**Curso de JavaScript**

Modulo A

Node js executa o JS fora dos navegadores

**O que vou aprender no modulo A?**

1 – O que o JS é capaz de fazer?

2 –Dicas de aprendizagem da linguagem.

Não pular vídeo

Anotar tudo

Praticar muito

3- JS versus ECMAScript.

4 – Requisitos de software?

5 – Primeiro script emJS.

**Aula 1 O que o JS é capaz de fazer?**

**Cliente x servidor** – **Cliente** é quem usa o serviço, **Servidor** é quem fornece o Serviço...**dados**. E como você usa esse serviço: através de um navegador...

Um **Servidor** é igual ao seu computador, só que muito mais potente...quando você solicita um (dado, arquivo, etc....) Ele faz uma cópia e envia para você, guardando o original.

O seu navegador aí começa a trabalhar para gerar o efeito visual do seu serviço solicitado.

No caso existe tecnologia para **Cliente** e tecnologia para **Servidores...**

Na tecnologia para Cliente focamos no **html**, **css** e **js.**

**HTML –** Responsável pela parte contextual e nas marcações contextuais de um website, o conteúdo em geral...

**CSS –** Responsável pelo design desenvolvido na web, a parte visual, perfórma se, embelezamento da parte contextual...

**JS –** Responsável pelas de interações entre **Cliente** e Servidor, dá efeito, magia; modifica documento html e css...

Html e Css não são linguagem de programação, são tecnologias de construção de sites, não podemos dizer que programamos em html e css, podemos dizer que desenvolvemos em html e css; mas podemos dizer que programamos em JS por que JS é uma linguagem de programação...

No seu Navegador ao clicar com o botão direito você pode ir na opção inspecionar, na função DevTools ou ferramenta do desenvolvedor, na reticencia, função settings ou configurações tem a função desabilitar o JavaScript...você verá que o site já perdeu toda sua interação...

No DevTools na função console você pode dar comandos ao seu navegador como mudar cor de fundo, esconder logo etc. Porem essas funções somente funcionam na sua magna no tempo que você está acessando após atualização volta ao estado original...

Ctrl + l você limpa toda área de escrita no console...

**Aula 2 JavaScript. Como chegamos até aqui?**

O surgimento da internet e suas transformações, na briga de empresas por navegadores, cada navegador tem uma linguagem por trás dela, O google crome usa o JS ele é o navegador mais famoso do mundo na atualidade, com o google Chrome sendo código aberto qualquer pessoa pode mexer, um grupo de pessoas pegou ele modificou para ele rodar fora do google Chrome já que ele somente rodava no Chrome; Com isso o motor Chrome V8 deu origem ao nodeJs em 2010 que uma máquina que roda JavaScript fora do navegador , com isso o JavaScript pode ser usado tanto no lado do **Cliente** como do **Servidor.**

A linguagem ECMAScript é a versão padronização do JavaScript. De 1997 até 2018 a mais atual, a linguagem passou por muitas versões, no entanto várias ferramentas surgiram com a evolução do JavaScript, para conhecer essas tecnologias que levaram o JS para outro patamar primeiro você precisa conhecer o JavaScript...

**Tecnologias que surgiram com a evolução do JavaScript**

**Jquery – É um conjunto de bibliotecas, criado pelo mozila, facilitava muito o uso de interatividade no JS...**

Angula- **Surgiu em 2009 criado e mantido pelo google, ele facilita muito a criação de aplicações web é uma linguagem menos imperativa e mais declarativa semelhante ao sql para banco de dados...**

**React- lançado pelo facebook tem mais ou menos a mesma funcionalidade do Angula porem tem algumas flexibilidades maiores, existe o react nat. para aplicações de celular dar uma pesquisa mais para a frente porem primeiro aprender o js...**

**Vue- Criado por um ex programador do google em 2014, frame work poderoso segundo o criador já estava cansado de usar o Angula e criou uma versão bem melhor, depois que aprender JS dar uma olhadinha nele...**

**Elétron- ferramenta muito importante mantido pela GitHub, especializado na criação de interface gráfica para programinhas de interface gráfica, com certeza você já utilizou ou vai utilizar um programa feito em elétron, o próprio VS code foi feito em elétron assim como outras tecnologias...**

**Ionic – Também baseado em JS um sbk para poder criar aplicações para dispositivos moveis, Ionic que utiliza o NodeJs, vários desses usam o nodeJs para rodar o js fora do navegador...**

**Não entrar na onda de aprender qualquer dessas tecnologias sem aprender JavaScrip!**

**Existem também várias ferramentas com JavaScript para criação de jogos...precisa saber a base e ter a adição dessas bibliotecas e Sbks.**

**Phaser, PixiJS, Impact, MelonJS, CraftyJS**

**Aula 3 Dando os primeiros passos**

Como aprender a programar, assistindo as aulas on line, lendo conteúdo, ler manuais, ler guia de referência, ler livros e praticando muito...

