

NODE JS (HANDS ON)

Server – Side Java Script Framework

V D S Krishna, Sr. Asst. Professor, CSE Dept., CVRCE

5/19/2022



GLOBAL OBJECT

The screenshot shows a web browser window displaying the Node.js documentation for Global Objects. The browser's address bar shows the URL `https://nodejs.org/dist/latest-v10.x/docs/api/globals.html`. The page has a dark sidebar on the left with the Node.js logo and a list of navigation links. The main content area is titled "Table of Contents" and lists various global objects and methods. The Windows taskbar is visible at the bottom, showing the search bar and several application icons.

Node.js

- About these Docs
- Usage & Example
- Assertion Testing
- Async Hooks
- Buffer
- C++ Addons
- C/C++ Addons - N-API
- Child Processes
- Cluster
- Command Line Options
- Console
- Crypto
- Debugger
- Deprecated APIs
- DNS

Table of Contents

- Global Objects
 - Class: Buffer
 - `__dirname`
 - `__filename`
 - `clearImmediate(immediateObject)`
 - `clearInterval(intervalObject)`
 - `clearTimeout(timeoutObject)`
 - `console`
 - `exports`
 - `global`
 - `module`
 - `process`
 - `require()`
 - `setImmediate(callback[, ...args])`
 - `setInterval(callback, delay[, ...args])`
 - `setTimeout(callback, delay[, ...args])`
 - URL
 - URLSearchParams

RECEIPT-LM-652511.pdf RECEIPT-LM-652509.pdf Show all

Type here to search

ENG IN 19:33 12-11-2018

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> node app
called after 2 seconds
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> █
```

JS app.js x

```
1 setTimeout(function () {
2     console.log("called after 2 seconds");
3 }, 2000);
```

JS app.js x

```
1 setInterval(function () {
2     console.log("called after 2 seconds");
3 }, 2000);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> node app.js
called after 2 seconds
called after 2 seconds
called after 2 seconds
called after 2 seconds
called after 2 seconds
█
```

JS app.js x

```
1 var timer = 0;
2
3 var interval = setInterval(function () {
4     timer += 2;
5     console.log("called after " + timer + " seconds");
6     if (timer > 10)
7         clearInterval(interval);
8 }, 2000);|
```

PROBLEMS OUTPUT DEBUG CONSOLE

```
PS E:\Softwares\Visual Studio Code (FEE)\Trai
called after 2 seconds
called after 4 seconds
called after 6 seconds
called after 8 seconds
called after 10 seconds
called after 12 seconds
PS E:\Softwares\Visual Studio Code (FEE)\Trai
```

```
JS app.js x
1 console.log(__dirname);
2
3 console.log(__filename);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> node app.js
E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node
E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node\app.js
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> █
```


MODULE

```
JS app.js x
1 // Normal function statement
2 function sayHello() {
3     console.log("hi");
4 }
5 sayHello();
6
7 //function expression
8 var sayBye = function () {
9     console.log("Bye");
10 }
11 sayBye();
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesgi
Function Called. . . . .
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesgi
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesgi
hi
Bye
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesgi
```

```
JS app.js x
1 function callFunction(callback) {
2     callback();
3 }
4 //function expression
5 var sayBye = function () {
6     console.log("Function Called. . . . .");
7 }
8 callFunction(sayBye);
```

Module & require()

```
JS app.js  JS count.js x
1  var counter = function (arr) {
2      return "The length of the arr is  : " + arr.length;
3  }
4  console.log(counter([1, 2, 3, 4]));
```

```
JS app.js  x  JS count.js
1  var counter = require('./count');
2
3  console.log(counter([1, 2, 3, 4]));
```

```
JS app.js  JS count.js x
1  var counter = function (arr) {
2      return "The length of the arr is  : " + arr.length;
3  }
4
5  module.exports = counter;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Pr
The length of the arr is  : 4
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Pr
```

Module Patterns

```
JS app.js  JS all.js  x
1  var counter = function (arr) {
2      return "The length of the arr is  : " + arr.length;
3  }
4  module.exports.anotherCounter = function (msg) {
5      return "You called me  : " + msg;
6  }
7
8  var adder = function (a, b) {
9      return `The addition of ${a} and ${b} is  : ${a + b}`;
10 }
11 module.exports.counter = counter;
12 module.exports.adder = adder;
```

