

Dev em dobro intensivão jan 2025

## HTML



HTML é a linguagem de **marcação** de hipertexto que se responsabiliza pela estrutura e esqueleto de uma página na web. Onde cada elemento visível na tela é uma `< nome da tag >`, `<p>`, ou seja, cada objeto a ser renderizado na tela é interpretado pelo navegador através de uma tag.

Abertura da tag

(`<` sinal de menor) `p` (`>` sinal de maior) `p` é o nome da tag

Conteúdo da tag

Eu sou um parágrafo

`</p>` fechamento da tag

Há tags que não possuem conteúdo:

``

As tags possuem atributos ou características, o atributo `class` representa a classe css e "paragrafo" é valor que representa o nome da classe:

`<p class="paragrafo"> Eu sou um parágrafo</p>`

`<html>` representa a raiz do documento HTML

`<head>` representa as informações do documento HTML, como título da página. Essas informações não aparecem para o usuário (exceto o favicon e o título da página).

`<body>` representa o corpo da página contendo o conteúdo visível.

Estrutura mínima do html:

`<!DOCTYPE html>` -> Declarando que estou usando o HTML 5

`<html lang="qualquer idioma: en,pt-BR,span,ita">`

`<head>`

`<meta charset="UTF-8">` -> fazer com que o navegador aceite acentos

`<meta name="viewport" content="width=device-width, initial-scale=1.0">` -> a página

sempre vai ter a largura do dispositivo e ampliação inicial de 100%

`<title>Título que quiser</title>`

(Opcional) `<link rel="icon" type="image/x-icon" href="/src/images/imagem.ico">` -> ícone da guia no navegador

`</head>`

`<body>`

`<header>` -> cabeçalho

`<nav>` -> menu de navegação

`</nav>`

`</header>`

`<footer>` -> rodapé

`</footer>`

`</body>`

</html>

Extensões usadas no vscode:

Live server

Omni Theme

Vs code icons

Color highlight

Habilitar salvamento automático na opção arquivo.

Nas configurações procurar por word wrap para quebrar a linha automaticamente.

Ctrl+Alt+O -> inicializa o live server

Estrutura de headings / títulos onde cada N significa o nível de importância (título e subtítulo)

<H1>PRINCIPAL DESTACADO </H1>

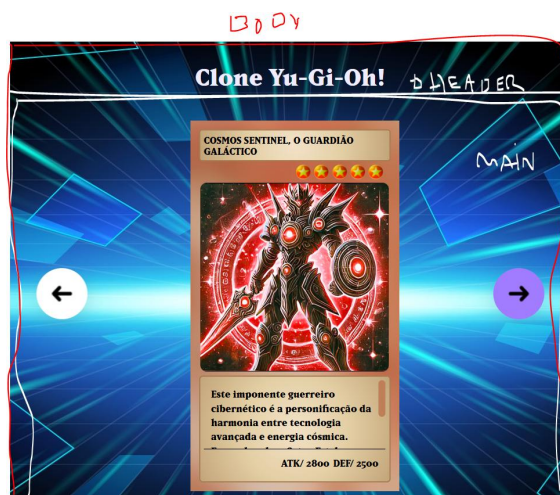
<H2>Subtítulo ou Segundo título</H2>

<H3></H3>

...</H6>

Tags semânticas são aquelas que não tem efeito visual, mas tem significado. Exemplo: <header>

</header> que funciona como esta área vou ser o cabeçalho da página.



<header class="cabecalho"></header> -> cabeçalho da página. Class é o atributo **css** que vai identificar todos os elementos que façam parte da classe com o valor "cabecalho".

<main></main> -> conteúdo principal



<ul> Lista não ordenada

<li>item da lista</li>

</ul>

<button id="btn-voltar">Botão com identificador único e exclusivo



</button>

<div>divisória sem peso semântico para agrupar conteúdos/elementos<span>pequeno trecho de conteúdo de menor importância</span></div>

<p>parágrafo</p>

<style> estilização dentro do documento html, sem usar um arquivo css separado </style>

Texto das cartas:

Cosmos Sentinel, Guardião Galático

Este imponente guerreiro cibernético é a personificação da harmonia entre tecnologia avançada e energia cósmica. Empunhando o Cetro Estelar, Cosmos Sentinel pode manipular o espaço-tempo, anulando ataques e fortalecendo aliados.

