Curso Presencial Programação Fullstack

Aula 02

Prof. MSc. Kelson | Senior Software Engineer

PROFKELSON.DEV

BORA CODAR?

Cronograma

01

JS

03

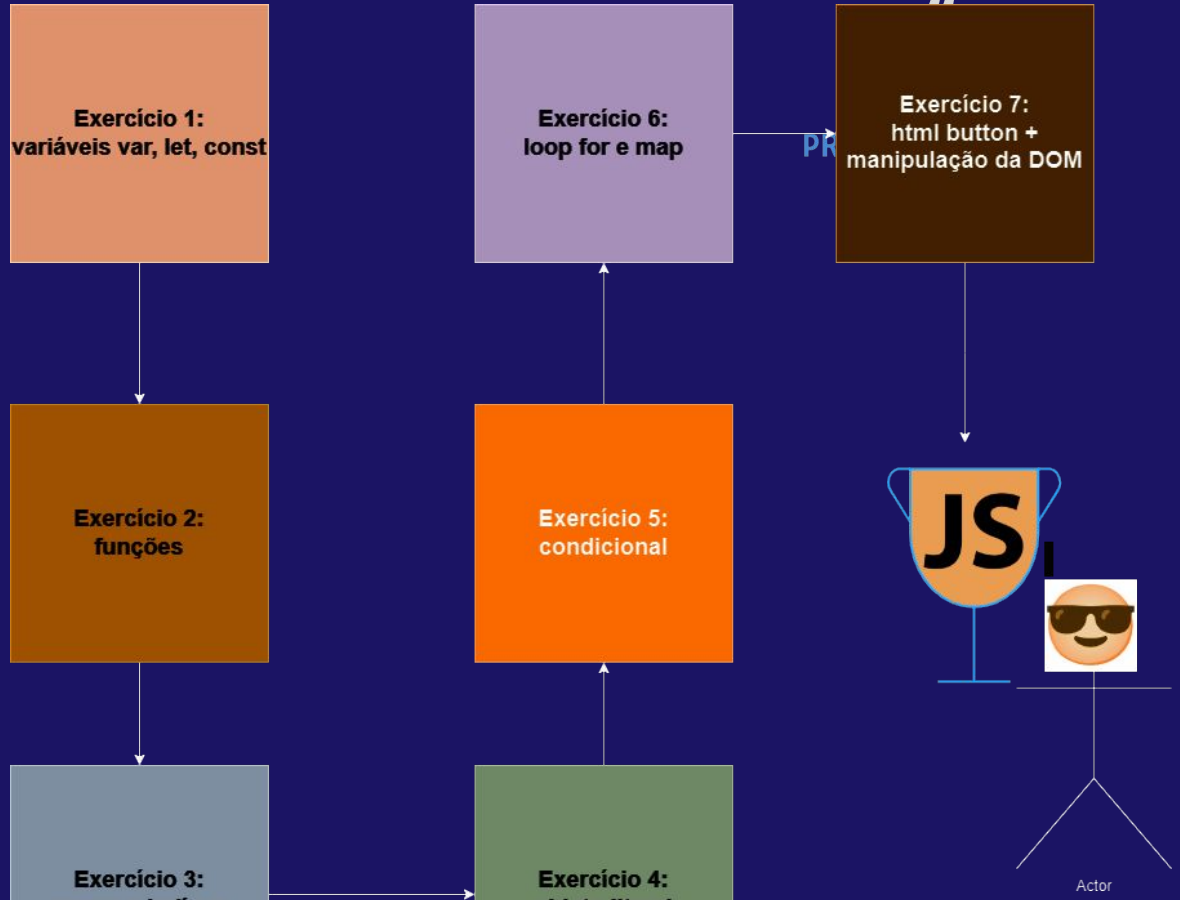
Bootstrap

02

Treinando JS

JavaScript

- Inglês: Practice
- Espanhol: Practicar
- Francês: Pratiquer
- Alemão: Üben
- Italiano: Praticare
- Português: Praticar
- Russo: Практиковать (Praktikovat')
- Chinês (Mandarim): 练习 (Liànxí)
- Japonês: 練習する (Renshū suru)
- Árabe: ممارسة (Mumarasa)
- Hindi: अभ्यास (Abhyās)
- Coreano: 연습하다 (Yeonseuphada)



PROFKELSON.DEV
BORA CODAR?

JavaScript

- Antes de tudo, JavaScript não é Java. Apesar do nome, elas não possuem ligação direta.

- JS é uma linguagem de programação. Foi criada, originalmente, com o objetivo de facilitar processos dentro páginas web, tornando a programação de animações e alertas mais simples.

PROFKELSON.DEV
BORA CODAR?

JavaScript

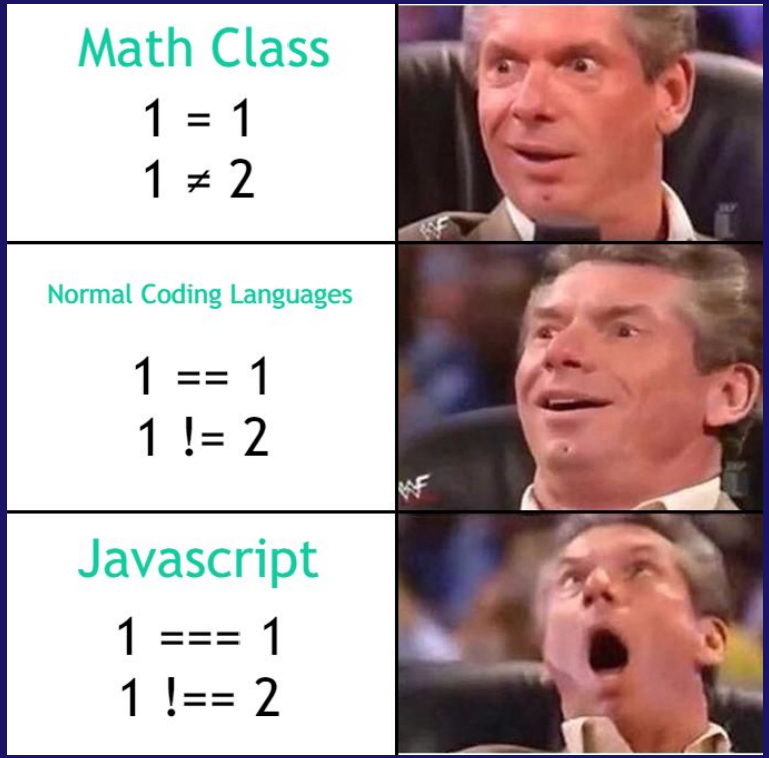
- Com isso, os browsers foram aceitando o JS e compatibilizando o seu uso dentro das páginas web.

- Pouco tempo depois foi se popularizando tanto que se tornou uma das principais linguagens de programação do mercado.

PROFKELSON.DEV
BORA CODAR?

JavaScript

- Vantagens:
- Não precisa de um compilador, o browser vai interpretá-lo com HTML;
- Fácil aprendizagem, comparada com outras linguagens de programação;
- Compatibilidade com várias plataformas e navegadores;
- Faz com que os sites/sistemas sejam bem mais interativos e menos estáticos, gerando uma melhor experiência ao usuário.



PROFKELSON.DEV
BORA CODAR?

