**JAVASCRIPT**

ECMAScipt é a padronização do JavaScript.

Para se usar o JS em line, é preciso usar a tag <script></script> no HTML.

**Comandos Básicos**

Alguns comandos básicos do JS:

window*.alert*('Minha primeira mensagem!') – Cria um alerta com uma mensagem na tela.  
window*.confirm*('Está gostando de JavaScript?') – Cria uma box de confirmação de OK ou Cancelar.  
window*.prompt*('Qual é seu nome?') – Cria uma box para inserir um texto.

Pode-se abreviar esses comandos tirado o window.

**Tipo de dados e variáveis**

Para identificar uma variável é preciso utilizar a palavra var, mas no JavaScript mais moderno, pode-se utilizar a palavra let.

Os identificadores podem iniciar com uma letra, $ ou \_ .  
Não podem começar como números.  
É possível usar letras ou números.  
É possível usar acentos e símbolos.  
Não podem conter espaços.  
Não podem ser palavras reservadas.

**Dicas**Maiúsculas e minúsculas fazem diferenças.  
Tente escolher nomes coerentes para variáveis.  
Evite se tornar um ‘programador alfabeto’ ou um ‘programador contador’.

**Tipos**

Interface gráfica do usuário

Descrição gerada automaticamente com confiança média

**Data Types**

Homem ao lado de tela de jogo de vídeo game

Descrição gerada automaticamente com confiança média

Existe um comando chamado typeof para saber os tipos primitivos das variáveis.

Uma variável não é definida para um certo tipo primitivo, ela apenas muda de acordo como é utilizada, por exemplo: pode se modificar um number para uma string.

**Manipulação de dados**

Para passar uma string para um número, é preciso usar um comando:

Number.parseInt(n) – Para números reais  
Number.parseFloat(n) – Para números flutuantes  
Number(n) – O JS vai escolher qual tipo irá ser necessário

Para passar um número para uma string. É preciso usar um comando chamado:

String(n)  
n.toString()

**Formatando Strings**

Para formatar uma string, é preciso usar o comando, ex:  
‘Eu estou aprendendo’ + s   
Para concatenação pode usar aspas simples ou duplas.

Mas existe uma forma melhor que se chama template string  
`Eu estou aprendendo ${s}`  
Para usar a template string, é preciso usar a crase.

**Alguns comandos que podem ser usados**s.length – Quantos caracteres a string tem  
s.toUpperCase() – Tudo para ‘MAIÚSCULAS’  
s.toLowerCase() – Tudo para ‘minúsculas’

**Escrever no documento HTML**

document.write – Podemos usar para escrever no arquivo html, nele é possível utilizar tags do html.

**Formatando números**

Vamos usar um exemplo com essa variável  
var n1 = 1545.5

Para deixarmos o números arredondados, usamos o comando toFixed(2) e colocamos o número de casas entre parênteses:  
n1.toFixed(2) = '1545.50'

Se quisermos trocar o “.” Pela “,”, podemos usar o replace para fazer a troca:  
n1.replace('.', ',') = '1545,50'

Para fazer a conversão de um número para um valor monetário, é preciso usar esse comando, mas caso queria outro moeda, é só trocar o seu nome:  
n1.toLocaleString('pt-BR', {style: 'currency', currency: 'BRL'}) = 'R$ 1.545,50'  
n1.toLocaleString('pt-BR', {style: 'currency', currency: 'USD'}) = 'US$ 1.545,50'

**Operadores**

Existem esses tipos de operadores:

**Aritmético**  
+ - Soma  
- - Subtração  
\* - Divisão  
/ - Multiplicação  
% - Resto da Divisão  
\*\* - Potência

Precedência  
1- ( )  
2- \*\*  
3- \*, /, %  
4- +, -

**Atribuição**Atribuição Simplesvar a = 5 + 3 = 8  
var b = a % 5 = 3  
var c = 5 \* b \*\* 2 = 45  
var d = 10 – a / 2 = 6  
var e = 6 \* 2 / d = 2  
var f = b % e + 4 / 3 = 3

Auto-Atribuções  
var n = 3 = 3  
n = n + 4 = 7 n+=4  
n = n – 5 = 2 n-=5  
n = n \* 4 = 8 n\*=4  
n = n / 2 = 4 n/=2  
n = n \*\* 2 = 16 n\*\*=2  
n = n % 5 = 1 n%=5

