JAVASCRIPT


0 == "0"
true
JS
0 == []
true
JS
So, if
0 == "0"
and
0 == []
then surely "0" == []
That makes sense to me.
"0" == []
JS
false
JS

SOBRE

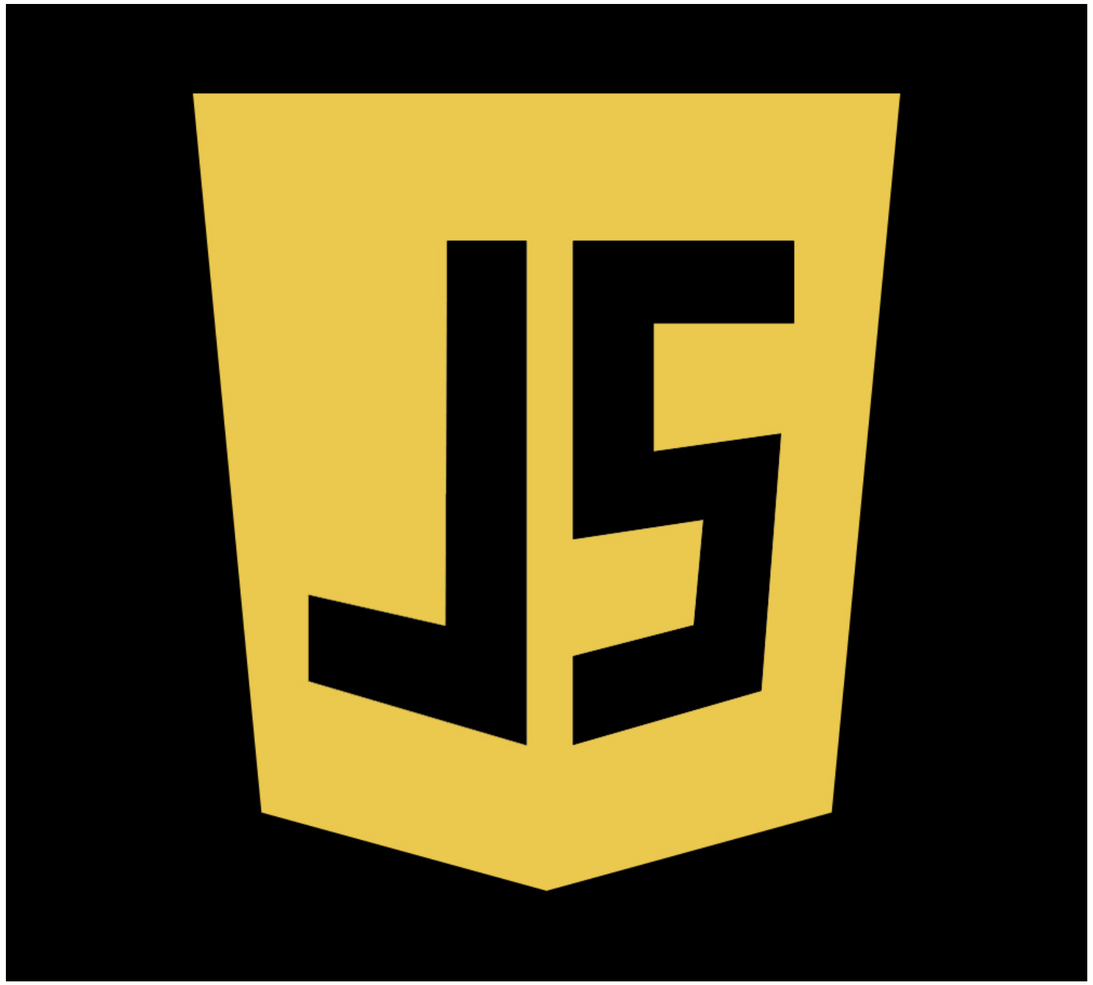
Curiosidade

Em 2021 segundo o StackOverflow, Javascript foi a lingaguem mais popular.

Lançado em 4 de dezembro de 1995.

Tiveram avanços incríveis desde 2015, mas boa parte dessa melhora na linguagem dá-se pela evolução dos proprios computadores.

Javascript é a única linguagem que faz o que ela faz!



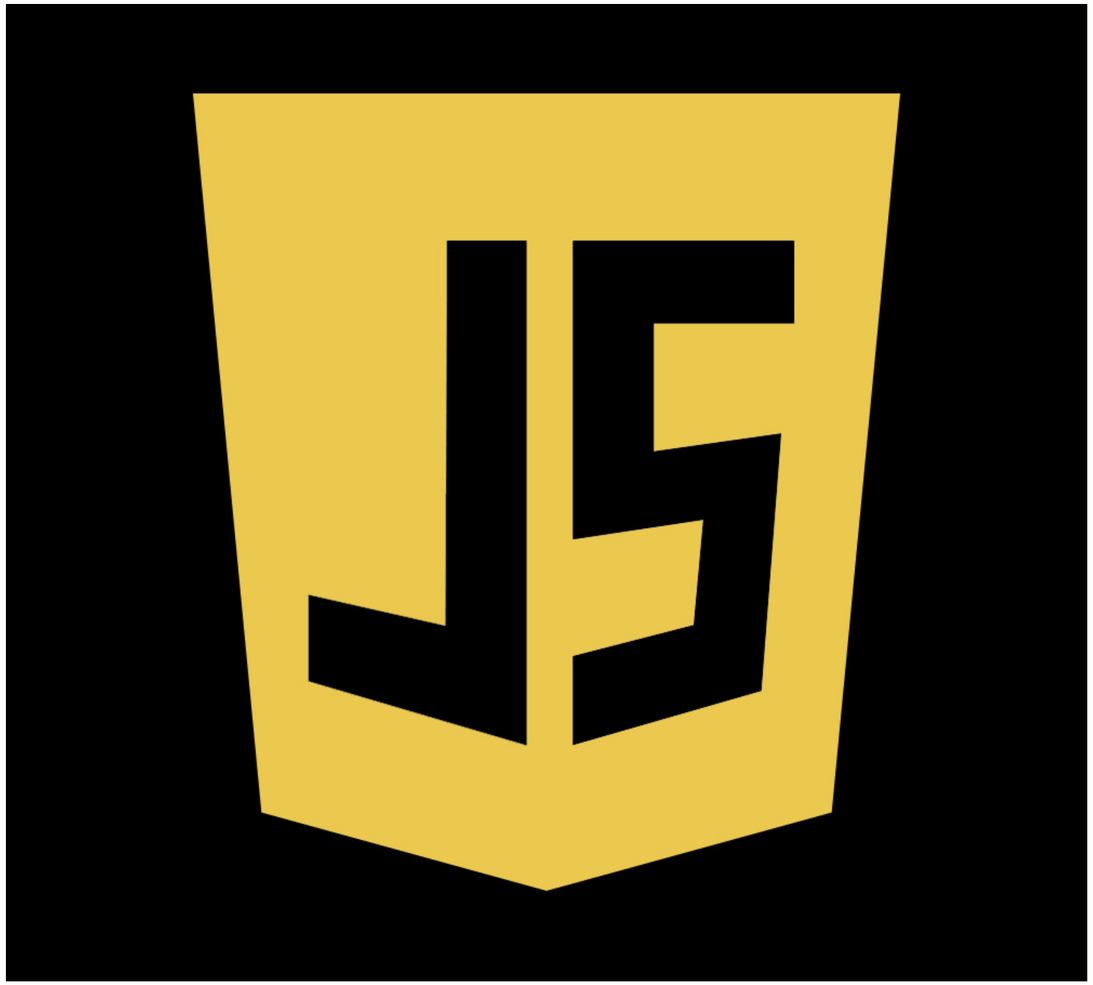
CONSIDERAÇÕES

Client-side language

Show, e????

