

Desenvolvimento Web II

Aula 11 - Web Components

Prof. Fabricio Bizotto

Instituto Federal Catarinense
fabricio.bizotto@ifc.edu.br

Ciência da Computação
5 de fevereiro de 2024

1 Design System

2 Web Components

- ciclo de vida
- DOM vs Shadow DOM
- HTML Templates e Slots
- HTML Imports - *Descontinuado*

3 Prós e Contras

4 Exemplo

5 Material Complementar

6 Experimento

- Design System é um conjunto de **padrões, práticas, diretrizes e componentes** que são usados para criar e manter produtos digitais.
- O objetivo é garantir a consistência e a qualidade do produto final.
- Um Design System inclui:
 - **Componentes:** Botões, formulários, cards, etc.
 - **Estilos:** Cores, tipografia, espaçamento, etc.
 - **Padrões:** Diretrizes de uso, acessibilidade, etc.
- Exemplos: Material Design, Bootstrap, Ant Design, Fast Design, Radix UI, GovBR-DS, etc.
- **Web Components** são uma forma de criar e distribuir componentes em um Design System.

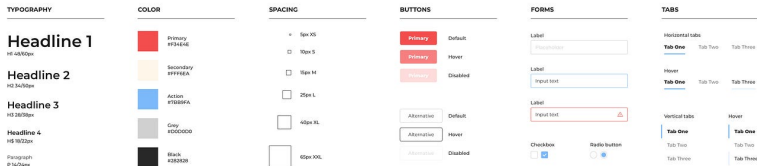


Figura: Design System

Web Components

Conceito

- Web Components são um conjunto de especificações elaboradas para permitir a criação de elementos web de forma customizada e independente.
- Web Components é uma especificação do W3C, que inclui:
 - Custom Elements
 - Shadow DOM
 - HTML Templates
 - HTML Imports
- Frameworks como Angular, React e Vue.js ajudaram a popularizar a ideia de componentização no desenvolvimento web.



Figura: Web Components

- Os Web Components têm um ciclo de vida que é gerenciado pelo navegador.
- O ciclo de vida é composto por callbacks que são chamados em diferentes momentos do ciclo de vida do componente.
- Os callbacks mais comuns são:
 - `connectedCallback`: Chamado quando o componente é conectado ao DOM.
 - `disconnectedCallback`: Chamado quando o componente é desconectado do DOM.
 - `attributeChangedCallback`: Chamado quando um atributo do componente é alterado.
 - `adoptedCallback`: Chamado quando o componente é movido para um novo documento.

- Custom Elements é uma API que permite a criação de novos elementos HTML personalizados, encapsulando HTML, CSS e JavaScript.
- A classe pode ser estendida a partir de `HTMLElement` ou `HTMLButtonElement`, por exemplo.

JS

```
class PopupInfo extends HTMLElement {  
  constructor() {  
    super();  
  }  
  // Element functionality written in here  
}
```

Figura: Custom Elements

Web Components

DOM vs Shadow DOM

- O DOM é uma árvore de elementos HTML/XML que representa a estrutura de uma página web. No contexto de Web Components, light DOM é o DOM padrão da página.
- O Shadow DOM é uma árvore de elementos HTML encapsulada que representa a estrutura de um componente web.

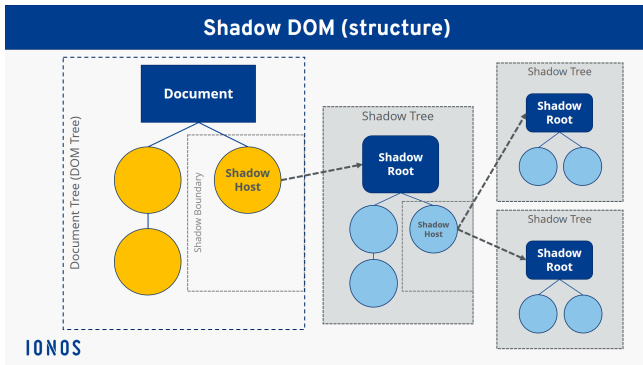


Figura: DOM vs Shadow DOM

- Shadow DOM é uma API que permite a criação de um DOM encapsulado para fornecer um encapsulamento de estilo e comportamento.
- Permite criar árvores de DOM independentes e isoladas que podem ser anexadas a um elemento HTML.
- Impede que estilos e scripts de fora do DOM encapsulado afetem o DOM encapsulado.
- Permite criar e reutilizar componentes.
- Exemplo: `<input type="range">` é um elemento nativo encapsulado.

```
▼<input type="range"> == $0
  ▼#shadow-root (user-agent)
    ▼<div>
      ▼<div pseudo="-webkit-slider-runnable-track" id="track">
        <div id="thumb"></div>
      </div>
    </div>
  </input>
```

