**JAVASCRIPT**

**OUTPUT:**

* Manipulando um item html pelo seu id através do JS;

Document.getElementById(“id”).innerHTML = “novo texto”;

* Janela de alerta da página;

Window.alert(“mensagem”);

* Exibir a mensagem no console;

console.log(‘o que quiser’);

**VARIÁVEIS:**

* Use o “let” ou “var” pra criar variáveis.
* Var pode ser usada em qualquer escopo global
* Let só pode ser usada no escopo local, não se vê ele fora do código, por exemplo, no navegador.
* Não pode começar com números
* Frases tem que estar entre parênteses
* Número não precisa
* Pra aparecer as informações no console, digite ‘console.log(variável)’
* ‘const’ é uma constante, uma variável que não pode ser modificada, é útil na utilização de arrays e objetos.
* Concatenação; adição, juntar, somar.

**IF E ELSE:**

* > maior
* < menor
* == igual
* === exatamente igual
* != diferente
* !== totalmente diferente
* >= maior ou igual
* <= menor ou igual
* && E, and
* || Ou, or

**CONDICIONAL TERNÁRIA:**

* Let isMember = true;

Let shipping = isMember ? 2 : 10;

Desmembrando:

isMember **{deu certo?}** ? **{então é 2}** 2 **{deu errado?}** : **{então é 10}** 10;

* ? se sim
* : caso contrário

**SWITCH:**

* let profession= “policial”

switch (profession) {

case ‘fiscal’ :

console.log(sua camisa será verde);

break;

case ‘policial’ :

console.log(‘Sua camisa será azul);

break;

default:

console.log(‘sua camisa será preta)

break;

Desmembrando:

switch (profession) **{a variável que será analisada}** {

case **{caso seja}** ‘fiscal’ :

console.log(sua camisa será verde);

break **{acabou esta case aqui}** ;

default **{normal, caso não satisfaça nenhuma das anteriores}** :

console.log(sua camisa será preta)

break;

}

* switch; troque, trocar
* case; caso, caso seja, if, true
* break: quebrar, separar, terminar
* default; padrão, else

**FUNÇÕES:**

**Arrow Function:** Diferença pra função normal é deixar o código mais limpo, ex;

Function;

Function somar(x, y) {

Return x + y;

}

Arrow Function;

Const somar = (x, y) => x + y;

**Variável de escopo global**: criada fora da função, pode ser usada em qualquer function

**Variável de escopo local**: criada dentro da função, é usada somente naquela situação;

**Função dentro de uma função:**

Function addPtc(a, b) {

Const square = (x) +> x \* x; **[aqui não precisa ser arrow function mas fica melhor pra não se perder]**

Let ptcA = square(a);

Let ptcB = square(B);

Return ptcA + ptcB;

}

**ARRAY´S**

**Lista de algo**

Como criar um array?

* Let colors = [‘blue’, ‘red’, ‘green’];
* Console.log(colors[1])

Vai exibir a cor **red**.

Da pra pegar as informações de um array e usar em outro? **Sim**

Da pra criar um array dentro de outro array? **Sim**

Como eu faço pra adicionar um item no array depois de tê-lo feito já?

* Nomedoarray.push(‘oquequeradd’);

Pop = remove o último item do array

Shift = remove o último item do array

Array.length = para visualizar quantos itens tem o array

Array[1] = visualizar o item que está no 1

**OBJECT´S**

**Grupo de informações**

Diferença dele pro array, nele eu posso dar características, ex;

Let personagem = {

Nome: ‘Arthur’,

Idade: 20,

Pais: ‘Brasil’

}

Como visualizar somente uma característica?

* personagem.nome

Da pra criar um objeto dentro de um objeto? **Sim, mesmo esquema do array.**

Da pra criar um array dentro de um objeto ou vice-versa? **Sim**

Mudar algum item:

Personagem.nome = ‘pedro’

Adicionar mais idade:

Personagem.idade += 10

Exemplo de um Objeto => Array => Objeto (lista de carros e cores deles:

Objeto = Azul

Array = Verde

Let personagem = {

Nome: ‘Arthur’,

Idade: 20,

Carros: [

{ modelo: ‘Fiat’, cor: ‘preto’ },

{ modelo: ‘Ferrari’, cor: ‘vermelha’ }

]

}

E pra eu acessar somente a cor do Fiat?