```
JS app.js  x  JS all.js
1  var all = require('./all');
2
3  console.log(all.counter([1, 2, 3, 4]));
4  console.log(all.adder(20, 30));
5  console.log(all.anotherCounter("Afroz"));
```

Reading and Writing Files

```
JS app.js x <> index.html ≡ readMe.txt
1 var fs = require('fs');
2
3 var dataRead = fs.readFileSync("readMe.txt", "utf8");
4
5 console.log(dataRead);
```

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice M
Lorem ipsum dolor sit, amet consectetur adipisicing elit.
Culpa eligendi assumenda voluptatibus expedita laudantium vero eaque eum
veritatis, error maiores earum repudiandae tempore similique,
quam porro consequatur quae soluta? Tenetur.
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice M
```

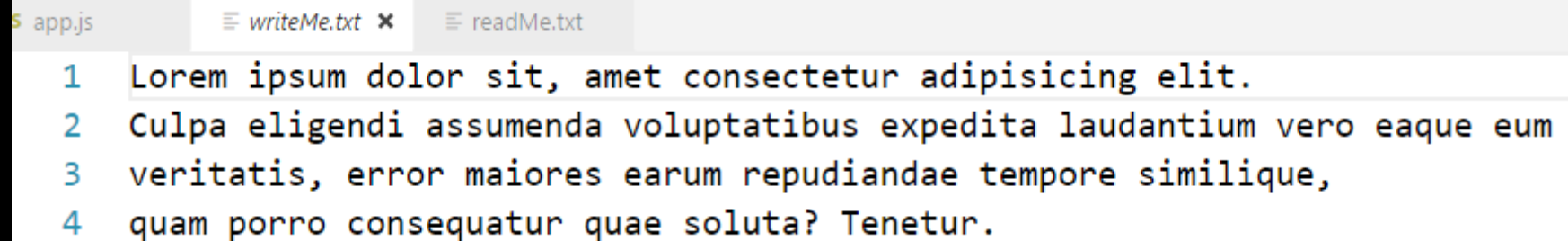
```
1 var fs = require('fs');
2
3 var dataRead = fs.readFileSync("readMe.txt", "utf8");
4 fs.writeFileSync("writeMe.txt", dataRead);
5
```

```
JS app.js ≡ writeMe.txt x ≡ readMe.txt
1 Lorem ipsum dolor sit, amet consectetur adipisicing elit.
2 Culpa eligendi assumenda voluptatibus expedita laudantium vero eaque eum
3 veritatis, error maiores earum repudiandae tempore similique,
4 quam porro consequatur quae soluta? Tenetur.
```


Reading and Writing Files cond..

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

fs.readFile("readMe.txt", 'utf8', function (err, dataRead) {
  if (err)
    console.log("Error in Reading the file content...");
  else
    fs.writeFile("writeMe.txt", dataRead, function (err) {
      if (err)
        console.log("Error in Writing the Data to the file . .");
    });
});
```

A screenshot of a code editor interface. At the top, there are three tabs: 'app.js', 'writeMe.txt', and 'readMe.txt'. The 'writeMe.txt' tab is currently selected and active. Below the tabs, the content of 'writeMe.txt' is displayed, showing four lines of Lorem Ipsum text. The text is as follows:

1 Lorem ipsum dolor sit, amet consectetur adipisicing elit.
2 Culpa eligendi assumenda voluptatibus expedita laudantium vero eaque eum
3 veritatis, error maiores earum repudiandae tempore similique,
4 quam porro consequatur quae soluta? Tenetur.

Directories and Files

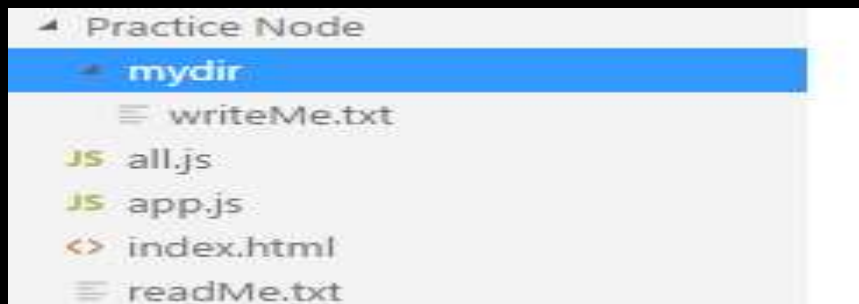
```
var fs = require('fs');

fs.mkdir('mydir', function () {
  fs.readFile('readMe.txt', 'utf8', function (err, data) {
    fs.writeFile('./mydir/writeMe.txt', data, function (err) {
      if (err)
        console.log("Error in Writing. . . ");
    });
  });
});
```

