



FS Module

#Node JS Notes

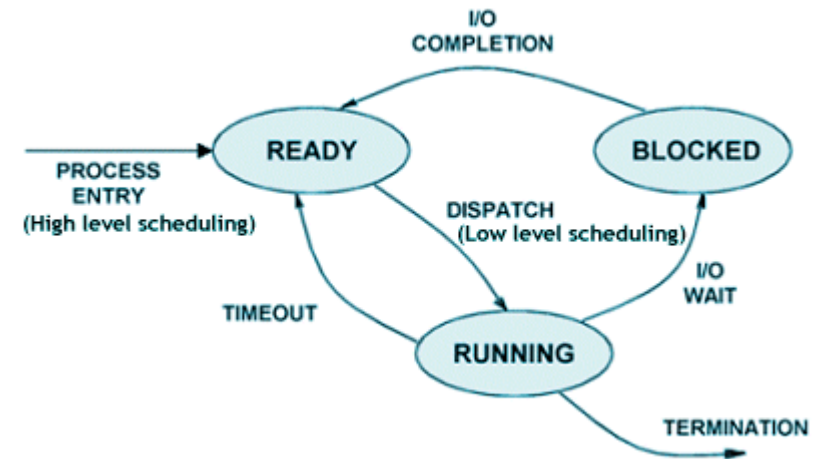
Blocking vs Non Blocking

■ Blocking

- When javascript execution in Node.js process (each program is a process) has to wait until a non-javascript operation completes is called blocking.

■ Non-Blocking

- This is the opposite of the blocking i.e. javascript execution do not wait until the non-javascript operation completes.

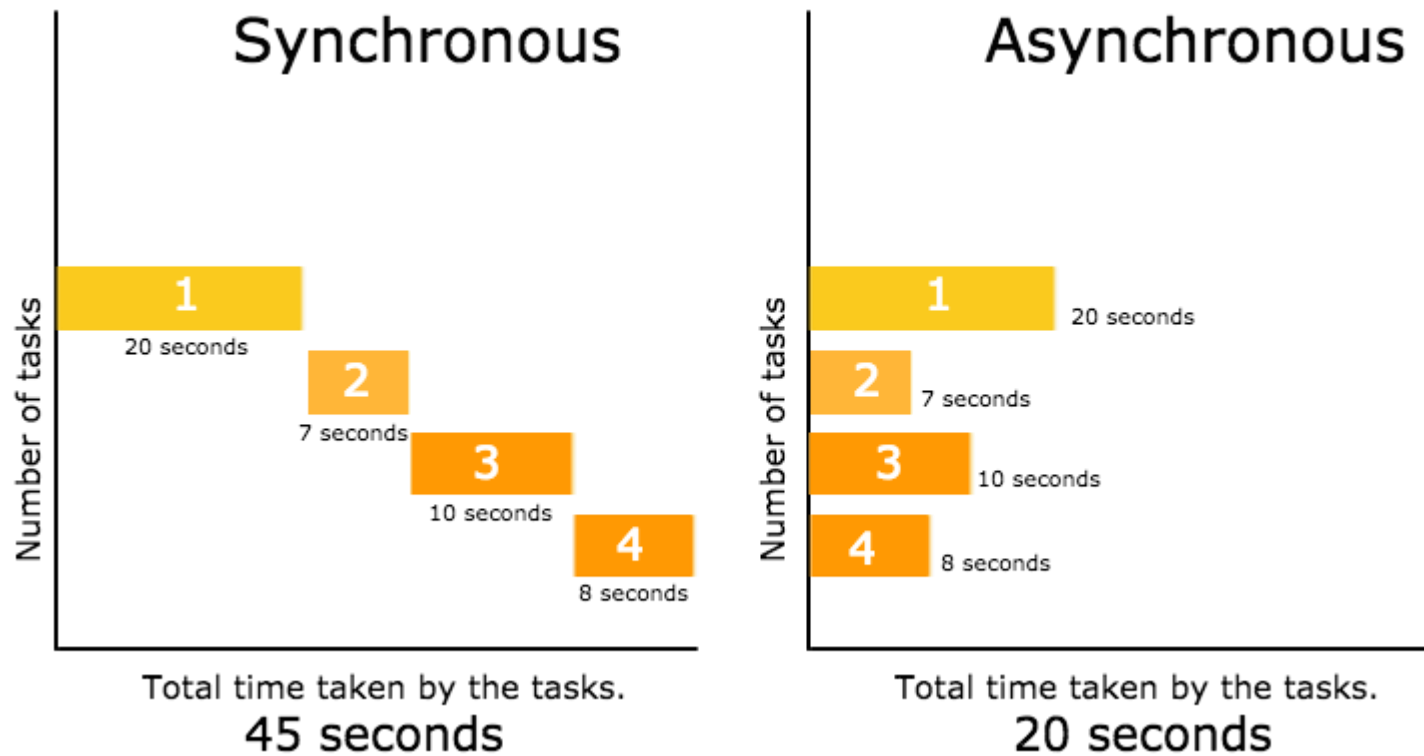


Synchronous vs. Asynchronous in Node.js

- Synchronous code is also called “blocking” because it halts the program until all the resources are available
- asynchronous code is also known as “non-blocking” because the program continues executing and doesn’t wait for external resources (I/O) to be available.



