



# NODE JS: Set up Node JS app

---

**Revathi G P**

Department of Computer Science and Engineering

***Acknowledgement***

*The slides are created from various internet resources with valuable contributions from multiple professors and teaching assistants in the university.*

To install and setup an environment for Node.js, you need the following two softwares available on your computer:

Text Editor.

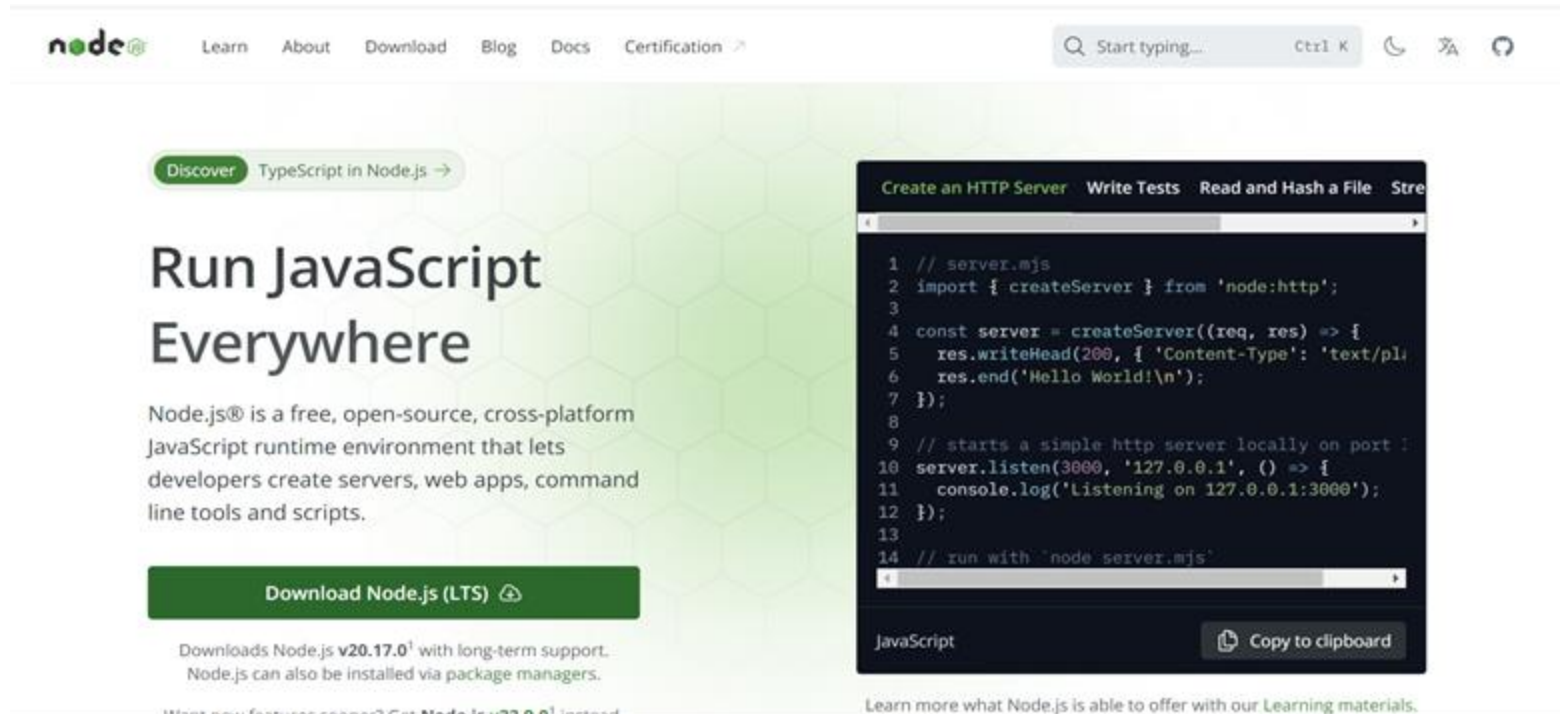
Node.js Binary installable

### **The Node.js Runtime:**

- The source code written in source file is simply JavaScript.
- It is interpreted and executed by the Node.js interpreter.

