Universidad Tecnológica Metropolitana

Estructuras de Datos Aplicadas

ISC. Ruth Betsaida Martínez Domínguez, MGTI

Práctica 3-4

Soberanis Acosta Jimena Monserrat

Desarrollo de Software Multiplataforma

Cuarto Cuatrimestre

4°B

Parcial II

Viernes, 13 de octubre de 2023

PRÁCTICA 3

Primero, creé el HTML, al que uso para llamar a sus elementos:

index.html > ...

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/style.css">
    <title>Registro de Alumnos</title>
</head>
<body>
    <div class="container">
        <h1>Registro de Alumnos</h1>
        <div id="app">
            <input type="text" id="nombre" placeholder="Nombre del alumno">
            <input type="number" id="calificacion" placeholder="Calificación">
            <button id="registrar">Registrar</button>
            <h2>Alumnos Aprobados:</h2>
            <ul id="aprobados"></ul>
            <h2>Alumnos Reprobados:</h2>
            <ul id="reprobados"></ul>
        </div>
        <div id="errores">
            <p id="mensaje"></p>
        </div>
    </div>
    <script type="module" src="js/App.js"></script>
</body>
</html>
```

Para la parte lógica, comencé con crear un archivo con una clase Default llamada Alumno, que contiene la información de este (nombre y calificación), también está un método que devuelve que la calificación para que sea aprobatoria debe ser mayor o igual a 7.

js > JS Alumno.js > ...

```javascript
export default class Alumno {
    constructor(nombre, calificacion) {
        this.nombre = nombre;
        this.calificacion = calificacion;
    }

    aprobo() {
        return this.calificacion >= 7;
    }
}
```

```
js > JS App.js > btnRegistrar.addEventListener('click') callback > li
1   import Alumno from './Alumno.js';
2
3   const btnRegistrar = document.getElementById('registrar');
4   const inputNombre = document.getElementById('nombre');
5   const inputCalificacion = document.getElementById('calificacion');
6   const ulAprobados = document.getElementById('aprobados');
7   const ulReprobados = document.getElementById('reprobados');
8
9   const alumnos = [];
10
```

Luego, creé otro archivo que importa Alumno, este archivo es el que se dedica a interactuar con el HTML.

Creé varías constantes para poder llamar a ciertas partes del HTML, el botón, nombre, calificación y las listas de aprobado y reprobados, luego creé la lista vacía alumnos.

