Node JS Document

* We can get data from (HTML forms, postman) through node using express and land that data on an URL but store in nothing.

<https://nodejs.org/docs/latest-v17.x/api/>

<https://www.w3schools.com/nodejs/> (important link)

File system Module

<https://www.w3schools.com/nodejs/ref_fs.asp>

To Play, Control, Read, Write, Delete, Update, Rename File Through JavaScript We Use File System

There are two types of file system in node JS modules

1. Synchronous file system (let me done and then run)
2. Asynchronous file system (don’t wait do your work I will be there )
3. **Synchronous file system**

* No call backs are allowed

const fs = require("fs")

//1..............synchronous file system

//create a folder and name it adil younas

fs.mkdirSync("./adil younas")

//create a file in it named bio.txt and add data into it.

fs.writeFileSync("./adil younas/bio.txt","hello i am adil younas and i am a full stack web developer")

//add more data in it (not override)

fs.appendFileSync("./adil younas/bio.txt"," asdjlfsdasdfjkl")

// read data without buffering

const a = fs.readFileSync("./adil younas/bio.txt","utf-8")

console.log(a);

// rename the folder name name to adil younas to aqil younas

fs.renameSync("./adil younas","aqil younas")

// rename the file in folder and named it mybio.txt

fs.renameSync("./aqil younas/bio.txt" , "./aqil younas/mybio.txt" )

// delete file from folder

fs.rmSync("./aqil younas/mybio.txt")

// delete folder

fs.rmdirSync("./aqil younas")

1. **Asynchronous file system**

* Asynchronous file system is all about call back

const fs = require("fs")

//2..............Asynchronous file system

//1..create a folder and name it adil younas

fs.mkdirSync("adil younas",(err)=>{

    console.log("folder created");

    console.log(err);

})

//2....create a file in it named bio.txt and add data into it.

fs.writeFile("./adil younas/bio.txt","my name is adil younas",(err)=>{

    console.log(err);

    console.log("file is created");

})

//3.....add more data in it (not override)

fs.appendFile("./adil younas/bio.txt","i am aqil younas",(err)=>{

    console.log("file updated");

    console.log(err);

})

//4..... read data without buffering

fs.readFile("./adil younas/bio.txt","utf-8",(err,data)=>{

    console.log(data);

    console.log(err);

})

//5...... rename the folder name name to adil younas to aqil younas

fs.rename("./adil younas","./abeeha younas",(err)=>{

    console.log("folder renamed");

    console.log(err);

})

// 6.... rename the file in folder and named it mybio.txt

fs.rename("./abeeha younas/bio.txt","./abeeha younas/mybio.txt",(err)=>{

    console.log("file renamed");

    console.log(err);

})

// 7... delete file from folder

fs.rm("./abeeha younas/mybio.txt",(err)=>{

    console.log(err);

    console.log("file deleted");

})

// 8..... delete folder

fs.rmdir("./abeeha younas",(err)=>{

    console.log(err);

    console.log("delete folder");

})

OS Modules

<https://www.w3schools.com/nodejs/ref_os.asp>

* Tell us All about you computer window ,memory, hostname, etc

Path Modules

<https://www.w3schools.com/nodejs/ref_path.asp>

* Tell us everything about file and path you can even join the paths

const path = require("path")

console.log(path.dirname("C:/Users/Aqil/Desktop/New folder/index.js"));

console.log(path.extname("C:/Users/Aqil/Desktop/New folder/index.js"));

console.log(path.basename("C:/Users/Aqil/Desktop/New folder/index.js"));

console.log(path.parse("C:/Users/Aqil/Desktop/New folder/index.js") );

const path1 = "adil younas"

const path2 = "/aqil younas"

console.log( path.join(path1 + path2)  );

wrapper function

* In node all data in a single file wrap in a function which is called wraper function

//considered this function a whole file

function wrapper(require,exports,modules,\_\_filename,\_\_dirname){

//all data you write in file of .js like

const fs = require("fs")

}

Express js

Step One

* Require express
* Store into app
* Set route for home, contact us, etc using app
* Use only sendFile, send and give it path by using path.join()
* Create a server using app

const express = require("express");

const path = require("path");

const app = express();

const port = 3000;

const home = path.join(\_\_dirname + "/public/home.html");

app.get("/", (req, res) => {

  res.sendFile(home);

});

app.post("/is url per send krna", (req, res) => {

  console.log(req.body) //undefined

});

app.listen(port, () => {

  console.log("Server is working fine");

});

Very Important Note: If you send data through form then data will send on action URL and you will get data in browser through URL but if you use thunder client body you will receive it only console. And remember routing is one thing and getting data is another thing and getting data through post man is an other thing so use app.get() for every individual and always use req.send() to prevent loading

Access form data or html input tag from body

* To access data from body you must acquire body-parser package from NPM

const bodyParse = require("body-parser")

app.use(bodyParse.urlencoded({extended:false}))

Access JSON data from body

* While testing api you have to use postman or thunder client suppose you use thunder client and in its body you use json data but express does not understand that data come from body is another from so you must have to do that mention in the following

app.use(express.json())

Node js and express concept are died here…..RIP

* For the sake of simplicity we use router which is also a concept of express but because it is out of league.

1. Ready route file

const express = require("express");

const router = express.Router();

const registerUser = require("./UserController");

router.route("/register").post(registerUser);

router.route("/product").delete(registerUser);

module.exports = router;

1. Ready function file

const registerUser = (req, res) => {

  const userName = req.body.name;

  const userEmail = req.body.email;

  const userPassword = req.body.password;

  res.json({

    success: true,

  });

};

module.exports = registerUser;

1. If you want to use function into route file then you must require it first
2. Tell you index file that you are using router

const router = require("./UserRoutes");

app.use(router);

All about node and express finished

Congratulation

………………………………………………………………