- Sync = Synchronous = Blocking I/O model
- Async = Asynchronous = Non-blocking I/O model

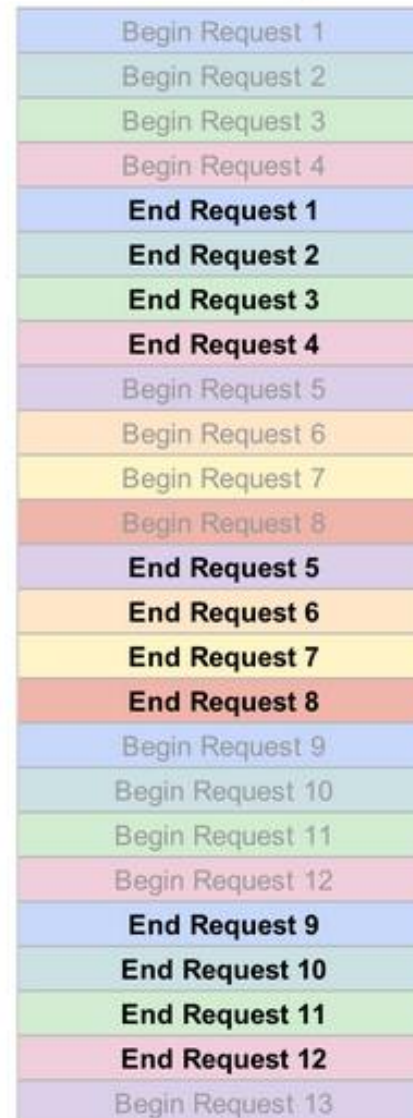


milliseconds

Blocking API



Non-Blocking API



Advantages of non-blocking code

- Non-blocking code is much more performant.
- Blocking code waste around 90% of CPU cycles waiting for the network or disk to get the data.
- Using non-blocking code is a more straightforward way to have concurrency without having to deal with multiple execution threads.



FS Module



Fs module

- Fs means File System Module

- Which help

- Create file
 - Read File
 - Delete File
 - Append File

- Example :

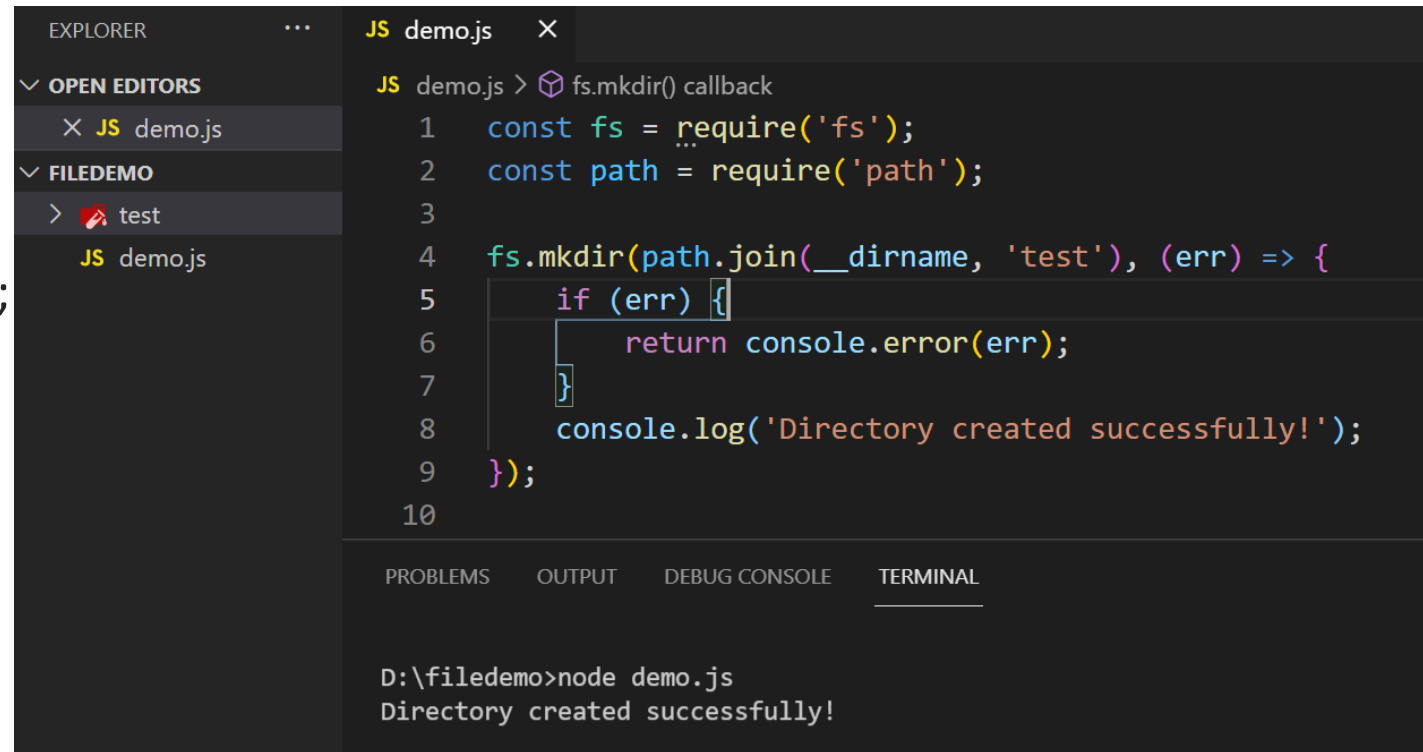
```
var fs = require('fs');
```

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



Make Directory

```
const fs = require('fs');  
const path = require('path');  
fs.mkdir(path.join(__dirname, 'test'), (err) => {  
  if (err) {  
    return console.error(err);  
  }  
  console.log('Directory created successfully!');  
});
```