Fazer as anotações, interações, fazer grupos de estudo, conversar com outras pessoas, participar de fórum, começar a pensar em projetos próprios, algumas situações que você precisa resolver no seu dia a dia e tentar aplicar a ferramenta de programação para resolver o problema

**Bibliografia**

JavaScript O guia definitivo

JavaScript guia do programador

Guia de referência da Mozilla e da ECMA (gratuito) <https://developer.mozilla.org/pt-BR/>

<https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>

Aula 4 Criando o seu primeiro Script

1. Normalmente a tag script é colocada dentro do tag body antes do fechamento da tag...também se faz tags script externos...
2. Comandos em JS sempre em letras minúsculas na maioria das vezes, porque tem diferença no JavaScrip entre minúsculas e maiúsculas...

**Comandos em JS**

1. window.alert (Uma janela que dá um aviso)
2. Window.confirm(semelhante ao alert que da opção de ok ou confirmar)
3. Window.prompt(interação com o usuário dando a opção de ele interagir)

Modulo B

Armazenando dados

Trata mentos de dados

Operações com dados

Aula 5 Variáveis e tipos primitivos

Tipos de dados e Variáveis – variáveis em JS tem uma pequena diferença.

Usar comentários na codificação para que ele seja bem documentado.

Com o passar do tempo vc não esquece.

No JS usa 2 simbologias (2 barras //) Comentario em 1 linha.

(/\* \*/) barra e asterisco mais de uma linha.

Comentário também ajuda outra pessoa a ver seu código de maneira mais claro...

Depuração= usa se para ver como seu comando está funcionando

**Armazenamento de dados**

**VARIÁVEIS**= imagina um estacionamento de carros, vc não pode simplesmente colocar todo tipo de automóveis lá se não congestiona, vc tem que delimitar o espaço criando vagas e numera-las, identificando elas, se vc cria 6 vagas para carros numere as, de 1 a 6, se criar 6 vagas pra caminhonetes identifica c1 a c6 e se for 6 para motos identifica com m1 a m 6. Vc organizou espaços específicos para diferentes tipos de automóveis...

Vaga 1= carro 1 (sinal de = igual é recebe em Js)

Fisicamente não tem como 2 carros usar o mesmo espaço, tem que tirar um e colocar o outro, vaga 1 = 2 ou também vc pode deixar a vaga vazia...vaga1 = null (nula, vazia, desocupada)

O mesmo se dá para computadores o estacionamento é a memória, os veículos são os dados, que no espaço em que estão também precisam ser delimitados, identificados como (variáveis) var1, var 2 e var 3 assim por diante e esses espaços ou variáveis recebem = valores em cada espaço ...5, 8,6, 15 esses valores recebem o nome de atribuição.

Essas delimitações também recebem tamanhos diferentes como nos estacionamentos que são chamadas de **string** s1, s2, s3 que recebem palavras (caracteres)

S1 = JavaScript

S2 = Sérgio dos Santos

S3 = Curso em vídeo

No Js a **string** tem que estar entre aspas (””) que tem 3 tipos, **aspas simples, aspas duplas, apócrifos** e a **crase**.

As variáveis precisam de um nome tem que ser nomeadas, identificadas

O nome das variáveis chama se de **identificador** que tem regras de nomeação

1. Pode começar com letra ou $ cifrão e sublinhado \_.

B- Não pode começar com número

C- É possível começar com letras e números

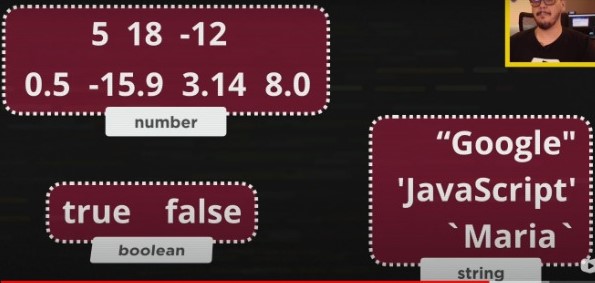
D- É possível usar acentos e símbolos

E- Não podem conter espaços

F- Não podem ser palavras **reservadas**- palavras que o Js usa como comando.

Sendo assim variáveis servem para guardar **DADOS**!

**Tipos primitivos principais**



**Números**

Para o JavaScript ele não diferencia números inteiros e reais para ele os inteiros e reais são do tipo Number.

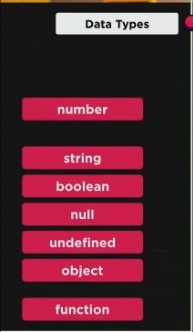
**Strings**

Um conjunto de números também pode ser uma string como seu cpf, rg, número de telefone.

**Boolean**

True/false (Verdadeiro e falso)

Com. exit você sai do node, com exit sai do node aberto no termino do code e Ctrl + l você limpa o terminal do node



Armazenamento de dados

Operações com dados

**Aula 6Tratamento dados**

window.alert // este comando envia uma mensagem!

window .confirm // este comando faz uma pergunta de confirmação!

window.prompt // este comando faz uma pergunta de resposta!

