**Bootcamp Launchbase (Dicas e Codigos)**

**Ambiente de Desenvolvimento**

**Link da documentação:** <https://gist.github.com/maykbrito/1bb488966fa48fe016ef742321022b10>

Windows = instalando o chocolatey

Site: <https://chocolatey.org/>

Documentação do Chocolatey para instalação: <https://chocolatey.org/install>

Usar o prompt como administrador >

roda a linha de comando > Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))

Verificando se o choco está instalado > comando > choco

Instalar o VS Code, e Node.js como pacote

Site versões de SO para node: <https://nodejs.org/en/download/>

Site para instalação do node como pacote: <https://nodejs.org/en/download/package-manager/>

Documentação do Node.js: <https://nodejs.org/en/docs/>

Instalando o VS Code por linha de comando: cinst vscode -yf

Instalando o Chrome por linha de comando: cinst googlechrome -yf

Instalando o Node.js por linha de comando: cinst nodejs-lts -yf

OBS: lts e a versão instável do node, -y para não fazer perguntas e –f para força instalação.

**CONFIGURAÇÕES DO VSCODE**

Ctrl+p = Buscar aplicações criadas.

**Extensões instaladas:**

Tema: Dracula Official>Dracula

Live Server: roda a aplicação em tempo real

Material Icon Theme: coloca ícones nos arquivos e pastas criadas

Nunjucks Template: são templates para usar na configuração do nunjucks porem tem que está instalado.

Watch in Chrome: roda o projeto no chrome, caminho para usa > ctrl+shift+p = pesquisa o que tem no vs code> watch in Chrome + click começa a inicializar o projeto.

Emmet atalhos na digitação do código: ctrl+shift+p > Open Settings (JSON)>

Criando o emmet No JSON segue exemplo abaixo,

{

    "workbench.colorTheme": "Dracula",

    "workbench.iconTheme": "material-icon-theme",

    "emmet.syntaxProfiles": {

        "javascript": "jsx",

    },

    "emmet.includeLanguages": {

        "javascript": "javascriptreact",

        "njk": "html"

    },

    "window.zoomLevel": 0,

}

**INSTRODUÇÃO A PROGRAMAÇÃO WEB**

Tipos Number, Strings

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

const aluno1 = 'Henrique'

const aluno2 = 'Ana'

const nota1 = 7.5

const nota2 = 9

*console*.log(`O ${aluno1} tirou a nota ${nota1}, ja o aluno(a) ${aluno2} tirou a nota ${nota2}`)

// //typeof mostrar qual o tipo de valor é a varivel string, number, boolean

// console.log(typeof nota1)

Calculando no JS

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

const aluno1 = 'Henrique'

const aluno2 = 'Ana'

const aluno3 = 'Fulano'

const nota1 = 7.5

const nota2 = 9

const nota3 = 8

const media = (nota1 + nota2 + nota3) / 3

*console*.log(media.toFixed(2)) // toFixed (valor) diminuir as casas decimais depois do ,

Condicionais JS

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

const aluno1 = 'Henrique'

const nota1 = 7.5

const aluno2 = 'Ana'

const nota2 = 8

const aluno3 = 'Fulano'

const nota3 = 7

const media = (nota1 + nota2 + nota3) / 3

// se a media for maior que 5, parabenizar a turma

if(media > 5) {

*console*.log(`A nota foi de ${media} . Parabéns`)

} else {

*console*.log('A média e menor que 5')

}

// console.log(media <5) menor return false no caso, console.log(media > 5) maior return verdadeiro.