The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane shows a file named 'test' inside a folder named 'FILEDEMO'. The main editor shows the 'demo.js' file with the following code:

```
1 const fs = require('fs');  
2 const path = require('path');  
3  
4 fs.mkdir(path.join(__dirname, 'test'), (err) => {  
5   if (err) {  
6     return console.error(err);  
7   }  
8   console.log('Directory created successfully!');  
9 });  
10
```

Below the code editor, the TERMINAL pane shows the command 'D:\filedemo>node demo.js' and the output 'Directory created successfully!'.



File Open

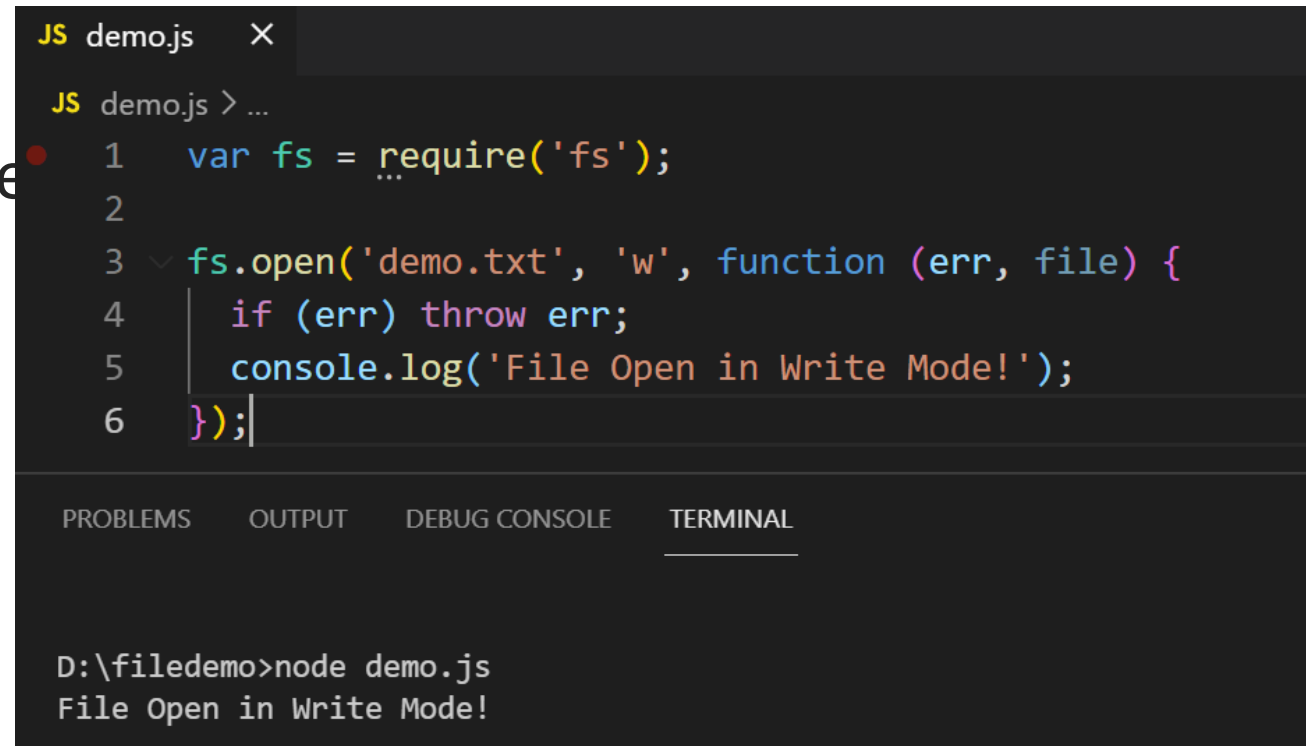
```
var fs = require('fs');
```

```
fs.open('demo.txt', 'w', function (err, file) {
```

```
  if (err) throw err;
```

```
  console.log('File Open in Write Mode!');
```

```
});
```



```
JS demo.js X
JS demo.js > ...
1  var fs = require('fs');
2
3  fs.open('demo.txt', 'w', function (err, file) {
4    if (err) throw err;
5    console.log('File Open in Write Mode!');
6  });
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
D:\filedemo>node demo.js
File Open in Write Mode!
```



Create File

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

```
fs.writeFile('test.txt', 'Hello Word!', function (err) {
```

```
  if (err) throw err;
```

```
  console.log('File Created!');
```

```
});
```



The screenshot shows a Visual Studio Code editor window with a file named 'demo.js' open. The code in the editor is as follows:

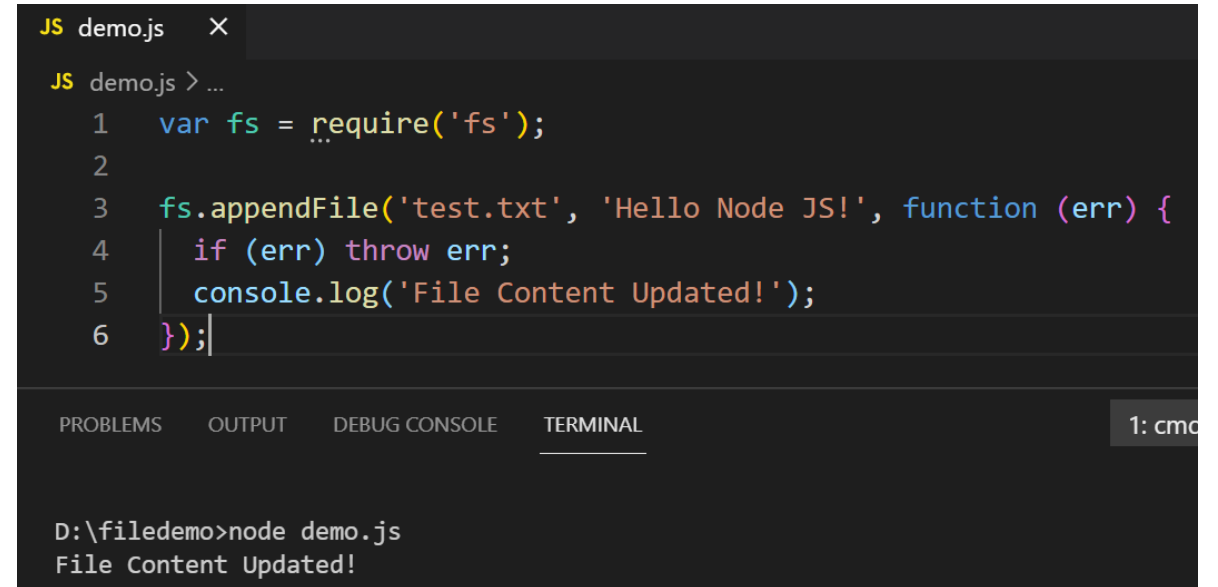
```
1 var fs = require('fs');
2
3 fs.writeFile('test.txt', 'Hello Word!', function (err) {
4   if (err) throw err;
5   console.log('File Created!');
6 });
```

The Explorer sidebar on the left shows the file structure with 'demo.js' and 'test.txt' listed under a 'FILEDEMO' folder. The terminal at the bottom shows the command 'D:\filedemo>node demo.js' and the output 'File Created!'.



File Content Append

```
var fs = require('fs');  
  
fs.appendFile('test.txt', 'Hello Node JS!', function (err) {  
  if (err) throw err;  
  console.log('File Content Updated!');  
});
```



The screenshot shows a code editor window titled 'demo.js' with the following JavaScript code:

```
1 var fs = require('fs');  
2  
3 fs.appendFile('test.txt', 'Hello Node JS!', function (err) {  
4   if (err) throw err;  
5   console.log('File Content Updated!');  
6 });
```

Below the code editor is a terminal window with the following output:

```
D:\filedemo>node demo.js  
File Content Updated!
```



File Rename

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

```
fs.rename('demo.txt', 'demo1.txt', function (err) {
```

```
  if (err) throw err;
```

```
  console.log('File Renamed!');
```

```
});
```



The screenshot shows a code editor with a file named 'demo.js'. The code in the editor is as follows:

```
JS demo.js > ...  
1  var fs = require('fs');  
2  
3  fs.rename('demo.txt', 'demo1.txt', function (err) {  
4    if (err) throw err;  
5    console.log('File Renamed!');  
6  });
```

Below the code editor, there is a terminal window with the following output:

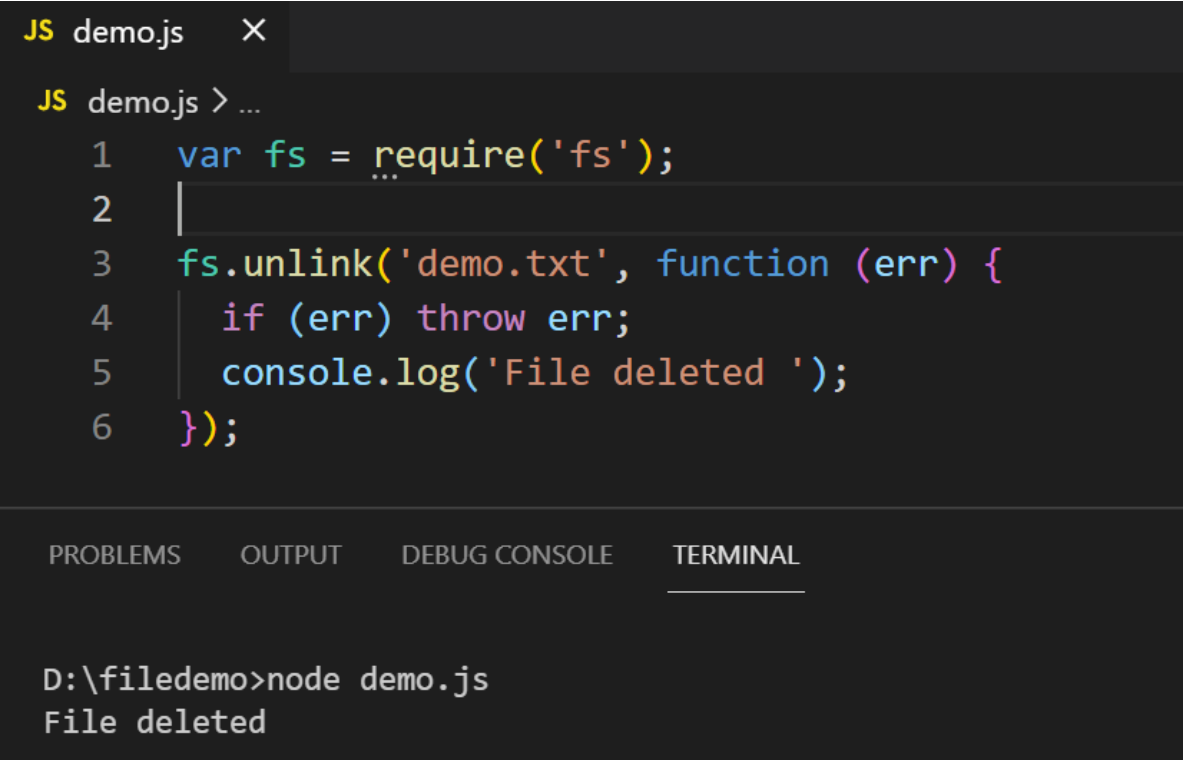
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  
  
