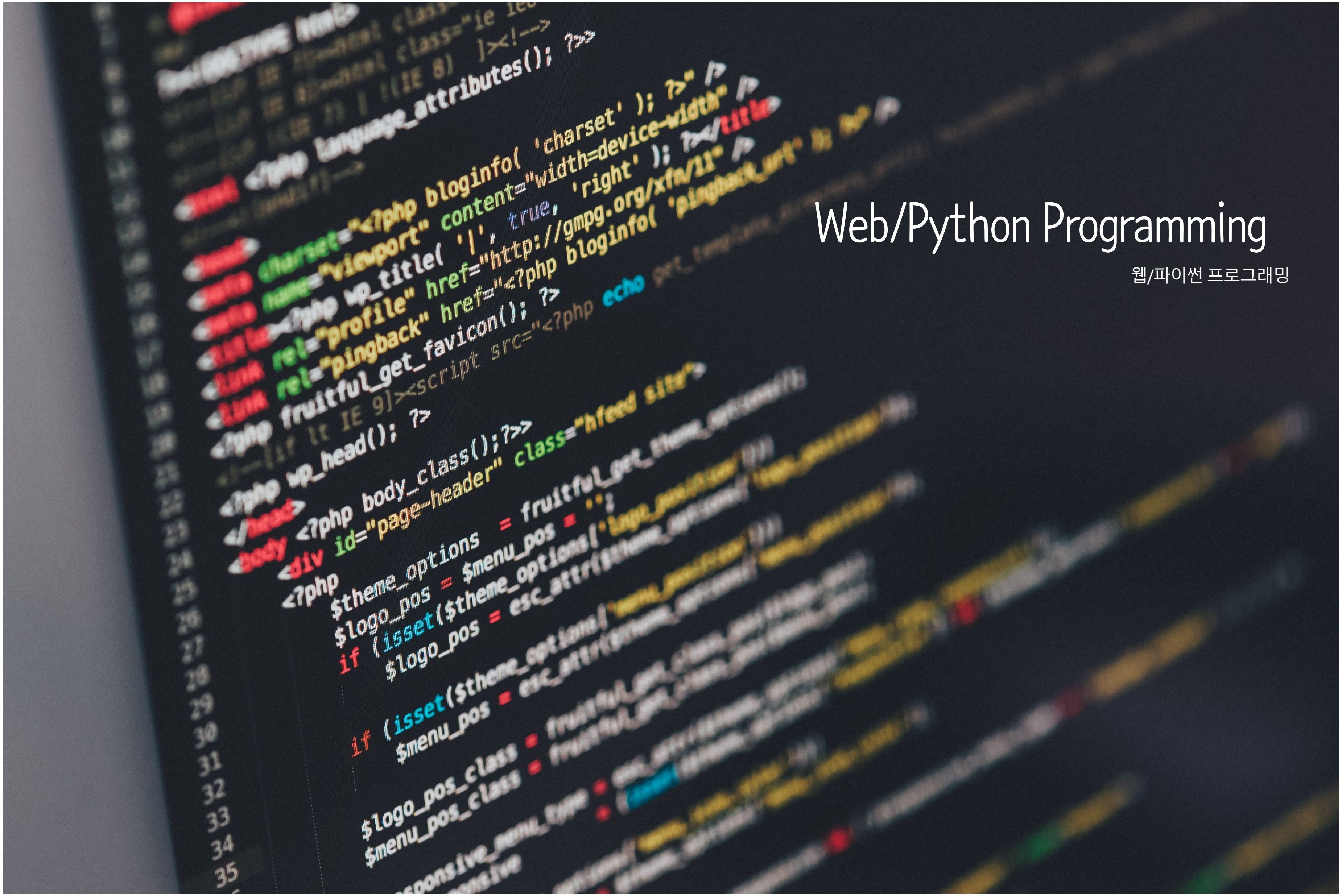


Web/Python Programming

웹/파이썬 프로그래밍

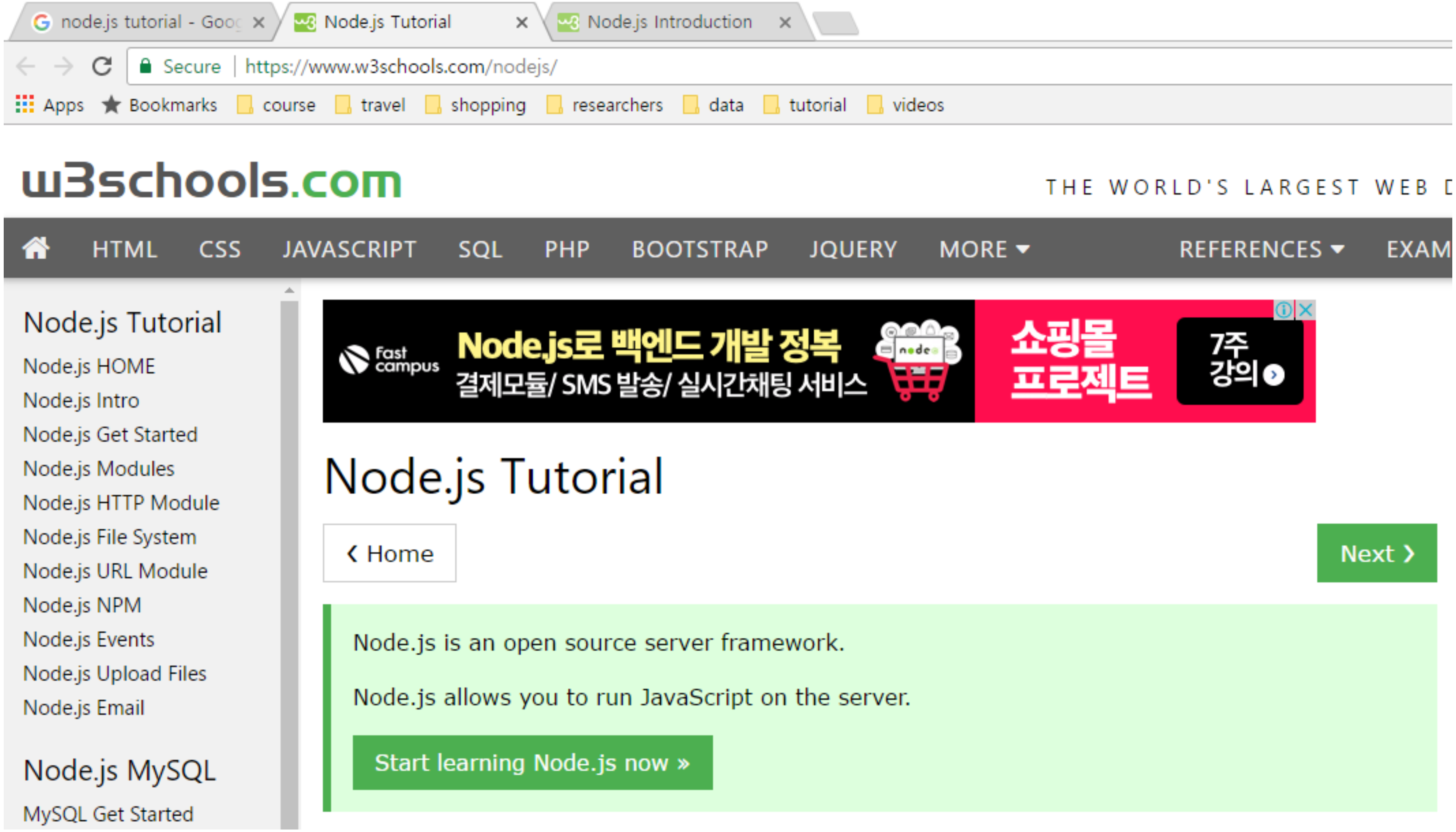
Today

- Let’s Build A Web Server
  - Javascript revisited
  - Node.js

What is Node.js?

- 노드제이에스 / 노드
- Historically, JavaScript was used primarily for client-side scripting, in which scripts written in JS are embedded in a webpage’s HTML and run client-side by a JS engine in the user’s web browser.
- Node.js lets developers use JS for server-side scripting – running scripts server-side to produce dynamic web page content before the page is sent to the user’s web browser.
