Learn NodeJS in 1 Day

By Krishna Rungta

Table Of Content

Chapter 1: Introduction

Introduction to Node.js

The modern web application has really come a long way over the years with the introduction of many popular frameworks such as bootstrap, Angular JS, etc. All of these frameworks are based on the popular JavaScript framework.

But when it came to developing server based applications there was just kind of a void, and this is where Node.js came into the picture.

Node.js is also based on the JavaScript framework, but it is used for developing server-based applications. While going through the entire tutorial, we will look into Node.js in detail and how we can use it to develop server based applications.

What is Node.js?

Node.js is an open-source, cross-platform runtime environment used for development of server-side web applications. Node.js applications are written in JavaScript and can be run on a wide variety of operating systems.

Node.js is based on an event-driven architecture and a non-blocking Input/Output API that is designed to optimize an application's throughput and scalability for real-time web applications.

Over a long period of time, the framework available for web development were all based on a stateless model. A stateless model is where the data generated in one session (such as information about user settings and events that occurred) is not maintained for usage in the next session with that user.

A lot of work had to be done to maintain the session information between requests for a user. But with Node.js there is finally a way for web applications to have a real-time, two-way connections, where both the client and server can initiate communication, allowing them to exchange data freely.

Why use Node.js?

We will have a look into the real worth of Node.js in the coming chapters, but what is it that makes this framework so famous. Over the years, most of the applications were based on a stateless request- response framework. In these sort of applications, it is up to the developer to ensure the right code was put in place to ensure the state of web session was maintained while the user was working with the system.

But with Node.js web applications, you can now work in real-time and have a 2-way communication. The state is maintained, and the either the client or server can start the communication.

Features of Node.js

Let’s look at some of the key features of Node.js

1. Asynchronous event driven IO helps concurrent request handling – This is probably the biggest selling points of Node.js. This feature basically means that if a request is received by Node for some Input/Output operation, it will execute the operation in the background and continue with processing other requests.

   This is quite different from other programming languages. A simple example of this is given in the code below

```
var fs = require('fs');
          fs.readFile("Sample.txt",function(error,data)
          {
                console.log("Reading Data completed");
     });
```

- The above code snippet looks at reading a file called Sample.txt. In other programming languages, the next line of processing would only happen once the entire file is read.
- But in the case of Node.js the important fraction of code to notice is the declaration of the function (‘function(error,data)’). This is known as a callback function.
- So what happens here is that the file reading operation will start in the background. And other processing can happen simultaneously while the file is being read. Once the file read operation is completed, this anonymous function will be called and the text “Reading Data completed” will be written to the console log.

2. Node uses the V8 JavaScript Runtime engine, the one which is used by Google Chrome. Node has a wrapper over the JavaScript engine which makes the runtime engine much faster and hence processing of requests within Node also become faster.
3. Handling of concurrent requests – Another key functionality of Node is the ability to handle concurrent connections with a very minimal overhead on a single process.
4. The Node.js library used JavaScript – This is another important aspect of development in Node.js. A major part of the development community are already well versed in javascript, and hence, development in Node.js becomes easier for a developer who knows javascript.
5. There are an Active and vibrant community for the Node.js framework. Because of the active community, there are always keys updates made available to the framework. This helps to keep the framework always up-to-date with the latest trends in web development.

Who uses Node.js

Node.js is used by a variety of large companies. Below is a list of a few of them.

- Paypal – A lot of sites within Paypal have also started the transition onto Node.js.
- LinkedIn - LinkedIn is using Node.js to power their Mobile Servers, which powers the iPhone, Android, and Mobile Web products.
- Mozilla has implemented Node.js to support browser APIs which has half a billion installs.
- Ebay hosts their HTTP API service in Node.js

When to Use Node.js

Node.js is best for usage in streaming or event-based real-time applications like

1. Chat applications
2. Game servers – Fast and high-performance servers that need to processes thousands of requests at a time, then this is an ideal framework.
3. Good for collaborative environment – This is good for environments which manage document. In document management environment you will have multiple people who post their documents and do constant changes by checking out and checking in documents. So Node.js is good for these environments because the event loop in Node.js can be triggered whenever documents are changed in a document managed environment.
4. Advertisement servers – Again here you could have thousands of request to pull advertisements from the central server and Node.js can be an ideal framework to handle this.
5. Streaming servers – Another ideal scenario to use Node is for multimedia streaming servers wherein clients have request's to pull different multimedia contents from this server.

Node.js is good when you need high levels of concurrency but less amount of dedicated CPU time.

Best of all, since Node.js is built on javascript, it’s best suited when you build client-side applications which are based on the same javascript framework.

When to not use Node.js

Node.js can be used for a lot of applications with various purpose, the only scenario where it should not be used is if there are long processing times which is required by the application.

