Práctica 2

JavaScript y jQuery

Ejercicio 1 – HTML – CSS

Al sitio web desarrollado en la práctica 1, se deben realizar las siguientes modificaciones:

1. Aplicar las propiedades CSS necesarias al menú de opciones para que cada una de ellas se liste en forma horizontal.
2. Agregar un nuevo ítem llamado **Encuesta** que referencie a una nueva página HTML. Dicha página debe contar con un formulario con los siguientes campos:

    a. Nombre.
    b. Apellido.
    c. Fecha de Nacimiento.
    d. Sexo.
    e. Grupo de opciones de valoración de la página: "Lamentable", "Mala", "Podría ser mejor", "Cumple con mis expectativas", "Me encantó".
    f. Email.
    g. Comentario.
    h. Botón **Enviar**.
    i. Botón **Cancelar**.
    j. Botón **Restablecer valores**.

Ejercicio 2 – JavaScript

Incorporar las funciones JavaScript necesarias para realizar las siguientes validaciones sobre los datos ingresados por el usuario en el formulario.

- ✓ Los campos **a, b, c, d, e** y **f** son obligatorios, por lo que no pueden estar vacíos.
- ✓ Los campos nombre y apellido sólo pueden aceptar caracteres de la "a-z" y "A-Z".
- ✓ La fecha de nacimiento debe tener formato **dd-mm-aaaa**
- ✓ El email debe tener formato de mail correcto.

Aclaración: *no se deben usar las validaciones por HTML5.*

Ejercicio 3 – JavaScript

Agregar los siguientes eventos:

- ✓ Cuando el usuario oprima el botón **Enviar,** todos los campos del formulario deben ser mostrados al usuario mediante un mensaje de alerta!
- ✓ Si el usuario cancela la encuesta, debe aparecer un mensaje de confirmación con las opciones *"Sí, deseo volver a la página anterior"* y *"No".* Al confirmar la acción, deberá llevar al usuario a la página anterior.
- ✓ El botón **Restablecer valores** debe limpiar todos los campos del formulario.

Ejercicio 4 – jQuery

Sobre una de las páginas HTML donde se muestran los productos, se debe simular un carrito de compras. Al hacer click sobre la imagen de un producto, se deben desencadenar las siguientes acciones:

1. La imagen debe cambiar su opacidad a un valor comprendido entre 50% y 75%.
2. El elemento HTML que contiene al producto debe modificar su color de fondo.
3. Una función deberá calcular el valor de todos los productos seleccionados, y deberán mostrarse al usuario en algún lugar de la página.
4. Si el producto es clickeado nuevamente, el mismo es deseleccionado, por lo que deberá recuperar su estado original (opacidad y color de fondo), y se deberá recalcular el valor total del carrito.

Aclaración: *para simplificar la implementación, suponer que todos los productos tienen el mismo costo.*

Ejercicio 5 – JSON

Modele un objeto en formato JSON que represente el producto promocionado por su comercio, teniendo en cuenta todos sus atributos. Por ejemplo, si su comercio es una librería, se debe modelar el objeto **libro**, el cual estará formado por los atributos: **id, título, imagen de portada, sinopsis, editorial, año de edición**.

Ejercicio 6 – JSON Generator

Hasta el momento los productos mostrados en cada una de las categorías son estáticos, fueron definidos directamente en el código HTML. En la realidad esto no ocurre, los productos que se muestran se van actualizando cada vez que el usuario accede al sitio web, obteniéndose de una base de datos por ejemplo.

Nuestra materia está enfocada en la creación de aplicaciones web poniendo el foco en el frontend, por lo que no vamos a utilizar bases de datos para obtener los productos a mostrar. **Pero podemos simularlos**.

Existen herramientas web que permiten generar datos aleatorios en distintos formatos. Una herramienta útil para crear datos en formato JSON es **json-generator**, la cual puede ser accedida desde la URL:

https://www.json-generator.com/#

Lo primero que debemos hacer es definir la estructura del objeto JSON que representa nuestro producto, sobre el panel izquierdo. Siguiendo con el ejemplo de la librería, la siguiente estructura representa el objeto libro con su id, título, una imagen, y la sinopsis.

```
[
  '{{repeat(1, 10)}}',
  {
    id: '{{index()}}',
    titulo: '{{name()}}',
    portada: 'http://placehold.it/32x32',
    sinopsis: '{{lorem(1, "paragraphs")}}'
  }
]
```

Al hacer click en **Generate** se creará una lista de 10 libros en formato JSON sobre el panel derecho.

Una gran ventaja de esta herramienta, es la posibilidad de disponibilizar estos datos fácilmente como un recurso del propio servidor con dos simples pasos: clickear en **"Upload JSON to server"** y luego en **"Copy JSON file URL".**

En el clipboard ya tienen guardada la URL que dispone el JSON generado. Si copian y pegan dicha URL en el navegador podrán acceder a la lista de libros.

Ejercicio 6 – AJAX

En este ejercicio se busca que los productos publicados por el comercio, no se encuentren embebidos en el HTML, sino que se obtengan dinámicamente al cargar la página. Para ello, vamos a hacer uso de AJAX.

Sobre una de las páginas HTML donde se muestran los productos (distinta a la que seleccionaron para implementar el carrito de compras), se debe realizar una petición HTTP GET sobre la URL generada en el **ejercicio 5**. Se busca que la respuesta (la lista de 10 productos en formato JSON), sea mostrada en forma ordenada en el navegador del usuario, manteniendo y respetando la estructura del Layout definida hasta el momento.

Puntos a tener en cuenta para la resolución de la práctica:

- Los documentos HTML deben seguir siendo semánticos, por lo que los recursos JavaScript y jQuery deben colocarse en el directorio **"js/**".
- Los ejercicios 2 y 3 se plantean para utilizar JavaScript puro, pero quien lo desee puede utilizar jQuery en toda la práctica. El objetivo de verlos por separado es que puedan observar las diferentes formas que existen para manipular los elementos del DOM, capturar y gestionar los eventos, generar animaciones y efectos, etc.
- Con los ejercicios 4, 5 y 6 se busca modificar la aplicación para que los productos que se muestran se carguen en forma dinámica y no se encuentren precargados en el HTML. Para ello, simulamos la obtención de productos mediante una petición HTTP a una **API** que nos provea dichos datos (proceso que utilizarán en el trabajo integrador).