```
11  btnRegistrar.addEventListener('click', () => {
12      const nombre = inputNombre.value;
13      const calificacion = parseFloat(inputCalificacion.value);
14
15      if (!isNaN(calificacion) && calificacion >= 0 && calificacion <= 10) {
16          const alumno = new Alumno(nombre, calificacion);
17          alumnos.push(alumno);
18
19          if (alumno.aprobo()) {
20              const li = document.createElement('li');
21              li.textContent = `${nombre} - Calificación: ${calificacion}`;
22              ulAprobados.appendChild(li);
23          } else {
24              const li = document.createElement('li');
25              li.textContent = `${nombre} - Calificación: ${calificacion}`;
26              ulReprobados.appendChild(li);
27          }
28
29          inputNombre.value = '';
30          inputCalificacion.value = '';
31      } else {
32          const errorMensaje = document.getElementById('mensaje');
33          errorMensaje.textContent = 'La calificación no es válida. Debe ser un número entre 0 y 10.';
34          const erroresDiv = document.getElementById('errores');
35          erroresDiv.appendChild(errorMensaje);
36      }
37  });
```

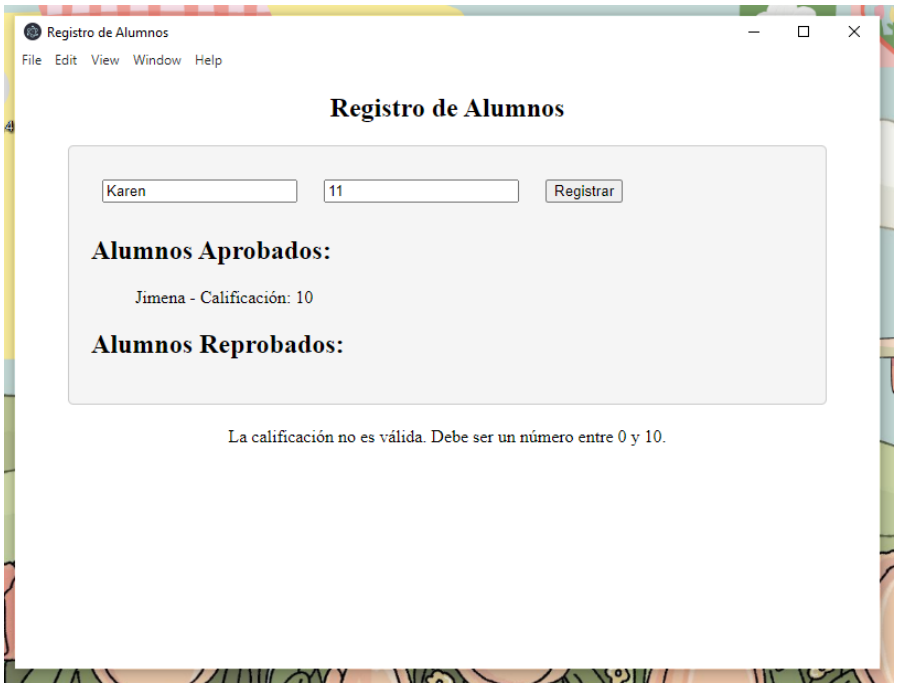
Por último, para poder pasar al alumno a su respectiva lista, tomé la constante del botón que registra al alumno y le agregué un evento al hacerle click. Aquí se hacen varias validaciones, primero se revisa si la calificación es un numero positivo y si es mayor que 0 y menor que 10, pues este es la base de las calificaciones, en caso contrario se manda un mensaje a través del elemento llamado ‘mensaje’, y dice que el número debe ser válido entre 0 y 10

Si se cumple, se crea un nuevo alumno (pues tiene una calificación válida) y pasa a la siguiente validación en donde se toma el método de aprobó y si es mayor o igual 7 se crea un nuevo

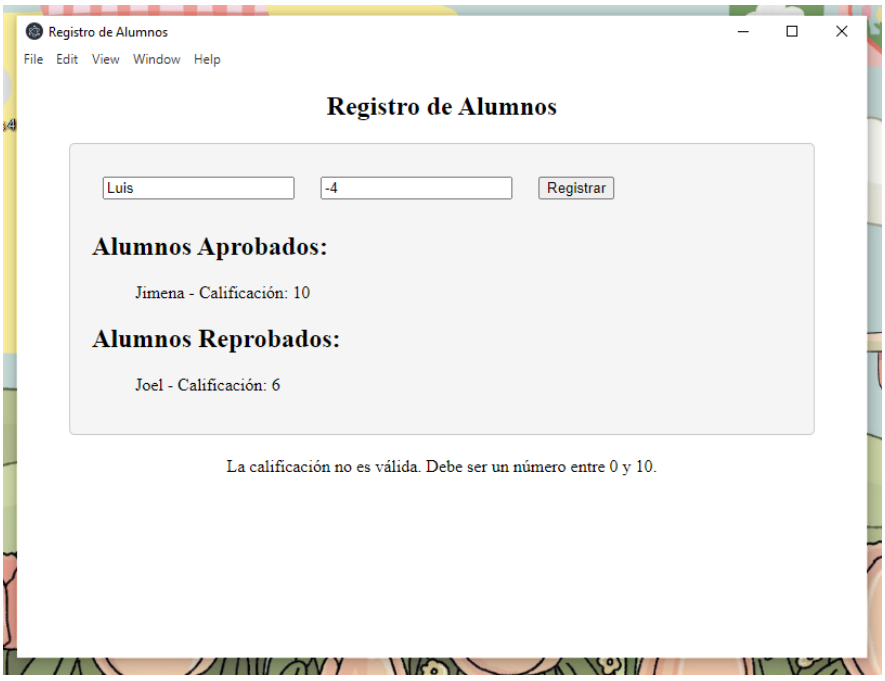
elemento en la lista de aprobado y se almacena ahí, en caso contrario es donde se pasa a la lista de reprobados, en donde también se crea un nuevo elemento de la lista y se almacena ahí.

Le agregué su respectivo CSS (que se encuentra en el GitHub, y para lanzarlo a aplicación usé Electron:

Agregué una calificación previa con 10 y luego una con 11, en donde sale el mensaje.



Y luego, agregué una calificación de 6, que es reprobatoria y una calificación negativa, en la que sale el mensaje.



PRÁCTICA 4

Comencé con esta práctica con el HTML, el cual iré llamando en la parte lógica:

index.html > html > body > div.container > div#app > ul#productos

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/styles.css">
    <title>Lista de Compras</title>
</head>
<body>
    <div class="container">
        <h1>Lista de Compras</h1>
        <div id="app">
            <input type="text" id="nombre" placeholder="Nombre del producto">
            <input type="number" id="precio" placeholder="Precio">
            <button id="agregar">Agregar Producto</button>
            <input type="text" id="claveEliminar" placeholder="Clave del producto a eliminar">
            <button id="eliminar">Eliminar Producto</button>
            <h2>Productos:</h2>
            <ul id="productos"></ul>
            <p>Total de Compra: <span id="totalCompra">0.00</span></p>
        </div>
    </div>
    <script type="module" src="js/App.js"></script>
</body>
</html>
```

js > JS Producto.js > ...

```javascript
export default class Producto {
    static claveCounter = 1;

    constructor(nombre, precio) {
        this.nombre = nombre;
        this.precio = precio;
        this.clave = Producto.claveCounter.toString().padStart(3, '0');
        Producto.claveCounter++;
    }
}
```

