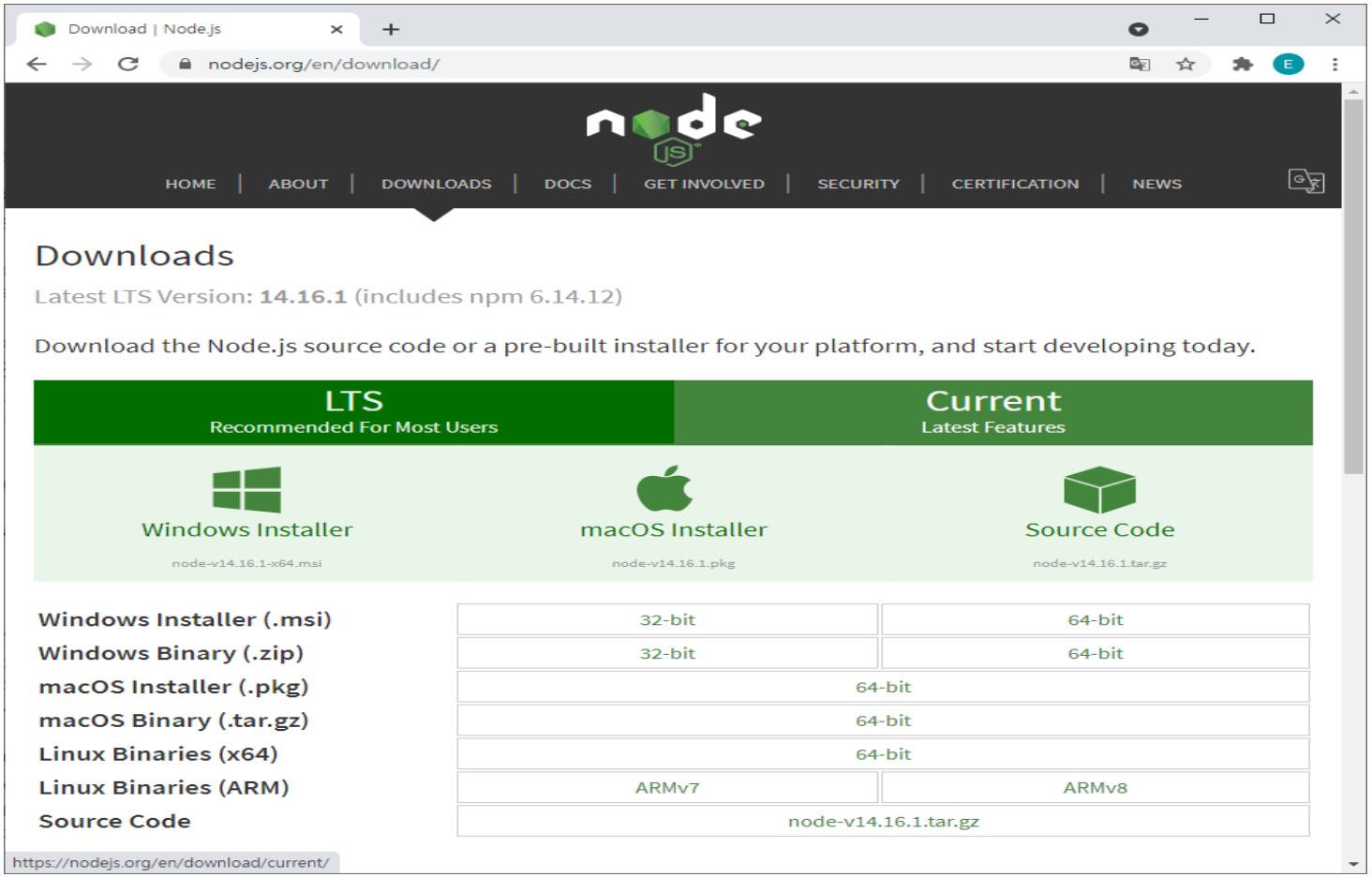
Node.js

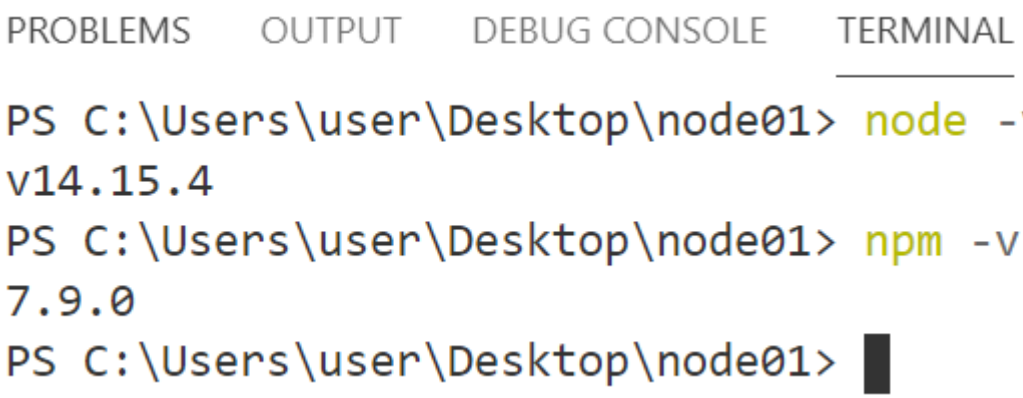
Node.js

- **An open-source JavaScript runtime environment**
  - Built on Chrome's V8 JavaScript engine
  - It allows you to run **JavaScript on the server**
  - It runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)

- **Node JS applications:**
  - Netflix
  - LinkedIn
  - Wal-Mart
  - Paypal
  - YouTube
  - Amazone.com
  - eBay
  - Reddit
  - …

Node.js 설치


Download | Node.js
nodejs.org/en/download/
HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS
Downloads
Latest LTS Version: 14.16.1 (includes npm 6.14.12)
Download the Node.js source code or a pre-built installer for your platform, and start developing today.
LTS
Recommended For Most Users
Current
Latest Features
Windows Installer
node-v14.16.1-x64.msi
macOS Installer
node-v14.16.1.pkg
Source Code
node-v14.16.1.tar.gz
Windows Installer (.msi) 32-bit 64-bit
Windows Binary (.zip) 32-bit 64-bit
macOS Installer (.pkg) 64-bit
macOS Binary (.tar.gz) 64-bit
Linux Binaries (x64) 64-bit
Linux Binaries (ARM) ARMv7 ARMv8
Source Code node-v14.16.1.tar.gz
https://nodejs.org/en/download/current/


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\user\Desktop\node01> node -v
v14.15.4
PS C:\Users\user\Desktop\node01> npm -v
7.9.0
PS C:\Users\user\Desktop\node01>

Creating Server

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

http.createServer(function (request, response) {
   // Send the HTTP header
   // HTTP Status: 200 : OK
   // Content Type: text/plain
   response.writeHead(200, {'Content-Type': 'text/plain'});

   // Send the response body as "Hello World"
   response.end('Hello World₩n');
}).listen(8081);

// Console will print the message
console.log('Server running at http://127.0.0.1:8081/');
```

Modules

- **Module**
  - Consider modules to be the same as JavaScript **libraries**.
  - A **set of functions** you want to include in your application.
- **Built-in Modules**
  - os
  - url
  - Query String
  - util
  - crypto
  - File system
  - ….
- **Include Modules**
  - Use the require() function with the name of the module:

HTTP Module

- **A built-in HTTP module**
  - It allows Node.js to transfer data over the Hyper Text Transfer Protocol
  - It can create an HTTP server that listens to server ports and gives a response back to the client.

```
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
```

- **Add an HTTP Header**
  - If the response from the HTTP server is supposed to be displayed as HTML:

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

- **Create Your Own Modules:**
  - Use the **exports** keyword to make properties and methods available outside the module file.