<! --Comentários em html se faz assim-->

/\* Comentário em CSS\*/

// Comentário em js

Var nome = window.prompt (‘Qual é o seu nome? ’);

No JS você também pode usar no lugar do variável var, let ou const.

Porem usaremos o var por uma questão de escopo, que ainda aprenderemos...

Var nome window.prompt(‘Qual é o seu nome?’)

Window.alert(‘É um grande prazer em te conhecer, ’ + nome + ‘! ’)

**O sinal de + é concatenação... junção nãoadição**

<script>

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

    var n2 = window.prompt('Digite outro número:')

    var s = n1 + n2

    window.prompt('A soma dos valores é ,' + s ) //Concatenação

   </script>

1. No Js se vc colocar 4+2 o resultado será 42 porque o sinal de + é uma concatenação (Relacionamento de ideias, fatos ou coisas entre si; ligação, encadeamento. Harmonização, conciliação.) O ***sinal de +*** serve tanto para **adição** como para **concatenação**.

String + string = concatenação

Number + Number = adição

O Js sempre te fornecera uma string. Nesse caso vc tem que transformar a strings para númber.

Como converter string para número? **Chamado de Conversão de tipos**. Existem várias formas de conversão de tipos...

1 – **Number.parseInt**(n) ou só **parseint** funciona igual. Para **número** **inteiro**.

2**- Number.pasefloat**(n) para **números reais com virgula,** float = ponto Flutuante o ponto que separa os números, 5.5. 10.4 e parse é converter, parear.

**<script>**

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

// + string convertida para Number

    var n2 = Number.parseInt(window.prompt('Digite outro número:'))

 // + string convertida para Number

**var** s = n1 + n2

    window.prompt('A soma dos valores é ,' + s )

 //Concatenação

    // (Number + Number) para adição

    //( string + string) para concatenação

**</script>**

Na conversão números reais **Number.parseInt (7 + 2 = 9), (7.5 + 2 = 9)**

**<script>**

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

    var n2 = Number.parseFloat(window.prompt('Digite outro número:'))

**var** s = n1 + n2

    window.prompt('A soma dos valores é ,' + s )

**</script>**

Para conversão para número inteiro **Number.paseFloat (7.5 + 2) = 9.5)**

Nas versões mais atuais do js **Number(n)** simplifica tanto para **números reais** como para **números inteiros** ele faz a conversão para o tipo definido. Porem essa função não elimina **parseInt** e **parseFloat** porque as vezes você precisa efetivamente tratar como real ou inteiro uma função.

**<script>**

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

    var n2 = Number.(window.prompt('Digite outro núme~ero:'))

**var** s = n1 + n2

    window.prompt('A soma dos valores é ,' + s )

**</script>**

Na versão mais atual do JS na conversão de tipo **Number. (7 + 2 = 9), (7.5 + 2 = 9.5)**

**Número para String**

Na conversão de **Number** para **string** também existe 2 maneiras.

1. Usa se os comandos **String (n)** ou o valor que você quer converter ele vai gerar uma **string**.
2. **n.tostring()** coloque o que deseja converter com essa função, caso seja necessário para frente.

    window.prompt('A soma dos valores são,' + s.toString()))

window.prompt('A soma dos valores são,' + string(s))

**Formatação de Strings**

‘O aluno’ + nome + ‘com’ + idade + ‘anos tirou a nata’ + nota

O aluno Sérgio com 41 anos tirou a nota 5.5

Porem nas formais recente de JavaScript tem formas mais legal de trabalhar...

**Técnica Template String**

var s = ‘JavaScript’

‘Eu estou aprendendo s ’ //Não faz interpolação

‘Eu estou estudando’ + s //Usa se a concatenação

`Eu estou aprendendo ${s}`// Usa se template string **(Com crase no lugar de aspas)**

Eu estou aprendendo ${s} //${s} nomenclatura placeholder

//Usa se formatador de String, no Template string se usa crase em vez de aspas simples

var nome = ‘Sérgio’

var idade = 40

var nota = 5.5

‘O aluno’ + **nome** + ‘com’ + idade + ‘anos, tirou a’ + nota

O aluno Sérgio com 1 anos, tirou a nota 5.5

//Modo antigo e mais complexo

`O aluno ${nome} com ${idade} anos tirou a nota, ` ${nota}

O aluno Sérgio com 1 anos, tirou a nota 5.5`

//com placeholder mais simplificado.

**Aqui no VsCode**

Window.alert(`A soma entre **${n1}** e **${n2}** e igual a **${s}`)**

Lembrando **aspas simples ou duplas** para **string simples**.

**Template string** para **interpolar** e **simplificar** a codificação, usando **placeholder com crase**.

**var** **s** = ‘JavaScript’

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

**s.toUpperCase()** //Transforma tudo em letra ‘MAIÚSCULAS’

**toLowerCase()** //Transforma tudo para ‘minúsculas’

**<script>**

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

document.write(`Seu nome tem ${nome.length} letras **<br/>`)**

document.write(`Seu nome em maiúsculas é ${nome.toUpperCase**()}<br/>`)**

document.write(`Seu nome em minúsculas é ${nome.toLowerCase()}`)

**</script>**

**Resultado do código acima no navegador**

Seu nome tem 6 letras.  
Seu nome em maiúsculas é SÉRGIO.  
Seu nome em minúsculas é sérgio.