ATK/ 2800

DEF/ 2500

Nebula Dragon, O Ser das Estrelas Eternas

Uma criatura mítica nascida da fusão de nebulosas e energia estelar. Com suas escamas cristalinas, Nebula Dragon canaliza o poder dos cosmos, devastando seus inimigos com rajadas de energia pura. Diz-se que ele aparece apenas em momentos de caos universal, como arauto de renovação e equilíbrio celestial

ATK/ 3200

DEF/ 2000

Cyberblade Paladin, o Cavaleiro Digital

Armado com a lendária Cyberblade, este cavaleiro cibernético é um protetor das dimensões digitais. Com reflexos aprimorados e um senso de justiça inabalável, ele combate invasores que ameaçam o equilíbrio dos mundos virtuais e reais. Sua espada emana um brilho tecnológico que pode cortar tanto matéria física quanto código digital.

ATK/ 2600

DEF/ 2300

Mechadragon X, o Destruidor Biomecânico

Uma fusão mortal de biologia dracônica e engenharia cibernética, Mechadragon X foi criado para ser uma arma definitiva. Com suas asas energizadas e múltiplos núcleos de poder, ele é capaz de disparar rajadas devastadoras e neutralizar até os mais fortes adversários. Sua presença no campo de batalha é um presságio de destruição iminente, conhecido como o "Fim das Eras".

ATK/ 3500

DEF/ 3000

Archmage Stellarion, o Guardião das Estrelas

Um mago enigmático que manipula as forças do cosmos para proteger o equilíbrio universal. Archmage Stellarion utiliza seu Cetro Cósmico para canalizar feitiços de luz estelar, podendo banir inimigos para dimensões distantes ou fortalecer aliados com energia astral. Sua presença no campo de batalha é envolta em mistério e poder, sendo reverenciado como o "Sábio do Infinito".

ATK/ 2400

DEF/ 2600

Aegis Knight, o Guardião do Firmamento

Vestindo uma armadura forjada com fragmentos de estrelas, Aegis Knight é o defensor final das dimensões. Empunhando a Espada Astral e o Escudo Celestial, ele possui o poder de anular ataques inimigos e contra-atacar com golpes de pura energia cósmica. Seu juramento é proteger o equilíbrio entre luz e trevas, sendo lembrado como a "Fortaleza do Cosmos".

ATK/ 3000

DEF/ 2800

Stormbringer Dragon, o Arauto das Tempestades

Este poderoso dragão domina os céus, invocando trovões e relâmpagos com o bater de suas asas. Stormbringer Dragon é a personificação da fúria da natureza, destruindo tudo em seu caminho com rajadas elétricas e ventos cortantes. Diz-se que sua aparição é o prelúdio de uma tempestade sem fim, trazendo destruição e renovação aos campos de batalha.

ATK/ 2900

DEF/ 2100

Alt+Shift+F -> Auto formatação do código no visual studio code

---

## CSS



CSS significa folha de estilo em cascata, ou seja, são arquivos que irão estilizar os elementos da página html de cima para baixo.

```
h1 {  
    color: blue;  
}
```

Neste exemplo, todos os títulos da página html vão ser estilizados com a cor azul (cacterística) do primeiro até o último título. Através destes **seletores**, o navegador vai ser informado quais elementos html vão ser estilizados e como vão ser estilizados.

Os seletores são divididos em categorias:

**P {**

SELETOR POR TAG, TODAS AS TAGS DE **PARÁGRAFO** DO HTML TERÃO DETERMINADAS **CARACTERÍSTICAS** COMO POR EXEMPLO, **TAMANHO DE FONTE**-> **FONT-SIZE: 20PX;**

**}**

**.cabecalho {**

SELETOR DE CLASSE, TODAS AS TAGS QUE CONTENHAM **A CLASSE** **cabecalho** TERÃO DETERMINADAS **CARACTERÍSTICAS**.

**}**

**\* {**

SELETOR UNIVERSAL, TODAS AS TAGS TERÃO DETERMINADAS **CARACTERÍSTICAS**.

**}**

**UL LI {**

SELETOR POR DESCENDÊNCIA, TODAS AS TAGS 'ANINHADAS' TERÃO DETERMINADAS **CARACTERÍSTICAS**, POR EXMPLO PARA TODOS LI QUE ESTÃO DENTRO DE UMA UL.