Node is structured to be single threaded. If any application is required to carry out some long running calculations in the background. So if the server is doing some calculation, it won’t be able to process any other requests. As discussed above, Node.js is best when processing needs less dedicated CPU time.

Chapter 2: How to Download & Install Node.js - NPM on Windows

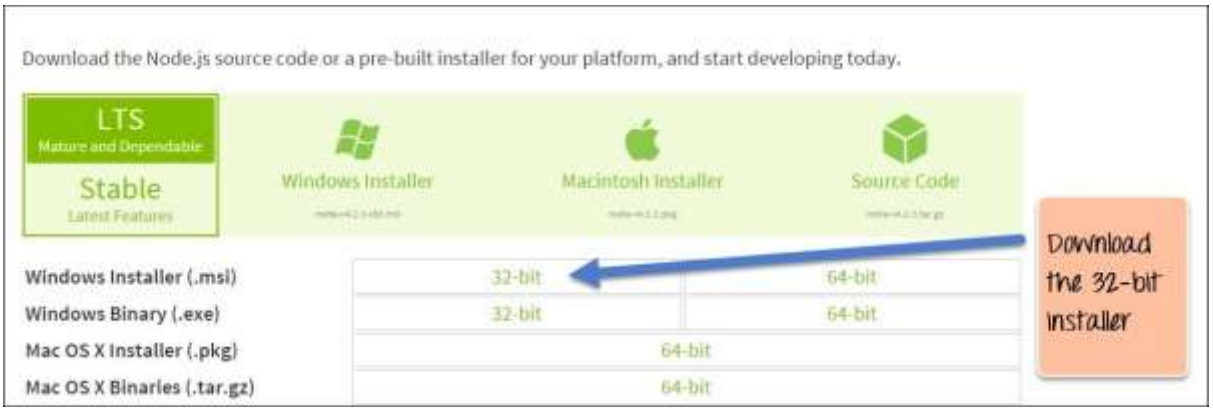
To start building your Node.js applications, the first step is the installation of the node.js framework. The Node.js framework is available for a variety of operating systems right from Windows to Ubuntu and OS X. Once the Node.js framework is installed you can start building your first Node.js applications.

Node.js also has the ability to embedded external functionality or extended functionality by making use of custom modules. These modules have to be installed separately. An example of a module is the MongoDB module which allows you to work with MongoDB databases from your Node.js application.

How to install Node.js on Windows

The first steps in using Node.js is the installation of the Node.js libraries on the client system. To perform the installation of Node.js, perform the below steps;

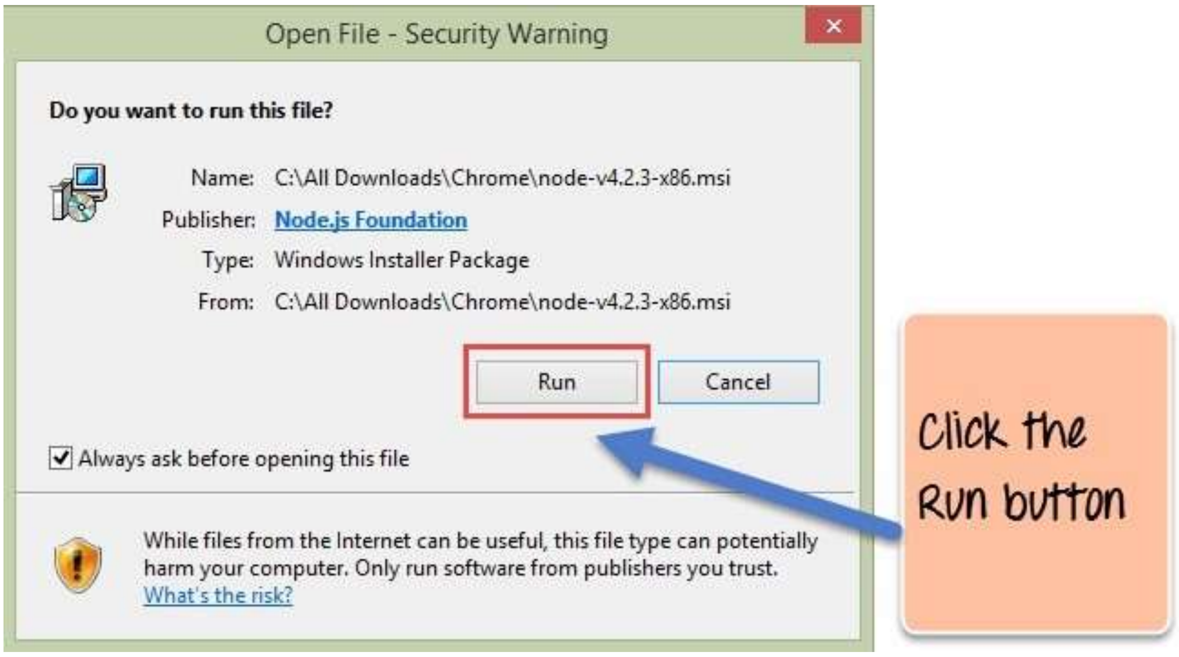
Step 1) Go to the site https://nodejs.org/en/download/ and download the necessary binary files. In our example, we are going to the download the 32-bit setup files for Node.js.



Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.exe)	32-bit	64-bit
Mac OS X Installer (.pkg)	64-bit	
Mac OS X Binaries (.tar.gz)	64-bit	

Step 2) Double click on the downloaded .msi file to start the installation. Click the Run button in the first screen to begin the installation.



Open File - Security Warning

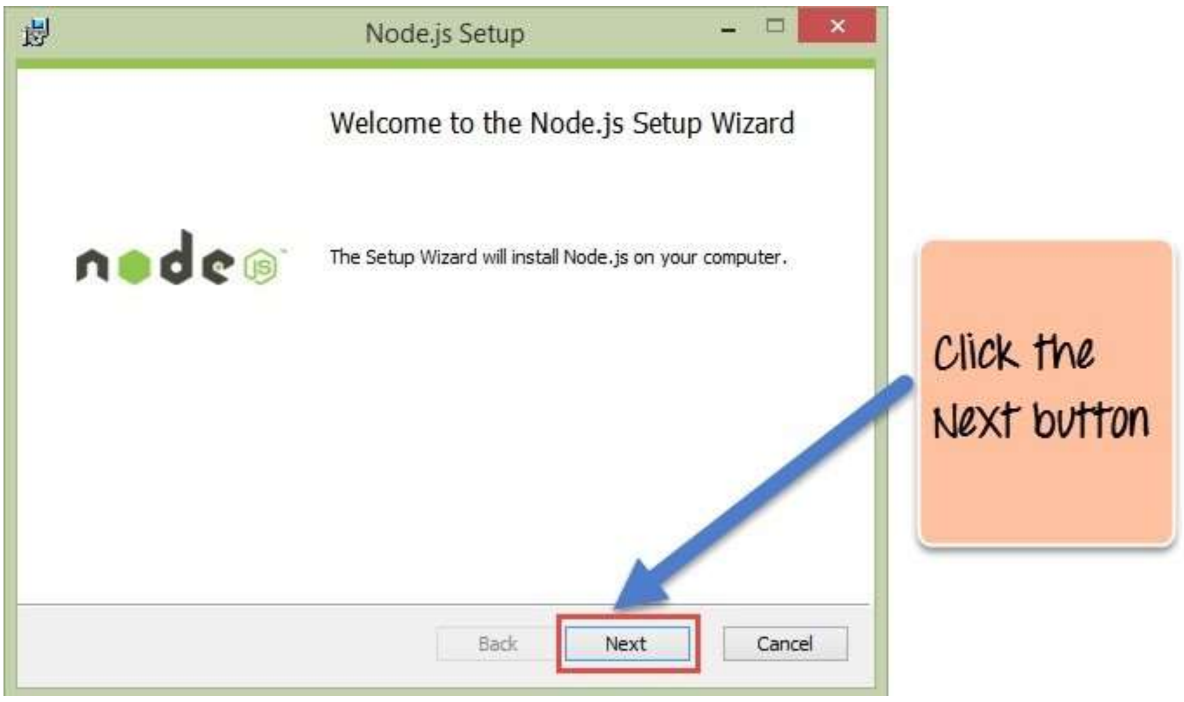
Do you want to run this file?

Name: C:\All Downloads\Chrome\node-v4.2.3-x86.msi
Publisher: Node.js Foundation
Type: Windows Installer Package
From: C:\All Downloads\Chrome\node-v4.2.3-x86.msi