### How to download Node.js:

You can download the latest version of Node.js installable archive file from <https://nodejs.org/en/>

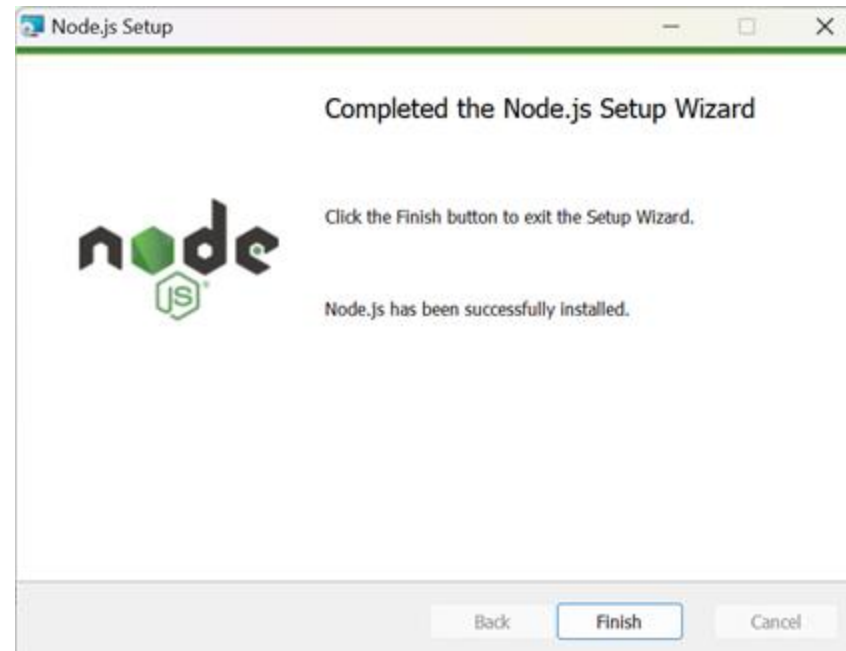


The screenshot shows the Node.js website homepage. The header includes the Node.js logo and navigation links: Learn, About, Download, Blog, Docs, and Certification. A search bar is on the right. The main content area features a green background with a hexagonal pattern and the text "Run JavaScript Everywhere". Below this, it states: "Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts." A prominent green button says "Download Node.js (LTS)". Below the button, it says: "Downloads Node.js v20.17.0<sup>1</sup> with long-term support. Node.js can also be installed via package managers." At the bottom, there is a link to "Learn more what Node.js is able to offer with our Learning materials." On the right side of the page, there is a dark-themed code editor showing a JavaScript snippet for creating an HTTP server. The code is as follows:

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain' });
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with `node server.mjs`
```

Below the code editor, there is a "Copy to clipboard" button and the text "JavaScript".

- Step 1: **Go to the NodeJS website and download NodeJS**
  - Go to the NodeJS website <https://nodejs.org/en/> and download NodeJS.
  - Look for the Latest Stable Version and download it



### **Step 2: Make sure Node and NPM are installed and their PATHs defined**

After Installing, Verify the installation by giving the command as

`node -v` //should return the version number

`Npm -v` // should return the version number of npm package

Returns 'node' is not recognized as an internal or external command, operable program or batch file. Nodejs not installed properly check for path.

Install a editor like Visual Code, Sublime Text or any suitable editors for running Node JS Applications

### **Step 3: Updating the Local npm version.**

The final step in node.js installed is the updation of your local npm version(if required) – the package manager that comes bundled with Node.js.

You can run the following command, to quickly update the npm

**`npm install npm –global // Updates the ‘CLI’ client`**

# NODE JS

## Node JS installation

---



### Node.js First Example

There can be console-based and web-based node.js applications.

### Node.js console-based Example

File: console\_example1.js

**console.log('Hello World');**

Open Node.js command prompt and run the following code:

node console\_example1.js

A screenshot of a Windows command prompt window titled "Node.js command prompt". The window shows the following text:

```
Your environment has been set up for using Node.js 4.4.2 (x64) and npm.  
C:\Users\javatpoint1>cd desktop  
C:\Users\javatpoint1\Desktop>node console_example1.js  
Hello JavaIpoint  
C:\Users\javatpoint1\Desktop>
```



# NODE JS

## Node JS installation

---



### web-based node.js applications.

Once you have downloaded and installed Node.js on your computer, let's try to display "Hello World" in a web browser.

Create a Node.js file named "myfirst.js", and add the following code:

**myfirst.js**

```
var http = require('http');
```

```
http.createServer(function (req, res) {  
  res.writeHead(200, {'Content-Type': 'text/html'});  
  res.end('Hello World!');  
}).listen(8080);
```

Save the file on your computer: C:\Users\Your Name\myfirst.js

The code tells the computer to write "Hello World!" if anyone (e.g. a web browser) tries to access your computer on port 8080.

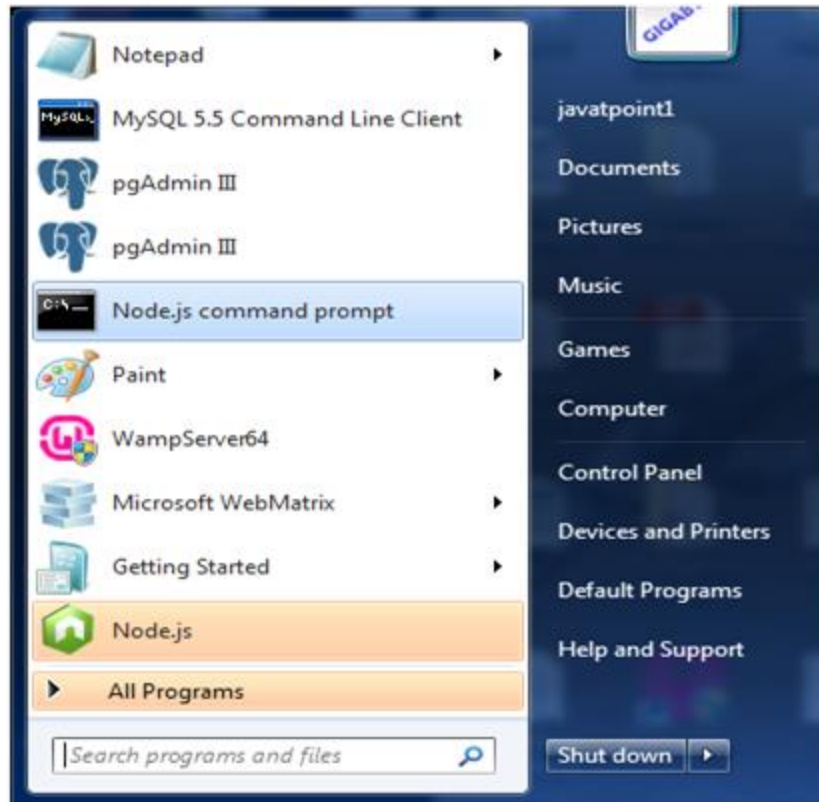
# NODE JS

## Node JS installation



How to start your server:

Go to start menu and click on the Node.js command prompt.



### Initiate the Node.js File

The file you have just created must be initiated by Node.js before any action can take place.

Start your command line interface, write `node myfirst.js`

# NODE JS

## Node JS installation

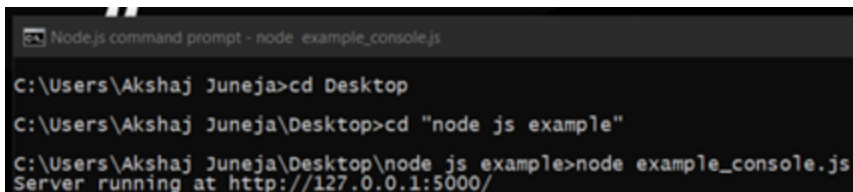
---



To run this myfirst.js file follow the steps as given below:

Search the node.js command prompt in the search bar and open the node.js command prompt.

Go to the folder using cd command in command prompt and write the following command node web.js

A screenshot of a Windows command prompt window titled "Node.js command prompt - node example\_console.js". The window shows the following commands and output:

```
C:\Users\Akshaj Juneja>cd Desktop
C:\Users\Akshaj Juneja\Desktop>cd "node js example"
C:\Users\Akshaj Juneja\Desktop\node js example>node example_console.js
Server running at http://127.0.0.1:5000/
```

- Step 3: **Create a New Project Folder**
- Step 4: **Start running NPM in your project folder**
  - open a terminal.
  - change directories until you are in your project folder.
  - run the command `npm init` in the terminal

```
$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help init' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (simple-node-server)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author: kris hill
license: (ISC)
About to write to C:\Users\krist\OneDrive\Documents\Blog Posts\Example Projects\simple-node-server\package.json:
{
  "name": "simple-node-server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "kris hill",
  "license": "ISC"
}

Is this ok? (yes)
```

- Step 4: **Start running NPM in your project folder**
  - a file called package.json in your project

```
{  
  "name": "u4_bsection",  
  "version": "1.0.0",  
  "description": "demo",  
  "main": "NewU4L2.js",  
  ▶ Debug  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1"  
  },  
  "author": "",  
  "license": "ISC"
```

- **Step 5: Install Any NPM Packages. (covered in later classes)**
  - there are two parts to connecting a package to our app:
    - Download/install the package from NPM
    - Save the package name and version number under “Dependencies” in your package.json
- **Step 6: Create an HTML file**
  - Add a file to your project folder with the extension ‘.html’.
- **Step 7: Create a Node/JavaScript file in the project folder**
  - Add a file to your project folder with the extension ‘.js’.

- **Step 8: Start the Node Server**
  - by opening up a terminal and running the command:
  - *node filenae.js*
- **Step 9: Visit Your (Local) Site!**

- Create a file with .js extension and run it using the command “node filename.js”

- `console.log("Hello World")`
- `const name1="John";`
- `console.log('Hello,',name1);`

- Non Blocking I/O

```
setTimeout(()=>{  
    console.log("Timer Stopped"),2000);  
console.log("Timer Started")
```



- Fetching a file: To include a module, use the require() function with the name of the module:

```
const fs=require('fs')
fs.stat('U4L2.js', (err,stats)=>{
  if (err) throw err;
  console.log('Stats of U4L2.js',JSON.stringify(stats))
})
```

- Renaming a file

```
const fs=require('fs')
fs.rename('U4L2.js',"NewU4L2.js",(err)=>{
  console.log("Rename Succesfull")
})
fs.stat('NewU4L2.js', (err,stats)=>{
  if (err) throw err;
  console.log('Stats of newer.js',JSON.stringify(stats))
})
```

- Non Blocking I/O

```
const fs=require('fs')
fs.readFile('NewU4L2.js','UTF-8' ,(err,data)=>{
  if(err) throw err
  console.log("Contents:",data)
})
console.log("Reading the Contents");
```

- Blocking I/O

```
const fs=require('fs')
const data=fs.readFileSync("NewU4L2.js",'UTF-8')
console.log("Reading the file contents...")
console.log("data:",data)
```



**THANK YOU**

---

**Revathi G P**

Department of  
Computer Science and Engineering