Feedback Modulo IV

DESARROLLO WEB EN ENTORNO CLIENTE

Vianca Tereza Franco Rodriguez

ENUNCIADO

El objeto de esta práctica es simular un escenario real de una página web de cualquier servidor existente en internet.

Esto es, dada una lista desplegable de datos, una vez seleccionado un dato de esta lista, se hace una petición asíncrona (AJAX) al servidor buscando los datos relacionados con el dato seleccionado por el usuario en la web. Esto se hace en segundo plano, sin interferir en cualquier otra acción que esté realizando el usuario.

Esta aplicación mostrará una lista de países, y cuando el usuario seleccione uno de ellas, buscará en un fichero JSON (como si fuese una base de datos), las ciudades del país seleccionado, que se presentarán en la página web.

Como el fichero de datos JSON es un recurso local, el programa que realizará el usuario (.html) se encontrará en el mismo directorio que el archivo de datos .json.

El archivo .html será ejecutado desde un servidor web (Web server Chrome sino tenemos otro) ya que se accederá al fichero de datos, listadoPaises.json que es un recurso local de nuestro equipo, por esto se tiene que ejecutar desde el servidor por temas de seguridad.

Téngase en cuenta la estructura de un fichero JSON (pares de clave/valor).

Téngase en cuenta la estructura de un fichero JSON (pares de clave/valor). Un extracto del fichero de datos JSON es como sigue (listadoPaises.json):

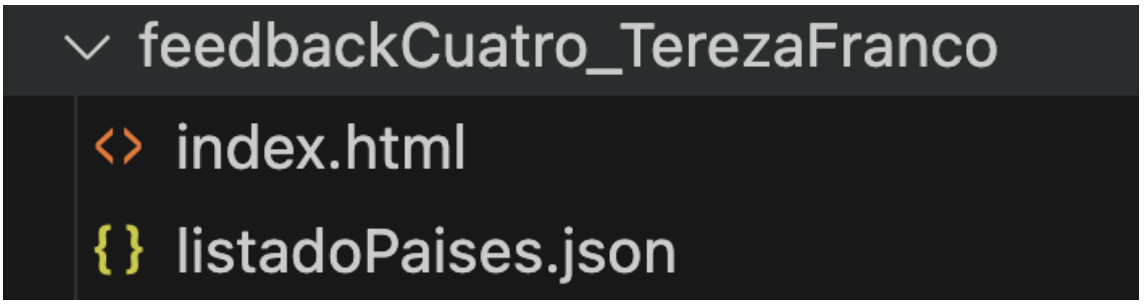

```
{
 "listadoPaises":

"pais": [ {

"nombre": "España",
 "capital": "Madrid",
 "textoCapital": "La mejor capital del mundo", "ciudades": [

"Madrid", "Barcelona", "Valencia", "Sevilla", "Zaragoza", "Málaga", "Murcia
] },

{ "nombre": "México",
"capital": "México D.F.",
"textoCapital": "La Ciudad de México, Distrito Federal o, en su forma abreviada, México, D.F., es... ",
```

"ciudades": ["México D.F.", "Ecatepec", "Guadalajara", "Puebla", "Juárez", "Tijuana", "León", "Zapopan"