* Console.log(personagem.carro[0].cor);

Utilizando uma função dentro de um objeto:

Function: Verde

Let pessoa = {

Nome: ‘Arthur’,

Sobrenome: ‘miranda’,

Idade: 20

nomeCompleto: function () {

return `${this.nome} ${this.sobrenome};

}

Ou ao invés de function se quiser usar arrow function pode também, da seguinte maneira:

nomeCompleto: () => {

return ‘Arthur Miranda’;

}

**LOOP´S**

**Quando eu quero executar algo várias vezes.**

Como é demonstrado o loop? **For**

**Ex:**

for (let n = 1; n <= 10; n++) {

console.log(‘Arthur’ + n);

}

**Iria fazer isto:**

Arthur 1

Arthur 2

Arthur 3

Arthur 4

Arthur 5

Arthur 6

Arthur 7

Arthur 8

Arthur 9

Arthur 10

Laranja: por onde vai começar a minha contagem

Azul: Até onde vai a minha contagem

Roxo: Vai avançar de quanto em quanto (no caso ali de 1 em 1)

Vermelho: pra mostrar o número depois do arthur pra ficar mais intuitivo, mas não é necessário

**LOOP´S COM WHILE**

*Enquanto a condição for satisfeita o código vai continuar rodando*

* let c = 0;

while(c < 10) {

console.log(‘”Número: “+c’);

c++;

}

Vermelho: Condição;

Verde: c = c + 1... serve pro código não ficar rodando pra sempre pois não sairia do 0

**LOOP´S EM ARRAY**

Let cores = [

{ nome: ‘preto’, quantidade: 10 }

{ nome: ‘vermelho’, quantidade: 12 }

];

**Maneiras de usar o loop nesse array:**

for (let = 0; n < cores.lenght; n++) {

console.log(cores[n].quantidade[0]);

}

**Outra maneira:**

For (let i in cores) {

Console.log(cores[i].quantidade[0]);

}

**Outra maneira:**

For (let cor of cores) {

Console.log(cor.quantidade[0]);

}

Vermelho: usamos o cores.length pra ele dar loop na quantidade de vezes em que se temos objetos, por exemplo, ali seriam duas vezes, preto e vermelho.

Se eu quiser visualizar só a quantidade do vermelho? Adiciona o verde no código

**EVENTOS:**

Eventos de clique;

1°: Coloque na sua tag ‘button’ no html, o ‘onclick=”função”’

2°: No JS crie uma função para quando o evento for disparado o que você programou aconteça.

Ou

1°: Usar o ‘this’ pra mudar o próprio botão

2°: onclick=”this.innerHTML=’Clicou’”

Eventos de passar o mouse;

1°: Mesmo esquema do onclick, muda-se para ‘onmouseover=”função”’ ou ‘onmouseout=”função”’

2°: No JS crie uma função para quando o evento for disparado o que você programou aconteça.

Existem diversos de eventos com o mouse, se você digitar ‘on’ no vscode ele já te dará diversas opções de eventos.

Eventos de teclado;

1°: Coloque na sua tag ‘button’ no html, o ‘onkey=”função”’, ai pode ser, onkeyup. onkeydown e onkeypress

2°: No JS crie uma função para quando o evento for disparado o que você programou aconteça.

KeyboardEvent ex;

Onkeyup=”digitou(event)”

* ‘event’ funciona tipo o ‘this’, que mostra quais teclas estão sendo apertadas naquele evento.

*Exemplo/Dica:*

Eu quero que o usuário digite algo, e quando ele apertar o **enter**, isto apareça no meu console...

* Function digitou(e) {

If (e.keyCode == 13) {

Let texto = document.getElementById(“campo”).value;

Console.log(‘texto’)

}

}

Azul: usamos para determinar que é um código de keyboard

Verde: código da tecla **enter**

Laranja: ID do campo que estava sendo digitado

Vermelho: Pra pegar o valor daquele campo específico

*Exemplo mais avançado:*

Tem que estar com o **enter** e o **ctrl** apertado...

* Function digitou(e) {

If (e.keyCode == 13 && e.ctrlKey == true) {

Let texto = document.getElementById(“campo”).value;

Console.log(‘texto’)

}

}

Roxo: porque não keyCode? Porque ctrl é assim e pronto o cara não explicou porque, mas está ai o exemplo.

**Usando botões no JS pra alterar o CSS:**

1°: Crie no seu CSS ‘class’ com o que você deseja alterar, neste exemplo, mudaremos a cor do H1.

2°: Crie no seu HTML buttons para referenciar cada uma das cores que desejará alterar, no nosso caso, vermelho e azul.

3°: Agora em cada button coloque a função que ele corresponderá.

4°: Agora crie as funções, azul e vermelho, e dentro delas usaremos os seguintes códigos;

* Limpar()
* Document.getElementById(“titulo”).classList.add(‘azul’ ou ‘vermelho’);

Azul: usaremos este para que quando o outro botão seja clicado, sejam excluídas as informações do botão anterior, mas, é uma function certo? Então que função será esta?

* Function limpar() {

Document.getElementById(‘titulo’).classList.remove(‘azul’ ou ‘vermelho’)

}

Laranja: Id do nosso item no HTML que está sendo trocado de cor.

Verde: é pra acessar a lista de classes do css no js.

Rosa: tag usada pra ADICIONAR a cor.

Cinza: tag usada pra REMOVER a cor.

*Outro exemplo;*

Quero Mostrar o número de telefone só depois de eu clicar no botão...

