PROGRAMAÇÃO FRONT END

João Choma Neto
joao.choma@unicesumar.edu.br
https://github.com/JoaoChoma/frontend
Unicesumar – Maringá

EVENTOS E FUNÇÕES

A linguagem de programação JavaScript proporciona a integração de conceitos lógicos, funções e orientações a objetivos aplicados com os elementos do HTML, assim, podemos utilizar formulário para criar eventos e funções

EVENTOS E FUNÇÕES

- Para utilizar um evento, deve utilizar uma função
- A FUNÇÃO já é interpretada pelo compilador do JavaScript
- *(on)click*, *(on)change*, *(on)submit*, *(on)load*, entre outros

FUNÇÃO

```
1. function nomeDaFuncao(){
2.         código
3. }
```

O QUE É PARÂMETRO

Um parâmetro de uma função é um valor que pode ser passado para a função quando ela é invocada

O QUE É PARÂMETRO

É uma forma de fornecer informações à função para que ela possa executar uma ação específica ou retornar um resultado baseado nos valores recebidos

O QUE É PARÂMETRO

```
function nomeDaFuncao(parametro1,
  parametro2, parametro3) {
  // Corpo da função que usa os parâmetros
}
```

O QUE É PARÂMETRO

```
function somar(a, b) {
  return a + b;
}
let resultado = somar(3, 5);
```

O QUE É RETURN

Estas funções se tornam mais úteis se retornam um valor específico

Para fazer um programa retornar um valor específico, utilizamos o comando ***"return"***, seguido do conteúdo que desejamos retornar

RETURN

```
function somar(a, b) {
  return a + b;
}
let resultado = somar(3, 5);
```

FUNÇÃO ANÔNIMA

Nas funções anônimas, é definida a programação de um evento sem atribuir um nome para a função

Utilizamos o comando ***"function()"*** e inserimos a lógica da programação que necessitamos resolver

FUNÇÃO ANÔNIMA

```
1. var dobro = function(a) {
3.    return a * 2;
4. }
5. var num = Number(prompt("Número: "));
6. alert("O dobro é: " + dobro(num));
```

OUTRA FORMA DE ESCREVER A FUNÇÃO

```
1. function situacaoAluno(nota, media) {
2.         var situacao = (nota >= media) ? “Aprovado” : “Reprovado”;
3.         return situacao;
4. }
```

O sinal de interrogação "?" é a primeira parte da estrutura condicional e o sinal de ":" seria a parte do ***"else"*** (se não)

EVENTOS

REAÇÃO

Quando JavaScript é usado em páginas HTML, o JavaScript pode “reagir” a esses eventos.

Um evento HTML pode ser algo que o navegador faz ou algo que um usuário faz.

