

# NODE JS

JavaScript on the server



INSTRUCTOR:

# LAURENCE SVEKIS



- Over 300 courses in technology and web applications.
- 20 years of web programming experience
- 600,000+ students across multiple platforms
- Digital instructor since 2002

**READY TO HELP YOU LEARN and ANSWER ANY questions you may have.**

**Course instructor : Laurence Svekis**

# Node Course Guide

You will need to know **JavaScript and Command line**

JavaScript and Command Line Ready

<https://nodejs.org/en/>  
<https://www.npmjs.com/> played a large role in how code was shared

## Windows Terminal

<https://cmdr.net/>

## Mac Terminal

Command+Space or select terminal in the applications list.

**JavaScript on the server.**

**open source** server environment find on Github

can **connect to a database**

**Create, open, read write, delete files on your server**

**Generate page content**

Node.js runs single-threaded very memory efficient.

**JavaScript uses browsers to run code on the frontend.**

On the **backend needs to be interpreted** (executed). **Node.js uses Google's V8 VM** which is the **same runtime environment the Chrome uses.**

Node.js comes with many useful modules - **saves from writing from scratch.**

**Node is runtime environment and a library**

# Windows Terminal

Windows - <https://cmder.net/> or use the command prompt terminal

**Launch the Command Prompt** - use the Run window or press the Win + R keys on your keyboard. Then, type cmd and press Enter.

List current directory of files - **dir**

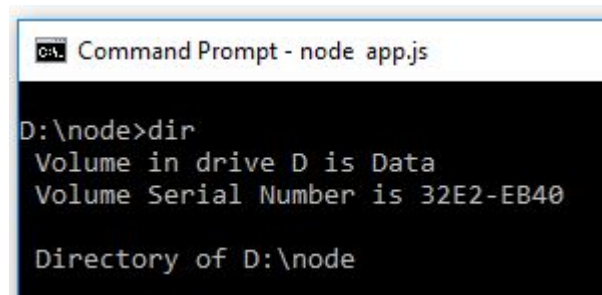
Change directory to D drive - **cd D:**

Change directory down one level - **cd..**

Change directory to folder by name - **cd folder**

Make new folder - **mkdir folderName**

Get help - **help**



```
C:\> Command Prompt - node app.js

D:\node>dir
Volume in drive D is Data
Volume Serial Number is 32E2-EB40

Directory of D:\node
```



# Mac Terminal

Open Terminal by pressing

List current directory of files - **ls**

Change directory to D drive - **cd D:**

Change directory down one level - **cd..**

Change directory to folder by name - **cd  
folder**

Make new folder - **mkdir folderName**

Get help - **help**



# Single Thread

Node.js is a single-threaded asynchronous JavaScript runtime. This means that your code will be executed in the same thread.

Can do more than one thing at time  
Doesn't have to wait for last order to complete and will pick up where it left off when ready.



# Command Line Launch

One node is installed check to see version installed. Type **node -v**

Open your editor and create a js file that contains **console.log('Hello World');** save it as test.js

In the terminal type **node test.js** and watch for a result. What gets returned?

NPM - check if its installed **npm -v** latest version install **npm install npm@latest -g**

```
node -v  
  
console.log('Hello World');  
  
node test.js  
  
npm install npm@latest -g
```

# Global Object

<https://nodejs.org/api/globals.html>

require('path');

Process objects - version , pid , env

```
},
✓[[Symbol(kCapture)]: false
} 3];
Hi, undefined undefined
webs-iMac:node web$ node app.js Laurence Svekis
```

```
global.console.log('hello');

console.log(global);
console.log(process.pid);
console.log(__dirname);
console.log(__filename);

console.log(process.argv);

const first = process.argv[2];
const last = process.argv[3];

let message = `Hi, ${first} ${last}`;
console.log(message);
```