Operadores Lógicos, Aritméticos

/\*============================================

    OPERADORES LÓGICOS, ARITMÉTICOS

    && "E" As  duas condições devem ser verdadeiras

        para que a condição final seja verdadeira.

    ||  "OU" Uma das condições deve ser verdadeira

        para que a condição final seja verdadeira

    !   "NÂO" Nega uma condição

    console.log(5 == 5 && 6 == 6) //true

    console.log(5 == 5 && 6 != 6) //false

    console.log(5 != 5 || 6 == 6) //true

    console.log(5 == 5 || 6 != 6) //true

    console.log(!(5 > 6)) //true o que é falso transforma em verdadeiro

    console.log(!(5 < 6)) //false o que é verdadeiro transforma em falso

     OPERADORES ARITMÉTICOS

     \*      Multiplicação

     /      Divisão

     %      Resto da divisão

     +      Adição

     -      Subtração

     console.log(2 \* 2) // 4

     console.log(2 / 2) // 1

     console.log(2 % 1.5) // 0.5

     console.log(2 + 3) // 5

     console.log(2 - 2) // 0

 ============================================\*/

// Desafio 1

const idade = 18

// verificar se a pessoa é maior igual a 18 anos

// se sim, deixar entrar, se não, bloquear a entrada

// se a pessoa tiver 17 anos

// avisar para voltar quando fizer 18 anos

if(!(idade >= 18) || idade === 17) {

*console*.log('Bloquear entrada')

} else {

*console*.log('Deixar entrar')

}

Operadores Relacionais

/\*============================================

    OPERADORES COMPARAÇÃO, RELACIONAIS

    >       MAIOR

    <       MENOR

    >=      MAIOR IGUAL A

    <=      MENOR IGUAL A

    ==      IGUAL A

    ===     IGUAL E DO MESMO TIPO

    !=      DIFERENTE DE

    !==     DIFERENTE, INCLUSIVE O TIPO

    console.log(5 > 4) //true

    console.log(5 < 4) //false

    console.log(5 >= 4) //true pois 5 maior porem nao é igual

    console.log(5 <= 4) //false pois 5 não é menor que 4 nem igual

    console.log(4 <= 4) //true pois mesmo 4 nao sendo menor que 4, porem são iguais

    console.log(4 == "4") //true pois o == apens vai olhar o valor

    console.log(4 === "4") //false pois o === vai verifica se são iguais e do mesmo tipo  no caso number

    console.log(4 != "5")  // true pois e diferente de 4

    console.log(4 !== "5")  // true pois tambem e difrente de 4 e o tipo tambem

 ============================================\*/

// Desafio 1

const idade = 17

// verificar se a pessoa é maior igual a 18 anos

// se sim, deixar entrar, se não, bloquear a entrada

if(idade >= 18) {

*console*.log('Deixar Entrar')

} else {

*console*.log('Bloquear Entrada')

}

// se a pessoa tiver 17 anos

// avisar para voltar quando fizer 18 anos

if(idade === 17) {

*console*.log('Volte quando fizer 18 anos')

}

Objetos

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

// objetos são agrupamento onde contem varios parametros, ex: 1 celular é objeto,

// nele contem alguns parametros como peso, cor, tamanho, tem algumas funçoes como desligar/ligar

const aluno1 = {

    nome: 'Henrique',

    nota: 9.8

}

const aluno2 = {

    nome: 'Ana',

    nota: 10

}

const aluno3 = {

    nome: 'Fulano',

    nota: 2

}

// console é um objeto onde tem varios metodos como .log onde no log vamos passar alguma coisa ()

const media = (aluno1.nota + aluno2.nota + aluno2.nota) / 3

*console*.log(media.toFixed(2)) // toFixed (valor) diminuir as casas decimais depois do ,

Array’s (vetores)

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

// Criando Arrays (vetores) são valores agrupados em uma unica variavel

const alunos = [

    {

        nome: 'Henrique',

        nota: 9.8

    },

    {

        nome: 'Ana',

        nota: 10

    },

    {

        nome: 'Fulano',

        nota: 2

    }

]

const media = (alunos[0].nota + alunos[1].nota + alunos[2].nota) / 3

*console*.log(media.toFixed(2)) // toFixed (valor) diminuir as casas decimais depois do ,

