

TUTORIAL NODE Y NPM



José Carlos Mora

11/11/2025

2ºDAM

Manual

Introducción.....	3
Instalación de Node.js y NPM.....	3
• Paso 1: Descargar NodeJS.....	3
• Paso 2: Instalar.....	3
• Paso 3: Verificar instalación.....	4
Uso básico de Node.js.....	4
Configuración de variables de entorno (PATH).....	4
Módulo HTTP en Node.js.....	5
Próximos pasos.....	7
Conclusión.....	7

Introducción

Node.js es un entorno de ejecución diseñado para ejecutar código **JavaScript** del lado del servidor.

Permite desarrollar aplicaciones escalables, rápidas y eficientes utilizando un solo lenguaje tanto en el cliente como en el servidor.

NPM es el gestor de paquetes oficial de **Node.js**. Facilita la instalación y administración de dependencias, librerías y frameworks, permitiendo reutilizar código y mantener los proyectos organizados.

El módulo **HTTP** ofrece las herramientas necesarias para crear servidores web y manejar peticiones y respuestas sin necesidad de instalar software adicional.

Por último, la variable **PATH** es clave para que el sistema operativo reconozca los comandos de **Node.js** y **NPM** desde cualquier ubicación en la terminal.

Instalación de Node.js y NPM

- **Paso 1: Descargar NodeJS**

- Vamos al sitio oficial: <https://nodejs.org>
- Usamos la versión LTS, junto con la versión más actualizada ya que la current, la cual es la otra versión puede ser muy inestable
- Descargamos la versión ya mencionada antes y el archivo.exe de 64 bits

- **Paso 2: Instalar**

- Ejecutamos el instalador y seguimos las instrucciones, es decir, “siguiente”, “siguiente”, hasta que salga finalizar
- A pesar de apenas tener que mirar nada, tenemos que estar pendientes de no saltarnos la casilla de “Add to PATH” y asegurarnos de marcarla, ya que si nos la saltamos, después es muy lioso poder agregarlo al PATH, porque pueden ocurrir errores

- **Paso 3: Verificar instalación**

- Cuando hayamos hecho los pasos anteriores, el siguiente paso es abrir el terminal o “cmd”, como quieras llamarlo, y ejecutar
 - **node -v**
 - **npm -v**
- Si vemos las versiones de ambos, está todo instalado correctamente, en caso contrario, algún paso habremos hecho mal, como puede ser seguramente el caso de agregar al PATH (vuelve al paso 1 para arreglar esto)

Uso básico de Node.js

Luego de verificar la instalación, comprobamos que funciona en Visual Studio Core creando nuestra carpeta, llamándola como quieras, por ejemplo 02ModulesIPM y creamos un archivo .js.

Dentro de ese mismo archivo creamos un simple código para que salga por consola:

- **console.log("¡Buenísimos días Javier!");**

Este mismo archivo lo ejecutamos desde el cmd para ver si de verdad funciona, por lo que escribimos:

- **node server.js**

Si te sale el mensaje que escribimos por pantalla, enhorabuena, tu Node ya funciona y si no te va, el problema está en el PATH (hasta poesía me sale enseñándote a instalar Node).

Antes te dije que si te pasaba un error con el PATH, volvieses al paso 1, pero no, mentía, porque ahora te voy a explicar como solucionar eso (Por este tipo de razón, no te puedes fiar de nadie y leer siempre documentos, ya sea tu hipoteca o un simple tutorial como este)

Configuración de variables de entorno (PATH)

Si estás aquí es porque no te va el Node y es por el PATH o simplemente que estás bajando el documento y te lo has encontrado. Pero en cualquier caso, te voy a enseñar cómo solucionar esto.

