**Unidad 6: Eventos.**

**Ejercicios**

**INTRODUCCIÓN**

#### EJERCICIO 1: Ocultar texto

Agrega el código JavaScript al ejercicio 1 para que, al hacer clic en el button, desaparezca el texto del div con id="text".

#### EJERCICIO 2: Ocultar botón

Crea una copia del ejercicio anterior.

Modifica el código, para que, al hacer clic en el button, desaparezca el button.

#### EJERCICIO 3: Mover el balón por el campo

Agrega el código JavaScript al ejercicio 3 para que, al hacer clic en algún lugar del campo, el balón se mueva a ese lugar.

* El balón debe quedar centrado donde se ha hecho click, si es posible.
* El balón no debería salir del campo.
* Si la página se desplaza, el balón debería seguir correctamente colocado.

#### EJERCICIO 4: Crear un menú deslizante

Modifica el HTML/CSS del documento y agrega el código JavaScript al ejercicio 4.

Al hacer click sobre el texto Sweeties (click me)! , se debería mostrar/ocultar la lista que está debajo.

#### EJERCICIO 6: Carrusel

Agrega el código JavaScript al ejercicio 5 para que:

* Inserte un botón de cierre en la esquina superior derecha de cada mensaje.
* Haciendo clic en ese botón, el panel se debe eliminar (no desaparecer)

**MÉTODOS PARA BUSCAR**

#### EJERCICIO 4: Buscar elementos en el DOM

https://es.javascript.info/searching-elements-dom#buscar-elementos

Dado un html, se pide escribir las expresiones para acceder a:

1. La tabla con id="age-table".
2. Todos los elementos labeldentro de la tabla (debería haber 3).
3. El primer td en la tabla (con la palabra “Age”).
4. El form con name="search".
5. El primer input en ese formulario.
6. El último input en ese formulario.

**PROPIEDADES DEL NODO: TIPO, ETIQUETA Y CONTENIDO**

#### EJERCICIO 5: Contar los descendientes

https://es.javascript.info/basic-dom-node-properties#contar-los-descendientes

Dado un árbol estructura como ul/li anidado, escribir el código para que

* Para cada <li> muestre:
  + Cuál es el texto que hay en el <li> (sin el subárbol)
  + El total de <li> descendientes

**ATRIBUTOS Y PROPIEDADES**

#### EJERCICIO 6: Obtener un atributo personalizado

https://es.javascript.info/dom-attributes-and-properties#obten-en-atributo

Escribe el código para mostrar el valor del elemento que tiene como atributo data-widget-name

<!DOCTYPE html>

<html>

<body>

<div data-widget-name="menu">Elige el genero</div>

<script>

/\* Tu código \*/

</script>

</body>

</html>

#### EJERCICIO 7: Cambiar el color del texto de algunos enlaces externos

https://es.javascript.info/dom-attributes-and-properties#haz-los-enlaces-externos-naranjas

Escribe el código para que algunos enlaces externos tengan como color del texto el naranja.

Los enlaces naranjas son los que:

* Su href tiene ://
* Pero no comienza con http://internal.com.

<a name="list">the list</a>

<ul>

<li><a href="http://google.com">http://google.com</a></li>

<li><a href="/tutorial">/tutorial.html</a></li>

<li><a href="local/path">local/path</a></li>

<li><a href="ftp://ftp.com/my.zip">ftp://ftp.com/my.zip</a></li>

<li><a href="http://nodejs.org">http://nodejs.org</a></li>

<li><a href="http://internal.com/test">http://internal.com/test</a></li>

</ul>

<script>

// establecer un estilo para un enlace

let link = document.querySelector('a');

link.style.color = 'orange';

</script>

**CUESTIONES**

#### EJERCICIO 8: Indicar la salida por consola

Indica la salida que nos darían por consola las siguientes expresiones, después de haber ejecutado las siguientes sentencias:

<!DOCTYPE html>

<html>

<head><meta charset="utf-8"></head>

<body>

<h1 id="id1" class="c2">Título</h1>

<div id="id2" class="c2">Coche</div>

<script>

let x = document.getElementById("id1");

let y = document.getElementById("id4");

let z = document.querySelectorAll("#id2");

let t = document.querySelectorAll("div.c2");

let v = document.querySelectorAll("#id4");

let w = document.querySelectorAll("h3.c2");