**}**

Exemplo:

CSS

**.cabecalho{**

**Background-color: red;**

**}**

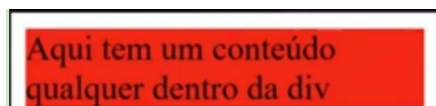
HTML

**<div class="cabecalho">**

Aqui tem um conteúdo qualquer dentro da div

**</div>**

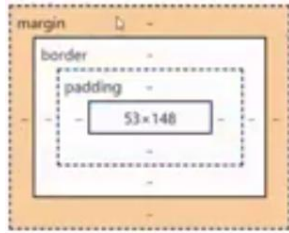
Navegador



Reset.css recebe o nome **reset**, pois todos navegadores por padrão implementão por conta própria alguns estilos para cada tag. Então podemos **redefinir** tais estilos pré-implantados.

Carregando arquivo css no html: <link rel="stylesheet" href="caminho onde arquivo css está armazenado">

Glossário de propriedades css:



Margin: 0; -> espaçamento **externo** entre os elementos html <>0<> espaço de 100px <>

Padding: 0; -> espaçamento **interno** dos elementos html [0px] -> [ espaço de 100px ]

Box-sizing: border-box; -> inclui o padding e a border na largura e altura do elemento.

List-style: none;-> determina o estilo da lista neste caso nenhum, para uls: square | circle °; para ol: upper-roman II.III | lower-alpha 1. 2.

Background-image: url('../imagens/foto-site.png') -> carrega a imagem de fundo da página

Background-repeat: no-repeat; -> caso a imagem de fundo não cubra toda a página, automaticamente vai se repetindo, para evitar isso -> no-repeat.

Background-size: **cover** | **contain**; -> faz com que a imagem de fundo **cubra** toda a dimensão da página ou **Contain** o conteúdo dentro da tag.

Background-color: red; -> fundo com cor vermelha.

Background: radial-gradient(circle,#f3e2c8, #e4cfa5,#c4a57a); -> fundo com efeito circular, começando pela **primeira cor**, no meio a **segunda cor**, de meio para o fim a **terceira cor**. (site de paletas de gradiente: [css.gradient.io](https://css.gradient.io))

**Display:** flex|inline-block|none; -> **Visualização** que define o posicionamento (X,Y) e o alinhamento(Horizontal e Vertical) dos objetos, flex-> em linha um do lado do outro por padrão.

**Inline-block**-> um do lado do outro em bloco, block-> um em cima do outro. \*span não possui peso para ocupar espaço por si própria, precisando do display. **None**-> nenhum, para retirar o objeto da página.

Flex-direction: column|row; column-> um embaixo do outro, row-> em linha

Justify-content: center|flex-end; -> alinhamento horizontal

Align-items: center;->alinhamento vertical

Align-self: center; -> alinhamento específico de um conteúdo

Gap: 130px; -> distância entre elementos **dentro do display**

Color: #f1ecff, rgb(qty red, qty green,qty blue),color name.

Opacity: 0,0.5,1; -> grau de visibilidade do conteúdo, 0-> vai esconder o elemento não retirar

Estrutura de importação de fonte local:

```
@font-face {
```

```
    Font-family: 'Yugioh'; ->nome da fonte
```

```
    Src: url(../fontes/fonte-yu-gi-oh.otf);-> caminho da fonte
```

}

Para usar a fonte basta usar o font-family:

Body {font-family: 'Yugioh' }

TAG PAI (TAG AGRUPADORA/CONTAINER, oque for estilizado no pai, o filho vai ser afetado/herdado)

<header>

<h1>TAG FILHO</h1>

</header>

.CARTAO.SELECIONADO{} -> estilizando conteúdo que pertença a classe cartao & selecionado

Max-width: 260px; -> largura máxima do conteúdo

Width: 300px; -> largura do conteúdo

Min-height: 100vh; -> altura mínima do conteúdo, útil quando o container está necessitando de mais espaço. 100viewportheight -> 100% da janela do navegador.

