APUNTES DEL CURSO DE JavaScript

Proyecto del carrito de compras

Derimán Tejera Fumero

08/08/2024

Índice

Paso 1. Declarar todas las “variables” necesarias

Atendiendo al código html de la web de index.html, localizar los elementos que utilizaremos para realizar el carrito de compra:

```
// Variables
const carrito = document.querySelector('#carrito');
const contenedorCarrito = document.querySelector('#lista-carrito tbody');
const vaciarCarritoBtn = document.querySelector('#vaciar-carrito');
const listaCursos = document.querySelector('#lista-cursos');
```

Paso 2. Cargar todos los addEventListeners necesarios

Se crea una función que es un addEventListener de tipo click sobre el botón “Agregar Al Carrito” situado debajo de cada Tarjeta de curso ofertado, se accede desde la clase agregar-carrito.

```
cargarEventListeners();
function cargarEventListeners() {
  // Cuando agregar un curso presionando en el botón: "Agregar al Carrito"
  listaCursos.addEventListener("click", agregarCurso);
}
```

```
// Funciones
function agregarCurso(e) {
  // Este preventDefault es añadido para evitar que la página se vaya hacia
  // arriba cada vez que se presione el botón "Agregar al carrito" ya que
  // tiene href="#" y # por defecto te lleva al incio de la página.
  e.preventDefault();

  if (e.target.classList.contains("agregar-carrito")) {
    const cursoSeleccionado = e.target.parentElement.parentElement;
    leerDatosCurso(cursoSeleccionado);
  }
}
```

Paso 3. Leer los cursos a los que el usuario ha hecho click

Crearemos un objeto donde se añadirán los diferentes datos del curso en el que el usuario ha hecho click previamente. Los datos se almacenarán en un array de objetos con el nombre:

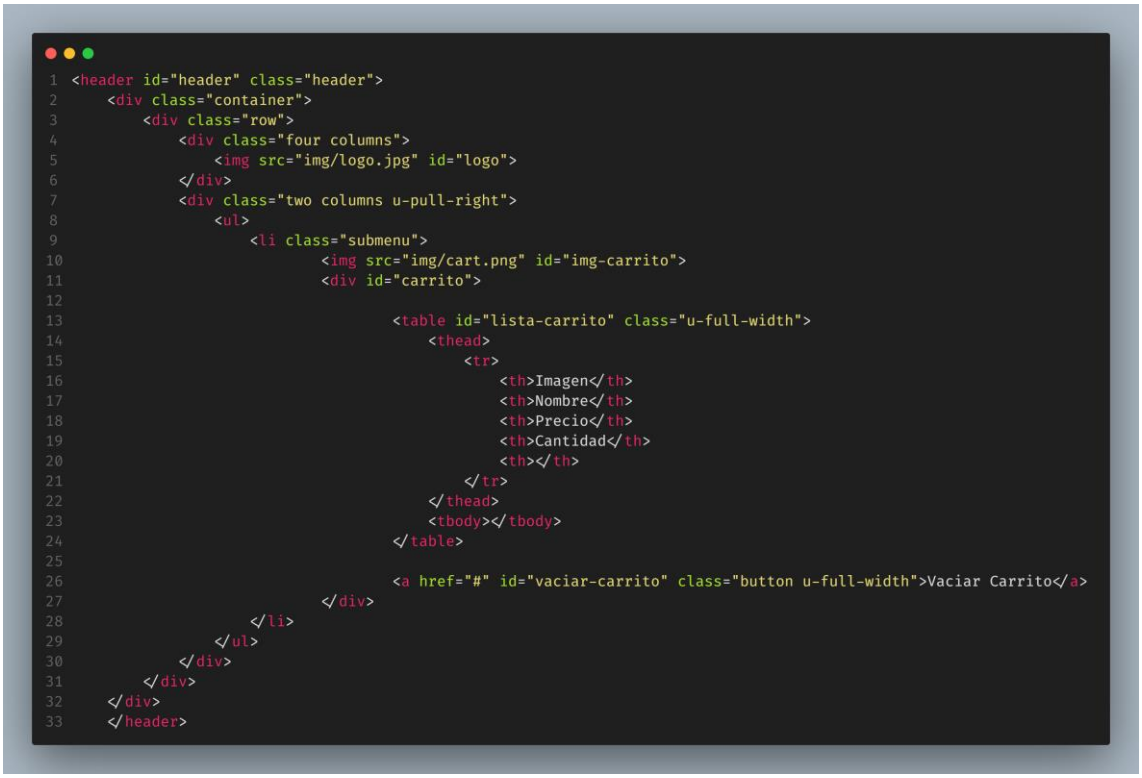
```
let articulosCarrito = [];
```

Estos datos también se mostrarán por consola.

```
// Lee el contenido del HTML al que le dimos click y extrae la información del curso
function leerDatosCurso(curso) {
  console.log(curso);

  // Crear un objeto con el contenido del curso actual
  const infoCurso = {
    imagen: curso.querySelector("img").src,
    titulo: curso.querySelector("h4").textContent,
    precio: curso.querySelector(".precio span").textContent,
    id: curso.querySelector("a").getAttribute("data-id"),
    cantidad: 1,
  };

  // Añadie elementos al array articulosCarrito (también se puede hacer con push)
  // en este caso se utilizarán los spread operators, así se logrará que sea
  // acumulativo y que los cursos que se vayan seleccionando se vayan agregando
  // al carrito.
  articulosCarrito = [ ...articulosCarrito, infoCurso];
  console.log(articulosCarrito);

  carritoHTML();
}
```

Paso 4. Mostrar los cursos en el carrito de la web

Se incrustará el código a modo de fila en el HTML, mas concretamente entre <tbody></tbody>:



```html
<header id="header" class="header">
    <div class="container">
        <div class="row">
            <div class="four columns">
                <img src="img/logo.jpg" id="logo">
            </div>
            <div class="two columns u-pull-right">
                <ul>
                    <li class="submenu">
                            <img src="img/cart.png" id="img-carrito">
                            <div id="carrito">

                                    <table id="lista-carrito" class="u-full-width">
                                        <thead>
                                            <tr>
                                                <th>Imagen</th>
                                                <th>Nombre</th>
                                                <th>Precio</th>
                                                <th>Cantidad</th>
                                                <th></th>
                                            </tr>
                                        </thead>
                                        <tbody></tbody>
                                    </table>

                                    <a href="#" id="vaciar-carrito" class="button u-full-width">Vaciar Carrito</a>
                            </div>
                    </li>
                </ul>
            </div>
        </div>
    </div>
    </header>
```

```javascript
// Muestra el Carrito de compras en el HTML
function carritoHTML() {
  // Limpiar el HTML. Esto es necesario para evitar duplicados de cursos en el carrito
  limpiarHTML();

  // Recorre el carrito y genera el HTML
  articulosCarrito.forEach((curso) => {
    const row = document.createElement("tr");
    row.innerHTML = `
    <td>
${curso.titulo}
    </td>
    `;

    // Añade el HTML del carrito en el tbody
    contenedorCarrito.appendChild(row);
  });
}
```

Paso 5. Eliminar los cursos del carrito antes de empezar a añadirlos al hacer click en los botones “Agregar al carrito”

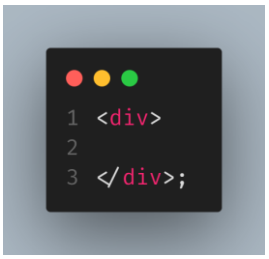
Crearemos la función de eliminación de elementos del carrito, esta función es necesaria porque al hacer click en un curso, por alguna razón VER POR QUÉ PASA ESTO, se duplican los

cursos añadidos, por lo que esta función deberá ser llamada antes de mostrar los artículos añadidos al carro en el HTML.

```
// Elimina los cursos del tbody
function limpiarHTML() {
  // Forma lenta de hacer el borrado:
  //contenedorCarrito.innerHTML = "";

  // Forma rápida de hacer el borrado:
  while (contenedorCarrito.firstChild) {
    contenedorCarrito.removeChild(contenedorCarrito.firstChild);
  }
}
```

Al usar un bucle, lo que se logra teóricamente es una mejor performance, mientras exista un elemento en el carrito, el bucle procederá a eliminar el primer hijo.

