https://shorturl.at/kAHNV

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Introduction to Node.js**

**Modules in Node.js**

**Http server**

**Http get parameters**

**Http post parameters**

**Express**

**Express get parameters**

**Express post parameters**

**Express URL parameters**

**Imports and Exports in express**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

=======================================================

Node.js

=======================================================

Introduction:-

- Node.JS is open source, cross platform, Javascript runtime environment.

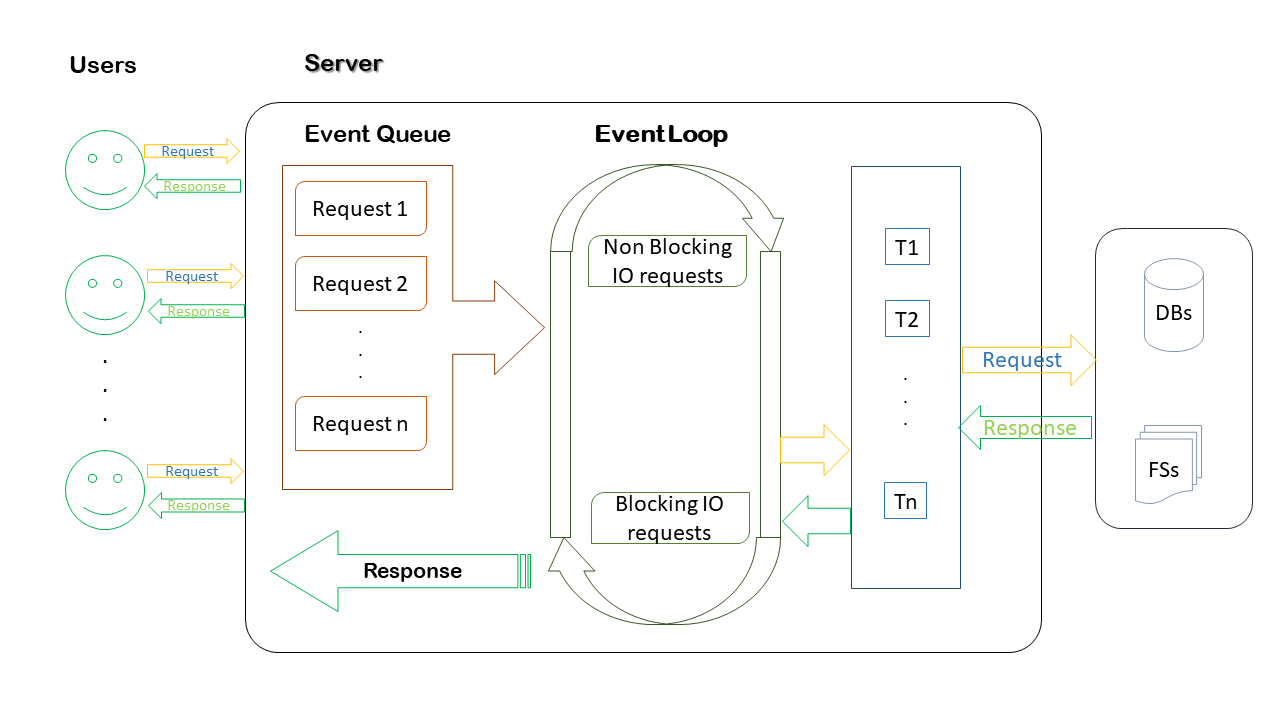
- Node applications can be developed using either Javascript or Typescript.

- Node.JS was released by Ryan Dahl on 27th May 2009, at netscape.

- Latest version of Node.JS is 21.6.1 22-01-2024

- Current stable version is 20.11.0 09-01-2024

- The applications (servers) developed by Node.JS are called 'Single Threaded Event Loop' applications.



# **Modules in NodeJS**

• Node supports modules

• Predefined modules

• Custom modules

# **http**

• This is the native module.

• This module is available along with 'Node Engine'.

• This module is used to develop http servers.

# **url**

• This is native module.

• This module is used to read get parameters in http servers.

# **query-string**

• This is native module.

• This module is used to read the post parameters in http servers.

# **fs**

• This is native module.

• fs stands for 'File system'.

• This module is used to interact with flat files.

• Eg:- txt, xml, json, etc

# **express**

• This is third party module.

• This module is used to develop 'ReST APIs' (web services.)