```
1 var fs = require('fs');
2
3 fs.unlink('writeMe.txt');
```

```
1 var fs = require('fs');
2 fs.mkdirSync('mydir');
```

```
1 var fs = require('fs');
2 fs.rmdirSync('mydir');
```



```
1 var fs = require('fs');
2
3 fs.unlink('./mydir/writeMe.txt', function () {
4   fs.rmdir('mydir');
5 });
```

Event Module

```
JS app.js x JS all.js
1 var events = require('events');
2
3 var myEmitter = new events.EventEmitter();
4
5 myEmitter.on("someEvent", function (message) {
6     console.log(message);
7 });
8
9 myEmitter.emit("someEvent", "Event Raised");
10
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> node app.js
Event Raised
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\Practice Node> 
```

Event Module (contd..)

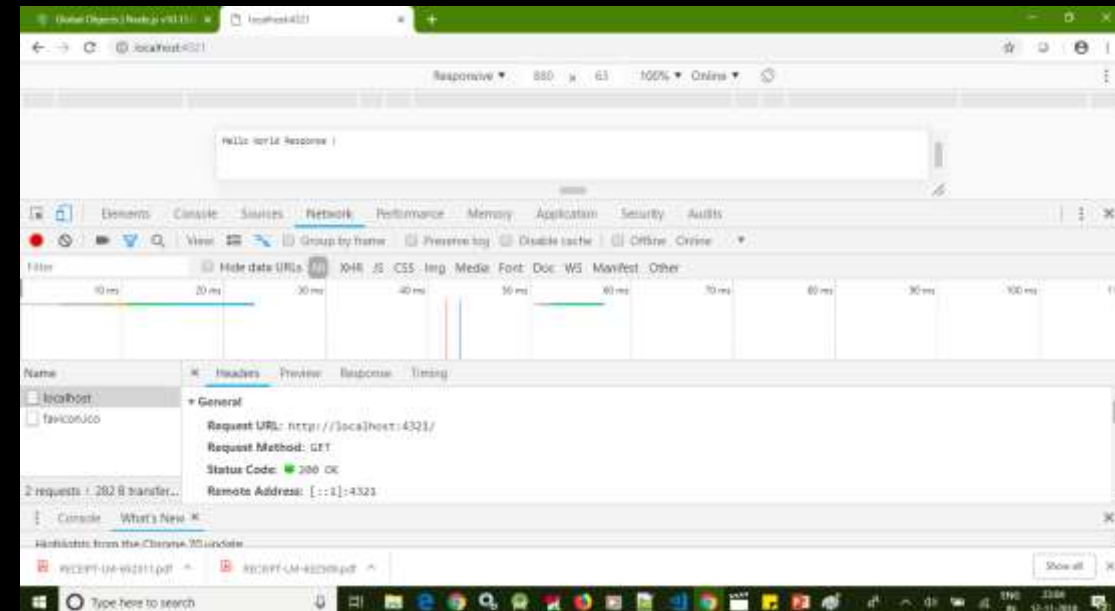
```
JS app.js x JS all.js
1  var events = require('events');
2  var util = require('util');
3  var Person = function (name) {
4    |   this.name = name;
5  };
6  util.inherits(Person, events.EventEmitter);
7  var afroz = new Person("Afroz");
8  var suhail = new Person("Suhail");
9  var people = [afroz, suhail];
10 people.forEach(function (person) {
11   |   person.on("speaking", function () {
12   |       |   console.log(person.name + " is speaking");
13   |   });
14 });
15 afroz.emit("speaking");
```

```
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\
Afroz is speaking
PS E:\Softwares\Visual Studio Code (FEE)\Training\Angular Tesging\
```


Creating a Server (http module)

```
1 var http = require('http');
2 var server = http.createServer(function (req, res) {
3     res.write("Hello World Response !");
4     res.end();
5 });
6 server.listen(4321);
7 console.log("Server listening at port number 4321");
```

