UNIVERSIDAD DE LAS FUERZAS ARMADAS "ESPE"

Stephano Santin
Programación Integrativa de Componentes Web
NRC 16496

HHHH111111PAGINA 1 COMPONENTE CRISTIAN del Alcazhar Ponce 1

Connectando al Element del HTML DOM desde JS

HHH22222segunda llamada

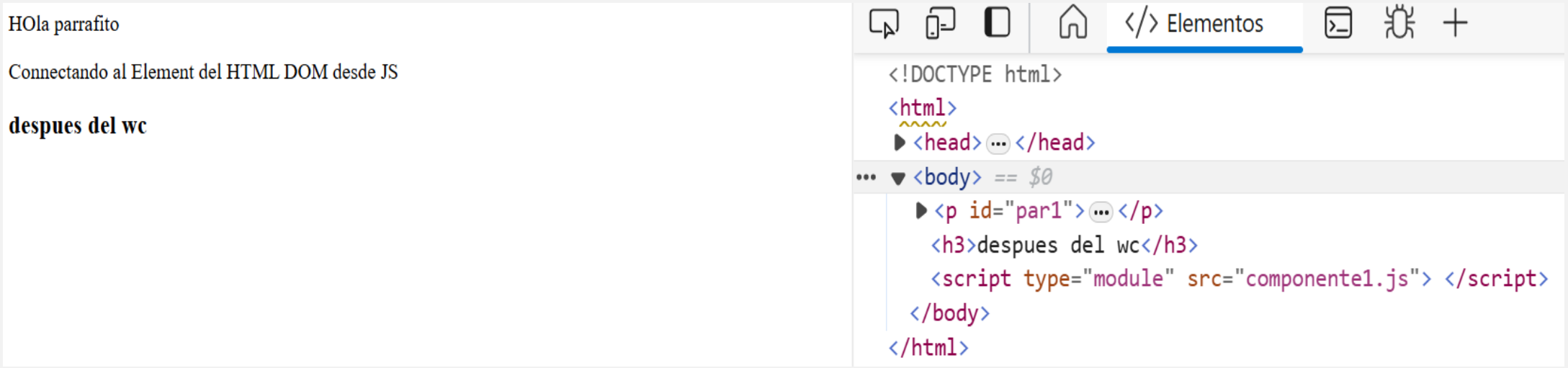
parrafito insertado

```
class Elemento1 extends HTMLElement {
    constructor() {
            super();
            console.log("Elemento1 constructor .... toy constructor  ");
            this.p = document.createElement('p'); // nuevo elemento

        } //---fin del constructor
    connectedCallback() {
            this.p.textContent = "Connectando al Element del HTML DOM  desde JS";
            this.appendChild(this.p);

        } // ---fin metodo connectedCallback

} //------fin de la clase Web Componente

customElements.define('elemento-uno', Elemento1);
```

Como se puede visualizar el HTML recibe los mensajes del Elemento1

El cual posee su constructor y métodos

```
<!DOCTYPE html>
<html>
 <head> ··· </head>
 <body>
   <h1> HHHH111111PAGINA 1 COMPONENTE CRISTIAN del Alcazhar Ponce 1</h1>
 ··· <elemento-uno> ··· </elemento-uno> == $0
   <h2> HHH22222segunda llamada </h2>
   <p> parrafito insertado</p>
   <script type="module" src="componente1.js"> </script>
 </body>
</html>
```




```
class Elemento1 extends HTMLElement {
    constructor() {
            super();
            console.log("Elemento1 constructor .... toy constructor  ");
            this.p = document.createElement('p'); // nuevo elemento

        } //---fin del constructor
    connectedCallback() {
            this.p.textContent = "Connectando al Element del HTML DOM  desde JS";
            this.appendChild(this.p);

        } // ---fin metodo connectedCallback

} //------fin de la clase Web Componente

customElements.define('elemento-uno', Elemento1);
```

PAGINA 2

Al igual que la pagina 1, la pagina 2 obtiene los datos del componente1.js, visualizando tanto la estructura que se tiene
y la funcionalidad ejecutada