</script>

</body>

</html>

1. x.outerHTML;
2. z[0].innerHTML;
3. v;
4. t[0].innerHTML;
5. x.innerHTML;

#### EJERCICIO 9: Seleccionar un elemento

Escribe dos sentencias diferentes para poder seleccionar el primer elemento del documento HTML con el selector CSS "p.destacado"

#### EJERCICIO 10: Utilizar .textContent

Indica si la siguiente sentencia es correcta. En caso de que no los sea, qué modificación harías.

elemento.textContent = "Hola" + "<br>";

#### EJERCICIO 11: Seleccionar texto

Suponiendo que elemento tiene almacenado el primer elemento de la lista, escribe el código necesario para almacenar en una variable la frase "**Ubuntu es una distribución de Linux**" utilizando propiedades y/o métodos del DOM, y partiendo de elemento.

<ul id="Linux">

<li>Ubuntu</li>

<li>Mint</li>

<li>Debian</li>

<li>Red Hat</li>

</ul>

#### EJERCICIO 12: nodeType

Completa el siguiente código:

elemento = document.getElementById("prueba");

if ( === 8){

console.log("Comentario");

} else if ( === 1){

console.log(" ElementoHTML");

}

#### EJERCICIO 13: nodeName

Completa el siguiente código:

elemento = document.getElementById("prueba");

switch ( ){

case ("INPUT"):

console.log("Es un input");

break;

    case ("LABEL"):

console.log("Es un label");

     break;

case ("BUTTON"):

console.log("Es un button");

break;

}

**MODIFICANDO EL DOCUMENTO**

#### EJERCICIO 14: Crear una tabla

Ejercicio hecho en clase, para entender algunos métodos: append, prepend, after, before, remove, clone.

* Crea una función que recibe como parámetros dos números (número de filas y número de columnas).  
  En la función se crea una tabla de esas filas y esas columnas. Cada celda tendrá algo de contenido de texto.  
  La tabla se "engancha" al árbol del DOM, al colocarla como hija del body.  
  La función devolverá un puntero al elemento del DOM asociado a la tabla.
* Crea una función a la que pasas como parámetro un puntero a la tabla y el número de columnas que tiene la tabla.  
  La función insertará, al principio de la tabla, una fila con ese número de columnas.
* Crea una función a la que pasas como parámetro un puntero a la tabla.  
  La función insertará, antes de la tabla, un elemento de tipo cabecera (h1, h2, ...) con un texto.
* Crea una función a la que pasas como parámetro un puntero a la tabla y un número de fila.  
  La función eliminará esa fila de la tabla.
* Crea una función a la que pasas como parámetro un puntero a la tabla.  
  La función creará otra tabla igual a la que se pasa como parámetro, y la colocará detrás de la tabla que ya existe.

#### EJERCICIO 15: Crear una lista

Escribir una interfaz para crear una lista de lo que ingresa un usuario.

Para cada item de la lista:

1. Preguntar al usuario acerca del contenido usando prompt.
2. Crear el <li> con ello y agregarlo a <ul>.
3. Continuar hasta que el usuario cancela el ingreso (presionando Esc o con un ingreso vacío).

Todos los elementos deben ser creados dinámicamente.

Si el usuario ingresa etiquetas HTML, deben ser tratadas como texto.

#### EJERCICIO 16: Crear un árbol desde el objeto

Escribe una función createTree que crea una lista ramificada ul/li desde un objeto ramificado.

Por ejemplo:

let data = {

"Fish": {

"trout": {},

"salmon": {}

},

"Tree": {

"Huge": {

"sequoia": {},

"oak": {}

},

"Flowering": {

"apple tree": {},

"magnolia": {}

}

}

};

La sintaxis:

let container = document.getElementById('container');

createTree(container, data); // crea el árbol en el contenedor

El árbol resultante debe verse así:

* Fish
  + trout
  + salmon
* Tree
  + Huge
    - sequoia
    - oak
  + Flowering
    - apple tree
    - magnolia

P.S. El árbol no debe tener elementos “extras” como <ul></ul> vacíos para las hojas.

#### EJERCICIO 17: Mostrar descendientes en un árbol

Hay un árbol organizado como ramas ul/li.