```
1 var http = require('http');
2 var server = http.createServer(function (req, res) {
3     console.log(req.url);
4     res.writeHead(200, { 'content-type': 'plaintext' });
5     res.write("Hello World Response !");
6     res.end();
7 });
8 server.listen(4321);
9 console.log("Server listening at port number 4321");
```



Streams and Buffers

Buffers

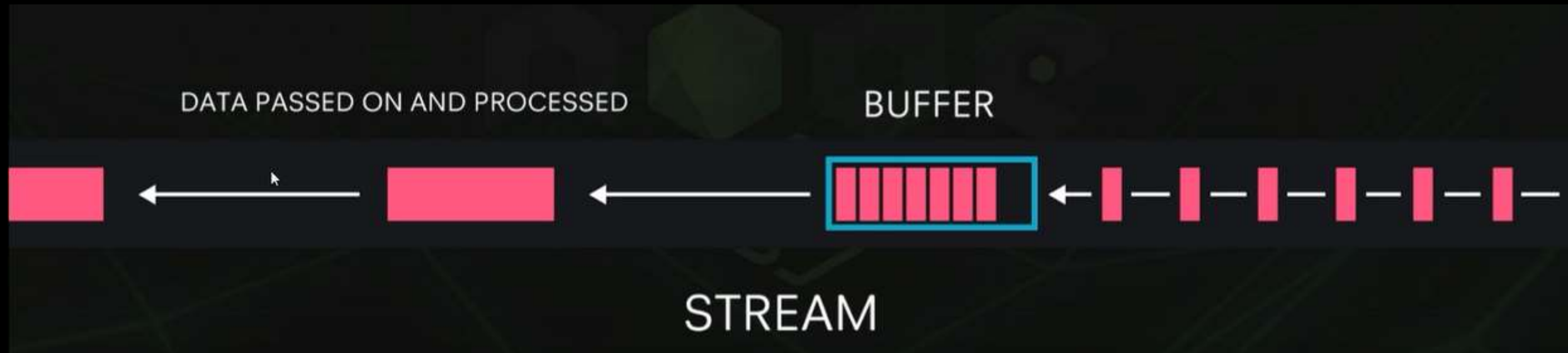
Temporary storage spot for a chunk of data that is being transferred from one place to another

Buffer is filled with data and is passed along

Transfers small chunks of data at a time



Streams



Streams in Node.js

- Writable Streams – allow node js to write data to a stream
- Readable Streams – allow node js to read data from a stream
- Duplex – can read and write to a stream
 - Can create streams in Node.js to transfer data
 - Increase Performance

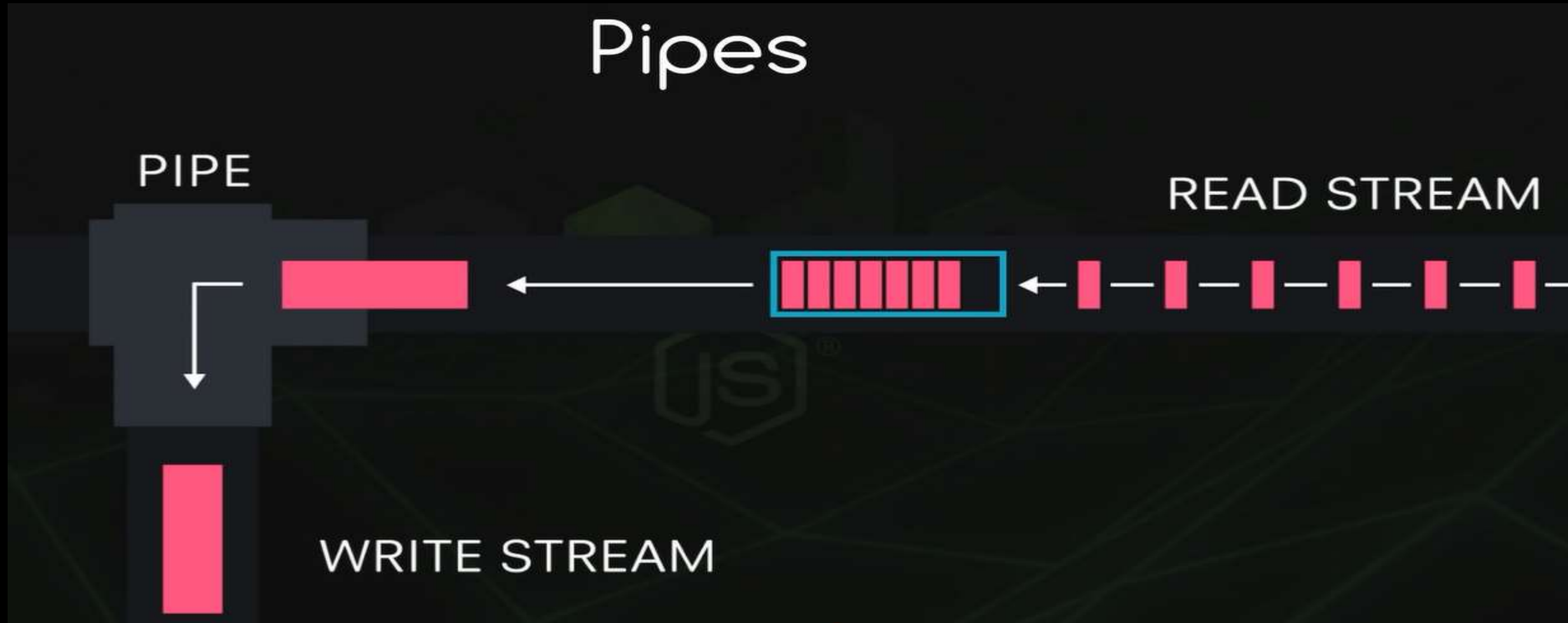
Readable Streams