Max-height: ; -> altura máxima do conteúdo

Height: 500px; -> altura do conteúdo

Border: 2px (espessura) solid (tipo) #a67c52 (cor);

Border: 0; -> anula as bordas

Border-bottom: 2px (espessura) solid (tipo) #a67c52 (cor);

Bottom: -13%; -> posicionamento da parte debaixo do conteúdo

Right: 20%; -> posicionamento do lado direito do conteúdo




Left: 20%; -> posicionamento do lado esquerdo do conteúdo


Top: 0; -> posicionamento do topo do conteúdo


Border-radius: 12px; -> Arredondamento da borda do conteúdo

Font-size: 13px; -> tamanho da fonte/letra

Text-transform: UPPERCASE|lowercase|Capitalize; -> modifica o texto para maiúsculo, minúsculo, primeira letra em maiúsculo.

Text-align: left|right|justified; -> alinhamento do texto (    )

Line-height: 1.5; -> altura da linha de 1.5 (  )

Overflow-y: scroll; -> barra de rolagem(scroll) na vertical(y) (  )

Position: absolute| relative; -> posicionamento em relação a página toda como um baralho, uma carta abaixo da outra ou debaixo de um container, de modo geral é importante deixar o pai(.lista-personagens) em relative e o filho absolute (.cartao).

Z-index: 1; -> faz com que o item da lista seja prioritário(1st) em relação ao demais, como num baralho, ser a carta do topo.

Cursor: pointer; -> mudar ponteiro para a mãozinha de seleção/click.

Box-shadow: rgba(100,100,111, 0.2) 0px 7px 29px 0px; -> Sombreamento sobre o elemento rgba significa tons de vermelho, verde, azul com transparência de 20% e usando os pixels para espalhamento da sombra.

Transition: **background-color** 0.2s easy-in-out; -> aplica um efeito de transição para alternar a cor de fundo do elemento em 2 milissegundos toda vez que passar o mouse **sobre(in)** o elemento e **tirar(out)** o mouse de **forma suave(easy)**.

.btn-seta:hover{**background-color**:#8351fe}-> quando passar o mouse sobre o elemento, troca a cor de fundo para roxo.

Transform: rotateY(180deg);-> vai transformar o elemento rotacionando-o verticalmente em 180 graus.

Estrutura para estilização da barra de rolagem(scrollbar)

```
body::-webkit-scrollbar-track{
  background-color: #c08057;
}

::-webkit-scrollbar {
  width: 10px;
}

::-webkit-scrollbar-thumb {
  background: #c08057;
  border-radius: 10px;
}

::-webkit-scrollbar-thumb:hover {
  background: #c08057;
}
```

Site: getscan css

Responsividade:

Extensão que ajuda a testar as resoluções menores: responsiveViewer

@media(max-width: 760px){

Qualquer dispositivo com resolução até 760px de largura recebe outras estilizações

Body{

Background-position: center; -> posiciona o fundo/imagem de fundo ao centro

}

}

---

## JAVASCRIPT

Javascript é a linguagem de programação\* da web usada para interagir com o cliente (elementos html) e funcionar do lado do cliente (navegador).

Os objetos são uma coleção de propriedades, onde uma propriedade/característica é uma associação de chave e valor.

Const Afazer {

    Descricao = "curso js"

    Estado = "em andamento"



....  
}

No console do navegador -> Afazer: {descricao:"curso js",estado:"em andamento"}.

Outro tipo de objeto é a instância de uma classe que além de propriedades, pode conter .comportamentos (métodos) que irão receber um ou mais valores (argumentos, parâmetros):

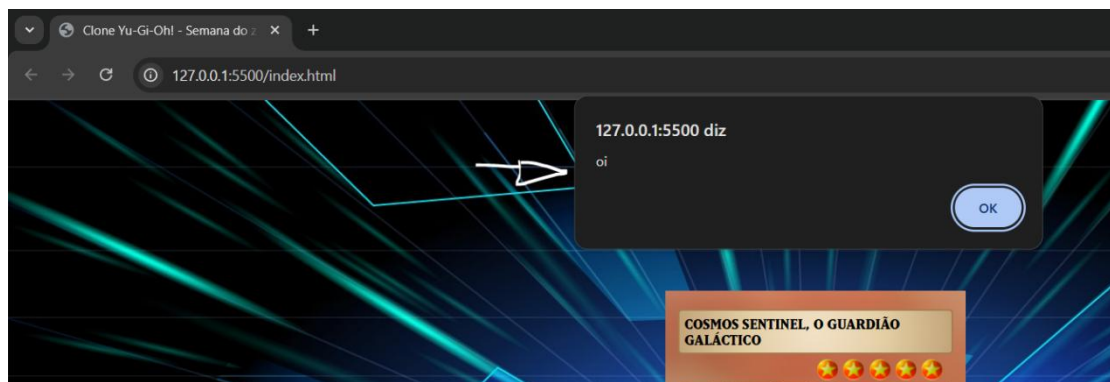
Document.getElementById('descricao');

Document = representa todos os elementos do documento html.

getElementById = resgata / captura o elemento pela identificação.