Significa que é de responsabilidade da máquina do usuário a execução e processamento do nosso código, tudo através de um software que fará a leitura, no caso, o navegador.

Em virtude disso, todas as requisições web, é feito o download dos arquivos js para a máquina do usuário, pois para que o navagedor faça a leitura desses arquivos eles precisam estar localmente disponíveis.



CONSIDERAÇÕES

É uma linguagem interpretada.

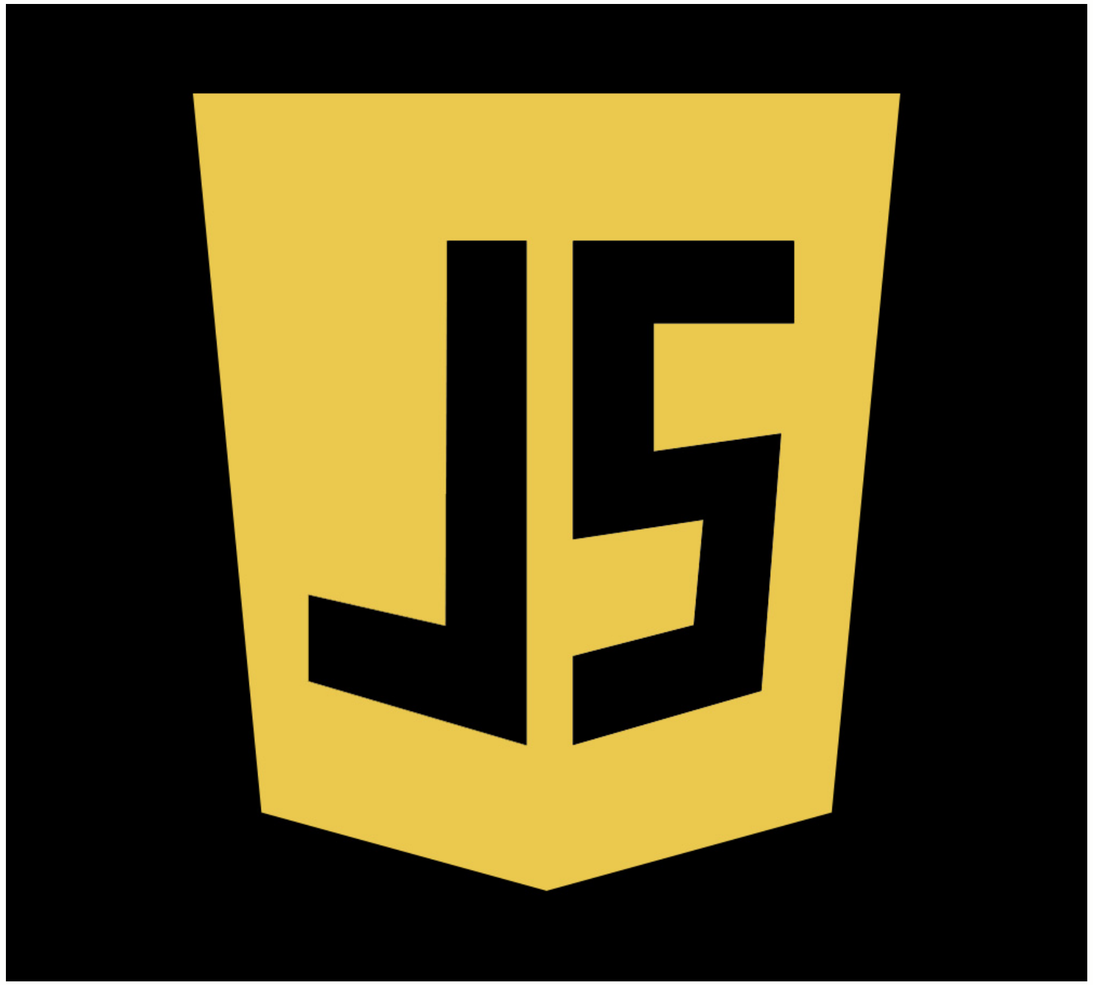
Tá, mas e daí????

Em resumo, significa que o interpretador (navegador) tem que ler linha a linha do nosso código, não há uma tradução ou uma geração de um arquivo diferente.

É considerada First-class function

Ok mano, nome legal, mas e...???

Significa que função pode ir e voltar, ou seja, pode ser parâmetro de entrada de uma função ou mesmo retorno de uma função.



CONSIDERAÇÕES

É Prototype-based programming.

Ou seja????

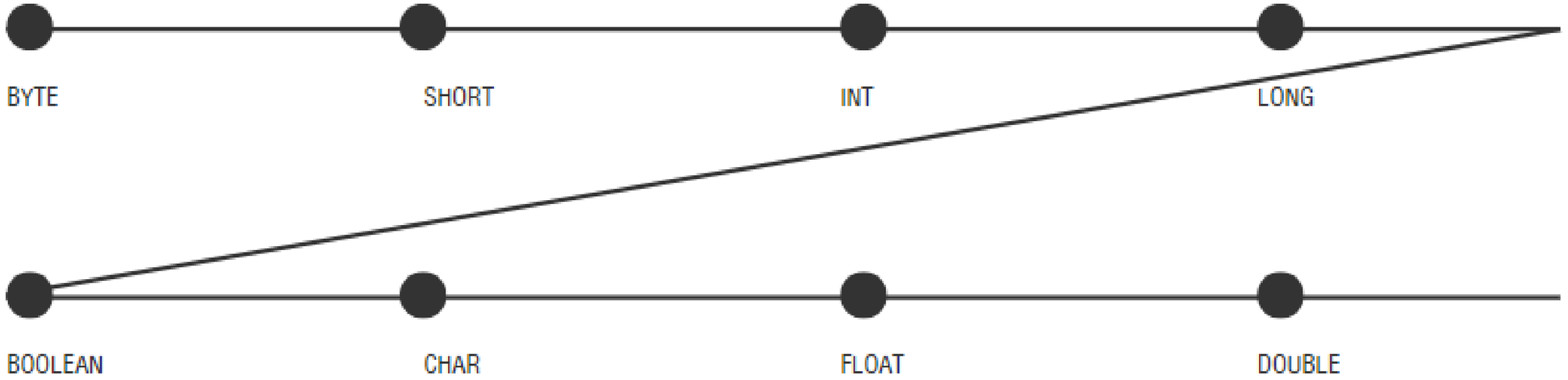
Em resumo, é como se fosse uma linguagem quase orientada a objetos. A herança é feita através de objetos, que servem como protótipos.

É multi-paradigma e dinâmica

Ah conheço os paradigmas, então beleza.

Significa que podemos trabalhar como se fosse orientada a objetos, funcional, baseada em eventos... E gohorse!

TIPAGEM__



VAR

Pouco utilizado, não deprecado

LET

Para variáveis locais, quando temos o intuito de alterar ao decorrer do código

CONST

Para variáveis "globais", mas ainda no contexto da função, que não são alteradas durante a execução. Constante mesmo.

Na declaração da variável, só precisamos utilizar a palavra chave, que na verdade é tanto a visibilidade quanto o nível dela.

TÁ CALMO... QUE??

Isso mesmo!