Figura: Shadow DOM - Exemplo

- HTML Templates é uma tag que permite declarar fragmentos de código HTML que não são renderizados quando a página é carregada.
- O conteúdo do template pode ser clonado e renderizado posteriormente.

HTML

```
<template id="my-paragraph">  
  <p>My paragraph</p>  
</template>
```

JS

```
let template = document.getElementById("my-paragraph");  
let templateContent = template.content;  
document.body.appendChild(templateContent);
```

Figura: HTML Templates - Exemplo

- Slots são um mecanismo que permite inserir conteúdo dinâmico dentro de um componente encapsulado.
- Se um componente tem um slot, o conteúdo inserido no slot é renderizado no lugar do slot.
- O slot é usado para adicionar texto customizado.

HTML

```
<my-paragraph>  
  <span slot="my-text">Let's have some different text!</span>  
</my-paragraph>
```

Figura: Slots - Exemplo

- HTML Imports era uma especificação da W3C que permitia importar e incluir componentes web em outros documentos HTML.
- Foi descontinuado em favor de módulos ES6 que permitem importar e exportar módulos JavaScript.

Exemplo

```
// Importa o componente - descontinuado  
<link rel="import" href="component.html">  
  
// Importa o módulo ES6  
import {Component} from "component.js"
```

Prós

- **Reusabilidade:** Componentes podem ser reutilizados em diferentes projetos.
- **Encapsulamento:** O Shadow DOM permite encapsular estilos e comportamentos
- **Produtividade:** Facilita a manutenção e evolução de aplicações web.
- **Padronização:** Padrão do W3C, suportado por todos os navegadores modernos.


Contras

- **Compatibilidade:** Não é suportado por navegadores mais antigos.
- **Complexidade:** Requer conhecimento avançado de HTML, CSS e JavaScript.
- **Performance:** Pode impactar a performance da aplicação.
- **Curva de Aprendizado:** Requer tempo para aprender e dominar.
- **Acessibilidade:** Pode impactar leitores de tela e outras tecnologias assistivas.

Para criar um Web Component, é necessário:

- Criar uma classe que estende `HTMLElement`
- Definir o template do componente
- Definir o Shadow DOM
- Registrar o componente
- Usar o componente

Exemplo

 [Gist - Exemplo prático de Web Components](#)

-  **Design System.**
Podcast - Hipsters #170
-  **Web Components.**
Vídeo - Web Components Crash Course
-  **Componente de Ícone em SVG.**
Vídeo - Criando um Componente de Ícone
-  **Web Components - Introduction.**
Site - WebComponents.org
-  **ShadCN - UI Components.**
Biblioteca de Componentes Web
-  **10 Best Web Component Libraries.**
Artigo - Medium
-  **Web Components - GovBR-DS.**
Design System do Governo Brasileiro
-  **Lifecycle.**
Lifecycle Hooks in Web Components

Escolha dois dos seguintes exercícios para entregar até a próxima aula:

- 1 Crie um Web Component que represente um botão personalizado, com diferentes cores e tamanhos.
- 2 Crie um Web Component para alterar o tema da página (claro/escuro).
- 3 Crie um Web Component que represente um formulário de contato com campos de entrada para nome, e-mail, mensagem e um botão de envio.
- 4 Crie um Web Component que represente um card com uma imagem, título e descrição.
- 5 Crie um Web Component que represente um menu de navegação com links para diferentes páginas.
- 6 Crie um Web Component que represente um carrossel de imagens.
- 7 Outro componente de sua escolha.

Entrega

- Crie um repositório no GitHub para entregar os exercícios.
- O componente precisa implementar o ciclo de vida.
- Crie um arquivo `README.md` com instruções sobre como usar os componentes.