/\*Essa é a maneira correta de se configurar fonte no css\*/

**<style>**

**body {**

            font: normal 20pt Arial;

**}**

**</style>**

//Usa se o<br> para quebra de linha e também pode col

//Para abrir o node, ctrl + ‘

**Formatação de números**

**Node**

var n1 =1545.5

n1

1545.5

**Se querer escrever com 2 casas decimais**

n1.toFixed(2)

1545.50

**Se querer mudar ponto para virgula**

n1. toFixed(2).replace(‘.’ , `,`) //Aspas depois virgula

1545,5

**Como fazer um comando que mostra o salário de uma pessoa?**

N1.toLocaleString(‘pt-BR’, {style: ‘currency’, currency: ‘BRL’})

Em Real: R$ 1545,50

N1.toLocaleString(‘pt-BR’ , {style: ‘currency’, currency: ‘USD’})

Em dólar: U$$ 1545.50

n1.toLocaleString('pt-BR', {style: 'currency', currency: 'EUR'})

Em Euro: '€ 1,545.50'

**Aula 7 Operadores em Js parte 1**

O JS possui várias famílias de operadores, nesse curso vamos focar nesses:

****

Nessa aula vamos focar em 2 e de atribuição

**Operadores Aritméticos**

(+, -), (\*), (/)**,** Resto da divisão (%) exp. 5%2 = 1🡨 resto da divisão, (\*\*)

Adoção, subtração, multiplicação, divisão, resto da divisão e expoente.

Os símbolos são operadores e os números são os operandos que fazem os operadores operarem...todos esses operadores são chamados de binários que precisam de 2 operandos

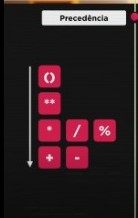
5/2

½ sobra 1 = resto da divisão inteira

Na programação a divisão vem primeiro 5+3/2 = 6.5, se você quer a soma primeiro (5+3)/2=4, tem que colocar entre parentes

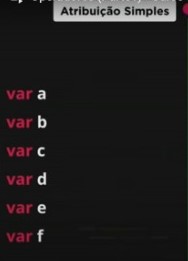
Quando vc programa a linguagem só mostro o erro se for erro sintático de escrita, as vezes quando vc não recebe mensagem de erro não significa que este certo, cuidar para não dar o comando errado...

**Com isso vamos a ordem de precedência dos operadores...**

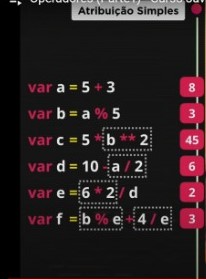


Quando \* / % estiverem numa mesma conta se resolva da esquerda para a direita...

**2 Atribuição Simples**



**Ordem de precedência com atribuição simples**

-

Pode se fazer também **o auto atribuições**

Onde você atribui valores a própria vaiável**...**tomando o valor a cada atribuição determinada...



3

7

2

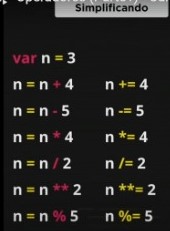
8

4

16

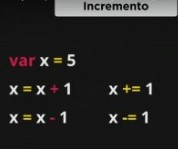
1

Pode também **simplificar esse auto atribuiçõe**s; quando você pega uma variável n simples e dá atribuições a ela; você pode reinscrevê-la simplificando como a baixo...



Porem se uma variável **n** receber uma mesma variável **n** ela poder ser simplificada, como exemplos de simplificações acima...

Outro auto atribuição ainda simplificadora ainda maior que é chamada de **operador de incremento...**





Nas linguagens de programação tem o incremento e o pré-incremento, por exemplo.

Var n = 10

n

10

n++

10

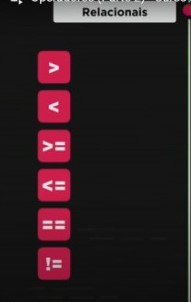
n

11

Ele mostrou o valor de **n++** depois pq eles vieram depois, nas linguagens de programação tem como ele vir antes colocando **++n** chama se **pré-incremento...**não significa que o n++ está errado, é uma questão de ordem existe o **pós-incremento e** o **pré-incremento...**

**Aula 8** **Operadores parte 2**

**3 relacionais**



Uma técnica para saber qual o < maior e o > maior, faz um traço nos símbolos um vira o 7 e o outro o 4, e o 4 é

menor e o 7 maior...



Maior

Menor

Maior ou igual

Menor ou igual

Igual

Não igual ou diferente

De todas essas expressões com relacionais o resultado será sempre booleano verdadeiro (V) ou falso (F).

**Exemplos**

Preço **>=** 200.50 //O preço é maior ou igual à 200.50?

Idade **<** 18 //A idade é menor que 18?