Frameworks

- Facilitar o processo de desenvolvimento de software.
- Oferecem a “estrutura básica” para o sistema ser programado.
- São criados por equipes de dev experientes, geralmente com ampla comunidade ativa.

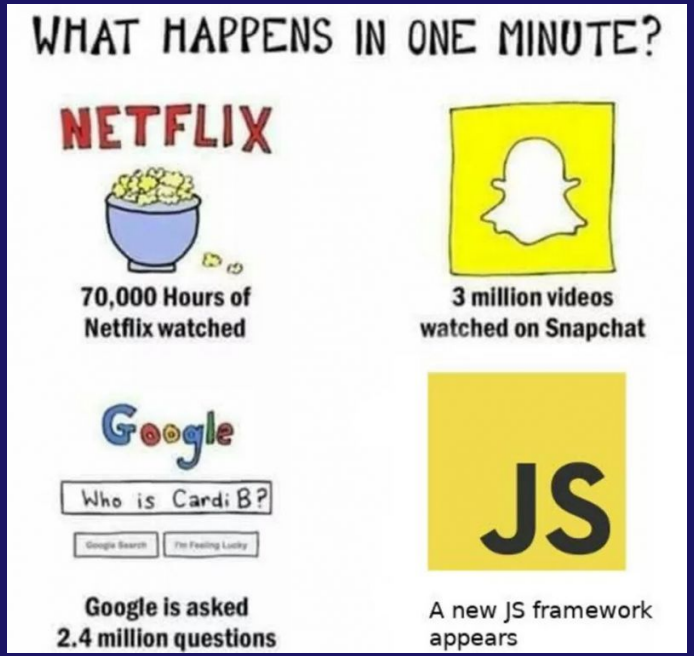
PROFKELSON.DEV
BORA CODAR?

E no mundo front tem framework?

Sim, e muitos!

- Entre os mais populares podemos destacar:

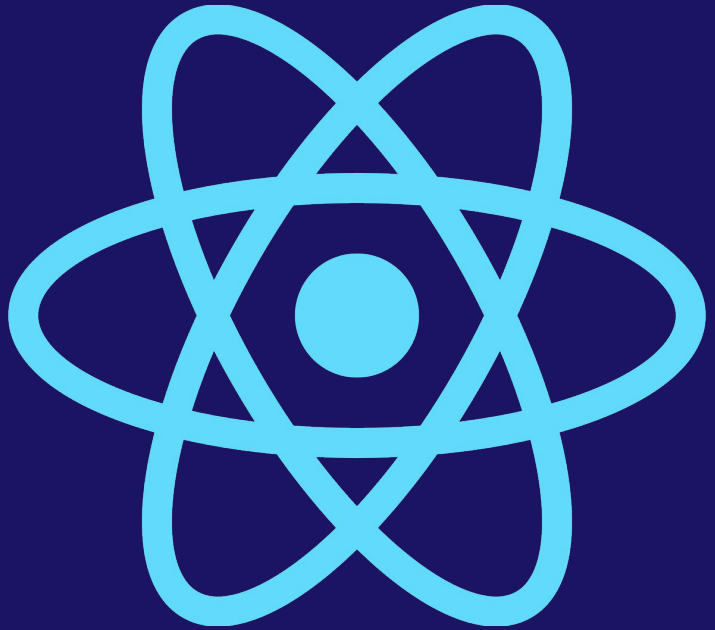
- ***ReactJS***
- Angular
- Vue.js
- Ember.js
- Entre outros!



PROFKELSON.DEV
BORA CODAR?

Nosso foco: ReactJS

- Um dos mais populares frameworks JavaScript.
- Desenvolvido inicialmente pelo Facebook em 2013.
- Utilizado por grandes empresas como Netflix, Airbnb, Whatsapp e Instagram.
- Maas antes, precisamos treinar JS!
  - Já treinamos HTML e CSS! :D



JavaScript

- JavaScript foi criado em apenas 10 dias por Brendan Eich em 1995, enquanto ele trabalhava na Netscape Communications Corporation.
- O nome "JavaScript" foi escolhido para capitalizar o sucesso da linguagem de programação Java, que estava em alta na época.
- O JavaScript é uma linguagem de programação interpretada, o que significa que o código fonte é executado diretamente pelo navegador ou aplicativo, sem a necessidade de compilar antes.
- O JavaScript é usado para criar interações dinâmicas e animações em páginas web, além de ser uma das principais linguagens de programação para desenvolvimento de aplicativos web.

JavaScript

- O JavaScript suporta programação orientada a objetos, funcional e procedural, oferecendo aos programadores uma grande flexibilidade no desenvolvimento de seus projetos.
- O JavaScript tem uma ampla variedade de bibliotecas e frameworks disponíveis, como o jQuery, React, Angular, Vue e muitos outros, que ajudam a simplificar o desenvolvimento web e acelerar o tempo de produção.
- O JavaScript é uma das linguagens de programação mais utilizadas em todo o mundo, sendo usada por mais de 95% dos sites ativos na internet.
- O JavaScript é suportado por todos os principais navegadores, incluindo Chrome, Firefox, Safari, Edge e Opera.
- A sintaxe do JavaScript foi influenciada por várias outras linguagens de programação, incluindo Java, C e Perl.
- JavaScript não tem relação com a linguagem de programação Java, apesar do nome similar. As duas linguagens são distintas e têm propósitos diferentes.

PROFKELSON.DEV
BORA CODAR?

Antes de tudo...

- Vamos brincar com o console.log() ?
- Você possui o Node JS instalado?

Variáveis

- Variável é nome simbólico para um valor
- Utilizadas para armazenar dados que podem ser usados mais tarde
- Para declarar uma variável em JS podemos utilizar:
  - var: Forma mais antiga de se declarar variáveis em JS. Pode ser acessada fora do escopo (caso seja declarada globalmente)
  - let: Forma mais moderna de se declarar vars, introduzido no ES6, possuem escopo de bloco, ou seja, só são acessíveis no bloco em que foram declaradas (exemplo: dentro de if/else/funções)
  - const: Semelhante ao let em termos de escopo, porém, uma vez declarado valor para


```javascript
var minhaVariavel;
let outraVariavel;
const terceiraVariavel;
```

PROFKELSON.DEV
BORA CODAR?

Tipos de Dados

- Os tipos de dados em JS são divididos em dois tipos principais:
  - Primitivos
  - Objetos
- Fracamente tipada.
- Tipos de Dados Primitivos: Valores simples que não tem propriedades métodos.
  - String: Sequência de caracteres entre aspas simples ou duplas
  - Number: Um número. Inteiros e números de ponto flutuante
  - Boolean: Representa um valor lógico
  - Null: Valor nulo
  - Undefined: Variável que ainda não foi atribuída a um valor

```javascript
var minhaString = "Olá, mundo!";
let meuNumero = 42;
const meuBoolean = true;
let meuNulo = null;
var minhaIndefinida;
```

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (1) [COM O PROF]

- Faça um script que tenha três variáveis:
  - var nome
  - let sobreNome
  - const cpf
- nome e cpf devem estar em um escopo global
- sobreNome deve estar dentro de uma função
- A execução do programa deve imprimir o nome completo na ordem correta (com quebra de linha):
  - Nome
  - Sobrenome
  - CPF
  - Outra linha com uma mensagem

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (1_1) [SOZINHO(A)]

- Crie um script que simule um sistema de registro de produtos. O script deve ter três variáveis:
  - var produto
  - let categoria
  - const codigoProduto
- As variáveis produto e codigoProduto devem estar em um escopo global.
- A variável categoria deve estar dentro de uma função.
- A execução do programa deve imprimir os detalhes do produto na ordem correta (com quebra de linha):
  - Produto
  - Categoria
  - Código do Produto

PROFKELSON.DEV
BORA CODAR?