# Process options

<https://nodejs.org/api/process.html>

- `process.report.reportOnFatalError`
- `process.report.reportOnSignal`
- `process.report.reportOnUncaughtException`
- `process.report.signal`
- `process.report.writeReport([filename][, err])`
- `process.resourceUsage()`
- `process.send(message[, sendHandle[, options]][, callback])`
- `process.setegid(id)`
- `process.seteuid(id)`
- `process.setgid(id)`
- `process.setgroups(groups)`
- `process.setuid(id)`
- `process.setUncaughtExceptionCaptureCallback(fn)`
- `process.stderr`
  - `process.stderr.fd`
- `process.stdin`
  - `process.stdin.fd`
- `process.stdout`
  - `process.stdout.fd`
  - A note on process I/O
- `process.throwDeprecation`

```
process.stdout.write('Laurence');
process.stdout.write('\nSvekis\n');
const arr = ["Dog", "Cat", "Rabbit"];
const animals = function (i = 0) {
  process.stdout.write(`${arr[i]}\n`);
}
animals();
process.stdin.on('data', function (data) {
  process.stdout.write(`You typed ${data}`);
  process.exit(); //exits program
}); //type into keyboard and press enter
//application will be open
//control+c to stop or type exit
process.on('exit', (code) => {
  console.log(`You exited`);
});

const num = Math.floor(Math.random() * 10) + 1;
process.stdout.write('Guess a number from 1 - 10\n');
process.stdin.on('data', data => {
  let guess = data;
  process.stdout.write(`Your guess is ${guess}`);
  if (guess == num) {
    process.stdout.write(`Correct it was ${num}\n`);
    process.exit();
  }
  else {
    process.stdout.write(`Wrong guess again\n`);
  }
})
```

# Intervals and Timeouts



```
const timer = 5000;
const outputInterval = 1000;
let val = 0;
process.stdout.write(`${timer/1000} second delay`);
const ready = () => {
  output('ready');
  process.stdout.write('\n');
  clearInterval(myInt);
};
const counter = () => {
  val++;
  output(`${(timer/1000)-val} seconds left`);
}
const output = (mess) => {
  process.stdout.clearLine();
  process.stdout.cursorTo(0);
  process.stdout.write(mess);
}
const myInt = setInterval(counter, outputInterval);
setTimeout(ready, timer);
```

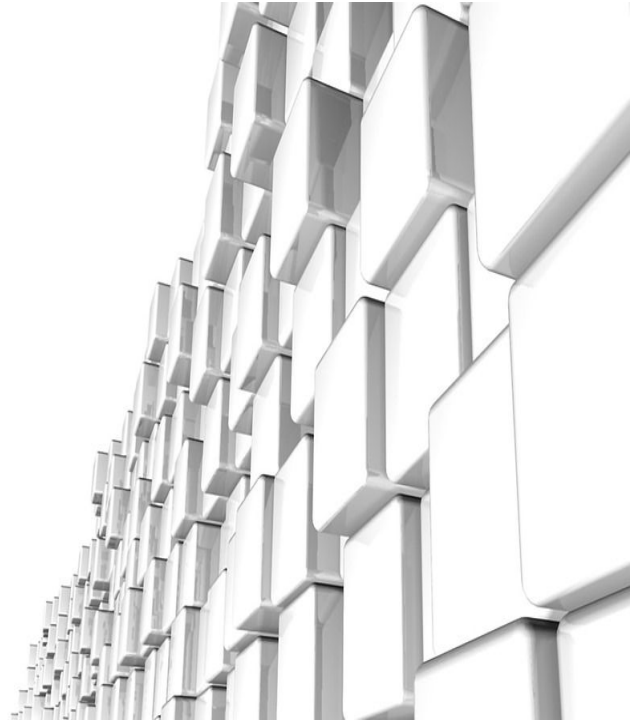
# Node modules

Modules in node are like JavaScript libraries. Pre Written code that you can pull into your project and use.

Node.js has a set of built-in modules which you can use without any additional installation.

