

8.3. Backend (NODE.js)

“NODE.JS” é um programa executável que quando chamado, executa um ficheiro “.js”. Pode ser instalado num computador, num servidor, e correr em Backend. Um ficheiro “.js” contém código/instruções em javascript. Além disso, pode também ficar à escuta na porta TCP enquanto executa código javascript.

Server-side with Node.js

<https://www.youtube.com/watch?v=wxbQP1LMZsw>

Node.js Tutorial for Beginners: Learn Node in 1 Hour | Mosh - YouTube

https://www.youtube.com/watch?v=TIB_eWDSMt4

8.3.1. Instalar o programa NODE.JS

Para instalar o “Node.js” aceda ao site:

<https://nodejs.org>

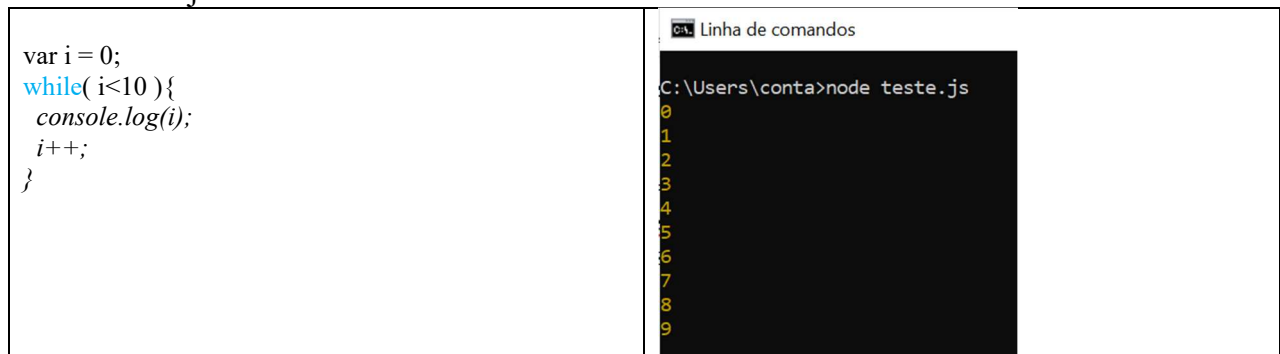
Depois de instalar o programa “Node.js”, deve executá-lo. Para isso, deve escrever “node” na janela de texto “cmd”.

Em síntese:

- 1- Download & install node.js
- 2- Após a instalação, numa janela de comandos (cmd) faça
 - > node -v
 - > npm install npm - global

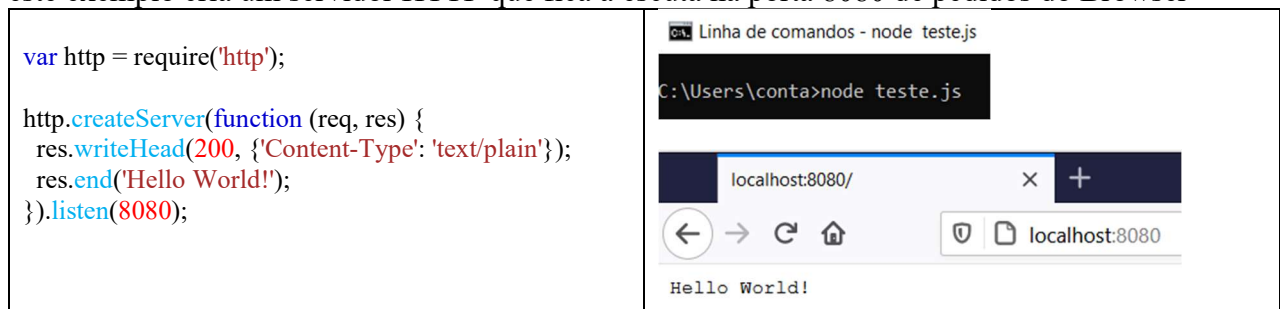
Para testar se o “Node.js” ficou bem instalado, edite este pequeno programa em javascript e guarde-o no ficheiro “teste.js” e na janela “linha de comandos” faça > node teste.js