Primer paso:

- Saber que es PATH
 - Es una variable del sistema que indica las rutas donde se buscan los ejecutables

Bien, ahora que sabes lo que es

Segundo paso:

- ¿Qué ha pasado y por qué no me va mi Node?
 - Seguramente se te haya olvidado marcar la casilla o si la has marcado, algo no ha funcionado, pero tenemos que arreglarlo porque si no, el sistema no reconocerá los comandos node ni npm

Tercer paso:

- ¿Cómo lo soluciono?
 - Buscamos “Editar las variables de entorno del sistema”
 - En “Variables del sistema”, seleccionamos Path y hacemos clic en Editar
 - Agregar la ruta donde está instalado Node.js, por ejemplo:

C:\Program Files\nodejs\
 - Y listo

Ya tienes el node agregado al PATH y en 3 sencillos pasos, cómo tremendo proNode

Módulo HTTP en Node.js

En [Node.js](#) hay módulos y uno de ellos es HTTP, pero, ¿Qué es HTTP?, como ya he dicho es un módulo el cual permite crear un servidor web de manera sencilla y sin configuraciones externas.

Te pongo un ejemplo, que aunque parezca muy largo y tedioso, tú copia y pega en tu archivo, que ahora verás la magia que me enseñó mi mentor Don Javier:

```
var http = require('http').createServer(webServer),  
  
form = require('fs').readFileSync('FORM.HTML'),  
  
querystring = require('querystring')  
  
util = require('util'),  
  
dataString = ""; //Cadena de texto vacia para ir concatenando el resultado  
  
  
function webServer(req,res)  
{  
  
if(req.method == 'GET'){  
  
    res.writeHead(200, {'Context-Type' : 'text/html'})  
  
    res.end(form)//Es lo que enviará al navegador web  
  
}  
  
    if(req.method == 'POST')  
  
    {  
  
        req  
  
        .on('data', function (data){ //Mientras haya datos, ejecutaremos la siguiente  
Callback  
  
            dataString += data //Que concatenará el dato en la variable dataString  
  
        })  
  
        .on('end', function (){ //Cuando terminen los datos, ejecutaremos la  
siguiente Callback
```

```
var dataObject = querystring.parse(dataString),

dataJSON = util.inspect(dataObject)

var templateString = `Los datos que enviaste por POST como string son:
${dataString}

Los datos que enviaste por POST JSON string son: ${dataJSON}`

//Declaramos una variable de texto

//Texto concatenado con el valor de la variable ${dataString}

console.log(templateString) //Lo mostramos en el terminal

res.end(templateString) //Es lo que enviará al navegador web

})

}

}

http.listen(3000)

console.log('Servidor corriendo en http://localhost:3000/')
```

El siguiente apartado sería los próximos pasos, pero no, te voy a enseñar un par de cositas más

¿Qué hacemos cuando ya tenemos lo anterior bien?

Pues muy sencillo, vamos a crear un proyecto estructurado, para que puedas empezar a utilizar package.json, donde se puede gestionar las dependencias y scripts personalizados.

Instalamos nodemon para reiniciar el servidor automáticamente cuando se detecten cambios en el código:

npm install -g nodemon

Y además de esto, pones en otro archivo:

npm i install cowsay y pones este código:

```
var cowsay = require("cowsay");  
  
console.log(cowsay.say({ e:"^^", T: "p", text: "Muuuuuuuuuuuuuuuy buenos dias" }  
));
```

Ahora ejecuta y verás lo que te sale

Próximos pasos

Explorar módulos externos:

Instalamos librerías con NPM, por ejemplo:

npm install express

- Express facilita la creación de APIs y aplicaciones web más avanzadas.
- Profundizar en Node.js:
Explora módulos como fs (archivos), path, os, y events.

Conclusión

En este manual aprendimos a instalar y configurar Node.js y NPM, comprender el uso del PATH para ejecutar comandos globales y crear un servidor básico con el módulo HTTP.

También aplicamos buenas prácticas para iniciar proyectos en Node.js, entendiendo su papel como una herramienta moderna, rápida y escalable para desarrollar aplicaciones completas en JavaScript. Además de tenerme a mí como el que te hace el tutorial, que todo es un plus