• ReST API:- Representational State Transfer Application Programing Interface.

# **mysql**

• This is third party module.

• This module is used to interact with My Sql database.

•

# **mongodb**

• This is third party module.

• This module is used to interact with mongodb without schema.

• [Note:- rules and regulations of db are called as schema].

# **mongoose**

• This is third party module.

• This module is used to interact with mongodb with schema.

# **mssql**

• This is third party module.

• This module is used to interact SQL Server.

# **multer**

• This is third party module.

• This module is used to upload images to server.

# **socket.io**

• This is third party module.

• This module is used to develop chat applications.

# **jwt-simple**

• This is third party module.

• This module is used to generate tokens for authentication purpose.

• This system is technically called as Token based authentication system. •

# **body-parser**

• This is third party module.

• This module is used to set MIME type.

# **cluster**

• This is third party module.

• This module is used to implement child process in http server.

• implementing child process is called as load balancing.

# **express-cluster**

• This third party module.

• This module is used to implement load balancing in ReST APIs.

# **cookie-parser**

• This is third party module.

• This module is used to work with cookies.

- We can download all third party modules by using either 'npm' or 'yarn' tool.

- npm stands for 'Node Packaging Manager'.

- npm is an integrated tool for NodeJS.

- yarn is the latest tool used to download 'Node modules'

- yarn is faster as compared to npm.

- All node modules will be downloaded to 'node\_modules' folder in current path.

- we can start node server in 3 ways

>node server

>npm start

>nodemon server (monitoring / watch mode)

Environmental setup

1. Download and install nodejs

https://nodejs.org/en/

2. Download and install git

https://git-scm.com/

3. Download and install VSCode

https://code.visualstudio.com/

4. Download and install postman

https://www.postman.com/downloads/

5. Install yarn tool using following command

>npm install -g yarn

npm :- node packaging manager

-g :- global installation

- Create a folder rename it as 'NodeJS'

- in that folder create one more folder 'modules eg'

- open 'modules eg' folder in VSCode

- create server.js file there

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Download following modules using yarn tool

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. express

2. mysql

3. mongodb@2.2.32

4. multer

5. jwt-simple

>npm init -y

1. express >yarn add express --save

2. mysql >yarn add mysql --save

3. mongodb@2.2.32 >yarn add mongodb@2.2.32 --save

4. multer

5. jwt-simple

>yarn add multer jwt-simple --save

=======================================================

Implementing HTTP Server

=======================================================

- ‘http’ is the predefined module used to create http servers.

- http is the native module, so no need to download it.

- ‘require()’ is used to import.

- Eg let http = require(‘http’)

- ‘createServer()’ is the predefined function in http module.

- This function is used to create the server.

- The argument to createServer() is arrow function.

- To this arrow function there are two arguments, ‘req’ and ‘res’.

- request and response objects provided by node engine respectively.

- req object is used to store client data.

- res object is used to send response to client.

- 'writeHead(-,-)' is the predefined function in res object.

- This function is used to set the MIME type.

- First argument to this function is the status code (200 - ok).

- Second argument is the JSON object.

- JSON key is ‘content-type’ and the value is ‘text/html’.

- ‘write(-)’ is the predefined function in res object.

- This function is used to append response to res object.

- ‘end()’ is the predefined function in res object.

- This function is used to lock the response.

//import http module

let http = require('http')

let server = http.createServer((req, res) => {

//set MIME type

res.writeHead(200, { 'content-type': 'text/html' })

res.write('<h1>Welcome to http server</h1>')

res.end()

})

//assign port no

server.listen(8080)

console.log('Server listening port no 8080')

/\*

start server

>node server

url http://localhost:8080

\*/

=======================================================

HTTP get parameters

=======================================================

- 'url' is the predefined module in node.

- url module is used to read get parameters in http server.

