

TEMA: Números Pares e Impares con Web Components

ESTUDIANTE: Nixon Alejandro Bravo Quijje

FECHA: 18-07-25

1. ENLANCE GITHUB







<https://github.com/AlejandroBravo26/N-meros-Pares-e-Impares-con-Web-Components/tree/main/proyecto-par-impar>

2. CODIGO FUENTE

```
<> index.html > html > body > h1
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Números Pares e Impares</title>
7    <link rel="stylesheet" href="style/main.css">
8    <script type="module" src="components/input-range.js"></script>
9    <script type="module" src="components/par-impar-lista.js"></script>
10 </head>
11 <body>
12   <h1>Números Pares e Impares</h1>
13
14   <div class="container">
15     <input-range></input-range>
16     <par-impar-lista></par-impar-lista>
17   </div>
18 </body>
19 </html>
```

Index.html

```

style > # main.css >  .container
1   body {
2       font-family: Arial, sans-serif;
3       margin: 20px;
4       background-color:  #f4f4f4;
5       color:  #333;
6       line-height: 1.6;
7   }
8
9   h1 {
10      text-align: center;
11      color:  #007bff;
12      margin-bottom: 30px;
13  }
14
15  .container {
16      max-width: 600px;
17      margin: 0 auto;
18      padding: 20px;
19      background-color:  white;
20      border-radius: 10px;
21      box-shadow: 0 4px 8px  rgba(0,0,0,0.15);
22  }

```

main.css

```

<> index.html  JS input-range.js  JS par-impar-lista.js X  # main.css
components > JS par-impar-lista.js > ...
1   class ParImparLista extends HTMLElement {
2       constructor() {
3           super();
4           //Shadow DOM
5           this.attachShadow({ mode: 'open' });
6
7
8           this.shadowRoot.innerHTML = `
9               <style>
10                  /* Estilos encapsulados para este componente */
11                  :host {
12                      display: block;
13                      padding: 20px;
14                      border: 1px solid #ccc;
15                      border-radius: 8px;
16                      background-color: #e6f7ff; /* Un color diferente para la lista */
17                      box-shadow: 0 2px 4px rgba(0,0,0,0.1);
18                  }
19                  ul {
20                      list-style-type: none; /* Quitar viñetas predeterminadas */
21                      padding: 0;
22                      margin: 0;
23                  }
24                  li {
25                      padding: 8px 0;
26                      border-bottom: 1px dashed #eee;
27                      font-size: 1.1em;
28                  }
29                  li:last-child {

```

```

30         border-bottom: none;
31     }
32     .par {
33         color: #28a745; /* Verde para par */
34         font-weight: bold;
35     }
36     .impar {
37         color: #dc3545; /* Rojo para impar */
38         font-weight: bold;
39     }
40     .empty-message {
41         color: #666;
42         font-style: italic;
43     }
44 </style>
45 <div class="list-container">
46     <h2>Resultados:</h2>
47     <ul id="number-list">
48         <li class="empty-message">Ingrese un rango para ver los resultados.</li>
49     </ul>
50 </div>
51 `;
52 }
53
54 connectedCallback() {
55
56     this.numberList = this.shadowRoot.getElementById('number-list');

```

```

57
58
59     document.addEventListener('rango-seleccionado', this.handleRangeSelected.bind(this));
60 }
61
62
63 handleRangeSelected(event) {
64     const { start, end } = event.detail;
65
66     //Limpiar la lista actual
67     this.numberList.innerHTML = '';
68
69
70     if (start > end) {
71         this.numberList.innerHTML = '<li class="empty-message">Rango inválido.</li>';
72         return;
73     }
74
75     for (let i = start; i <= end; i++) {
76         const listItem = document.createElement('li');
77         if (i % 2 === 0) {
78             listItem.textContent = `${i} - Par`;
79             listItem.classList.add('par');
80         } else {
81             listItem.textContent = `${i} - Impar`;
82             listItem.classList.add('impar');
83         }
84         this.numberList.appendChild(listItem);
85     }

```

```

86     }
87
88     disconnectedCallback() {
89         document.removeEventListener('rango-seleccionado', this.handleRangeSelected.bind(this));
90     }
91 }
92
93
94
95 customElements.define('par-impar-lista', ParImparLista);

```

par-impar-lista.js

<> index.html
JS input-range.js X
JS par-impar-lista.js
main.css

components > JS input-range.js > InputRange > constructor

```

1  class InputRange extends HTMLElement {
2      constructor() {
3          super();
4          //Shadow DOM
5          this.attachShadow({ mode: 'open' });
6
7          this.shadowRoot.innerHTML = `
8              <style>
9
10                 :host {
11                     display: block;
12                     margin-bottom: 20px;
13                     padding: 20px;
14                     border: 1px solid #ccc;
15                     border-radius: 8px;
16                     background-color: #f9f9f9;
17                     box-shadow: 0 2px 4px rgba(0,0,0,0.1);
18                 }
19                 .input-group {
20                     margin-bottom: 15px;
21                 }
22                 label {
23                     display: block;
24                     margin-bottom: 5px;
25                     font-weight: bold;
26                     color: #333;
27                 }
28                 input[type="number"] {
29                     width: calc(100% - 22px);

```

```

30         padding: 10px;
31         border: 1px solid #ddd;
32         border-radius: 4px;
33         font-size: 16px;
34     }
35     button {
36         background-color: #007bff;
37         color: white;
38         padding: 10px 20px;
39         border: none;
40         border-radius: 5px;
41         cursor: pointer;
42         font-size: 16px;
43         transition: background-color 0.3s ease;
44     }
45     button:hover {
46         background-color: #0056b3;
47     }
48     .error-message {
49         color: red;
50         margin-top: 10px;
51         font-size: 0.9em;
52     }
53 </style>
54 <div class="input-container">
55     <div class="input-group">
56         <label for="start-number">Inicio del rango:</label>

```

```

58     </div>
59     <div class="input-group">
60         <label for="end-number">Fin del rango:</label>
61         <input type="number" id="end-number" value="10">
62     </div>
63     <button id="submit-button">Mostrar Pares/Impares</button>
64     <div class="error-message" id="error-message"></div>
65 </div>

```

```

66 `;
67
68
69 connectedCallback() {
70
71     const startInput = this.shadowRoot.getElementById('start-number');
72     const endInput = this.shadowRoot.getElementById('end-number');
73     const submitButton = this.shadowRoot.getElementById('submit-button');
74     const errorMessage = this.shadowRoot.getElementById('error-message');
75
76     submitButton.addEventListener('click', () => {
77         const start = parseInt(startInput.value);
78         const end = parseInt(endInput.value);
79
80
81         if (isNaN(start) || isNaN(end)) {
82             errorMessage.textContent = 'Por favor, ingrese números válidos.';
83             return;
84

```

```

86         if (start > end) {
87             errorMessage.textContent = 'El número de inicio debe ser menor o igual al número final.';
88             return;
89         }
90
91
92         errorMessage.textContent = '';
93
94
95         const rangoSeleccionadoEvent = new CustomEvent('rango-seleccionado', {
96             detail: { start, end },
97             bubbles: true,
98             composed: true
99         });
100         this.dispatchEvent(rangoSeleccionadoEvent);
101     });
102 }
103 }
104
105
106 customElements.define('input-range', InputRange);

```

Input-range.js

FUNCIONAMIENTO:



