Estas anotações são sobre o bootcamp da instituição IGTI iniciado em 20/11/2020.

<https://online.igti.com.br/courses>

MÓDULO 01

Aula 02 - Node.js:

Para iniciar um projeto node.js é necessário criar um pacote do tipo .json, para isto, basta abrir o cmd, navegar até a pasta em que se deseja criar os pacotes e então digitar o seguinte:

npm init -y

E estará criado o projeto node.js em formato .json.

Existem dois tipos de dependências:

* Dependency: o tipo de dependência padrão local;
* Dev-dependency: o tipo de dependência que é utilizado pelo desenvolvedor apenas para o desenvolvimento, ou seja, isso não será utilizado no deploy quando a aplicação for para o servidor de fato;

Para criar uma dev-dependency para desenvolver com node.js, basta navegar pelo terminal até a pasta desejada e digitar:

npm install live-server –-save-dev

Então este comando irá inserir no package.json uma área responsável pelo controle de dependências de desenvolvimento (devDependencies); e também irá criar um arquivo, que não deve ser modificado, chamado package-lock.json que é responsável pelo controle de versões destas dependências.

Isto é de extrema utilidade pois cria um arquivo .json que grava quais dependências o projeto tem e caso seja necessário trocar de máquina o projeto, movendo apenas estes arquivos e executando o comando ‘npm install’ o npm instalará todas as dependências necessárias.

Para rodar o live-server, primeiro deve-se configurar que ele será responsável por rodar a aplicação e para isto, deve-se adicionar a key “start” no package.json com o valor “live-server”, ficando assim:

{

  "name": "aula-02-node-js",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1",

    "start": "live-server"

  },

  "keywords": [],

  "author": "",

  "license": "ISC",

  "devDependencies": {

    "live-server": "^1.2.1"

  }

}

E então no terminal, na pasta em que o arquivo se encontra rodar:

npm start

Pronto, o navegador abrirá a aplicação.

Aula 03 – Noções HTML:

Aula 04 – Noções CSS:

Aula 5 - Introdução ao JavaScript

Aula 6 - JavaScript - comandos de bloco

Sintaxes:

// Comparando 2 números com if/else

var a = 3;

var b = 5;

if (a < b) {

  console.log(a + ' é menor que ' + b);

} else {

  if (a > b) {

    console.log(a + ' é maior que ' + b);

  } else {

    console.log(a + ' e ' + b + ' são iguais');

  }

}

// Validação de dia com if/else

var day = 1;

if (day === 1) {

  console.log('Domingo');

} else {

  if (day === 2) {

    console.log('Segunda-feira');

  } else {

    if (day === 3) {

      console.log('Terça-feira');

    } else {

      if (day === 4) {

        console.log('Quarta-feira');

      } else {

        if (day === 5) {

          console.log('Quinta-feira');

        } else {

          if (day === 6) {

            console.log('Sexta-feira');

          } else {

            if (day === 7) {

              console.log('Sábado');

            } else {

              console.log('Dia inválido');

            }

          }

        }

      }

    }

  }

}

// Validação de dia com switch

switch (day) {

  case 1:

    console.log('Domingo');

    break;

  case 2:

    console.log('Segunda-feira');

    break;

  case 3:

    console.log('Terça-feira');

    break;

  case 4:

    console.log('Quarta-feira');

    break;

  case 5:

    console.log('Quinta-feira');

    break;

  case 6:

    console.log('Sexta-feira');

    break;

  case 7:

    console.log('Sábado');

    break;

  default:

    console.log('Dia inválido');

}

// Comparando 2 números com operador ternário

var compareResult = a > b ? 1 : a < b ? -1 : 0;

// Verificando dia da semana com operador ternário

var weekDay =

  day === 1

    ? 'Domingo'

    : day === 2

    ? 'Segunda-feira'

    : day === 3

    ? 'Terça-feira'

    : day === 4

    ? 'Quarta-feira'

    : day === 5

    ? 'Quinta-feira'

    : day === 6

    ? 'Sexta-feira'

    : day === 7

    ? 'Sábado'

    : 'Dia inválido';

Aula 7 - JavaScript - manipulação do DOM

**document.querySelector()**

Para manipular elementos da página HTML via JS utiliza-se o método do objeto document ‘querySelector’, ele é responsável por selecionar um ou vários elementos da página HTML, estes podendo ser de três tipos:

var input1 = document.querySelector('#idGenerico');

input1.value = 'Raphael Gomide';

var input2 = document.querySelector('.classeGenerica');

input3.value = 'Abilonson';

var input3 = document.querySelector('elementoHTML'); //<p>, <h1>, etc...

input3.value = 'Jurema';

**document.querySelectorAll();**

Semelhante ao ‘querySelector()’ porém retorna uma nodeList com todos os elementor de um tipo especificado no argumento do método;

const important = document.querySelectorAll('urgent');

Uma maneira interessante de se manipular o conteúdo do texto de um elemento é após capturar o elemento desejado usando um querySelector, utiliza-se o método ‘.textContent();’ para alterar seu valor.

const important = document.querySelectorAll('urgent');

important.textContent = 'Olá Mundo! sou importante!';

Aqui é interessante mostrar que é possível manipular o CSS através do JS. Isto é feito através das classes em CSS.

Supões que se deseja alterar a formatação de um texto quando ele for clicado, para isto utilizamos a propriedade onclick no HTML que será responsável por executar esta função, e então criamos uma classe CSS com a propriedade desejada para então escrever uma função para adicionar essa classe à respectiva tag HTML.

Ficando com:

HTML:

<h1 *id*="toBeEmphasized" *onclick*="clicked(event)" *class*="">Olá Mundo!</h1>

CSS:

*.emphasis* {

  font-weight: bold;

  color: darkgreen;

}

JS:

function clicked() {

  const text = document.querySelector('#toBeEmphasized');

  classList = *Array*.from(text.classList);

  //Se tiver a classe emphasis a função irá remover esta classe, caso contrário irá adicionar:

  if (classList.includes('emphasis')){

    text.classList.remove('emphasis');

    text.textContent = 'Sem o ".empahsis" nas classes deste objeto';

  } else {

    text.classList.add('emphasis');

    text.textContent = '".emphasis" foi adicionado';

  }

}

Este foi um exemplo de manipulação simples de CSS de um elemento HTML utilizando JS, demonstrando o método ‘remove()’ e ‘add()’ da classList do elemento HTML.

Aula 8 - JavaScript - manipulação do DOM