Ficheiro teste.js



Ficheiro teste.js

este exemplo cria um servidor HTTP que fica à escuta na porta 8080 de pedidos do Browser



Alguns exemplos do www.w3school.com

8.3.2. Exemplo - Variáveis com o NODE.JS

Ficheiro teste.js

```
var length = 16;           // Number
var lastName = "Johnson"; // String
var cars = ["Saab", "Volvo", "BMW"]; // Array
console.log(cars[0]);
console.log(cars.length);
var x = {firstName:"John", lastName:"Doe"}; // Objecto
console.log(x);
console.log(x.firstName);
console.log(x.lastName);
```

CAJ Linha de comandos

```
C:\Users\conta>node teste.js
Saab
3
{ firstName: 'John', lastName: 'Doe' }
John
Doe
C:\Users\conta>
```

8.3.3. Exemplo – Manipulação de string com o NODE.JS

Ficheiro teste.js

```
var str = "Please locate where 'locate' occurs!";
console.log(str);
var pos = str.indexOf("locate");
console.log(pos);

str = "Please locate where last 'locate' occurs!";
console.log(str);
pos = str.lastIndexOf("locate");
console.log(pos);

str = "Apple, Banana, Kiwi";
var res = str.slice(7, 13);
console.log(res);

str = "Apple, Banana, Kiwi";
res = str.substring(7, 13);
console.log(res);

str = "Please visit Microsoft!";
var n = str.replace("Microsoft", "W3Schoolsssss");
console.log(n);
```

CAJ Linha de comandos

```
C:\Users\conta>node teste.js
Please locate where 'locate' occurs!
7
Please locate where last 'locate' occurs!
26
Banana
Banana
Please visit Microsoft!

C:\Users\conta>node teste.js
Please locate where 'locate' occurs!
7
Please locate where last 'locate' occurs!
26
Banana
Banana
Please visit W3Schools!
C:\Users\conta>
```

8.3.4. Exemplo – Endereço URL com o NODE.JS

Ficheiro teste.js (trata o endereço URL)

```
//Split a web address into readable parts:
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'
```

CAJ Linha de comandos

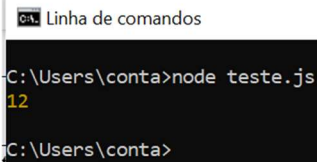
```
C:\Users\conta>node teste.js
localhost:8080
/default.htm
?year=2017&month=february
february
C:\Users\conta>
```

8.3.5. Exemplo – funções com o NODE.JS

Ficheiro teste.js

```
var x = myFunction(4, 3); // Function is called
console.log(x);

function myFunction(a, b) {
  return a * b; // Function returns the product of a and b
}
```



```
Linha de comandos
C:\Users\conta>node teste.js
12
C:\Users\conta>
```

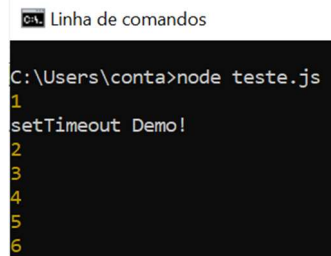
```
/* setInterval de 2 em 2 seg chama a função tt() */
var contador=0;

function tt(){
  contador +=1;
  console.log(contador);
}

var a=setInterval(tt,2000); // de 2 em 2 seg chama a função tt()

//setTimeout() is a callback function to be executed once after the timer expires.
function task() {
  console.log('setTimeout Demo!')
}

setTimeout(task, 3000); // executa a função “task” uma só vez, após 3 segundos
```



```
Linha de comandos
C:\Users\conta>node teste.js
1
setTimeout Demo!
2
3
4
5
6
```

8.3.6. Exemplo – Escrita e leitura de ficheiros de texto com o NODE.JS

Ficheiro teste.js (escrita e leitura em ficheiros de texto)

