CS3215: Web Technology TY Div C n D AY 2021-22

Study Material for Section-II-Part-III- Node JS

Learning Node JS [Javascript Library for Server side module development]

[Source: www.w3schools.com]

Prerequisites:

HTML5, CSS3, BOOTSTRAP4.0, jQuery, JSON, DOM, JavaScript, ES6, Node.js, npm- node package manager

What is Node.js?

- Node.js is an open source **server** environment
- Node.js is free
- Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- Node.js uses JavaScript on the server

Node.js uses asynchronous programming!

A common task for a web server can be to open a file on the server and return the content to the client.

Here is how PHP or ASP handles a file request:

1. Sends the task to the computer's file system.
2. **Waits while the file system opens and reads the file.**
3. Returns the content to the client.
4. Ready to handle the next request.

Here is how Node.js handles a file request:

1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting, and simply continues with the next request.

Node.js runs single-threaded, non-blocking, asynchronous programming, which is very memory efficient.

What Can Node.js Do?

- Node.js can generate dynamic page content
- Node.js can create, open, read, write, delete, and close files on the server
- Node.js can collect form data
- Node.js can add, delete, modify data in your database

What is a Node.js File?

- Node.js files contain tasks that will be executed on certain events
- A typical event is someone trying to access a port on the server
- Node.js files must be initiated on the server before having any effect
- Node.js files have extension ".js"

Download Node.js

The official Node.js website has installation instructions for Node.js: https://nodejs.org

After Installation:

Create a folder on D/I drive as : I:>nodejs

Store your files to be run in this folder

You can run your code: Either from console as I:>nodejs> node first.js

```
var msg = 'Hello World';
console.log(msg);
```

Run it from console:

I:>nodejs> node first.js

You can run your code: or from browser by using localhost:8080

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

http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World!');
}).listen(8080);
```

Run it from browser as

http://localhost:8080

```
const FtpSrv = require('ftp-srv');

const ftpServer = new FtpSrv({

    url: "ftp://0.0.0.0:" 21, anonymous: true});

ftpServer.listen().then(() => {

    console.log('Ftp server is starting...')

});
```

```
const  SMTPServer = require("smtp-server");

const server = new SMTPServer({

  secure: true,

  key: fs.readFileSync("private.key"),

  cert: fs.readFileSync("server.crt")

});

server.listen(465);
```

Node.js Modules

What is a Module in Node.js?

Consider modules to be the same as JavaScript libraries.

A set of functions you want to include in your application.

Built-in Modules

Node.js has a set of built-in modules which you can use without any further installation.

List of modules:

Module	**Description**

assert	Provides a set of assertion tests
buffer	To handle binary data
child_process	To run a child process
cluster	To split a single Node process into multiple processes
crypto	To handle OpenSSL cryptographic functions
dgram	Provides implementation of UDP datagram sockets
dns	To do DNS lookups and name resolution functions
domain	Deprecated. To handle unhandled errors
events	To handle events
fs	To handle the file system
http	To make Node.js act as an HTTP server
https	To make Node.js act as an HTTPS server.
net	To create servers and clients
os	Provides information about the operation system

path	To handle file paths
punycode	Deprecated. A character encoding scheme
querystring	To handle URL query strings
readline	To handle readable streams one line at the time
stream	To handle streaming data
string_decoder	To decode buffer objects into strings
timers	To execute a function after a given number of milliseconds
tls	To implement TLS and SSL protocols
tty	Provides classes used by a text terminal
url	To parse URL strings
util	To access utility functions
v8	To access information about V8 (the JavaScript engine)
vm	To compile JavaScript code in a virtual machine
zlib	To compress or decompress files

Node.js using http module

Create a server that listens on port 8080 of your computer.

Definition and Usage

The HTTP module provides a way of making Node.js transfer data over HTTP (Hyper Text Transfer Protocol).

Syntax

The syntax for including the HTTP module in your application:

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

HTTP Properties and Methods

Method	Description
createClient()	Deprecated. Creates a HTTP client
createServer()	Creates an HTTP server
get()	Sets the method to GET, and returns an object containing the user's request
globalAgent	Returns the HTTP Agent
request()	Returns an object containing the user's request

How to Include Modules?

To include a module, use the require() function with the name of the module:

var http = require('http');

Now your application has access to the HTTP module, and is able to create a server:

```
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Hello World!');
}).listen(8080);
```

Create Your Own Modules

You can create your own modules, and easily include them in your applications.

The following example creates a module that returns a date and time object:

Example

Create a module that returns the current date and time:

```
exports.myDateTime = function () {
  return Date();
};
```

Use the exports keyword to make properties and methods available outside the module file.

Save the code above in a file called "myfirstmodule.js"