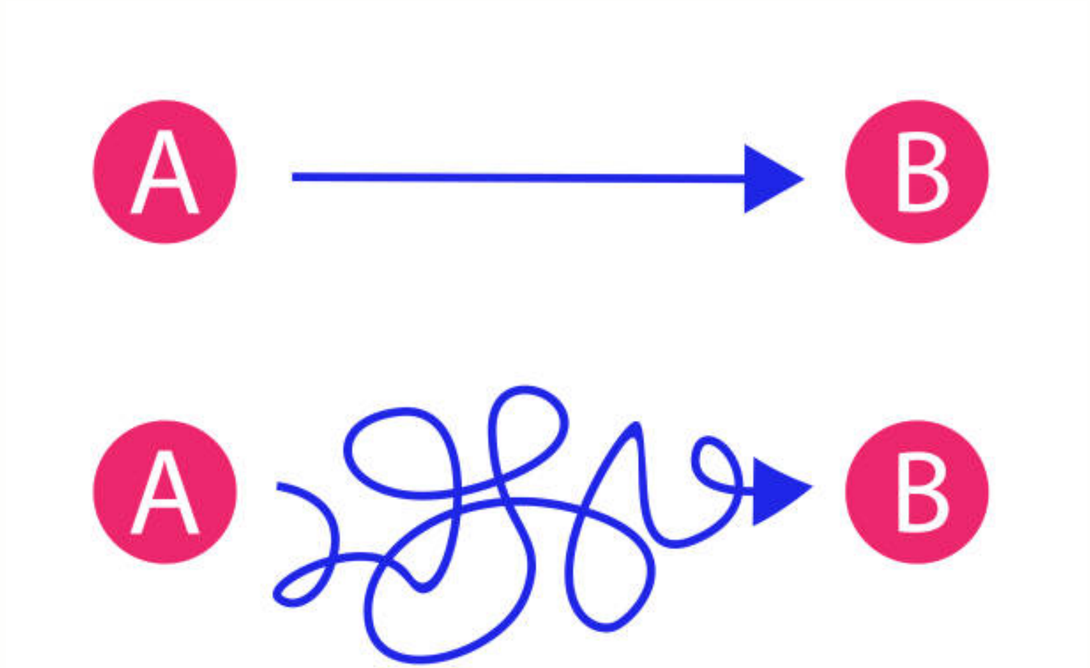
Não há declaram de tipagem das variáveis em javascript.

Maaaas... por baixo dos panos ele trabalha com 3 tipos, basicamente:

- number -> contempla todos os tipos de valores numéricos;
- boolean -> o classico true/false;
- text -> contempla a string e o char e também o texto gigante;

E temos também para os tipos compostos:

- arrays -> os dados do array também não são tipados, sendo assim, não precisamos respeitar somente um tipo de dado. Respeitar somente o fato de termos informações que fazem sentido estarem juntas;
- objeto -> Pode ser criado ou alterado a qualquer momento em qualquer lugar. Seus atributos também não são tipados, e seus valores podem ser diferentes conforme a necessidade;



CONVERSÕES

Não precisa!

Se não tem tipo não precisa converter.

Também não é assim...

Temos que ter em mente que por trás dos panos ao interpretar uma variável de forma errada o código estará errado, tão logo, precisamos converter a informação pro "tipo" desejado.

Agora vamos ao lado obscuro do javascript!

Estejam preparados!

OPERADORES DE ATRIBUIÇÃO

```
» let lado = 2;
  const pi = 3.1426;
  var texto = "Javascript é vida!";
  const valorBool = true;
```

Utilizado para definir/atribuir um valor inicial ou sobrescrever o valor de uma variável.

Lembrando que var não é muito bem vindo no código, pois não trata os detalhes que os demais trabalham. Não chega a ser um erro, porém não recomendado.

OPERADORES ARITMÉTICOS

```
» let area = 2 * 2;
  let area = 2;
  area *= 2;
```

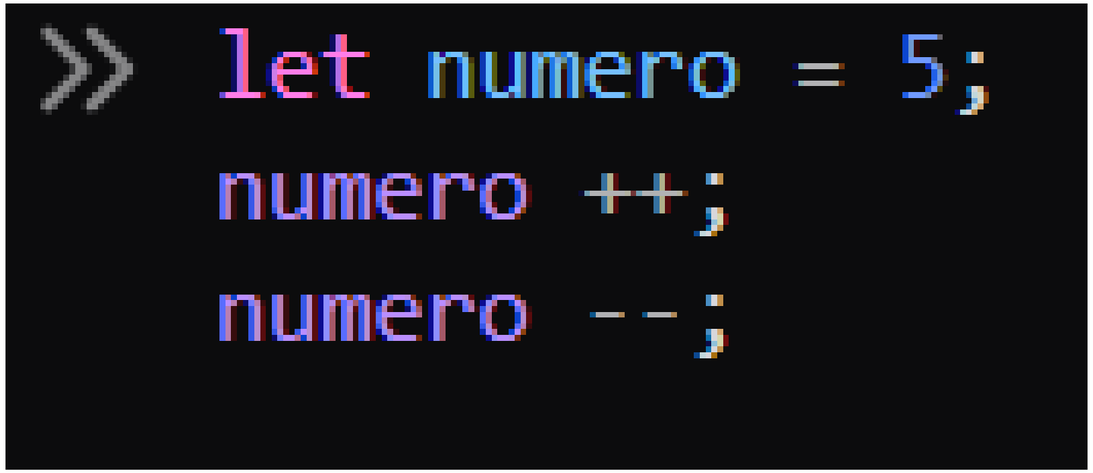
Realizam as operações fundamentais da matemática. Caso seja necessário escrever operações maiores ou mais complexas, podemos combinar esses operadores e criar expressões, o que nos permite executar todo tipo de cálculo de forma programática.

+	operador de adição
-	operador subtração
*	operador de multiplicação
/	operador de divisão
%	operador de módulo (ou resto da divisão)

Tomando sempre cuidado com o tipo da variável atribuído em tempo de execução.

Use **Math.abs** para precisão em calculos.

INCREMENTO E DECREMENTO



Basicamente temos dois deles: ++ e --, os quais podem ser declarados antes ou depois da variável e incrementam ou decrementam em 1 o valor da variável.

OPERADORES DE IGUALDADE

```
const valorA = 1;
const valorB = 2;

if (valorA == valorB) {
  console.log("valores iguais");
} else {
  console.log("valores diferentes");
}
```

Os operadores de igualdade verificam se o valor ou o resultado da expressão lógica à esquerda é igual (“==”) ou diferente (“!=”) ao da direita, retornando um valor booleano.

Isso é o comum.

Agora no Javascript temos uma diferença. Como as variáveis não são tipadas, existem dois novos tipos de operadores de igualdade, que validam também o "tipo" dos dados.

=== (3 iguais)
e
!== (exclamação e 2 iguais)

OPERADORES RELACIONAIS

```
const valorA = 1;
const valorB = 2;

if (valorA > valorB) {
  console.log("A maior que B");
}

if (valorA >= valorB) {
  console.log("A maior ou igual a B");
}

if (valorA < valorB) {
  console.log("A menor que B");
}

if (valorA <= valorB) {
  console.log("A menor ou igual a B");
}
```

Um em relação do outro. Como as variáveis se relacionam quando comparamos uma com a outra, dependendo, claro, do operador de igualdade.