Curso **==** JavaScript //O curso é igual a Java Script?

n1 **!=** n2 //O primeiro número é diferente do segundo?

Quando houver operadores aritméticos e relacionais na mesma linha primeiro se resolve os aritméticos

Var a = 8

Var b = 15

A < b ==true

a <= b +10== false

Operadores relacionais de Identidade o JavaScript não compara o

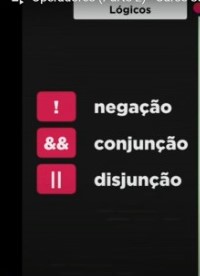
tipo 5 == 5 true o que o script faz e analisar se o 5 tem o mesmo valor que o outro, não o tipo, mas a grandeza, valor...

5 ==´5’ true

No Caso de comparação de **identidade restrita** **usa se o símbolo** **=== desigual restrito**, ele analisa o **valor e o tipo** **5 === ‘5’ false,**

**5===5 true**

**4 Lógicos**



**Os lógicos são 3**

1. ! Exclamação que representa negação **(não),**

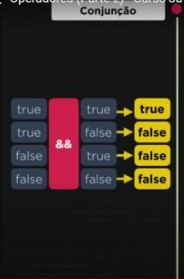
Tratado como operador unário ele só tem um operando **True e False**. **Algo que não é** **verdadeiro** **é** **falso** e **algo que não é** **falso** **é** **verdadeiro**...

**Exemplo:** Eu quero uma caneta, mas ela **não** pode ser azul, é só você entregar uma caneta qualquer **menos** uma azul...



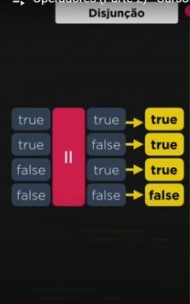
1. && e comerciais, significa conjunção **(e)** Operadores binário com 2 operadores lógicos um de cada lado dando o resultado lógico de cada expressão,
2. **Exemplo:** Eu quero uma caneta azul **e** uma vermelha, **eu só vou ficar reluz se você me entregar as duas...** pq eu quero uma azul **e** uma vermelha

Só satisfaz se as duas expressões forem verdadeiras



1. || Pappi disjunção **(ou)** também é um operador binário com 2 operadores lógicos um de cada lado, dependendo das variações também se obtém um valor lógico...

**Exemplo:** Eu quero uma caneta azul **ou** uma vermelha, **qualquer uma que lhe der você fica feliz.** Se lhe entregarem nenhuma ou de uma cor que você não quer aí dá problema...



Basta de tenha um valor **verdadeiro** que os outros são **verdadeiros** a menos que os dois sejam **falsos**...

Se numa mesma expressão tiver um **e** um **não** ou um **ou** a ordem de execução é **1º não** depois **e** depois o **ou**...

**Aritméticos primeiro** depois de todos, os **relacionais. que não tem ordem de precedência** quem aparecer primeiro será feito primeiro, leitura da esquerda para a direita e após os **lógicos que tem ordem de precedência** primeiro o **não,** depois o **(e &&) e após (ou ||)...**



**Exemplos:**

**Idade** **>=** **15** **&&** **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 homem.**

**5 Ternario**

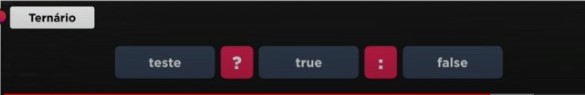
****

Ternários são compostos por 2 símbolos que aparecem na mesma expressão **(?) (:)** isso é 2 pontos. Se chama ternário porque tem **3operanos**.

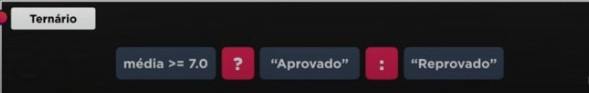
**Teste ? true : false**

**Teste:** é um teste lógico que dá verdadeiro o falso

O símbolo de interrogação vai quando o teste for verdadeiro. E no final o que vai acontecer quando o teste lógico for falso...



**Exemplo...**



**var media** = 5.5

* **Media** **>** 7 ? **Aprovado** : **Reprovado**

== **reprovado...**

Sintaxe da expressão se a média for maior que 7 recebe aprovado se for menor recebe **reprovado**

**Var x** **=** **8**

**Var res** **=** **x %** 2 **==** **0** ? **5 :** **8**

Sintaxe da expressão, se a expressão receber 0, for verdadeira **então**: ela recebe 5 e **Se?** Recebe 9

**X** **==** **8**

**Res** **==5 Verdadeiro**

Var idade = 19

Var r = idade >= 18 ? ‘MAIOR’ : ‘MENOR’

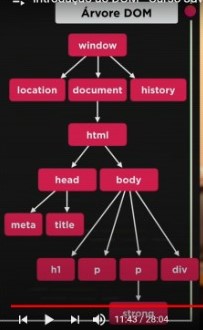
Expressão se a idade for maior ou igual 18 então ela recebe maior se não recebe menor...

