

# Exercise (Instructions): Node and the HTTP Module

## Objectives and Outcomes

In this exercise, you will explore three core Node modules: HTTP, fs and path. At the end of this exercise, you will be able to:

- Implement a simple HTTP Server
- Implement a server that returns html files from a folder

## A Simple HTTP Server

- Create a folder named *node-http* at a convenient location and move into the folder.
- In the *node-http* folder, create a subfolder named *public*.
- Create a file named *server-1.js* and add the following code to it:

```
1 var http = require('http');
2
3 var hostname = 'localhost';
4 var port = 3000;
5
6 var server = http.createServer(function(req, res){
7   console.log(req.headers);
8   res.writeHead(200, { 'Content-Type': 'text/html' });
9   res.end('<html><body><h1>Hello World</h1></body></html>');
10 })
11 server.listen(port, hostname, function(){
12   console.log(`Server running at http://${hostname}:${port}/`);
13 });
```

- Start the server by typing the following at the prompt:

```
1 node server-1
```

- Then you can type `http://localhost:3000` in your browser address bar and see the result.
- You can also use postman chrome extension to send requests to the server and see the response.

## Serving HTML Files

- In the *public* folder, create a file named *index.html* and add the following code to it:

```

1 <html>
2 <title>This is index.html</title>
3 <body>
4 <h1>Index.html</h1>
5 <p>This is the contents of this file</p>
6 </body>
7 </html>

```

- Similarly create an aboutus.html file and add the following code to it:

```

1 <html>
2 <title>This is aboutus.html</title><body>
3 <h1>Aboutus.html</h1><p>This is the contents of the aboutus.html file</p></body>
4 </html>

```

- Then create a file named *server-2.js* and add the following code to it:

```

1 var http = require('http');
2 var fs = require('fs');
3 var path = require('path');
4
5 var hostname = 'localhost';
6 var port = 3000;
7
8 var server = http.createServer(function(req, res){
9   console.log('Request for ' + req.url + ' by method ' + req.method);
10  if (req.method == 'GET') {
11    var fileUrl;
12    if (req.url == '/') fileUrl = '/index.html';
13    else fileUrl = req.url;
14    var filePath = path.resolve('./public'+fileUrl);
15    var fileExt = path.extname(filePath);
16    if (fileExt == '.html') {
17      fs.exists(filePath, function(exists) {
18        if (!exists) {
19          res.writeHead(404, { 'Content-Type': 'text/html' });
20          res.end('<html><body><h1>Error 404: ' + fileUrl +
21                ' not found</h1></body></html>');
22          return;
23        }
24        res.writeHead(200, { 'Content-Type': 'text/html' });
25        fs.createReadStream(filePath).pipe(res);
26      });
27    } else {
28      res.writeHead(404, { 'Content-Type': 'text/html' });
29      res.end('<html><body><h1>Error 404: ' + fileUrl +
30            ' not a HTML file</h1></body></html>');
31    }
32  } else {
33    res.writeHead(404, { 'Content-Type': 'text/html' });
34    res.end('<html><body><h1>Error 404: ' + req.method +
35          ' not supported</h1></body></html>');
36  }
37 }
38
39 server.listen(port, hostname, function(){
40   console.log('Server running at http://${hostname}:${port}/');
41 });

```

- Start the server, and send various requests to it and see the corresponding response.

## Conclusions

In this exercise you learnt about using the Node HTTP module to implement a HTTP server.

✓ Complete

