nuestra aplicación ejecutando el comando:

npm start

Y una vez ejecutado entramos al navegador web a la dirección localhost:3000