Funções

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

// usando funçoes, passando parametros

const alunosdaTurmaA = [

    {

        nome: 'Henrique',

        nota: 9.8

    },

    {

        nome: 'Ana',

        nota: 10

    },

    {

        nome: 'Fulano',

        nota: 2

    }

]

const alunosdaTurmaB = [

    {

        nome: 'Maria',

        nota: 7

    },

    {

        nome: 'Joao',

        nota: 2

    },

    {

        nome: 'Joaquina',

        nota: 5

    }

]

function calculaMedia(*alunos*) {

    return (alunos[0].nota + alunos[1].nota + alunos[2].nota) / 3

}

const media1 = calculaMedia(alunosdaTurmaA)

const media2 = calculaMedia(alunosdaTurmaB)

function enviaMensagem(*media*, *turma*) {

    if(media > 5) {

*console*.log(`A média da ${turma} foi de ${media}. Parabéns`)

    } else {

*console*.log(`A média da ${turma} é menor que 5`)

    }

}

enviaMensagem(media1.toFixed(2), 'Turma A')

enviaMensagem(media2.toFixed(2), 'Turma B')

Estrutura de repetição

// Criando um programa que calcula a média

// das notas entre alunos e envia

// mensagem do cálculo da média

// usando funçoes, passando parametros

const alunosdaTurmaA = [

    {

        nome: 'Henrique',

        nota: 9.8

    },

    {

        nome: 'Ana',

        nota: 10

    },

    {

        nome: 'Fulano',

        nota: 2

    }

]

const alunosdaTurmaB = [

    {

        nome: 'Maria',

        nota: 7

    },

    {

        nome: 'Joao',

        nota: 2

    },

    {

        nome: 'Joaquina',

        nota: 5

    }

]

function calculaMedia(*alunos*) {

    let soma = 0

    for(let i = 0; i < alunos.length; i++) {

        soma = soma + alunos[i].nota

    }

    const media = soma / alunos.length

    return media

}

const media1 = calculaMedia(alunosdaTurmaA)

const media2 = calculaMedia(alunosdaTurmaB)

function enviaMensagem(*media*, *turma*) {

    if(media > 5) {

*console*.log(`A média da ${turma} foi de ${media}. Parabéns`)

    } else {

*console*.log(`A média da ${turma} é menor que 5`)

    }

}

enviaMensagem(media1.toFixed(2), 'Turma A')

enviaMensagem(media2.toFixed(2), 'Turma B')

Boolean

// Criando um programa que calcula a média, das notas entre alunos e envia

// mensagem do cálculo da média

// usando funçoes, passando parametros

const alunosdaTurmaA = [

    {

        nome: 'Henrique',

        nota: 9.8

    },

    {

        nome: 'Ana',

        nota: 10

    },

    {

        nome: 'Fulano',

        nota: 2

    }

]

const alunosdaTurmaB = [

    {

        nome: 'Maria',

        nota: 7

    },

    {

        nome: 'Joao',

        nota: 2

    },

    {

        nome: 'Joaquina',

        nota: 5

    }

]

function calculaMedia(*alunos*) {

    let soma = 0

    for(let i = 0; i < alunos.length; i++) {

        soma = soma + alunos[i].nota

    }

    const media = soma / alunos.length

    return media

}

const media1 = calculaMedia(alunosdaTurmaA)

const media2 = calculaMedia(alunosdaTurmaB)

function enviaMensagem(*media*, *turma*) {

    if(media > 5) {

*console*.log(`A média da ${turma} foi de ${media}. Parabéns`)

    } else {

*console*.log(`A média da ${turma} é menor que 5`)

    }

}

enviaMensagem(media1.toFixed(2), 'Turma A')

enviaMensagem(media2.toFixed(2), 'Turma B')

function marcarComoReprovado(*aluno*) {

    aluno.reprovado = false;

    if(aluno.nota < 5) {

        aluno.reprovado = true;