Incremento  
var x = 5 = 5 x++  
x = x + 1 = 6 x--

++x – Pre-Incremento   
x++ Pos-Incremento

**Relacionais**> - Maior que 5 > 2 - True  
< - Menor que 7 < 4 - False  
>= - Maior ou Igual 8 >= 8 - True  
<= - Menor ou Igual 9 <= 7 - True  
== - Igual 5 == 5 - True   
!= - Diferente 4 != 4 – False  
=== - Identidade 5 === ‘5’ - False - Confere se os número tem os mesmos tipos  
!== - Desigual Restrito 5 !== ‘5’ - True - Confere se os número tem valores ou tipos diferentes

Identidade  
5 == 5 - True  
5 == ‘5’ - True  
5 === ‘5’ - False  
5 === 5 - True

**Lógicos**A ORDEM DE EXECUÇÃO É ESSA:! - Negação - Não  
&& - Conjunção - E  
|| - Disjunção - Ou

Negação - Não

Texto

Descrição gerada automaticamente

Conjunção - E Disjunção - Ou  
Calculadora preta com letras brancas

Descrição gerada automaticamenteUma imagem contendo Texto

Descrição gerada automaticamente

Precedência  
**1-**  
1- ( )  
2- \*\*  
3- \*, /, %  
4- +, -  
**2-**>, <, >= , <=, == , !=, ===, !== - O que aparecer primeiro da esquerda para direita  
**3-**1- !  
2- &&  
3- ||

Tela de um aparelho celular

Descrição gerada automaticamente

**Ternário**teste ? true : false

Ex:  
media >= 7.0 ? “Aprovado” : “Reprovado” – Basicamente um if/else

**Entendendo o DOM**

O que é?  
Document Object Model  
É um modelo de objetos para documentos, é um conjunto de objetos dento do navegador que vai dar acesso aos componentes internos do Website.

**Árvore DOM**

Um exemplo de árvore dom

window   
location / document/ history  
html  
head / body

 Tela de computador com imagem de homem na frente de um laptop

Descrição gerada automaticamente

window – É a janela do navegador.  
location – É a localização do seu site, ou seja, url, página atual ou página anterior.  
document – É o documento atual.  
history – Vai guardado de onde você veio, de onde você vai.

Aqueles que estão acima, são os pais, já os que estão abaixo dos pais, é os filhos.

**Métodos de acesso dos elementos**

**Por marca**getElementsByTagName( )[] – Conseguimos selecionar vários elementos de uma vez.

**Por ID**getElementById( ) – Pegar o elemento por id do html.

**Por Nome**getElementsByName( )[] – Pegar os elementos pelo name do html, mas para pegar só um, é preciso usar o coxete para escolher

**Por Classe**getElementsByClassName( ) – Pegar os elementos pela classe do html.

**Por Seletor**querySelector( ) – Pode colocar entre os parênteses o nome do elemento mais a class ou id, ex: div#msg ou div.msg  
querySelectorAll( )

**Inner**

InnerText – Não pega elementos do html, ou seja, só pega o texto.

InnerHTML- Pega os elementos do html por completo.

Alguns Exemplos:

**var** d **=** window*.*document*.querySelector*('div.msg')

d.style.background **=** 'blue'

**var** corpo **=** window.document.body

**var** p1 **=** window*.*document*.getElementsByTagName*('p')[1]

*var d = window.document.getElementById('msg')*

*d.style.background = 'green'*

*d.innerText = 'Estou aguardando...'*

*window.document.getElementById('msg').innerText = 'Olá'*

**Eventos**

É possível utilizar diversos tipos de eventos, um desses é a interação do mouse, de acordo com o que o mouse fizer, vai acontecer algo.

Ex: mouseenter, mousemove, mousedown, mouseup, click, mouseout e etc.

Referência para eventos: <https://developer.mozilla.org/en-US/docs/Web/Events>

Podemos criar uma função para quando acontecer algo no html, executar um bloco de comando pelo JS.

Um ex:

Existem dois jeito, pode colocar um onclick=”clicar()” no html e chamar ele no Js com uma function clicar() {}, assim ele vai executar um comando quando clicar em tal elemento.

Também é possível fazer um listener para quando precisar executar várias functions, assim não vai poluir o código html, para fazer isso é preciso pegar um elemento por um seletor, logo após é preciso criar um addEventListener(‘função’, nome\_da\_função).

**Erros**

Para descobrir erros, é preciso abrir o inspecionar e ver o erro pelo console, geralmente o erro está na linha ou acima da linha detectada.