D:\filedemo>node demo.js  
File Renamed!
```



Delete File Code

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

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



```
JS demo.js X
JS demo.js > ...
1  var fs = require('fs');
2
3  fs.unlink('demo.txt', function (err) {
4    if (err) throw err;
5    console.log('File deleted ');
6  });
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
D:\filedemo>node demo.js
File deleted
```



Read File (Asynchronous)

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

```
fs.readFile("demo.txt","utf-8", function (err, data) {
```

```
if (err) throw err;
```

```
console.log(data);
```

```
});
```

```
JS demo.js ×
JS demo.js > ...
1  var fs = require('fs');
2
3  fs.readFile("demo.txt", function (err, data) {
4    if (err) throw err;
5    console.log(data);
6  });|

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

D:\filedemo>node demo.js
<Buffer 41 6b 61 73 68>

D:\filedemo>|
```

```
JS demo.js ×
JS demo.js > ...
1  var fs = require('fs');
2
3  fs.readFile("demo.txt","utf-8", function (err, data) {
4    if (err) throw err;
5    console.log(data);
6  });|

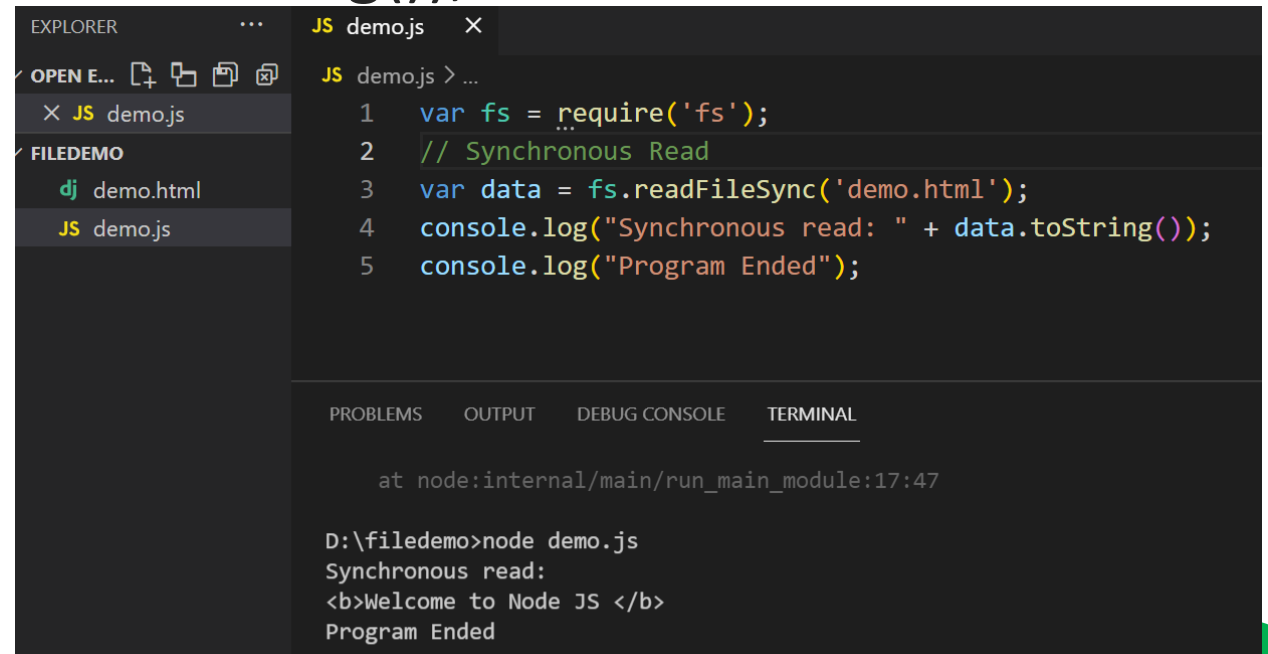
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

D:\filedemo>node demo.js
Akash
```



Synchronous File Read

```
var fs = require('fs');  
// Synchronous Read  
var data = fs.readFileSync('demo.html');  
console.log("Synchronous read: " + data.toString());  
console.log("Program Ended");
```



The screenshot shows a Visual Studio Code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'FILEDEMO' with files 'demo.html' and 'demo.js'. The code editor shows the following JavaScript code in 'demo.js':

