

Lesson 05 Demo 06

Creating HTTP Routes

Objective: To create HTTP routes in a Node.js app for handling incoming HTTP requests from a web server

Tools required: Node Package Manager and Visual Studio Code

Prerequisites: Basic Linux Commands, NPM commands, JavaScript, and HTTP module

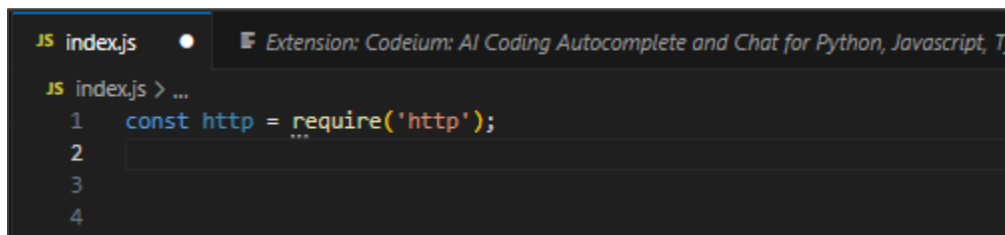
Steps to be followed:

1. Create HTTP routes to handle incoming requests

Step 1: Create HTTP routes to handle incoming requests

1.1 Import the HTTP module:

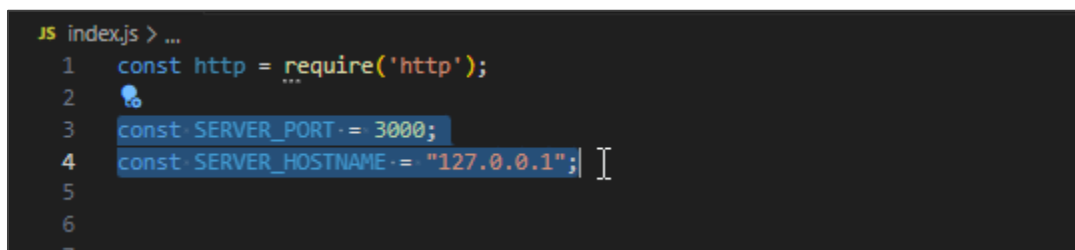
```
const http = require('http');
```



```
JS index.js • Extension: Codeium: AI Coding Autocomplete and Chat for Python, Javascript, T
JS index.js > ...
1  const http = require('http');
2
3
4
```

1.2 Create the server port and hostname:


```
const SERVER_PORT = 3000;
const SERVER_HOSTNAME = "127.0.0.1";
```



```
JS index.js > ...
1  const http = require('http');
2
3  const SERVER_PORT = 3000;
4  const SERVER_HOSTNAME = "127.0.0.1";
5
6
7
```

1.3 Add the function **createServer** to get a reference to the server object:

```
const server = http.createServer();
```



The screenshot shows a VS Code editor window with a file named `index.js`. The code is as follows:

```
JS index.js > ...
1  const http = require('http');
2
3  const SERVER_PORT = 3000;
4  const SERVER_HOSTNAME = "127.0.0.1";
5
6  const server = http.createServer();
7
8
```

1.4 Set the server to listen to various events:

```
server.on("listening", () => console.log("Server Listening"))
```

```
server.on("error", () => console.log("Error Occur while handling request"))
```



The screenshot shows the same VS Code editor window with the following code:

```
JS index.js > ...
1  const http = require('http');
2
3  const SERVER_PORT = 3000;
4  const SERVER_HOSTNAME = "127.0.0.1";
5
6  const server = http.createServer();
7
8  server.on("listening", () => console.log("Server Listening"))
9  server.on("error", () => console.log("Error Occur while handling request"))
10
11
```

1.5 Create the endpoints using the if condition:

```
server.on("request", (req, res) => {  
  const url = req.url;  
  if (url === '/') {  
    res.setHeader("Content-Type", "text/html");  
    res.writeHead(200);  
    res.end('<html><body><h1>Welcome to My App</h1></body></html>');  
  }  
})
```



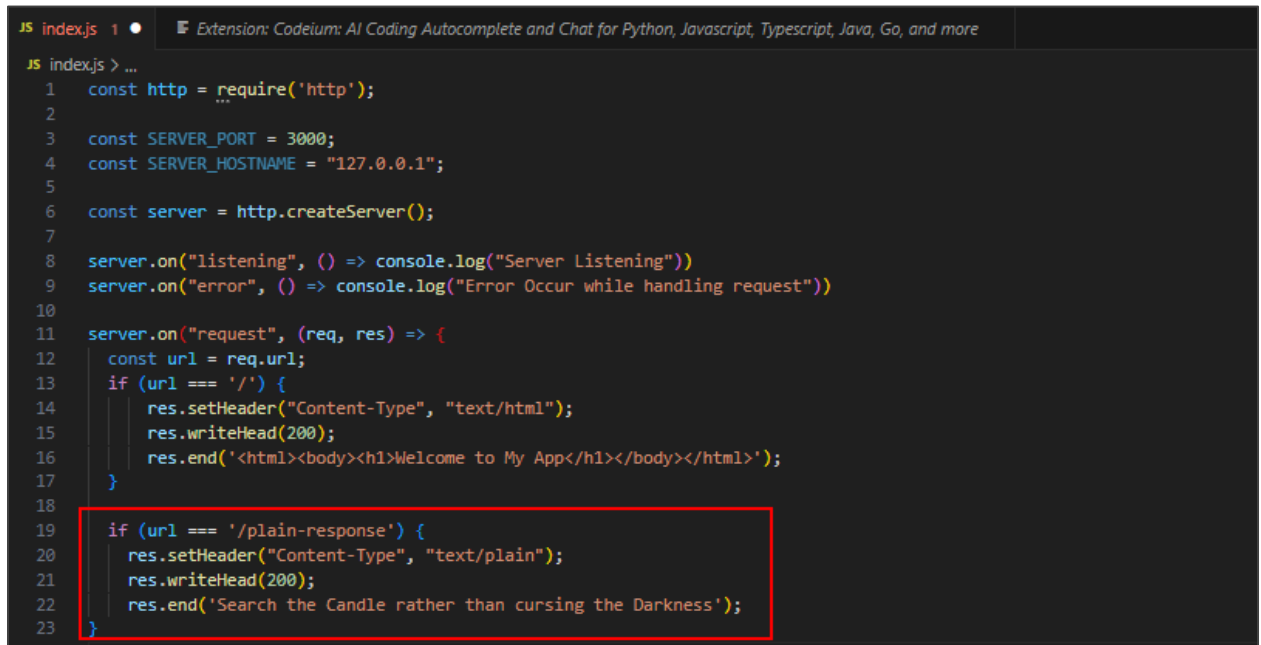
The screenshot shows a code editor with a dark theme. The file name is 'index.js'. The code is as follows:

```
1  const http = require('http');  
2  
3  const SERVER_PORT = 3000;  
4  const SERVER_HOSTNAME = "127.0.0.1";  
5  
6  const server = http.createServer();  
7  
8  server.on("listening", () => console.log("Server Listening"))  
9  server.on("error", () => console.log("Error Occur while handling request"))  
10  
11  server.on("request", (req, res) => {  
12    const url = req.url;  
13    if (url === '/') {  
14      res.setHeader("Content-Type", "text/html");  
15      res.writeHead(200);  
16      res.end('<html><body><h1>Welcome to My App</h1></body></html>');  
17    }  
18  })
```

The code block from line 11 to 17 is highlighted with a red rectangle.

1.6 Create another endpoint using the **if** condition where the URL is **/plain-response** and share the quote as a plain text response:

```
if (url === '/plain-response') {  
    res.setHeader("Content-Type", "text/plain");  
    res.writeHead(200);  
    res.end('Search the Candle rather than cursing the Darkness');  
}
```



The screenshot shows a code editor with a dark theme. The file is named 'index.js'. The code implements a simple HTTP server using Node.js's 'http' module. It listens on port 3000 at the hostname '127.0.0.1'. There are two endpoints defined: a root endpoint '/' that returns an HTML welcome message, and a '/plain-response' endpoint that returns a plain text response. The second endpoint is highlighted with a red rectangular box. The code is as follows:

```
JS index.js 1 ● Extension: Codeium: AI Coding Autocomplete and Chat for Python, Javascript, Typescript, Java, Go, and more  
JS index.js > ...  
1  const http = require('http');  
2  
3  const SERVER_PORT = 3000;  
4  const SERVER_HOSTNAME = "127.0.0.1";  
5  
6  const server = http.createServer();  
7  
8  server.on("listening", () => console.log("Server Listening"))  
9  server.on("error", () => console.log("Error Occur while handling request"))  
10  
11 server.on("request", (req, res) => {  
12     const url = req.url;  
13     if (url === '/') {  
14         res.setHeader("Content-Type", "text/html");  
15         res.writeHead(200);  
16         res.end('<html><body><h1>Welcome to My App</h1></body></html>');  
17     }  
18  
19     if (url === '/plain-response') {  
20         res.setHeader("Content-Type", "text/plain");  
21         res.writeHead(200);  
22         res.end('Search the Candle rather than cursing the Darkness');  
23     }  
24 }
```

1.7 Create another endpoint using the **if** condition where the URL is **/json-response** and share the JSON object as a response:

```
if (url === '/json-response') {
  res.setHeader("Content-Type", "application/json")
  res.end(JSON.stringify({
    "platform": process.platform,
    "date": new Date(),
    "message": "Hellos"
  }));
}
```

```
JS index.js > ...
1  const http = require('http');
2
3  const SERVER_PORT = 3000;
4  const SERVER_HOSTNAME = "127.0.0.1";
5
6  const server = http.createServer();
7
8  server.on("listening", () => console.log("Server Listening"))
9  server.on("error", () => console.log("Error Occur while handling request"))
10
11  server.on("request", (req, res) => {
12    const url = req.url;
13    if (url === '/') {
14      res.setHeader("Content-Type", "text/html");
15      res.writeHead(200);
16      res.end('<html><body><h1>Welcome to My App</h1></body></html>');
17    }
18
19    if (url === '/plain-response') {
20      res.setHeader("Content-Type", "text/plain");
21      res.writeHead(200);
22      res.end('Search the Candle rather than cursing the Darkness');
23    }
24
25    if (url === '/json-response') {
26      res.setHeader("Content-Type", "application/json")
27      res.end(JSON.stringify({
28        "platform": process.platform,
29        "date": new Date(),
30        "message": "Hellos"
31      }));
32    }
33  })
```

1.8 Specify the port number and hostname:

```
server.listen(SERVER_PORT, SERVER_HOSTNAME, () => {
  console.log(`Server is up and listening on port ${SERVER_PORT}`);
})
```

```
JS index.js > ...
8 server.on("listening", () => console.log("Server Listening"))
9 server.on("error", () => console.log("Error Occur while handling request"))
10
11 server.on("request", (req, res) => {
12   const url = req.url;
13   if (url === '/') {
14     res.setHeader("Content-Type", "text/html");
15     res.writeHead(200);
16     res.end('<html><body><h1>Welcome to My App</h1></body></html>');
17   }
18
19   if (url === '/plain-response') {
20     res.setHeader("Content-Type", "text/plain");
21     res.writeHead(200);
22     res.end('Search the Candle rather than cursing the Darkness');
23   }
24
25   if (url === '/json-response') {
26     res.setHeader("Content-Type", "application/json")
27     res.end(JSON.stringify({
28       "platform": process.platform,
29       "date": new Date(),
30       "message": "Hellos"
31     }));
32   }
33 })
34
35 server.listen(SERVER_PORT, SERVER_HOSTNAME, () => {
36   console.log(`Server is up and listening on port ${SERVER_PORT}`);
37 })
38
```

1.9 Test the code by executing the following command in the terminal:

node index.js

```
demopythonlyopm@ip-172-31-16-204:~/Desktop/nodeProjec/demo4$ node index.js
```

Following are the outputs:



By following these steps, you have successfully created HTTP routes in a Node.js app for handling incoming HTTP requests from a web server.