

A woman with long dark hair, wearing glasses, is shown from the chest up, looking down at a laptop screen. The background is dark and slightly blurred.

IN

# INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 10 - PROJETO

# O QUE IREMOS APRENDER

**01**

REVISÃO

**02**

PROJETO

IN

# REVISÃO - CONDICIONAIS

As estruturas condicionais permitem que o programa tome decisões baseadas em condições específicas. As estruturas mais comuns são if, else if e else. Elas permitem que o código execute diferentes blocos de código com base em testes de condição.



```
1 if (condicao) {  
2     // Código que será executado caso a condição seja verdadeira  
3 } else if (outraCondicao){  
4     // Código que será executado caso a primeira condição seja falsa e a segunda seja verdadeira  
5 } else {  
6     // Código que será executado caso todas as condições forem falsas  
7 }
```



```
1 let numero = 5  
2  
3 if (numero > 10) {  
4     console.log("O número é maior que 10 ")  
5 } else if (numero == 5) {  
6     console.log("O número é igual a 5 ")  
7 } else {  
8     console.log("O numero é menor que 5 ")  
9 }
```

# REVISÃO - FUNÇÕES

As funções são blocos de código reutilizáveis que executam tarefas específicas. Elas ajudam a organizar o código em partes menores e mais gerenciáveis, promovendo a reutilização e a modularidade. Funções podem receber argumentos e retornar valores.



```
1 <script>
2  /* No exemplo abaixo temos uma função que se chama calcularAreaTerreno,
3  e como parâmetros temos largura e comprimento. */
4  function calcularAreaTerreno(largura, comprimento){
5  /* criamos uma variável chamada area que calcula a área que é a largura*comprimento */
6  let area = largura*comprimento
7  /* utilizamos o return para finalizar a execução de uma função e especificar
8  os valores que devem ser retomados para onde a função foi chamada */
9  return area
10 }
11 </script>
```

# REVISÃO - FUNÇÕES

Por exemplo:

```
● ● ●  
1 <body>  
2     <input type="number" id="num1">  
3     <input type="number" id="num2">  
4     <button id="enviar">ENVIAR</button>  
5     <script>  
6  
7         let enviar = document.getElementById('enviar').addEventListener('click', somar)  
8  
9         function somar() {  
10             let num1 = Number(document.getElementById('num1').value)  
11             let num2 = Number(document.getElementById('num2').value)  
12             let soma = num1 + num2  
13             console.log(soma)  
14         }  
15     </script>
```

# REVISÃO - MANIPULAÇÃO DO DOM

O DOM representa a estrutura de objetos de uma página web e permite a interação e manipulação desses objetos usando JavaScript. A manipulação do DOM envolve a atualização de elementos, conteúdo, estilos e atributos da página. Algumas operações comuns incluem a seleção de elementos, modificação de texto, adição ou remoção de elementos e alteração de estilos.

# REVISÃO - FUNÇÕES

Por exemplo:



```
1 <body>
2   <div id="idDoElemento"></div>
3   <script>
4     // Selecionar um elemento pelo ID
5     const elemento = document.getElementById('idDoElemento');
6
7     // Modificar o texto de um elemento
8     elemento.innerHTML = 'Novo texto';
9
10    // Adicionar um novo elemento filho
11    const novoElemento = document.createElement('div');
12    novoElemento.innerHTML = 'Novo elemento';
13    elemento.appendChild(novoElemento);
14
15    // Alterar o estilo de um elemento
16    elemento.style.color = 'red';
17
18  </script>
19 </body>
```



# REVISÃO - EVENTOS DO DOM

Os eventos do DOM (Document Object Model) são ações que ocorrem em elementos HTML e que podem ser capturadas e tratadas por meio do JavaScript. Alguns exemplos de eventos incluem cliques de mouse, pressionamento de teclas, foco em elementos, envio de formulários etc. Você pode atribuir funções a eventos para responder às ações do usuário.

# REVISÃO - FUNÇÕES

Por exemplo:



```
1 <body>
2   <button id="enviar">ENVIAR</button>
3   <script>
4
5     let enviar = document.getElementById('enviar').addEventListener('click', mostrar)
6
7     function mostrar() {
8
9       const novoElemento = document.createElement('div');
10      novoElemento.innerHTML = 'Raama Batista';
11      document.body.appendChild(novoElemento);
12    }
13  </script>
14 </body>
```

# ATIVIDADE PRÁTICA

## **Atividade 01**

Crie um formulário com campos de nome, email e senha. Adicione um evento de submit ao formulário que valida se todos os campos foram preenchidos. Se não, exiba uma mensagem de erro.

## **Atividade 02**

Crie um formulário com um campo de texto e um botão. Ao clicar no botão, adicione um novo elemento à página com o texto digitado no campo de texto.

# ATIVIDADE PRÁTICA

## **Atividade 03**

Crie uma função que recebe um array de números e retorna a soma deles.

## **Atividade 04**

Crie uma função que calcula o Índice de Massa Corporal (IMC) e retorna se a pessoa está abaixo do peso, no peso ideal ou acima do peso.

# PROJETO

## Projeto da Lista de Tarefas (To-Do List) em JavaScript

Você deve criar uma aplicação web simples para gerenciar uma lista de tarefas. A aplicação deve permitir que o usuário adicione novas tarefas, marque tarefas como concluídas e remova tarefas da lista.

### Requisitos:

#### 1. Adição de Tarefas:

- A aplicação deve ter um campo de entrada de texto e um botão para adicionar novas tarefas.

# PROJETO

- Ao inserir uma nova tarefa e clicar no botão, a tarefa deve ser adicionada à lista de tarefas.

## 2. Remoção de Tarefas:

- Cada tarefa na lista deve ter um botão de remoção associado.
- Ao clicar no botão de remoção, a tarefa correspondente deve ser removida da lista.

## 3. Marcação de Tarefas Concluídas:

- Cada tarefa na lista deve ter uma caixa de seleção (checkbox).

# PROJETO

- Ao marcar a caixa de seleção, a tarefa deve ser visualmente marcada como concluída (por exemplo, com um texto riscado).
- A tarefa ainda deve permanecer na lista, mesmo quando marcada como concluída.

## 4. Estilo e Layout:

- A aplicação deve ter um estilo agradável usando CSS para torná-la visualmente atraente.
- A lista de tarefas deve ser exibida de maneira organizada.

# PROJETO

## Implementação:

- Utilize HTML para a estrutura da página.
- Use JavaScript para manipulação do DOM e lógica da aplicação.
- Estilize a aplicação com CSS para uma apresentação agradável.

## Observações:

- Evite o uso de objetos para representar as tarefas.
- Concentre-se em manipulação do DOM, eventos e funções para gerenciar as tarefas.

# SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 11 DE JAVASCRIPT.  
FUNÇÕES AGREGADORAS



INFINITY SCHOOL  
VISUAL ART CREATIVE CENTER

# EVENTOS DOM

As funções agregadoras, também conhecidas como funções de agregação ou funções agregadas, são funções que realizam cálculos em um conjunto de valores e retornam um único resultado. Essas funções são frequentemente usadas em conjunto com conjuntos de dados, como arrays ou listas, para realizar operações de resumo.



# EVENTOS DOM

Exemplos da **funções agregadoras** em JavaScript:



```
1 const arr = [1, 2, 3, 4, 5]
2 const arrManipulado = arr.join('-')
3 console.log(arrManipulado)
```

1-2-3-4-5

IN

A woman with long, dark hair and glasses is shown from the chest up, looking down at a laptop screen. The background is dark and slightly blurred.

IN

# INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 10 - PROJETO