    }

}

marcarComoReprovado(alunosdaTurmaA)

marcarComoReprovado(alunosdaTurmaB)

function enviarMensagemReprovada(*aluno*) {

    if(aluno.reprovado) {

*console*.log(`O aluno(a) ${aluno.nome} está Reprovado!`)

    }

}

function alunoReprovado(*alunos*) {

    for(let aluno of alunos) {

        marcarComoReprovado(aluno)

        enviarMensagemReprovada(aluno)

    }

}

alunoReprovado(alunosdaTurmaA)

alunoReprovado(alunosdaTurmaB)

Organização, padronização e escrita

Ctrl + a = seleciona tudo

Ctrl + shift + tab = tirar a endentação

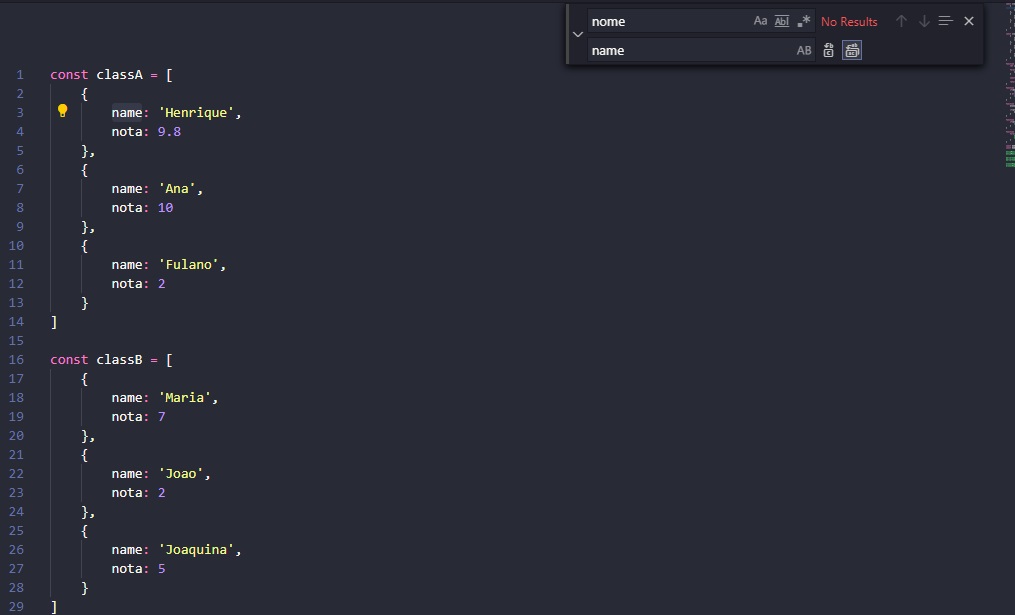
Tab = endenta o código

Formatando o código tudo de uma vez

Ctrl + shift + p = Format Selection

Selecionado todo código para alterar os comandos

Ctrl + F = ex:



Código limpo e traduzido para o inglês

const classA = [

    {

        name: 'Henrique',

        grade: 9.8

    },

    {

        name: 'Ana',

        grade: 10

    },

    {

        name: 'Fulano',

        grade: 2

    }

]

const classB = [

    {

        name: 'Maria',

        grade: 7

    },

    {

        name: 'Joao',

        grade: 2

    },

    {

        name: 'Joaquina',

        grade: 5

    }

]

function calculateAverage(*students*) {

    let sum = 0

    for(let i = 0; i < students.length; i++) {

        sum = sum + students[i].grade

    }

    const average = sum / students.length

    return average

}

function sendMessage(*average*, *turma*) {

    if(average > 5) {

*console*.log(`${turma} average: ${average}. Congrats`)

    } else {

*console*.log(`${turma} average: ${average}. Is not good.`)