```
1 var fs = require('fs');  
2 // Synchronous Read  
3 var data = fs.readFileSync('demo.html');  
4 console.log("Synchronous read: " + data.toString());  
5 console.log("Program Ended");
```

Below the code editor is a terminal window showing the output of running the program:

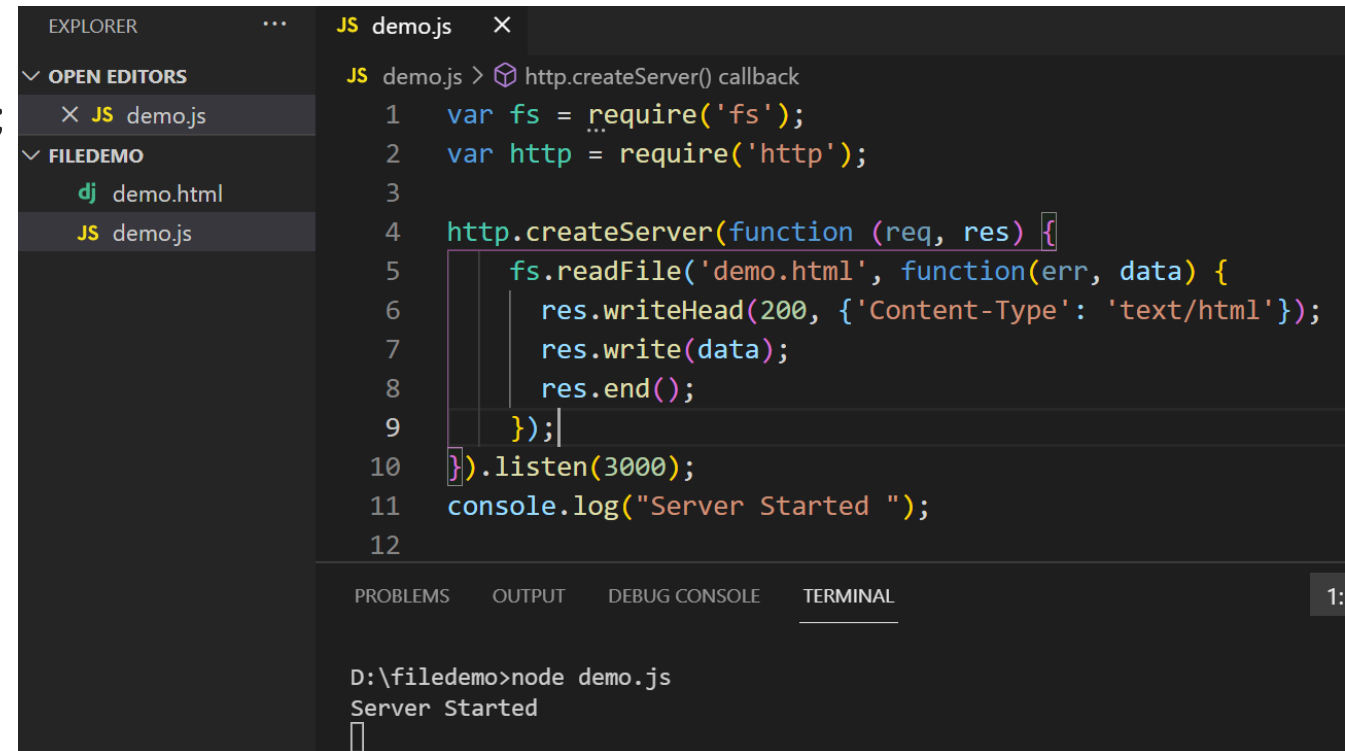
```
at node:internal/main/run_main_module:17:47  
  
D:\filedemo>node demo.js  
Synchronous read:  
<b>Welcome to Node JS </b>  
Program Ended
```



Read HTML File

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

http.createServer(function (req, res) {
  fs.readFile('demo.html', function(err, data) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    res.end();
  });
}).listen(3000);
console.log("Server Started ");
```



The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane shows a project named 'FILEDEMO' with two files: 'demo.html' and 'demo.js'. The main editor area displays the 'demo.js' file with the following code:

```
JS demo.js > http.createServer() callback
1  var fs = require('fs');
2  var http = require('http');
3
4  http.createServer(function (req, res) {
5    fs.readFile('demo.html', function(err, data) {
6      res.writeHead(200, {'Content-Type': 'text/html'});
7      res.write(data);
8      res.end();
9    });
10 }).listen(3000);
11 console.log("Server Started ");
12
```

At the bottom, the TERMINAL pane shows the command 'D:\filedemo>node demo.js' and the output 'Server Started'.



