

Web Components



JORGE ARANEDA - VICENTE CALISTO - TRISTAN HEUER

Average Joe ->



Average Joe ->



*"Quiero ser más
estiloso"*



Average Joe ->



Stylish Joe ->



Stylish Joe ->



Stylish Joe ->



Stylish Joe ->



"Soy más
estiloso"



Stylish Joe ->













etplace

Busca en Marketplace

Sugerencias destacadas

Madrid, Comunidad de Madrid · 10 kilómetros

 45 € Se lo acerco a domicilio Madrid, Comunidad de Madrid · hace ...	 45 € Se lo acerco a domicilio Madrid, Comunidad de Madrid · hace ...	 45 € Se lo acerco a domicilio Madrid, Comunidad de Madrid · hace ...	 35 € Se lo acerco a domicilio Madrid, Comunidad de Madrid · hace ...
			

Chat (4)











Homemade

Cortavientos Discreto

\$59.990



NotCo

Not Mayo

\$3.990



sell-item: template

Native

```
const sellItemTemplate = document.createElement('template');
sellItemTemplate.innerHTML = `
  <style>
    [...]
  </style>
  <div class="main">
    <img class="image" />
    <span class="brand"></span>
    <span class="title"></span>
    <span class="price"></span>
  </div>
`;
```

```
connectedCallback() {
  // Triggered on component insertion into the DOM
  this.attachShadow({ mode: 'open' });
  this.shadowRoot.appendChild(sellItemTemplate.content.cloneNode(true));
  this.update();
}
```

Lit

```
render() {
  return html`
    <style> .main { background-color: ${this.color} </style>
    <div class="main">
      <img class="image" src=${this.image}/>
      <span class="brand">${this.brand}</span>
      <span class="title">${this.title}</span>
      <span class="price">${currencyFormatter(this.price)}</span>
    </div>
  `;
}
```

```
static styles = css`
  [...]
`;
```

sell-item: atributos y actualización

Native

```
update() {  
  const title = this.shadowRoot.querySelector('.title');  
  title.textContent = this.getAttribute('title') || 'No title';  
  
  const price = this.shadowRoot.querySelector('.price');  
  const formattedPrice = currencyFormatter(this.getAttribute('price'));  
  price.textContent = formattedPrice || 'No price';  
  
  const brand = this.shadowRoot.querySelector('.brand');  
  brand.textContent = this.getAttribute('brand') || 'No brand';  
  
  const image = this.shadowRoot.querySelector('.image');  
  image.src = this.getAttribute('image') || '';  
  
  const main = this.shadowRoot.querySelector('.main');  
  main.style.backgroundColor = this.getAttribute('color') || '#fff';  
}
```

Lit

```
static properties = {  
  title: {},  
  price: {},  
  brand: {},  
  image: {},  
  color: {},  
};
```

```
render() {  
  return html`  
    <style> .main { background-color: ${this.color} </style>  
    <div class="main">  
      <img class="image" src=${this.image}/>  
      <span class="brand">${this.brand}</span>  
      <span class="title">${this.title}</span>  
      <span class="price">${currencyFormatter(this.price)}</span>  
    </div>  
  `;  
}
```


LISTA DE COMPRAS

Escriba aquí

AGREGAR ÍTEM

MAYONESA

ELIMINAR

CORTAVIENTOS

ELIMINAR

ENTREGA IIC3585

ELIMINAR

HECKER

ELIMINAR

todo-list: template

Native

```
const todoListTemplate = document.createElement('template');
todoListTemplate.innerHTML = `<html*>`
  <style>
    [...]
  </style>
  <div class="main">
    <p class="title"></p>
    <div class="input-container">
      <input
        class="input-text"
        type="text"
      />
      <button class="button-add">Agregar item</button>
    </div>
    <div class="items">
      <!--
      <div class="item">
        <span>Item title</span>
        <button class="button-remove">Eliminar</button>
      </div>
      -->
    </div>
  </div>
`;
```

Lit

```
render() {
  return html`
    <div class="main">
      <p class="title">${this.title}</p>
      <div class="input-container">
        <input
          class="input-text"
          type="text"
          placeholder=${this.placeholder}
        />
        <button class="button-add" @click=${this.addItem}>Agregar item</button>
      </div>
      <div class="items">
        ${this._list.map((item, index) => html`
          <div class="item">
            <span>${item}</span>
            <button class="button-remove" @click=${() => this.removeItem(index)}>Eliminar</button>
          </div>
        `)}
      </div>
    </div>
  `;
}
```


todo-list: event handlers

Native

```
connectedCallback() {  
  // Triggered on component insertion into the DOM  
  this.attachShadow({ mode: 'open' });  
  this.shadowRoot.appendChild(todoListTemplate.content.cloneNode(true));  
  // Add item list event listener  
  this.shadowRoot  
    .querySelector('.button-add')  
    .addEventListener('click', (e) => {  
      this.handleButtonAdd(e);  
    });  
  this.update();  
}
```

