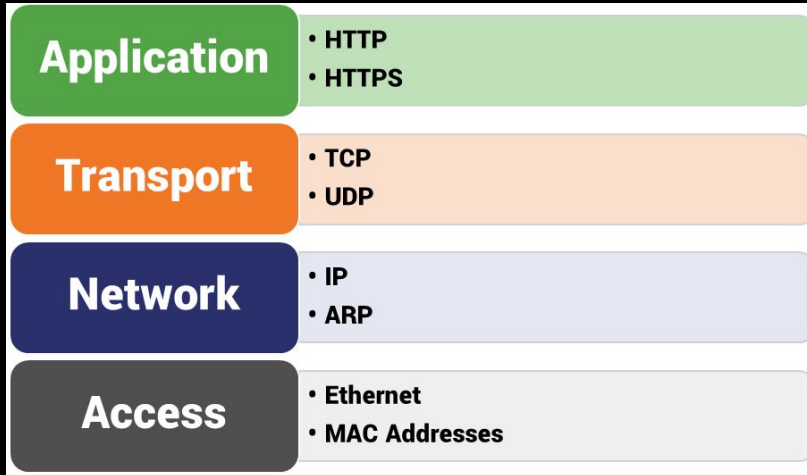# BCDV 1022
Node - HTTP & Web

2023 March
week 02 - class 02

Topics

- ○ **Node Networking and HTTP**
- ○ **Node HTTP Module**
- ○ **Creating a Web Server**

Node Networking and HTTP

Node.js and Networking

- Although, Node.js can be use for a wide variety of tasks, it's mostly known for building web applications.

- Node.js is thrives in the networking due to its asynchronous nature and build-in modules such as net and http.

What is a Web Server?

- A Web Server is a software application which handles HTTP requests sent by the HTTP client, like web browsers, and returns web pages in response to the clients.

- Web servers usually deliver html documents along with images, style sheets, and scripts.
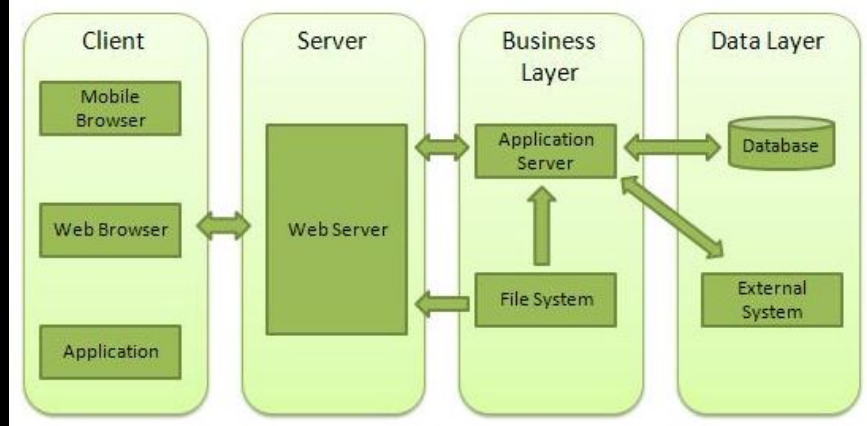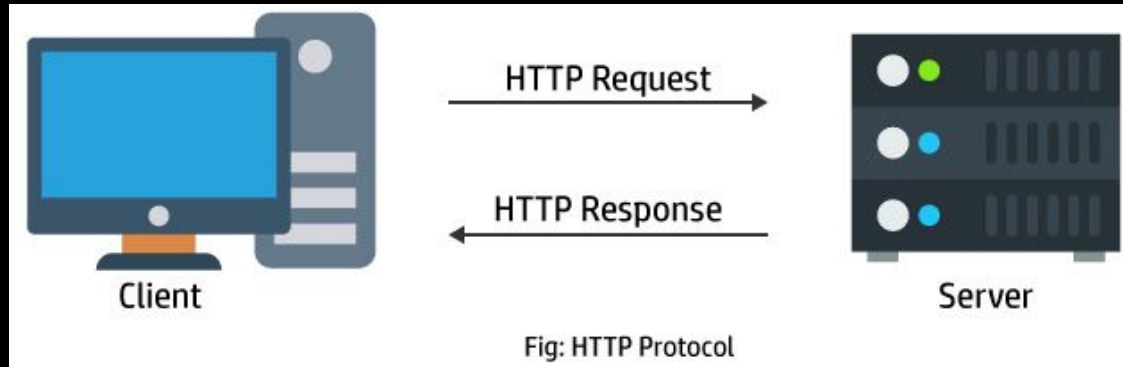
HTTP Module

What is HTTP?

- Hypertext Transfer Protocol is a connectionless text based protocol.

- It is used to send and receive web pages and files on the internet. It is the foundation of WWW (world wide web)

- Client (web browsers) send a request to web servers for web files. The server will transfer the data over the web, then disconnect.

