**CURSO EM VIDEO - JAVASCRIPT**

Os 5 compromissos:

- Não pule aulas;

- Anote tudo (de preferência num caderno);

- Use duas telas, sempre que possível;

- Praticar, praticar, praticar!

- Compartilhe o conhecimento adquirido.

**O QUE JAVASCRIPT FAZ?**

**Conceito: Cliente x Servidor**

Clientes são as pessoas que acessam a internet. Ela usam aparelhos como computadores ou celulares, por exemplo. Esses aparelhos possuem programas para acessar a internet, os navegadores (ex: Google Chrome).

Através dos navegadores, os clientes acessam a estrutura da internet (a nuvem), que os levam até os servidores, que são identificados através de URLs.

Dentro dos servidores há pastas contendo documentos, que são as páginas da internet. Os navegadores solicitam uma cópia destes documentos, para que retornem até os clientes, onde esta cópia é exibida. Desta forma, vemos a página desejada (website).

Este website (client side) é formado principalmente por meio de três linguagens:

- HTML - Responsável pelo conteúdo (informações);

- CSS - Responsável pelo estilo (design);

- Javascript - Responsável pela interação com o site.

O Javascript permite que se possa fazer quase tudo, inclusive modificar dados dos arquivos HTML/CSS.

Ex:

Para desabilitar o JS numa página da internet:

- Botão direito do mouse na página -> Inspecionar

- Separar a página de inspeção da página do navegador

- Selecionar "desabilitar JS"

Veja o resultado e entenda como o JS é usado para a interação na página.

**A EVOLUÇÃO DO JAVASCRIPT**

1970 - DARPA - Criação de uma rede de dados para evitar a perda de informações.

ARPANET - Junção de outras redes.

INTERNET

1993 - Tim Berners-Lee - CERN - Ligações entre documentos

HTML - HTTP - WWW

NCSA - Mosaic (1º navegador) - Mark Andersen

1994 - Jim Clark - Netscape (O navegador do "boom")

1995 - Brendan Eich - Linguagem Mocha - LiveScript

- Sun Microsystems - Linguagem Java (que passou a ter grande sucesso no mercado)

Por inveja do Java, Brendan Eich passou a chamar a LiveScript de JavaScript

1995 - Microsoft Internet Explorer (usando a base do Mosaic) - Linguagem JScript

1997 - Netscape procura a ECMA (a ISO da Europa), empresa de padronização - ECMAScript (versão padronizada do JS)

2002 - Fim da guerra Netscape x Internet Explorer

- Um grupo deixa a Netscape e funda a Mozilla - Firefox

2008 - a Google cria o Google Chrome

2009 - V8 - motor do navegador Google Chrome, de código aberto

2010 - A partir do V8, criou-se o NodeJS - máquina que roda JS fora do navegador.

**ECMAScript**

1997 - 1.0

1998 - 2.0

1999 - 3.0 - Tratamento de erros.

2009 - ES5 - compatível com quase todos os navegadores, compatível com JSON e com métodos de tratamento de arrays.

2015 - ES6 - declaradores let e const e strings de template

**TECNOLOGIAS**

jQuery - conjunto de bibliotecas - Mozilla

Angular - criação de aplicações web - Google

React - Facebook

Vue -

Electron - GitHub

Ionic - para dispositivos móveis

Cordova

**PRIMEIROS PASSOS EM JAVASCRIPT**

Antes de começar...

- Pratique!

- Persista

Como aprender?

- Assistindo os vídeos

- Praticando

- Lendo conteúdo relacionado

- Fazendo anotações

- Interagindo com outras pessoas que façam o curso ou tenham o conhecimento

- Criando projetos próprios

**BIBLIOGRAFIA**

- JavaScript: O Guia Definitivo (Ed. O'Reilly) - David Flanagan - CARO

- JavaScript: Guia do Programador (Ed. Novatec) - Mauricio Samy Silva (Maujor) - "barato"

- Guias de Referência - Mozilla/ECMA - grátis

**DÚVIDAS COMUNS:**

Aprender a programar ajuda a exercitar o cérebro, não importa a sua idade, o seu gênero ou onde você mora.

Para programar, não é necessário ter nenhuma experiência anterior, nem saber inglês ou matemática com maestria.

Fundamentos IMPORTAM! Aprender lógica de programação é necessário.

Todo iniciante passa por dificuldades.

Não existe a melhor linguagem do mundo. Torcidas ou idolatrias são inúteis e inócuas.

O ECMAScript é o ponto de partida para todo mundo. Usar bibliotecas é bom, mas tem de haver o passo-a-passo.

Aprender a programar é uma prática diária.