- Consequently, Node.js represents a “JavaScript everywhere” paradigm, unifying web application development around a single programming language, rather than different languages for server side and client side scripts.

Node.js



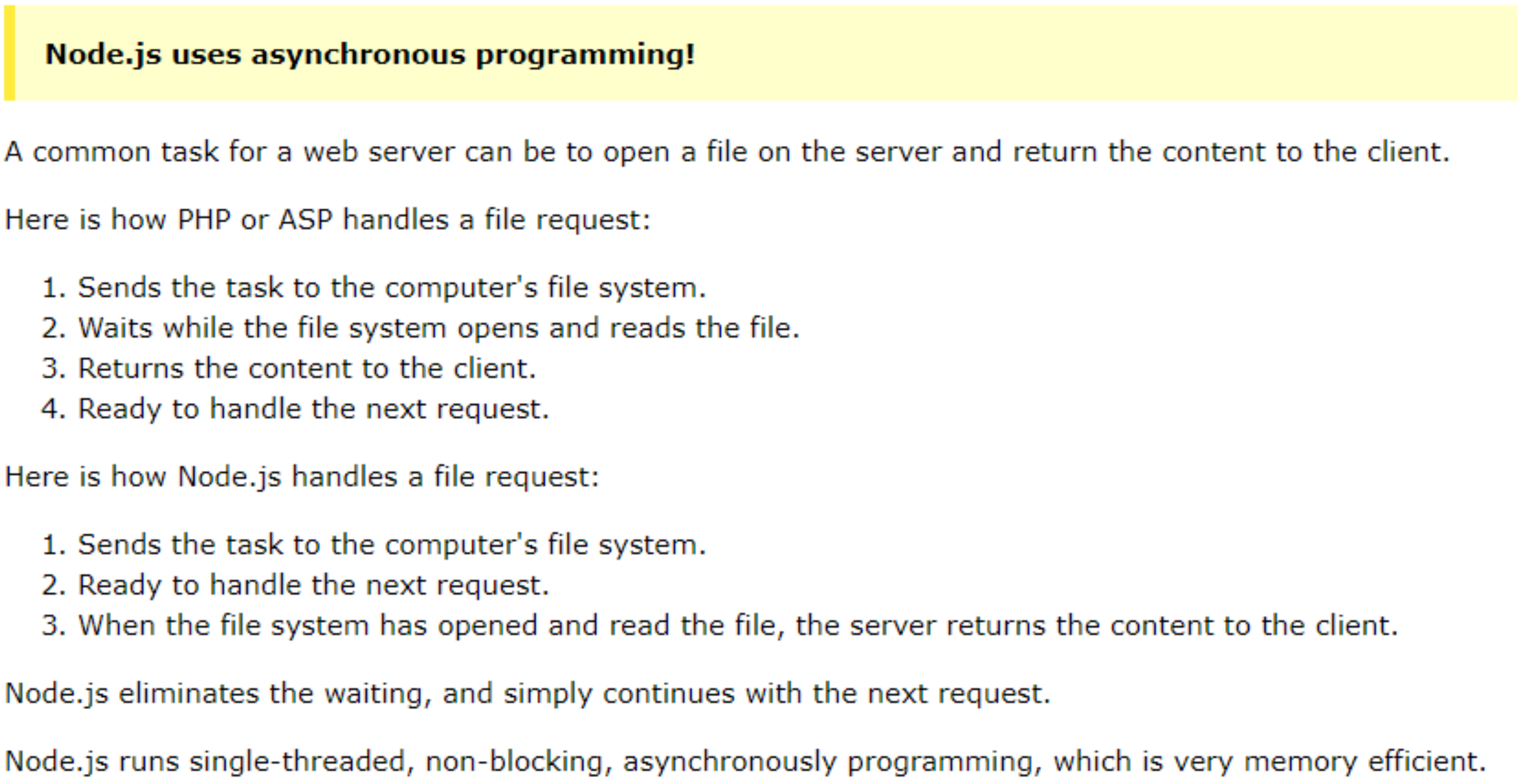
Node.js Tutorial

Node.js is an open source server framework.

Node.js allows you to run JavaScript on the server.

Start learning Node.js now »

Characteristics of Node.js



Node.js uses asynchronous programming!

A common task for a web server can be to open a file on the server and return the content to the client.