Lit

```
render() {  
  return html`  
    <div class="main">  
      <p class="title">${this.title}</p>  
      <div class="input-container">  
        <input  
          class="input-text"  
          type="text"  
          placeholder=${this.placeholder}  
        />  
        <button class="button-add" @click=${this.addItem}>  
          Agregar item  
        </button>  
      </div>  
      <div class="items">  
        ${this._list.map((item, index) => html`  
          <div class="item">  
            <span>${item}</span>  
            <button class="button-remove" @click=${() => this.removeItem(index)}>  
              Eliminar  
            </button>  
          </div>  
        `)}  
      </div>  
    </div>  
  `;  
}
```

todo-list: atributos y actualización

Native

Lit

```
update() {  
  const title = this.getAttribute('title') || 'No title';  
  this.shadowRoot.querySelector('.title').innerHTML = title;  
  
  const placeholder = this.getAttribute('placeholder') || '';  
  this.shadowRoot.querySelector('input-text').placeholder = placeholder;  
  
  const list = this.getAttribute('list')?.split(',') || [];  
  list.forEach((item) => {  
    this.addItem(item);  
  });  
}
```

```
render() {  
  return html`  
    <div class="main">  
      <p class="title">${this.title}</p>  
      <div class="input-container">  
        <input  
          class="input-text"  
          type="text"  
          placeholder=${this.placeholder}  
        />  
        <button class="button-add" @click=${this.addItem}>  
          Agregar ítem  
        </button>  
      </div>  
      <div class="items">  
        ${this._list.map((item, index) => html`  
          <div class="item">  
            <span>${item}</span>  
            <button class="button-remove" @click=${() => this.removeItem(index)}>  
              Eliminar  
            </button>  
          </div>  
        `)}  
      </div>  
    </div>  
  `;  
}
```

todo-list: addItem

Native

```
addItem(content) {  
  if (content === '') return;  
  // Create new item  
  const newTodoItem = document.createElement('div');  
  newTodoItem.classList.add('item');  
  // Build content  
  const span = document.createElement('span');  
  span.innerHTML = content;  
  const button = document.createElement('button');  
  button.addEventListener('click', (e) => {  
    this.removeItem(e);  
  });  
  button.innerHTML = 'Eliminar';  
  // Append content  
  newTodoItem.appendChild(span);  
  newTodoItem.appendChild(button);  
  this.shadowRoot.querySelector('.items').appendChild(newTodoItem);  
}
```

Lit

```
addItem(event) {  
  const value = this.renderRoot.querySelector('.input-text').value;  
  if (value) {  
    this._list.push(value);  
    this.requestUpdate();  
  }  
}
```

```
`${this._list.map((item, index) => html`  
  <div class="item">  
    <span>${item}</span>  
    <button class="button-remove" @click="${() => this.removeItem(index)}">Eliminar</button>  
  </div>  
`)}  
`}
```

todo-list: removeItem

Native

```
removeItem(e) {  
  e.target.parentNode.remove();  
}
```

Lit

```
removeItem(idx) {  
  this._list.splice(idx, 1);  
  this.requestUpdate();  
}
```

```
${this._list.map((item, index) => html`  
  <div class="item">  
    <span>${item}</span>  
    <button class="button-remove" @click="${() => this.removeItem(index)}">Eliminar</button>  
  </div>  
`)}  
`))}
```


Carga de los componentes

Native

```
window.customElements.define('sell-item', SellItem);  
window.customElements.define('todo-list', TodoList);
```

```
<script src="SellItem.js"></script>  
<script src="TodoList.js"></script>
```

Lit

```
import {LitElement, css, html} from 'lit';
```

```
customElements.define('sell-item', SellItem);  
customElements.define('todo-list', TodoList);
```

```
<script type="module" src="SellItem.js"></script>  
<script type="module" src="TodoList.js"></script>
```

Uso de componente: sell-item

```
<sell-item  
  brand="Marca"  
  title="Nombre del producto"  
  price="10000"  
  image="images/product.png"  
  color="#ff0000"  
></sell-item>
```

Homemade

Cortavientos Discreto

\$59.990



Uso de componente: todo-list

```
<todo-list  
  title="Nombre de la lista"  
  placeholder="Escriba aquí"  
  list="Producto 1, Producto 2, Producto 3"  
></todo-list>
```

LISTA DE COMPRAS

Escriba aquí

AGREGAR ÍTEM

MAYONESA

ELIMINAR

CORTAVIENTOS

ELIMINAR

ENTREGA IIC3585

ELIMINAR

HECKER


ELIMINAR

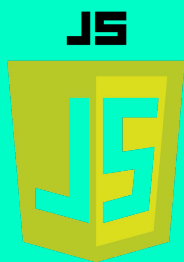
Conclusiones

Lit pros

- Pesa 5 KB, extremadamente liviano 🐣
- Lit = Web components + reactividad + templates declarativas + utilidades para reducir boilerplate 📦

Lit cons

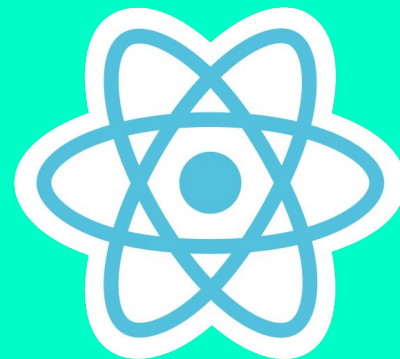
- Dependencia de la librería para el desarrollo 🤔
- Relativamente reciente (~5 años desde su primera release) 



¿Qué usarías tú?



Lit



Muchas gracias

goodbye floppa

