



Node.js http Module

Mohamed Mukhtar Ahmed

1



Mohamed Mukhtar Ahmed

2

The http Module



Outline

HTTP Server

Common Methods

The request and response objects

Status Codes

Methods

Navigating

Response URL property

Getting content from HTML file



Mohamed Mukhtar Ahmed

3

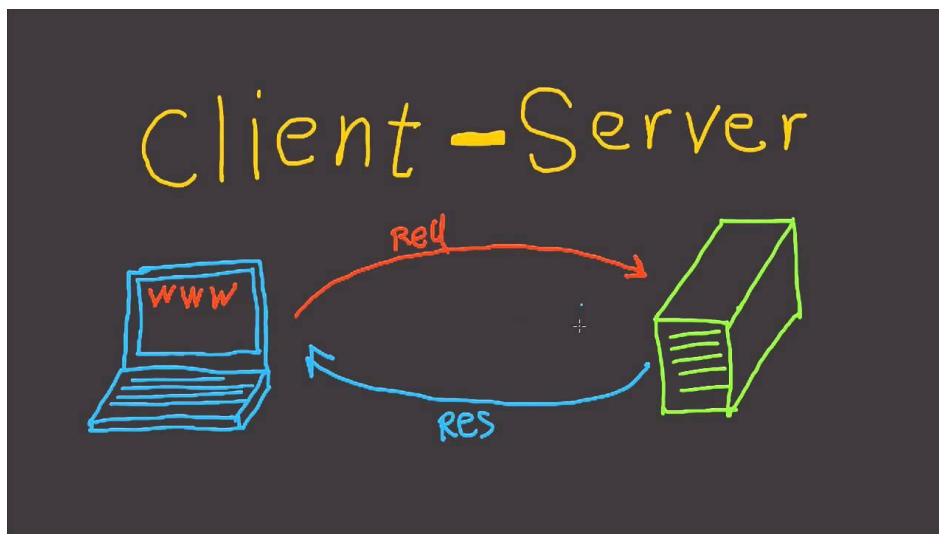
HTTP Server



- We use the **http** module to create an **HTTP server**.
- The **createServer()** method is used to create the HTTP server
- The server is set to **listen()** on the specified port. When the server is ready, the listen callback function is called.
- The **callback function we pass** (to **createServer()**) **is the one that's going to be executed upon every request** that comes in.
- Whenever a new request is received, the request event is called, providing two objects: a **request** (an **http.IncomingMessage** object) and a **response** (an **http.ServerResponse** object).

Mohamed Mukhtar Ahmed

4



Mohamed Mukthar Ahmed

5

The request and response obj



- **request** object provides the request details. Through it, we access the **request headers** and **request data**.
- **response** object is used to populate the data we're going to return to the client.
- Traditionally called as **req** and **res**
- In this case with:

```
res.statusCode = 200;
```

- We set the **statusCode** property to 200, to indicate a successful response.
- We also set the **Content-Type** header:

```
res.setHeader('Content-Type', 'text/html');
```

Mohamed Mukthar Ahmed

6

The request and response obj

- In this case with:

```
res.statusCode = 200;
```

- We set the **statusCode** property to 200, to indicate a successful response.

- We also set the **Content-Type** header:

```
res.setHeader('Content-Type', 'text/html');
```

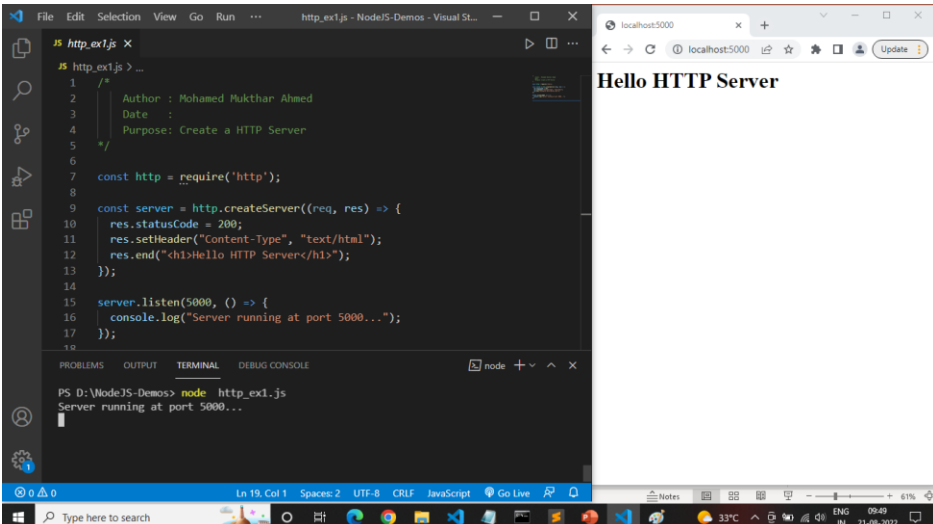
- and we end / close the response, adding the content as an argument to **end()**:

```
res.end('<h1>Hello, World!</h1>');
```