Here is how PHP or ASP handles a file request:

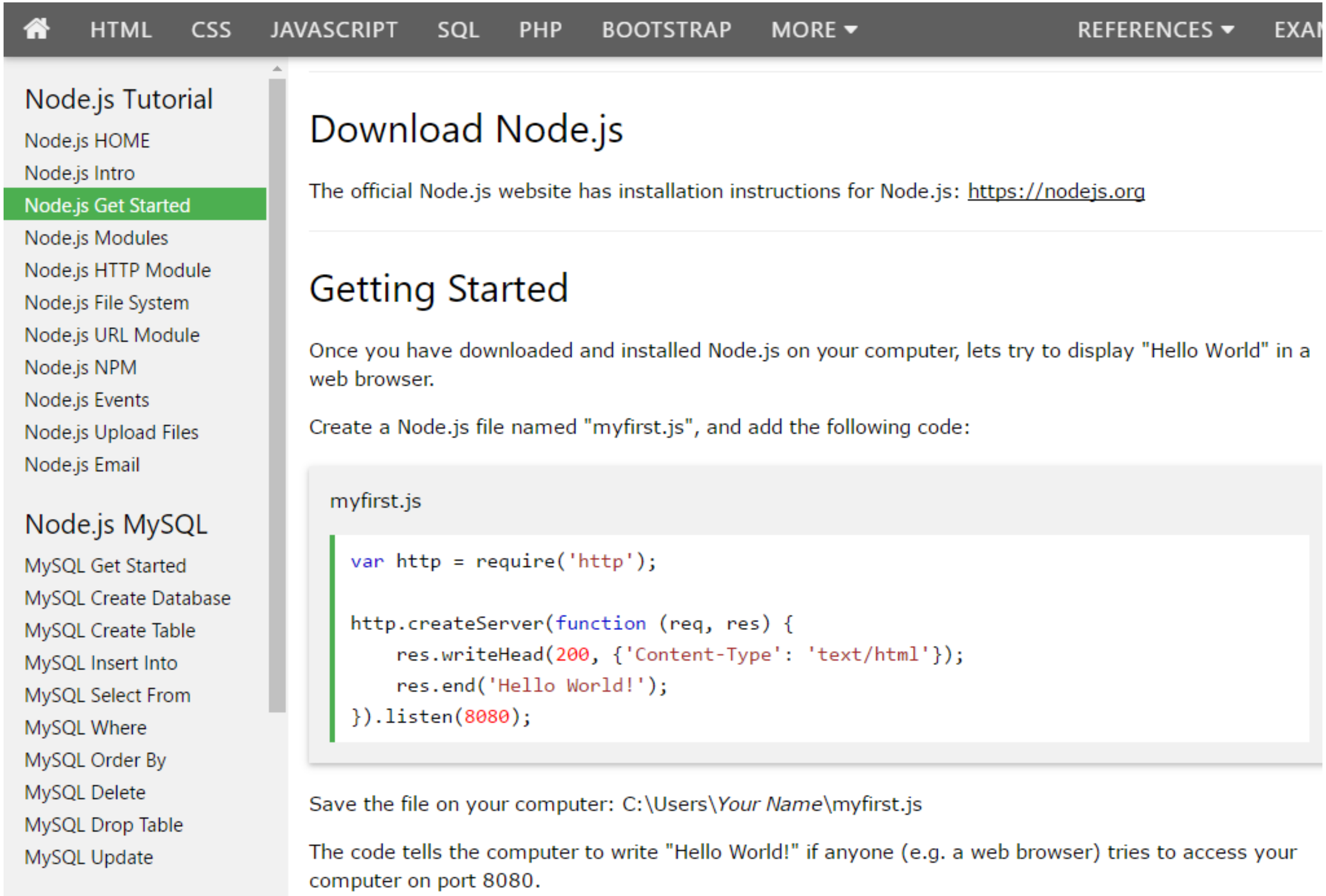
1. Sends the task to the computer's file system.
2. Waits while the file system opens and reads the file.
3. Returns the content to the client.
4. Ready to handle the next request.

Here is how Node.js handles a file request:

1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting, and simply continues with the next request.

Node.js runs single-threaded, non-blocking, asynchronously programming, which is very memory efficient.



Download Node.js

The official Node.js website has installation instructions for Node.js: https://nodejs.org