☑ Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

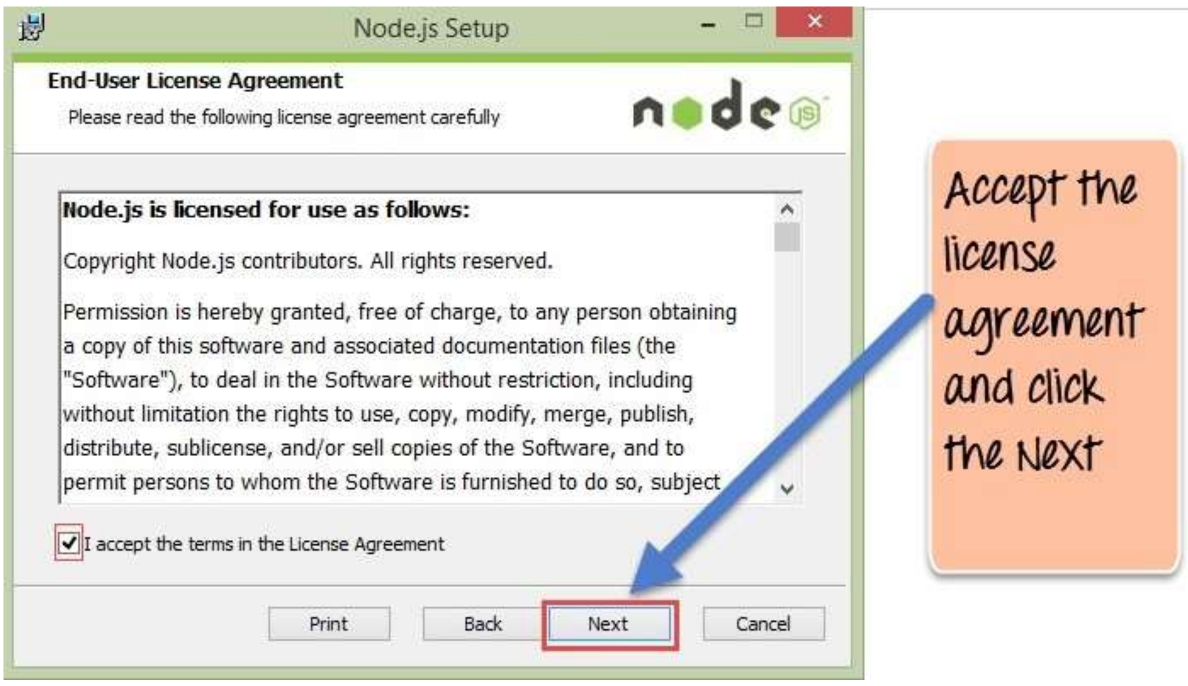
Step 3) In the next screen, click the “Next” button to continue with the installation


Node.js Setup

Welcome to the Node.js Setup Wizard

The Setup Wizard will install Node.js on your computer.

Step 4) In the next screen Accept the license agreement and click on the Next button.


Node.js Setup

End-User License Agreement

Please read the following license agreement carefully

Node.js is licensed for use as follows:

Copyright Node.js contributors. All rights reserved.

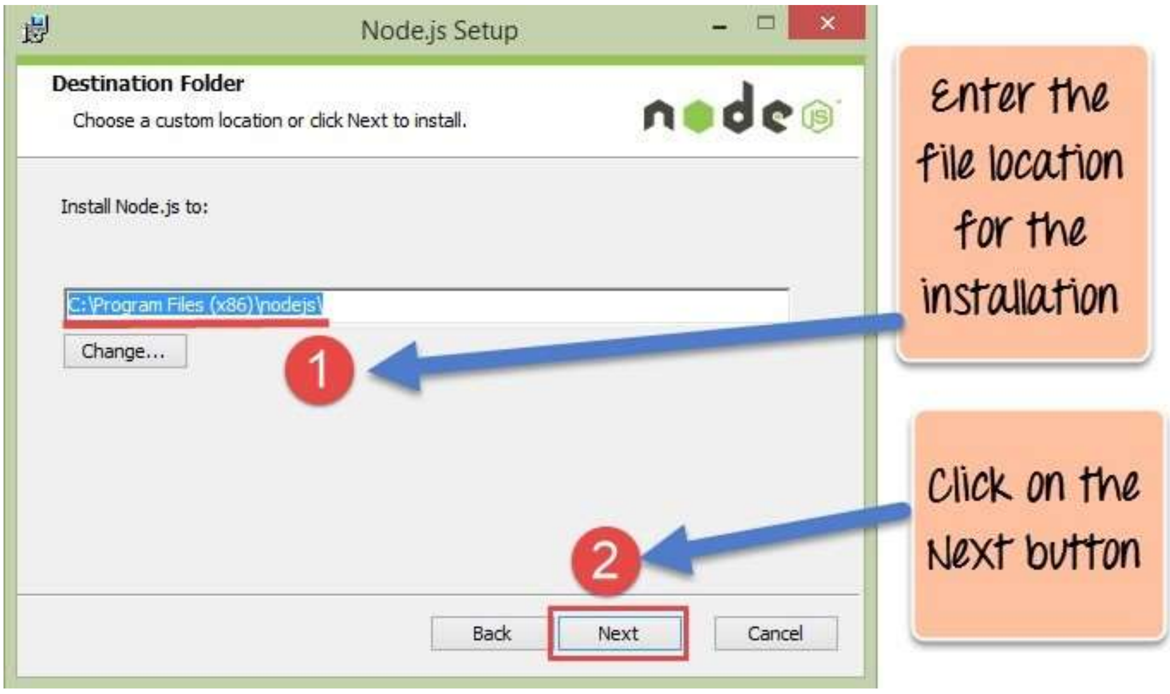
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject

☑ I accept the terms in the License Agreement

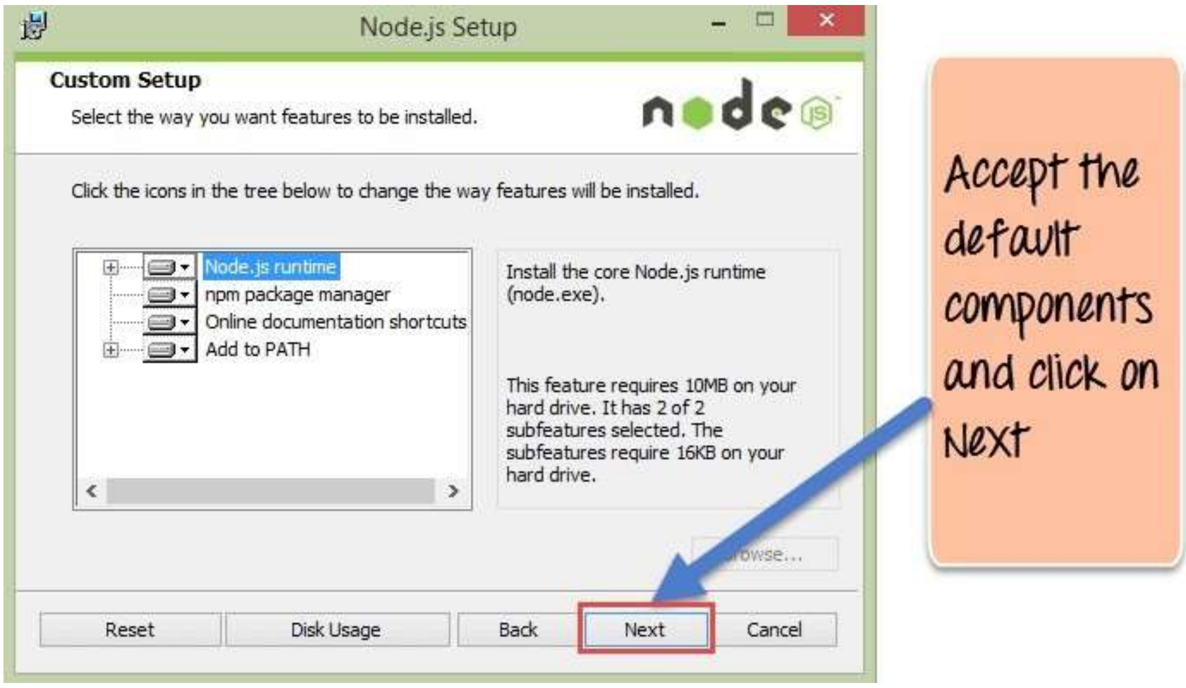
Step 5) In the next screen, choose the location where Node.js needs to be installed and then click on the Next button.

1.First enter the file location for the installation of Node.js. This is where the files for Node.js will be stored after the installation.

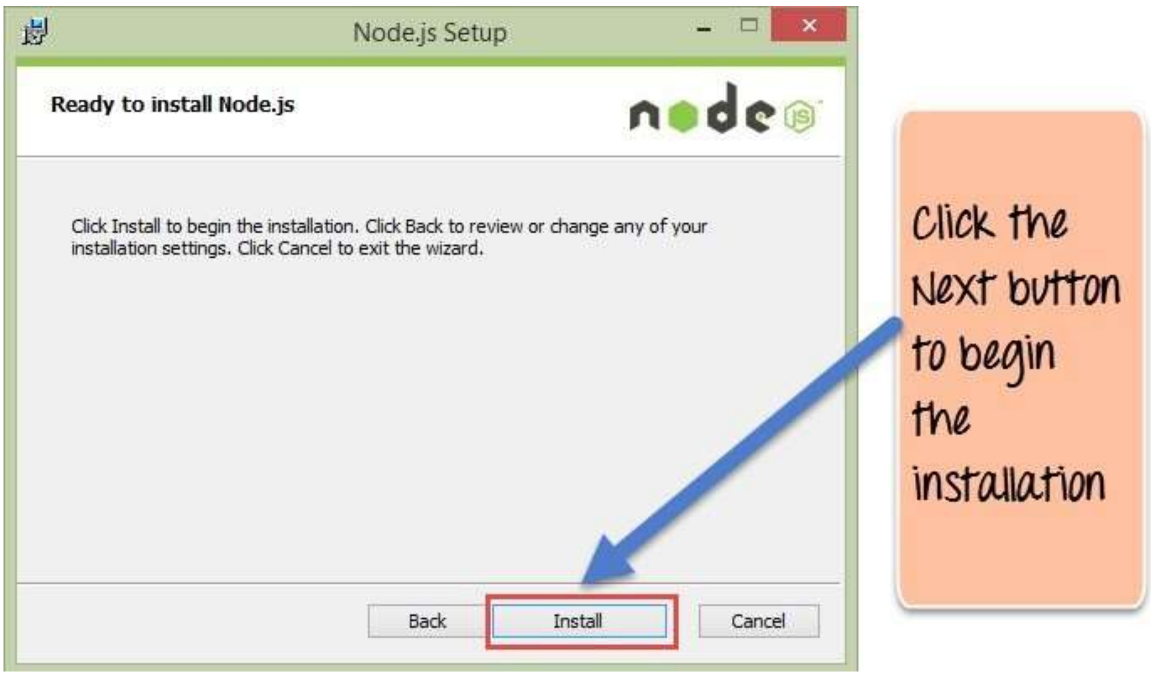
2. Click on the Next button to proceed ahead with the installation.