```
var http = require('http');  
var fs = require('fs');  
  
var myReadStream = fs.createReadStream(__dirname + '/readMe.txt', 'utf8');  
myReadStream.on('data', function (chunk) {  
    console.log('New Chunk Received');  
    console.log(chunk);  
});
```

Writable Streams

```
1  var http = require('http');
2  var fs = require('fs');
3
4  var myReadStream = fs.createReadStream(__dirname + '/readMe.txt', 'utf8');
5  var myWriteStream = fs.createWriteStream(__dirname + '/writeMe.txt');
6  myReadStream.on('data', function (chunk) {
7      console.log('New Chunk Received');
8      myWriteStream.write(chunk);
9  });
```


Pipes



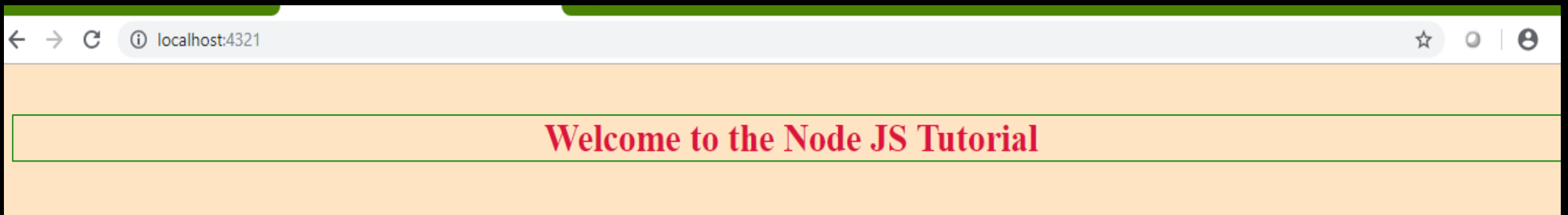
```
1  var http = require('http');
2  var fs = require('fs');
3
4  var myReadStream = fs.createReadStream(__dirname + '/readMe.txt', 'utf8');
5  var myWriteStream = fs.createWriteStream(__dirname + '/writeMe.txt');
6  myReadStream.pipe(myWriteStream);
7
8
```

```
1  var http = require('http');
2  var fs = require('fs');
3
4  var http = require('http');
5  var server = http.createServer(function (req, res) {
6    console.log(req.url);
7    res.writeHead(200, { 'content-type': 'plaintext' });
8    var myReadStream = fs.createReadStream(__dirname + '/readMe.txt', 'utf8');
9    //var myWriteStream = fs.createWriteStream(__dirname + '/writeMe.txt');
10   myReadStream.pipe(res);
11 });
12 server.listen(4321);
13 console.log("Server listening at port number 4321");
```


Serving HTML pages to client