Getting Started

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

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

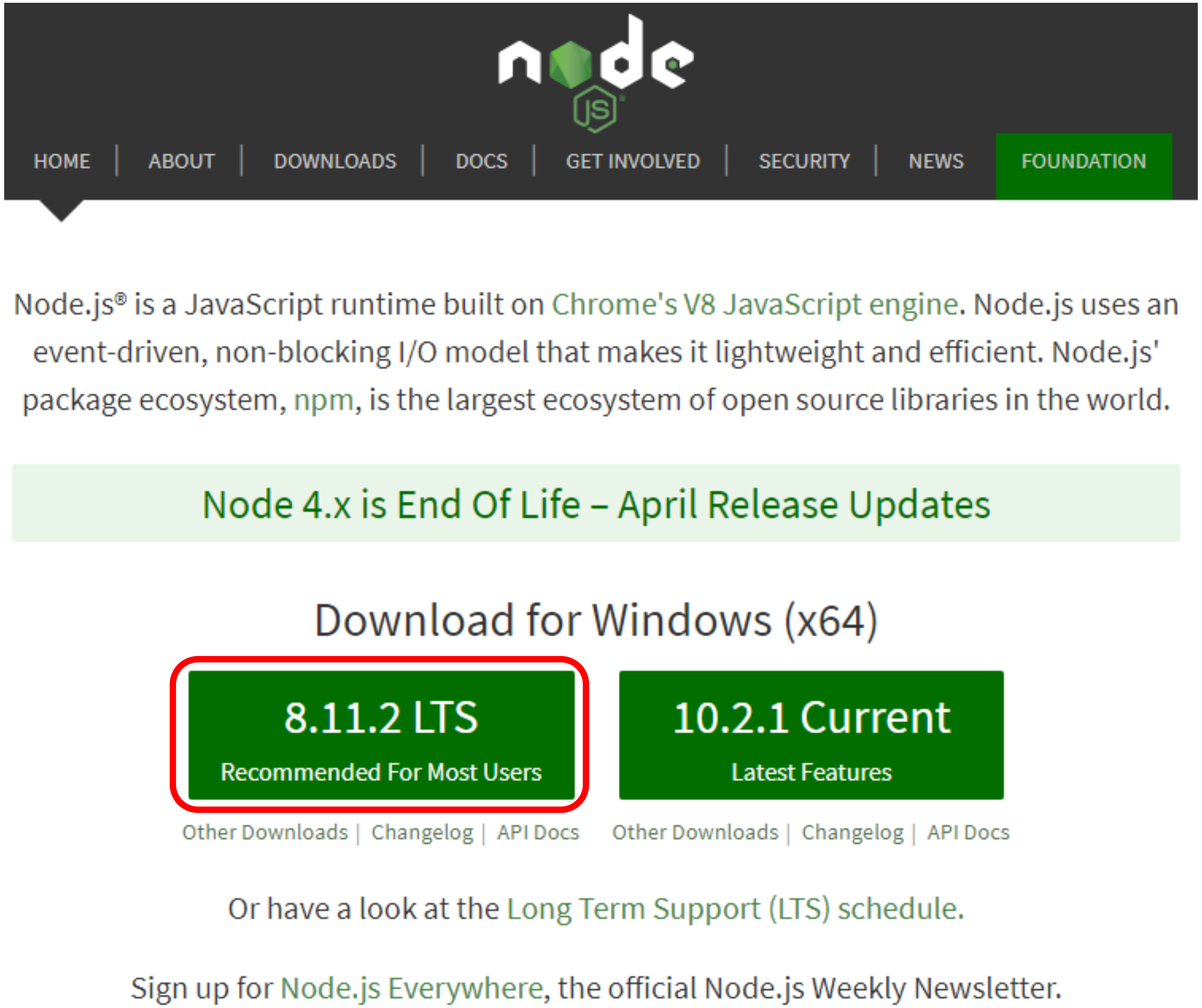
myfirst.js

```
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.

Nodejs.org



node JS
HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | NEWS | FOUNDATION

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

Node 4.x is End Of Life – April Release Updates

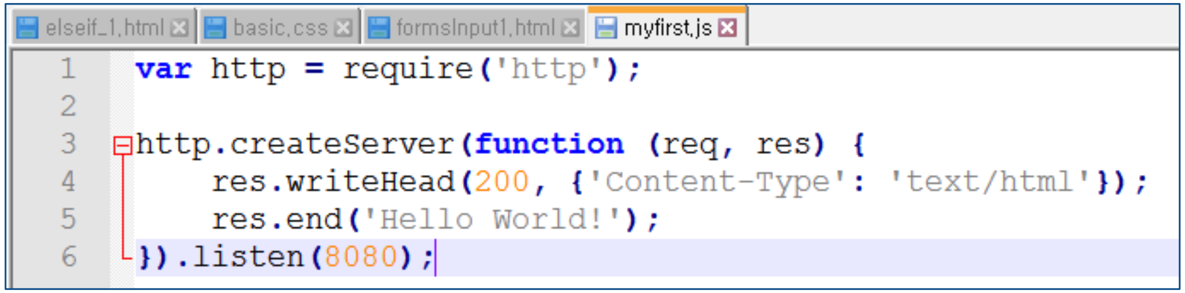
Download for Windows (x64)

8.11.2 LTS
Recommended For Most Users
10.2.1 Current
Latest Features
Other Downloads | Changelog | API Docs
Other Downloads | Changelog | API Docs

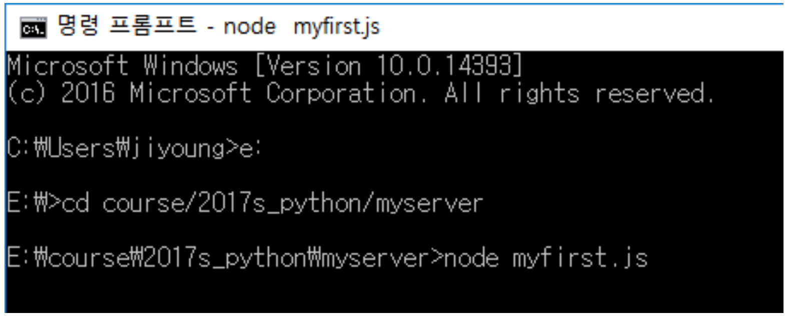
Or have a look at the Long Term Support (LTS) schedule.

Sign up for Node.js Everywhere, the official Node.js Weekly Newsletter.

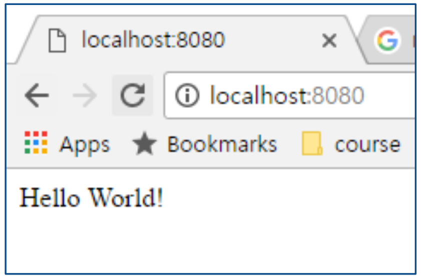
“Hello World!”



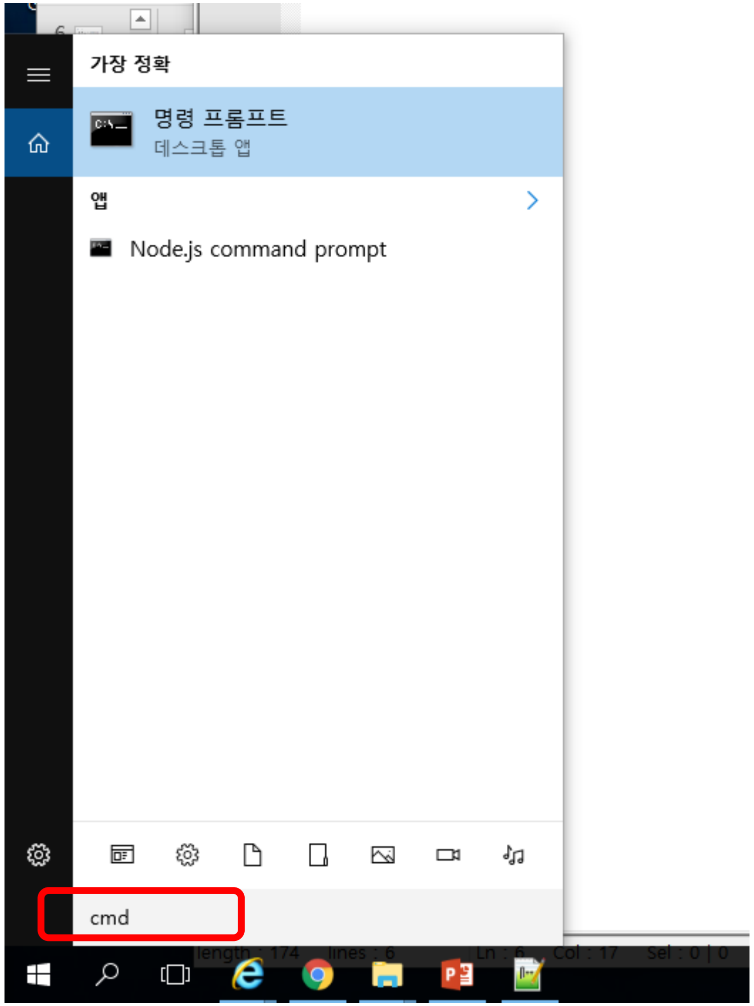
```
var http = require('http');

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



```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:₩Users₩jiyoung>e:

E:₩>cd course/2017s_python/myserver

E:₩course₩2017s_python₩myserver>node myfirst.js
```



Hello World!