INDEX 2

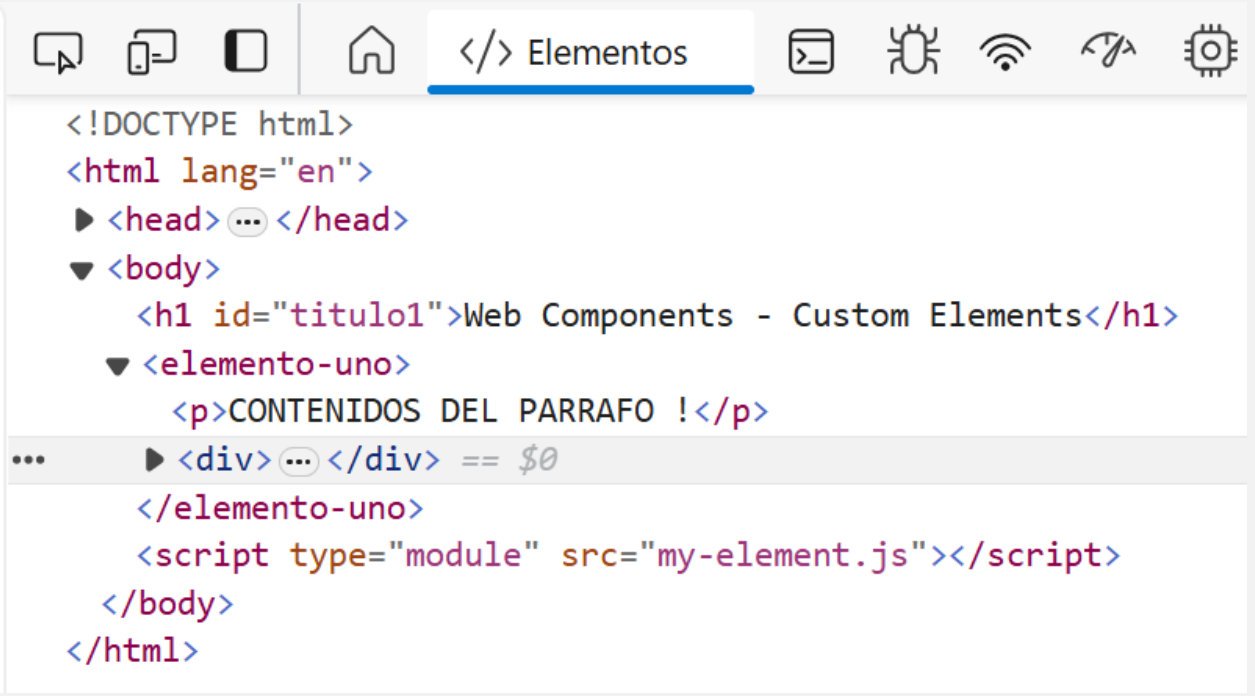
Web Components - Custom Elements

CONTENIDOS DEL PARRAFO !