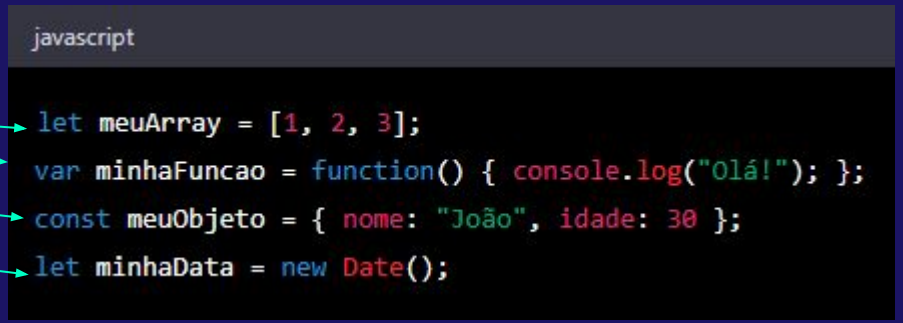
VAMOS CODAR? (1_2) [SOZINHO(A)]

- Crie um script que simule o registro de informações de um aluno. O script deve ter três variáveis:
  - var nomeAluno
  - let curso
  - const matricula
- As variáveis nomeAluno e matricula devem estar em um escopo global.
- A variável curso deve estar dentro de uma função.
- A função deve retornar uma string que concatene todas as informações do aluno em uma única linha utilizando template literals ${var}.

PROFKELSON.DEV
BORA CODAR?

Tipos de Dados

- **Tipos de Dados Objetos: Valores complexos que possuem propriedades e métodos.**
  - **Arrays**
  - **Funções**
  - **Objetos Regulares**
  - **Objetos de Data**



PROFKELSON.DEV
BORA CODAR?

Funções

- As funções são blocos de código que podem ser chamados para executar uma tarefa específica.
- Para definir uma função em JS utilizamos a palavra-chave “function”
- Seguida pelo nome da função e parâmetros entre parênteses
- O corpo da função é colocado entre chaves {} e contém as instruções a serem executadas quando a função é chamada.
- No exemplo ao lado temos a utilização de um “return” na função. O return será responsável por retornar um resultado final ao processamento da função.

javascript — Copy code

```javascript
function minhaFuncao(parametro1, parametro2) {
  // corpo da função
  console.log("O parâmetro 1 é " + parametro1 + " e o parâmetro 2 é " + parametro2);
}

minhaFuncao("Hello", "World");
// Output: O parâmetro 1 é Hello e o parâmetro 2 é World
```

javascript

```javascript
function somar(num1, num2) {
  return num1 + num2;
}

let resultado = somar(2, 3);
console.log(resultado);
// Output: 5
```

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (2) [COM O PROF]

- Faça um script que contenha uma função: objetivoDoCurso(tecnologia)
- A função deve retornar a string: “Meu objetivo é aprender [parametro tecnologia]”
- Fora da função defina uma const tecnologia que possua o valor “React”.
- Chame a execução da função para que a mesma retorne: “Meu objetivo é aprender React”
  - Faça um exemplo com function
  - Outro com arrow function

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (2_1) [SOZINHO]

- Crie um script que contenha duas funções, uma usando a sintaxe tradicional e outra usando arrow function. Ambas as funções devem receber um parâmetro carreira e retornar uma string concatenada usando template literals ${var}.
  - Retorno: "Meu objetivo de carreira é me tornar um [parametro carreira]"

PROFKELSON.DEV
BORA CODAR?

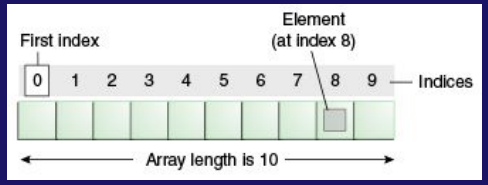
Arrays

- Em JS, Array é uma estrutura de dados que permite armazenar vários valores em uma única variável.
- Pode conter qualquer tipo de valor incluindo números, strings, objetos e até mesmo outros arrays.
- A primeira posição de um array sempre é indicada pelo índice “0” (zero).


```javascript
const myArray = [1, 2, 3, 4, 5];
```

PROFKELSON.DEV
BORA CODAR?

Manipulando Arrays



- Podemos criar um array com valores pré-definidos ou criar um array vazio e adicionar os valores posteriormente.
- A propriedade “push” vai ser a responsável por adicionar novos elementos a um array.
- Podemos acessar cada elemento individual de um array através do seu respectivo índice.

javascript
```javascript
const myArray = [1, 2, 3, 4, 5];
```

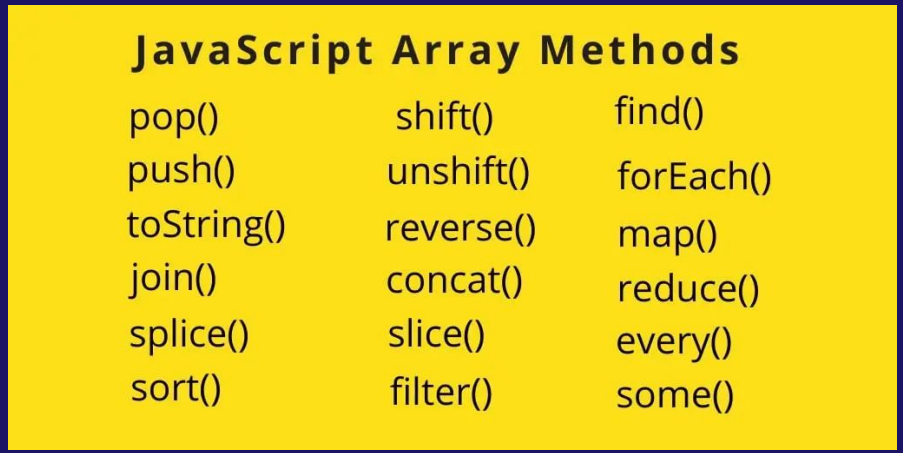
scss
```scss
const myArray = [];
myArray.push(1);
myArray.push(2);
myArray.push(3);
```

javascript
```javascript
console.log(myArray[0]); // imprime 1
console.log(myArray[2]); // imprime 3
```

PROFKELSON.DEV
BORA CODAR?

Manipulando Arrays

- Outras propriedades importantes dos arrays são:
  - lenght: retorna o número de elementos de um array
  - push(): adiciona um ou mais elementos no final do array
  - pop(): remove o último elemento e retorna-o
  - shift(): remove o primeiro elemento do array e retorna-o
  - unshift(): adiciona um ou mais elementos no início do array.



PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (3) [COM O PROF]

- Faça um script que contenha um array chamado notas, esse array armazena 3 notas de um aluno.
- O programa deve imprimir:
  - A primeira nota do aluno é: ...
  - A segunda nota do aluno é: ...
  - A média do aluno é: ...
  -
- OBS: Cálculo da média: (soma das notas)/quantidade de notas

VAMOS CODAR? (3_1) [SOZINHO(A)]

PROFKELSON.DEV
BORA CODAR?

- **Crie um script que contenha um array chamado temperaturas, que armazena 4 temperaturas em graus Celsius. O programa deve:**
  - **Imprimir todas as temperaturas armazenadas no array.**
  - **Selecionar uma temperatura aleatória do array.**
  - **Converter a temperatura selecionada de Celsius para Fahrenheit.**
  - **Imprimir a temperatura original em Celsius e sua conversão para Fahrenheit.**
  - **Para converter de Celsius para Fahrenheit:**
    - **$F=(C\times 9/5)+32$**
  - **Utilize a sintaxe *temperaturas[Math.floor(Math.random() * temperaturas.length)]* para selecionar um item aleatório do array.**

```
As temperaturas armazenadas são: [25, 30, 15, 20]
A temperatura sorteada é 30°C, que corresponde a 86°F.
```

PROFKELSON.DEV
BORA CODAR?

Objetos

- Em JS, um objeto é uma coleção de propriedades. Cada propriedade é uma chave-valor.
- Ao lado encontramos um exemplo de objeto literal em JS. Um objeto chamado “pessoa”, com três propriedades: nome, idade e cidade. Observe que cada propriedade/chave tem o seu respectivo valor.
- Podemos acessar o valor das propriedades como mostram os exemplos ao lado.
- Também podemos remover por completo uma propriedade presente em algum

javascript

```javascript
let pessoa = {
  nome: "João",
  idade: 30,
  cidade: "São Paulo"
};
```

javascript

```javascript
console.log(pessoa.nome); // "João"
console.log(pessoa["idade"]); // 30
```

perl Copy code

```perl
pessoa.email = "joao@gmail.com";
console.log(pessoa.email); // "joao@gmail.com"

delete pessoa.cidade;
console.log(pessoa); // {nome: "João", idade: 30, email: "joao@gmail.com"}
```

VAMOS CODAR? (4) [COM O PROF]

PROFKELSON.DEV
BORA CODAR?

- Crie um script que contenha um objeto literal chamado pessoa. O objeto deve armazenar informações sobre uma pessoa, incluindo:
  - nome: O nome da pessoa.
  - cpf: O CPF da pessoa.
  - cidade: A cidade onde a pessoa mora.
- Declarar e inicializar o objeto pessoa com valores para nome, cpf e cidade.
- Imprimir cada uma das propriedades do objeto pessoa usando console.log.

```
Kelson
089233554321
João Pessoa
```

VAMOS CODAR? (4_1) [SOZINHO(A)]

PROFKELSON.DEV
BORA CODAR?

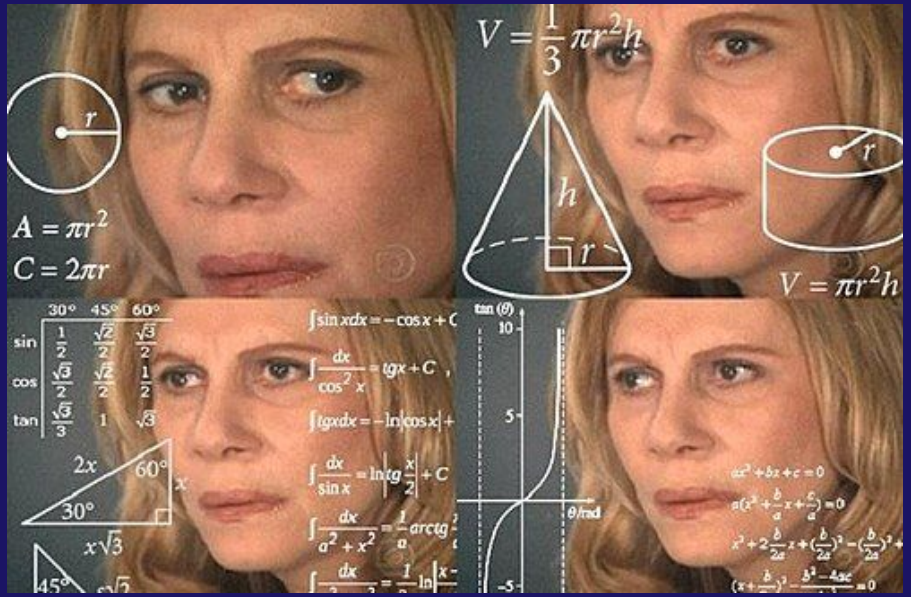
- Crie um script que combine o uso de objetos literais, arrays e cálculos simples para armazenar e manipular informações pessoais e acadêmicas de uma pessoa. O script deve realizar as seguintes tarefas:
  - nome: O nome do aluno.
  - cpf: O CPF do aluno.
  - cidade: A cidade onde o aluno mora.
  - notas: Um array com 4 notas do aluno.
- Imprimir o nome, CPF e cidade do aluno.
- Imprimir cada uma das notas do array notas
- Calcular a média das notas do aluno e imprimir
- Selecionar uma das notas aleatoriamente, convertê-la de uma escala de 0-10 para uma escala de 0-100
  - nota convertida = nota × 10
- Utilize a sintaxe notas[Math.floor(Math.random() * notas.length)] para selecionar uma nota aleatória do array.
- Utilize template literals para a concatenação de

```
Nome: Maria
CPF: 12345678900
Cidade: São Paulo
A primeira nota do aluno é: 8.5
A segunda nota do aluno é: 7.2
A terceira nota do aluno é: 9.0
A quarta nota do aluno é: 6.8
A média do aluno é: 7.875
A nota sorteada foi: 7.2 e convertida para a escala de 0-100 é: 72
```

PROFKELSON.DEV
BORA CODAR?

Operadores

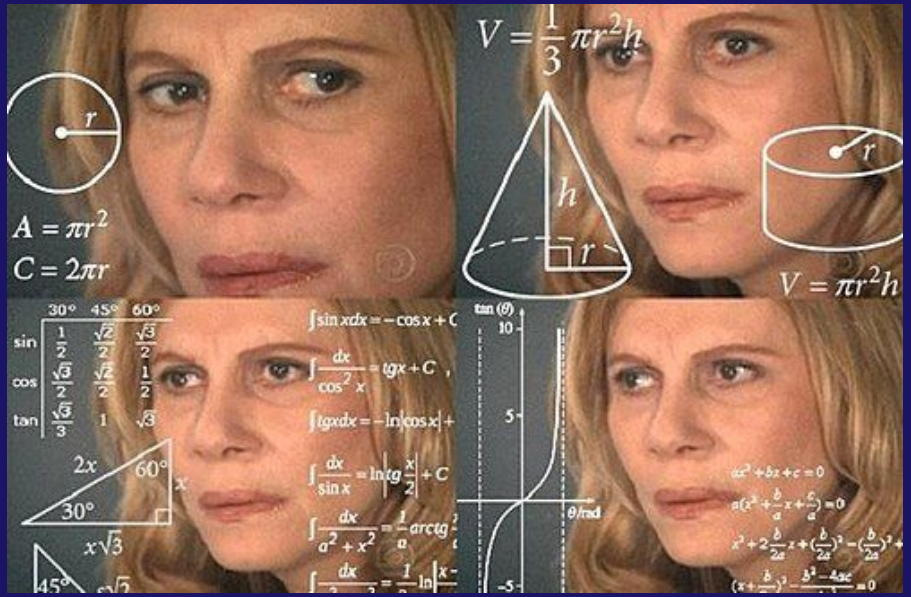
- São símbolos especiais que executam operações matemáticas ou lógicas em valores.
- Alguns tipos de operadores:
  - Operadores aritméticos
  - Operadores de comparação
  - Operadores lógicos
  - Operadores de atribuição



PROFKELSON.DEV
BORA CODAR?

