Path Module

```
JS logger.js   JS path_demo.js ×   JS os_demo.js   JS index.js
reference > JS path_demo.js > ...
1   // working with path module
2
3   const path = require('path');
4
5   // Base file name
6   console.log(path.basename(__filename));
7
8
9   // Directory name
10  console.log(path.dirname(__filename)); // or we can use __dirname instead of usingpath
11
12
13  // File Extension
14  console.log(path.extname(__filename));
15
16
17  // Crete path object
18  console.log(path.parse(__filename));
19
20  console.log(path.parse(__filename).base);
21
22
23  // Concatenate path
24  // ../ test/ hello.html
25  console.log(path.join(__dirname, 'test', 'hello.html'));
```

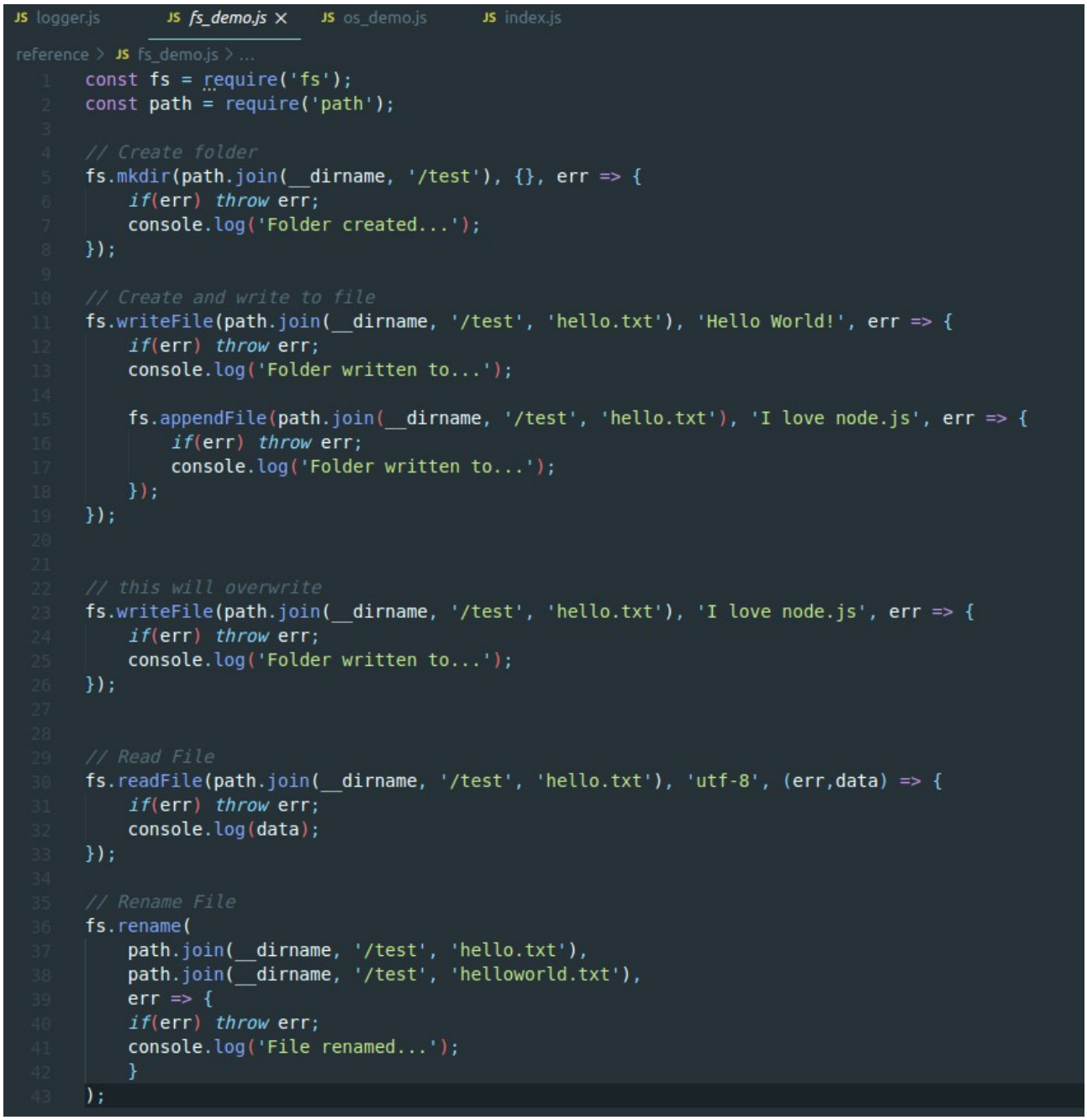
```
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC$ cd reference
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC/reference$ node path_demo
path_demo.js
/home/anuragjanghala/Desktop/nodejs_CC/reference
.js
{
  root: '/',
  dir: '/home/anuragjanghala/Desktop/nodejs_CC/reference',
  base: 'path_demo.js',
  ext: '.js',
  name: 'path_demo'
}
path_demo.js
/home/anuragjanghala/Desktop/nodejs_CC/reference/test/hello.html
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC/reference$
```