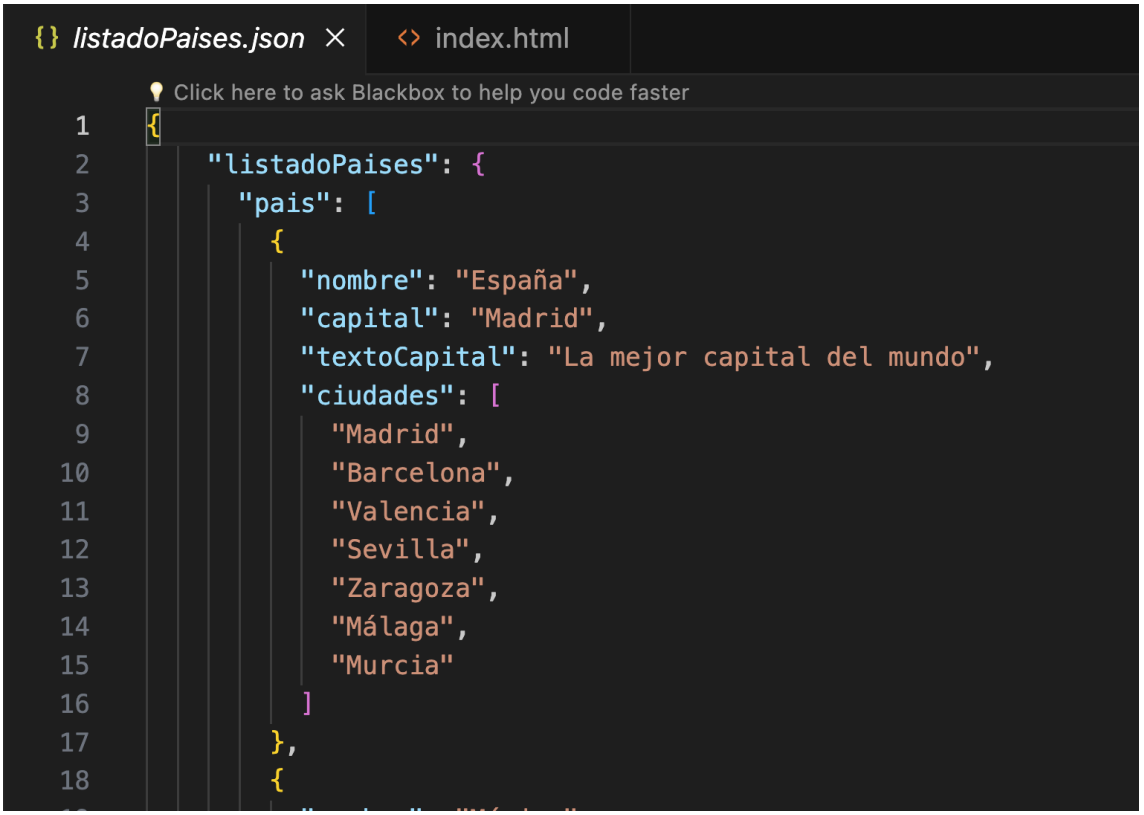
] }, ...

RESUMEN DEL PROGRAMA

Primero, se crean dos archivos uno .html y uno .json



En el archivo listadoPaises.json se agrega la lista de países dada en el enunciado, añadiendo un país adicional.



En el archivo ***index.html***, primero se crea un elemento ***select*** con un identificador único y se le añade varias opciones de selección, asignando los nombres de los países. Después, se crea una lista con el identificador ***listaCiudades*** para mostrar las ciudades según el país seleccionado. Ahí se mostrarán los datos extraídos del archivo ***listadoPaises.json*** que he creado anteriormente.

```
<div class="contenedor-central">
    <!-- Se crea un selector con el nombre de los paises -->
    <select id="paises">
        <option value="" selected>Selecciona un país</option>
        <option value="España">España</option>
        <option value="México">México</option>
        <option value="Venezuela">Venezuela</option>
    </select>

    <div class="ciudades">
        <h2>Ciudades del país seleccionado:</h2>

        <!-- Se crea una lista con las ciudades del pais seleccionado, inicialmente vacia -->
        <ul id="listaCiudades">
            No hay pais seleccionado.
        </ul>
    </div>
</div>
```

Se crea un script de JavaScript dentro del HTML para manejar la solicitud HTTP.

1. Se utiliza una función lambda llamada **dameCiudades** que toma el nombre de un país en formato String como argumento.

2. Se crea un objeto de tipo ***XMLHttpRequest.***

3. Se indica la ruta relativa del archivo ***listadoPaises.json*** que contiene los datos a utilizar. Luego con el método ***.open()*** abrimos la petición HTTP asíncrona de tipo ***GET*** en la URL indicada.

```
<script>
    // Se crea una lambda para manejar la peticion HTTP
    // Toma como argumento el nombre de un pais en formato String
    const dameCiudades = (nombrePais) => {
        // Se crea un objeto de tipo XMLHttpRequest
        const peticion = new XMLHttpRequest();

        // indicamos la ruta relativa de la peticion
        const url = './listadoPaises.json';

        // Abrimos la petición HTTP asíncrona, de tipo GET en la URL indicada
        peticion.open('GET', url, true);
```

4. Se crea un manejador de eventos ***onreadystatechange*** para la petición, que se dispara cuando esta ha cambiado de estado. Luego mediante un condicional, primero evalúo si el evento ***readystate*** se ha completado y después evalúo si la petición ha sido correcta.