Get File Information

```
EXPLORER  ...  JS demo.js  X
  v OPEN EDITORS
    X JS demo.js
  v FILEDEMO
    JS demo.js
    dj demo.txt

JS demo.js > ...
1  var fs = require('fs');
2  //Get File Information using stats
3  fs.stat('demo.txt', function (err, stats) {
4      if (err) {
5          return console.error(err);
6      }
7      //Check File Information
8      console.log(stats);
9      // Check file type
10     console.log("isFile ? " + stats.isFile());
11     console.log("isDirectory ? " + stats.isDirectory());
12 });

D:\filedemo>node demo.js
Stats {
  dev: 855470817,
  mode: 33206,
  nlink: 1,
  uid: 0,
  gid: 0,
  rdev: 0,
  blksize: 4096,
  ino: 4222124650669704,
  size: 21,
  blocks: 0,
  atimeMs: 1620913090864.303,
  mtimeMs: 1620913090864.303,
  ctimeMs: 1620913090864.303,
  birthtimeMs: 1620830153088.68,
  atime: 2021-05-13T13:38:10.864Z,
  mtime: 2021-05-13T13:38:10.864Z,
  ctime: 2021-05-13T13:38:10.864Z,
  birthtime: 2021-05-12T14:35:53.089Z
}
isFile ? true
isDirectory ? false
```



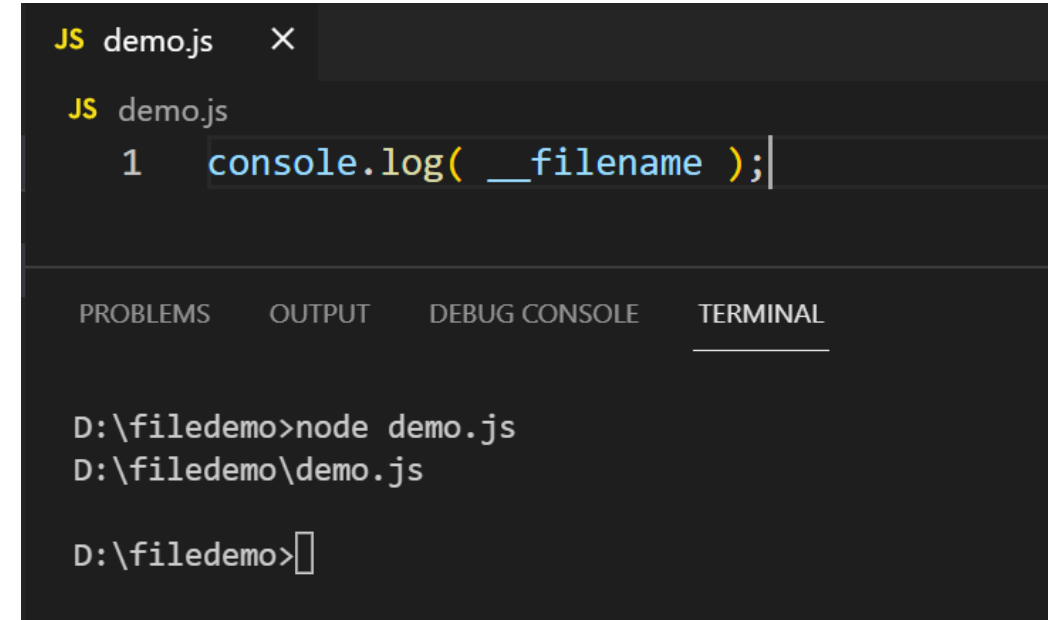
File Information

```
var fs = require('fs');  
//Get File Information using stats  
fs.stat('demo.txt', function (err, stats) {  
  if (err) {  
    return console.error(err);  
  }  
  //Check File Information  
  console.log(stats);  
  // Check file type  
  console.log("isFile ? " + stats.isFile());  
  console.log("isDirectory ? " + stats.isDirectory());  
});
```



Print FileName

- `console.log(__filename);`



```
JS demo.js X
JS demo.js
1 console.log( __filename );

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

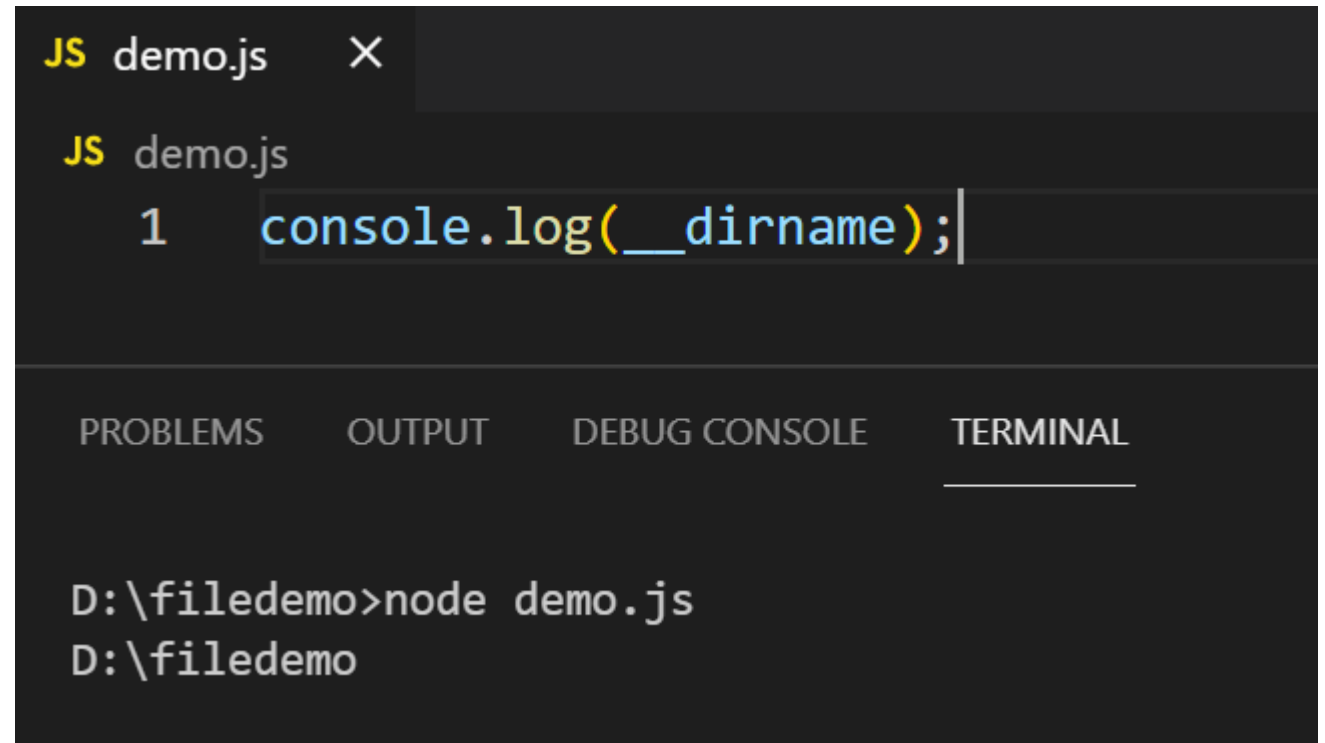
D:\filedemo>node demo.js
D:\filedemo\demo.js

D:\filedemo>
```