OS MODULE


logger.js
os_demo.js
url_demo.js
reference > os_demo.js > ...
const os = require('os');
// platform
console.log(os.platform());
// CPU Arch
console.log(os.arch())
// CPU core info
console.log(os.cpus());
// Free memory
console.log(os.freemem());
// total memory
console.log(os.totalmem());
// Home dir
console.log(os.homedir());
// Uptime
console.log(os.uptime());

```
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC/reference$ node os_demo
linux
x64
[
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3984,
    times: { user: 204810, nice: 2680, sys: 82750, idle: 5021030, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3980,
    times: { user: 217850, nice: 4730, sys: 81800, idle: 4951540, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3972,
    times: { user: 219940, nice: 2300, sys: 80610, idle: 4986700, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3966,
    times: { user: 195560, nice: 950, sys: 89970, idle: 4994820, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3985,
    times: { user: 210450, nice: 1740, sys: 81800, idle: 5009580, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3997,
    times: { user: 206800, nice: 3330, sys: 83430, idle: 5014340, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3997,
    times: { user: 210610, nice: 2780, sys: 90490, idle: 5001070, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3967,
    times: { user: 210520, nice: 2700, sys: 79800, idle: 5024160, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3946,
    times: { user: 210620, nice: 2660, sys: 80530, idle: 5014070, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3998,
    times: { user: 215970, nice: 1620, sys: 80220, idle: 5004810, irq: 0 }
  },
  {
    model: 'Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz',
    speed: 3993,
1317564416
8185262080
/home/anuragjanghala
5367
```

FS Module


logger.js
fs_demo.js
os_demo.js
index.js
reference > fs_demo.js > ...
const fs = require('fs');
const path = require('path');
// Create folder
fs.mkdir(path.join(__dirname, '/test'), {}, err => {
if(err) throw err;
console.log('Folder created...');
});
// Create and write to file
fs.writeFile(path.join(__dirname, '/test', 'hello.txt'), 'Hello World!', err => {
if(err) throw err;
console.log('Folder written to...');
fs.appendFile(path.join(__dirname, '/test', 'hello.txt'), 'I love node.js', err => {
if(err) throw err;
console.log('Folder written to...');
});
});
// this will overwrite
fs.writeFile(path.join(__dirname, '/test', 'hello.txt'), 'I love node.js', err => {
if(err) throw err;
console.log('Folder written to...');
});
// Read File
fs.readFile(path.join(__dirname, '/test', 'hello.txt'), 'utf-8', (err,data) => {
if(err) throw err;
console.log(data);
});
// Rename File
fs.rename(
path.join(__dirname, '/test', 'hello.txt'),
path.join(__dirname, '/test', 'helloworld.txt'),
err => {
if(err) throw err;
console.log('File renamed...');
}
);

Event Emitter using log

-> uuid was used to get the id of the message

logger.js

```
const EventEmitter = require('events');
const uuid = require('uuid');

class Logger extends EventEmitter{
    log(msg){
        // Call event
        this.emit('message', {id : uuid.v4(), msg});
    }
}

module.exports = Logger
```

index.js

```
const Logger = require('./logger');

const logger = new Logger();

logger.on('message', (data) => console.log('called listener',data));

logger.log('Hello world!');
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL                    3: bash

(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC$ cd reference
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC/reference$ cd ..
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC$ node index
called listener { id: 'de4293e9-0533-413e-9364-0fc4dead2e21', msg: 'Hello world!' }
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC$
```

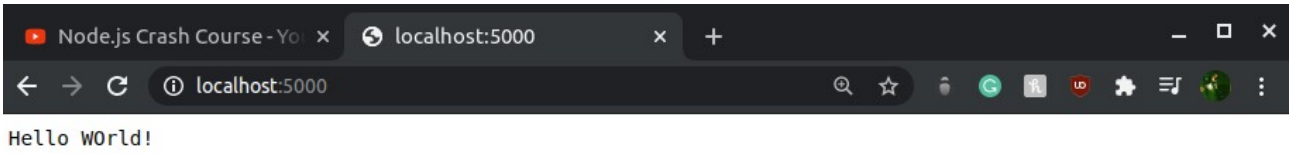
Classes using export modules:

HTTP Module Server

- Simplest web server -> hello world

```
JS http_demo.js ×
reference > JS http_demo.js > ...
1   const http = require('http');
2
3   // Create server object
4   http.createServer((req, res) => { //request and respone
5       // Write a response
6       res.write('Hello WOrld!');
7       res.end();
8   }).listen(5000, () => console.log('Server running...'));
```