Ejemplo visual de como ocurre esto en cada iteración del bucle hasta que se para:



Paso 6. Añadir el resto de elementos necesarios para mostrar en el carrito

```
// Muestra el Carrito de compras en el HTML
function carritoHTML() {
  // Limpiar el HTML. Esto es necesario para evitar duplicados de cursos en el carrito
  limpiarHTML();

  // Recorre el carrito y genera el HTML
  articulosCarrito.forEach((curso) => {
    const row = document.createElement("tr");
    row.innerHTML = `
    <td>
    <img src="${curso.imagen}" width="100">
    </td>
    <td>${curso.titulo}</td>
    <td>${curso.precio}</td>
    <td>${curso.cantidad}</td>
    <td>
    <a href="#" class="borrar-curso" data-id="${curso.id}"> X </a>
    </td>
    `;

    // Añade el HTML del carrito en el tbody
    contenedorCarrito.appendChild(row);
  });
}
```

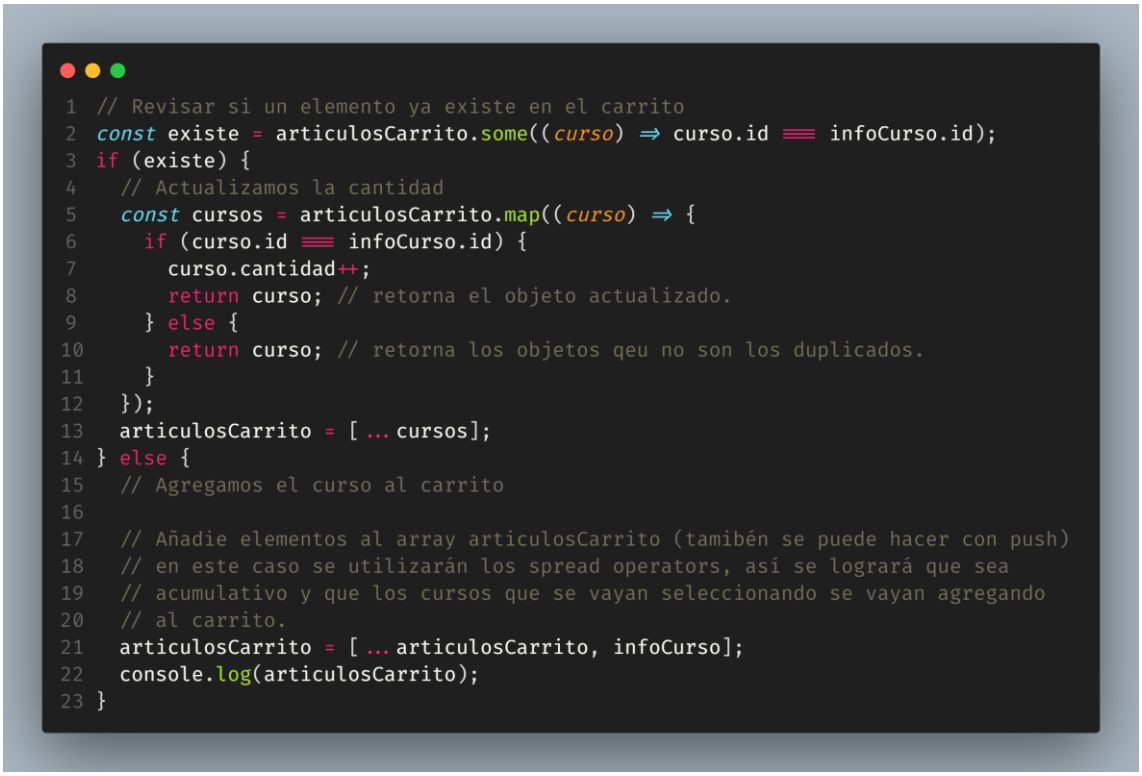
Ahora lo hacemos mediante desestructuring, para simplificar la lectua del código:

```
// Muestra el Carrito de compras en el HTML
function carritoHTML() {
  // Limpiar el HTML. Esto es necesario para evitar duplicados de cursos en el carrito
  limpiarHTML();

  // Recorre el carrito y genera el HTML
  articulosCarrito.forEach((curso) => {
    const { imagen, titulo, precio, cantidad, id } = curso;
    const row = document.createElement("tr");
    row.innerHTML = `
    <td>
    <img src="${imagen}" width="100">
    </td>
    <td>${titulo}</td>
    <td>${precio}</td>
    <td>${cantidad}</td>
    <td>
    <a href="#" class="borrar-curso" data-id="${id}"> X </a>
    </td>
    `;

    // Añade el HTML del carrito en el tbody
    contenedorCarrito.appendChild(row);
  });
}
```