1°: Lá no CSS você pega o id da div na qual está situado o seu botão, e escreve que você quer ‘display:none’ pra o número ficar oculto.

2°: Crie um botão

3°: Agora no seu JS crie a seguinte função;

* Function mostrarTelefone() {

Document.getElementById(telefone”).style.display = ‘block’;

}

*E se eu quiser que o botão suma depois de eu aperta-lo?*

4°: Function mostrarTelefone(**elemento**) {

**Elemento.stlye.display = ‘none’;**

Document.getElementById(telefone”).style.display = ‘block’;

}

E lá no HTML dentro do ‘onclick = mostrarTelefone(**this**)’ adiciono o this...

**DOM = Document Object Model**

* document.getElementById(‘id’).**innerHTML** = “Arthur”;

*Esse comando irá pegar o elemento que está identificado com o id que selecionei e por conta do* ***innerHTML****, que quer dizer ‘****dentro do htm****l’ e irá adicionar o item que vem depois nas aspas.*

* document.getElementsByClassName(‘class’)[0].innerHTML = ‘item alterado’;

*Vai selecionar as classes que selecionei, se escreve no plural, porque se pode ter mais de uma class com o mesmo nome, ela sempre te retornará um array.*

* Document.getElementsByTagName(‘tag’)

*Vai selecionar todas as tags desejadas*

* Document.getElementsByName(‘nome’)

*Menos utilizado...*

* Document.querySelector(‘’)

*Substitui todos os acima, usando ‘#’ pra id´s e ‘.’ Pra classes*

* Document.querySelectorAll(‘’)

*Seleciona todos os itens se tiver mais de um, em forma de array*

Criando uma função para mudar a cor de uma div quando o botão for apertado;

1°Crie os botões que servirão como gatilho para as funções

2°Coloque neles a função onclick=’cor de cada um’

3°Crie no css uma seção para cada uma das cores

4°Crie a div que mudará de cor

5°Crie as funções abaixo

Function azul() {

Document.querySelector(‘div’).**classList**.**add**(‘azul’);

}

Function vermelho() {

Document.querySelector(‘div’).classList.add(‘vermelho’);

}

Vermelho: Acessa a lista de classes do css

Azul: adiciona a classe a seguir na div

*Trocar o ‘add’ por* ***‘contains’*** *vai verificar se há aquela classe naquele elemento.*

**TIPOS DE DADOS**

Var nome = “arthur” -> **String**

Var idade = “20” -> **Number**

Var save = true -> **Boolean**

Var cidade -> **Undefined**

*Exemplo: ‘typeof arthur -> string’*

**PARÂMETROS**

Funções do próprio jS;

* function alterar(titulo) {

document.getElementById(“titulo”).innerHTML = titulo;

}

Alterar(‘mudei o titulo’)

* function somar(x, y) {

let total = x + y;

document.getElementById(“titulo”).innerHTML = total;

}

Somar(‘2, 2’);

Ou...

* function somar(x, y) {

let total = x + y;

return total;

}

var resultado (‘10, 15’)

console.log(‘resultado’)

**MANIPULANDO ATRIBUTOS**

*Mudando a foto (gato e cachorro) da página através de um botão;*

1°: crie a tag ‘img’ pra alocar a imagem principal, neste caso do cachorro

2°: crie dois buttons, um pro cachorro e outro para o gato

3°: coloque no button a função ‘onclick=”trocarImagem(‘cachorro.jpg’)

e no outro botão substitua pelo gato.jpg.

4°crie a função abaixo:

function trocarImagem(filename) {

document.querySelector(.imagem’).setAttribute(‘src’, ‘images/’+filename);

}

Vermelho: tag para setar a troca de atributos

Azul: o atributo que eu desejo trocar

Verde: como vai ser mudado

*Agora quero que a página identifique qual animal está aparecendo.*

0°: dentro da tag img, adicione ‘data-animal=”cachorro”’

1°: adicione dentro do ‘onclick=”trocarImagem(‘cachorro.jpg’, ‘cachorro’)’

2°: crie outro button com o ‘onclick=”pegarAnimal()”’

3°: crie uma outra função como abaixo:

function pegarAnimal() {

let animal = document.querySelector(‘.imagem’).getAttribute(‘data-animal’);

alert(“O animal é: “+animal);

}

Laranja: adicionados onde já tinha coisa

Roxo: seleciona um atributo

**DIMENSÕES**

* document.querySelector(‘.texto’).offsetHeight ou Width

*offset vai mostrar qual a dimensão do item escolhido*

* document.querySelector(‘.texto’).clientHeight ou Width

*é descontado a barra de rolagem e borda, deixa só a largura e o padding*

* document.querySelector(‘.texto’).scrollHeight ou Width

*tamanho real do conteúdo*

**SCROLL**

* document.querySelector(‘.texto’).scrollTop ou Left

*vai mostrar em que px está localizado o scroll naquele momento*

* document.querySelector(‘.texto’).scrollTo(0, 0)

*via mandar o scroll pro px que eu mandar*