



# Manual instalación

Antonio José Rendón Rodríguez

<b>1.¿Qué es node?</b>	<b>3</b>
• Características	3
<b>2.Instalación</b>	<b>3</b>
<b>3.¿Qué es npm?</b>	<b>3</b>
• Instalar paquetes mediante npm	3
<b>4.Instalación</b>	<b>3</b>
<b>5.¿Qué es módulo HTTP?</b>	<b>3</b>
<b>6.Cómo usar HTTP</b>	<b>4</b>
• Ejemplo de creación.	4
<b>7.¿Qué es módulo path?</b>	<b>4</b>
<b>8.Cómo usar path</b>	<b>4</b>
• Ejemplo de uso	4

## 1. ¿Qué es node?

Node.js es un entorno de ejecución de código abierto para el lenguaje JavaScript que modifica las características de JavaScript y mejora su funcionalidad.

- Características

Node tiene varios aspectos importantes:

- mecanismos para interactuar con el exterior.
- librería estándar.
- sistema con un solo hilo.
- Event loop.
- Es concurrente sin paralelismo.
- Es asíncrono y no bloqueante.
- Es orientado a eventos.

## 2. Instalación

para su instalación descargaremos el archivo .msi y ejecutaremos su instalación en nuestro dispositivo windows, para otros dispositivos la extensión del archivo será distinta

## 3. ¿Qué es npm?

Es el gestor de paquete que usa por defecto [Node.js](#).

- Instalar paquetes mediante npm

La forma más sencilla será acceder a la [web oficial de NPM](#), donde podremos encontrar los paquetes que podremos instalar y nos mostrarán el comando necesario para este proceso. Este paquete se instalará mediante la terminal.

## 4. Instalación

Su instalación se hace automáticamente al instalar [node.js](#).

## 5. ¿Qué es módulo HTTP?

Es un módulo usado para levantar un servidor web y poder empezar a hacer aplicaciones web.

## 6. Cómo usar HTTP

Para usar http usaremos “require(http)” para cargar dicho módulo, para crear el servidor usaremos la función http.createServer().

- Ejemplo de creación.

Un ejemplo de creación de un servidor web mediante http sería creando un archivo [ejemplo.js](#) y con este código:

```
const { createServer } = require('node:http');

const hostname = '127.0.0.1';
const port = 3000;

const server = createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World');
});

server.listen(port, hostname, () => {
  console.log(`Server running at
http://${hostname}:${port}/`);
});
```

## 7.¿Qué es módulo path?

Es un módulo usado para trabajar con las rutas dentro del sistema de archivos, es necesario importarlo de manera explícita.

## 8.Cómo usar path

Para importarlo y de esta forma poder usarlo usaremos “require('path')”

- Ejemplo de uso

```
usaremos path para manejar y transformar rutas var http =
require 'http'  createServer webServer
path = require('path'), //Requerimos el módulo path
urls = [ //Declaramos un array con 3 objetos
  { //Este primer objeto no contiene nada en la ruta
    ruta : '',
    output : '<h2>Home</h2>'
  },
  { //El segundo contendrá la ruta hacia acerca
    ruta : 'acerca',
    output : '<h2>Acerca</h2>'
  },
  { //El tercer contendrá la ruta hacia contacto
    ruta : 'contacto',
    output : '<h2>Contacto</h2>'
  },
];
```

```

        { //El tercero contendrá la ruta hacia contacto
          ruta : 'contacto',
          output : '<h2>Contacto</h2>'
        }
      ]
    }
  }

function webServer(req, res)
{
  var message = '<h1>Hola Node.js</h1>',
      pathURL = path.basename(req.url)
  //El método basename eliminará todo de la ruta
  //excepto el nombre del fichero final (index.html, por
  //ejemplo)
  //El objeto req devolverá la ruta completa de la petición
  console.log("Ruta completa: "+req.url)
  console.log("Ruta corta: "+pathURL)
  //Recorremos todas las rutas del array
  urls.forEach(function (pos) {
    if(pos.ruta == pathURL)
    {
      res.writeHead(200, {'Content-Type':'text/html'})
      res.end(message + pos.output)
    }
  })
  //En caso de que la URL no exista (SI NO OBTENEMOS RESPUESTA)
  if(!res.finished)
  {
    res.writeHead(404, {'Content-Type':'text/html'})
    res.end('<h1>Error 404: Not Found</h1><br><h2>Eres un
    melon, la ruta no existe!!</h2>')
  }
}

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

```

## 9.próximos pasos

Esto será una introducción para los próximos pasos, los cuales serían, los formularios gestionados mediante GET y POST , para usar estos necesitamos tener los conocimientos anteriores.

## 10.conclusiones.

Node es una herramienta con una gran versatilidad y escalabilidad la cual nos permitirá en futuros proyectos personales o para nuestro trabajo, tener el conocimiento básico es necesario para poder profundizar en este.

## 11.Actividad para practicar(Creatividad)

Para reforzar los conocimientos adquiridos se realizará la siguiente actividad:

- Busca al menos 3 módulos adicionales de NPM muy usados y explica lo que hacen.
- Procede a instalar 2, mostrar los cambios en el proyecto y a desinstalarlo.
- Haz un despliegue en otro proyecto con el PACKAGE.JSON y lánzalo.
- Crear un servidor con http