**PROGRAMAS NECESSÁRIOS PARA O CURSO:**

- Navegador (Chrome, Firefox, Edge, Opera, Safari...)

- Visual Studio Code

- NodeJS

**PRIMEIRO PROGRAMA EM JAVASCRIPT**

No Visual Studio Code:

(para criar o arquivo HTML, basta digitar "html" na área de edição do VSC)

*<!DOCTYPE html>*

*<html lang="pt-br">*

*<head>*

*<meta charset="UTF-8">*

*<meta http-equiv="X-UA-Compatible" content="IE=edge">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0">*

*<title>Meu primeiro programa</title>*

*<style>*

*body {*

*background-color: rgb(113, 113, 165);*

*color: wheat;*

*font: normal 20pt Arial;*

*}*

*</style>*

*</head>*

*<body>*

*<h1>Olá, Mundo!</h1>*

*<p>Já me livrei da maldição.</p>*

*<script>*

*window.alert('Minha primeira mensagem!')*

*window.confirm('Está gostando de Javascript?')*

*window.prompt('Qual é o seu nome?')*

*</script>*

*</body>*

*</html>*

*window.alert()* - Mostra uma mensagem na tela.

*window.confirm()* - Mostra uma mensagem interativa para o usuário, para ele escolher (botões Ok e Cancelar).

*window.prompt()* - Mostra uma mensagem para o usuário preencher um espaço de escrita.

Mas, uma vez digitados, para onde vão esses dados?

Inicialmente, para lugar nenhum. Portanto, é necessário alocar um espaço na memória para guardar as informações.

OBS: Formatos de comentários no arquivo da página de internet:

// - Comentário em uma única linha, em JavaScript.

/\* \*/ - Comentário em mais de uma linha, em JavaScript. Também serve para comentário em CSS.

<!-- --> - Comentário em HTML.

**VARIÁVEIS**

São espaços na memória, delimitados por nossos comandos.

Pode ser VAR ou LET.

Ex:

*var n1 = 5* (o símbolo após o nome da variável não é "igual", é "recebe".)

*var n2 = 8.5*

*var n3 = 15*

Esse apontamento das variáveis e seus valores é a ATRIBUIÇÃO.

Os espaços de memória alocados podem ter tamanhos diferentes, para alocar não apenas números, mas também strings (cadeias de caracteres).

Ex:

*var s1 = "JavaScript"* (atribuição com aspas duplas)

*var s2 = 'Curso em Vídeo'* (atribuição com apóstrofo ou aspas simples)

*var s3 = `Guanabara`* (atribuição com crase)

Cada símbolo usado pra atribuir as strings tem uma função diferente (a ver).

Regras para a nomeação de Identificadores (as variáveis, em JS):

- Podem começar com uma letra, cifrão ($) ou símbolo de sublinhado (\_).

- Não podem começar com números.

- É possível usar letras ou números.

- É possível usar acentos e símbolos.

- Não pode haver espaço no meio de um identificador (caso queira, pode ser usado o símbolo de sublinhado (\_) para representar o espaço).

- Não podem ser usadas palavras reservadas (palavras que o JS usa como comandos).

Dicas para criação de nomes de Identificadores:

- Maiúsculas e minúsculas fazem diferença.

- Tente escolher nomes coerentes para as variáveis.

- Evite se tornar um "programador alfabeto" (variáveis a, b, c, d, e...) ou "programador contador" (n1, n2, n3, n4, n5...)

Tipos mais comuns de variáveis:

number - números, sejam inteiros ou reais, portanto, aborda tanto o inteiro quanto o float.

string - uma cadeia de caracteres, que podem ser letras (palavras) como números (telefone, CPF...)

boolean - verdadeiro ou falso (true/false)

Apesar desses 3 tipos serem os mais frequentes, há outros tipos:

number:

infinity

NaN (Not a Number)

null

undefined

object

array (vetor)

function

Para sabermos qual é o tipo da variável, o JS tem um comando específico: typeof

**MANIPULAÇÃO DE DADOS**

No exercício 002:

*<script>*

*var nome = window.prompt('Qual é o seu nome?')*

*window.alert('É um grande prazer te conhecer, ' + nome + '!')*

*</script>*

A função do sinal de mais (+) é a CONCATENAÇÃO dos dados.

Como ele é usado para concatenação, a simples digitação dele no código para representar uma soma NÃO FUNCIONA.

Ex:

*<script>*

*var n1 = window.prompt('Digite um número: ')*

*var n2 = window.prompt('Digite mais um número: ')*

*var s = n1 + n2*

*window.alert('A soma dos valores é ' + s)*

*</script>*

Para converter os dados coletados de strings para números, usamos:

- *Number.parseInt(n)* - número inteiro

- *Number.parseFloat(n)* - número real

Na versão mais atual do JS, basta usar o comando Number(n), e o código verifica se o número é real ou inteiro.