Paso 7. Como detectar si ya existe un curso comprado y aumentar la cantidad si el usuario le sigue dando a comprar a los mismos cursos

```
// Lee el contenido del HTML al que le dimos click y extrae la información del curso
function leerDatosCurso(curso) {
  console.log(curso);

  // Crear un objeto con el contenido del curso actual
  const infoCurso = {
    imagen: curso.querySelector("img").src,
    titulo: curso.querySelector("h4").textContent,
    precio: curso.querySelector(".precio span").textContent,
    id: curso.querySelector("a").getAttribute("data-id"),
    cantidad: 1,
  };

  // Revisar si un elemento ya existe en el carrito
  const existe = articulosCarrito.some((curso) => curso.id === infoCurso.id);
  if (existe) {
    // Actualizamos la cantidad
    const cursos = articulosCarrito.map((curso) => {
      if (curso.id === infoCurso.id) {
        curso.cantidad++;
        return curso; // retorna el objeto actualizado.
      } else {
        return curso; // retorna los objetos qeu no son los duplicados.
      }
    });
    articulosCarrito = [...cursos];
  } else {
    // Agregamos el curso al carrito

    // Añadie elementos al array articulosCarrito (también se puede hacer con push)
    // en este caso se utilizarán los spread operators, así se logrará que sea
    // acumulativo y que los cursos que se vayan seleccionando se vayan agregando
    // al carrito.
    articulosCarrito = [...articulosCarrito, infoCurso];
    console.log(articulosCarrito);
  }

  carritoHTML();
}
```

Mas concretamente ocurre en esta parte del código:



```
// Revisar si un elemento ya existe en el carrito
const existe = articulosCarrito.some((curso) => curso.id === infoCurso.id);
if (existe) {
  // Actualizamos la cantidad
  const cursos = articulosCarrito.map((curso) => {
    if (curso.id === infoCurso.id) {
      curso.cantidad++;
      return curso; // retorna el objeto actualizado.
    } else {
      return curso; // retorna los objetos qeu no son los duplicados.
    }
  });
  articulosCarrito = [...cursos];
} else {
  // Agregamos el curso al carrito

  // Añadie elementos al array articulosCarrito (también se puede hacer con push)
  // en este caso se utilizarán los spread operators, así se logrará que sea
  // acumulativo y que los cursos que se vayan seleccionando se vayan agregando
  // al carrito.
  articulosCarrito = [...articulosCarrito, infoCurso];
  console.log(articulosCarrito);
}
```

Paso 7. Eliminar un curso del carrito

```
// Elimina un curso del carrito
function eliminarCurso(e) {
  if (e.target.classList.contains("borrar-curso")) {
    const cursoId = e.target.getAttribute("data-id");

    // Elimina del arreglo de articulosCarrito utilizando su data-id
    articulosCarrito = articulosCarrito.filter((curso) => curso.id !== cursoId);

    // Es necesario volver a llamar a la función que imprime los valores en el HTML para que muestre los actualizados
    carritoHTML();
  }
}
```

Paso 8. Limpiar todos los cursos del carrito

Para esto y al ser muy poco código, no se creará una función específica para esto, sino desde la propia escucha:

```
// Variables
const carrito = document.querySelector("#carrito");
const contenedorCarrito = document.querySelector("#lista-carrito tbody");
const vaciarCarritoBtn = document.querySelector("#vaciar-carrito");
const listaCursos = document.querySelector("#lista-cursos");
// Carrito de compras que será un array inicialmente vacío
let articulosCarrito = [];

cargarEventListeners();
function cargarEventListeners() {
  // Cuando agregar un curso presionando en el botón: "Agregar al Carrito"
  listaCursos.addEventListener("click", agregarCurso);

  // Elimina cursos del carrito
  carrito.addEventListener("click", eliminarCurso);

  // Botón de eliminar todo el contenido del carrito
  vaciarCarritoBtn.addEventListener("click", () => {
    articulosCarrito = []; // Reseteamos el array devolviendolo a vacío
    limpiarHTML(); // Llamamos a la función para actualizar el estado del carrito
  });
}
```

FIN