Node.js Setup
Destination Folder
Choose a custom location or click Next to install.
Install Node.js to:
C:\Program Files (x86)\nodejs\
Change...
1
2
Back
Next
Cancel
Enter the file location for the installation
Click on the Next button

Step 6) Accept the default components and click on the next button.


Node.js Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
Node.js runtime
npm package manager
Online documentation shortcuts
Add to PATH
Install the core Node.js runtime (node.exe).
This feature requires 10MB on your hard drive. It has 2 of 2 subfeatures selected. The subfeatures require 16KB on your hard drive.
Browse...
Reset
Disk Usage
Back
Next
Cancel
Accept the default components and click on Next

Step 7) In the next screen, click the Install button to start the installation.

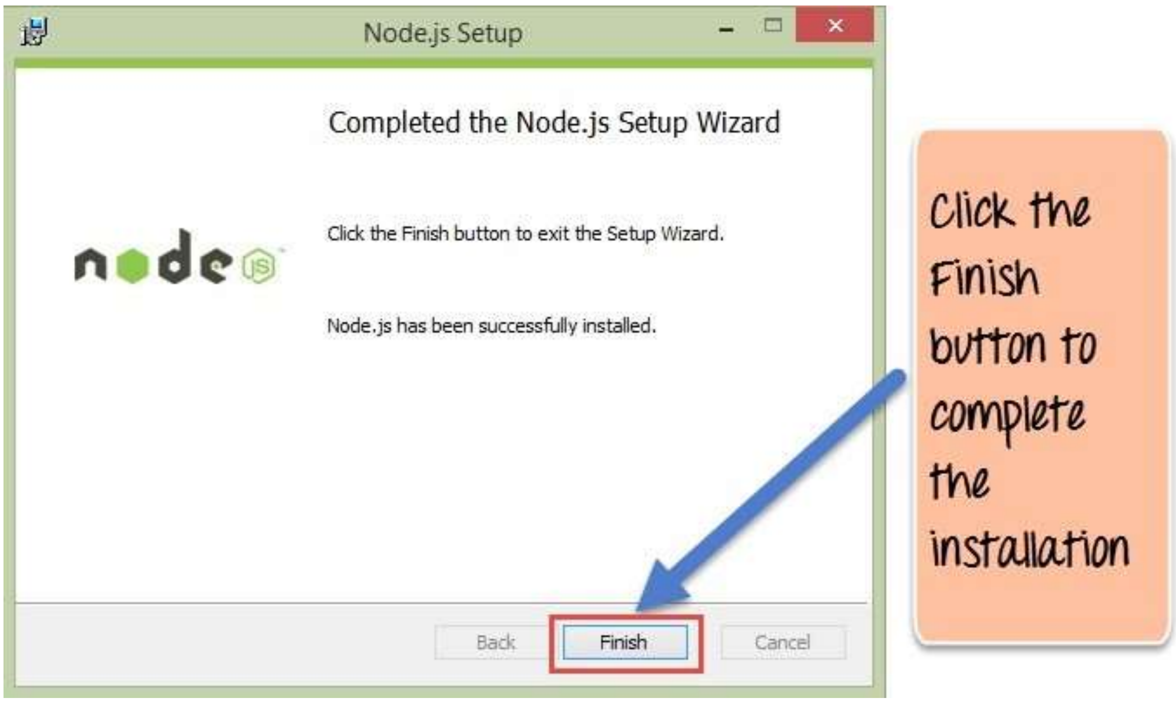


Node.js Setup

Ready to install Node.js

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Step 8) Click the Finish button to complete the installation.



Node.js Setup

Completed the Node.js Setup Wizard

Click the Finish button to exit the Setup Wizard.

Node.js has been successfully installed.

Installing NPM (Node Package Manager) on Windows

The other way to install Node.js on any client machine is to use a “package manager”.

