DOM, eventos y programación asíncrona

NOTA: los objetivos de aprendizaje de esta tarea buscan las siguientes metas:

Referentes a la UT07

a) Sobre el DOM, tipos de nodos, nodelist, manipulación de atributos, manipulación de los contenidos de los elementos, manipular clase, navegación por el DOM.
b) Añadir elementos, crear elementos, crear nodos texto, añadir un nodo hijo, inserción de nodos, remplazar elementos, eliminar elementos.
c) Colecciones vivas, temporizadores, settimeout y setinterval.

Referentes a la UT08

d) Uso de escuchadores de eventos, asignación de eventos de forma dinámica, obtener tecla pulsada, obtener los botones del ratón, anular comportamientos predeterminados, tipos de eventos, eventos del portapapeles.
e) Formularios, eventos en formularios, propiedades de los controles.

Referentes a la UT09

f) JavaScript estricto, lanzar errores propios, gestionar excepciones, módulos y paquetes, creación de promesas, el método Then, el método catch.

·Tareas sobre el DOM BLOQUE 1·

1. Construye un ejemplo en el que se puedan visualizar los diferentes tipos de nodos del DOM

(Página 259) Realiza una aplicación que recupere los diferentes tipos de nodos;

1.1. Deben aparecer "Elemento", "Atributo", "Texto", "Apartado CDATA", "Referencia a entidad", "Entidad", "Instrucción de procesado", "Comentarios".

2. Construye una aplicación que vaya recuperando imágenes de forma aleatoria

(Páginas 273, 264,) La aplicación debe disponer de un botón que indique añadir nueva imagen, y cada vez que se pulse se añada una imagen en la ventana visible del navegador:

2.1. Se debe crear el elemento con el método "createElement".

2.2. Se deben crear los atributos necesarios de la imagen con los métodos de añadir atributo

2.3. Se debe añadir el valor de la url y el resto de valores del atributo con los métodos correspondientes para ello.

2.4. Las imágenes se deben recuperar del servidor, para ello se tendrán varias imágenes alojadas en el servidor con unos nombre prefijados, y se recuperará un valor aleatorio para cada una, por ejemplo 1.jpg , 2.jpg , 3.jpg , 4.jpg , 5.jpg siendo los números aleatorios a recuperar del 1 al 5.

3. Crea un ejemplo que permita vislumbrar el comportamiento de las colecciones vivas

(Página 276, 278 y 280) Construye una aplicación que muestre 10 elementos recuperados con "getElementByClassName" y "querySelectorAll", de manera que se visualice como se va actualizando cada uno.

3.1. El programa debe mostrar dos listas de elementos una a la derecha y otra a la izquierda, en la ventana visible del navegador.

3.2. A través de un botón se deben ir eliminando elementos. Cada vez que se elimine un elemento, se deben mostrar de nuevo las listas, pero si llamar al método, según estaban.

3.3. Se debe mostrar el contenido recuperado con los métodos anteriores "getElementByClassName" y "querySelectorAll", SIN ACTUALIZARLOS, no volver a llamarlos, si no, solamente recorrerlos. Fijarse en ejemplo de la página 178.

3.4. Añade un temporizador para que la funcionalidad del botón se ejecute cada segundo. Esta funcionalidad se debe poner activar y desactivar a través de un segundo botón.

·Gestión de eventos BLOQUE 2·

4. Construye un programa que permita lanzar dardos a una diana electrónica

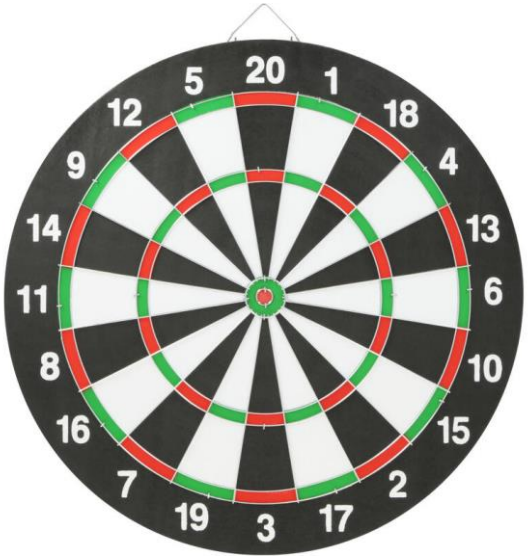
4.1. Al pinchar en la diana, el dardo se debe colocar en la posición que se ha pinchado.

4.2. Los dardos pueden aparecer al pinchar (se posicionan al aparecer), o situarse en la parte inferior de la ventana visible (se mueven de posición al pinchar).

4.3. Existirán 6 dardos, 3 por cada jugador, primero lanzará los dardos un jugador y después el otro.

4.4. Se contabilizarán las puntuaciones de cada jugador, según la posición de la diana en la que ha pinchado.

4.5. Aunque los colores de la diana se deben ajustar en cada circulo, no es obligatorio que se cuenten la posición de cada valor, solo los valores de cada circulo concéntrico.

