NODE.JS PRACTICE NOTES.

- Node.js is javascript running on server.
- Npm = node package manager.
- Repl = repl is nothing but cmd of node.js.
- To enter in node server: > node command execute kar.
- To enter in editior mode : > .editor and perform loops,functions or anything..
- .help = command help you in know all basic things like(.exit)
- Double_Tab = all funcions or method are listed.If we want to know about a particular module or
- method type it like (fs);
- Template/String literal use incase :-

Although, using `$` does not require using a library. All you have to do is swap `document.getElementById()` for `$()`. The purpose of `getElementById()` and add the following definition of the `$()` function to your code:

```
function $(x) {
  return document.getElementById(x);
}
```

However, in ECMAScript 6 (ES6) the $ may represent a Template Literal, just like we use ust {} in python f-strings we use ${} in javascript to indicate a placeholder for variables.

```
let user = 'Bob'
console.log(`We love ${user}.`);
```

❖ FILE MANIPULATION.....(SYNCHJRONOUS)

```
const fd = require('fs');
fd.writeFileSync("doc.txt","HELLO SUJAL");
const toe = fd.readFileSync("doc.txt");
const tot = toe.toString();
console.log(tot);
fd.renameSync("doc.txt","pdf.txt");
```

- When you read file it represent data in binary format because Node.js contain additional buffer data type as compared to javascript.

- So to read or print actual data on console use toString() method.
- AppendFileSync is also use to add additional data in file.
- RenameSync is use to rename the file name.
- Delete file = use unlinkSync(); //command.
- To mkdir and for going to use cd .\dir\ name.

❖ **NOW, USING SYNCHRONOUS DONE MANIPULATION OF FILE.**

```
const fs = require('fs');
const { rm } = require('fs/promises');
fs.writeFile("doc.txt","Hello Sujal",(err)=>
{
    console.log("File is created");
    console.log(err);
});

fs.appendFile("doc.txt","AGAYA ME ",(err)=>{
    console.log("data add hogyaa");
    console.log(err);
})
const t1 = fs.readFile("doc.txt","UTF-8",(err,data)=>{
    console.log(data);
    console.log(err);
});
//const t2=t1.toString();

//console.log(t2);

fs.unlink("doc.txt",(err)=> {
    console.log("FILE DELETED.");
    console.log(err);
});

fs.rmdir(".\node.program\",(err)=> {
    console.log("folder is deleted.");
});
```

- While reading file we have to use “UTF-8” for getting avoid the buffer data.
- In this type, we have to use call back function compulsory…

❖ **MODULE => OS**

Reference link =>https://nodejs.org/api/os.html#osarch

```
const fs = require('os');
console.log(fs.arch()); //check operating system
const t1 = console.log(fs.freemem()); //check memory space
console.log(`${t1 /1024/1024/1024}`);
console.log(fs.cpus());  //Returns an array of objects containing information
about each logical CPU core.
console.log(fs.hostname());  //Returns the host name of the operating system as a
string.
console.log(fs.machine());  //Returns the machine type as a string, such as arm,
arm64, aarch64, mips, mips64, ppc64, ppc64le, s390, s390x, i386, i686, x86_64.
console.log(fs.networkInterfaces()); //Returns an object containing network
interfaces that have been assigned a network address.
const t2 = console.log(fs.totalmem()); //Returns the total amount of system
memory in bytes as an integer.
console.log(`{t2 /1024/1024/1024}`); // To check memory in gb.
console.log(fs.uptime());  //Returns the system uptime in number of seconds.
console.log(fs.platform());
console.log(fs.type());
```

- ❖ **MODULE PATH**

```
const pt = require('path');
console.log(pt.dirname("S:/PROGRAMS/Node_program/index.js")); //returns the
directory name of a path, similar to the Unix dirname command.
console.log(pt.extname("S:/PROGRAMS/Node_program/index.js")); //returns the
extension of the path.
console.log(pt.basename("S:/PROGRAMS/Node_program/index.js")); // returns the
last portion of a path, similar to the Unix basename command.
console.log(pt.parse("S:/PROGRAMS/Node_program/index.js")); // returns an object
whose properties represent significant elements of the path.
```

- ➢ **CREATE OUR OWN MODULE AND EXPORTS IT.....**

```
const add = (a,b)=>{
    return a+b;
};

const sub = (a,b)=>{
    return a-b;
};

const name = "sujal";

// module.exports.add1=add;
// module.exports.sub1=sub;
// module.exports.name1 = name;

module.exports={add,sub,name};
```

```
const {add,sub,name}= require("./end");
console.log(add(9,8));
console.log(sub(5,9));
console.log(name);

//using object calling the function:-

// const oper = require("./end");
// console.log(oper.add(3,5));
// console.log(oper.sub(4,5));
```

IMPORT NPM PAKCAGE.....

To instal or get this below json package your have to initiate npm in terminal.

Package.json = Details of all package and lib which you use in your project…store here.

```
{
  "name": "node_program",
  "version": "1.0.0",
  "description": "LOVE TO NODE JS....",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Sujal Dingankar",
  "license": "ISC"
}
```

Now we use chalk module in our project so that module details are add in that package.

To use chalk module you have to install command > npm install chalk.

Imp_Note = to use chalk module first go to chalk npm site and check supported version currently.

To install that supported version use command > npm install chalk@4.0.0.

```
const chalk = require("chalk");
console.log(chalk.yellow("hello sujal"));

console.log(`
    name:${chalk.red('sujal')}
    marks:${chalk.green('92')}
`);
```

Inverse = both text and background affected….

Below example is simple play with chalk module.

Reference link = https://www.npmjs.com/package/chalk

```
const chalk = require("chalk");
console.log(chalk.yellow("hello sujal"));

console.log(`
    name:${chalk.red('sujal')}
    marks:${chalk.green('92')}
`);

console.log(chalk.bold.underline.red("shadow"));

console.log(chalk.green.underline.inverse("success"));

console.log(chalk.red.inverse("cancel"));
```

IMPORT VALIDATOR PACKAGE.....

Command > npm install validator

```
const validator = require("validator");
const chalk = require("chalk");
const val = validator.isEmail("sujal@sujal.com");
console.log(val ?
chalk.green.inverse(val):chalk.red.inverse(val));
```

Simple to use any module of npm package just you have to install than import and use it..

Very strong validation In node.js npm package we combine chalk module with it to do more fun in above example.

NODEMON INSTALL USING COMMAND > npm I nodemon

Amazing feature of nodemon = sirf save karo or restart hoil direct...

Error occur = while downloading nodemon some errror occur like check command >nodemon -v if is not showing than go to that specified dir and delete that folder.

...MODULE WRAPPER FUCNTION...

Module Wrapper Function: Under the hood, NodeJS does not run our code directly, it wraps the entire code inside a function before execution. This function is termed as Module Wrapper Function. Refer

https://nodejs.org/api/modules.html#modules_the_module_wrapper for official documentation.
Before a module's code is executed, NodeJS wraps it with a function wrapper that has the following structure:

```
(function (exports, require, module, __filename, __dirname) {
  //module code
});
```

```
const fs =require("fs");
(function (exports, require, module, __filename, __dirname){
    const a = "ASHU";
    console.log(a);
    module.exports=a;
});
```

NODE.JS WEB SERVER

To access web pages of any web applications, you need a web server. The web server will handle all the http requests for the application.

e.g IIS is a web server for ASP.NET web application and Apache is a web server for PHP and Java web applications.

Node.js provides capabilities to create your own web server which wil handle HTTP requests asynchronously .You can use IIS or Apache to run Node.js web application but It is recommended to use Node.js web server.

Note = Creating own server in Node.js we didn.t need of third parties application to install or require.

```
// The http.createServer() method includes request and response
parameters which is supplise req and res
//The request object can be use to get information about the
current HTTP request.
//e.g url, request, header, and data.
//The response  object can be used to send a response for a
current HTTP request.
```

```
//If the reponse from the HTTP servebr is supposed to be
displayed as HTML.
//You should include an HTTP header with the correct content
type;
```

CODE:-CREATING SERVER...

```
const http = require("http");
const server = http.createServer((req,res)=>{
    res.end("HELLO FROM OTHER SIDE KAYSE HO...");
});

server.listen(9001,"127.0.0.1",()=>{
    console.log("I am listining on the port no 9000");
});
```

OUTPUT :-

NODE.JS ROUTING……

Basic routing example:-

```
const http = require("http");
const url  = require("url");

const server =http.createServer((req,res)=>{
    if(req.url == "/"){
        res.end("kayse ho ham he home side");
    }else if(req.url == "/about"){
        res.end("kayse ho ham about side");
    }else if(req.url == "/contact"){
        res.end("kayse ho contact side");
```

```
    }else{
        res.end("404 error found");
    }
});

server.listen(8993,"127.0.0.1",()=>{
    console.log("ME SUN RHA HU IS NO PE 8993");
});
```

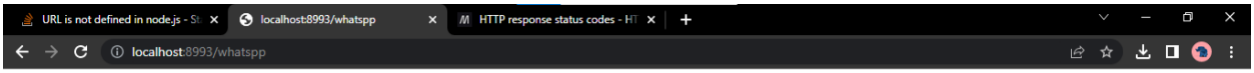
Now,we perform details routing like about 404 error.

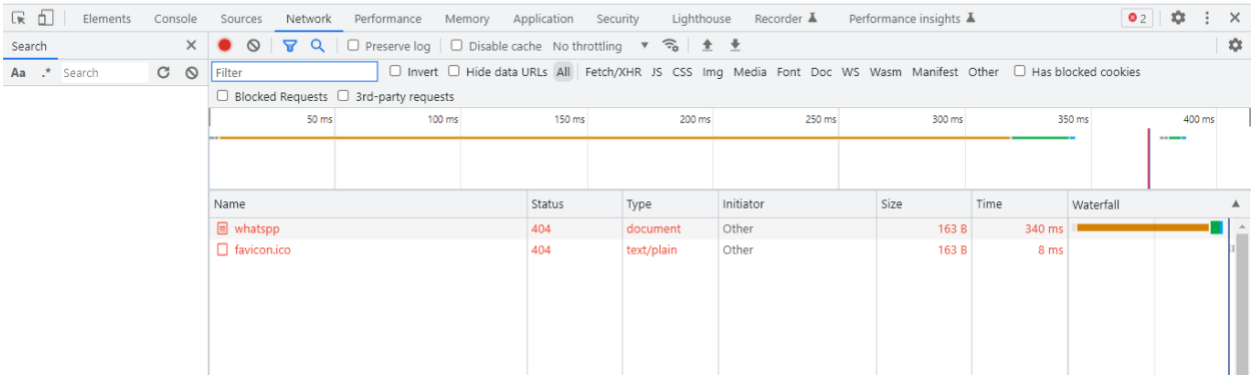
HTTP response status codes

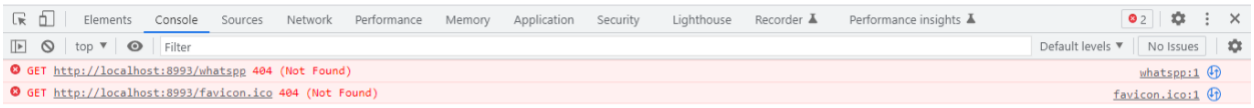
HTTP response status codes indicate whether a specific HTTP request has been successfully completed. Responses are grouped in five classes:

1. Informational responses (`100` – `199`)
2. Successful responses (`200` – `299`)
3. Redirection messages (`300` – `399`)
4. Client error responses (`400` – `499`)
5. Server error responses (`500` – `599`)

Note = Res.write() method se bhi likh sakte he but your response send nahi hoga tab tak res.end() execute nhi hota….







Perform routing in details……

```
//const chalk = require('chalk');
const http = require("http");
const url  = require("url");

const server =http.createServer((req,res)=>{
    if(req.url == "/"){
        res.end("kayse ho ham he home side");
    }else if(req.url == "/about"){
        res.end("kayse ho ham about side");
    }else if(req.url == "/contact"){
        res.end("kayse ho contact side");
    }else{
        res.writeHead(404,{"content-type":"text/html"});
        res.end("<h1>404 error found</h1>");
    }
});
```

```
server.listen(8993,"127.0.0.1",()=>{
    console.log("ME SUN RHA HU IS NO PE 8993");
});
```

- **JSON FORMAT BASIS:-CONVERSION**

```
const data = {
    name : "sujal",
    lname : "dingankar",
    mname : "sandeep",
};

console.log(data);

// convert obj to JSON format.
const jsonData =JSON.stringify(data);
console.log(jsonData.name);

// convert JSON to obj format.
const tobj = JSON.parse(jsonData);
console.log(tobj.name);
```

pratice Set 1 solution : -

```
const fs = require("fs");
const chalk = require("chalk");
const data = {
    name : "sujal",
    lname : "dingankar",
    mname : "sandeep",
};
console.log(data);

const toJson =JSON.stringify(data);
console.log(`${chalk.red.inverse(toJson)}`);

fs.writeFile("doc.txt",toJson,(err)=>{
```

```
    console.log(err);
});
fs.readFile("doc.txt","utf-8",(err,data)=>{
    console.log(data);
    //console.log(err);
    console.log(JSON.parse(data));
    //console.log(`${chalk.blue.inverse(tObj)}`);
});
```

Creating First API in Node.js……..

```
const fs = require("fs");
const url = require("url");
const http = require("http");
const { dirname } = require("path");
const server = http.createServer((req,res)=>{
    if(req.url == "/"){
        res.end("Hello from home side");
    }else if(req.url == "/about"){
        res.end("Hello from about side");
    }else if(req.url == "/contact"){
        res.end("Hello from contact side");
    } else if (req.url == "/userapi") {
        fs.readFile(`${__dirname}/userAPI/userapi.json`,"utf-
8",(err,data)=>{
            console.log(data);
            const toObj = JSON.parse(data);
            res.end(toObj[0].include.type);
        });
    }else{
        res.writeHead(404,{"content-type":"text/html"});
        res.end("<h1>404 error found</h1>")
    }
});
```

```
server.listen(9888,"127.0.0.1",()=>{
    console.log("I am lisitning from port no 9888");
});
```

Sample of data for performing api…

```
{
    "data": [
        {
            "type": "articles",
            "id": "1",
            "attributes": {
                "title": "JSON:API paints my bikeshed!",
                "body": "The shortest article. Ever.",
                "created": "2015-05-22T14:56:29.000Z",
                "updated": "2015-05-22T14:56:28.000Z"
            },
            "relationships": {
                "author": {
                    "data": {
                        "id": "42",
                        "type": "people"
                    }
                }
            }
        }
    ],
    "included": [
        {
            "type": "people",
            "id": "42",
            "attributes": {
                "name": "John",
                "age": 80,
                "gender": "male"
            }
```

```
        }
    ]
}
```

Note = When we working with html.data you have to metion {“conten-type”:”text/html”}.

When we working with json.data you have to mention {“content-type”:”application/json”}.

```
const fs = require("fs");
const url = require("url");
const http = require("http");
const { dirname } = require("path");
const server = http.createServer((req,res)=>{

    const data =
fs.readFileSync(`${__dirname}/userAPI/userapi.json`);
        //console.log(data);
        const data1 = data.toString();
        const toObj = JSON.parse(data1);

        if(req.url == "/"){
        res.end("Hello from home side");
    }else if(req.url == "/about"){
        res.end("Hello from about side");
    }else if(req.url == "/contact"){
        res.end("Hello from contact side");
    } else if (req.url == "/userapi") {
        res.writeHead(200,{"content-type":"application/json"});
        res.end(toObj);
    }
    else{
        res.writeHead(404,{"content-type":"text/html"});
        res.end("<h1>404 error found</h1>");
    }
});
```

```
server.listen(9888,"127.0.0.1",()=>{
    console.log("I am lisitning from port no 9888");
});
```