As variáveis são nomes associados a um endereço/espço de memória que armazenam informações manipuladas durante a execução do algoritmo. Especificação: const -> valor constante e inalterável, var -> valor mutável e global\*, let -> valor mutável e local\*, = -> atribuição.

const botao = document.getElementById('botao');



Para carregar o arquivo js no html, é recomendado colocar a tag <script src="caminho onde o arquivo js está armazenado"></script> antes da tag </body> e como teste dentro do arquivo js, usar o método alert('oi'); para exibir uma mensagem na tela.

Uma boa prática de estudo é descrever o que o código está fazendo através de comentários:

Múltiplas linhas:

/\*Texto descritivo que não vai ser interpretado como código\*/

Única linha:

//Texto descritivo que não vai ser interpretado como código

OBJETIVO 1 - quando clicarmos na seta de avançar temos que mostrar o próximo cartão da lista

- passo 1 - dar um jeito de pegar o elemento HTML da seta avançar.
- passo 2 - dar um jeito de identificar o clique do usuário na seta avançar.
- passo 3 - fazer aparecer o próximo cartão da lista
- passo 4 - buscar o cartão que está selecionado e esconder

OBJETIVO 2 - quando clicarmos na seta de voltar temos que mostrar o cartão anterior da lista

- passo 1 - dar um jeito de pegar o elemento HTML da seta voltar.
- passo 2 - dar um jeito de indentificar o clique do usuário na seta de voltar.
- passo 3 - fazer aparecer o cartão anterior da lista
- passo 4 - buscar o cartão que está selecionado e esconder

\*/

Estrutura:

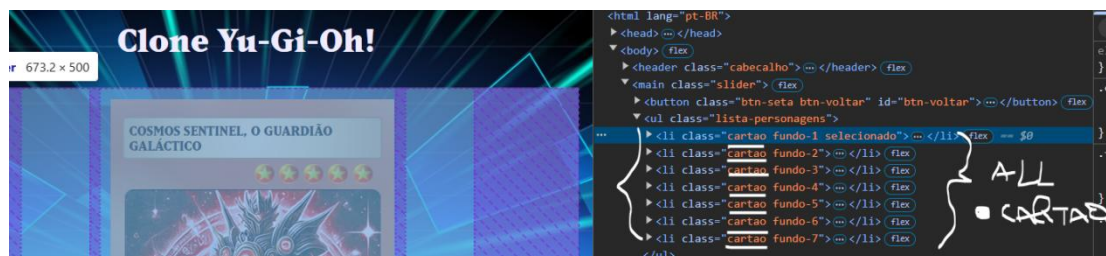
getElementById('nome-id'); -> resgata / captura o elemento pela identificação.

const btnAvancar = document.getElementById('btn-avancar') -> a variável btnAvancar recebe o elemento de botão com a identificação btn-avancar->

<button class="btn-seta btn-avancar" id="btn-avancar">



</button>



Document.querySelector('seletor');-> busca todos os elementos por seletor (.cartao) -> todos os itens de lista que contém a classe .cartao -> todos os cartões -> [ li.cartao.fundo-1.selecionado, li.cartao.fundo-2, li.cartao.fundo-3, li.cartao.fundo-4, li.cartao.fundo-5, li.cartao.fundo-6, li.cartao.fundo-7 ].

A lista em linguagem de programação é conhecida como Vetor/Array -> 0: li.cartao.fundo-1.selecionado, 1: li.cartao.fundo-2, 2:li.cartao.fundo-3, 3:li.cartao.fundo-4, 4:li.cartao.fundo-5, 5:li.cartao.fundo-6, 6:li.cartao.fundo-7, length: 7 -> onde cada item da lista é acessado por um índice | posição, ou seja, um conjunto de variáveis acessadas por um índice.

var cartaoAtual = 0; -> 0 pois a lista começa pela posição 0 onde vai estar o primeiro elemento, e cada iteração irá incrementar ++ o valor -> cartaoAtual = cartaoAtual + 1.