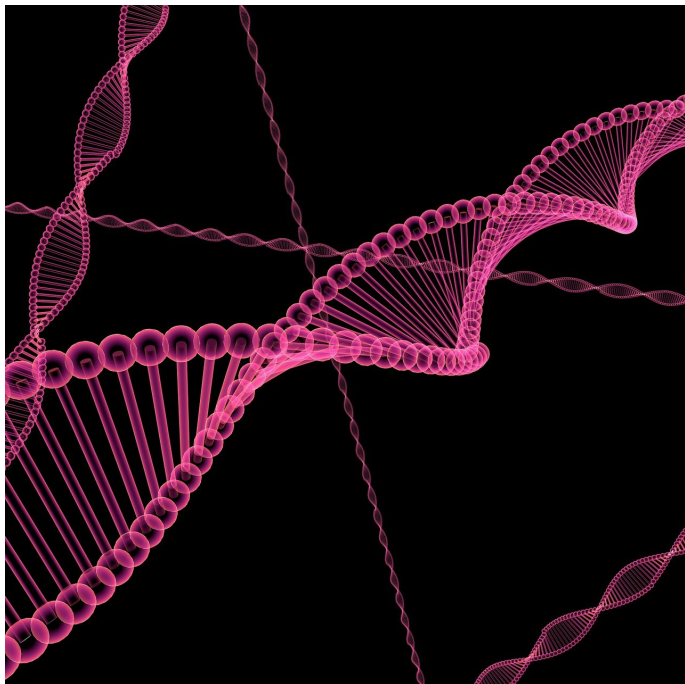
To include a module, use the `require()` function with the name of the module.

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



# Modules

<https://nodejs.org/api/modules.html>



## app.js

```
const pathMod = require('path');
console.log(__filename);
console.log(pathMod.basename(__filename));
const utilMod = require('util');
//console.log(utilMod);
utilMod.log(pathMod.basename(__filename));
const first = require('./mods/first.js');
first();
const ranNum = require('./mods/randomNum.js');
console.log(ranNum.random(50));
```

## first.js

```
const welcome = () => {
  console.log('welcome');
}
module.exports = welcome;
//console.log(module);
```

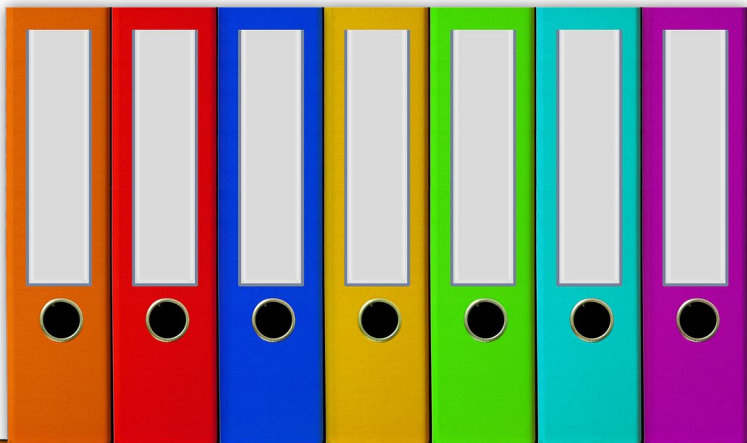
## randomNum.js

```
module.exports.random = (max) => {
  let rep = Math.floor(Math.random()*max);
  return rep;
}

//console.log(module);
```

# Fs Module

<https://nodejs.org/api/fs.html>



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

const myFiles = fs.readdirSync('./mods');
console.log(myFiles);

fs.readdir('./mods',(error,files)=>{
  if(error){throw error;}
  console.log('readdir');
  console.log(files);
})

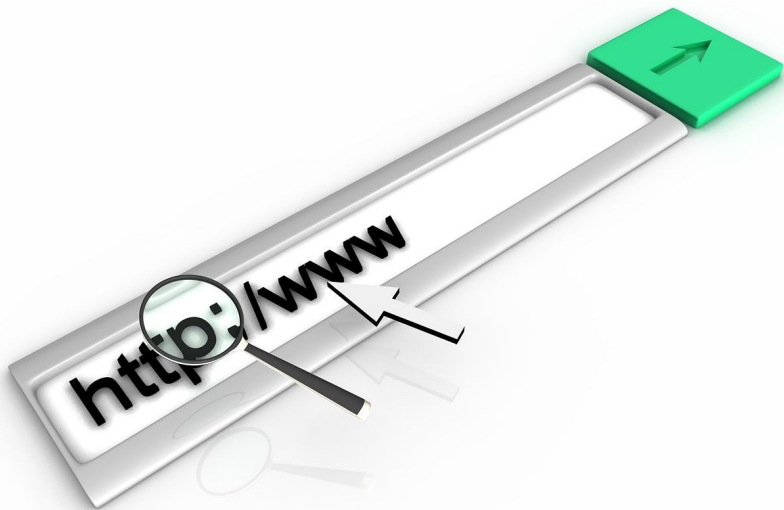
const getSec = fs.readFileSync('./mods/test.txt','UTF-8');
console.log(getSec);