```
12
13     h1 {
14         text-align: center;
15         font-size: 30px;
16         border: 1px solid  green;
17     }
18 </style>
19 </head>
20
21 <body>
22     <h1>Welcome to the Node JS Tutorial</h1>
23 </body>
24
25 </html>
```

```
1  var http = require('http');
2  var fs = require('fs');
3
4  var http = require('http');
5  var server = http.createServer(function (req, res) {
6      console.log(req.url);
7      res.writeHead(200, { 'content-type': 'text/html' });
8      var myReadStream = fs.createReadStream(__dirname + '/index.html', 'utf8');
9      //var myWriteStream = fs.createWriteStream(__dirname + '/writeMe.txt');
10     myReadStream.pipe(res);
11 });
12 server.listen(4321);
13 console.log("Server listening at port number 4321");
```



Serving JSON to client

```
1  var http = require('http');
2  var fs = require('fs');
3
4  var http = require('http');
5  var server = http.createServer(function (req, res) {
6      console.log(req.url);
7      res.writeHead(200, { 'content-type': 'application/json' });
8      var myObj = {
9          name: 'Suhail Afroz',
10         designation: 'Associate Professor',
11         department: 'Computer Science and Engineering',
12         place: 'CVR College of Engineering'
13     };
14
15     res.write(JSON.stringify(myObj));
16     res.end();
17 });
18 server.listen(4321);
19 console.log("Server listening at port number 4321");
```

← → ↻ ⓘ localhost:4321



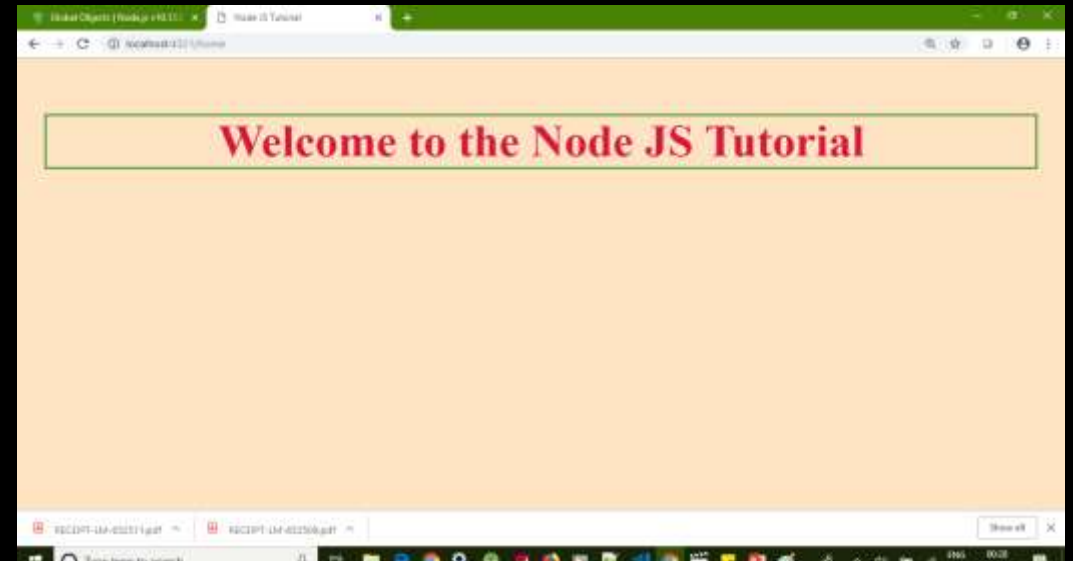
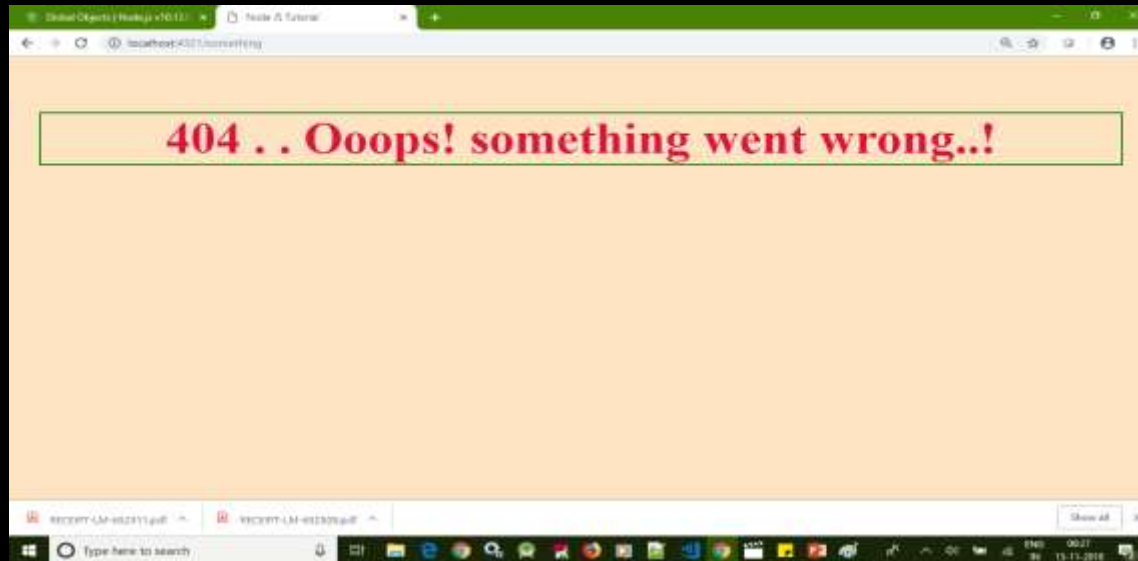
```
{"name":"Suhail Afroz","designation":"Associate Professor","department":"Computer Science and Engineering","place":"CVR College of Engineering"}
```


Basic Routing