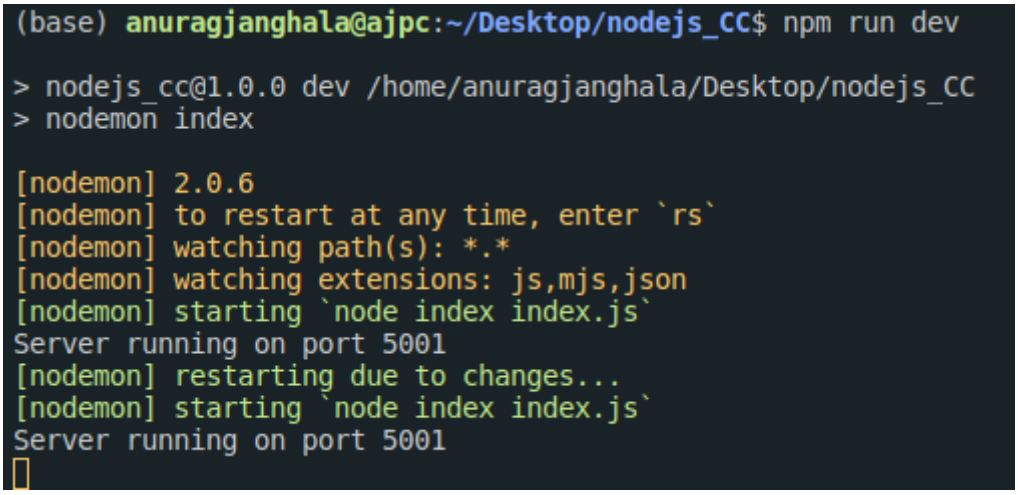
```
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC/reference$ node http_demo
Server running...
^C
(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC/reference$ cd ..
```



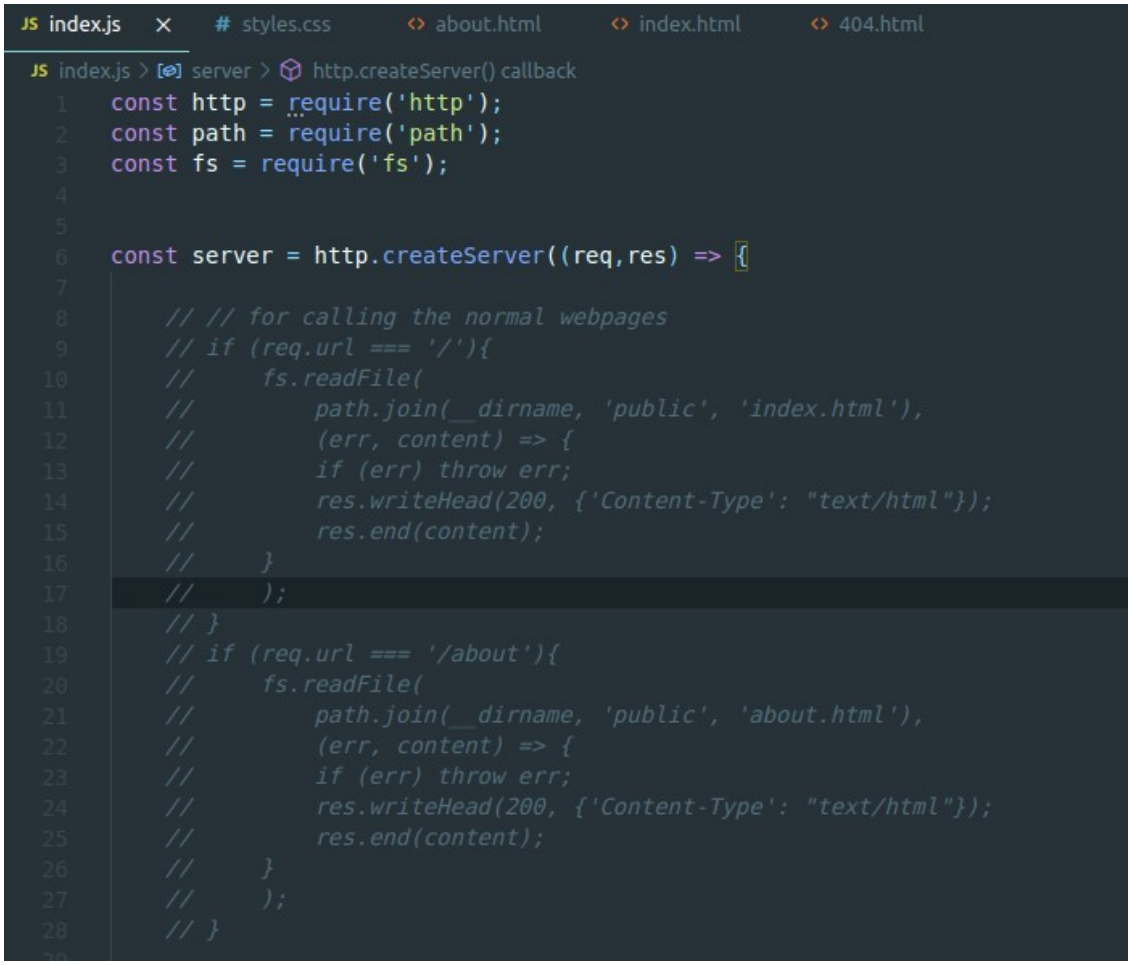

```
Hello WOrld!
```

Nodemon :

is used to watch over the running server so that we need not to update server on our own.


