Diferenças e Anotações de HTML, CSS e JavaScript

Obs.: O FOCO DESTE DOC. É AS ANOTAÇÕES SOBRE JAVASCRIPT. SEJAM ELAS TRECHOS DE CÓDIGOS, BUGS DE TAIS COAMNDO, GANBIARRAS E ENTRE OUTRAS COISAS INTERESSANTES SOBRE O MESMO. SOBRE AS DEMAIS (CSS E HTML) ESTÃO NOS OUTROS REPOSITORIOS COM O MESMO TITULO DESTE DOC.

**HTML:**

HYPERTEXT MARKUP LANGUAGE

LINGUAGEM DE MARCAÇÃO PARA HIPERTEXTO

* HTML TEM FOCADO EM CONTEUDO (TEXTOS, IMAGENS, VIDEOS, TABELAS, LISTAS...)

**CSS:**

CASCANDING STYLE SHEETS

FOLHAS DE ESTILO EM CASCATA

* CSS TEM FOCADO EM DESIGN (CORES, SOMBRAS TAMANHOS, POSICIONAMENTO...)

**JS:**

JAVASCRIPT

* JS TEM FOCADO NAS INTERAÇÕES (MENUS INTERATIVOS, ANIMAÇÕES, POPUPS, VALIDAÇÕES...)

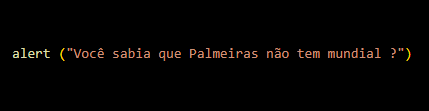
**ESTRUTURA BASICA DO JAVASCRIPT**

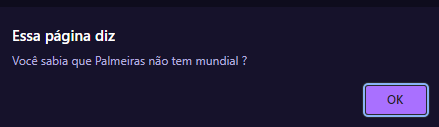
* JAVASCRIPT NÃO POSSUI UMA ESTRUTURA BAISCA COMO O HTML.

**SAIDA DE DADOS**

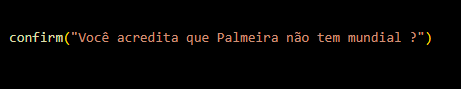
Existem diferentes tipos de caixa de diálogo, são elas:

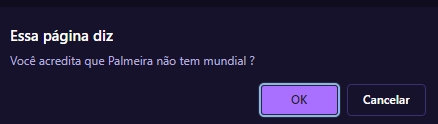
* alert( ) – “alert” é um dos tipos de saídas de dados, além disso é uma caixa de diálogos que aparece para o usuário e um recurso bastante usado no JS. Essa caixa de diálogo vem acompanhada de uma mensagem e um botão de confirmação (Ex.: OK). É muito usado em validações.



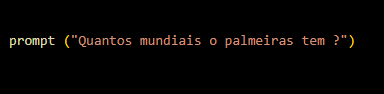


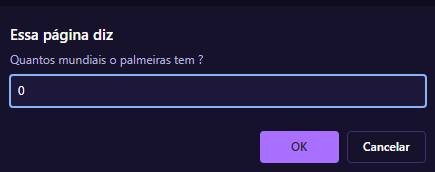
* confirm( ) – “confirm” é semelhante com o “alert”, porém o “confirm” não só vem acompanhado de uma mensagem e um botão de confirmação, como também um botão de cancelamento (Ex.: CANCELAR). É muito utilizado em sistemas que utilizem estruturas condicionais.



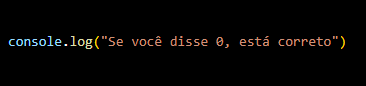


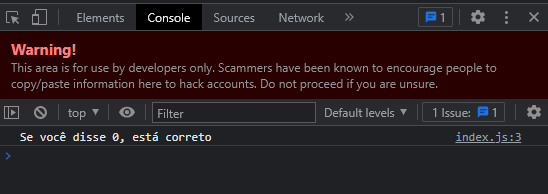
* prompt( ) – “prompt” serve tanto para saída de dados como entrada de dados, pois nele o usuário necessita inserir algum valor. O “prompt” vem acompanhado de um campo input (campo/local interativo para receber dados do usuário), botão de confirmação e um botão de cancelamento.





* console.log( ) – “console.log” é uma área do navegador que serve para visualizar erros, alertas de Javascript e entre outros.





**ENTRADA DE DADOS**

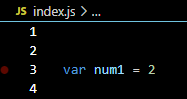
* prompt( ) – “prompt” serve tanto para saída de dados como entrada de dados, pois nele o usuário necessita inserir algum valor. O “prompt” vem acompanhado de um campo input (campo/local interativo para receber dados do usuário), botão de confirmação e um botão de cancelamento.

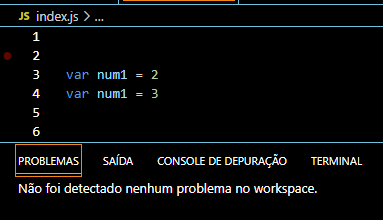
**VARIAVEIS**

* **Declarações de Variáveis**

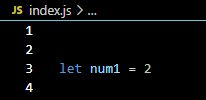
Existem três formar de criar variáveis, são elas:

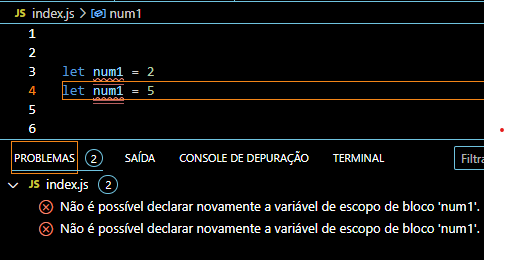
* var <nome da variável> - O “var” é uma das formas de criar variáveis no JS, porém ela te possibilita criar várias variáveis com a mesma nomenclatura, o que acaba prejudicando por não saber onde tais dados foram guardados, por possuir várias variáveis com o mesmo nome, além disso o conteúdo armazenado pode ser mudado/alterado.