```
1 var http = require('http');
2 var fs = require('fs');
3 var http = require('http');
4 var server = http.createServer(function (req, res) {
5     console.log(req.url);
6     if (req.url === '/home') {
7         res.writeHead(200, { 'content-type': 'text/html' });
8         fs.createReadStream(__dirname + '/index.html').pipe(res);
9     } else if (req.url === '/data') {
10        res.writeHead(200, { 'content-type': 'application/json' });
11        var myObj = {
12            name: 'Suhail Afroz',
13            designation: 'Associate Professor',
14            department: 'Computer Science and Engineering',
15            place: 'CVR College of Engineering'
16        };
17        res.write(JSON.stringify(myObj));
18        res.end();
19    } else {
20        res.writeHead(404, { 'content-type': 'text/html' });
```

Basic Routing contd..

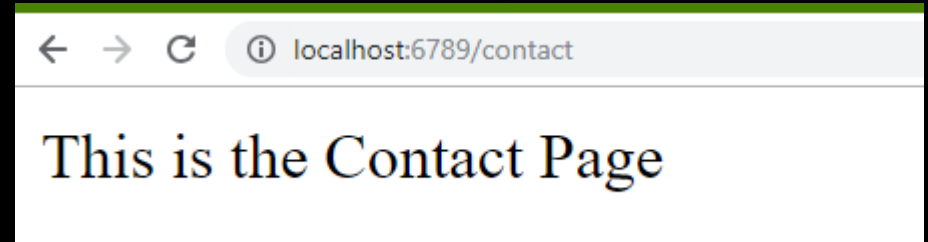
```
19     } else {
20         res.writeHead(404, { 'content-type': 'text/html' });
21         fs.createReadStream(__dirname + '/error.html').pipe(res);
22     }
23 });
24 server.listen(4321);
25 console.log("Server listening at port number 4321");
```



Express install and use

- `npm install express --save`

```
1  var express = require('express');
2  var app = express();
3
4  app.get('/', function (req, res) {
5    res.send("This is the Home Page");
6  });
7
8  app.get('/contact', function (req, res) {
9    res.send("This is the Contact Page");
10 });
11
12 app.listen(6789);
```



Route Parameters