A idade é maior que 18 verdadeiros = **MAIOR**

**Aula 9 Modulo C-Entendendo o DOM**

**Document Object Model ==** DOM / Modelo de objetos para documentos basicamente um conjunto de objetos dentro do seu navegador que dá acesso aos componentes internos do seu website ele só funciona quando rodamos JavaScript dentro do navegador...compreender e fazer a arvore DOM

Arvore DOM



A arvore **dom** começa pela raiz dentro do navegador essa raiz chama se window / janela. Tudo dentro do JavaScript está dentro do window, dentro do window contém vários outros objetos, exemplos **location. Document, history...**

**Location**-diz sobre a localização do website...

**Document** –dentro do document existe o html-dentro do html tem head e Bady quem está dentro ou a baixo é filho e quem está a baixo ou fora é pai...dentro de head tem meta, title e outros. Dentro de Bady tem h1,p, div e outros mais...

**History**—diz se sobre de onde vc vem e para onde vc vai facilitando a navegação...

Pode se ter acesso a vários componentes utilizando diretamente dom dentro do JavaScript

Pode se navegar dentro da arvore DOM da maneira que achar melhor e por isso existe várias maneiras de navegar entre os **elementos** é todo e qualquer coisa que aparece na arvore DOM são ditos elementos e vc pode selecionar esses elementos para poder navegar dentro da arvore DOM, e existem vários métodos para isto...exemplos abaixo.

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

**UTF-8**

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

**Netscape**

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

**http://127.0.0.1:5500/aula6/ex005.html**

**window. Document.write('Olá, Mundo!')**

**Olá, Mundo!**

Pode se navegar dentro da arvore **DOM** da maneira que você achar melhor entre os elementos, elementos são qualquer coisa existente dentro da arvore **DOM.**

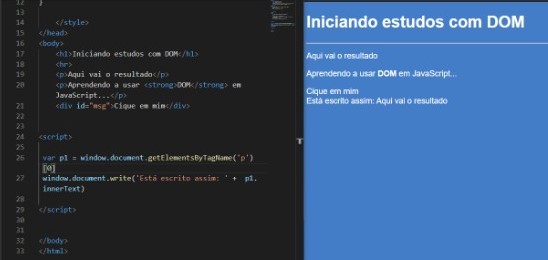
Você pode selecionar esses elementos para poder navegar dentro da arvore **DOM.** e existem vários métodos para isso.

Abaixo estão os 5 métodos de acesso:



Manipulando DOM

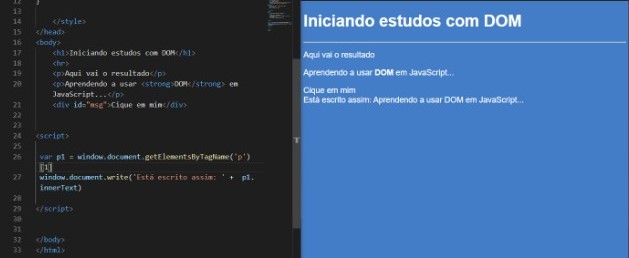
Usando o elemento por marca, tag, você pode selecionar mais de 1 objeto...por exemplo



Como havia 2 parágrafos você pode selecionar qual deles chamar, começando pelo zero, veja o que acontece quando se troca o zero pelo 1 em colchetes, muda a escrita no navegador, o computador lê zero como um...

Innertext é o texto que está dentro dos parágrafos...

Na programação o número 1 é o zero (0)

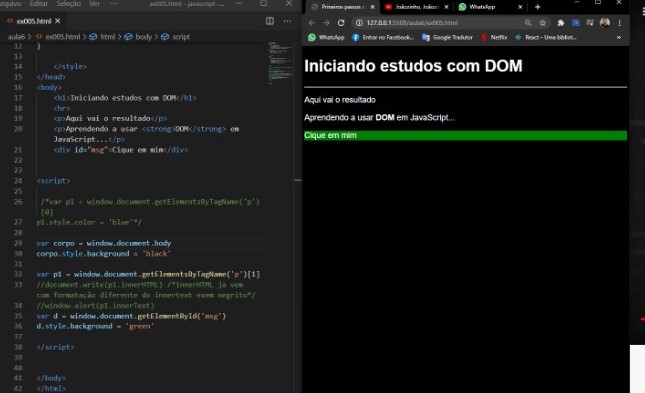


Pode se fazer o acesso a todos os componentes utilizando getElementsByTagName, note que essa função está no plural pq tem 2 parágrafos...

Mas essa não é a única técnica, também pode se utilizar objetos quando site for mais evoluído, por id, nome se tiver uma propriedade name, quando tiver uma propriedade no plural usar o [colchete] para selecionar o objeto.

Quando você formatar um parágrafo, ele te dá as 2 opções Innertext, ele não puxa algumas formatações. Como negrito por exemplo, então você tem a opção de InnerHTML, que já traz todas as informações.

**Por getElementById**



Innertext = ‘Estou aguardando...’ mudando a escrita dentro da div. Pq foi mandada por JavaScript.