Parrafo 1: template fuera de la Clase

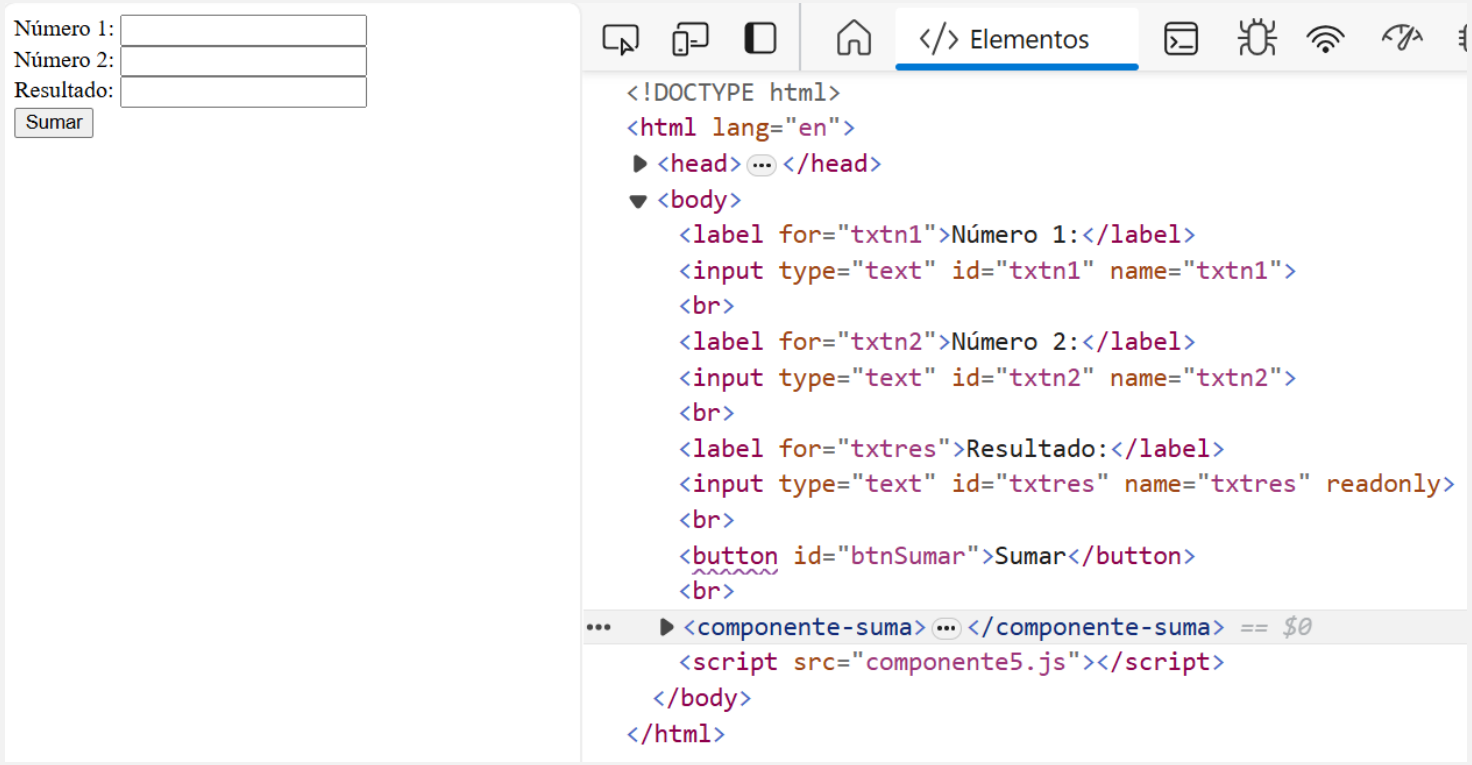
Parrafo 2

```
const template = document.createElement("div");
template.innerHTML = `
  <style>
    .texto {
      color: red;
    }
    p {
      color: blue;
    }
  </style>
  <p class="texto" id="template1">Parrafo 1:  template fuera de la Clase</p>
  <p id="template2">Parrafo 2</p>
`;

class myElement extends HTMLElement {
    constructor() {
        super();
        this.p = document.createElement("p");
    }
    connectedCallback() {
        this.p.textContent = "CONTENIDOS DEL PARRAFO !";
        this.appendChild(this.p);
        this.appendChild(template);
    }
}
customElements.define("elemento-uno", myElement);
```



En este caso se obtiene la información, solo que adicionando un estilo propio al párrafo de salida

