**🧠 Manual: Introducción a Node.js, npm y creación de un proyecto con TypeScript**

Tabla de contenido

[1. ¿Qué es Node.js? 1](#_Toc212190577)

[2. ¿Qué es npm? 1](#_Toc212190578)

[3. ¿Cómo crear un proyecto npm desde cero? 2](#_Toc212190579)

[4. ¿Cómo añadirle dependencias? (Ejemplo con TypeScript) 3](#_Toc212190580)

[5. ¿Cómo ejecutar y arrancar el proyecto? 4](#_Toc212190581)

[✅ Resumen final 5](#_Toc212190582)

# 1. ¿Qué es Node.js?

**Node.js** es un entorno de ejecución que permite ejecutar **código JavaScript fuera del navegador**.

Antes de Node.js, JavaScript solo podía ejecutarse dentro del navegador web (por ejemplo, en Chrome o Firefox).  
Con Node.js, podemos usar JavaScript en el **lado del servidor** para crear aplicaciones, servidores web, scripts automatizados, herramientas de desarrollo, etc.

🔹 **Características principales:**

* Basado en el motor **V8** de Google Chrome.
* Permite trabajar con **módulos** y librerías.
* Es **multiplataforma** (Windows, macOS, Linux).
* Usa un modelo de **E/S no bloqueante**, lo que lo hace muy rápido.

💡 **Ejemplo de uso:**  
Node.js se utiliza para crear servidores web, APIs REST, herramientas de construcción como Vite, Webpack, etc.

# 2. ¿Qué es npm?

**npm** (Node Package Manager) es el **gestor de paquetes de Node.js**.  
Sirve para **instalar, actualizar o eliminar** librerías y dependencias necesarias en un proyecto.

Cuando instalas Node.js, **npm viene incluido automáticamente**.

🔹 **Funciones principales de npm:**

* Instalar librerías de terceros (npm install nombre\_paquete).
* Gestionar dependencias del proyecto.
* Crear scripts personalizados (por ejemplo, npm start).
* Publicar tus propios paquetes.

💡 **Ejemplo:**  
Instalar la librería Express:

npm install express

# 3. ¿Cómo crear un proyecto npm desde cero?

A continuación veremos cómo crear un proyecto paso a paso.

**🧩 Paso 1. Crear una carpeta para el proyecto**

Abre una terminal y escribe:

mkdir mi-proyecto

cd mi-proyecto

**🧩 Paso 2. Inicializar npm**

Ejecuta:

npm init

Este comando te hará una serie de preguntas (nombre del proyecto, versión, autor, etc.)  
Al final generará un archivo llamado **package.json** que contiene toda la información del proyecto y sus dependencias.

💡 Si quieres saltarte las preguntas y usar valores por defecto:

npm init -y

📁 **Ejemplo de package.json generado:**

{

"name": "mi-proyecto",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {},

"keywords": [],

"author": "",

"license": "ISC"

}

# 4. ¿Cómo añadirle dependencias? (Ejemplo con TypeScript)

Vamos a instalar **TypeScript** como dependencia de desarrollo.

**🧩 Paso 1. Instalar TypeScript**

En la terminal, dentro del proyecto:

npm install typescript --save-dev

📌 La opción --save-dev indica que esta dependencia se usará **solo en desarrollo**.

Esto creará una carpeta node\_modules/ y añadirá TypeScript al package.json:

"devDependencies": {

"typescript": "^5.6.3"

}

**🧩 Paso 2. Inicializar configuración de TypeScript**

Crea el archivo de configuración ejecutando:

npx tsc --init

Se generará un archivo llamado **tsconfig.json**, que contiene todas las opciones de compilación.  
Por ejemplo:

{

"compilerOptions": {

"target": "es6",

"module": "commonjs",

"outDir": "./dist",

"rootDir": "./src",

"strict": true

}

}

**🧩 Paso 3. Crear estructura de carpetas**

Crea la siguiente estructura:

mi-proyecto/

│

├── src/

│ └── index.ts

├── dist/

├── package.json

├── tsconfig.json

Dentro de src/index.ts, escribe un código simple:

console.log("¡Hola desde TypeScript!");

# 5. ¿Cómo ejecutar y arrancar el proyecto?

**🧩 Paso 1. Compilar el código TypeScript**

Ejecuta:

npx tsc

Esto compilará los archivos .ts a .js dentro de la carpeta dist/.

**🧩 Paso 2. Ejecutar el código con Node.js**

Una vez compilado:

node dist/index.js

💡 **Salida esperada:**

¡Hola desde TypeScript!

**🧩 Paso 3. Añadir scripts de ejecución a package.json**

Podemos automatizar los comandos añadiendo scripts personalizados:

"scripts": {

"build": "tsc",

"start": "node dist/index.js"

}

Ahora puedes ejecutar:

npm run build

npm start

# ✅ Resumen final

| **Concepto** | **Descripción** | **Comando principal** |
| --- | --- | --- |
| **Node.js** | Entorno para ejecutar JavaScript fuera del navegador | node archivo.js |
| **npm** | Gestor de paquetes de Node.js | npm install |
| **Crear proyecto** | Inicializa un nuevo proyecto npm | npm init -y |
| **Añadir TypeScript** | Instala dependencia de desarrollo | npm install typescript --save-dev |
| **Inicializar TypeScript** | Crea archivo de configuración | npx tsc --init |
| **Compilar** | Transforma .ts a .js | npx tsc |
| **Ejecutar** | Lanza el programa con Node | node dist/index.js |