__dirname

- `console.log(__dirname);`

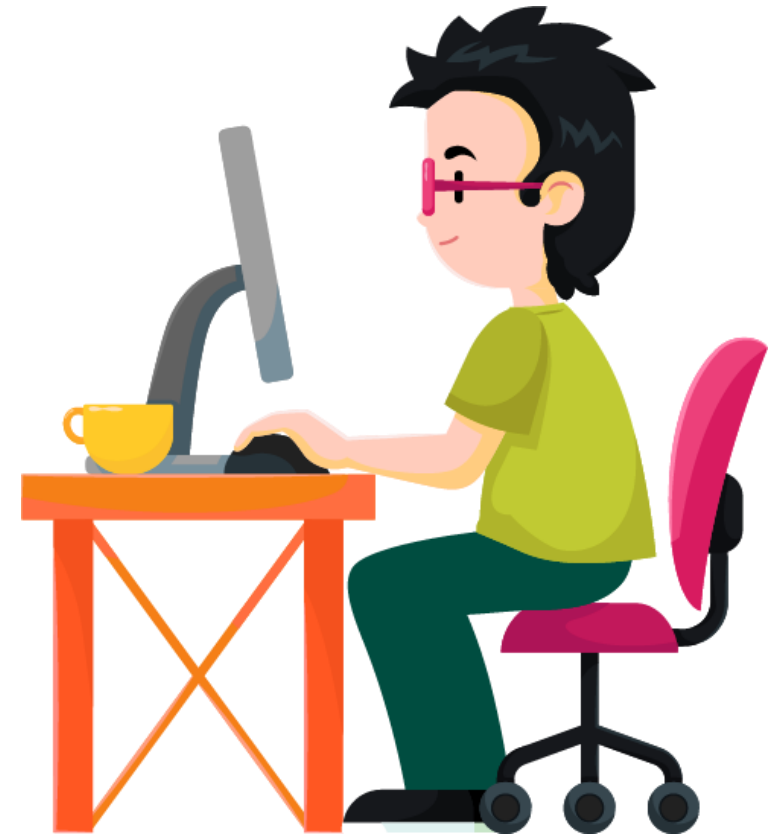


```
JS demo.js X
JS demo.js
1 console.log(__dirname);

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
D:\filedemo>node demo.js
D:\filedemo
```



Get Exclusive Video Tutorials



www.apptutorials.com

<https://www.youtube.com/user/Akashtips>





Get More Details

www.akashsir.com



If You Liked It !

Rating Us Now



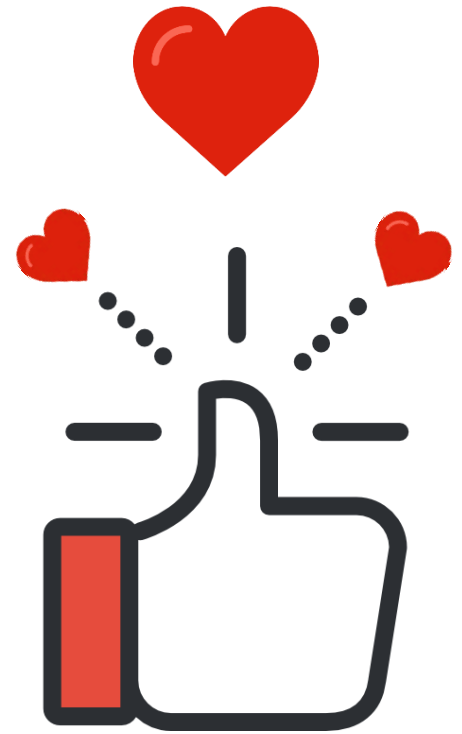
Just Dial

https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET



Sulekha

<https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad>



Connect With Me



Akash Padhiyar
#AkashSir

www.akashsir.com
www.akashtechlabs.com
www.akashpadhiyar.com
www.apptutorials.com

Social Info



Akash.padhiyar



Akashpadhiyar



Akash_padhiyar



+91 99786-21654



#Akashpadhiyar
#apptutorials