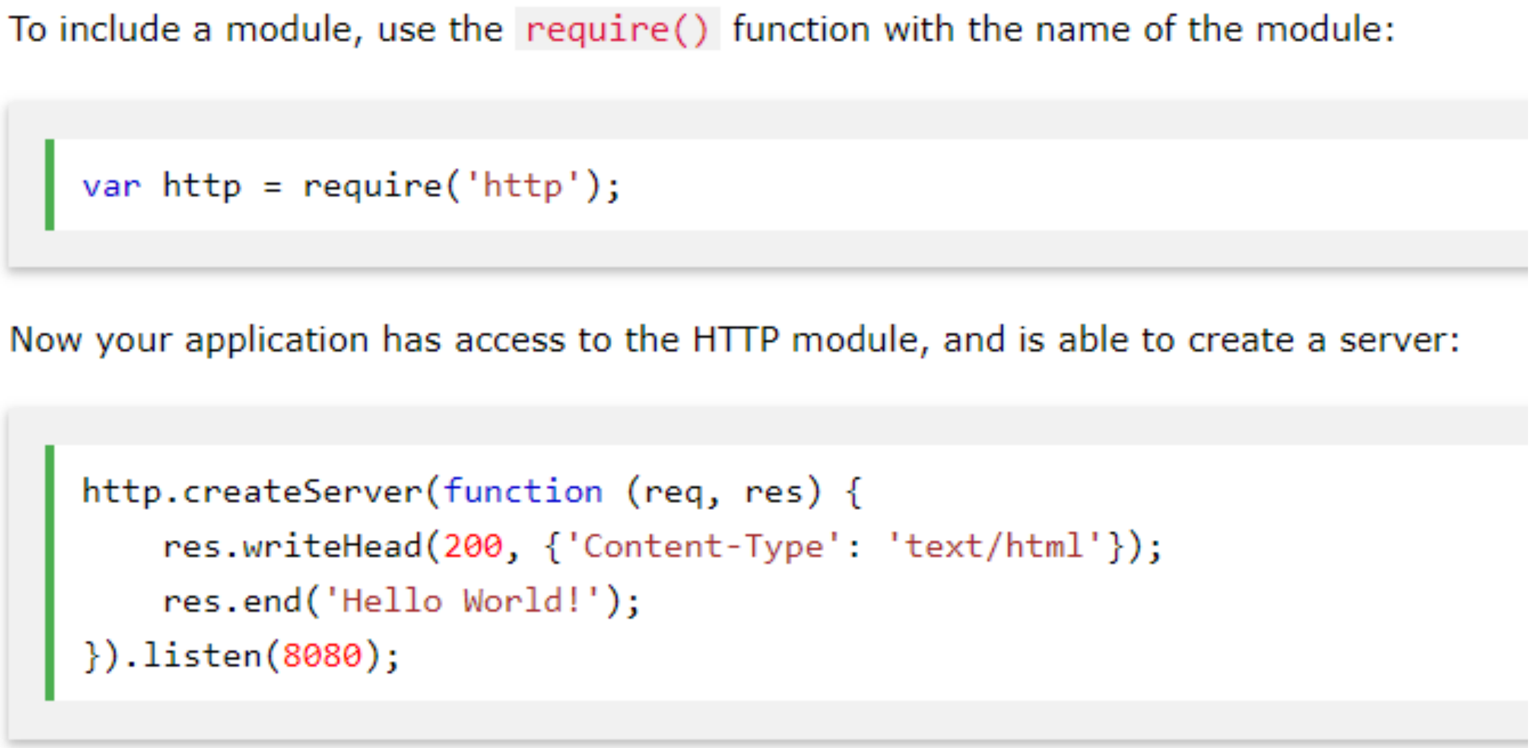


cmd

Modules

- Consider modules to be the same as JavaScript libraries.
- A set of functions you want to include in your application.
- Node.js has a set of built-in modules which you can use without any further installation.
- List of Node.js build-in modules. https://www.w3schools.com/nodejs/ref_modules.asp

Include Modules



To include a module, use the `require()` function with the name of the module:

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

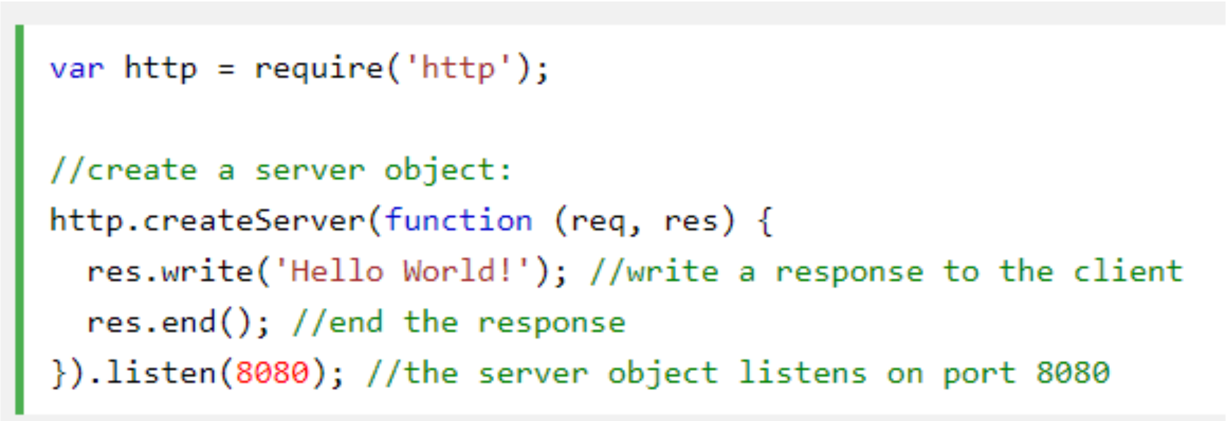
Now your application has access to the HTTP module, and is able to create a server:

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

The Built-in HTTP Module

- Node.js has a built-in module called HTTP, which allows Node.js to transfer data over the Hyper Text Transfer Protocol (HTTP).
- The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client. Use the createServer() method to create an HTTP server:

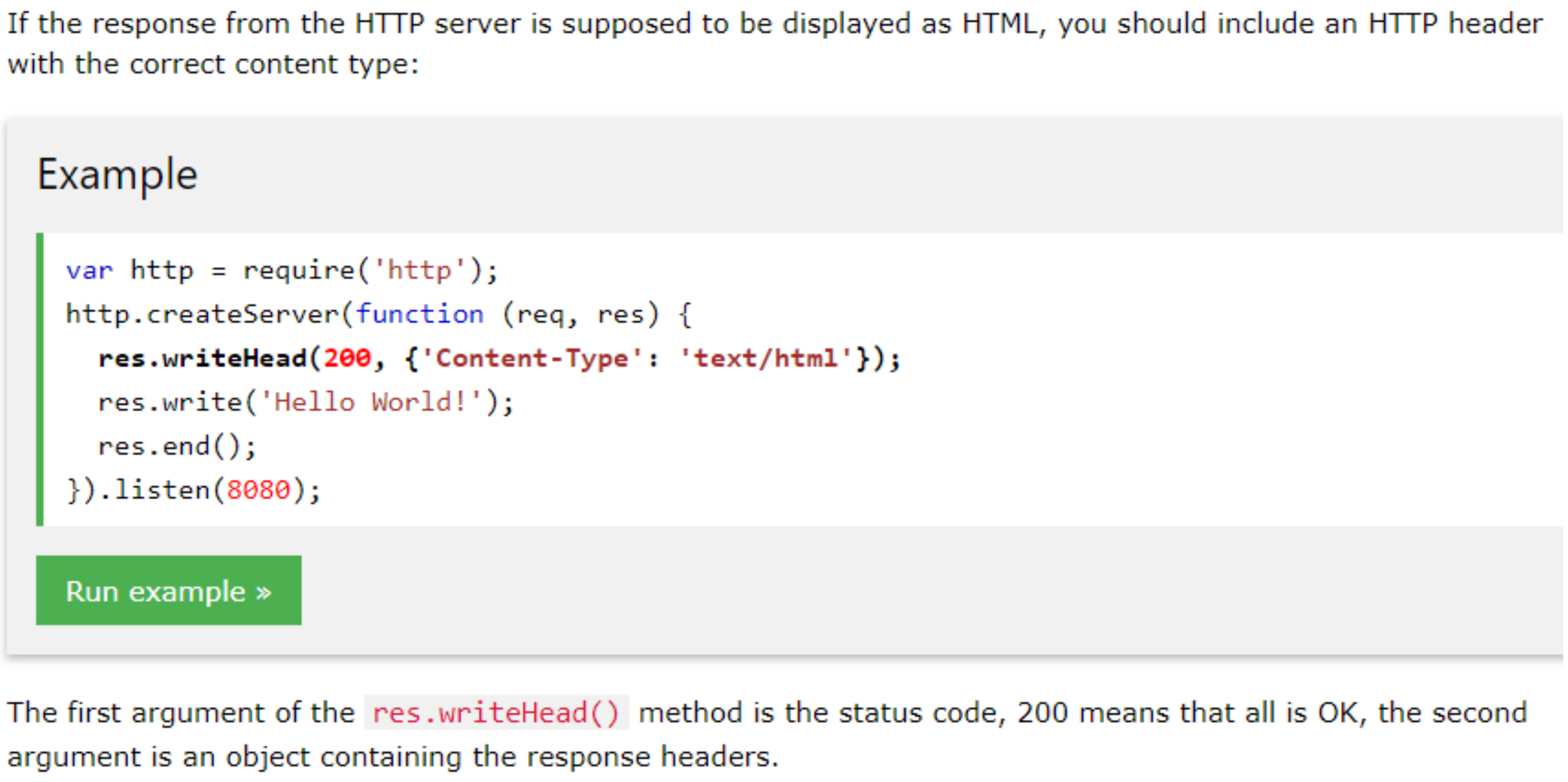
demo_http.js



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

//create a server object:
http.createServer(function (req, res) {
  res.write('Hello World!'); //write a response to the client
  res.end(); //end the response
}).listen(8080); //the server object listens on port 8080
```

The function passed into the createServer() method, will be executed when someone tries to access the computer on port 8080.

Add an HTTP header



If the response from the HTTP server is supposed to be displayed as HTML, you should include an HTTP header with the correct content type:

Example

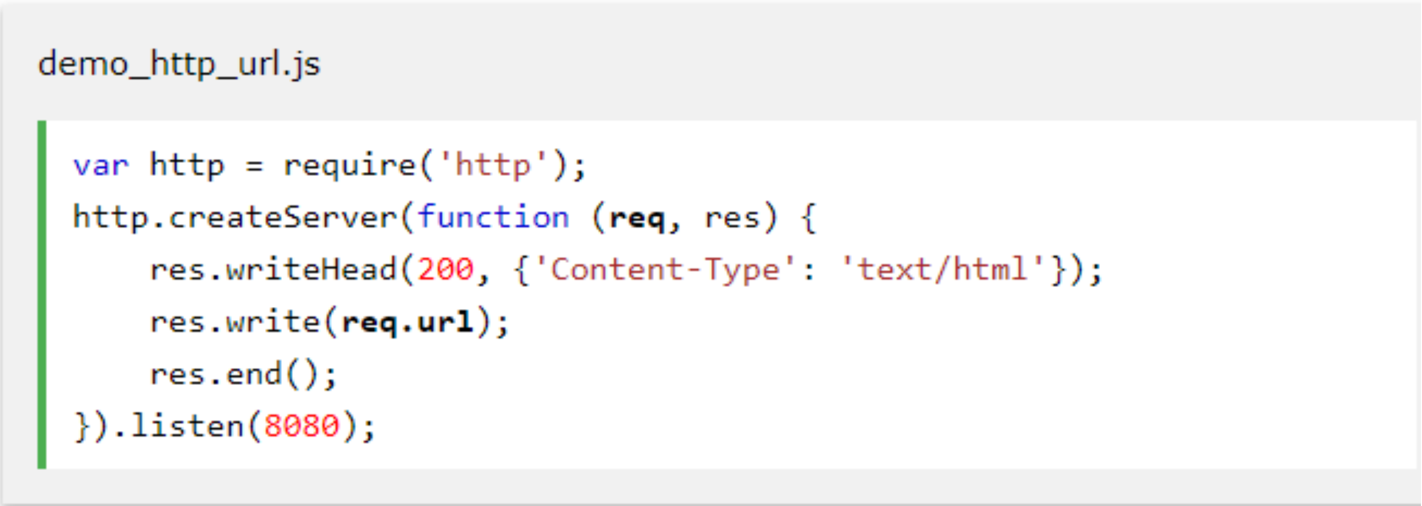
```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write('Hello World!');
  res.end();
}).listen(8080);
```

The first argument of the `res.writeHead()` method is the status code, 200 means that all is OK, the second argument is an object containing the response headers.

Read the Query String (URL)

The function passed into the `http.createServer()` has a `req` argument that represents the request from the client, as an object (http.IncomingMessage object).

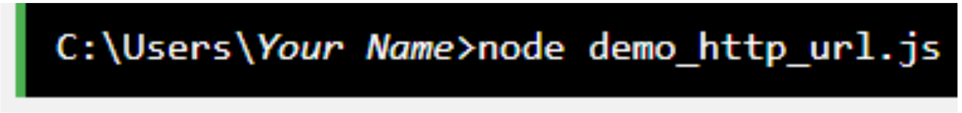
This object has a property called "url" which holds the part of the url that comes after the domain name:



demo_http_url.js

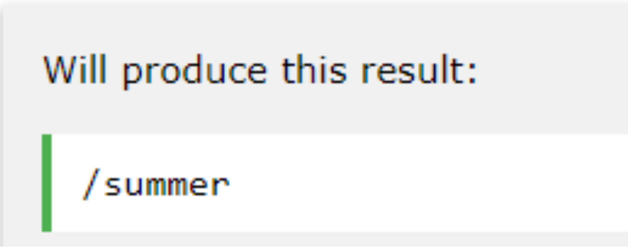
```
var http = require('http');
http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(req.url);
    res.end();
}).listen(8080);
```

Save the code above in a file called "demo_http_url.js" and initiate the file:



```
C:\Users\Your Name>node demo_http_url.js
```

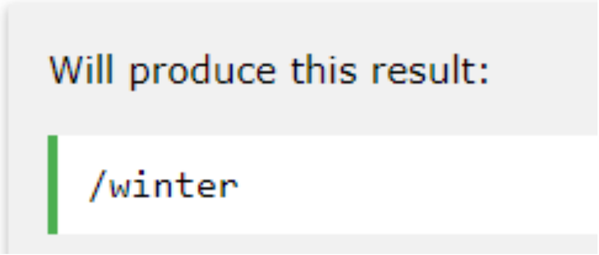
http://localhost:8080/summer



Will produce this result:

```
/summer
```

http://localhost:8080/winter

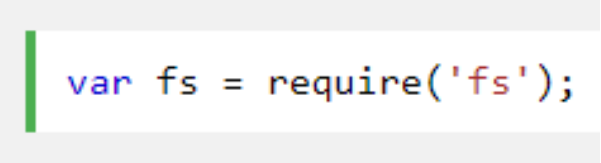


Will produce this result:

```
/winter
```

File System Module

- The Node.js file system module allow you to work with the file system on your computer. To include the File System module,

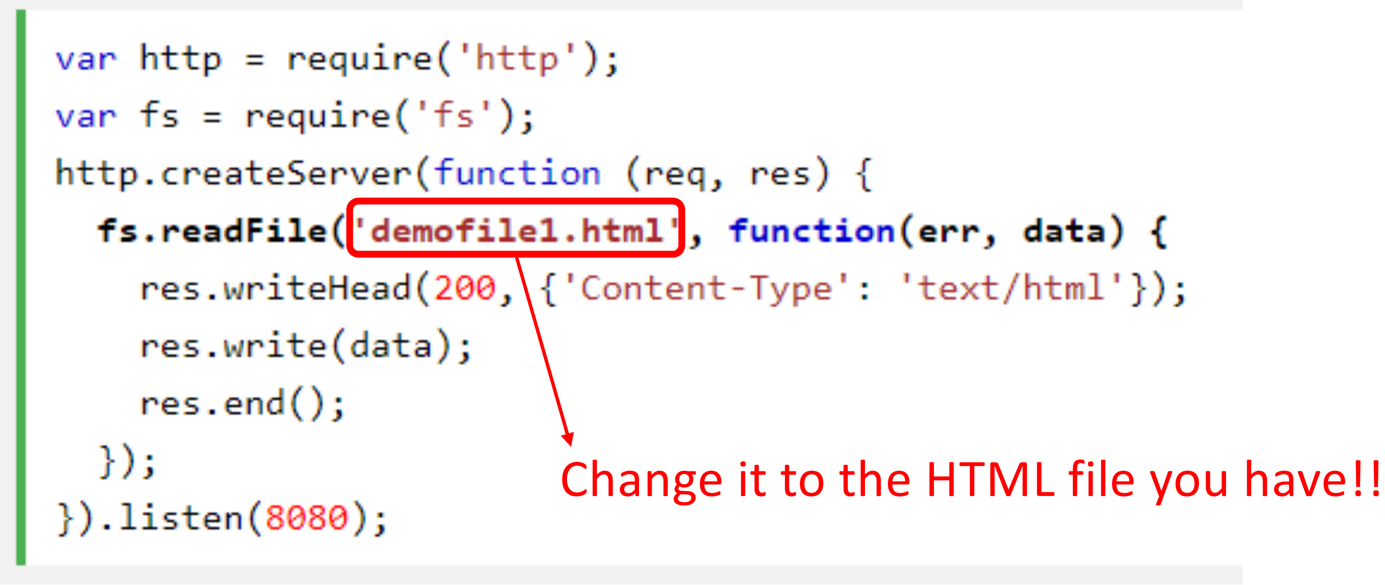


```
var fs = require('fs');
```

- Common use for the File System module:
  - Read files
  - Create files
  - Update files
  - Delete files
  - Rename files

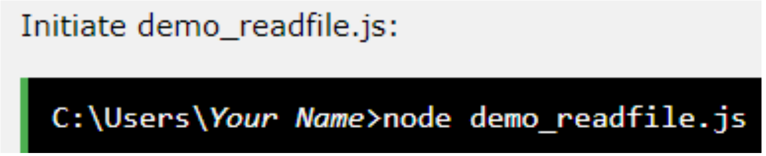
Read files: Send HTML files to Browser

- The fs.readFile() method is used to read files on your computer.
- Save below as demo_readfile.js



```
var http = require('http');
var fs = require('fs');
http.createServer(function (req, res) {
  fs.readFile('demofile1.html', function(err, data) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    res.end();
  });
}).listen(8080);
```

Change it to the HTML file you have!!

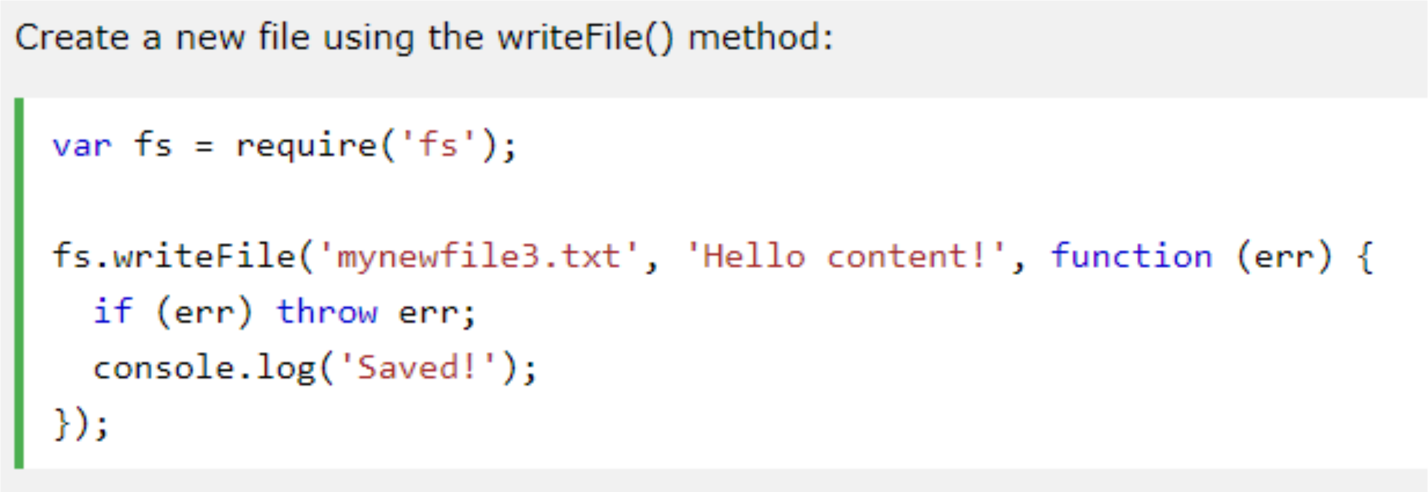


Initiate demo_readfile.js:

```
C:\Users\Your Name>node demo_readfile.js
```

Create/Write Files

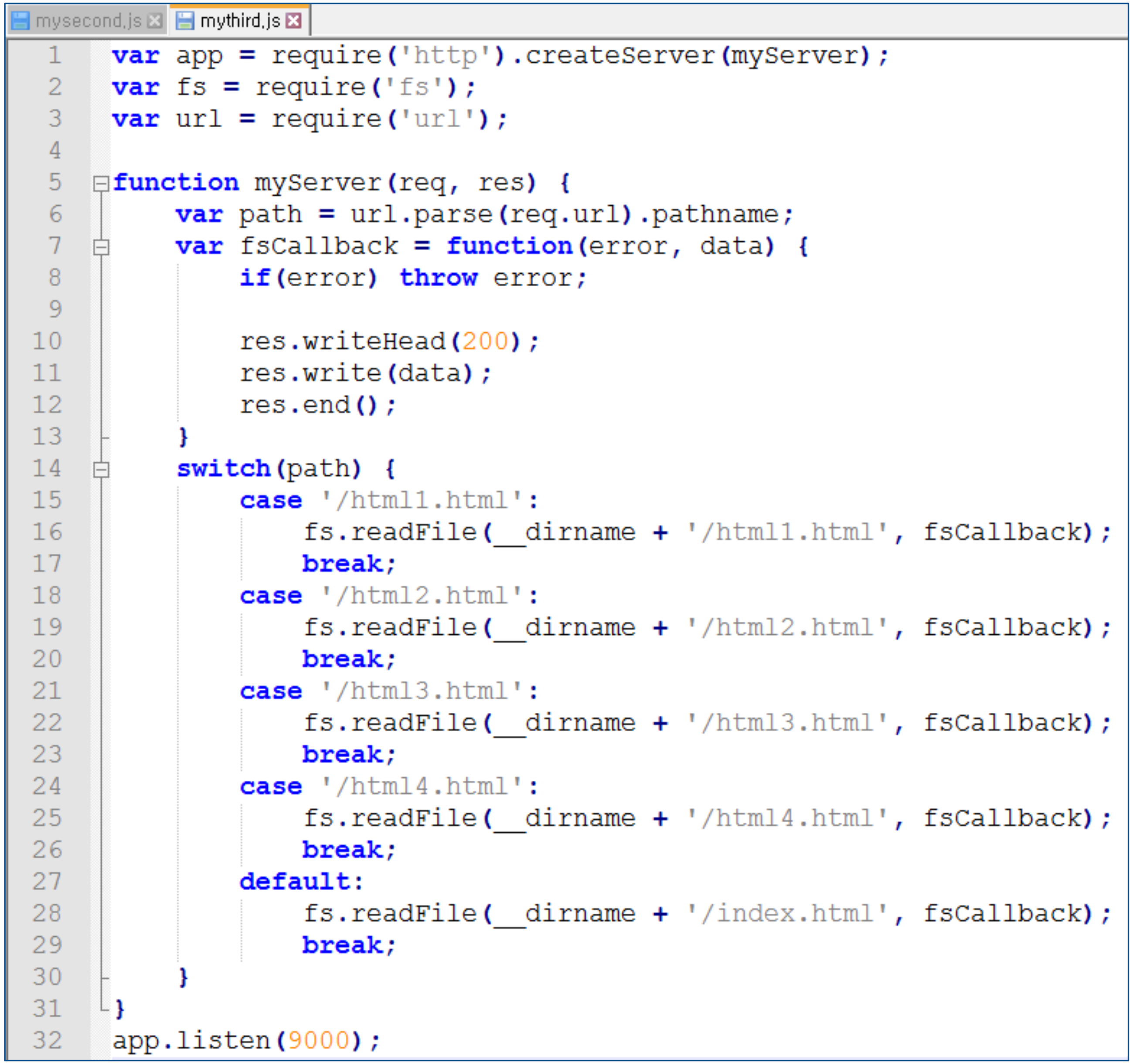
- The File System module has methods for creating new files:
  - fs.appendFile() appends specified content to a file. If the file does not exist, the file will be created.
  - fs.open() takes a "flag" as the second argument, if the flag is "w" for "writing", the specified file is opened for writing. If the file does not exist, an empty file is created.
  - fs.writeFile() replaces the specified file and content if it exists. If the file does not exist, a new file, containing the specified content, will be created.



Create a new file using the writeFile() method:

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

fs.writeFile('mynewfile3.txt', 'Hello content!', function (err) {
  if (err) throw err;
  console.log('Saved!');
});
```

Navigation



```javascript
var app = require('http').createServer(myServer);
var fs = require('fs');
var url = require('url');

function myServer(req, res) {
    var path = url.parse(req.url).pathname;
    var fsCallback = function(error, data) {
        if(error) throw error;

        res.writeHead(200);
        res.write(data);
        res.end();
    }
    switch(path) {
        case '/html1.html':
            fs.readFile(__dirname + '/html1.html', fsCallback);
            break;
        case '/html2.html':
            fs.readFile(__dirname + '/html2.html', fsCallback);
            break;
        case '/html3.html':
            fs.readFile(__dirname + '/html3.html', fsCallback);
            break;
        case '/html4.html':
            fs.readFile(__dirname + '/html4.html', fsCallback);
            break;
        default:
            fs.readFile(__dirname + '/index.html', fsCallback);
            break;
    }
}
app.listen(9000);
```

Summary

- Node.js is an open-source server framework.
- Node.js allows you to run JavaScript on the server.
- Node.js uses asynchronous programming.
- You can now use HTTP, URL, and File System modules in Node.js to implement a web server.
- When the browser sends a request to the server, you can read and send the file of request, and process query parameters from the user.