

[< Previous](#)[Next >](#)

Node.js Web Server

In this section, we will learn how to create a simple Node.js web server and handle HTTP requests.

To access web pages of any web application, you need a [web server](#) . The web server will handle all the http requests for the web application e.g IIS is a web server for ASP.NET web applications and Apache is a web server for PHP or Java web applications.

Node.js provides capabilities to create your own web server which will handle HTTP requests asynchronously. You can use IIS or Apache to run Node.js web application but it is recommended to use Node.js web server.

Create Node.js Web Server

Node.js makes it easy to create a simple web server that processes incoming requests asynchronously.

The following example is a simple Node.js web server contained in server.js file.

server.js

 Copy

```
var http = require('http'); // 1 - Import Node.js core module

var server = http.createServer(function (req, res) { // 2 - creating server

    //handle incoming requests here..

});

server.listen(5000); //3 - listen for any incoming requests

console.log('Node.js web server at port 5000 is running..')
```

In the above example, we import the http module using `require()` function. The http module is a core module of Node.js, so no need to install it using NPM. The next step is to call `createServer()` method of http and specify callback function with request and response parameter. Finally, call `listen()` method of server object which was returned from `createServer()` method with port number, to start listening to incoming requests on port 5000. You can specify any unused port here.

Run the above web server by writing `node server.js` command in command prompt or terminal window and it will display message as shown below.

```
C:\> node server.js
Node.js web server at port 5000 is running..
```

This is how you create a Node.js web server using simple steps. Now, let's see how to handle HTTP request and send response in Node.js web server.

NOW PLAYING

SETUP AND INTRODUCTION TO NGRX

1/4 NgRx...

Wordpress vs...

Are you too old to learn programming or coding at the age... [Watch Video](#)

NgRx is quite tricky to setup for production projects,... [Watch Video](#)

Wordpress or custom website, two ways of choosing... [Watch Video](#)

Handle HTTP Request

The `http.createServer()` method includes [request](#) and [response](#) parameters which is supplied by Node.js. The request object can be used to get information about the current HTTP request e.g., url, request header, and data. The response object can be used to send a response for a current HTTP request.

The following example demonstrates handling HTTP request and response in Node.js.

server.js

Copy

```
var http = require('http'); // Import Node.js core module

var server = http.createServer(function (req, res) { //create web server
  if (req.url == '/') { //check the URL of the current request

    // set response header
    res.writeHead(200, { 'Content-Type': 'text/html' });

    // set response content
    res.write('<html><body><p>This is home Page.</p></body></html>');
    res.end();

  }
  else if (req.url == "/student") {

    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write('<html><body><p>This is student Page.</p></body></html>');
    res.end();

  }
  else if (req.url == "/admin") {

    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write('<html><body><p>This is admin Page.</p></body></html>');
    res.end();

  }
  else
    res.end('Invalid Request!');

});

server.listen(5000); //6 - listen for any incoming requests

console.log('Node.js web server at port 5000 is running..')
```

In the above example, req.url is used to check the url of the current request and based on that it sends the response. To send a response, first it sets the response header using writeHead() method and then writes a string as a response body using write() method. Finally, Node.js web server sends the response using end() method.

Now, run the above web server as shown below.

```
C:\> node server.js
Node.js web server at port 5000 is running..
```

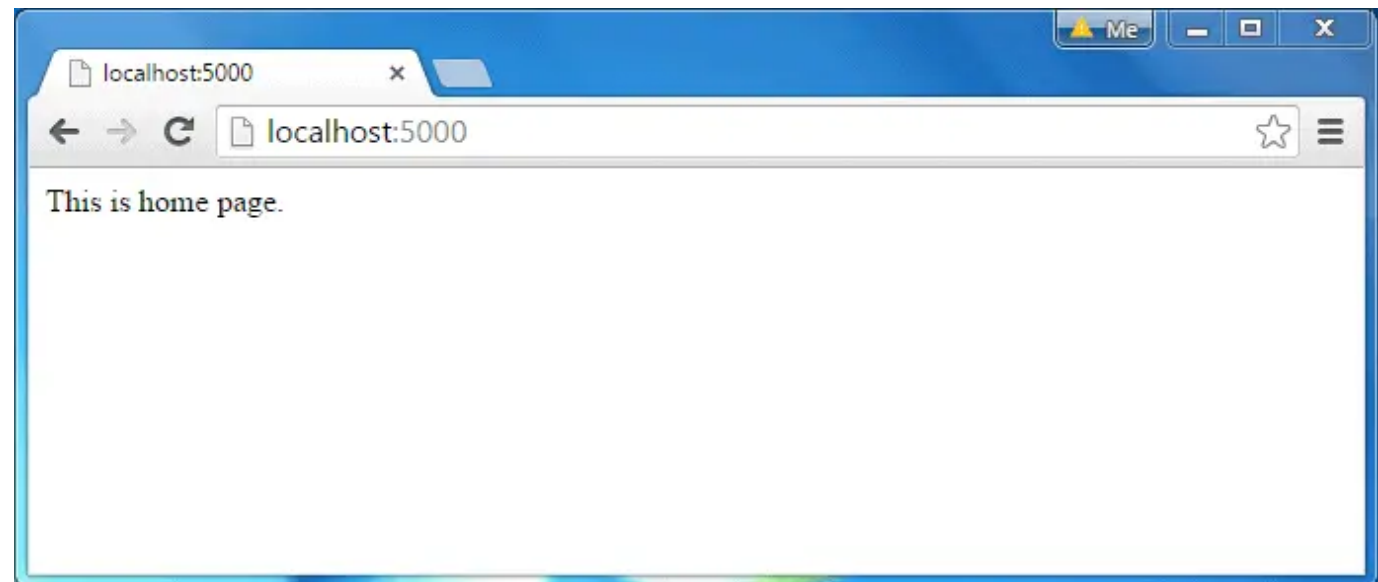
To test it, you can use the command-line program curl, which most Mac and Linux machines have pre-installed.

```
curl -i http://localhost:5000
```

You should see the following response.