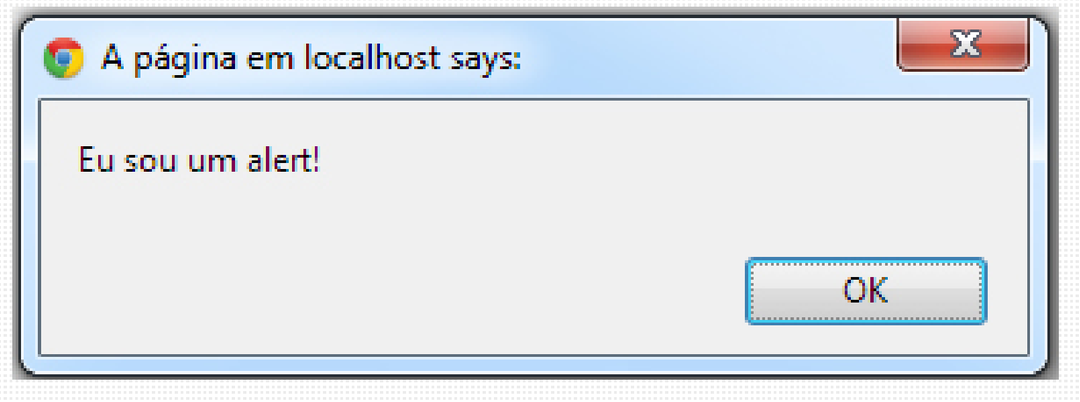
- > Maior que
- >= Maior igual
- < Menor que
- <= Menor igual

OPERADORES LÓGICOS

```
if((1 == (2 -1)) && (2 == (1 + 1))){
     System.out.println(“Ambas as expressões são verdadeiras”);
}
```

&&	Utilizado quando desejamos que as duas expressões sejam verdadeiras.
||	Utilizado quando precisamos que pelo meno um das expressões seja verdadeira.

Os operadores lógicos representam o recurso que nos permite criar expressões lógicas maiores a partir da junção de duas ou mais expressões. Para isso, aplicamos as operações lógicas E (representado por “&&”) e OU (representado por “||”).



INFO BONUS

Pra facilitar a vida!

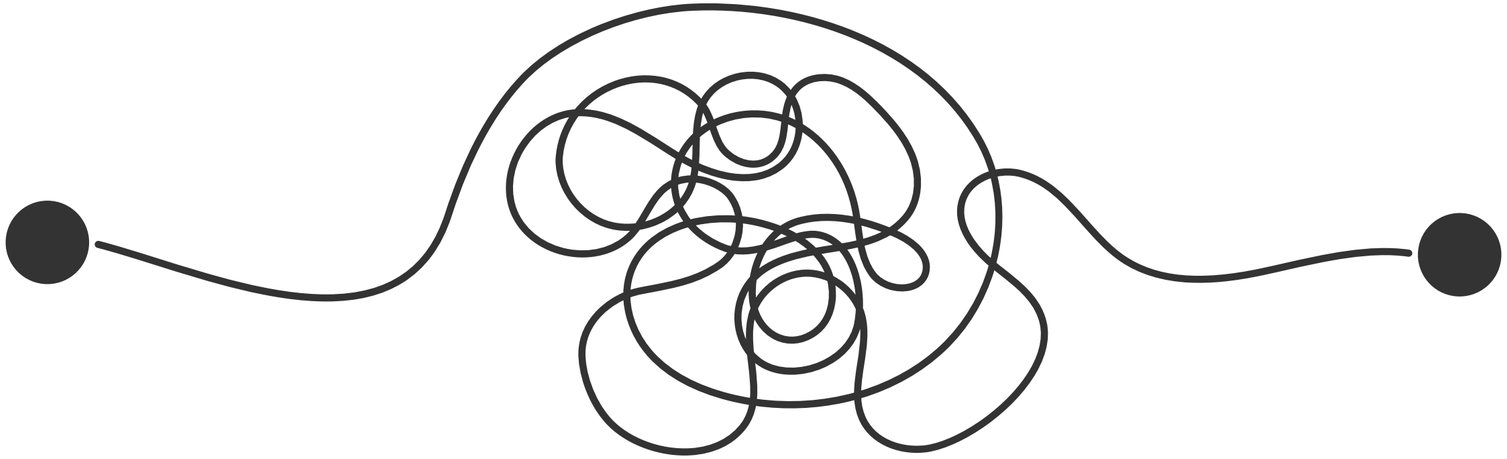
No javascript existem varias coisas prontas e nativas. Uma delas são os alertas de tela, dos quais destaco-lhes 3:

- alert
- prompt
- confirmation

Cada um com sua atribuição.

Para as atividades vamos utilizar esses caras input das informações e para informar o resultado.

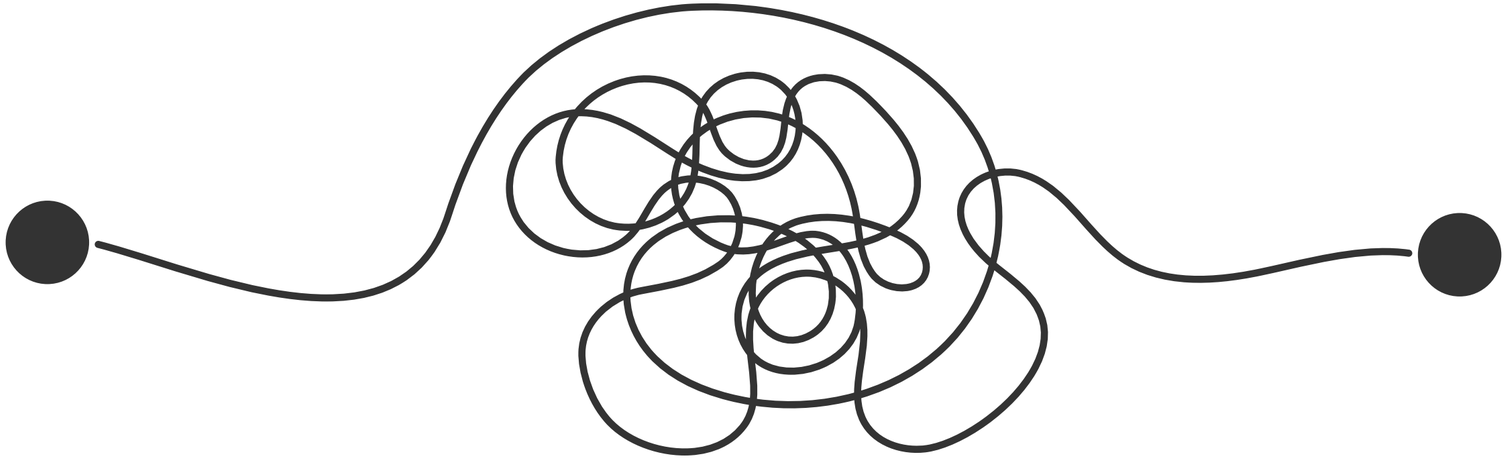
Isso enquanto não vermos DOM.



_ DESAFIO

Vamos ao código!! Como resultado, é esperado exibir o valor juntamente com uma legenda, o que é esse valor.

1. Crie um script que calcula a média de três notas, atribui o resultado a uma variável e exiba esse valor. As notas podem ser quaisquer valores numéricos.
2. Crie um script que converte uma temperatura em graus Celsius para Fahrenheit. A fórmula para conversão é: Fahrenheit = (Celsius * 9/5) + 32.
3. Crie um script que calcula a área de um retângulo com base na largura e altura fornecidas. A fórmula para cálculo da área é: Área = Largura * Altura.
4. Crie um script que calcula o Índice de Massa Corporal (IMC) com base no peso e altura fornecidos. A fórmula para cálculo do IMC é: IMC = Peso / (Altura * Altura).
5. Crie um script que converte um determinado número de horas em minutos. Atribua o resultado a uma variável chamada "minutos". Considere que 1 hora possui 60 minutos.



