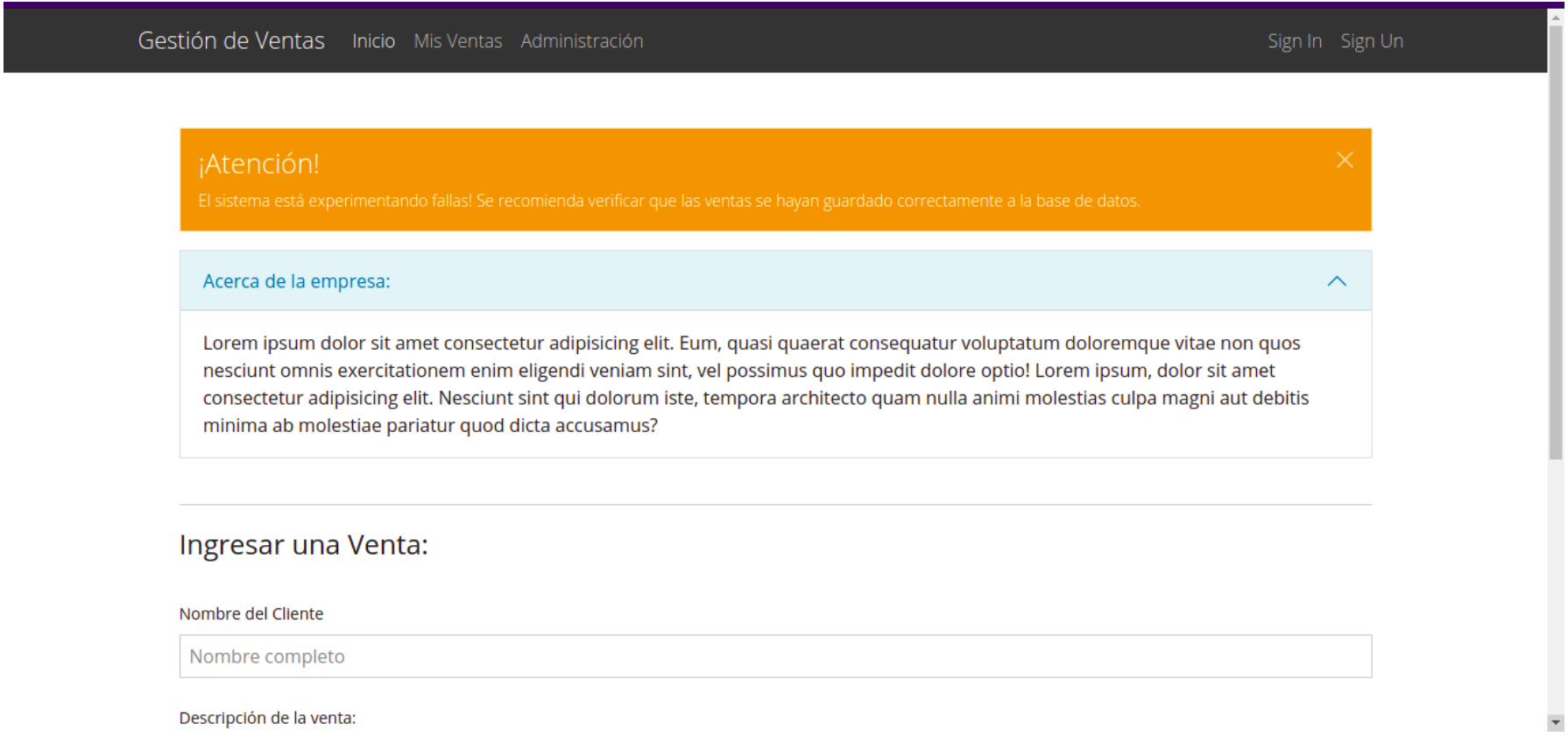
DOM: Agregar, modificar y eliminar elementos desde js

Consigna del día de hoy
Editar esta página desde Javascript



Ejercicio 1:

- Eliminar el cuadro amarillo con la alerta

- Para esto, debemos obtener el elemento padre de la alerta, y eliminar dicha alerta.
- Para identificar dichos elementos, podemos hacerlo usando los métodos vistos la clase pasada:
- document.getElementById(id);
- document.getElementsByTagName(tag);
- document.getElementsByClassName(class);
- document.getElementsByName(name);