Operadores

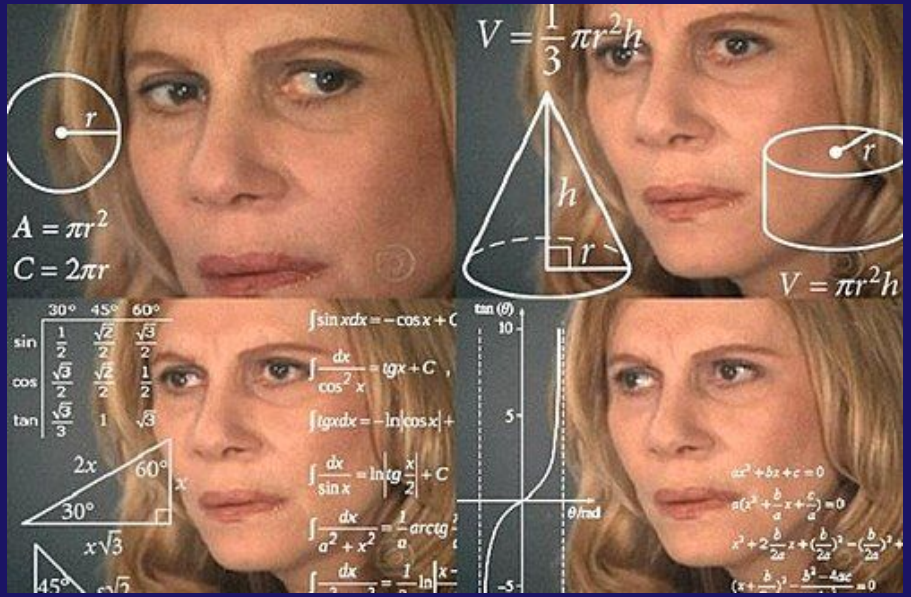
- Aritméticos:
  - + (adição)
  - - (subtração)
  - * (multiplicação)
  - / (divisão)
  - % (módulo / resto da divisão)
  - ++ (incremento)
  - - - (decremento)



PROFKELSON.DEV
BORA CODAR?

Operadores

- **Comparação:**
  - **== (igual a)**
  - **!= (diferente de)**
  - **> (maior que)**
  - **< (menor que)**
  - **>= (maior ou igual a)**
  - **<= (menor ou igual a)**
  - **=== (igual a em valor e tipo)**
  - **!== (diferente de em valor ou tipo)**

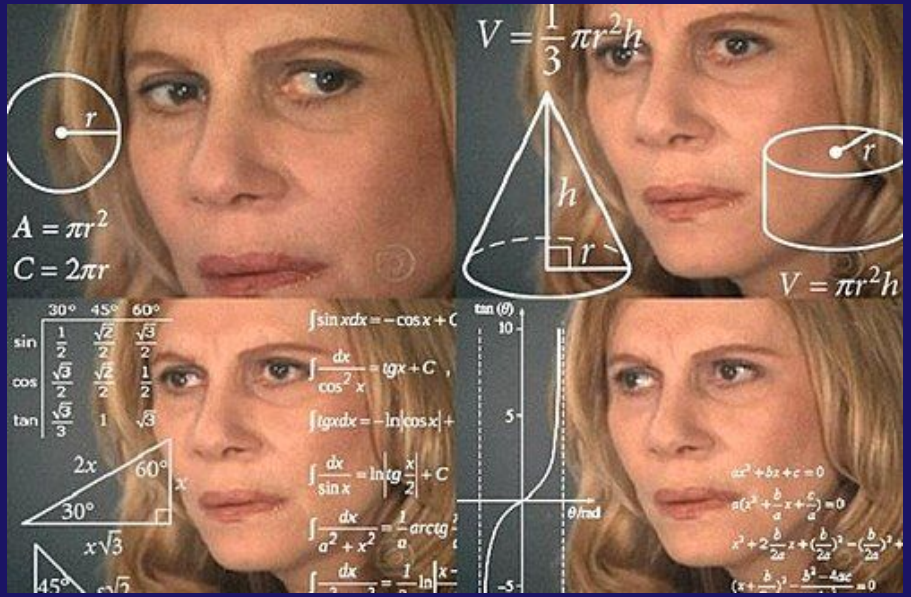


PROFKELSON.DEV
BORA CODAR?

Operadores

- **Lógicos:**
  - **&& (e lógico)**
  - **|| (ou lógico)**
  - **! (negação lógica)**

- **Atribuição:**
  - **= (atribuição simples)**
  - **+= (adição e atribuição)**
  - **-= (subtração e atribuição)**
  - ***= (multiplicação e atribuição)**
  - **/= (divisão e atribuição)**
  - **%= (módulo e atribuição)**



PROFKELSON.DEV
BORA CODAR?

Condicionais

- São utilizadas para executar diferentes blocos de código com base em uma condição.

- Geralmente criadas usando a palavra-chave “if” seguida de uma expressão entre parênteses. Se a expressão for avaliada como verdadeira, o bloco de código dentro das chaves é executado, se não for verdadeira, o bloco é ignorado.

javascript

```javascript
let idade = 18;
if (idade >= 18) {
  console.log("Você é maior de idade");
} else {
  console.log("Você é menor de idade");
}
```

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (5) [SOZINHO(A)]

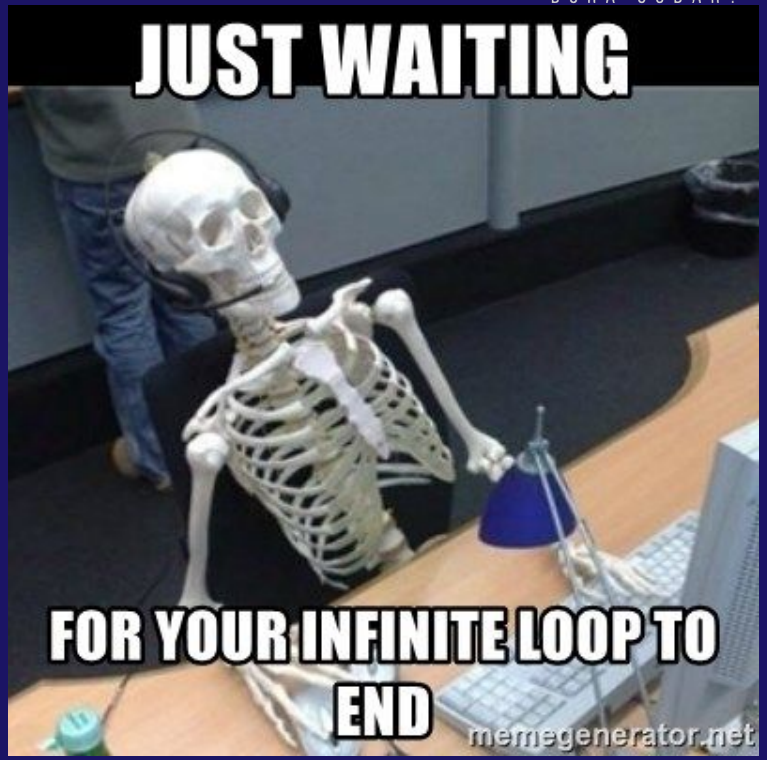
- Faça um script que contenha um array chamado notas, esse array armazena 3 notas de um aluno.
  - O programa deve imprimir:
    - A primeira nota do aluno é: ...
    - A segunda nota do aluno é: ...
    - A terceira nota do aluno é: ...
    - A média do aluno é: ...
    - O aluno está: APROVADO ou

PROFKELSON.DEV
BORA CODAR?