5. Parseamos el JSON de la respuesta para obtener la lista de países. Después, con un bucle forEach recorremos cada país de la lista, y con un condicional comprobamos que el nombre del país sea el que hemos seleccionado. Si lo es, limpiamos la lista de ciudades.

```
        // Se crea un manejador de eventos para la peticion,
        // que se dispara cuando la peticion cambia de estado
        peticion.onreadystatechange = () => {
            // Cuando el estado es DONE, la peticion se ha completado
            if (peticion.readyState === XMLHttpRequest.DONE) {
                // Se analiza el estado de la peticion
                // Cuando es HTTP 200, la peticion ha sido correcta
                if (peticion.status === 200) {

                    // Parseamos el JSON de la respuesta, para obtener la lista de paises
                    const datos = JSON.parse(peticion.responseText);

                    // Para cada pais de la lista de paises
                    datos.listadoPaises.pais.forEach(pais => {
                        // Comprobamos que el nombre del pais sea el que hemos seleccionado
                        if (pais.nombre === nombrePais) {
                            // Si el pais es el seleccionado, limpiamos la lista de ciudades
                            const ciudades = document.getElementById('listaCiudades');
                            ciudades.innerHTML = ''
```

6. Posteriormente, recorremos la lista de ciudades de los datos que hemos obtenido de la petición ***Ajax*** y la agregamos a la lista.

7. Con el método ***createElement*** creamos un elemento ***li***, con ***innerHTML*** asignamos el valor de la ciudad y con ***appendChild*** agregamos el elemento a la lista ciudades.

```
                    // Posteriormente, recorremos la lista de ciudades de los datos
                    // que hemos obtenido de la peticion AJAX, y lo agregamos a la lista
                    pais.ciudades.forEach(ciudad => {
                        // Se crea un elemento li
                        const el = document.createElement("li")

                        // Le asignamos el valor de la ciudad
                        el.innerHTML = ciudad

                        // Agregamos el elemento a la lista
                        ciudades.appendChild(el)
                    })
                }
            })
        } else { // Si no es correcta, lanzamos un error por la consola
            console.error('Error al cargar los datos: ' + peticion.status);
        }

    }
};
```

8. Evaluamos si el país seleccionado no es correcto, y lanzamos un error por consola.

9. Por último, lanzamos la petición HTTP utilizando el método ***.send()***

```
        // Lanzamos la peticion HTTP
        peticion.send();
    }
```

Por otra parte, se crea un manejador de eventos para el evento ***onchange*** en el selector con id **paises**. Primero, se obtiene el valor del selector que sería el nombre del país. Luego, con un condicional evaluamos si el selector no está vacío. Si no lo está, se llama a la función ***dameCiudades*** con el nombre del país seleccionado. En caso de estar vacío, se limpia la lista de ciudades y se agrega un texto a la lista.

```
    // Se crea un manejador de eventos para el evento change en el selector con id paises
    document.getElementById('paises').onchange = (e) => {
        // Se obtiene el valor del selector, que seria el nombre del pais
        const pais = e.currentTarget.value

        // Si el pais no es vacio, se llama a la funcion dameCiudades
        // con el nombre del pais
        if (pais !== "") {
            dameCiudades(pais)
        } else { // Si es vacio, se limpia la lista de ciudades
            const ciudades = document.getElementById('listaCiudades');
            // Se agrega el texto dentro de la lista de ciudades
            ciudades.innerHTML = 'No hay pais seleccionado.'
        }
    }
</script>
```

Por último, se agrega código css para darle mejor estilo a la caja de texto y mejorar la visualización del programa.

```
/* Estilo para centrar todo el contenido*/
.contenedor-central {
    display: flex;
    flex-direction: column;
    align-items: center;
}


/* Estilo para el botón de selección de países */
#paises {
    padding: 10px;
    font-size: 16px;
    border: 2px solid #3498db;
    border-radius: 5px;
    background-color: #ffffff;
    color: #3498db;
    cursor: pointer;
}
```