Já no caso de conversão de número para strings, há duas formas:

*- String(n)*

*- n.toString()*

FORMATANDO STRINGS

Utilizando o node para mostrar as informações em JavaScript:

*var s = 'JavaScript'*

*'Eu estou aprendendo s'* // não faz interpolação

*'Eu estou aprendendo ' + s* // usa concatenação

Exemplo:

*> nome = 'Marcos'*

*'Marcos'*

*> idade = 38*

*38*

*> nota = 6.5*

*6.5*

*> nome*

*'Marcos'*

*> idade*

*38*

*> nota*

*6.5*

*> 'O aluno ' + nome + ' com ' + idade + ' anos de idade tirou a nota ' + nota + '.'*

*'O aluno Marcos com 38 anos de idade tirou a nota 6.5.'*

*>*

Esta forma de concatenação é muito trabalhosa e, por isso, nas novas versões do JS, usa-se template strings, da seguinte forma:

*`Eu estou aprendendo ${s}`*

ATENÇÃO! Para o uso da template string, não se coloca a string entre aspas, mas sim entre CRASES, e o símbolo ${} é o place holder, ou seja, o local onde vai estar a variável criada ou informada.

Exemplo:

*> `O aluno ${nome} com ${idade} anos de idade tirou a nota ${nota}.`*

*'O aluno Marcos com 38 anos de idade tirou a nota 6.5.'*

Existem outros comandos que podem ser feitos para formatar strings, por exemplo:

*s.length* // informa quantos caracteres a string tem

*s.toUpperCase()* // mostra os caracteres em 'MAIÚSCULAS'

*s.toLowerCase()* // mostra os caracteres em 'minúsculas'

Também existem comandos para formatar números, por exemplo:

*n.toFixed(2)* // mostra o número com duas casas decimais

*n.replace(‘ ‘, ‘ ‘)* // substitui o caractere entre as primeiras aspas pelo da segunda

*n.toLocaleString('pt-BR', {style: 'currency', currency: 'BRL'})*

Nesse comando, o "toLocaleString" serve para informar que está sendo adaptado a uma forma de digitação localizada, ou seja, pertencente a um lugar específico, e então é digitado o local, no caso, 'pt-BR', e depois é informado que o estilo da formação do número é financeiro (style: 'currency'), sendo que o formato financeiro é o do real brasileiro (currency: 'BRL').

O número pode ser mostrado em outras moedas do mundo, no formato localizado do Brasil, basta mudar, no comando "currency: " a moeda, como, por exemplo, 'USD' para o dólar estadunidense ou 'EUR' para o euro.

**OPERADORES EM JAVASCRIPT**

O JS possui algumas famílias de operadores, por exemplo:

- Aritméticos;

- De atribuição;

- Relacionais;

- Lógicos;

- Ternário.

Neste momento, vamos focar nos operadores aritméticos:

+ : operador de soma (Ex: 5 + 2 -> 7)

- : operador de subtração (Ex: 5 - 2 -> 3)

\* : operador de multiplicação (Ex: 5 \* 2 -> 10)

/ : operador de divisão real (Ex: 5 / 2 -> 2.5)

% : operador de resto da divisão inteira (Ex: 5 % 2 -> 1)

\*\* : operador de poteniação (Ex: 5 \*\* 2 -> 25)

Ordem de precedência dos operadores:

() > \*\* > \* / % > + -

Atribuições:

var 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 / e -> 3

Auto-atribuições:

var n = 3

n = n + 4 -> 7

n = n - 5 -> 2

n = n \* 4 -> 8

n = n / 2 -> 4

n = n \*\* 2 -> 16

n = n % 5 -> 1

Essas auto-atribuições podem ser simplificadas, da seguinte forma:

n += 4

n -= 5

n \*= 4

n /= 2

n \*\*= 2

n %= 5

Variáveis de Incremento:

var x = 5

x ++ -> 6

x -- -> 5

Operadores Relacionais:

> maior

< menor

>= maior ou igual

<= menor ou igual

== igual

!= diferente

Exemplos:

5 > 2 : true

7 < 4 : false

8 >= 8 : true

9 <= 7 : false

5 == 5 : true

4 != 4 : false

Operadores de Identidade:

== igual

!= diferente

=== idêntico

!== desigual restrito

Exemplos:

5 == 5 : true

5 == '5' : true

5 === '5' : false

5 !== '5' : true

Operadores Lógicos:

! negação (não)

&& conjunção (e)

|| disjunção (ou)

Exemplos:

! true : false

! false : true

true && true : true

true && false : false

false && true : false

false && false : false

true || true : true

true || false : true

false || true : true

false || false : false

OBS: Quando existem operadores aritméticos, relacionais e lógicos juntos na mesma expressão, a ordem de precedência é: aritméticos > relacionais > lógicos

Exemplo:

var a = 5

var b = 8

a > b && b % 2 == 0 : false

a <= b || b / 2 == 2 : true

OBS: Quando existirem operadores lógicos diferentes na mesma expressão, a ordem de precedência é: ! > && > ||

Exemplos:

idade >= 15n && idade <= 17 // a idade está entre 15 e 17?

estado == 'RJ' || estado == 'SP' // o estado é RJ ou SP?

salário > 1500 && sexo != 'M' // o salário é acima de 1500 e não é homem?

Operador Ternário:

teste ? true : false

Exemplo:

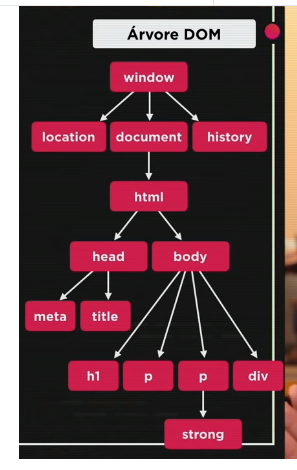
media >= 7.0 ? "Aprovado" : "Reprovado"

**DOM - Document Object Model**

Conjunto de objetos dentro do navegador que vai dar acesso aos componentes internos do website.

DOM funciona quando se roda JS dentro do navegador.

ARVORE DOM - árvore hierárquica



Para demonstrar a árvore de hierarquia dentro do website, basta acessar o arquivo html desta aula (ex005.html) e na área de script dentro de body, digitar “window”. Veja a quantidade de childs dentro da raiz do arquivo.

Ex:

*<script>*

*window.document.write(‘Olá, Mundo!’) - função para escrever*

*window.document.write(window.document.charset)*

*window.document.write(window.navigator.appName)*

*window.document.write(window.document.URL)*

*</script>*

Existem várias maneiras de se navegar e manipular os elementos da árvore DOM.

As formas de selecionar os elementos podem ser:

- por Marca - *getElementsByTagName()*

- por ID - *getElementById()*

- por Nome - *getElementsByName()*

- por Classe - *getElementsByClassName()*

- por Seletor – *querySelector() / querySelectorAll()*

Ex:

*<script>*

*var p1 = window.document.getElementsByTagName(‘p’)[0]* // Cria uma variável

*window.document.write(p1.innerText)* // Escreve o conteúdo do texto da variável

*document.write(p1.innerHTML)* // Escreve a variável com as formatações feitas pelos comandos de HTML

*window.alert(p1.innerText)* // O conteúdo da variável aparece numa janela de alerta

*window.document.write(‘Está escrito assim: ‘ + p1.innerText)* // Concatenação

*</script>*

Podemos usar a menção ao elemento selecionado para mudar seu estilo:

Ex:

*<script>*

*var corpo = window.document.body*

*var p1 = window.document.getElementsByTagName(‘p’)[0]*

*corpo.style.background = ‘black’* // Muda a cor do background da página

*p1.style.color = ‘blue’* // Muda a cor das letras do parágrafo referenciado

*</script>*

*Ex:*

*<script>*

*var corpo = window.document.bodyk*

*var p1 = window.document.getElementsByTagName(‘p’)[0]*

*var d = window.document.getElementById(‘msg’)* // Cria variável

*d.style.background = ‘green’* // Muda a cor do elemento selecionado

*d.innerText = ‘Estou aguardando…”* // Muda o texto exibido

*window.document.getElementById(‘msg’).innerText = ‘Olá!’* // Muda o texto do elemento selecionado, sem necessidade da variável

*</script>*

OBS: Na versão atual do JavaScript podemos omitir a menção ao “window” no script, sem problemas.

Ex:

*<script>*

*var corpo = window.document.bodyk*

*var p1 = window.document.getElementsByTagName(‘p’)[0]*

*var d = window.document.getElementsByName(‘msg’)[0]*

d.innerText = ‘Olá!’

*</script>*

Ex:

*<script>*

*var corpo = window.document.bodyk*

*var p1 = window.document.getElementsByTagName(‘p’)[0]*

*var d = window.document.querySelector(‘div#msg’)*

d.style.color = ‘blue’

*</script>*

OBS: # (tralha) serve para selecionar por ID; . (ponto) serve para selecionar por classe.