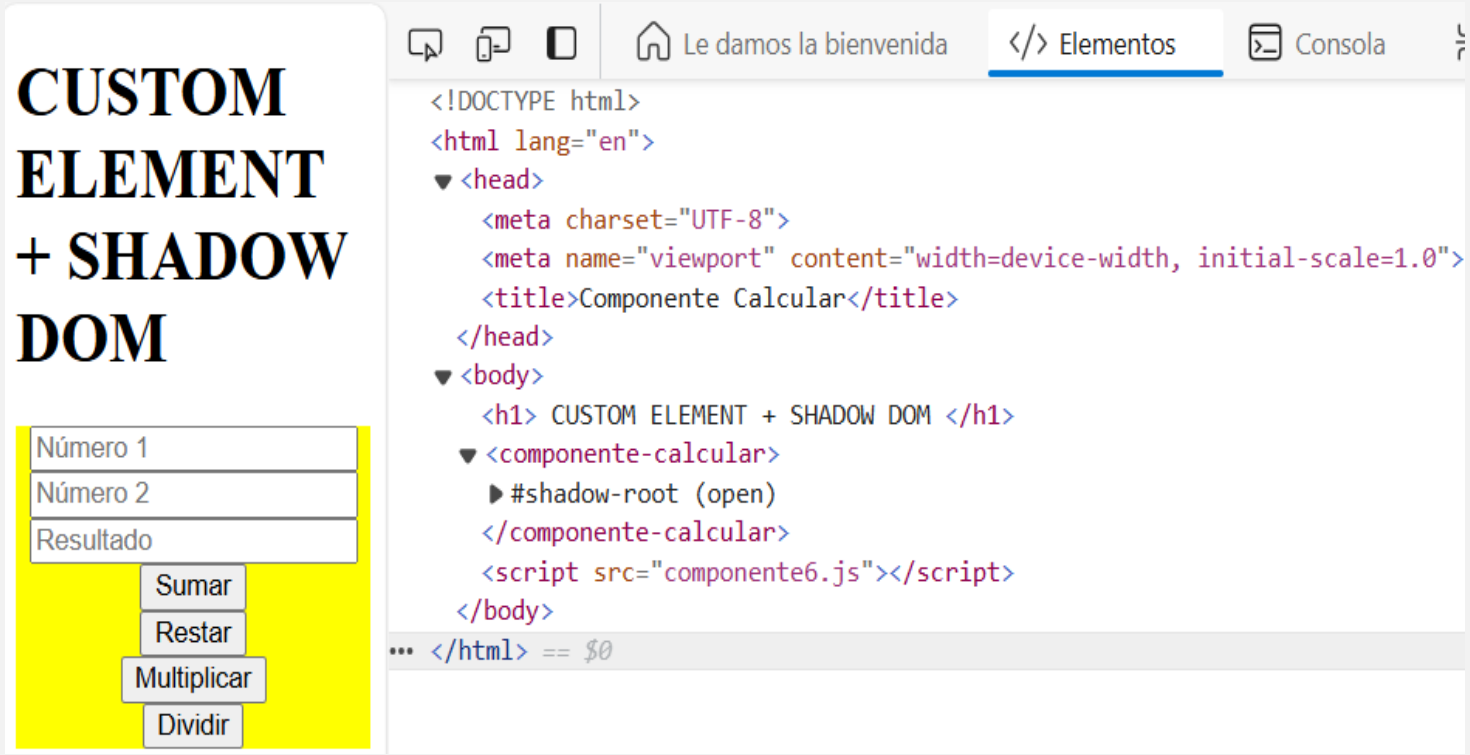
```
class Suma extends HTMLElement {
    constructor() {
        super();
        this.attachShadow({ mode: 'open' });


        // Crear un contenedor para los estilos y la funcionalidad
        const container = document.createElement('div');
        container.innerHTML = `
        <style>
          /* Estilos del componente */
          :host {
            display: block;
            margin-top: 10px;
          }
        </style>
        <slot></slot>
      `;

        this.shadowRoot.appendChild(container);
    }

    connectedCallback() {
        // Obtener referencia al botón de suma y añadir el listener de clic
        const btnSumar = document.querySelector('#btnSumar');
        if (btnSumar) {
            btnSumar.addEventListener('click', () => this.sumar());
        }
    }

    sumar() {
        // Obtener los valores de los inputs
        const n1 = parseFloat(document.querySelector('#txtn1').value);
        const n2 = parseFloat(document.querySelector('#txtn2').value);

        // Comprobar si los valores son números válidos
        if (isNaN(n1) || isNaN(n2)) {
            alert('Por favor, introduce números válidos');
            return;
        }

        // Realizar la suma
        const resultado = n1 + n2;

        // Mostrar el resultado en el input de resultado
        document.querySelector('#txtres').value = resultado;
    }
}
```