fs.readFile('./mods/test.txt','UTF-8',(error,output)=>{
  if(error){throw error;}
  console.log(output);
});

const myText = `UPDATED`;
fs.writeFile('./mods/secret.txt',myText,error => {
  if(error){throw error;}
  console.log('File ready');
});
```

# HTTP module

<https://nodejs.org/api/http.html>

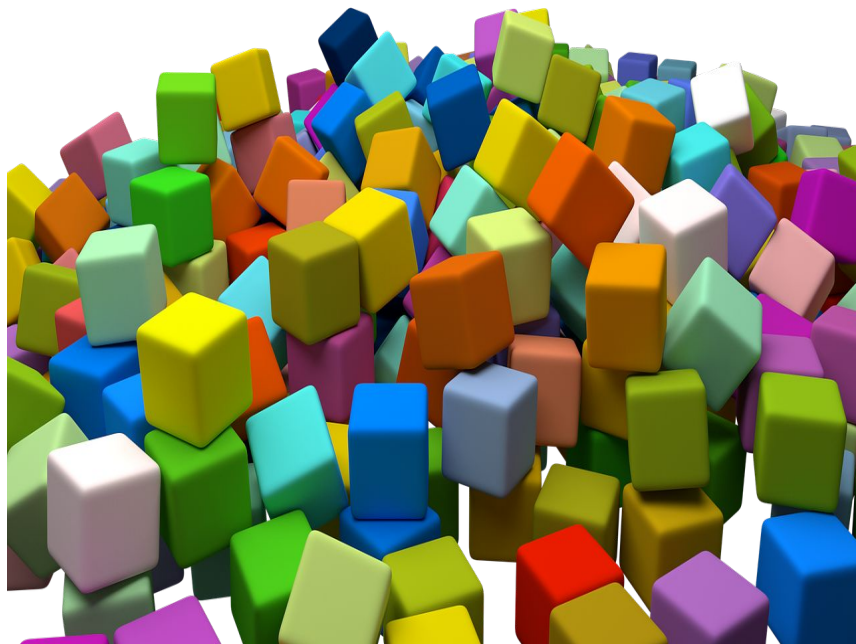


```
const http = require('http');
const fs = require('fs');
const port = 8080;
const host = '127.0.0.1';
const server = http.createServer((request, response) => {
  /*
   console.log(request.url);
   response.writeHead(200, {
     'Content-Type': 'text/html'
   });
   response.write(`Request URL : ${request.url}<br>`);
   response.write('Hello World!'); //write a response to the
client
   response.end(); //end the response

  */
  fs.readFile('./mods/index.html', (error, html) => {
    response.writeHead(200, {
      'Content-Type': 'text/html'
    });
    response.write(html);
    response.end();
  })
});
server.listen(port, host, () => {
  console.log('server running');
});
```

# Files from Folder output

## Reading files and folders



```
const http = require('http');
const fs = require('fs');
const url = require('url');
const port = 8080;
const host = '127.0.0.1';
const server = http.createServer((request, response) => {
  const base = url.parse(request.url);
  //console.log(base);
  const fileName = './mods' + base.pathname;
  fs.readFile(fileName, (error, html) => {
    if (error) {
      response.writeHead(404, {
        'Content-Type': 'text/html'
      });
      response.end(`<h1>Not Found</h1>`);
    }
    else {
      response.writeHead(200, {
        'Content-Type': 'text/html'
      });
      response.write(html);
      response.end();
    }
  })
})
server.listen(port, host, () => {
  console.log('ready');
})
```

# Node Package Manager

<https://lodash.com/>

<https://www.npmjs.com/package/lodash>

Lo

Lodash


A modern JavaScript utility library delivering modularity, performance & extras.