\*\*\*server.js\*\*\*

//import http module

let http = require('http')

//import url module

let url = require('url')

//create server

let server = http.createServer((req, res) => {

let obj = url.parse(req.url, true).query

let uname = obj.uname

let upwd = obj.upwd

//set MIME type

res.writeHead(200, { 'content-type': 'text/html' })

if (uname === 'admin' && upwd === 'admin')

res.write("<h1> Login Success </h1>")

else

res.write("<h1> Login Failed </h1>")

res.end()

})

//assign port no

server.listen(8080)

console.log("Server Listening port no 8080")

//url :- http://localhost:8080/?uname=admin&upwd=admin

\*\*\*index.html\*\*\*

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="style.css">

</head>

<body>

<form action="http://localhost:8080" method="get" class="box">

<h1>Login</h1>

<input type="text" placeholder="Username" name="uname">

<input type="password" placeholder="Password" name="upwd">

<input type="submit" value="Login">

</form>

</body>

</html>

\*\*\*style.css\*\*\*

h1 {

color: white;

text-transform: uppercase;

font-weight: normal;

}

body {

background: radial-gradient(white, black);

font-family: sans-serif;

}

.box {

background-color: black;

width: 300px;

margin: 50px auto;

padding: 40px;

border-radius: 20px;

text-align: center;

}

input {

margin: 20px auto;

text-align: center;

padding: 14px 10px;

width: 200px;

border-radius: 24px;

background: none;

}

input[type="text"],

input[type="password"] {

border: 2px solid skyblue;

color: lightyellow;

}

input[type="submit"]

{

border: 2px solid burlywood;

color: white;

cursor: pointer;

}

=================================================

HTTP post parameters

=================================================

- 'querystring' is the predefined module in nodejs.

- querystring module is used to read post parameters in http server

\*\*\*server.js\*\*\*

//import modules

let http = require('http')

let qs = require('querystring')

//create server

let server = http.createServer((req, res) => {

//set MIME type

res.writeHead(200, { 'content-type': 'text/html' })

let body = ''

//listen post parameters

req.on("data", (result) => {

body = body + result

})

//end callback to node engine

req.on("end", () => {

let obj = qs.parse(body)

//read post parameters

let uname = obj.uname

let upwd = obj.upwd

if (uname === 'admin' && upwd === 'admin')

res.write("<h1 style = 'color:Green'>Login Success</h1>")

else

res.write("<h1 style = 'color:red'>Login Failed</h1>")

res.end()

})

})

//assign port no

server.listen(8080)

console.log('Server listening port no 8080')

\*\*\*index.html\*\*\*

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="style.css">

</head>

<body>

<form action="http://localhost:8080" method="post" class="box">

<h1>Login</h1>

<input type="text" placeholder="Username" name="uname">

<input type="password" placeholder="Password" name="upwd">

<input type="submit" value="Login">

</form>

</body>

</html>

\*\*\*style.css\*\*\*

h1 {

color: white;

text-transform: uppercase;

font-weight: normal;

}

body {

background: radial-gradient(white, black);

font-family: sans-serif;

}

.box {

background-color: black;

width: 300px;

margin: 50px auto;

padding: 40px;

border-radius: 20px;

text-align: center;

}

input {

margin: 20px auto;

text-align: center;

padding: 14px 10px;

width: 200px;

border-radius: 24px;

background: none;

}

input[type="text"],

input[type="password"] {

border: 2px solid skyblue;

color: lightyellow;

}

input[type="submit"]

{

border: 2px solid burlywood;

color: white;

cursor: pointer;

}

=================================================

Express

=================================================

- Download express module

> yarn add express --save

//initialyse project

//>npm init -y

//download express module

//>yarn add express --save

//import express module

let express = require('express')

//create rest object

let app = express() //where app is rest object

//create get request

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

console.log('Default get message')

res.json({ 'message': 'default get request' })

})

//create one more get request

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

res.send({ 'message': 'fetch get request' })

})

//create post request

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

res.send({ 'message': 'default post request' })

})

//create one more post request

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

res.send({ 'message': 'login post requrest' })

})

//create a port

let port = 8080

//assign port no

app.listen(port, () => {

console.log('Server listening port no ', port)

})

/\*

Test urls with postman

http://localhost:8080 Default GET

http://localhost:8080/fetch fetch GET

http://localhost:8080 Default POST

http://localhost:8080/login login POST

\*/

=================================================

Reading get parameters in express

=================================================

//url :- http://localhost:8080/login/?uname=admin&upwd=admin

//initialyse project

//>npm init -y

//download express module

//>yarn add express --save

//import express module

let express = require('express')

//create rest object

let app = express()

//create port

let port = 8080

//create rest api

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

//query is the predefined key in req object

//query is used to read get parameters

let uname = req.query.uname

let upwd = req.query.upwd

if (uname === 'admin' && upwd === 'admin')

res.send({ 'login': 'success' })

else

res.send({ 'login': 'failed' })