Escribe el código que agrega a cada <li> el número de sus descendientes. No cuentes las hojas (nodos sin hijos).

El resultado:

* Animals [9]
  + Mammals [4]
    - Cows
    - Donkeys
    - Dogs
    - Tigers
  + Other [3]
    - Snakes
    - Birds
    - Lizards
* Fishes [5]
  + Aquarium [2]
    - Guppy
    - Angelfish
  + Sea [1]
    - Sea trout

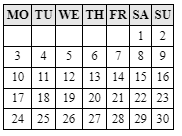
#### EJERCICIO 18: Crear un calendario

Escribe una función createCalendar(elem, year, month).

Cuando se llama, debe crear un calendario para el año y mes dados y ponerlo dentro de elem.

El calendario debe ser una tabla, donde una semana es <tr>, y un día es<td>. Los encabezados de la tabla deben ser <th> con los nombres de los días de la semana: el primer día debe ser “lunes” y así hasta “domingo”.

Por ejemplo, createCalendar(cal, 2012, 9) debe generar en el elemento cal el siguiente calendario:



#### EJERCICIO 19: Reloj coloreado con setInterval

Crea un reloj que muestra la fecha actual. Cuando se pulsa en Start, muestra la hora/minutos/segundos, que se actualiza cada segundo; y cuando se pulsa en Stop, deja de actualizarse.

#### EJERCICIO 20: Insertar HTML en la lista

Escribe el código para insertar <li>2</li><li>3</li> entre dos <li> aquí:

<ul id="ul">

<li id="one">1</li>

<li id="two">4</li>

</ul>

Es más fácil utilizando .adjacentInnerHTML().

#### EJERCICIO 21: Ordenar la tabla

Tenemos una tabla como la siguiente. Escribe el código para ordenarla por la columna "name".

<table>

<thead>

<tr>

<th>Name</th><th>Surname</th><th>Age</th>

</tr>

</thead>

<tbody>

<tr>

<td>John</td><td>Smith</td><td>10</td>

</tr>

<tr>

<td>Pete</td><td>Brown</td><td>15</td>

</tr>

<tr>

<td>Ann</td><td>Lee</td><td>5</td>

</tr>

<tr>

<td>...</td><td>...</td><td>...</td>

</tr>

</tbody>

</table>

#### EJERCICIO 22: Crear una notificación

Escribir una función showNotification(options) que cree una notificación: <div class="notification"> con el contenido dado. La notificación debería desaparecer automáticamente despues de 1.5 segundos.

Ejemplo:

// muestra un elemento con el texto "Hello" cerca de la parte superior de la ventana

showNotification({

top: 10, // 10px desde la parte superior de la ventana (por defecto es 0px)

right: 10, // 10px desde el borde derecho de la ventana (por defecto es 0px)

html: "Hello!", // el HTML de la notificación

className: "welcome" // una clase adicional para el "div" (opcional)

});

**TAMAÑO DE ELEMENTOS Y DESPLAZAMIENTO**

#### EJERCICIO 23: Coloca la pelota en el centro del campo

Utilizando JavaScritp (no CSS), coloca la pelota en el centro del campo verde.

El código debería funcionar con una pelota de cualquier tamaño y con cualquier tamaño de campo.

**COORDENADAS**

#### EJERCICIO 24: Muestra las coordenadas del campo en la ventana

Utilizando JavaScript, calcula las coordenadas de las esquinas de la ventana marcadas con triángulos.

El código proporcionado permite saber las coordenadas de cualquier punto al hacer clic sobre él.

#### EJERCICIO 25: Muestra una nota cercana al elemento

Crea una función positionAt(anchor, position, elem) que posicione elem, respecto al elemento de referencia (anchor), dependiendo de lo indicado por position:

* "top": encima de anchor
* "right": a la derecha de anchor
* "bottom" debajo de anchor

Esta función será utilizada por showNote(elemRef, position, html) que crea un elemento "note" con el html y lo muestra.

#### EJERCICIO 26: Muestra una nota cercana al elemento (absolute)

Modifica la solución de la tarea anterior, de forma que la nota utilice position:absolute en lugar de position:fixed.

Esto evitará que la nota se descoloque si se hace scroll en la página.

Para comprobarlo, agrega el estilo:  <body style="height: 2000px">.