Mohamed Mukthar Ahmed

7

Code Example



```

1  //
2  // Author : Mohamed Mukthar Ahmed
3  // Date   :
4  // Purpose: Create a HTTP Server
5  //
6
7  const http = require('http');
8
9  const server = http.createServer((req, res) => {
10     res.statusCode = 200;
11     res.setHeader("Content-Type", "text/html");
12     res.end("<h1>Hello HTTP Server</h1>");
13 });
14
15 server.listen(5000, () => {
16     console.log("Server running at port 5000...");
17 });
18


```

PS D:\NodeJS-Demos> node http_ex1.js
Server running at port 5000...

localhost:5000
Hello HTTP Server

Mohamed Mukthar Ahmed

8



Status Codes

developer.mozilla.org/en-US/docs/Web/HTTP/Status

mdn web docs References Guides MDN Plus Theme

References > HTTP > HTTP response status codes

HTTP response status codes


HTTP response status codes indicate whether a specific [HTTP](#) request has been successfully completed. Responses are grouped in five classes:

1. [Informational responses](#) (100 – 199)
2. [Successful responses](#) (200 – 299)
3. [Redirection messages](#) (300 – 399)
4. [Client error responses](#) (400 – 499)
5. [Server error responses](#) (500 – 599)

The below status codes are defined by [section 10 of RFC 2616](#). You can find an updated specification in [RFC 7231](#).

Mohamed Mukthar Ahmed

9



HTTP Methods

developer.mozilla.org/en-US/docs/Web/HTTP/Methods

mdn web docs References Guides MDN Plus Theme

References > HTTP > HTTP request methods

HTTP request methods

HTTP defines a set of **request methods** to indicate the desired action to be performed for a given resource. Although they can also be nouns, these request methods are sometimes referred to as *HTTP verbs*. Each of them implements a different semantic, but some common features are shared by a group of them: e.g. a request method can be [safe](#), [idempotent](#), or [cacheable](#).

GET

The **GET** method requests a representation of the specified resource. Requests using **GET** should only retrieve data.

HEAD

The **HEAD** method asks for a response identical to a **GET** request, but without the response body.

Mohamed Mukthar Ahmed

10



■ Navigating / Routing

- The **request** object has the **url** property that can be checked and accordingly navigated (routed).

```
if (req.url === '/') {
  res.end('<h1>Welcome to HOME page!</h1>')
}
```

```
if (req.url === '/about') {
  res.end('<h1>About!</h1>')
}
```

- The **HTML template string** helps to respond with multi-line text.

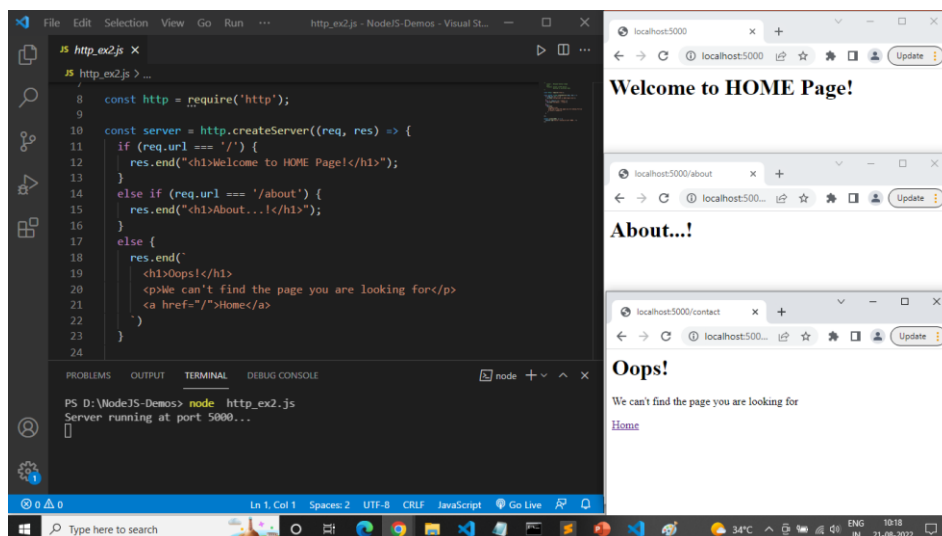
- HTML template string is enclosed in **BACK TICKS**

Mohamed Mukthar Ahmed

11



■ Code Example



Mohamed Mukthar Ahmed

12