.addEventListener("nome do evento", function(){interação a ser feita}) -> adiciona uma escuta de evento ao botão seta, onde o evento é o clique do botão e a função vai ser a interação a ser feita.

Para remover a classe .selecionado do então cartão atual antes de avançar para o próximo, resgata o elemento pela classe const cartaoSelecionado = document.querySelector('.selecionado') e remove a classe cartaoSelecionado.classList.remove('selecionado').

Usando o cartaoAtual como índice da lista de cartões podemos sempre avançar para o próximo cartão [próxima posição] e acessando a lista de classes do cartão (li) e adicionar a classe selecionado que pelo css a opacidade é 100% e o cartão fica no topo (z-index:1).

Porém, quando chegamos na última carta [última posição==6] e incrementamos para 7, não haverá próximo e vai retornar erro, para evitar isso usamos a condicional if(condição) então se a posição atual == quantidade de elementos da lista - 1 então pare.

OB1P1 -> const btnAvancar = document.getElementById('btn-avancar')

OB1P2 -> btnAvancar.addEventListener("click", function(){});

OB1P3 -> const cartoes = document.querySelectorAll('.cartao');

```

let cartaoAtual = 0;
btnAvancar.addEventListener("click", function(){
  if(cartaoAtual === cartoes.length-1) return;
  OB1P4 -> const cartaoSelecionado = document.querySelector('selecionado');
  cartaoSelecionado.classList.remove('selecionado');
  cartaoAtual++;
  cartoes[cartaoAtual].classList.add('selecionado');
});

```

Porém quando chegamos na primeira carta [primeira posição==0] e decrementamos para -1 não haverá anterior e vai retornar erro, para evitar isso usamos a condicional if(condição) então se a posição atual == 0 então pare.

```

OB2P1-> const btnVoltar = document.getElementById('btn-voltar');
OB2P2 -> btnVoltar.addEventListener("click", function(){});
OB2P3 -> btnVoltar.addEventListener("click", function(){
  if(cartaoAtual === 0) return;
  OB2P4 -> const cartaoSelecionado = document.querySelector('.selecionado');
  cartaoSelecionado.classList.remove('.selecionado');
  cartaoAtual--;
  cartoes[cartaoAtual].classList.add('selecionado');
});

```

Para deixarmos o código mais legível, refatoramos o mesmo, transformando trechos **repetidos** de código em chamadas de funções() e juntando categorias iguais (variáveis) em linhas sequenciais.

```

const btnAvancar = document.getElementById('btn-avancar');
const btnVoltar = document.getElementById('btn-voltar');
const cartoes = document.querySelectorAll('.cartao');
let cartaoAtual = 0;

```

```

//onde faz tais ações, chama a função
esconderCartaoSelecionado();
mostrarCartao(cartaoAtual);

```

```
//no final do documento, coloca o mesmo conteúdo dentro da função
Function esconderCartaoSelecionado(){
....
}
Function mostrarCartao(){
...
}
```

Para adicionar o efeito de virar a carta:

Varre a lista de cartões e para cada cartão adiciona o evento de clique, então armazena a div de carta virada em uma constante, aplica o efeito de toggle\* no cartão para virar o cartão via css, com o cartão virado, aplica o efeito de toggle na div para mostrar fundo da carta via css e por fim aplica o efeito de toggle na descrição via css para não sobrepor as bordas do fundo da carta;

```
cartoes.forEach(cartao => {
  cartao.addEventListener("click", function(){
    const cartaVirada = cartao.querySelector('.carta-virada');

    cartao.classList.toggle("virar");
    cartaVirada.classList.toggle("mostrar-fundo-carta");

    const descricao = cartao.querySelector('.descricao');
    descricao.classList.toggle("esconder");
  });
});
```

Ao criar um novo projeto no github, o nome do repositório vai ser projeto-clone-yu-gi-oh.

Para usar o projeto para criar o github pages, vai até settings->pages->branch->main->save-> home page-> deployments-> github-pages->clicar no link gerado.

Para atualizações, home page->upload files.

\*linguagem de programação é a meio comunicação entre o homem e a máquina.

\*global -> funciona em qualquer lugar do javascript.

\*local -> funciona apenas onde foi criado dentro ou fora de método/classe.

\*toggle -> método da interface classList para excluir uma classe css caso ela já exista e quando for acionado, adicionar a mesma classe que foi excluída.