DESAFIO

Vamos ao código!! Como resultado, é esperado exibir o valor juntamente com uma legenda, o que é esse valor.

1. Crie um script que receba o nome e sobrenome de uma pessoa e concatene-os em uma única string.
2. Crie um script que conte quantos caracteres uma determinada string possui e exiba essa informação.
3. Crie um script que converta uma string fornecida para letras maiúsculas e exiba a frase em caixa alta.
4. Crie um script que extraia uma parte específica de uma string. Por exemplo, extraia os primeiros três caracteres de uma palavra e exiba o resultado.
5. Crie um script que substitua as letras, conforme:
    a. A -> 4
    b. I (letra i) -> 1
    c. E -> 3
    d. O (letra ó) -> 0 (zero)
    e. S -> 5

```js
function testNum(a) {
  let result;
  if (a > 0) {
    result = 'positivo';
  } else {
    result = 'não positivo';
  }
  return result;
}

console.log(testNum(-5));
// "não positivo"

```

IF/ELSE

Essa estrutura condicional é normal, não tem nada de diferença para o que já estamos acostumados.

Mas pra não esquecer vamos falar sobre.

SWITCH/CASE

Mesma coisa aqui.

Só vamos rever pra ter certeza que tá fresco.

```js
const expr = 'Papaya';
switch (expr) {
  case 'Laranja':
    console.log('Laranja está R$ 1,99 o kilo.');
    break;
  case 'Manga':
  case 'Papaya':
    console.log('Manga e Melão Papaya estão R$ 7,99 o kilo.');
    break;
  default:
    console.log(`Mals, não temos ${expr}.`);
}

```

LAÇOS DE REPETIÇÃO

```
for (key in obj) {
  console.log(key, obj[key]);
}
```

```
const nums = [1, 2, 3];
for (num of nums) {
  console.log(num);
}
```

For

Nada de novo, exatamente a mesma sintaxe.

While

Nada de novo, exatamente a mesma sintaxe.

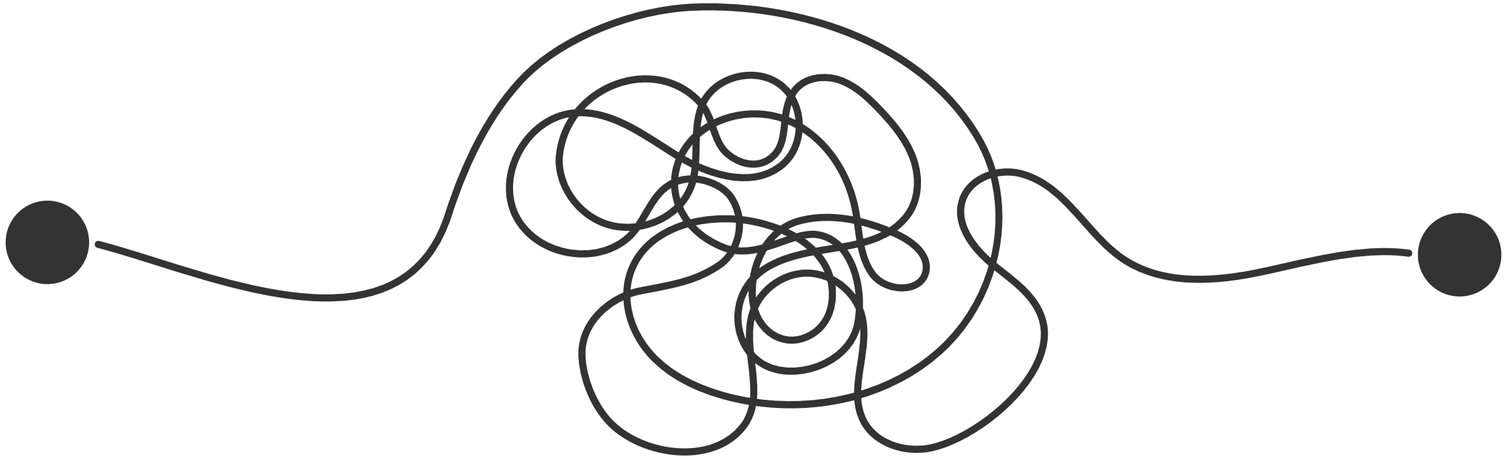
Do-while

Nada de novo, exatamente a mesma sintaxe.

Foreach

Aqui temos uma pequena alteração e um detalhe importante, a sintaxe é a mesma do java, só estar atento à um detalhe:

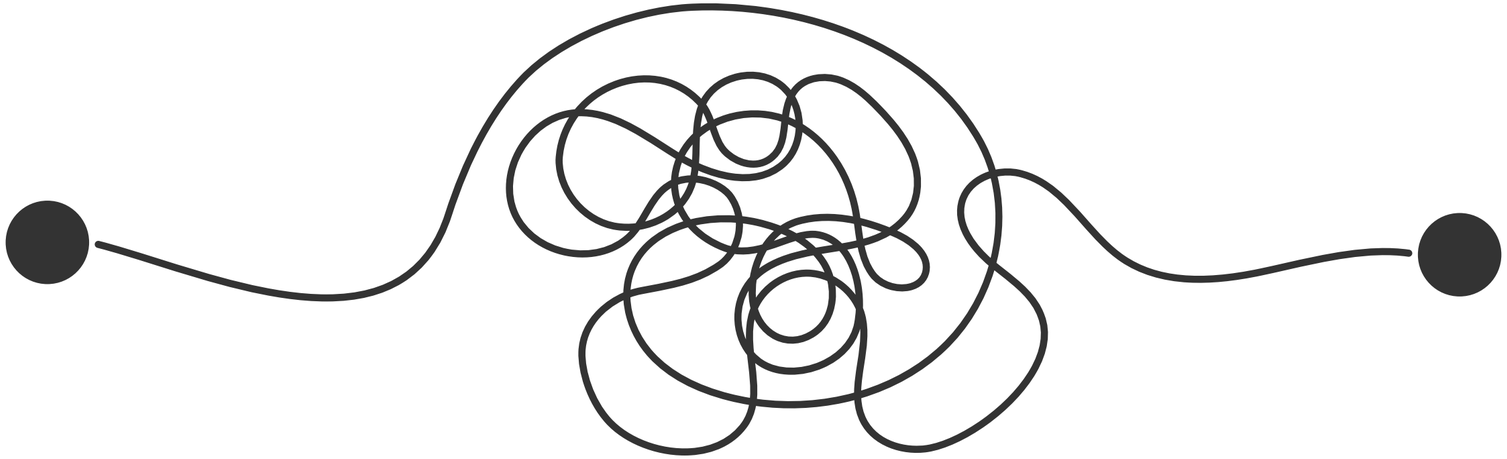
- in para objetos
- of pra arrays



__ DESAFIO

Vamos ao código!!