Element.children

- La propiedad children nos da una lista (HTMLCollection) con todos los elementos hijos de un elemento.
- Las listas se comportan similar a los Arrays
- Podemos acceder a cada elemento de la lista por su índice
-

```
> let container = document.getElementsByClassName('container')[1]
< undefined
> container.children
<   HTMLCollection(5) [div.alert.alert-dismissible.alert-warning,
    div#accordionExample.accordion, br, hr, form#form_nueva_venta, accordion
    div#accordionExample.accordion, form_nueva_venta: form#form_nueva_venta]
      0: div.alert.alert-dismissible.alert-warning
      1: div#accordionExample.accordion
      2: br
      3: hr
      4: form#form_nueva_venta
      length: 5
      accordionExample: div#accordionExample.accordion
      form_nueva_venta: form#form_nueva_venta
      [[Prototype]]: HTMLCollection
```

Node.removeChild()

- Este método elimina un nodo hijo del DOM.
- Nos retorna el elemento eliminado

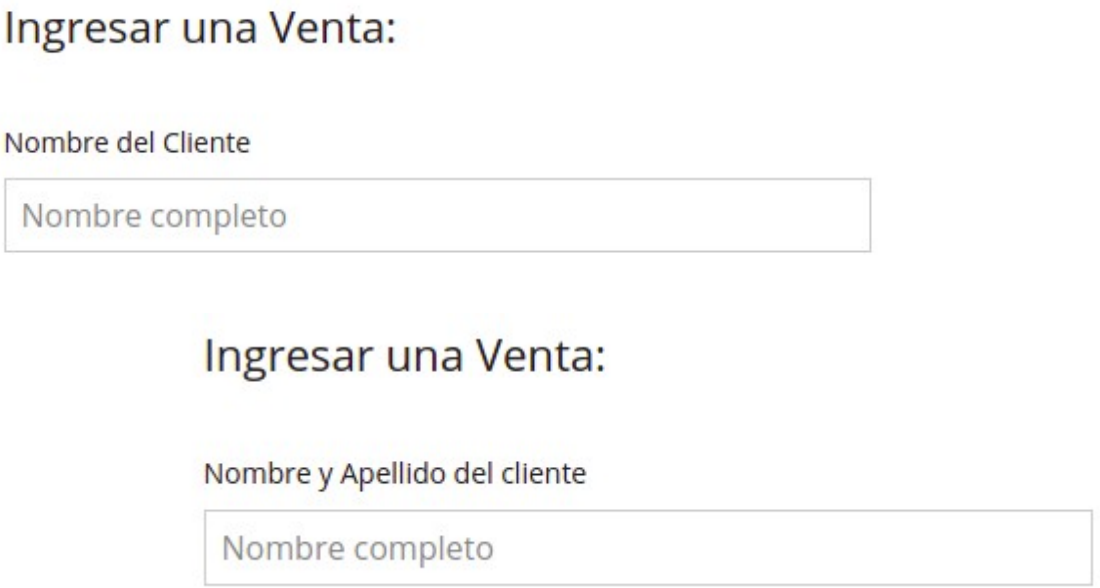
- Ejemplo:

```
> container = document.getElementsByClassName('container')[1]
  ▶<div class="container p-5">…</div>
> alerta = document.getElementsByClassName("alert-warning")[0]
  ▶<div class="alert alert-dismissible alert-warning">…</div>
> container.removeChild(alerta);
  ▶<div class="alert alert-dismissible alert-warning">…</div>
>
```

Resolución del Ej.1

```
/**
 * Ej 1: Eliminar el cuadro amarillo con la alerta
 */
const removeAlert = () => {
    let container = document.getElementsByClassName('container')[1]
    let alertElement = document.getElementsByClassName("alert-warning")[0]
    container.removeChild(alertElement);
}
```

Ejercicio 2:

- Modificar el label "Nombre del Cliente", para que diga "Nombre y Apellido del cliente"
- 

Obteniendo el Label como nodo de otros elementos:

- Podemos llegar al label, obteniendo el fieldset, y de ahí obtener su segundo child.
- Dentro de este child, obtenemos el primer child que es el label

```
▼<form id="form_nueva_venta">
  ▼<fieldset>
      <legend>Ingresar una Venta:</legend>
    ▶<div class="form-group">…</div>
    ▼<div class="form-group">
        <label for="textAreaDescription" class="form-label m
        venta:</label>
        <textarea class="form-control" id="textAreaDescripti
      </div>
    ▶<div class="form-group">…</div>
      <br>
    ▶<div class="d-grid gap-2">…</div> grid
    </fieldset>
  </form>
```

Obtener childrens del form

- Podemos crear una función como esta:

```
const getFormChilds = ()=> {
    return
    document.getElementById('form_nueva_venta').children[0].children
}
```