(base) anuragjanghala@ajpc:~/Desktop/nodejs_CC$ npm run dev
> nodejs_cc@1.0.0 dev /home/anuragjanghala/Desktop/nodejs_CC
> nodemon index
[nodemon] 2.0.6
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index index.js`
Server running on port 5001
[nodemon] restarting due to changes...
[nodemon] starting `node index index.js`
Server running on port 5001

Full Server code using nodeJS and Some html, css files used to deploy on heroku using git.




```
// // for calling the apis
// if (req.url === '/api/users'){
//     const users = [
//         { name: 'Bob Smith', age: 40 },
//         { name: 'John Doe', age: 30}
//     ];
//     res.writeHead(200, {'Content-Type': "application/json"});
//     res.end(JSON.stringify(users));
// }

let filePath = path.join(
    __dirname,
    'public',
    req.url === '/' ? 'index.html' : req.url
);

// extension of the file
let extname = path.extname(filePath);

// Initial content type
let contentType = 'text/html';
```

```
//check extension and set content type
switch(extname){
    case '.js':
        contentType = 'text/javascript';
        break;
    case '.css':
        contentType = 'text/css';
        break;
    case '.json':
        contentType = 'application/json';
        break;
    case '.png':
        contentType = 'image/png';
        break;
    case '.jpg':
        contentType = 'image/jpg';
        break;
}
```

```
    // Read file
    fs.readFile(filePath,
        (err, content) => {
            if(err){
                if(err.code == 'ENOENT'){
                    // Page not found
                    fs.readFile(path.join(__dirname,'public','404.html'), (err,content) =>{
                        if(err) throw err;
                        res.writeHead(200, {'Content-Type': "text/html"});
                        res.end(content, 'utf-8');
                    })
                }else {
                    // Some server error
                    res.writeHead(500);
                    res.end(`Server error: `+err.code);
                }
            }else {
                // Success
                res.writeHead(200, {'Content-Type': contentType});
                res.end(content, 'utf-8');
            }

        }
    );
});

const PORT = process.env.PORT || 5001;

server.listen(PORT, () => console.log(`Server running on port `+ PORT));
```

Html files:

public > index.html > html > head > link

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Homepage</title>
    <link rel="stylesheet" href="./css/styles.css" />
  </head>
  <body>
    <h1>Welcome to the Homepage!</h1>
  </body>
</html>
```

index.js styles.css about.html index.html 404.html

public > about.html > html > head > link

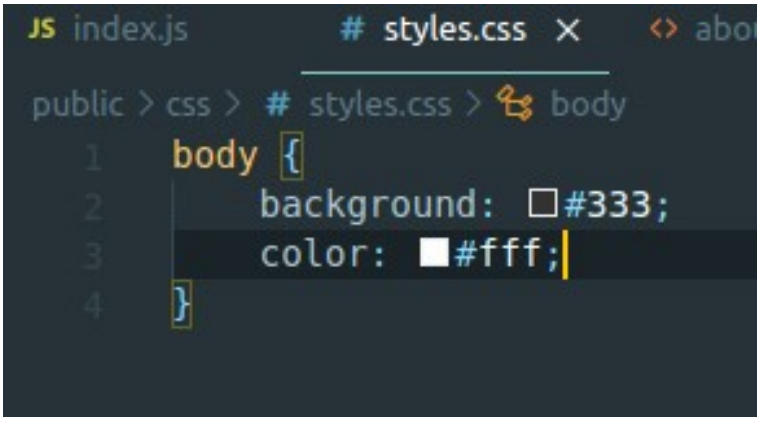
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>About</title>
    <link rel="stylesheet" href="./css/styles.css" />
  </head>
  <body>
    <h1>About</h1>
  </body>
</html>
```

index.js styles.css about.html index.html 404.html

public > 404.html > ...

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>error 404</title>
    <link rel="stylesheet" href="./css/styles.css" />
  </head>
  <body>
    <h1>404 Error</h1>
  </body>
</html>
```

CSS file:


index.js
styles.css
public > css > # styles.css > body
1 body {
2 background: #333;
3 color: #fff;
4 }