```
exports.myDateTime = function () {
  return Date();
};
```

  - Include Your Own Module

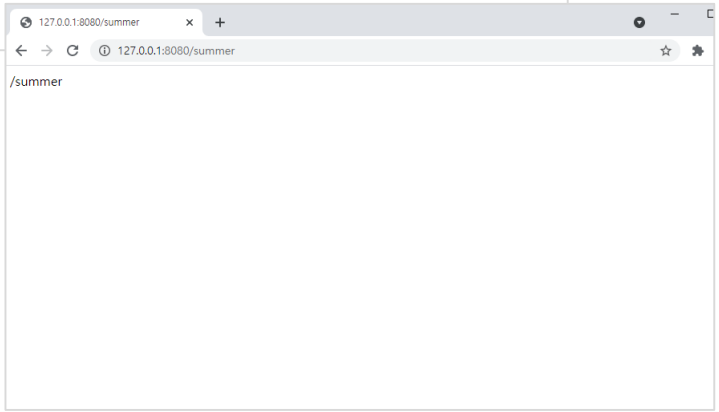
```
var http = require('http');
var dt = require('./myfirstmodule');

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write("The date and time are currently: " + dt.myDateTime());
  res.end();
}).listen(8080);
```

moduleEx.js

- **req.url :**

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

http://localhost:8080/summer



queryString.js

URL Module

- It splits up a web address into readable parts.
- Use **url.parse()** method

```
var url = require('url');
var adr = 'http://localhost:8080/default.htm?year=2017&month=february';
var q = url.parse(adr, true);

console.log(q.host); //returns 'localhost:8080'
console.log(q.pathname); //returns '/default.htm'
console.log(q.search); //returns '?year=2017&month=february'

var qdata = q.query; //returns an object: { year: 2017, month: 'february' }
console.log(qdata.month); //returns 'february'
```

urlparse.js

File System Module

- It allows you to work with the file system on your computer
  - Read / Create / Update / Delete / Rename files

- **Read Files : fs.readFile()**

```
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);
    return res.end();
  });
}).listen(8080);
```

fs.js

```
<html>
<body>
<h1>My Header</h1>
<p>My paragraph.</p>
</body>
</html>
```

Create Files

- **fs.appendFile()**
- **fs.open()**
- **fs.writeFile()**

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

fs.appendFile('mynewfile1.txt', 'Hello content!', function (err) {
  if (err) throw err;
  console.log('Saved!');
});
```

fsappend.js

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

fs.open('mynewfile2.txt', 'w', function (err, file) {
  if (err) throw err;
  console.log('Saved!');
});
```

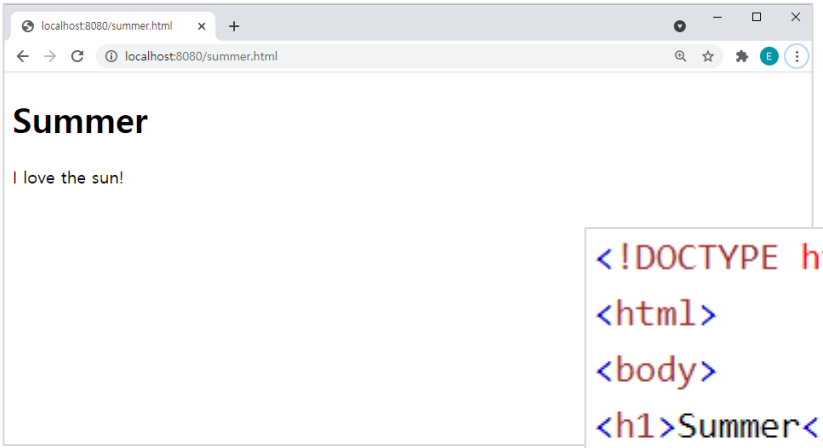
```
var fs = require('fs');

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

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

fs.writeFile('input.txt', 'Simply Easy Learning!', function(err) {
   if (err) {
      return console.error(err);
   }
   console.log("Data written successfully!");

   fs.readFile('input.txt', function (err, data) {
      if (err) {
         return console.error(err);
      }
      console.log(data.toString());
   });
});
```

- **Ex) File Server**
  - Create a Node.js file that opens the requested file and returns the content to the client.
  - If anything goes wrong, throw a 404 error:

http://localhost:8080/summer.html

http://localhost:8080/winter.html