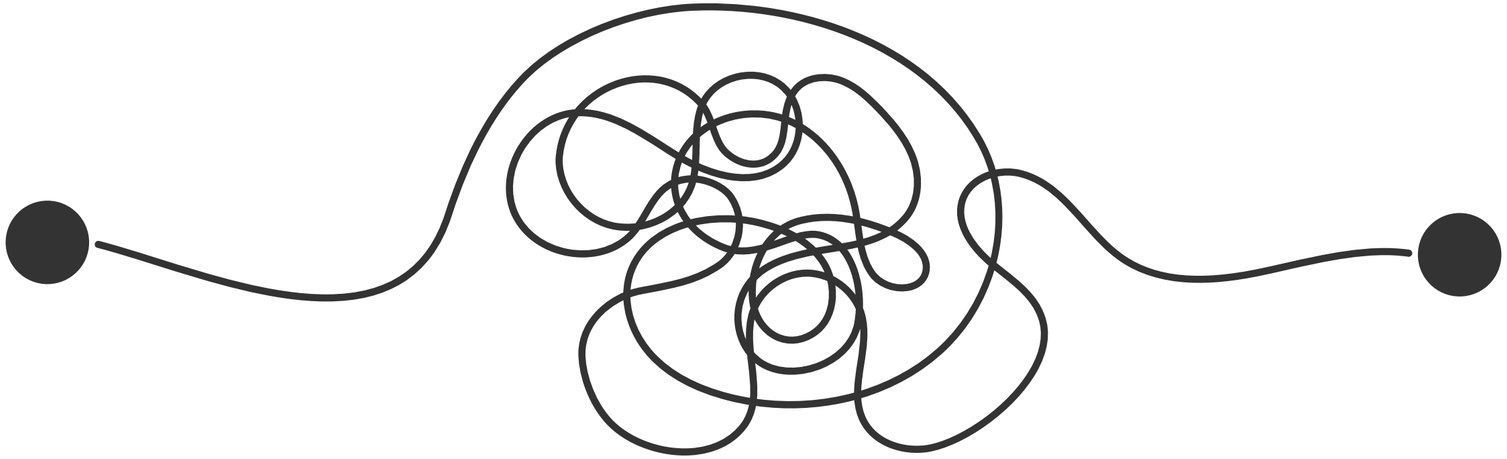
1. Crie um script que com um número verifique se é par ou ímpar e exiba uma mensagem informando se o número é par ou ímpar.
2. Crie um script que com o preço de um produto calcule o valor com desconto de 10% e exiba o valor com desconto.
3. Escreva um script que determine se um número é primo.
4. Crie um script que com a idade de uma pessoa em anos e converta para dias e exiba o resultado.
5. Crie um script que solicite um número de telefone ao usuário e formate-o no formato "(XX) XXXX-XXXX" e exiba o número de telefone formatado.



_ DESAFIO

Vamos ao código!! De novo!!

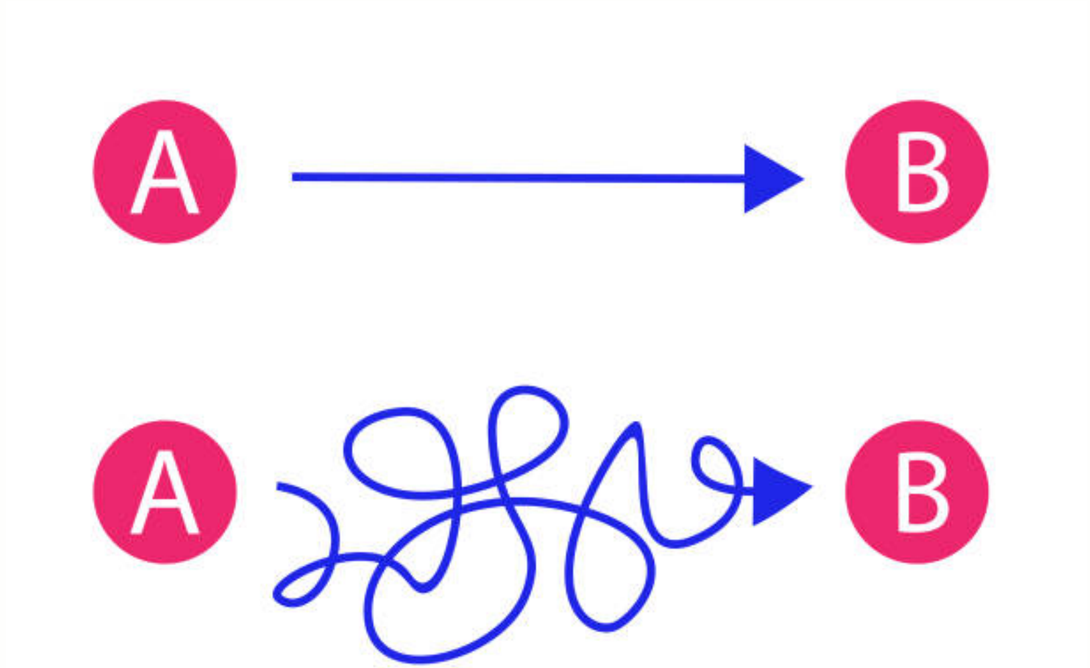
1. Crie um programa que exiba todos os números pares de 1 a 10. Utilize um laço de repetição e condicional para verificar se cada número é par.
2. Crie um programa que solicite ao usuário três notas e calcule a média. Se alguma nota for inválida (for menor que 0 ou maior que 10), solicite novamente essa nota. Utilize um laço de repetição e condicional para validar as notas.
3. Crie um programa que exiba uma contagem regressiva de 10 a 1. Utilize um laço de repetição para iterar de 10 a 1 e exiba cada número.
4. Crie um programa que solicite uma palavra ao usuário e verifique se contém uma letra específica. Utilize um laço de repetição para percorrer cada letra da palavra e um condicional para verificar se a letra desejada está presente.
5. Crie um programa que solicite ao usuário uma frase e converta todas as letras para maiúsculas. Utilize um laço de repetição para percorrer cada letra da frase e atribua o resultado a uma nova variável.



_ DESAFIO

Vamos ao código!! De novo!! Once more!!

1. (while) Escreva um programa que imprima na tela os números ímpares de 1 a 20.
2. (for) Faça um programa que calcule a média de um vetor de números reais com 10 elementos.
3. (do-while) Crie um jogo de adivinhação em que o usuário tem que tentar acertar um número aleatório de 1 a 100, dando dicas de "maior" ou "menor" até acertar.
4. (for-each) Crie um programa que some todos os valores de um vetor de inteiros com 100 elementos.
5. (for) Escreva um programa que gere a sequência de Fibonacci até o décimo termo.
6. (while) Faça um programa que leia uma lista de números inteiros até o usuário digitar zero e exiba o maior número digitado.
7. (do-while) Escreva um programa que solicite ao usuário para digitar uma senha, repetindo a solicitação até que ele acerte a senha correta.
8. (for-each) Crie um programa que conte a quantidade de letras "a" em uma string informada pelo usuário.
9. (for) Faça um programa que imprima na tela os números primos de 1 a 100.
10. (while) Escreva um programa que calcule a soma dos números pares de 1 a 50.