    }

}

function markAsFlunked(*student*) {

    student.flunked = false

    if(student.grade < 5) {

        student.flunked = true

    }

}

function sendFlunkedMessage(*student*) {

    if(student.flunked) {

*console*.log(`${student.name} flunked!`)

    }

}

function studentsflunkeds(*students*) {

    for(let student of students) {

        markAsFlunked(student)

        sendFlunkedMessage(student)

    }

}

const average1 = calculateAverage(classA)

const average2 = calculateAverage(classB)

sendMessage(average1.toFixed(2), 'Class A')

sendMessage(average2.toFixed(2), 'Class B')

markAsFlunked(classA)

markAsFlunked(classB)

studentsflunkeds(classA)

studentsflunkeds(classB)

**INICIANDO NO FRONT END**

**Primeiros comandos usando o HTML5 e CSS3**

Criando HTML da pagina

<!DOCTYPE *html*>

<html *lang*="en">

    <head>

        <meta *charset*="UTF-8">

        <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

        <title>Henrique</title>

        <style>

            @import url('https://fonts.googleapis.com/css?family=Roboto:400,700&display=swap');

            \* {

                border: none;

                padding: 0;

                margin: 0;

                font-family: 'Roboto', sans-serif;

            }

            a {

                text-decoration: none;

                list-style: none;

            }

            body {

                background: #282a36;

            }

            header {

                padding: 32px;

                border-bottom: 1px solid #333;

                background: rgba(0, 0, 0, 0.1);

            }

            header div {

                text-align: center;

            }

            header div a {

                color: #fff;

                font-size: 25px;

                line-height: 35px;

                margin: 0 16px;

            }

            header div a*:hover* {

                color: #50fa7b;

                transition: color 200ms;

            }

        </style>

    </head>

    <body>

        <header>

            <div *class*="links">

                <a *href*="/">Sobre</a>

                <a *href*="/">Aulas</a>

            </div>

        </header>

    </body>

</html>

Primeiros passos com HTML5 e CSS3

<!DOCTYPE *html*>

<html *lang*="en">

    <head>

        <meta *charset*="UTF-8">

        <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

        <title>Henrique</title>

        <link *rel*="stylesheet" *href*="styles.css">

    </head>

    <body>

        <header>

            <div *class*="links">

                <a *href*="/">Sobre</a>

                <a *href*="/">Portfólio</a>

            </div>

        </header>

        <div *id*="wrapper">

            <img *src*="https://avatars3.githubusercontent.com/u/56804642?s=460&v=4" *alt*="">

            <h1>Henrique</h1>

            <h2>Desenvolvedor Web</h2>

            <p>Programador Front-End usando às tecnologias Javascript, HTML5, CSS3, React JS e Node JS.

                Foco em ser um Desenvolvedor Full Stack.

            </p>

        </div>

        <div *id*="links-footer" *class*="links">

            <a *href*="https://github.com/Henrique1818" *target*="\_blank">Github</a>

            <a *href*="https://www.instagram.com/henrique18\_89/" *target*="\_blank">Instagram</a>

            <a *href*="linkedin.com/in/luiz-henrique-23915916a/" *target*="\_blank">LinkedIn</a>

        </div>

    </body>

</html>

Estilo da Pagina

@import url('https://fonts.googleapis.com/css?family=Roboto:400,700&display=swap');

*:root* {

    --primary-color: #50fa7b;

}

\* {

    border: none;

    padding: 0;

    margin: 0;

    font-family: 'Roboto', sans-serif;

}

a {

    text-decoration: none;

    list-style: none;

}

body {

    background: #282a36;

    color: #fff;

}

header {

    padding: 30px;

    border-bottom: 1px solid #333;

    background: rgba(0, 0, 0, 0.1);

}

*.links* {

    text-align: center;

}

*.links* a {

    color: #fff;

    font-size: 25px;

    line-height: 35px;

    margin: 0 16px;

