Text Instructions: Module 7  
  
  
GitHub Link for Node.js:  
  
<https://github.com/Apollo-Level2-Web-Dev/batch4-learning-node>

GitHub Link for Express.js:  
  
<https://github.com/Apollo-Level2-Web-Dev/batch4-project-1>

7-1 What is Nodejs , a high level overview of node.js

7-2 What is module, commonjs vs es

Create: index.js , local-1.js

index.js:

node .\local-1.js

{

id: '.',

path: 'E:\\web\\Programming Hero\\Level 2\\Mission 01-Be A Typescript Technocrat\\Module 7-Mastering The Foundation of Express\\learning-node',

exports: {},

filename: 'E:\\web\\Programming Hero\\Level 2\\Mission 01-Be A Typescript Technocrat\\Module 7-Mastering The Foundation of Express\\learning-node\\local-1.js',

loaded: false,

children: [],

paths: [

'E:\\web\\Programming Hero\\Level 2\\Mission 01-Be A Typescript Technocrat\\Module 7-Mastering The Foundation of Express\\learning-node\\node\_modules',

'E:\\web\\Programming Hero\\Level 2\\Mission 01-Be A Typescript Technocrat\\Module 7-Mastering The Foundation of Express\\node\_modules',

'E:\\web\\Programming Hero\\Level 2\\Mission 01-Be A Typescript Technocrat\\node\_modules',

'E:\\web\\Programming Hero\\Level 2\\node\_modules',

'E:\\web\\Programming Hero\\node\_modules',

'E:\\web\\node\_modules',

'E:\\node\_modules'

],

[Symbol(kIsMainSymbol)]: true,

[Symbol(kIsCachedByESMLoader)]: false,

[Symbol(kIsExecuting)]: true

}

Index.js:

const add = require("./local-1")

console.log(add(2, 3));

Now,

Local-1.js:

const add = (param1, param2) => param1+ param2

const a = 10;

module.exports = {

    a,

    add

}

// console.log(module);

Index.js:

const add = require("./local-1")

console.log(add); // { a: 10, add: [Function: add] }

const add = require("./local-1")

console.log(add.add(2,3));

const { a, add } = require("./local-1")

console.log(a);

console.log(add(2, 3));

Now,

Local-1.js, Local-2.js, index.js

Local-1.js:

const add = (param1, param2) => param1+ param2

const a = 10;

module.exports = {

    a,

    add

}

Local-2.js:

const add = (param1, param2, param3) => param1 + param2 + param3

const a = 20

module.exports = {

    a,

    add

}

Index.js:

const { a, add } = require("./local-1")

const { a, add } = require("./local-2")

console.log(a);

console.log(add(2, 3));

console.log(a2);

console.log(add2(2, 3, 7));

// error

But, Index.js:

const { a, add } = require("./local-1")

const { a:a2, add:add2 } = require("./local-2")

console.log(a);

console.log(add(2, 3));

console.log(a2);

console.log(add2(2, 3, 7));

// OK , no error

Docs : <https://nodejs.org/docs/latest/api/path.html>

**7-3 File System Module , synchronous vs asynchronous**

Google > node js . > file system > search ( Ctrl+F ): fs.readfilesync

<https://nodejs.org/docs/latest/api/fs.html>

google > lorem ipsum generator >

const fs = require('fs') // fs-module

// reading a file text

const readText = fs.readFileSync('./texts/read.txt')

console.log(readText);

output : error

Again,

const fs = require('fs') // fs-module

// reading a file text

const readText = fs.readFileSync('./texts/read.txt', 'utf-8')

console.log(readText);

output -> ok

// writing a text

const writtenText = fs.writeFileSync('./texts/write.txt', readText+ 'This is my written text')

console.log(writtenText);

search : fs.readfile

const fs = require('fs')

// reading text asynchronously

fs.readFile('./texts/read.txt', 'utf-8', (err, data) => {

    if (err) {

        throw Error('Error reading text')

    }

    // console.log(data);

    // writing text asynchronously

    fs.writeFile('./texts/read2.text', data, 'utf-8', (err) => {

        if(err){

            throw Error("Error writing data")