Node.js and HTTP

- One of the powerful building blocks of node is the HTTP module that we use for creating networking applications.
- We can create a web server that listens for HTTP requests on a given port.
- We can use this and create a back-end service for our client application like a web application

Fig: HTTP Protocol

HTTP Module

- The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client.
- The HTTP module can included using require

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

Methods

- http.createServer() - return a new instance of the http.Server class
- http.request() - Makes an HTTP request to a server
- http.get() - similar to http.request, but automatically sets the HTTP method to GET

```
const server = http.createServer((req, res) => {
  //handle every single request with this callback
})
```

Creating a Web Server
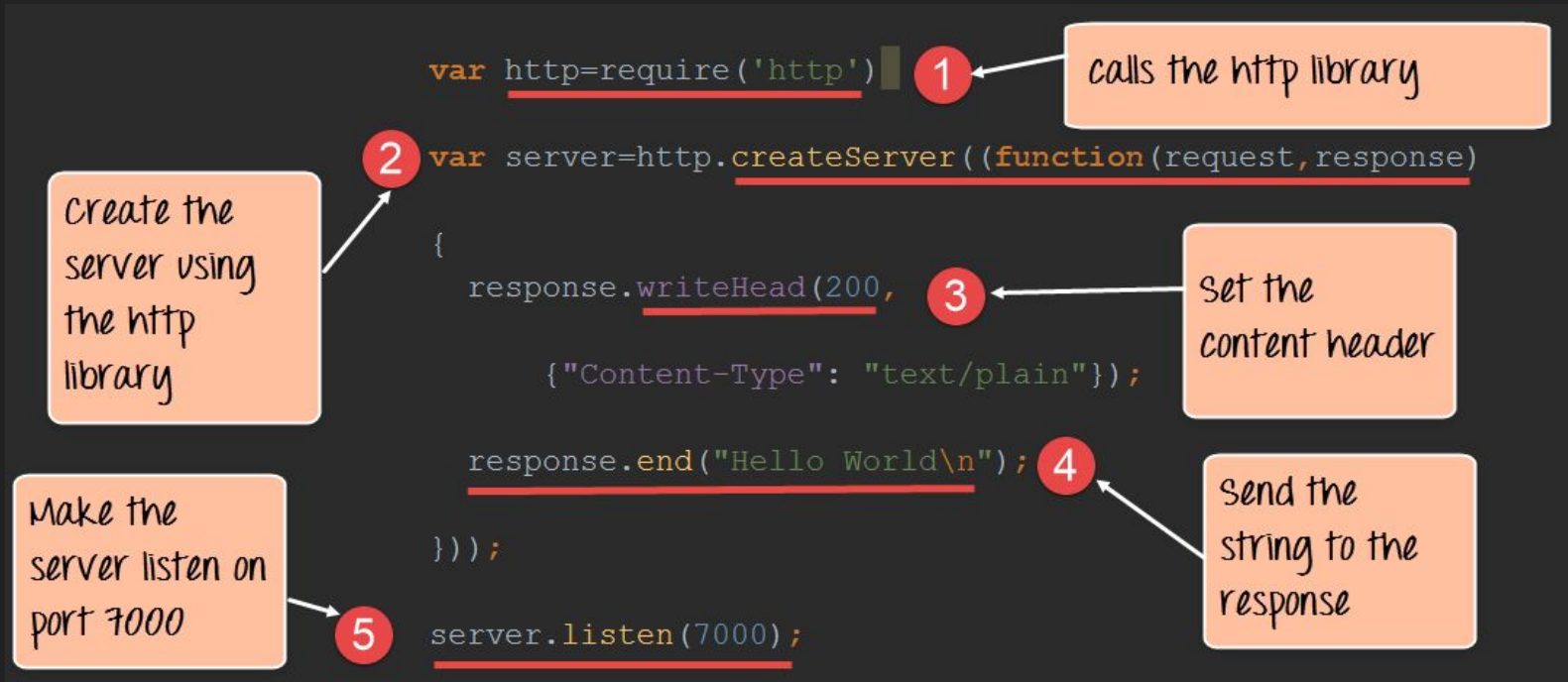
Node.js Server in minutes..

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

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Node as a web server using HTTP


```
var http=require('http')
```
1 calls the http library

```
var server=http.createServer((function(request,response)
```
2 Create the server using the http library

```
{
    response.writeHead(200,
```
3 Set the content header

```
        {"Content-Type": "text/plain"});

    response.end("Hello World\n");
```
4 Send the string to the response

```
}));
```

```
server.listen(7000);
```
5 Make the server listen on port 7000

Node as Client: Handling GET Requests

Using the request module

Sending the response from Google to console.log

```
var request = require("request");

request("http://www.google.com", function(error, response, body)
{
    console.log(body);
});
```

1

2

3

Making a GET request to Google.com

Alternate Syntax


```
http.createServer(function(request, response){ ... });
```

Same as

```
var server = http.createServer();
server.on('request', function(request, response){ ... });
```

This is how we add add event listeners

Event: 'close'

```
function () { }
```

Emitted when the server closes.

```
server.on('close', function(){ ... });
```