```
_.defaults({ 'a': 1 }, { 'a': 3, 'b': 2 });  
// → { 'a': 1, 'b': 2 }  
_.partition([1, 2, 3, 4], n => n % 2);  
// → [[1, 3], [2, 4]]
```

 Star 44,250  Fork 4,825  Follow @bestiejs  Tweet

## Download

 [Core build \(~4kB gzipped\)](#)

 [Full build \(~24kB gzipped\)](#)

 [CDN copies](#)

Lodash is released under the [MIT license](#) & supports modern environments.  
Review the [build differences](#) & pick one that's right for you.

```
$ sudo npm i -g npm  
$ sudo npm i --save lodash
```

```
const lodash = require('lodash');  
let random1 = lodash.random(100);  
console.log(random1);  
let random2 = lodash.random(50, 100);  
console.log(random2);  
const arr = [3, 4, 56, 76, 'test', 765, 65, 56];  
console.log(lodash.shuffle(arr));  
console.log(lodash.shuffle(arr));  
lodash.times(10, () => {  
  console.log(lodash.random(50, 100));  
})
```



# Loading Packages NPM

Popular package called lodash

<https://www.npmjs.com/package/lodash>

## npm install lodash

- Installs a new folder with all dependant files.
- Warnings if no package.json which lists dependant packages

**package.json** - Specifics of npm's package.json handling. At the heart of node system all metadata for the project making it easier to package and redistribute.

You can create a package.json file **npm init**  
Answer the questions to complete

Short format is **npm init --yes**

```
D:\node
λ npm install lodash
```

```
D:\node
λ npm install lodash
npm WARN saveError ENOENT: no such file or directory, open 'D:\node\package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN enoent ENOENT: no such file or directory, open 'D:\node\package.json'
npm WARN node No description
npm WARN node No repository field.
npm WARN node No README data
npm WARN node No license field.

+ lodash@4.17.11
added 1 package from 2 contributors and audited 1 package in 2.128s
found 0 vulnerabilities

D:\node
λ
```

# Packages Express

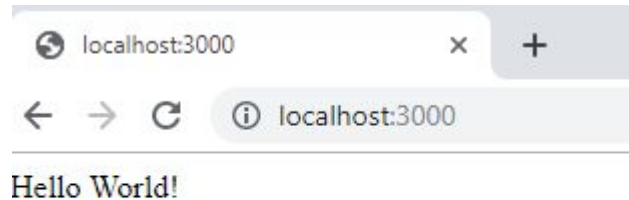
Express is the most popular Node web framework, and is the underlying library for a number of other popular Node web frameworks. Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

<https://expressjs.com/>

`npm install express --save`

<http://localhost:3000/>

```
npm install express --save
```



```
const express = require('express');
const app = express();
const port = 3000;
app.get('/', function (req, res) {
  res.send('Hello World!');
});
app.get('/about', function(req, res) {
  res.send('About Page');
});
app.listen(port, function () {
  return console.log("Port is ".concat(port, "!"));
});
```

# Express static files

<http://expressjs.com/en/guide/routing.html>

<https://www.npmjs.com/package/express>

<http://localhost:3000/2>

<http://localhost:3000>

```
const express = require('express');
const app = express();
const port = 3000;
app.use(express.static(__dirname)); // change directory to root

app.listen(port, function () {
  return console.log("Port is "+port + "' + __dirname);
});
```

```
const express = require('express');
const app = express();
const port = 3000;
app.get('/', function (req, res) {
  res.sendFile(__dirname + '/index1.html');
});
app.get('/2', function (req, res) {
  res.sendFile(__dirname + '/index2.html');
});
app.listen(port, function () {
  return console.log("Port is "+port + "' + __dirname);
});
```

# Congratulations on completing the section

Find out more about my courses at <http://www.discoveryvip.com/>

**Course instructor : Laurence Svekis -  
providing online training to over  
600,000 students across hundreds of  
courses and many platforms.**