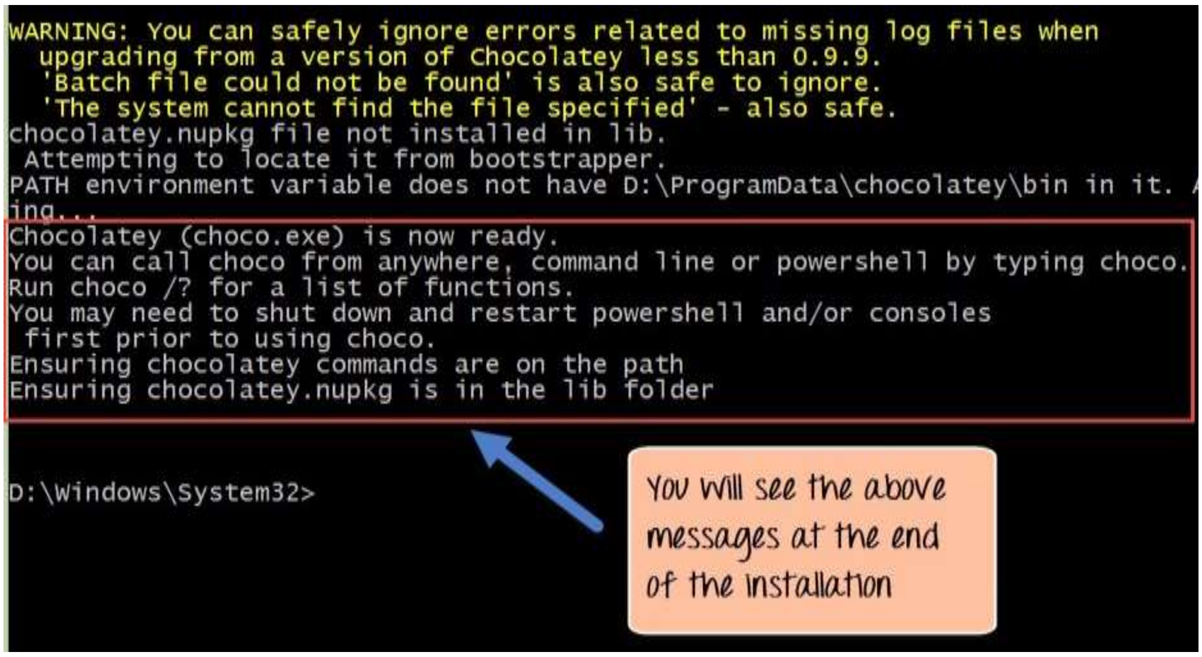
On windows, the node package manager is known as Chocolatey. It was designed to be a decentralized framework for quickly installing applications and tools that you need.

To install Node.js via Chocolatey, the following steps need to be performed.

Step 1) Installing Chocolatey – The Chocolatey website (https://chocolatey.org/) has very clear instructions on how this framework needs to be installed.

- The first steps is to run the below command in the command prompt windows. This command is taken from the Chocolatey web site and is the standard command for installing Node.js via Chocolatey.
- The below command is a PowerShell command which calls the remote PowerShell script on the Chocolatey website. This command needs to be run in a PowerShell command window. This PowerShell script does all the
- necessary work of downloading the required components and installing them accordingly.

```
@powershell -NoProfile -ExecutionPolicy Bypass -Command “iex ((new-object
wet.webclient).DownloadString(‘https://chocolatey.org/install.ps1’))” && SET
PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin
```



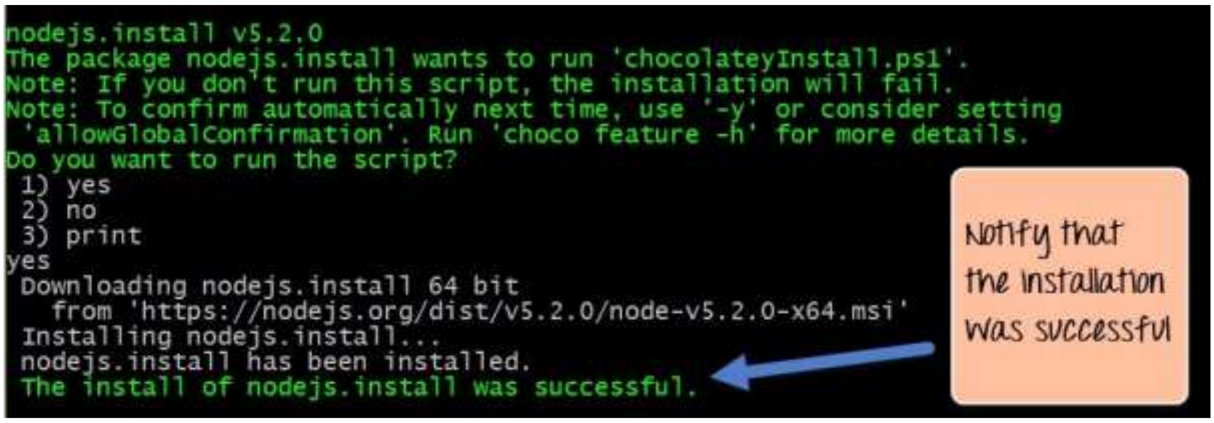
```
WARNING: You can safely ignore errors related to missing log files when
  upgrading from a version of Chocolatey less than 0.9.9.
  'Batch file could not be found' is also safe to ignore.
  'The system cannot find the file specified' - also safe.
chocolatey.nupkg file not installed in lib.
 Attempting to locate it from bootstrapper.
PATH environment variable does not have D:\ProgramData\chocolatey\bin in it.
ing...
Chocolatey (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
 first prior to using choco.
Ensuring chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder

D:\Windows\System32>
```