```
1 var express = require('express');
2 var app = express();
3
4 app.get('/', function (req, res) {
5   res.send("This is the Home Page");
6 });
7
8 app.get('/contact', function (req, res) {
9   res.send("This is the Contact Page");
10 });
11 app.get('/profile/:id', function (req, res) {
12   res.send('You are requesting the profile of the user with id : ' + req.params.id);
13 });
14 app.listen(6789);
```

← → ↻ ⓘ localhost:6789/profile/12345

You are requesting the profile of the user with id : 12345

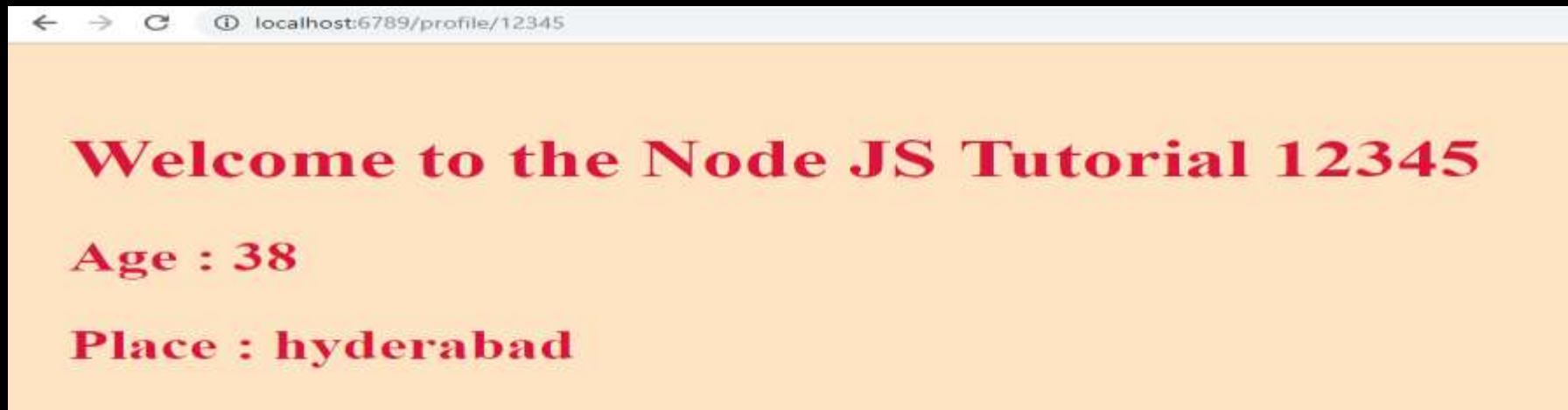
Template Engines

- npm install ejs ~save

```
1  var express = require('express');
2  var app = express();
3
4  app.set('view engine', 'ejs');
5
6  app.get('/', function (req, res) {
7    res.sendFile(__dirname + '/index.html');
8  });
9  app.get('/contact', function (req, res) {
10    res.sendFile(__dirname + '/index.html');
11  });
12  app.get('/profile/:name', function (req, res) {
13    //res.send('You are requesting the profile of the user with id : ' + req.params.id)
14    res.render('profile', { person: req.params.name });
15  });
16  app.listen(6789);
```

```
12 app.get('/profile/:name', function (req, res) {
13     //res.send('You are requesting the profile of the user with id : ' + req.params.id)
14     var data = { age: 38, place: 'hyderabad' }
15     res.render('profile', { person: req.params.name, data: data });
16 });
17 app.listen(6789);
```

```
<body>
  <h2>Welcome to the Node JS Tutorial  <%= person %>
</h2>
  <h3>Age : </h3> <%= data.age %>
  <h3>Place : </h3> <%= data.place %>
</body>
```



Partial Views

```
19 </head>
20
21 <body>
22   <% include partials/nav.ejs %>
23   <h1>Welcome to the Node JS Tutorial</h1>
24 </body>
```

```
EXPLORER
├── OPEN EDITORS
│   └── nav.ejs Practice No...
├── ANGULAR TESTING
│   ├── .vscode
│   ├── node_modules
│   └── Practice Node
│       ├── views
│       │   ├── partials
│       │   │   └── nav.ejs
│       │   ├── contact.ejs
│       │   ├── index.ejs
│       │   └── profile.ejs
│       ├── all.js
│       └── app.js
```

```
1 <nav>
2   <ul>
3     <li>
4       <a href="/">Home</a>
5     </li>
6     <li>
7       <a href="/contact">Contact</a>
8     </li>
9   </ul>
10 </nav>
```

- [Home](#)
- [Contact](#)

Welcome to the Node JS Tutorial



Not over yet. Lot to learn and discuss