        }

    })

})

//console.log('testing asynchronous'); // after uncommeting this the output only this text

7-4 Event driven architecture, create your own events

Google > node js docs > Events > emitter

Docs: <https://nodejs.org/docs/latest/api/events.html#emitteremiteventname-args>

const EventEmitter = require('events')

const myEmitter = new EventEmitter()

// Listener

myEmitter.on('birthday', () => {

    console.log(`Happy Birthday To You `)

})

myEmitter.on('birthday', (gift) => {

    console.log(`I will send a ${gift} To you`)

})

myEmitter.emit('birthday', 'watch')

**7-5 Stream and buffer, create your own server**

Google > node js doc > HTTP > search (Ctrl+F) : createServer >

const http = require('http');

// creating a server using raw node.js

const server = http.createServer()

// listener

server.on('request', (req, res)=> {

    // console.log(req);

    console.log(req.url, req.method);

    res.end('Hello From World !')

})

server.listen(5000, ()=> {

    console.log(`server is listening on port 5000`)

})

Run the code , goto chrome > <http://localhost:5000>

Change the url : <http://localhost:5000/read-file>

Check the console , output

Node js docs > File System > search :fs.createReadStream >

const http = require('http');

const fs = require('fs');

const { buffer } = require('stream/consumers');

// creating a server using raw node.js

const server = http.createServer()

// listener

server.on('request', (req, res) => {

    // console.log(req);

    // console.log(req.url, req.method);

    if (req.url === '/read-file' && req.method === 'GET');

    // streaming file reading

    const readableStream = fs.createReadStream(\_\_dirname + '/texts/read.txt')

    // const readableStream = fs.createReadStream(process.cwd() +'/texts/read.txt')

    readableStream.on('data', (buffer) => {

        res.write(buffer)

    })

    readableStream.on('end', () => {

        res.end('Hello From World !')

    })

    // res.end('Hello From World !')

})

server.listen(5000, () => {

    console.log(`server is listening on port 5000`)

})

7-6 Installing express with typescript

sream-buffer.js:

const http = require('http');

const fs = require('fs');

const { buffer } = require('stream/consumers');

// creating a server using raw node.js

const server = http.createServer()

// listener

server.on('request', (req, res) => {

    // console.log(req);

    // console.log(req.url, req.method);

    if (req.url === '/read-file' && req.method === 'GET');

    // streaming file reading

    const readableStream = fs.createReadStream(\_\_dirname + '/texts/read.txt')

    // const readableStream = fs.createReadStream(\_\_dirname + '/texts/readvul.txt') // something went wrong !

    // const readableStream = fs.createReadStream(process.cwd() +'/texts/read.txt')

    readableStream.on('data', (buffer) => {

        res.statusCode = 200;

        res.write(buffer)

    })

    readableStream.on('error', (error) => {

        // res.end('Hello From World !')

        console.log(error);

        res.statusCode = 500;

        // res.end('The streaming is over !')

        res.end('something went wrong !')

    })

    // res.end('Hello From World !')

})

server.listen(5000, () => {

    console.log(`server is listening on port 5000`)

})

Google > express js > getting started > hello world >

project-1 > terminal (git bash) :

* npm install express
* npm add -D typescrip
* tsc –init

tsconfig.json > search: rootdir > "rootDir": "./src/",

outdir > "outDir": "./dist",

terminal (git bash) : npm i --save-dev @types/node

add : import express from ‘express’

npm i --save-dev @types/express

move :

app.listen(port, () => {

  console.log(`Example app listening on port ${port}`)

})

to the ‘server.ts’

app.ts

export default app;

last ,

app.ts :

import express from 'express'

// const express = require('express')

const app = express()

const port = 3000

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

  res.send('Hello World!')

})

export default app;

server.ts:

import { Server } from 'http';

import app from './app'

const PORT = 5000;

let server: Server

async function bootStrap() {

    server = app.listen(PORT, () => {

        console.log(`Example app listening on port ${PORT}`)

    })

}

bootStrap();