REAÇÃO

Uma página da web HTML terminou de carregar

Um campo de entrada HTML foi alterado

Um botão HTML foi clicado

EVENTOS

onchange - Um elemento HTML foi alterado

onclick - O usuário clica em um elemento HTML

onmouseover O usuário move o mouse sobre um elemento HTML

EVENTOS

onmouseout - O usuário afasta o mouse de um elemento HTML

onkeydown - O usuário pressiona uma tecla do teclado

onload - O navegador terminou de carregar a página

DOM

DOM

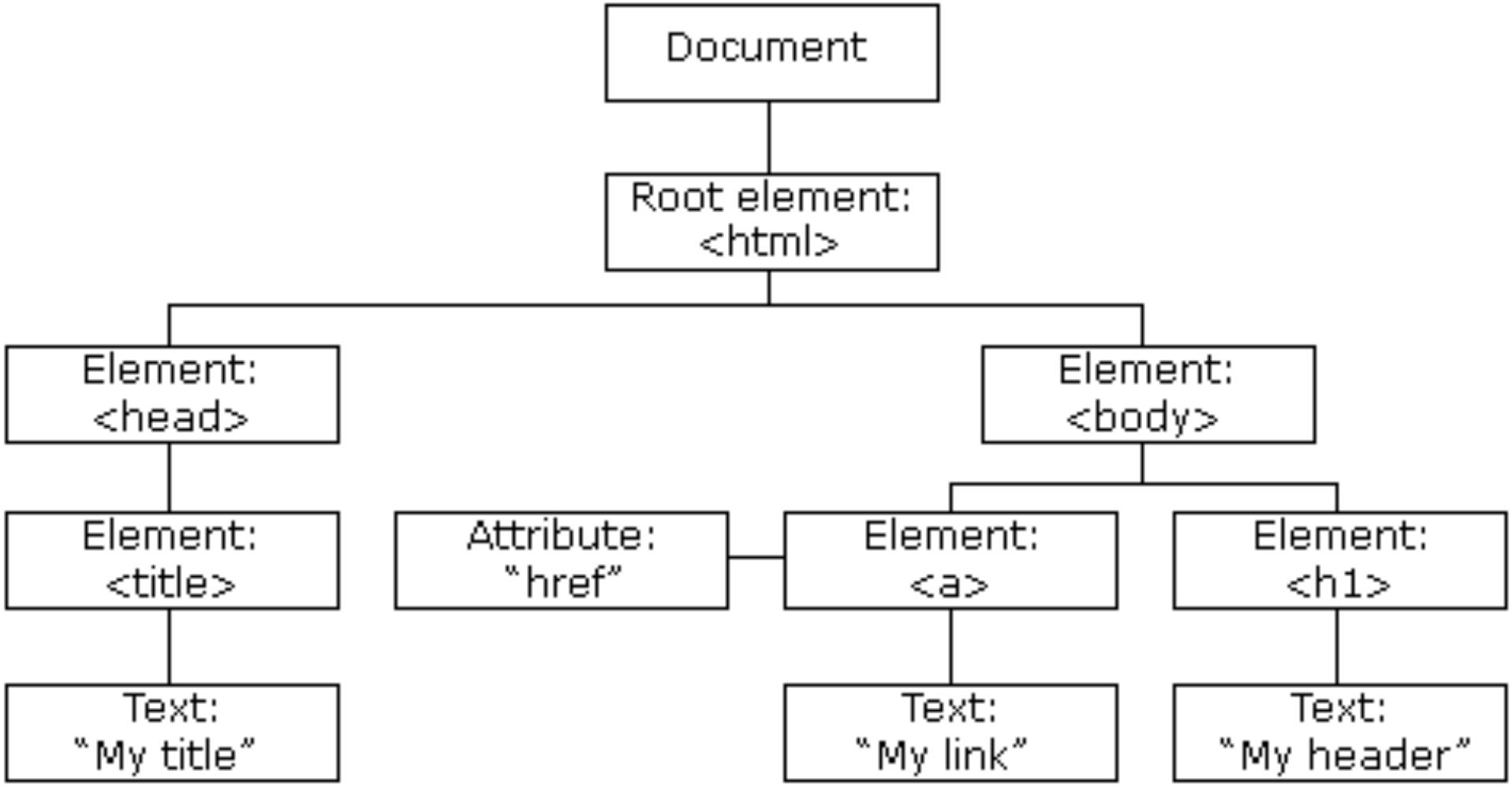
O DOM (Document Object Model) do HTML é uma interface de programação para documentos HTML e XML. Ele representa a página para que programas possam mudar a estrutura do documento, seu estilo e conteúdo.

O DOM representa o documento como uma árvore de objetos; cada objeto corresponde a uma parte do documento, como um elemento, atributo ou texto.

DOM

Quando uma página é carregada no navegador, ela é convertida em um DOM. Isso permite que scripts, como JavaScript, interajam com a estrutura da página, acessando e modificando elementos HTML dinamicamente.

Por exemplo, você pode usar JavaScript para adicionar, remover ou alterar elementos HTML, modificar atributos de elementos e responder a eventos na página, como cliques ou movimentos do mouse, tudo através do DOM.


Document
Root element:
<html>
Element:
<head>
Element:
<body>
Element:
<title>
Attribute:
"href"
Element:
<a>
Element:
<h1>
Text:
"My title"
Text:
"My link"
Text:
"My header"

Métodos disponíveis

https://www.w3schools.com/jsref/dom_obj_document.asp

EXEMPLO

O método **addEventListener**() anexa um manipulador de eventos a um documento

EXEMPLO

O método **createElement** é usado para criar um novo elemento no documento. Ele é um método do objeto document

Quando você cria um elemento, ele não aparece automaticamente na página; você precisa inseri-lo no documento usando métodos como **appendChild** ou **insertBefore**

EXEMPLO

Enquanto createElement é usado para criar um novo elemento, **removeChild** é um método usado para remover um elemento existente da árvore DOM

Este método é chamado no elemento pai do elemento que você deseja remover

MODULARIZAÇÃO

MODULARIZAÇÃO

Na aula de hoje, discutiremos o conceito de modularização no desenvolvimento frontend e como aplicá-lo em projetos web. A modularização é uma prática essencial para manter o código organizado, reutilizável e fácil de manter.

O QUE É MODULARIZAÇÃO

- **Modularização** é o processo de dividir um sistema em partes independentes e autônomas, chamadas de módulos.
- Cada módulo tem uma responsabilidade específica e pode ser desenvolvido, testado e mantido separadamente dos outros módulos.

O QUE É MODULARIZAÇÃO

- Em desenvolvimento frontend, a modularização envolve dividir o código JavaScript, HTML e CSS em módulos distintos, cada um responsável por uma funcionalidade ou conjunto de funcionalidades específicas.

POR QUE MODULARIZAR?

- **Reutilização de código**: A modularização permite reutilizar módulos em diferentes partes do aplicativo, economizando tempo e esforço de desenvolvimento.

POR QUE MODULARIZAR?

