FUNDAMENTOS NODE

19/04/2023

O QUE SÃO MÓDULOS?

Scripts reaproveitáveis.

Internos: que nós desenvolvemos;

Core módulos: Modulos que vem no node.js;

Externos: módulos que instalamos via npm.

Aprendi argumentos;

Métodos de console.log

Minimist (modulo externo)

parseInt

require

setTimeout (limpar o console depois de um determinado time)

console.count (contagem de impressões)

impressão com vários valores em apenas um console.log

20/04/2023

Chalk – recurso que você consegue alterar as propriedades de texto quanto as cores fontes e etc..

Entrada de dados e saída – Fizemos uma pergunta e resposta com readline (pasta 2 aula 9)

const readline = require("readline").createInterface({

    input: process.stdin,

    output: process.stdout,

})

readline.question("Qual a sua linguagem preferida? ", (language) => {

    if(language === "python") {

        console.log("Isso nem é linguagem!")

    } else{

        console.log(`A minha linguagem preferida é: ${language}`)

    }

    readline.close()

})

INQUIRE

Abstração de input – trabalhando com o método inquirer 8.0.0 – programa simples com entrada e saída de resposta com calculo de média no final. Usamos o comando prompt para perguntas – Mapa lógico inquirer (prompt) > then(então – resposta) / erro, caso o código não esteja legal, usar o debugger para analisar o código.

const inquirer = require("inquirer")

inquirer

    .prompt([

        {

            name: 'p1',

            message:'Qual a Primeira nota? ',

        },

        {

            name:'p2',

            message:'Qual a segunda nota? ',

        },

])

.then((answers) => {

    console.log(answers)

    const media = ((parseInt(answers.p1) + parseInt(answers.p2)) / 2)

    console.log(`a média é: ${media}`)

})

.catch(err => console.log(err))

21/04/2023

Event loop – de forma síncrona, um programa simples para saber como o código funciona.

function a(){

    console.log("Letra A")

}

function b(){

    console.log("Letra B")

}

function C(){

    console.log("Letra C")

    a()

    b()

}

C()

Event emitter – se comporta como os eventos do navegador, um core module chamado events, cuja classe EventEmitter.

const EventEmitter = require('events')

const eventEmitter = new EventEmitter()

eventEmitter.on('start', () => {

    console.log('Durante')

})

console.log('Antes')

eventEmitter.emit('start')

console.log('Depois')

Sincrona e assíncrona

Sincrona – é uma forma de código que precisa ser executado em ordem, scripts por scripts, de maneira organizada.

const fs = require('fs')

console.log('Inicio')

fs.writeFileSync('arquivo.txt', 'oi')

console.log('Fim')

assíncrona – com ele você consegue executar os códigos de maneira com o que não precise de uma ordem logica.

const fs = require('fs')

console.log('Inicio')

fs.writeFile('arquivo.txt', 'Oi', function(err) {

    setTimeout(function(){

        console.log('Arquivo Criado')

    }, 1000);

})

Erros no Node.js

Throw e Try catch

Throw – ele encerrará logo que for identificado o código e retorna a mensagem de erro.

Try catch – é uma maneira mais amigável de mostrar os erros durante a execução do código.

23-04-2023

Introdução a Core Modules

Alguns dos modelos:

http – criação de servidores;

path – extrair informações de paths(caminhos) de arquivos;

fs: file system, leitura e escrita de arquivos;

url: modulo para trabalhar com url’s;

fizemos um sistema de roteamento simples, com paginas de home, contato, 404

fizemos as leituras fs, patch, html, url, renomear arquivo, excluir arquivo, atualização, padrões básico. Gostei bastante, falta menos um pouquinho em nome de jesus.

25-04-2023

Detalhamento do arquivo

Stats – mostra se é uma pasta, arquivo, diretório e quando foi criado, muito usado no Linux.

Saber as propriedades do arquivo

Modulo path – Core module

Extração de informações de caminhos e arquivos, nome do diretório, nome do arquivo, extensão do arquivo e etc;

Path absoluto e formar path – função JOIN e RESOLVE, saberá o caminho completo até o arquivo. FUNÇÕES IMPORTANTES.

Trabalhando com diretórios – com a função FS é possível trabalhar com diretórios (pastas);  
Metodo EXISTS poderá evidenciar se um diretório existe ou não;  
E o método MKDIR pode criar um diretório.

--

Modulo OS  
Será comandos para conhecermos melhor sobre a maquinha..  
Será para sistema operacional

Nova seção – NPM FUNDAMENTAL

O QUE É UMA NPM NODE.JS > gerenciador de pacotes.

Gerenciador que administra as funções de instalação no node.js

Módulos lodash

Instalação de módulos novamente.

Onde encontrar os módulos: npmjs.com

--------

Dependencia em desenvolvimento: npm install –save-dev

-------

Atualização de pacotes: npm update //atualizar tudo  
npm update <module> //atualizar somente o modulo desejado  
forçar e checar atualizações de pacotes. npx npm-check-updates – u

------

Criando scripts

Criação de rotina com comandos que deixamos prontos.

Instalamos scripts de execução do index.js

------

Pacote global

O pacote global não fica salvo em node\_modules e sim no servidor que estamos utilizando o projeto;  
podemos acessar de qualquer local via terminal;  
Utilizamos o comando ‘flag –g’

Npm install –g ‘lodash’  
no caso do lodash não temos como executar via global, sendo assim executamos o comando ‘npm link lodash’ e assim foi criado uma copia para que possamos utilizar.

------

Executando pacotes

Na execução de outros pacotes utilizamos o NPX   
exemplo: cowsay “Hello World!”

26-04-2023

Modulo Criação do Projeto Conta de Banco

Criação da conta, já criado com o balance: 0

27-04-2023

Conclusão do projeto bancário.

Resumão do projeto Moraes Bank  
-----

Função: process.exit() – executa o encerramento do sistema;

Função de deposito: accountData.balance = parseFloat(*amount*) + parseFloat(accountData.balance)