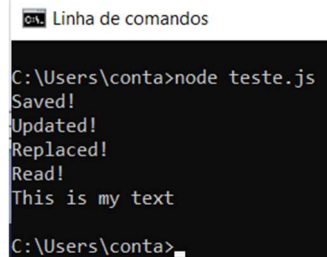
```
var fs = require('fs'); // Usa o módulo fs

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

fs.appendFile('mynewfile1.txt', ' This is my text.', function (err) {
  if (err) throw err;
  console.log('Updated!');
});

fs.writeFile('mynewfile3.txt', 'This is my text', function (err) {
  if (err) throw err;
  console.log('Replaced!');
});

fs.readFile('mynewfile3.txt', function(err,data) {
  if (err) throw err;
  console.log('Read!');
  console.log(data.toString());
});
```



```
Linha de comandos
C:\Users\conta>node teste.js
Saved!
Updated!
Replaced!
Read!
This is my text
C:\Users\conta>
```

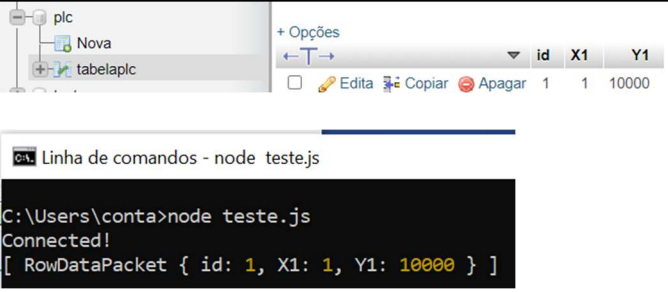
8.3.7. Exemplo – Envio de email com o NODE.JS

Ficheiro teste.js (o servidor node.js envia um email)

<pre>var nodemailer = require('nodemailer'); var transporter = nodemailer.createTransport({ service: 'gmail', auth: { user: 'youremail@gmail.com', pass: 'yourpassword' } }); var mailOptions = { from: 'youremail@gmail.com', to: 'myfriend@yahoo.com', subject: 'Sending Email using Node.js', text: 'That was easy!' }; transporter.sendMail(mailOptions, function(error, info){ if (error) { console.log(error); } else { console.log('Email sent: ' + info.response); } });</pre>	
---	--

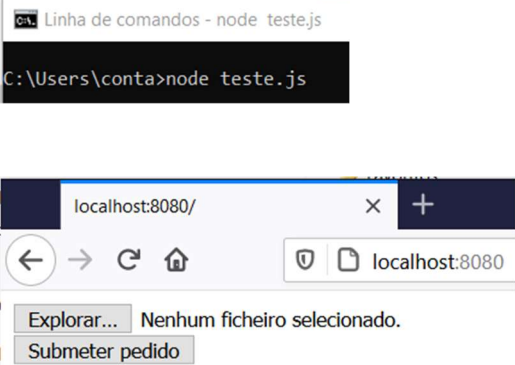
8.3.8. Exemplo – Base de dados MySQL com o NODE.JS

>npm install mysql

<pre>var mysql = require('mysql'); var con = mysql.createConnection({ host: "localhost", user: "root", password: "", database: "plc" }); con.connect(function(err) { if (err) throw err; console.log("Connected!"); con.query("SELECT * FROM tabelaplc", function (err, result, fields) { if (err) throw err; console.log(result); }); });</pre>	
--	--

8.3.9. Exemplo I - Servidor HTTP com o NODE.JS

Ficheiro teste.js (cria um servidor WEB/HTTP que responde com uma página HTML)

<pre>//This code will produce an HTML form: var http = require('http'); http.createServer(QQ).listen(8080); function QQ(req, res) { res.writeHead(200, {'Content-Type': 'text/html'}); res.write('<form action="fileupload" method="post" enctype="multipart/form-data">'); res.write('<input type="file" name="fileupload">
'); res.write('<input type="submit">'); res.write('</form>'); return res.end(); }</pre>	
--	--

8.3.10. Exemplo II - Servidor HTTP com o NODE.JS

Aos 38 min do vídeo https://www.youtube.com/watch?v=TIB_eWDSMt4

--- Editar o ficheiro “app_http.js”

