Node JavaScript.

Node Js is a Backend Language Which is Run a Server For You.

Node Js is a JavaScript RunTime Environment that Execute JavaScript Code Outside a Web Browser.

Introduction To 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.**

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, asynchronously programming, which is very memory efficient.

In Node Js REPL Concept Are Include In Which Where Data Are Read Then Evaluate Then Print Next Come In Loops That's It These Are Concept of REPL and We Use Node Js as RealTime JavaScript Excuter & Nothing Else. Here We Study About Node Js Server

- **Backend Node.js Modules With Examples. Here We Know How We Use Node Js External Modules.**

Modules is a Externals Code When we Wants to Use Then We Import in Our Projects.

```
Modules is a External Code When we Wants to Use it
Then we Import it in Our Pojects.

// This is How We Read File
// const fs = require("fs");
// const text = fs.readFileSync("Satya.txt", "utf-
8");
// console.log("The Content of File is")
// console.log(text);

//When we Want to Edit Our File Then How we Do.
const fs = require("fs");
let text = fs.readFileSync("Satya.txt", "utf-8");
console.log("The Content of File is")
console.log(text);

text = text.replace("Trail", "Ultra Trail");
console.log(text);


console.log("Creating a New File")
fs.writeFileSync("Misha.txt",text);
let text1 = fs.readFileSync("Misha.txt", "utf-8");
```

```
console.log(text1)

//NPM IS USE TO Install Third Party Modules in Node
.js
```

- **Here We Know About How To Serve HTML File In Our Server.**

```
// console.log("Hello World");


//Here We Require Http Module So We Import Http Module.

const http = require('http');

//Here We Make Two Const Variable Which is Name is hostname & port.
const hostname = '127.0.0.1';
const port = 3000;

//Here We Create Server Http Take Two Arugment Which is Request & Respose &
Here We Use Arrow Function StatusCode 200 Means All is OK Here res.end is
Endres in Which Give Our Html Code To Serve.
const server = http.createServer((req, res) => {
    res.statusCode = 200;
    res.setHeader('Content-Type', 'text/html');
    res.end(` Here We Add Our Whole HTML Code.`);
});

//Here We Make A Function  in Which Server Listen That Our Port No. & Hostname
Which is Give in Front.
server.listen(port, hostname, () => {
    console.log(`Server running at http://${hostname}:${port}/`);
});

//If You Want To Run Your Server Then Type This in Your Console node Filename.
js
// Click  Double Tab to See All Variables In NodeJs.
// REPL is Read Evaluate Print & Loop Thats Clear.
```

- **Blocking & Unblocking Execution In Node js.**

```
// Synchronous or blocking
// - line by line execution
//Here All Code Execute Line By Line If Any Code Does’t Execute Then It Not Execute Other Code Untill It Become Free That’s It.


// Asynchronous or non-blocking
// - line by line execution not guaranteed
// Here Also All Codes are Execute Line By Line But If Any Code Does’t Execute Then It Not Wait For Run This Code It Skip That Code & Run Other Code. If Any Code Take 10 Mintus For Execution Then It Not Wait For 10 Mintus It Run Other Code After 10 Mintus It Run Automically.
// - callbacks will fire


const fs = require("fs");
fs.readFile("Satya.txt", "utf-8", (err, data) => {
    console.log(data);
});
console.log("This is a message");
```

- **Serving HTML Files Using Node js**

```
//Here We Import These All Modules As We Know.
const http = require('http');
const fs = require('fs');
//Here We Make File Variable In Which We Read A Using Fs Module. We Read Index.html File.
const File = fs.readFileSync('Index.html')


//Here We Create a Server & Write It On Our Server To Serve Our File In Our Server.

const server = http.createServer((req, res) => {
    res.writeHead(200,{'Content-type':'text/html'});
    res.end(File)
})

//Here We Tell To Our Server Where We Want to Run Our Server Which Is Port No. & Hostname.
server.listen(80,'127.0.0.1', () => {
    console.log(`Server running at http://127.0.0.1:80/`);
});
```

- # Create Our Custom Backend

Here We Also Need Index.html,about.html & services.html We Make These All Folder in Our File Check Out It.

```
//Here We Import These Module As We Know.
const http = require('http');
const fs = require('fs');

//Here We Create const Variables As We Know. & Here We Also Read All File.
Like Index.html,about.html & services.html.
const hostname = '127.0.0.1';
const port = 3000;
const home = fs.readFileSync('Index.html')
const about = fs.readFileSync('./about.html')
const services = fs.readFileSync('./services.html')

//Here We Create Server As We Know About it.
const server = http.createServer((req, res) => {
    console.log(req.url);
    url = req.url;

    res.statusCode = 200;
    res.setHeader('Content-Type', 'text/html');

//Here We Redirect Link According To Need. We Understand All Url. But We Use
Express Module For Redirect These All Links.

    if (url == '/') {
        res.end(home);
    }
    else if (url == '/about') {
        res.end(about);
    }
    else if (url == '/services') {
        res.end(services);
    }
    else {
        res .statusCode = 404;
        res.end("<h1>404 not found</h1>");
    }
});

//Here We Tell To Our Server Where We Want to Run Our Server Which Is Port No.
& Hostname.
server.listen(port, hostname, () => {
    console.log(`Server running at http://${hostname}:${port}/`);
});
```

- **Create Custom Module Using Node Js**

Here We Make Two File Which is Mod.js & Index.js.

Index.js Code =

```
// console.log(average([3,4]))
// const average = require("./mod");
//Here We Import Our Mod Module Which is Made By User Or Me.
const mod = require("./mod");

//In This Module We Use These All The Function.
console.log(mod.repo)
console.log(mod.name)
console.log(mod.avg([3,4]))

console.log("This is index.js");
```

Mod.js Code =

```
console.log("This is Mod.js")

console.log("This is module");

function average(arr) {
    let sum = 0;
    arr.forEach(element => {
        sum += element;
    });
    return sum / arr.length;
}

//Here Finally Export of This Modules.
module.exports = {
    avg: average,
    name: "Harry",
    repo: "GitHub"
}

module.exports.name = "Harry";
```

- NPM Node Package Manager

Here 1st We Make Our App Which is app.js Then Open Terminal & Run npm init in Terminal This is Because These Command Download All The Module Which Is Use In Our Project.

Note - If We Want To Download Any Module Then npm install Module Name.

Now Download Express Module to Redirect url Open Terminal & Type npm install express

App.js

```
console.log("This is tutorial 69.0.0")
// 1.0.0
// 1.0.1
// 1.1.0
// 2.0.0
// node --version
// npm --version
```

Check Out All File In Node js Folder.

Package.json

Use `npm install <pkg>` afterwards to install a package and save it as a dependency in the package.json file.

Here All Module We Install Which is Show Here.

```
{
  "name": "node-js",
  "version": "1.0.0",
  "description": "Package_Manager",
  "main": "Blocking_Non-Blocking.js",
  "scripts": {
```

```
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "Package"
  ],
  "author": "Satya",
  "license": "ISC",
  "dependencies": {
    "browserify": "^16.5.1",
    "express": "^4.17.1",
    "slugify": "^1.4.4"
  },
  "devDependencies": {
    "nodemon": "^2.0.4"
  }
}
```

- **Installing Express & Postman In Node Js.**

Open ExpressNode Folder In Which Write Our Code Here We Only Discuss Code.

Open Terminal>Npm install express

```
Express is Web Application FrameWork For
Route Your URL or Page Like /contact /Home
Etc..
```

- **Now Lets Talk About Postman Here Postman is Use For**

Postman is a popular API client that makes it easy for developers to create, share, test and document APIs. This is done by allowing users to create and save simple and complex HTTP/s requests, as well as read their responses.

Install Postman From Google And You Can Check Your Link Which is Response or Not You Can Do Get, Post And Many Request According to Your Need.

- ## Writing Our 1st App Using Express in Node js.

1st Create Folder Then Open Terminal >npm init >npm Install Express. Then It Make node_module File, Package.json & Package.lock.json.

```
// Express is Web Application FrameWork For Route Your URL or Page Like /contact /Home
 Etc..
// req is Request res is Response

const express = require("express");
const app = express();
const port = 80;

app.get("/", (req, res) => {
    res.status(200).send("This is homepage of my first express app with Satya"
);
});

app.get("/about", (req, res) => {
    res.send("This is about page of my first express app with Satya");
});

app.post("/about", (req, res) => {
    res.send("This is a post request about page of my first express app with S
atya");
});
app.get("/this", (req, res) => {
    res.status(404).send("This page is not found on my website cwh");
});

app.listen(port, () => {
    console.log(`The application started successfully on port ${port}`);
});
```

- **Static Files & Pug Templates Engine.**

Lets Talk About Capture Static Files In Our Server & Also Know About Templates Engine.

A template engine enables you to use static template files in your application. At runtime, the template engine replaces variables in a template file with actual values, and transforms the template into an HTML file sent to the client.

Here We Use Pug Template Engine

Pug is a template engine for Node and for the browser. It compiles to HTML and has a simplified syntax, which can make you more productive and your code more readable. Pug makes it easy both to write reusable HTML, as well as to render data pulled from a database.

Here We Make Open It In Vs Code<Terminal<npm install Express & Pug.

And Make Static Folder For Static Files & views Folder For Pug or Html Files You Can See It Node Js Folder & Understand.

```
// Express is Web Application FrameWork For Route Your URL or Page Like
/contact /Home Etc..
// req is Request res is Response

const express = require("express");
const app = express();
const path = require("path");
const port = 80;

// For Serving static files
app.use('/static', express.static('static'))

// Set the template engine as pug
```

```
app.set('view engine', 'pug')

// Set the views directory
app.set('views', path.join(__dirname, 'views'))

// Our pug demo endpoint
app.get("/demo", (req, res) => {
    res.status(200).render('demo', { title: 'Hey Harry', message: 'Hello there
 and thanks for telling me how to use pubG!' })
});

app.get("/", (req, res) => {
    res.status(200).send("This is homepage of my first express app with Harry"
);
});

app.get("/about", (req, res) => {
    res.send("This is about page of my first express app with Harry");
});

app.post("/about", (req, res) => {
    res.send("This is a post request about page of my first express app with H
arry");
});
app.get("/this", (req, res) => {
    res.status(404).send("This page is not found on my website cws");
});

app.listen(port, () => {
    console.log(`The application started successfully on port ${port}`);
});
```

For See Static File

localhost:port/static/filename.

- **Using HTML in Pug Templates Engine.**

We Make A Folder Pug Express Check out This On In Nodejs Folder.

About nodemon Module it is Module in which If We Make Any Change in Our Server Then It Automically Save Your Change & Restart Your Server. So If You Want To Use Nodemon Then >Terminal>nodemon serverfile.js & Run.

Here Also We Make Folder As It is. Static Folder For Static Files. & views Folder For Pug Files As We Know.

```
const express = require("express");
const path = require("path");
const fs = require("fs");
const app = express();
const port = 80;

// EXPRESS SPECIFIC STUFF
app.use('/static', express.static('static')) // For serving static files
app.use(express.urlencoded())

// PUG SPECIFIC STUFF
app.set('view engine', 'pug') // Set the template engine as pug
app.set('views', path.join(__dirname, 'views')) // Set the views directory

// ENDPOINTS
app.get('/', (req, res) => {
    const con = "This is the best content on the internet so far so use it wis
ely"
    const params = { 'title': 'Gym Form!', "content": con }
    res.status(200).render('index.pug', params);
})

app.post('/', (req, res) => {
    name = req.body.name
    age = req.body.age
    gender = req.body.gender
    address = req.body.address
    more = req.body.more

    let outputToWrite = `the name of the client is ${name}, ${age} years old,
${gender}, residing at ${address}. More about him/her: ${more}`
    fs.writeFileSync('output.txt', outputToWrite)
    const params = { 'message': 'Your form has been submitted successfully' }
    res.status(200).render('index.pug', params);
```

```
})


// START THE SERVER
app.listen(port, () => {
    console.log(`The application started successfully on port ${port}`);
});
```

Our index.pug File =

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- <title>Using plain html in Pug</title> -->
    <title>#{title}</title>

    style
        include ../static/style.css

</head>
<body>
    <nav>
        <ul>
            <li><a href="/">Home</a></li>
            <li><a href="/">About</a></li>
            <li><a href="/">Contact Us</a></li>
        </ul>
    </nav>
    <div class="container">

        <h2>Get this gym membership for 60$ - Fill this form now!</h2>
        <!-- <p>
            | #{content}
            | This is a plain html using pug
        </p> -->

        <form action="/" method="post" id="contact">
            <input type="text" name="name" id="name" placeholder="Enter your
                name">
            <input type="text" name="age" id="age" placeholder="Enter your age
                ">
            <input type="text" name="gender" id="gender" placeholder="Enter yo
                ur gender">
```

```
            <textarea name="address" id="address" cols="30" rows="10" placehol
               der="Enter your address"></textarea>
            <textarea name="more" id="more" cols="30" rows="10" placeholder="M
                ore about you.."></textarea>
            <button class="btn">Submit</button>
        </form>
    </div>
</body>
</html>
```

& You Can Make Pug File According To Your Need.

Now We Make A Dance Website Using Node js You Can Check Out Folder and See All Code & Understand. That's It For Node js.

Dance_website_nodejs_pug.