}

*.links* a*:hover* {

    color: var(--primary-color);

    transition: color 200ms;

}

*#wrapper* {

    text-align: center;

    padding: 22px;

}

*#wrapper* img {

    border-radius: 50%;

    border: 5px solid  var(--primary-color);

    margin-bottom: 15px;

    width: 235px;

    height: 235px;

}

*#wrapper* h1 {

    font-size: 42px;

    line-height: 52px;

    font-weight: bold;

}

*#wrapper* h2 {

    font-size: 24px;

    line-height: 34px;

    font-weight: normal;

    margin-top: 8px;

}

*#wrapper* p {

    max-width: 500px;

    margin: 15px auto;

    font-size: 18px;

    line-height: 28px;

}

*#wrapper* p a {

    color: var(--primary-color);

}

*#links-footer* a {

    font-size: 18px;

    border-bottom: 1px solid var(--primary-color);

}

**BACK END**

Instalando dependências, extensões

Npm init –y = criando o projeto como .JSON

Npm install express

instalando o package.json

npm init -y

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

express é uma dependecia onde se encontra varios extenções

para instalar o express no node

npm install express

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

chamando o express no js

const express = require('express')

const server = express() = função

iniciando o servido

server.listen(5000, function() {

console.log("servidor funcionando")

// mensagem de que o servidor esta ok

})

vai usa a porta 5000, callback função

npm start = inicia o servidor

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

criando rotas enviado uma resposta

server.get("/", function(req, res) {

return res.send("Hi")

}

passando para o get renderizar o index

server.get("/", function(req, res) {

return res.render("index")

})

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

OBS: PARA PODE RENDERIZAR UM EXTENSÃO .HTML TEM QUE PRIMEIRO INSTALAR O NUNJUCKS TEMPLATES

PASSANDO PRIMEIRO ONDE VAI BUSCA NA PASTA E QUE O EXPRESS:SERVER

POIS SE NAO INSTALAR PRIMEIRO O NUNJUCKS VAI DA UM ERRO DE RENDERIZAÇÃO

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

npm install -D nodemon = nodemon fica carregando o servidor de uma forma automatica

alterando no package.json script para nodemon

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

Instalando o nunjucks = motor que trabalha com templates e configurando

npm install nunjucks

chamando o nunjucks

const nunjucks = require("nunjucks")

server.set("view engine", "html") = configurando o motor vai pegar tudo que for html

nunjucks.configure("views", {

express:server,

autoescape: false,

nocache: true

//informando que estou utilizando o express na variavel server passando um opção

}) = passando o caminho da pasta, opções em formato de objeto

adicionando um bloco (variavel no html) usando o nunjucks

{% block content %}

dinâmico

{% endblock %}

passando a extenção no nunjucks

{% extends "layout.html" %}

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

carregar o estilo

server.use(express.static("public")) express usando arquivos staticos

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

depois de muda a extensão .html passar para .njk

instalar a template nunjucks template

no settings json

para o vscode interpreta como html o njk

instalando o emmet

"emmet.includeLanguages": {

"njk": "html"

},

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

Exportando dados

modules.exports = [

{

id: "GykTLqODQuU",

title: "Formulário Animado com JS puro e CSS",

duration: "57 min",

price: "Free"

}

]

criando o modules exports para pode exporta de uma forma mais facil para outras aplicações;

LAÇO DE REPETIÇÃO COM O NUNJUCKS

{% for item in items %}

<div class="card" id="{{item.id}}">

<div class="card\_\_image-container">

<img src="https://img.youtube.com/vi/{{item.id}}/maxresdefault.jpg" alt="Imagem de {{item.title}}">

</div>

<div class="card\_\_content">

<p>{{item.title}}</p>

</div>

<div class="card\_\_info">

<p>{{item.duration}}</p>

<p class="card\_\_price">{{item.price}}</p>

</div>

</div>

{% endfor %}