You will see the above messages at the end of the installation

Step 2) The next step is to install Node.js to your local machine using the Chocolatey, package manager. This can be done by running the below command in the command prompt.

cinst nodejs install



```
nodejs.install v5.2.0
The package nodejs.install wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider setting
 'allowGlobalConfirmation'. Run 'choco feature -h' for more details.
Do you want to run the script?
 1) yes
 2) no
 3) print
yes
 Downloading nodejs.install 64 bit
   from 'https://nodejs.org/dist/v5.2.0/node-v5.2.0-x64.msi'
 Installing nodejs.install...
 nodejs.install has been installed.
 The install of nodejs.install was successful.
```

If the installation is successful, you will get the message of the successful installation of Node.js.

Note: If you get an error like "C:\ProgramData\chocolatey\lib\libreoffice\tools\chocolateyInstall.ps Then manually create the folder in the path

Running your first Hello world application in Node.js

Once you have downloaded and installed Node.js on your computer, lets try to display “Hello World” in a web browser.

Create file Node.js with file name firstprogram.js

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

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

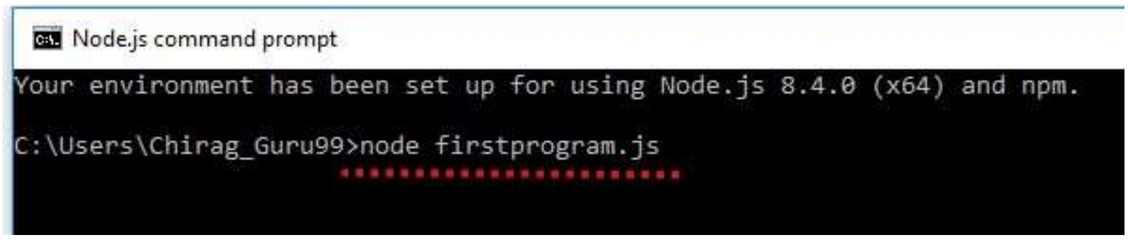
Code Explanation:

1. The basic functionality of the “require” function is that it reads a JavaScript file, executes the file, and then proceeds to return an object. Using this object, one can then use the various functionalities available in the module called by the require function. So in our case, since we want to use the functionality of http and we are using the require(http) command.
2. In this 2nd line of code, we are creating a server application which is based on a simple function. This function is called, whenever a request is made to our server application.
3. When a request is received, we are asking our function to return a “Hello World” response to the client. The writeHead function is used to send header data to the client and while the end function will close the connection to the client.
4. We are then using the server.listen function to make our server application listen to client requests on port no 8080. You can specify any available port over here.

Executing the code

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

2. In the command prompt, navigate to the folder where the file is stored. Enter the command Node firstprogram.js

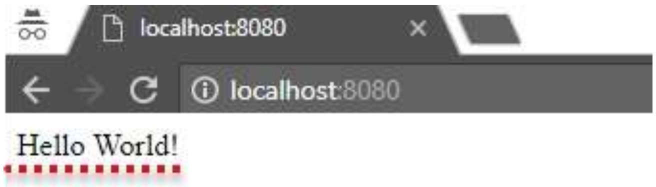


```
Your environment has been set up for using Node.js 8.4.0 (x64) and npm.

C:\Users\Chirag_Guru99>node firstprogram.js
```

3. Now, your computer works as a server! If anyone tries to access your computer on port 8080, they will get a "Hello World!" message in return!
4. Start your internet browser, and type in the address: http://localhost:8080

   OutPut



```
Hello World!
```

Summary

- We have seen the installation of Node.js via the msi installation module which is available on the Node.js website. This installation installs the necessary modules which are required to run a Node.js application on the client.
- Node.js can also be installed via a package manager. The package manager for windows is known as Chocolatey. By running some simple commands in the command prompt, the Chocolatey package manager automatically downloads the necessary files and then installs them on the client machine.
- A simple Node.js application consists of creating a server which listens on a particular port. When a request comes to the server, the server automatically sends a 'Hello World' response to the client.