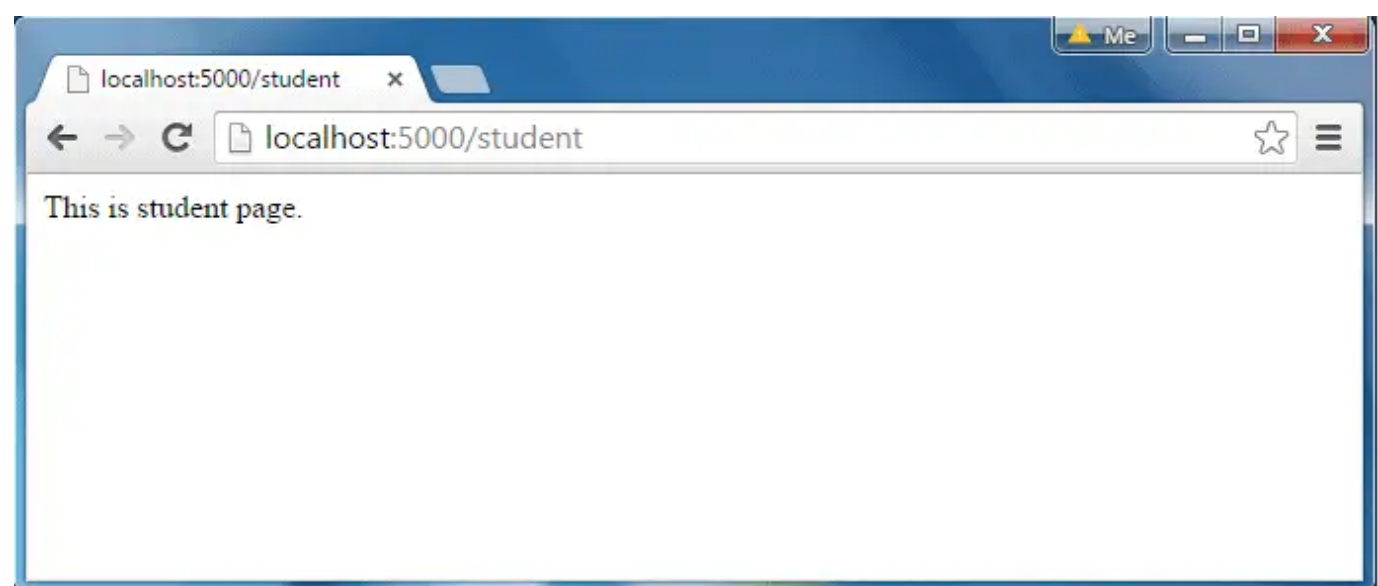
```
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Tue, 8 Sep 2015 03:05:08 GMT
Connection: keep-alive
This is home page.
```

For Windows users, point your browser to *http://localhost:5000* and see the following result.



Node.js Web Server Response

The same way, point your browser to *http://localhost:5000/student* and see the following result.



Node.js Web Server Response

It will display "Invalid Request" for all requests other than the above URLs.

Sending JSON Response

The following example demonstrates how to serve JSON response from the Node.js web server.

server.js

Copy

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

var server = http.createServer(function (req, res) {

    if (req.url == '/data') { //check the URL of the current request
        res.writeHead(200, { 'Content-Type': 'application/json' });
        res.write(JSON.stringify({ message: "Hello World"}));
        res.end();
    }
});

server.listen(5000);


console.log('Node.js web server at port 5000 is running..')
```

So, this way you can create a simple web server that serves different responses.

Learn how to [create Node.js Web Application in Visual Studio](#).

Want to check how much you know Node.js?

Start Node.js Test

-   Share
-   Tweet
-   Share
-   Whatsapp

[< Previous](#)

[Next >](#)

.NET Tutorials

C#

Object Oriented C#

ASP.NET Core

ASP.NET MVC

LINQ

Inversion of Control

Web API

Database Tutorials

SQL

SQL Server

PostgreSQL

MongoDB

JavaScript Tutorials

JavaScript

TypeScript

jQuery

Angular 11

Node.js

D3.js

Sass

Programming Tutorials

Python

Go lang

HTTPS (SSL)

TutorialsTeacher.com

TutorialsTeacher.com is optimized for learning web technologies step by step. Examples might be simplified to improve reading and basic understanding. While using this site, you agree to have read and accepted our terms of use and privacy policy.

✉ Contact Us

E-mail list

Subscribe to TutorialTeacher email list and get latest updates, tips & tricks on C#, .Net, JavaScript, jQuery, AngularJS, Node.js to your inbox.

Email address

GO

We respect your privacy.

© 2023 TutorialTeacher.com. All Rights Reserved.