PAGINA 5

Esta página HTML presenta un formulario simple para sumar dos números, utilizando etiquetas <label> e <input> para la entrada de datos y un botón para realizar la suma. Incluye también un componente web personalizado <componente-suma>

```
    connectedCallback() {
        // Obtener referencia a los botones y añadir los listeners de clic
        this.shadowRoot.getElementById('btnSumar').addEventListener('click', ()
=> this.calcular('sumar'));
        this.shadowRoot.getElementById('btnRestar').addEventListener('click', ()
=> this.calcular('restar'));

this.shadowRoot.getElementById('btnMultiplicar').addEventListener('click', () =>
this.calcular('multiplicar'));
        this.shadowRoot.getElementById('btnDividir').addEventListener('click', ()
=> this.calcular('dividir'));
    }

    calcular(operacion) {
        // Obtener los valores de los inputs
        const n1 = parseFloat(this.shadowRoot.querySelector('#txtn1').value);
        const n2 = parseFloat(this.shadowRoot.querySelector('#txtn2').value);

        // Comprobar si los valores son números válidos
        if (isNaN(n1) || isNaN(n2)) {
            alert('Por favor, introduce números válidos');
            return;
        }

        // Realizar la operación correspondiente
        let resultado;
        switch (operacion) {
            case 'sumar':
                resultado = n1 + n2;
                break;
            case 'restar':
                resultado = n1 - n2;
                break;
            case 'multiplicar':
                resultado = n1 * n2;
                break;
            case 'dividir':
                resultado = n1 / n2;
                break;
            default:
                return;
        }

        // Mostrar el resultado en el input de resultado
        this.shadowRoot.querySelector('#txtres').value = resultado;
    }
}
```



PAGINA 6

Esta página HTML presenta un formulario simple para sumar dos números, utilizando etiquetas <label> e <input> para la entrada de datos y un botón para realizar la suma. Incluye también un componente web personalizado <componente-suma>
Se toma en cuenta que se agrega Shadow Element para compartir funcionalidades y estilos.

PAGINA 7

Este archivo JavaScript define un componente web personalizado. La clase CustomButon extiende HTMLElement y, al ser instanciada, imprime un mensaje en la consola indicando que el botón ha sido creado.

```
class CustomButon extends HTMLElement {
    constructor() {
        super();
        console.log('boton creado');

    }
}
window.customElements.define('custom-button', CustomButon);
```

```
class MiBoton extends HTMLElement {
    constructor() {
        super();
        // Attach a shadow root to the element.
        this.attachShadow({ mode: 'open' });

        // Create a button element.
        const button = document.createElement('button');
        button.textContent = 'Botón Personalizado';

        // Apply styles to the button.
        const style = document.createElement('style');
        style.textContent = `
            button {
                background-color: yellow;
                border: none;
                color: white;
                padding: 15px 32px;
                text-align: center;
                text-decoration: none;
                display: inline-block;
                font-size: 16px;
                margin: 4px 2px;
                cursor: pointer;
                border-radius: 12px;
            }
        `;

        // Append the style and button to the shadow DOM.
        this.shadowRoot.append(style, button);

        // Add event listener to handle button click.
        button.addEventListener('click', this.handleClick);
    }

    handleClick() {
        alert('¡Botón personalizado clickeado!');
    }
}

// Define el cutom element del web component
customElements.define('mi-boton', MiBoton);
```

Boton

Este archivo JavaScript define un componente web personalizado <mi-boton>. La clase MiBoton extiende HTMLElement y utiliza Shadow DOM para encapsular el botón personalizado. Al ser instanciado, crea un botón con texto "Botón Personalizado", aplica estilos específicos y añade un evento de clic que muestra un mensaje de alerta al hacer clic en el botón.