// criar um servidor http em "javascript" e executado pelo "node.exe

// usa a classe http.ClientRequest

// esta classe tem propriedades, métodos e eventos

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

```
const server= http.createServer(); //cria um objecto do tipo servidor http
```

```
server.on('connection',(socket)=>{           // quando chegar um pedido de ligação
  console.log('nova ligação na porta TCP 3000 ');
}
```

```
);
```

```
server.listen(3000);
```

```
console.log('Á escuta na porta TCP 3000 de novas ligações ');
```

---- Executar o ficheiro js na janela “CMD”

```
> node app_http.js
```

---- No browser aceder a:

<http://localhost:3000>

8.3.11. Exemplo III - Servidor HTTP com o NODE.JS

---- criar outro servidor WEB de mais alto nível para responder ao browser

---- Editar o ficheiro index_http.js ----

// criar um servidor http em "javascript" e executado pelo "node.exe

// usa a classe http.createServer

// esta classe tem propriedades, métodos e eventos

```
const http=require('http');
const server= http.createServer(           // o objecto "server" executa esta função
                                           //quando chegarem dados à porta 3000
    function(req,res){

        if(req.url === '/'){               // quando no browser "Http://localhost:3000"
            res.write('Hello world');      // o browser recebe o texto "Hello world"
            res.end();
        }

        if(req.url === '/curso'){          // quando no browser "Http://localhost:3000/curso"
            res.write(JSON.stringify([1,2,3])); // o browser recebe [1,2,3]
            res.end();
        }

    }
);

server.listen(3000);                       // fica escuta de pedidos de ligação e de dados porta3000
console.log('Á escuta na porta TCP 3000 de novas ligações ');
```

---- Executar o ficheiro js na janela "CMD"

> node index_http.js

---- No browser aceder a:

<http://localhost:3000/curso>

8.3.12. Exemplo IV - Servidor HTTP com o NODE.JS + MySQL

//---- criar outro servidor WEB de mais alto nível para responder ao browser e aceder à BD

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

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

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

```
http.createServer(QQ).listen(8081); // Cria um servidor HTTP na porta 8081 do computador
```

//--- Esta função é chamada quando chegar uma msg HTTP ao computador -----

```
function QQ(req, res) {
```

```
  console.log(req.url);
```

```
  var str=String(req.url);
```

```
  console.log(str);
```

```
  var q = url.parse(req.url, true).query;
```

```
  var Y00=q.saidaY0;
```

```
  var Y11=q.saidaY1;
```

```
  var Y22=q.saidaY2;
```

```
  var X22=q.saidaX2;
```

```
  var X33=q.saidaX3;
```

```
  var Nivel=q.Nivel;
```

// -----Define a localização do MySQL e o nome BD -----

```
var con = mysql.createConnection({
```

```
  host: "localhost",
```

```
  user: "root",
```

```
  password: "",
```

```
  database: "reservatorio"
```

```
});
```

// ----- connect to MySQL -----

```
con.connect(function(err) {
```

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

```
  console.log("Connected!");
```

// ----- Envia Query SQL para a BD -----

```
con.query("INSERT INTO historico (y0, y1, y2, x2, x3, nivel) VALUES (" + Y00 + "," + Y11 + "," + Y22 + "," + X22 + "," + X33 + "," + Nivel + ")",
```

```
  function (err, result, fields) {
```

```
    console.log(result);
```

```
  }
```

```
});
```

// ----- Envia Query SQL para a BD -----

```
con.query("UPDATE supervisao_reservatorio SET data=NULL,hora=NULL,y0=" + Y00 + ",y1=" + Y11 + ",y2=" + Y22 + ",x2=" + X22 + ",x3=" + X33 + ",Nivel=" + Nivel + " WHERE leitura=1",
```

```
  function (err, result, fields) {
```

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

```
    console.log(result);
```

```
  }
```

```
});
```

```
con.query("SELECT * FROM supervisao_reservatorio" + " WHERE leitura=1",
```

```
  function (err, result, fields) {
```

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

```
    console.log("SELECT");
```

```
    console.log(result);
```

```
    ss= JSON.stringify(result);
```

// ----- Cria a resposta HTML -----

```
    res.writeHead(200, {'Content-Type': 'text/html'});
```

```
    res.write(ss);
```

```
    res.end();
```

```
  });
```

```
}); // connect to MySQL
```

```
};
```