FUNCTIONS

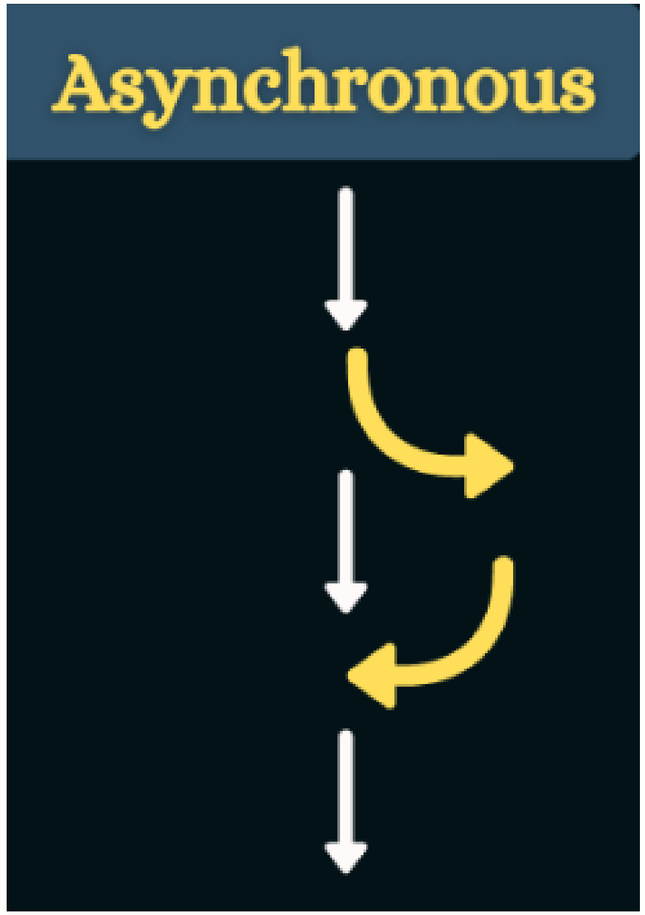
ué, vamos ver funções???

Calma jovem!

Vimos que javascript tem varias particularidades, sendo uma delas o first-class function.

Então temos que ver as diversas maneiras de trabalhar com funções e ainda sim ter um código bonito e funcional.

Mas aqui é dificil de falar sobre então vamos ao código direto



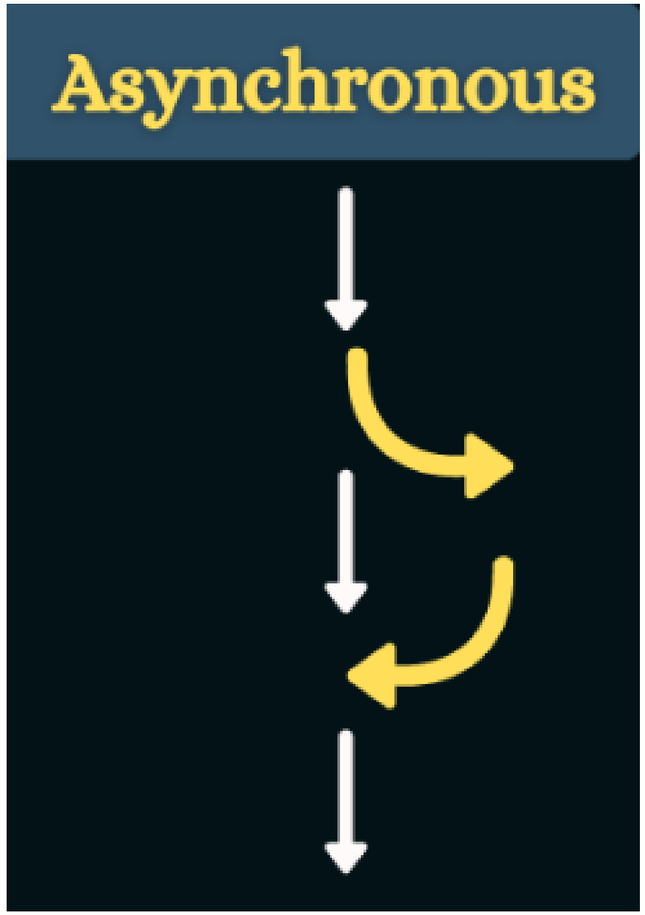
ASYNC PROGRAMING

Para trabalhar com funções assíncronas, basta utilizarmos o comando **async**.

Claro, tem que cuidar sempre com a questão da ordenação dos comandos no código, se o retorno da função assíncrona for necessário em algum momento, certifique-se de que tá tudo certo.

Toda a async function retorna uma Promise, que já vamos ver, então esse retorno precisa ser tratado.

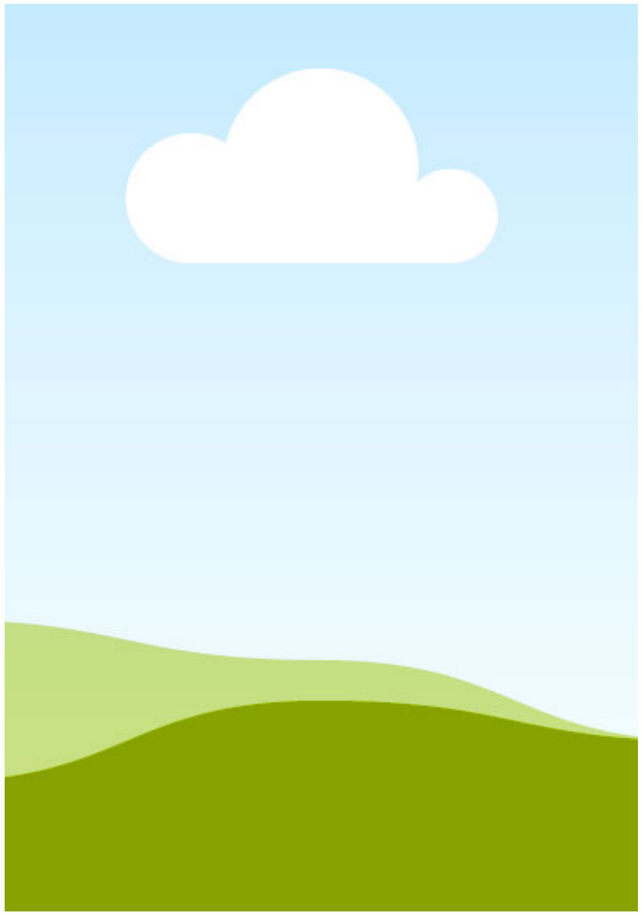
Para trabalhar bem com async, temos o comando await, que é exatamente um ponto de espera do retorno de alguma coisa.



CALLBACK E PROMISE

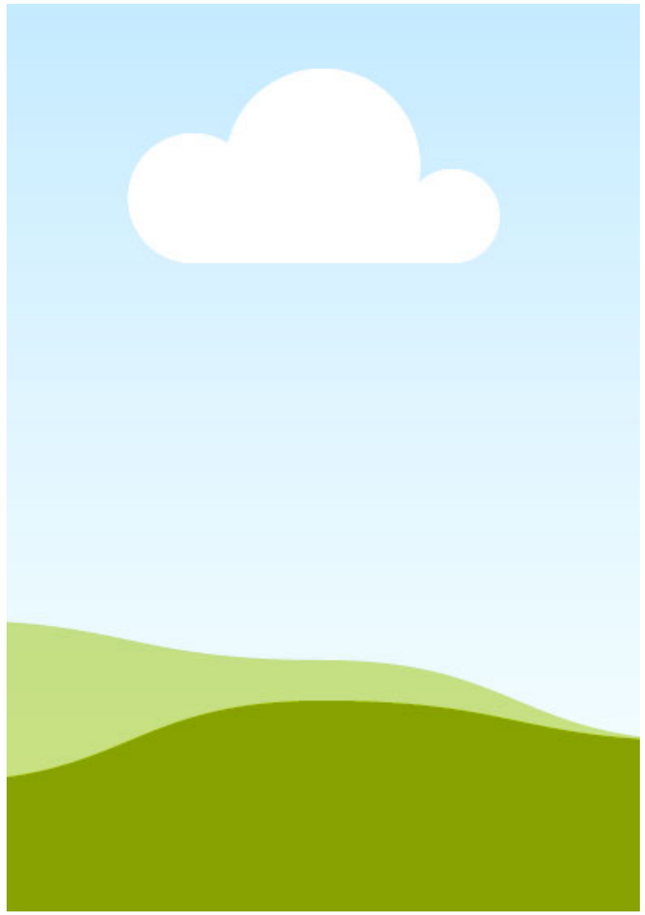
Callbackfunction também são funções assíncronas, que tem um retorno que precisa ser tratado com novas funções.
É possível fazer um tratamento simples, das opções de retorno de sucesso ou erro.
Tem um detalhe aqui que é a falta de retorno, não é possível trabalhar com o tempo da resposta, e nesse caso não teremos retorno nenhum.