Também escrever o comando completo...

Window.document.getElementById(‘msg’). InnerText = ‘Olá!’

Se você trocar a id da div por nome ou classe também funciona.

<div name=’msg’>clique em mim</div>

Var d = window.document.getElementByName(‘msg’)[0]

Innertext = ‘Olá! ’ ..também por classe, dependendo da situação do elemento...tudo mesmo princípio, a forma de acesso é a que você prefere

Aprendendo o querySelector ()

**QuerySelectorAll ()**

Nova forma, recomendada pela maioria dos manuais...

Toda div é representada por uma # hechtag e toda classe é representada por um (. ) Pontinho...

 <div id="msg">Cique em mim</div>

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

d.style.color = 'blue'

Cique em mim

O querySelector fica mais prático quando se faz alterações mais rápidas no site...querySelector é mais recente navegadores mais antigo talvez não terão suporte para ele...

Evento em JavaScript - Os **eventos** são basicamente um conjunto de ações que são realizadas em um determinado elemento da página web, seja ele um texto, uma imagem, ou uma div, por exemplo. Muitas das interações do usuário que está visitando sua página com o conteúdo do seu site podem ser consideradas **eventos**.

**Funções** são blocos de construção fundamentais em **JavaScript**. Uma **função** é um procedimento de **JavaScript** - um conjunto de instruções que executa uma tarefa ou calcula um valor. Para usar uma **função**, você deve defini-la em algum lugar no escopo do qual você quiser chamá-la.

**Eventos DOM Aula 10**

Evento é todo aquilo que possa acontecer com um elemento, por exemplo uma div.-

Com apenas o movimento do mouse dentro de um elemento por exemplo uma div você pode disparar uma série de eventos (efeitos)



Pesquisar por Event referência na web...

<https://developer.mozilla.org/pt-BR/docs/Web/Events>

Uma função ou uma funcionalidade é um conjunto de códigos(conjunto de linhas) que serão executadas somente quando o evento ocorrer, por exemplo esse conjunto de linhas ou códigos são chamados de blocos...

<div id="area" onclick="clicar()" onmouseover ="entrar()" onmouseout="sair()">

Interaja...

</div>

**Ao invés de fazer isso no html vc pode puxar os elementos direto do script assim:**

var a = window.document.getElementById('area')

        a.addEventListenner('click', clicar)

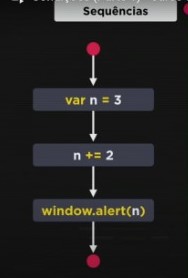
        a.addEventListenner('mouseover', entrar)

        a.addEventListenner('mouseout', sair)

**Se você errar código em JS, olhar no devetools.**

Dica: se você digitar div[#area](https://www.youtube.com/hashtag/area) e apertar tab ele cria automático <div id="area"></div>. Isso funciona tbm para classes e para tags personalizadas!

Modulo D Condições em JavaScript Aula 11



Essa é uma sequência em JS quando a função é executada de cima para baixo seguindo ordem de leitura...

Porem tem alguns momentos que vc não quer executar as funções na ordem de sequência, nesse caso tem as **condições**

Condição é quando uma função começa a ser executada e você precisa de opções para ir por um caminho ou outro, ele te dá a opção de escolha, caso aconteça uma coisa vc vai pôr um caminho, caso aconteça outra vc vai por outro caminho...esse desvio que acontece de resolver o problema e continuar o caminho chama se de **desvio condicional;** esse desvio faz parte dos fundamentos da programação, existem vários tipos de desvio condicional.

Uma condição começa a ser representada por **if = se,** A condição if é criada por um bloco- um bloco é uma abertura e fechamento de chaves { } . Uma condição pode ter vários blocos ou comandos

**If** (condição) {

|  |
| --- |
| true |

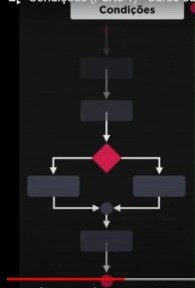
}

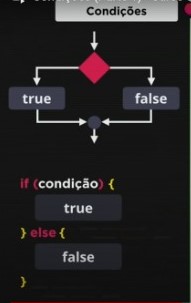
E o outro bloco é o :

**else = senão**{

|  |
| --- |
| false |

}





Condições é uma estrutura de controle muito importante pra programação...

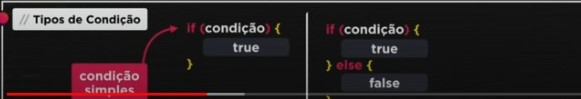
Existem vários tipos de condições, vamos ver uma segunda maneira:

A condição que só tem um tipo de bloco: if (condição) {

true

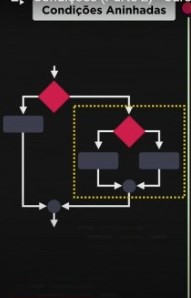
}

Essa estrutura por ser simples é chamada de **condição simples!**

****

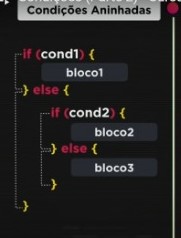
Próxima aula 12, fazer exercício 9 em forma de site

**Aula 12 Condições (Parte 2)**

****

Condições aninhadas é quando você pega uma condição composta e põe outra; **Se** (if) uma condição for verdadeira ele executa **se não(else)** coloca outra condições dentro dela, isso amplia muita as minhas possibilidades...

E assim por diante, você pode fazer vários níveis de condições uma dentro da outra.



Na endentação que vem de indent que significa recuo a execução é feita por blocos o que for verdadeiro. E assim por diante aumentando níveis no aninhamento.

Essa ilustração chama se condição aninhada!

var idade = 22

if (idade < 18) {

    console.log('Não vota')

}else{

    console.log('Vota')

}

Nesse caso a codificação está escrita totalmente correta, porem isso não significa que o seu programa está correto, por que no brasil uma pessoa com 16 anos pode votar**. O código não tem erro sintático mas tem erro de logica...**

**var idade = 22**

**if (idade < 16) {**

**console.log('Não vota')**

**}else {**

**if (idade >= 16 && idade < 18) {**

**console.log('Voto opcional')**

**}**

**}**

Na codificação acima também tem um erro sintático, Menor que 16 ok,, maior ou igual a 16 ok, menor que 16 ok, e se a idade for maior que 18, não tem possibilidade foi tirada do programa...

Por exemplo se a idade é menor que 16 obviamente que ela é maior que 16, porem essa parte do código **(idade >= 16)** é desnecessária se tirarmos o programa funciona bem...

Outra coisa para simplificar como else e if e muito comum vc pode fazer dessa maneira.

var idade = 22

if (idade < 16) {

    console.log('Não vota')

}else  if ( idade < 18) {

   {

        console.log('Voto opcional')

        }

    }

Ai você^ economiza a escrita de um bloco.

**Essas variáveis servem para nos dão a hora atual do sistema. Porem temos que cuidar pq essa é a hora que está rodando seu script; se seu script estiver rodando no cliente é a hora do cliente, se você estiver rodando js para rodar no servidor é a hora do servidor**

var atual = new Date()

var hora = atual.getHours()

console.log(`Agora são exatamente ${hora} hora:`)

if (hora < 5) {

    console.log('Boa Madrugada!')

} else if (hora <= 12) {

    console.log('Bom dia!')

} else if (hora <= 18) {

    console.log('Boa tarde!')

} else {

    console.log('Boa noite!')

}

**Condição Múltiplas**

Muito útil para trabalhar valores fixos, não servem muito para intervalos de valores como else e if como acabamos de ver, ela não tem a possibilidade de sim e não, mas de outros valores e valores fixos...



Recomendadas para condições pontuais, especificas

****

O comando s**wifch** não é uma **condição** é uma **expressão**, e o **switch** tem um **bloco** relacionado, **bloco** **= abre e fecha chaves** { }, dentro do **bloco** que tem um **case** em cada **valor**, Se a **expressão** for o 1º **valor** faz o 1º **valor**, se for o 2º faz o 2º e assim por diante, e termina com o **default** se nenhuma das **expressões** for escolhida ele executa o **default**... Para cada ***case*** tem um **bloco.** O **defaut** é o padrão se nenhuma das **expressões** acima forem satisfeitos.

O **break é** **obrigatório** para cada case, O JS tentar a **expressão** pula para a **expressão resultante** ele executa os **comandos** do **case resultante** que pode ser 1 ou pode ser vários e depois ele para no **break** e é desviado para o **fluxo da expressão**, se o **break** não existir ele continua executando todos os **cases** um a um...Por essa razão o **break é obrigatório...**

var agora = new Date()

var diaSem = agora.getDay()

/\*

0 = Domingo

1 = Segunda

2 = Terça

3 = Quarta

4 = Quinta

5 = sexta

6 = Sábado

console.log(diaSem)\*/

switch(diaSem) {

case 0:

console.log('Domingo')

break

case 1:

console.log('Segunda-Feira')

break

case 2:

console.log('Terça-Feira')

break

case 3:

console.log('Quarta-Feira')

break

case 4:

console.log('Quinta-Feira')

break

case 5:

console.log('Sexta-Feira')

break

case 6:

console.log('Sábado')

break

default:

console.log('[ERRO] Dia Invalido')

break /\*Break meio que opcional, caso à alguma problema posteriormente\*/

}

Se você der esse comando para o JS ele te mostrará o número da semana, sendo que para ele o número 1 é o Zero

console.log(diaSem) – Então com esse comando se hoje fosse segunda-feira ele mostraria o numero 1..

0 = Domingo

1 = Segunda

2 = Terça

3 = Quarta

4 = Quinta

5 = sexta

6 = Sábado...

Se você forçadamente pôr o dia da semana aleatório var diaSem = 5 na expressão switch e dará sexta-feira, e se você não colocar o break nos outros case, ele dará sexta, sábado e o chamando do default que na expressão acima é

ERRO] Dia Invalido' dia 7

Aula 13