- **Facilidade de manutenção**: Módulos independentes são mais fáceis de entender, testar e modificar, facilitando a manutenção do código ao longo do tempo.

POR QUE MODULARIZAR?

- **Escalabilidade**: Projetos modulares são mais fáceis de escalar à medida que o aplicativo cresce, pois novos recursos podem ser adicionados como novos módulos sem afetar o código existente.

COMO MODULARIZAR NO FRONTEND?

- **Divisão em Componentes**: No frontend, os componentes são a unidade básica de modularização. Cada componente é responsável por uma parte específica da interface do usuário e encapsula seu próprio comportamento e estilo.

COMO MODULARIZAR NO FRONTEND?

- **Arquivos Separados**: Separe cada componente em arquivos JavaScript, HTML e CSS separados. Isso ajuda a manter o código organizado e facilita a localização e edição de código relacionado.

COMO MODULARIZAR NO FRONTEND?

- **Módulos JavaScript**: Use módulos JavaScript para organizar e encapsular funcionalidades relacionadas. Isso pode ser feito usando a sintaxe import e export do ECMAScript.

OBJETOS

O QUE SÃO OBJETOS?

- **Objetos** em JavaScript são estruturas de dados que permitem armazenar e organizar informações relacionadas em pares de chave-valor.
- Cada valor é acessado por meio de uma chave exclusiva, tornando os objetos uma estrutura de dados eficiente para representar e manipular dados complexos.

ESTRUTURA DE UM OBJETO

- Um **objeto** em JavaScript é definido usando a sintaxe de chaves {} e pode conter zero ou mais pares de chave-valor.
- Cada par consiste em uma chave única seguida por dois pontos : e o valor correspondente.
- Os pares de chave-valor são separados por vírgulas.

OBJETO JS

```
const pessoa = {
  nome: 'João',
  idade: 30,
  cidade: 'São Paulo’
};
```

ACESSANDO PROPRIEDADES DE UM OBJETO

- As propriedades de um objeto podem ser acessadas usando a notação de ponto . ou a notação de colchetes [].

```
console.log(pessoa.nome); // Saída: João
console.log(pessoa['idade']); // Saída: 30
```

ACESSANDO PROPRIEDADES DE UM OBJETO

- Novas propriedades podem ser adicionadas a um objeto atribuindo um valor a uma chave que ainda não existe.

```
pessoa.profissao = 'Desenvolvedor';
pessoa.idade = 31;
```

ACESSANDO PROPRIEDADES DE UM OBJETO

- Propriedades de um objeto podem ser removidas usando o operador delete.

```
delete pessoa.cidade;
```

ACESSANDO PROPRIEDADES DE UM OBJETO

- Além de armazenar dados, os objetos em JavaScript também podem conter métodos, que são funções definidas como propriedades.

```
const pessoa = {
  nome: 'Maria',
  idade: 25,
  saudacao: function() {
    return 'Olá, meu nome é ' + this.nome + '!';
  }
};

console.log(pessoa.saudacao()); // Saída: Olá, meu nome é Maria!
```

this

- O this é uma palavra-chave especial em JavaScript que é usada para se referir ao objeto atual em que um determinado código está sendo executado.
- O valor de this é determinado pelo contexto de execução no qual o código está sendo executado e pode variar dependendo de como e onde uma função é chamada.

this

- Quando uma função é chamada como um método de um objeto, o valor de this é o próprio objeto no qual o método é chamado.
- O this é usado para acessar outras propriedades e métodos do objeto dentro do método.
- Quando uma função de evento é acionada em resposta a uma interação do usuário, como um clique de botão, o valor de this geralmente se refere ao elemento DOM que disparou o evento.

JSON

JSON

- JSON significa JavaScript Object Notation.
- É um formato leve e fácil de ler para troca de dados.
- Baseado na sintaxe de objetos JavaScript.
- Independente de linguagem.

JSON

- Consiste em pares chave-valor.
- Os valores podem ser strings, números, objetos, arrays, booleanos ou nulos.
- Exemplo:

{ "nome": "Maria", "idade": 25, "cidade": "Rio de Janeiro", "ativos": ["caminhada", "leitura", "cozinhar"], "casada": false, "endereco": null }

- Os objetos são delimitados por chaves {} e os pares chave-valor são separados por vírgulas.

Por que JSON é Importante?

- Ampla adoção na web devido à sua simplicidade.
- Facilita a troca de dados entre servidores e clientes.
- Usado em APIs, armazenamento de dados, etc.

Como Usar JSON em JavaScript

- JSON.parse(): Converte uma string JSON em um objeto JavaScript.
- JSON.stringify(): Converte um objeto JavaScript em uma string JSON.

PROGRAMAÇÃO FRONT END

João Choma Neto
joao.choma@unicesumar.edu.br
https://github.com/JoaoChoma/frontend
Unicesumar – Maringá