Obtener el label:

```
> formChilds = getFormChilds()
< ▶ HTMLCollection(6) [legend, div.form-group, div.form-group, div.form-group, br, div.d-grid.gap-2]
> formChilds[1]
< ▶ <div class="form-group">…</div>
> label = formChilds[1].children[0]
<   <label for="inputNameClient" class="form-label mt-4">Nombre del Cliente</label>
>
```

Propiedad textContent

- Con eeta propiedad obtenemos el contenido de texto de un nodo.
- Podemos obtenerlo o modificarlo.

```
> label.textContent
"Nombre del Cliente"
>
```

- Ahora vamos a modificarlo:

```
> label.textContent = "Nombre y Apellido de Cliente:"
"Nombre y Apellido de Cliente:"
>
```

Ejercicio 3:

- Modificarle el tipo al input del Precio, de "text" a "number"
- Para esto, podemos hacer una función como la siguiente:

```
const modifyPriceInput = () => {

    document.getElementById("inputPrice").type = "number";

}
```

Ejercicio 4:

- Agregar un input y un label para indicar el Subtotal de la venta
- Agregarlo justo debajo del input de precio

Creando elementos desde js

- Vamos a crear el siguiente DIV desde Javascript (incluidos todos sus childs, y todas las propiedades)

```
▼<div class="form-group">
   <label class="form-label mt-4">Sub Total:</label>
   <input type="number" id="inputSubTotal" class="form-control">
 </div>
```

- Para crear un elemento, utilizaremos el método:

  document.createElement(nombre)

- document.createElement(“div”);

¿Qué es un div?

- Sirve para crear secciones o agrupar contenidos.
- Es como una caja contenedora de otros elementos
-
  - Elementos..
-
-

Agregándole clases a nuestro elemento

- Usaremos el siguiente método:

- Element.classList.add(clase1, claseN)
- El método add() nos permite agregar una o varias clases.
-
- Con Element.classList obtenemos una lista con todas las clases que ya tiene dicho elemento.

Creando los childs de nuestro div

- Usaremos createElement() para crearlos.
- Luego, definiremos sus propiedades, y les agregamos las clases

```
// create a label:
let labelSubTotal = document.createElement("label");
labelSubTotal.classList.add("form-label", "mt-4");
labelSubTotal.appendChild(document.createTextNode("Sub Total:"));
// create a new input:
let inputSubTotal = document.createElement("input");
inputSubTotal.type = "number";
inputSubTotal.id = "inputSubTotal";
inputSubTotal.classList.add("form-control")
```

createTextNode

- Con este método, podemos crear un nodo de texto
- Nos servirá para ingresarle el texto a la etiqueta de un botón o un label

```
labelSubTotal.appendChild(document.createTextNode("Sub Total:"));
```

- En el código de arriba, le agregamos como child a el label de subtotal, un nodo de texto.
- EL resultado es el siguiente:

```
<label class="form-label mt-4">Sub Total:</label>
```

Agregando childs a un Elemento

- Con este método agregaremos elementos como hijos, a otro elemento.

- En este caso, haremos lo siguiente:

```
// append childs:
divFormGroup.appendChild(labelSubTotal)
divFormGroup.appendChild(inputSubTotal)
```

Agregando un elemento antes que otro

- Ya tenemos nuestro div, ahora debemos agregarlo como child al fieldset de nuestro form

- Usaremos el método insertBefore() para agregarlo justo antes de el div de precio total

```
formFieldset.insertBefore(divFormGroup, formGroupTotalPrice);
```

Ejercicio 5:

- Agregar un botón con texto rojo que diga "Limpiar", debajo del botón de Guardar.
- Obtenemos el form, creamos el button, le agregamos un nodo de texto que diga “Limpiar”, y luego agreamos el botón como child al form.

```
const insertCearButton = () => {
    const form = getForm();
    const clearButton = document.createElement("button");
    clearButton.appendChild(document.createTextNode("LIMPIAR"));
    clearButton.style.color = "red";

    form.appendChild(clearButton);

}
```

Propiedad Style

- Con style podemos modificar o declarar características de css a nuestro Elemento Html.
- Boton.style.color = “red”;

  Es parecido a hacer esto en css:

```
<style>

    #clearButton {
        color: red;
    }

</style>
```

- Con la propiedad color definimos el color del texto de un elemento.

Próxima clase:

- Comenzamos a trabajar con **Eventos**
- Veremos como crear escuchas de eventos con addEventListener()
- Trabajaremos con algunos eventos típicos, como el evento click, el load, y algunos eventos del mouse y teclado.