Promise são promessas de entrega, e temos muitas diferenças, mas de semelhança por enquanto temos a questão da resposta async e a necessidade de tratar as resposta, porque a resposta é um promise.
De diferente, conseguimos empilhar promises ou ainda serializar, que sempre em um retorno será feita uma nova promise.



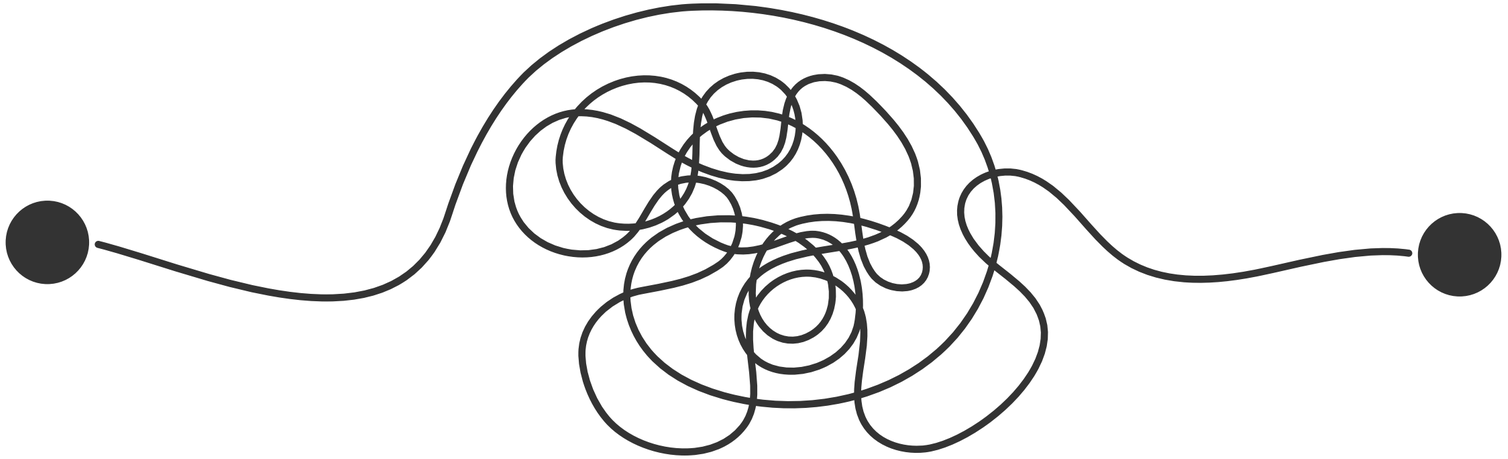
REQUISIÇÕES

ipsum



DOM

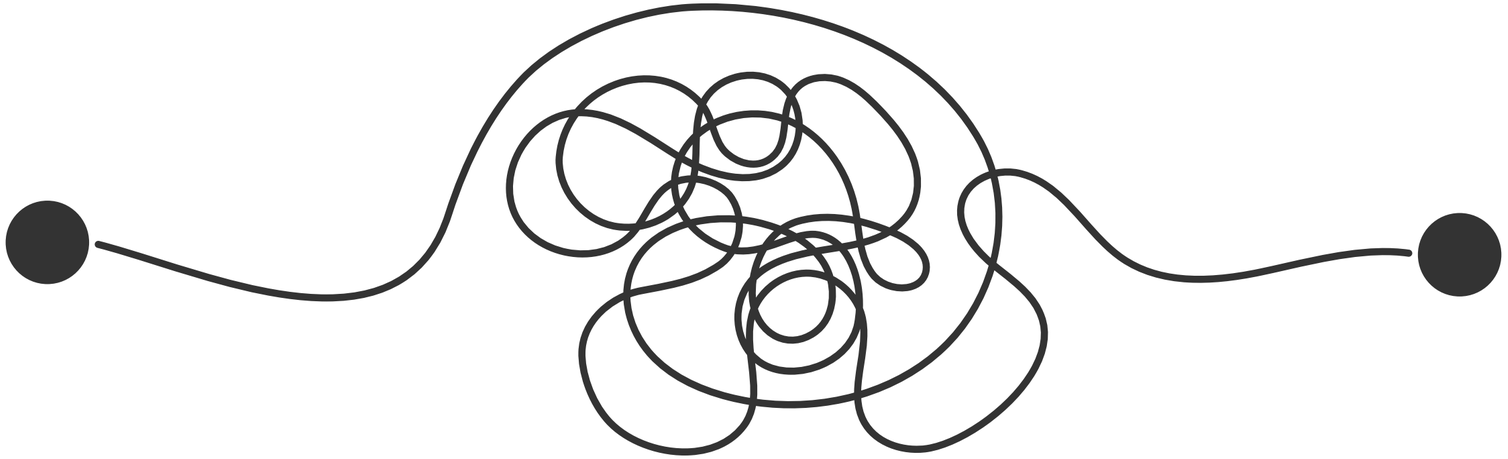
ipsum



_ DESAFIO SUPREMO HARDCORE

Agora para avacalhar mesmo, vamos tornar um exercício anterior um pouco mais difícil!
Vai ser o 2.

1. Nele precisávamos só converter Celsius para Fahrenheit.
2. Agora vamos solicitar ao usuário que coloque a informação em um campo (qual forma de input, fica livre escolha), ai vamos receber esse campo e fazer o calculo normal.
3. Quanto a exibição também deve ser em tela, vamos manipular o HTML para exibir esse resultado de forma destacada.
4. Não sendo o bastante, vamos ainda vamos pegar a geolocalização do usuário, através do javascript, isso é uma função que PRECISA ser async.
5. Com a geolocalização vamos ir em uma API de clima aberta, buscar a temperatura atual segundo a localização.
6. Com a temperatura segundo a coordenada vamos converter também, e comparar se está mais quente ou mais frio que a temperatura informada.
7. Exibir tudo isso em tela (como preferir) bem bonitinho e organizado.



_ DESAFIO SUPREMO HARDCORE

Agora para avacalhar mesmo, vamos tornar um exercício anterior um pouco mais difícil!
Vai ser o 2.

1. Nele precisávamos só converter Celsius para Fahrenheit.
2. Agora vamos solicitar ao usuário que coloque a informação em um campo (qual forma de input, fica livre escolha), ai vamos receber esse campo e fazer o calculo normal.
3. Quanto a exibição também deve ser em tela, vamos manipular o HTML para exibir esse resultado de forma destacada.
4. Não sendo o bastante, vamos ainda vamos pegar a geolocalização do usuário, através do javascript, isso é uma função que PRECISA ser async.
5. Com a geolocalização vamos ir em uma API de clima aberta, buscar a temperatura atual segundo a localização.
6. Com a temperatura segundo a coordenada vamos converter também, e comparar se está mais quente ou mais frio que a temperatura informada.
7. Exibir tudo isso em tela (como preferir) bem bonitinho e organizado.