Loops

- São uma estrutura de controle em JS que permite repetir a execução de um bloco de código várias vezes.

- 3 tipos de loops em JS:
  - For
  - While
  - do-while

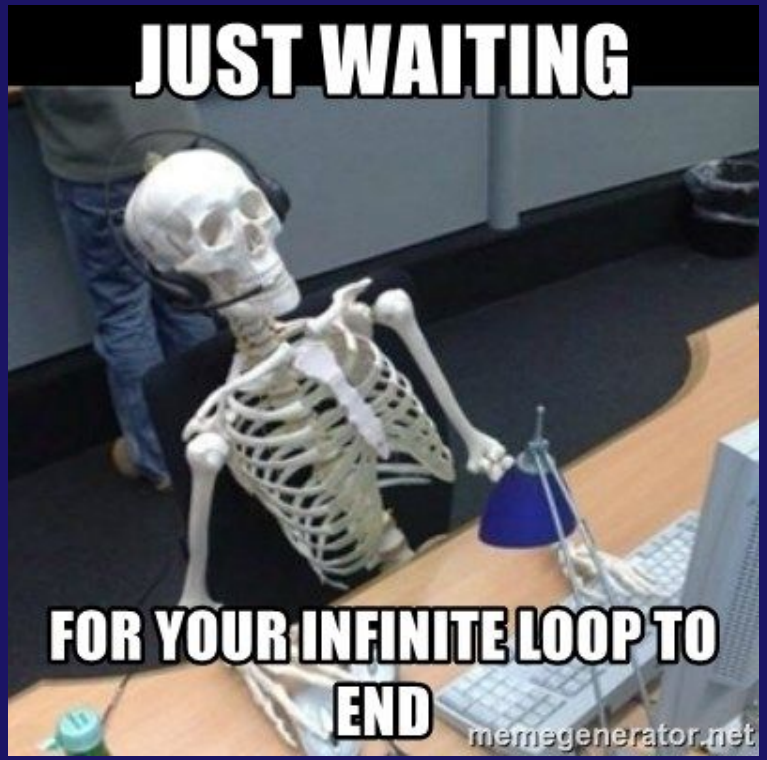


profkelson.tech

Loop for

- São uma estrutura de controle em JS que permite repetir a execução de um bloco de código várias vezes.

- 3 tipos de loops em JS:
  - For
  - While
  - do-while



PROFKELSON.DEV
BORA CODAR?

Loop for

- For é usado quando se sabe quantas vezes deseja repetir o bloco de código.
- Veja a sintaxe básica do for ao lado.
- A inicialização é uma expressão que executa apenas uma vez antes do início do loop.
- Utilizada para declarar e inicializar uma variável de controle.

```javascript
for (inicialização; condição; incremento) {
  // bloco de código a ser repetido
}
```

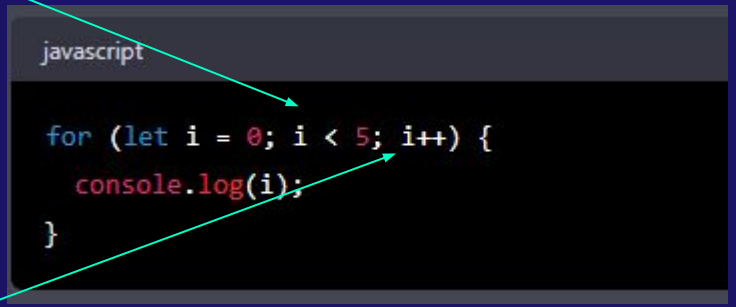
```javascript
for (let i = 0; i < 5; i++) {
  console.log(i);
}
```

PROFKELSON.DEV
BORA CODAR?

Loop for

- A condição é uma expressão que é testada no início de cada iteração do loop.
  - Se a condição é verdadeira, o bloco de código é executado. Se a condição for falsa, o loop é encerrado.

- O incremento é uma expressão que é executada no final de cada iteração do loop.
- É geralmente utilizada para incrementar a variável de controle.



PROFKELSON.DEV
BORA CODAR?

Loop while

- É utilizado quando não se sabe quantas vezes deseja repetir o bloco de código.
- Veja a sintaxe do while ao lado.
- A condição é uma expressão que é testada no início de cada iteração do loop.
- Se a condição é verdadeira, o bloco é executado. Se a condição é falsa, o loop é encerrado.

javascript

```javascript
let i = 0;
while (i < 5) {
  console.log(i);
  i++;
}
```

PROFKELSON.DEV
BORA CODAR?

Método .map

- O método .map() é uma função de array em JavaScript.
- Ele cria um novo array com os resultados da aplicação de uma função a cada elemento do array original.

javascript

```javascript
const numeros = [1, 2, 3, 4, 5];
const numerosDobrados = numeros.map(numero => numero * 2);

console.log(numerosDobrados);
// Saída: [2, 4, 6, 8, 10]
```

```javascript
const numeros = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

const numerosTransformados = numeros.map(numero => {
    if (numero % 2 === 0) {
        return `Par: ${numero}`;
    } else {
        return `Ímpar: ${numero}`;
    }
});

console.log(numerosTransformados);
// Saída: ["Ímpar: 1", "Par: 2", "Ímpar: 3", "Par: 4", "Ímpar: 5", "Par: 6", "Ímpar: 7",
```

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? (6) [COM O PROF]

- **Escreva um programa que imprima os números de 1 a 100. Mas, para múltiplos de 3, imprima "Fizz" em vez do número e, para múltiplos de 5, imprima "Buzz". Para números que são múltiplos de ambos 3 e 5, imprima "FizzBuzz".**

- **Dica: Exemplo para verificar se um número é múltiplo de 3:**
  - **número % 3 == 0**

PROFKELSON.DEV
BORA CODAR?