Proseguí con crear la clase Producto, que comienza con una variante estática que comienza en 1, en el constructor paso el nombre y precio, pero la clave es la que toma a la variable estática y la convierte en una cadena de texto que al usar padStart ajusta la longitud de la cadena, en donde 3 es la longitud y el 0 es lo que se agregará al principio de cada clave.

```
import Producto from "./Producto.js";

const btnAgregar = document.getElementById('agregar');
const btnEliminar = document.getElementById('eliminar');
const inputNombre = document.getElementById('nombre');
const inputPrecio = document.getElementById('precio');
const inputClaveEliminar = document.getElementById('claveEliminar');
const ulProductos = document.getElementById('productos');
const spanTotalCompra = document.getElementById('totalCompra');

const productos = [];
```

Después, en otro archivo js llamado App, importé a Producto para usar su contenido.

Comencé con este archivo llamando a ciertas partes del HTML como a los botones y la lista ‘ul’ de producto.

Por último, en esta parte creé una lista vacía de productos.

```
btnAgregar.addEventListener('click', () => {
    const nombre = inputNombre.value;
    const precio = parseFloat(inputPrecio.value);

    if (nombre && !isNaN(precio)) {
        const producto = new Producto(nombre, precio);
        productos.push(producto);

        actualizarListaProductos();
        calcularTotalCompra();

        inputNombre.value = '';
        inputPrecio.value = '';
    }
});
```

Ahora, agregué el evento cuando se hace click al botón de agregar, el cual toma los valores de nombre y precio (lo convierte a float), y si el nombre y precio es correcto, se crea un nuevo producto y se agrega a productos, de igual manera se actualiza la las funciones de actualizarListaProductos y calcularTotalCompra, que explicaré más a delante.

```
btnEliminar.addEventListener('click', () => {
    const claveEliminar = inputClaveEliminar.value;
    if (claveEliminar) {
        const index = productos.findIndex(producto => producto.clave === claveEliminar);
        if (index !== -1) {
            productos.splice(index, 1);
            actualizarListaProductos();
            calcularTotalCompra();
        }
        inputClaveEliminar.value = '';
    }
});
```

En el caso del botón de eliminar, se toma la clave el valor del clave del producto, el cual hace un recorrido para buscar esa misma clave, si se encuentra esa misma clave, se elimina de la lista de productos, y además de actualizan la lista y el total de compra con las funciones mencionadas anteriormente.

```
function actualizarListaProductos() {
    ulProductos.innerHTML = '';
    productos.sort((a, b) => a.nombre.localeCompare(b.nombre));
    productos.forEach(producto => {
        ulProductos.innerHTML += `<li>${producto.clave} - ${producto.nombre} - $${producto.precio.toFixed(2)}</li>`;
    });
}

function calcularTotalCompra() {
    const total = productos.reduce((sum, producto) => sum + producto.precio, 0);
    spanTotalCompra.textContent = total.toFixed(2);
}
```

Ahora es cuando, están las dos funciones, la primera, lo que primeramente hace es ordenar los productos en orden alfabético. Después, se van creando en la variable ‘ulProductos’ elementos de lista con un formato en específico, esta función es llamada constantemente pues trabaja con las nuevas listas de productos, ya se si se agrega o elimina alguno.

Después está el calcularCompra, que toma los productos y los suma, y los devuelve con 2 decimales.

Al final, le agregué estilo con CSS (que se encuentra en el repositorio de GitHub, y para lanzarla a una aplicación usé Electron:


Lista de Compras
File Edit View Window Help
Lista de Compras
Nombre del producto
Precio
Agregar Producto
Clave del producto a elimina
Eliminar Producto
Productos:
002 - Playera - $165.00
001 - Taza - $65.00
Total de Compra: 230.00


Lista de Compras
File Edit View Window Help
Lista de Compras
Nombre del producto
Precio
Agregar Producto
Clave del producto a elimina
Eliminar Producto
Productos:
002 - Playera - $165.00
Total de Compra: 165.00

Las carpetas en ambas prácticas (3 y 4) quedaron de esta manera: