Somos un ecosistema de desarrolladores de software

</Riwi>

Java Script DOM (Document Object Model)

DOM

Insertar elementos en el DOM

Si tenemos que hacer múltiples inserciones o necesitamos insertar elementos manteniendo intacta la estructura existente (o los listeners de eventos asociados), estaría bien disponer de formas en las que podamos indicar exactamente donde queremos añadir el elemento, de forma más controlada. Y eso es lo que vamos a ver en este artículo.

Veamos 3 formas (de más tradicional a más moderna) de inserción o modificación de elementos:

1. La API de nodos
2. La API de elementos
3. La API de inserción adyacente

DOM

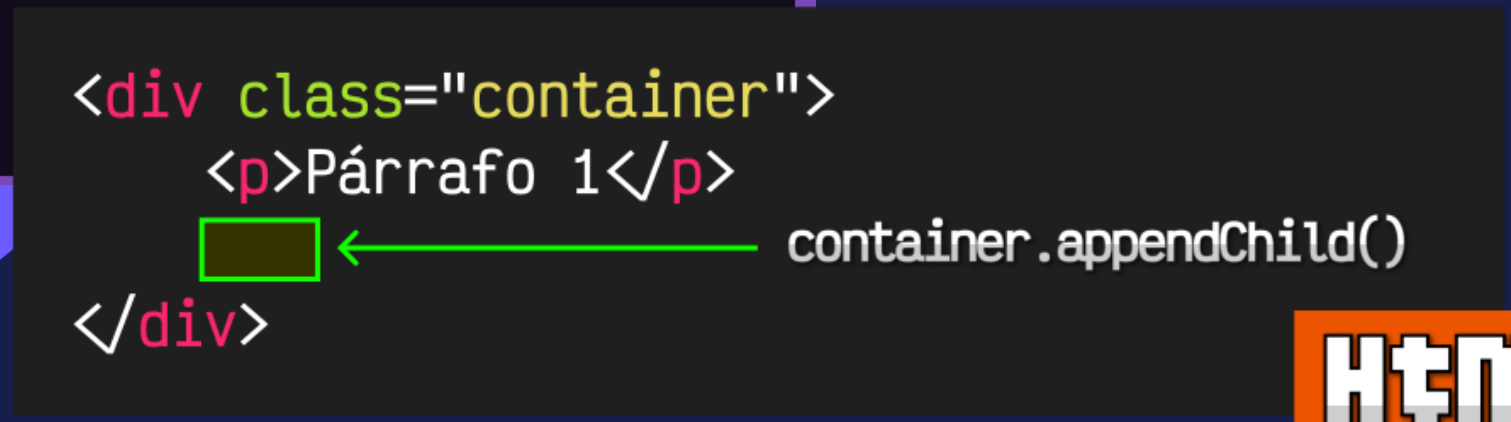
API de nodos

Métodos	Descripción
NODE.appendChild(node)	Añade como hijo el nodo node. Devuelve el nodo insertado.
NODE.removeChild(node)	Elimina y devuelve el nodo hijo node.
NODE.replaceChild(new, old)	Reemplaza el nodo hijo old por new. Devuelve old.
NODE.insertBefore(new, node)	Inserta el nodo new antes de node y como hijo del nodo actual.
NODE.insertBefore(new, NULL)	Inserta el nodo new después del último nodo hijo. Equivale a .appendChild().

DOM

El Metodo .appendChild

```
const container = document.querySelector(".container");

const img = document.createElement("img");
img.src = "https://lenguajejs.com/assets/logo.svg";
img.alt = "Logo Javascript";

container.appendChild(img);
```



DOM

.removeChild( )

```
const container = document.querySelector(".container");
const secondItem = container.querySelector(".item:nth-
child(2)");

container.removeChild(secondItem); // Desconecta el
segundo .item
```

DOM

.replaceChild()


```
const container = document.querySelector(".container");
const secondItem = container.querySelector(".item:nth-
child(2)");

const newNode = document.createElement("div");
newNode.textContent = "DOS";

container.replaceChild(newNode, secondItem);
```

DOM

.insertBefore( )

</Riwi>

► NULL inserta newnode después del último nodo hijo. Equivale a .appendChild().
► ELEMENT inserta newnode antes de dicho node de referencia.

```
const container = document.querySelector(".container");
const secondItem = container.querySelector(".item:nth-
child(2)");

const newNode = document.createElement("div");
newNode.textContent = "DOS";

container.replaceChild(newNode, secondItem);
```

DOM

API de elementos

Métodos	Descripción
.before()	Añade el nuevo elemento justo antes.
.after()	Añade el nuevo elemento justo después.
.prepend()	Se añade el nuevo elemento antes del primer hijo.
.append()	Se añade el nuevo elemento después del último hijo.
.replaceChildren()	Elimina todos los hijos y los sustituye por el nuevo elemento.
.replaceWith()	Se sustituye por el nuevo elemento.
.remove()	Elimina el propio elemento.

DOM

.before( ) - .after( ) y .prepend( ) - .append( )

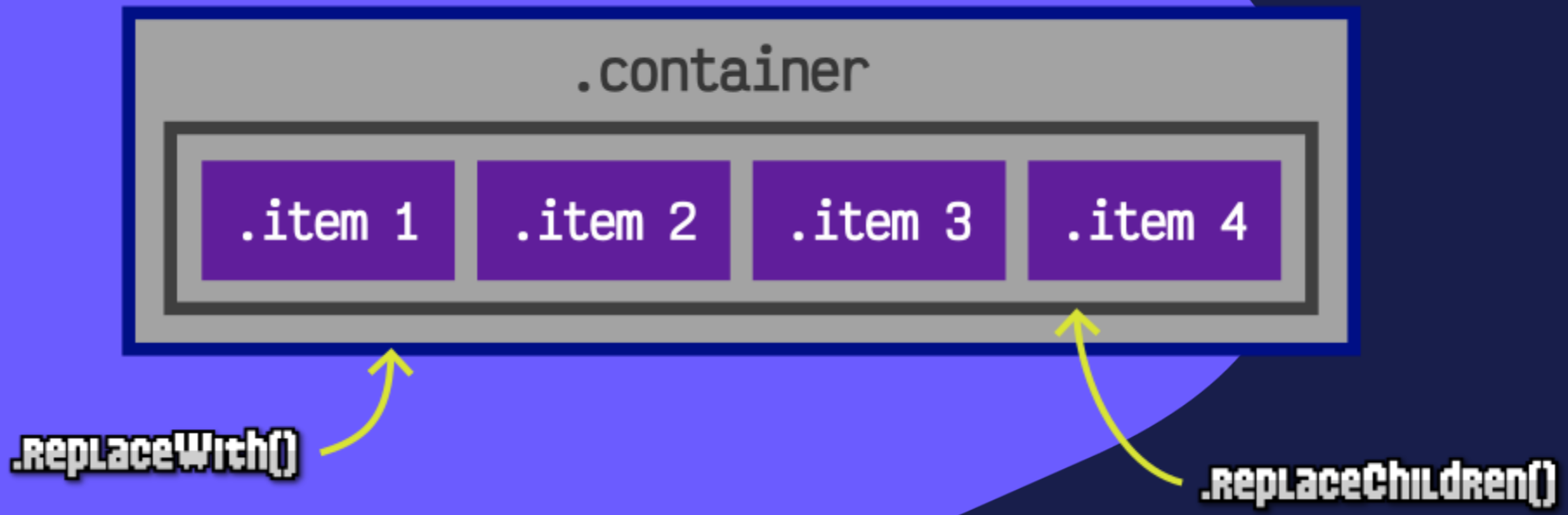
DOM

.before( ) - .after( ) y .prepend( ) - .append( )

```
const element = document.createElement("div");
element.textContent = "Item insertado";

// A) Inserta antes de .container
container.before(element);

// B) Inserta después de .container
container.after(element);
//////————————————//////////

// A) Inserta antes del primer hijo de .container
container.prepend(element);

// B) Inserta después del último hijo de .container
container.append(element);
```

DOM

.replaceChildre( ) - .replaceWith()



DOM

.replaceChildre( ) - .replaceWith()

```
const element = document.createElement("div");
element.textContent = "Nuevo Item hijo";

// A) Reemplaza por todos sus hijos
container.replaceChildren(element);

// B) El container es reemplazado por el nuevo item hijo
container.replaceWith(element);
```

DOM

.remove()


```
const div = document.querySelector(".deleteme");

div.isConnected;     // true
div.remove();
div.isConnected;     // false
```

DOM

API de inserción adyente

Métodos	Descripción
ELEMENT .insertAdjacentElement(position, element)	Inserta el element en la posición position. Si falla, NULL .
.insertAdjacentHTML(position, htmlCode)	Inserta el código HTML de htmlCode en la posición position.
.insertAdjacentText(position, text)	Inserta el texto text en la posición position.

.insertAdjacentElement() donde insertamos una etiqueta HTML ELEMENT
.insertAdjacentHTML() donde insertamos código HTML directamente (similar a innerHTML)
.insertAdjacentText() donde insertamos un texto directamente (similar a textContent)

DOM

API de inserción adyente

DOM

API de inserción adyente

Valor	Descripción	Equivalente a...
beforebegin	Inserta el elemento **antes** de la etiqueta HTML de apertura.	before()
afterbegin	Inserta el elemento dentro, **antes de su primer hijo**.	preprend()
beforeend	Inserta el elemento dentro, **justo antes de la etiqueta HTML de cierre**.	append() o appendChild()
afterend	Inserta el elemento **después** de la etiqueta HTML de cierre.	after()

DOM

API de inserción adyente

```
const container = document.querySelector(".container");

// Creamos un nuevo <div>Ejemplo</div>
const div = document.createElement("div");
div.textContent = "Ejemplo";

container.insertAdjacentElement("beforebegin", div);
// A) <div>Ejemplo</div> <div
class="container">container</div>

container.insertAdjacentElement("afterbegin", div);
// B) <div class="container"> <div>Ejemplo</div>
container</div>

container.insertAdjacentElement("beforeend", div);
// C) <div class="container">container
<div>Ejemplo</div> </div>

container.insertAdjacentElement("afterend", div);
// D) <div class="container">App</div>
<div>Ejemplo</div>
```

DOM

API de inserción adyente (
Otras opciones)

```
container.insertAdjacentElement("beforebegin", div);
// A) <div>Ejemplo</div> <div
class="container">App</div>

container.insertAdjacentHTML("beforebegin", "
<p>Hola</p>");
// B) <p>Hola</p> <div class="container">App</div>

container.insertAdjacentText("beforebegin", "Hola a
todos");
// C) Hola a todos <div class="container">App</div>
```

DOM

Navegar por los elementos de DOM

Propiedades de elementos HTML	Descripción
ARRAY children	Devuelve una lista de elementos HTML hijos.
ELEMENT parentElement	Devuelve el padre del elemento o NULL si no tiene.
ELEMENT firstElementChild	Devuelve el primer elemento hijo.
ELEMENT lastElementChild	Devuelve el último elemento hijo.
ELEMENT previousElementSibling	Devuelve el elemento hermano anterior o NULL si no tiene.
ELEMENT nextElementSibling	Devuelve el elemento hermano siguiente o NULL si no tiene.

DOM

Navegar por los elementos de DOM

Propiedades de elementos HTML	Descripción
ARRAY children	Devuelve una lista de elementos HTML hijos.
ELEMENT parentElement	Devuelve el padre del elemento o NULL si no tiene.
ELEMENT firstElementChild	Devuelve el primer elemento hijo.
ELEMENT lastElementChild	Devuelve el último elemento hijo.
ELEMENT previousElementSibling	Devuelve el elemento hermano anterior o NULL si no tiene.
ELEMENT nextElementSibling	Devuelve el elemento hermano siguiente o NULL si no tiene.

DOM

Navegar por DOM

DOM

Navegar por DOM

```
document.body.children.length;   // 1
document.body.children;          // <div id="app">
document.body.parentElement;     // <html>

const app = document.querySelector("#app");

app.children;                    // [div.header, p, a]
app.firstElementChild;           // <div class="header">
app.lastElementChild;            // <a href="/">

const a = app.querySelector("a");

a.previousElementSibling;        // <p>
a.nextElementSibling;            // null
```

DOM

Navegar a través nodos

Propiedades de nodos HTML	Descripción
ARRAY childNodes	Devuelve una lista de nodos hijos. Incluye nodos de texto y comentarios.
NODE parentNode	Devuelve el nodo padre del nodo o NULL si no tiene.
NODE firstChild	Devuelve el primer nodo hijo.
NODE lastChild	Devuelve el último nodo hijo.
NODE previousSibling	Devuelve el nodo hermano anterior o NULL si no tiene.
NODE nextSibling	Devuelve el nodo hermano siguiente o NULL si no tiene.

DOM

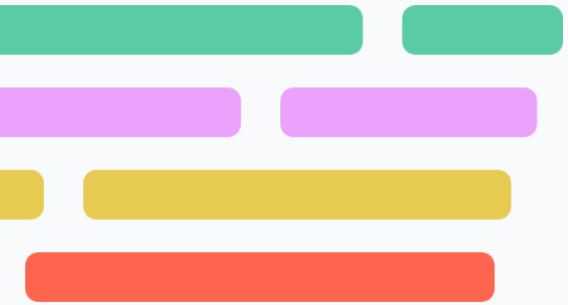
Navegar a través nodos

```
document.body.childNodes.length;   // 3
document.body.childNodes;          // [text, div#app, text]
document.body.parentNode;          // <html>

const app = document.querySelector("#app");

app.childNodes;                    // [text, div.header, text, p, text, a, text]
app.firstChild.textContent;        // "                                        "
app.lastChild.textContent;         // "                                        "
const a = app.querySelector("a");

a.previousSibling;                 // #text
a.nextSibling;                     // #text
```

DOM

Animar elementos de DOM

Método	Descripción
.animate(keyframes,NUMBERduration)	Crea y aplica una animación CSS que dura duration segundos.
.animate(keyframes,OBJECT options)	Crea y aplica una animación CSS, con las opciones indicadas en options.



DOM

.innerText y .outerText

```
const element = document.querySelector(".container");
element.innerText;

// "Hola a todos.
//
// Me llamo Mailet.
//
// Más información"
```

DOM

.innerHTML

```
const element = document.querySelector(".message");

element.innerHTML;     // "Mi nombre es
<strong>Mailet</strong>."
element.textContent;   // "Mi nombre es Mailet."
```

```
element.innerHTML = "<strong>Importante</strong>";
 // Se lee "Importante" (en negrita)
element.textContent = "<strong>Importante</strong>";
 // Se lee "<strong>Importante</strong>"
```

DOM

.outerHTML

```
const data = document.querySelector(".data");
data.innerHTML = "<h1>Tema 1</h1>";

data.textContent;    // "Tema 1"
data.innerHTML;      // "<h1>Tema 1</h1>"
data.outerHTML;      // "<div class="data"><h1>Tema
1</h1></div>"
```

</Be a coder>