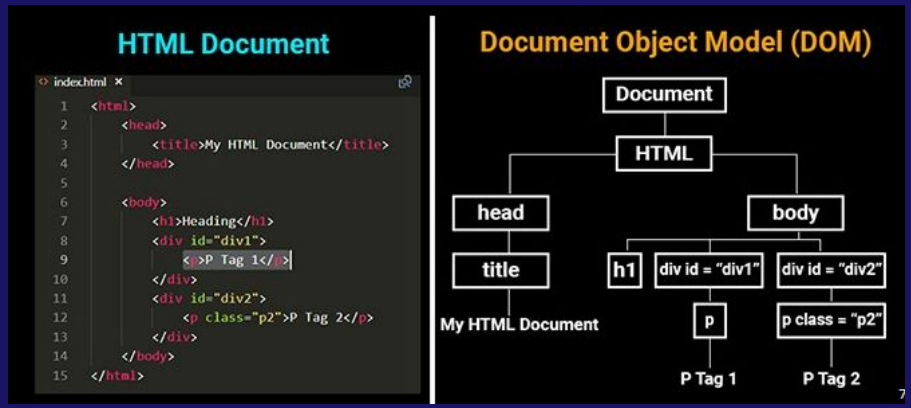
VAMOS CODAR? (6_1) [SOZINHO(A)]

- Refaça o exercício anterior utilizando .map
- Para criar um array com números de 1 a 100:
  - const numeros = Array.from({ length: 100 }, (_, index) => index + 1);

PROFKELSON.DEV
BORA CODAR?

Manipulação de DOM

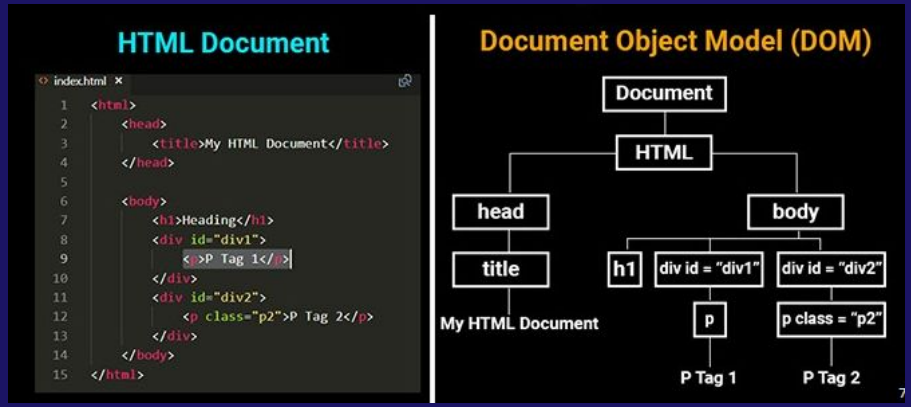
- Manipulação de DOM (Document Object Model) é uma técnica usada em programação web para modificar o conteúdo, a estrutura ou estilo de uma página web após ela ser carregada pelo navegador.
- Mas o que é DOM, professor?
- DOM é uma representação, em memória, da estrutura da página web.
- Estrutura essa que é criada pelo navegador a partir do código HTML enviado pelo servidor web.



PROFKELSON.DEV
BORA CODAR?

Manipulação de DOM

- A manipulação de DOM é realizada principalmente utilizando JavaScript. Existem várias maneiras de manipular a DOM com JavaScript, incluindo:
  - Modificar estilos:
    - backgroundColor
    - color
    - fontSize

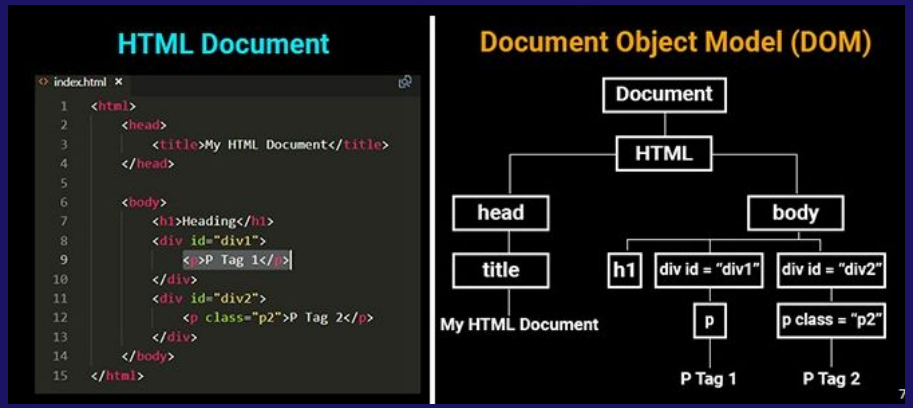


PROFKELSON.DEV
BORA CODAR?

Manipulação de DOM

- A manipulação de DOM é uma técnica poderosa e flexível, que permite aos devs criar páginas web dinâmicas e interativas. Porém, se utilizada em excesso pode levar a um desempenho lento da página.

- Por isso, é importante usá-lo com cuidado e de forma eficiente! :)



VAMOS CODAR? (7) [COM O PROF]

- Suponha que temos a seguinte página HTML
- Crie um script em JS (no arquivo script.js) que tenha uma função “mudaTexto()”, ela deve ser responsável por mudar o texto do elemento <h1> para “Novo Título” quando o botão for clicado.
- Dica: para selecionar o elemento você pode utilizar document.getElementById(“id-do-elemento”)
- Com esse elemento selecionado você pode mudar o texto: variavelQueSelecionouOElemento = innerText = “Novo Título”

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Manipulação de DOM</title>
  </head>
  <body>
    <h1 id="titulo">Título da Página</h1>
    <p id="paragrafo">Este é um parágrafo de exemplo.</p>
    <button id="botao" onclick="mudaTexto()">Clique Aqui</button>
    <script src="script.js"></script>
  </body>
</html>
```

PROFKELSON.DEV
BORA CODAR?

Eventos em JS

- **Eventos em JavaScript são ações ou ocorrências que acontecem dentro de uma página web, por exemplo, como o clique em um botão, a digitação em um campo de uma formulário, carga da página ou a mudança de estado de um elemento.**

javascript

```javascript
const meuBotao = document.getElementById('meu-botao');

meuBotao.addEventListener('click', function() {
  // código para executar quando o botão for clicado
});
```

PROFKELSON.DEV
BORA CODAR?

Eventos em JS

- O JavaScript permite que você capture esses eventos e crie respostas personalizadas a eles.
- Por exemplo, você pode criar uma função que será executada quando um usuário clicar em um botão.
- Existem muitos tipos de eventos em JS, incluindo eventos do mouse (cliques e movimentos), eventos do teclado (ex: pressionamento de teclas), eventos de formulários (envio e reset), eventos de página (carregamento / descarregamento), eventos de animação (início e término de animações) e muitos outros.

javascript

```javascript
const meuBotao = document.getElementById('meu-botao');

meuBotao.addEventListener('click', function() {
  // código para executar quando o botão for clicado
});
```

PROFKELSON.DEV
BORA CODAR?

Eventos em JS

- Para capturar um evento em JS, você precisa adicionar um ouvinte de eventos ao elemento HTML correspondente.
- O método addEventListener(), no exemplo ao lado, vai capturar o evento de clique em um botão.
- Neste exemplo, o "getElementById()" é utilizado para selecionar o elemento HTML com o ID "meu-botao", em seguida um ouvinte de eventos é adicionado a ele usando o método "addEventListner()".
- A função passada como segundo argumento será executada sempre que o botão for clicado.

javascript