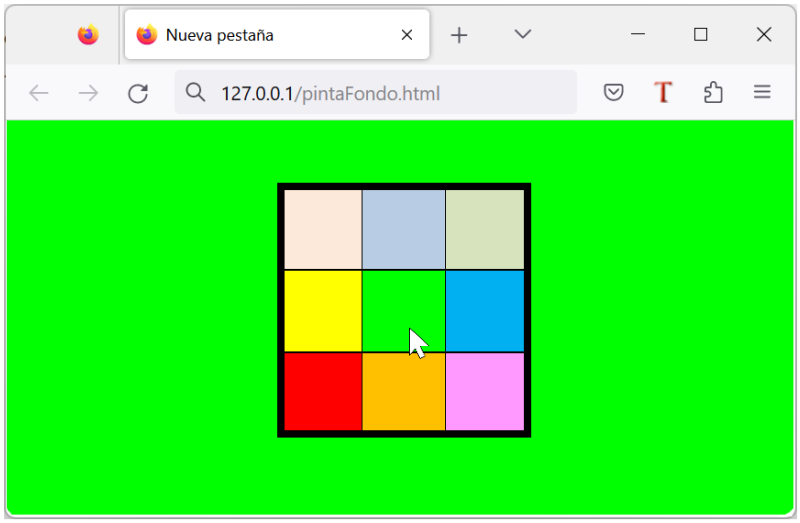


5. Construye un programa que haga que cambie el fondo de la ventana visible

(Página 312) A través de los eventos del ratón, construye un programa que cuando el ratón pase por dentro de cada elemento cuadrado, el fondo de la ventana visible del navegador cambia a ese color. Se debe seguir una estructura como la que se muestra a continuación:

5.1. Al introducir el puntero del ratón en cada uno de los cuadros el fondo de la ventana visible debe cambiar al color del cuadrado.

5.2. Al salir de los 9 cuadros, el fondo debe volver a tener el color blanco.



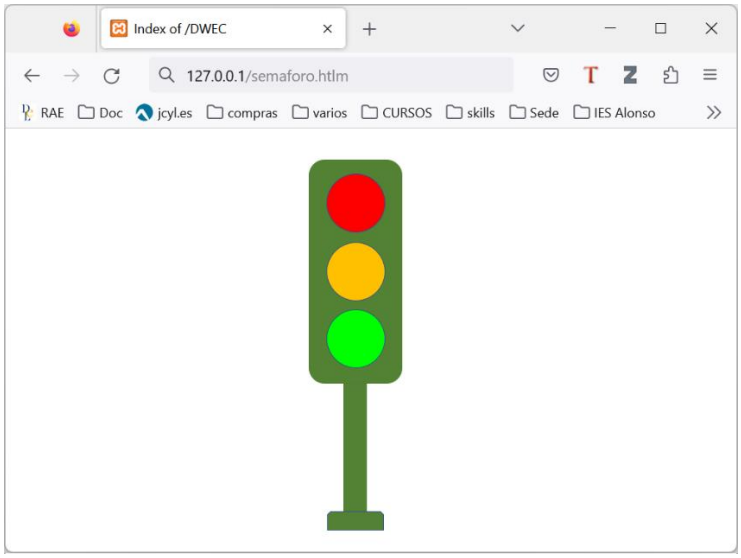
6. Construye un semáforo que realice diferentes comportamientos prefijados

El programa debe activar el semáforo, en función de si se hace clic con el ratón en el círculo del color correspondiente, según las siguientes indicaciones:

6.1. Si se hace clic en el circulo superior el semáforo debe realizar la función tradicional de un semáforo, y debe permanecer con el color rojo durante 6 segundos, pasar a encenderse el color verde y permanecer durante 6 segundos, parpadear el verde durante 3 segundos, pasar al naranja durante 2 segundos, y volver al estado del rojo. Este ciclo se debe realizar cíclicamente.

6.2. Si se hace clic en el círculo del medio, el semáforo debe ir cambiando de rojo a verde cada segundo.

6.3. Si se hace clic en el círculo de abajo el semáforo debe permanecer parpadeando solo con el color ámbar, cada segundo, un segundo encendido y un segundo apagado.



·Conceptos avanzados BLOQUE 3·

7. Suma y lanza errores

(Página 348 y 349) Crea una aplicación web que muestre sumas aleatorias, permitiendo al usuario añadir la solución, en caso de que el resultado de la suma no sea correcto, se debe lanzar un error haciendo uso del, new Error("Mensaje del error").

7.1. Crea una nueva versión del programa que controle el error a través de un bloque try...catch. En este caso, los errores se lanzarán, por ejemplo si el valor introducido es una cadena de texto.

Nota: deben poder ejecutarse de forma independiente las dos formas de gestionar el error

8. La carrera de las tortugas

(Página 364) Crea una aplicación web basada en el ejemplo de **funciones async** de la **página 364**, que permita realizar una carrera de tortugas. El funcionamiento debe ser el siguiente:

8.1. Cada tortuga debe obtener un número aleatorio de 1 a 10. En función de ese número, la tortuga debe ir hasta la meta a una velocidad diferente, con 1 despacio, con 2 más deprisa, y así sucesivamente hasta el 10, que es la velocidad más rápida.

8.2. Además, la tortuga obtendrá otro número aleatorio de 1 a 3, de manera que si obtiene el número 1 sale en 1 segundo, si obtiene el 2, sale en 2 segundos y si obtiene el 3, sale a los 3 segundos del inicio de salida.