**Learn Node Js**

**Introduction to Node Js**

**\*\*What is Node js?**:

Node js is an open source, server side javascript runtime. It allows us to run our javascript code outside the browser like in our local machine or servers. Node javascript is built on the V8 javascript engine for speed. It has a robust ecosystem with NPM.

**\*\* What is javascript runtime environment?**

Javascript runtime environment refers to the environment in which our javascript code runs. It typically runs in a browser like Chrome or Firefox to handle frontend tasks.

What node js does is, it provides us an environment to run the javascript code outside the browser like in a local machine or inside a server.

**Node Module System:**

**\*\*Module Wrapper Function:**

In Node Js, whatever module we create, is wrapped in a function before execution. This function is called a module- wrapper function.

This is what a module-wrapper function looks like:

1. (function(exports, require, module, \_\_filename, \_\_dirname) {

2. // Your module code actually lives here

3. });

4.

Here we can see that a module wrapper function has 5 parameters. **Remember, if we try to print any of the parameter values, the names need to be exactly the same that’s in the example.**

**Node Package Manager (NPM):**

**\*\*What is NPM:**

Node package manager or NPM is the default package manager for node js. It allows us to **install packages**, **manage dependencies**, **publish packages** and **run scripts to automate tasks like testing, building and deployment**.

**\*\*package.json File:**

It is a configuration file that contains the metadata about my project. It also contains:

1. Project details like name, author name etc.
2. Lists all dependencies
3. Scripts to automate tasks like starting the project or testing etc.

**\*\*Initializing NPM in project or folder:**

To initialize NPM in a project we go the folder that we want to initialize it in and type **“npm init -y”**. This initializes the npm in that folder and gives us a **“package.json”** file.

**\*\*Installing, updating and deleting packages from NPM:**

\*To install a package:

**npm install packageName**

We can also install multiple packages at once:

**npm install packageName1 packageName2**

\*to update a package:

**npm update packageName**

\*to delete a package:

**npm delete packageName**

**\*\*Adding a new script in package.json file:**

We can add a new script inside the **package.json** file under the “**scripts**” section. Inside the **package.json** file everything is a key value pair. For the scripts section, the key determines the action. Which means, what will it do when we run the script and the value determines with what command do we run the script.

Example:

**“start”: “node index.js”**

This means the index.js file should start or should be executed when we write **node index.js** in the terminal.

**Path Module**

**\*\*What is path module?**

Path module is a built-in module in node js which provides us utilities while working with file and directory/folder paths.

Here’s some examples on how to get the directory name of a file, how to get the file name of a file and how to get the extension name of a file. We have to remember, we need to always import/require the path module first.

1. // importing the path module

2. const path = require("path");

3.

4. // printing out the current directory of this index.js file

5. console.log("Directory Name: ", path.dirname(\_\_dirname));

6.

7. // Printing out the file name of this index.js file

8. console.log("File name: ", path.basename(\_\_filename));

9.

10. // Printing out file extension name of this index.js file

11. console.log("File Extension name: ", path.extname(\_\_filename));

**File Module**

**\*\*What is file system or file module?**

File system or file module helps us to work with files.

To get started we have to require the file system and path module. Like:

1. const fs = require(“fs”);

2. const path = require(“path”);

**\*\*Joining paths using path.join():**

It’s a method which we can use to join multiple directories/paths together to create a one complete directory/path. We can create both files and folders with this.

**Example:**

1. const path = require('path');

2. // Joining path segments to create files and folders

3. const fullPath = path.join('documents', 'projects', 'report.txt');

**\*\* checking if a folder exists with fs.existsSync() method:**

To check if a folder or a file exists we can use the fs.existsSync() method.

It goes like:

1. if(fs.existsSync(yourFolderPath)){

2. // Then do this

3. }

**\*\* creating a new folder with fs.mkdirSync() method:**

We can create a new folder with fs.mkdirSync() method.

It goes like:

1. fs.mkdir(yourfolderPath);

**\*\*writing some text or content inside the file with fs.writeFileSync() method:**

To write some content inside the file, we use fs.writeFileSync() method.

It goes like:

fs.writeFileSync(yourFilePath, "file content.");

**\*\*Reading content from the file with fs.readFileSync() method:**

To read content from the file, we use fs.readFileSync() method.

It goes like this:

*// Reading the content of a file*

const readContentFromFile = fs.readFileSync(filePath, "utf8");

console.log("File Contents:", readContentFromFile);

**\*\*Adding content into the file using fs.appendFileSync() method:**

To add something to a file, we use fs.appendFileSync() method.

It goes like this:

*// Adding content to the file*

fs.appendFileSync(filePath, "\nThis is a new line added to the file");