```javascript
const meuBotao = document.getElementById('meu-botao');

meuBotao.addEventListener('click', function() {
  // código para executar quando o botão for clicado
});
```

PROFKELSON.DEV
BORA CODAR?

Bora treinar mais?

- Agora com o JS dentro do HTML

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- Dentro do seu workspace/aula02, crie uma pasta chamada: trabalhando_com_js_mais_html
- Crie uma página HTML chamada:
  - 1_conectando_o_js.html
- Crie uma pasta “scripts” e dentro dela crie um arquivo:
  - script_1.js
- Faça um <h1> que contenha a mensagem:
  - *Foi exibido no alert o resultado de 2+2*
- Faça que um alert (em JS) dê o

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 2_manipulando_elementos.html:

- Crie um html com o h1 que tem o id “myHeader”
  - Esse h1 tem a seguinte frase: “Essa frase não será exibida e será trocada pelo JS”
  - No fim do body importe o “script_2.js”:
    - Esse script terá uma variável que recebe o elemento de id ‘myHeader’
    - Um “innerHTML é aplicado nessa variável para mudar o valor contido no h1 para: “Essa frase vem do JS”
    - O JS também tem que mudar a cor da frase para azul (blue).

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

3_eventos_e_manipulacao.html:

- **Crie um html com um botão de id “myButton” e a frase “clica em mim”**
  - **Importe ao do body o script_3.js:**
    - **Crie uma variável chamada “myButton” que recebe o elemento html de id ‘myButton’**
    - **Com essa variável crie um “listener” que espera um “click” e aciona a função com um alerta “botão clicado!”**

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

4_variaveis_e_estrutura_de_controle.html:

- **Crie um html apenas com o título Exercício 4**
  - **Importe ao do body o script_4.js:**
    - **Crie uma variável chamada idade que recebe o valor 20**
    - **Logo abaixo faça uma estrutura condicional que verifica SE a idade é maior que 18, se sim:**
      - **Imprime na tela: "Você é maior de idade"**
    - **Se não:**
      - **Imprime na tela: "Você é menor de idade"**

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

5_arrays_e_loops.html:

- **Crie um html apenas com o título Exercício 5**
  - **Importe ao do body o script_5.js:**
    - **Crie um variável que armazena um array com o nome de 3 frutas.**
    - **Crie um loop que percorre esse array e imprime o nome de cada letra (use a estrutura de map)**

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

6_funcoes.html:

- **Crie um html apenas com o título Exercício 6**
  - **Importe ao do body o script_6.js:**
    - **Crie uma função**

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

Bora fazer uma calculadora?

Segue o prof! :D

PROFKELSON.DEV
BORA CODAR?

Bootstrap

- https://getbootstrap.com/
- Bootstrap é um framework front-end gratuito e de código aberto.
  - Desenvolvido pelo Twitter e lançado como um projeto de código aberto em 2011.
  - Facilitar o desenvolvimento web, especialmente o design responsivo e a criação de interfaces consistentes.

PROFKELSON.DEV
BORA CODAR?

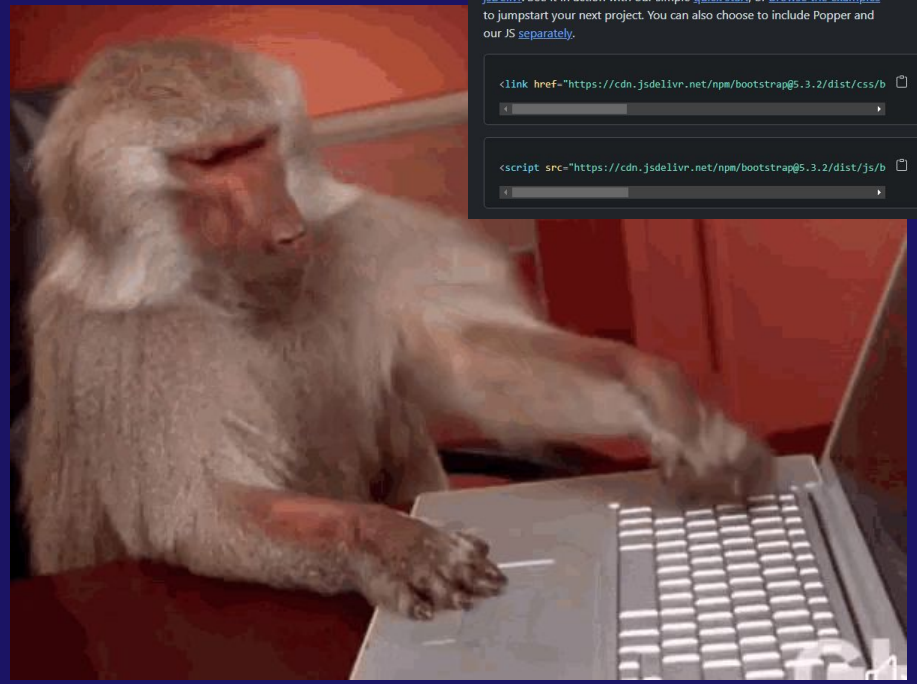
Bootstrap

- Por que usar Bootstrap?
  - Oferece um sistema de grade flexível, facilitando o design responsivo.
  - Vários componentes prontos para uso, como botões, formulários, alertas, etc.
  - Suporte a navegadores modernos, tornando-o uma escolha segura.

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- No workspace do curso, crie uma pasta aula02/trabalhando_com_bootstrap
- Crie o arquivo:
  - 1_importando_o_bootstrap.html
- Crie a estrutura base de um arquivo html
- Importe o script (js) do bootstrap e o seu css



PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 2_grid_system.html:
- Com o arquivo acima crie o sistema de grid com 3 colunas
- https://getbootstrap.com/docs/5.3/layout/grid/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 3_tipografia.html:
- Com o arquivo acima crie 3 textos com diferentes tipografias.
- https://getbootstrap.com/docs/5.3/content/typography/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 4_botoes_e_icones.html:
- Com o arquivo acima, crie dois botões (um primário e um secundário).
- https://getbootstrap.com/docs/5.3/components/buttons/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 5_formularios.html:
- Com o arquivo acima, crie um formulário para o preenchimento de nome e email.
- https://getbootstrap.com/docs/5.3/forms/overview/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 6_tabelas.html:
- Com o arquivo acima, crie uma tabela com três colunas (id, nome e email). Preencha essa tabela com as informações de duas pessoas.
- https://getbootstrap.com/docs/5.3/content/tables/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 7_alertas_e_mensagens.html:
- Com o arquivo acima, crie três alertas:
  - Sucesso: Sucesso! Sua operação foi concluída com êxito.
  - Atenção: Atenção! Algo pode estar errado.
  - Perigo: Erro! Algo deu errado.
- https://getbootstrap.com/docs/5.3/components/alerts/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 8_navegacao.html:
- Com o arquivo acima, crie uma barra de navegação com 3 links para o site: Home, Sobre e Contato.
- https://getbootstrap.com/docs/5.3/components/navbar/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 9_componentes_modais.html:
- Com o arquivo acima, crie um botão que abre um modal com a mensagem:
  - Título: Cadastrado com sucesso
  - Mensagem: Você foi cadastrado no nosso sistema!
- https://getbootstrap.com/docs/5.3/components/modal/

PROFKELSON.DEV
BORA CODAR?

VAMOS CODAR? [COM O PROF/DEPOIS SOZINHO(A)]

- 10_carrossel.html:
- Com o arquivo acima, crie um carrossel com 3 imagens a sua escolha.
- https://getbootstrap.com/docs/5.3/components/carousel/