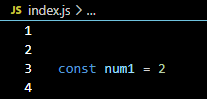


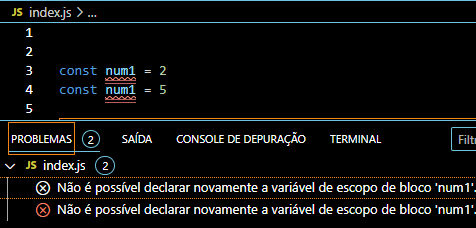
* let <nome da variável> - O “let” é outra forma de criar variáveis, diferente do “var” o “let” não deixa criar variáveis com o mesmo nome, elas tem que ter nomes diferentes, e assim como o “var” o seu conteúdo armazenado pode ser mudado/alterado.

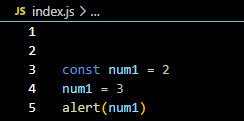


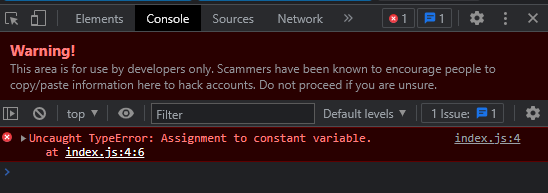


* const <nome da variável> - O “const” é umas das formas de criar variáveis, só que diferente do outros seu valor é constante, ou seja, seu conteúdo não pode ser alterado como os anteriores métodos de criar variáveis. Porém tem semelhança em relação com a “let” de não permitir criar variáveis com o mesmo nome.





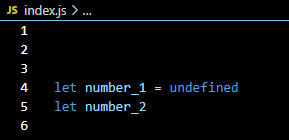




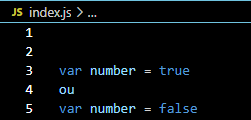
* **Tipos Primitivos**

Nessa parte veremos os métodos primitivos de inserir valores nas variáveis, ou seja, inserir diretamente os valores sem a interação do usuário, são elas:

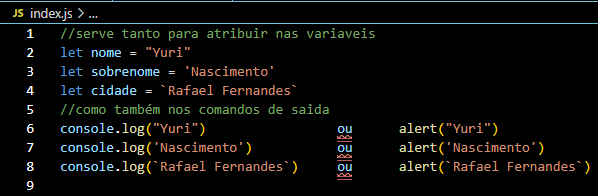
* undefined – quando falamos de “undefined” se trata de atribuir valores indefinidos, ou seja, há uma variável, mas não há valor atribuído a ela. Outra forma de fazer isso é criando apenas a variável, pois no JS as variáveis não armazenam lixo.



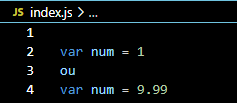
* booleanas: true/false – é um tipo de dado lógico que pode ter apenas um de dois valores possíveis: verdadeiro (true) ou falso (false). Elas são usadas para decidir quais trechos do código serão executados ou repetidas.



* string: aspas, aspas simples ou crase – são esses três tipos de inserir uma string em uma variável.



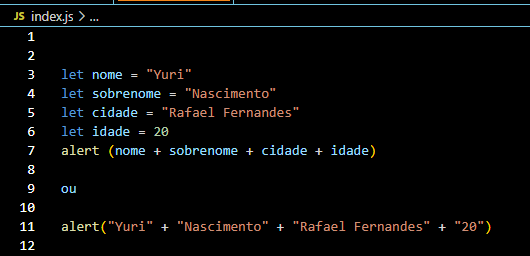
* number – você pode também armazenar números diretamente na variável, seja esse número do tipo de inteiro (int), real (float) e etc.

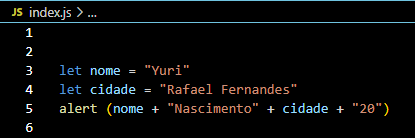


**CONCATENAR**

Concatenar é conectar duas strings assim como variáveis também através do operador de soma (+). Porém existem outras formas de concatenar que é através do cifrão ($).

* Concatenar por soma (+): a concatenação pelo sinal de soma (+) funciona com todas as formas de dizer que é uma string, ou seja, aspas (“”), aspas simples (‘’) e craze (``). Vale lembrar que na concatenação por soma as variáveis não ficam entre aspas, apenas as mensagens ficam entre aspas enquanto as variáveis ficam por fora.





* Concatenar por cifrão ($): a concatenação por cifrão ($) é outra formar de juntar essas vaiáveis ou strings em mensagem a ser exibida. A diferença dela para a de soma (+), além de deixar o código visualmente mais bonito, é que as variáveis vão estar dentro das aspas, ou melhor dizendo, crase (``). Na concatenação pro cifrão as variáveis e frases que você pretende exibir devem estar todas dentro da crase, ou seja, as demais como aspas e aspas simples não funciona nesse método de concatenação, além disso a crase tem que ser no mesmo sentido, da esquerda para a direita (``) e não (`´). Veja os exemplos abaixo.