```
<!DOCTYPE html>
<html>
<body>
<h1>Summer</h1>
<p>I love the sun!</p>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<body>
<h1>Winter</h1>
<p>I love the snow!</p>
</body>
</html>
```

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

http.createServer(function (req, res) {
  var q = url.parse(req.url, true);
  var filename = "." + q.pathname;
  fs.readFile(filename, function(err, data) {
    if (err) {
      res.writeHead(404, {'Content-Type': 'text/html'});
      return res.end("404 Not Found");
    }
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    return res.end();
  });
}).listen(8080);
```

http://localhost:8080/summer.html

http://localhost:8080/winter.html

fsserver.js

Delete Files

- **fs.unlink()**

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

fs.unlink('mynewfile2.txt', function (err) {
  if (err) throw err;
  console.log('File deleted!');
});
```

fsdelete.js

Rename Files

- **fs.rename()**

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

fs.rename('mynewfile1.txt', 'myrenamedfile.txt', function (err) {
  if (err) throw err;
  console.log('File Renamed!');
});
```

fsrename.js

Get File Information

- **fs.stat(path, callback)**

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

console.log("Going to get file info!");
fs.stat('input.txt', function (err, stats) {
   if (err) {
      return console.error(err);
   }
   console.log(stats);
   console.log("Got file info successfully!");

   // Check file type
   console.log("isFile ? " + stats.isFile());
   console.log("isDirectory ? " + stats.isDirectory());
});
```

fsstat.js

Callback

- **A function** passed **as an argument to another function.**
  - It allows a function to call another function.

- A callback function can **run after** another function has finished.

- **Node makes heavy use of callbacks.**
  - All the APIs are written in such a way that they support callbacks.

- **NoCallback.js**

```
var fs = require("fs");
var data = fs.readFileSync('input.txt');

console.log(data.toString());
console.log("Program Ended");
```

callback01.js

- **Callback.js**

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

fs.readFile('input.txt', function (err, data) {
   if (err) return console.error(err);
   console.log(data.toString());
});

console.log("Program Ended");
```

callback02.js

Events

- Node.js is a **single-threaded** application, but it can support concurrency via the concept of **event** and **callbacks**.

- Node.js uses **events** heavily and it is also one of the reasons why Node.js is pretty **fast** compared to other similar technologies.

- Every action on a computer is an event
  - Like when a connection is made or a file is opened.

```
var fs = require('fs');
var rs = fs.createReadStream('./demofile.txt');
rs.on('open', function () {
  console.log('The file is open');
});
```

events.js

Events Module

- Node. js allows us to create and handle **custom events** easily by using **events** module
  - 1) use the require() method and create an EventEmitter object:

```
// Import events module
var events = require('events');

// Create an eventEmitter object
var eventEmitter = new events.EventEmitter();
```

  - 2) bind an event handler with an event

```
// Bind event and event  handler as follows
eventEmitter.on('eventName', eventHandler);
```

  - 3) fire an event

```
// Fire an event
eventEmitter.emit('eventName');
```

```
var events = require('events');
var eventEmitter = new events.EventEmitter();

//Create an event handler:
var myEventHandler = function () {
  console.log('I hear a scream!');
}

//Assign the event handler to an event:
eventEmitter.on('scream', myEventHandler);

//Fire the 'scream' event:
eventEmitter.emit('scream');
```

eventEmitter.js

```
var events = require('events');
var eventEmitter = new events.EventEmitter();

var connectHandler = function connected() {
   console.log('connection succesful.');
   eventEmitter.emit('data_received');
}

eventEmitter.on('connection', connectHandler);
eventEmitter.on('data_received', function() {
   console.log('data received succesfully.');
});

eventEmitter.emit('connection');

console.log("Program Ended.");
```

eventex.js

NPM

- **Node Package Manager**
  - Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.

- **Installing Modules**

  $ npm install <Module Name>

- **Global vs Local Installation**

  $ npm install express

  $ npm install express –g

- To check all the modules installed globally :

  $ npm ls –g

- **Ex)** $npm install upper-case

```
var http = require('http');
var uc = require('upper-case');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  /*Use our upper-case module to upper case a string:*/
  res.write(uc.upperCase("Hello World!"));
  res.end();
}).listen(8081);

console.log('Server running at http://127.0.0.1:8081/');
```

Node.js

- **Node.js**
  - HTTP
  - FS
  - URL
  - NPM
  - Event