})

app.listen(port, () => {

console.log('server listening port no ', port)

})

=================================================

Reading Express Post parameters

=================================================

Download express and body-parser modules

//initialyse project

//>npm init -y

//Download express and body-parser modules

//>yarn add express body-parser --save

//import modules

let express = require('express')

let bodyparser = require('body-parser')

//create rest object

let app = express()

//create port

let port = 8080

//set JSON as MIME type

app.use(bodyparser.json())

//front end encoding data

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

//create rest api

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

//client parameters are stored in body part of request

let uname = req.body.uname

let upwd = req.body.upwd

if (uname == 'admin' && upwd == 'admin')

res.send({ 'login': "success" })

else

res.send({ 'login': 'failed' })

})

//assign port no

app.listen(port, () => {

console.log('Server listeing port no ', port)

})

/\*

Start server

>node server

Test Rest api in postman with url

http://localhost:8080/login

in postman

1. req -> post

body -> raw

-> text <-> json

{"uname":"admin","upwd":"admin"}

2. req -> post

body -> x-www...

-> key and values

\*/

=================================================

Reading url parameters from express

=================================================

//initialyse project

//>npm init -y

//download express module

//>yarn add express --save

//import express module

let express = require('express')

//create rest object

let app = express()

//create port

let port = 8080

//url:http://localhost:8080/login/admin/admin

app.get("/login/:uname/:upwd", (req, res) => {

//params is the predefined key used to read parameters from url

let uname = req.params.uname

let upwd = req.params.upwd

if (uname === 'admin' && upwd === 'admin')

res.json({ 'login': 'success' })

else

res.json({ 'login': 'faild' })

})

app.listen(port, () => {

console.log(`Server listeing port no ${port}`)

})

Imports and exports

- module is the predefined object in node.

- exports is the predefined key in module object.

- exports key is used to export (JSON object or function)

=================================================

Exporting and importing JSON object

=================================================

Eg01

<>

config

- db\_config.js

- server.js

\*\*\*db\_config.js\*\*\*

module.exports = {

"host": "localhost",

"user": "root",

"password": "root",

"database": "nodedb",

"table": "products"

}

\*\*\*server.js\*\*\*

//initialyse project

//>npm init -y

//download express module

//>yarn add express --save

//import express module

let express = require('express')

//create rest object

let app = express()

//create port

let port = 8080

//import db\_config

let obj = require("./config/db\_config")

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

res.json(obj)

})

//assign port no

app.listen(port, () => {

console.log('Server listening port no ', port)

})

Eg02

<>

config

- db\_connection.js

- server.js

\*\*\*db\_connection.js\*\*\*

module.exports = {

oracle: () => { return 'Oracle connection soon....!' },

mysql: () => { return 'mysql connection soon...!' },

mongodb: () => { return 'mongodb connection soon...!' },

mssql: () => { return 'sql server connection soon...!' }

}

\*\*\*server.js\*\*\*

//initialyse project

//>npm init -y

//download express module

//>yarn add express --save

//import express module

let express = require('express')

//create rest obj

let app = express()

//import db\_connection

let obj = require('./config/db\_connection')

//create rest api

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

res.send(obj.oracle())

})

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

res.send(obj.mongodb())

})

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

res.send(obj.mysql())

})

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

res.send(obj.mssql())

})

//assign port no

app.listen(8080)

console.log('Server listening port no 8080')

/\*

http://localhost:8080/oracle

http://localhost:8080/mongodb

http://localhost:8080/mssql

http://localhost:8080/mysql

\*/

=================================================

Function Import and Export

=================================================

Eg03

<>

config

- my\_fun.js

- server.js

\*\*\*my\_fun.js\*\*\*

module.exports = (arg1, arg2) => {

if (arg1 == 'admin' && arg2 == 'admin')

return 'Login Success'

else

return 'Login Failed'

}

\*\*\*server.js\*\*\*

//initialyse project

//>npm init -y

//download express module

//>yarn add express --save

//import express module

let express = require('express')

//create rest object

let app = express()

//import my\_fun

let my\_fun = require('./config/my\_fun')

//create rest api

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

res.send(my\_fun(req.query.uname, req.query.upwd))

})

//assign port no

app.listen(8080)

console.log("Server listening port no 8080")

//url http://localhost:8080/login?uname=admin&upwd=admin

================================================================

================================================================