

Kahoot - Tema 8

¿Cuál de estas funciones no dibuja directamente un rectángulo?

- A) drawRect ☒
- B) strokeRect
- C) clearRect
- D) fillRect

Explicación: En la API de Canvas de HTML5, las funciones estándar son `fillRect` (relleno), `strokeRect` (contorno) y `clearRect` (borrar). La función `drawRect` no existe como tal en la API nativa; se suele usar `rect()` dentro de un trazado o es un nombre inventado en este contexto.

¿Qué ocurre si se llama a `fill()` sobre un trazado abierto?

- A) Produce un error
- B) Cierra automáticamente la forma ☒
- C) Solo rellena si hay `closePath`
- D) No dibuja nada

Explicación: El método `fill()` de Canvas rellena automáticamente el trazado actual. Si el camino está abierto, la función traza una línea recta imaginaria desde el punto final hasta el punto inicial para cerrar la forma y poder aplicar el relleno.

¿Cuál de las siguientes propiedades estándar de un objeto Error devuelve el mensaje descriptivo?

- A) value
- B) description
- C) message ☒
- D) type

Explicación: En JavaScript, la propiedad **message** del objeto Error contiene el texto descriptivo asociado al error que se ha producido.

¿En qué unidad se miden los ángulos del método arc()?

- A) Píxeles
- B) Porcentaje
- C) Grados
- D) Radianes ☒

Explicación: Los métodos de dibujo de arcos en canvas (como arc()) trabajan con ángulos expresados en **radianes**, no en grados.

¿Qué API permite almacenar datos sin enviarlos al servidor?

- A) Fetch API
- B) Notification API
- C) Web Storage API ☒
- D) Cookies

Explicación: La **Web Storage API** (localStorage y sessionStorage) permite guardar datos directamente en el navegador sin necesidad de mandarlos al servidor.

Una aplicación deja de ejecutarse tras una operación matemática inválida no controlada. ¿Qué error se ha producido?

- A) Error ☒
- B) Exception
- C) Warning
- D) Error de desarrollo

Explicación: Una operación matemática inválida no controlada provoca un **error** en tiempo de ejecución que detiene la aplicación.

¿Por qué los módulos mejoran la calidad del código?

- A) Favorecen reutilización y encapsulación ☒
- B) Eliminan la necesidad de funciones
- C) Evitan errores de sintaxis

D) Reducen el uso de memoria

Explicación: Los módulos permiten **reutilizar** código y mantenerlo encapsulado, separando responsabilidades y haciendo el código más limpio y mantenible.

¿Cuál es la sintaxis correcta para importar dos funciones desde un módulo?

A) import { f1, f2 } from "./modulo.js" 

B) import modulo from "modulo.js"

C) include modulo.js

D) require("./modulo.js")

Explicación: En los módulos ES6 se usan llaves para importar **múltiples** exportaciones con nombre desde un mismo archivo.

¿Cuál de las siguientes situaciones generaría un error en tiempo de ejecución?

A) Olvidar un punto y coma

B) Llamar a una variable no definida 

C) Escribir mal una palabra reservada

D) Usar una indentación incorrecta

Explicación: Intentar usar una **variable** que no ha sido definida provoca un error en tiempo de ejecución (por ejemplo, `ReferenceError` en JavaScript).

¿Desde qué punto se empieza a dibujar en canvas?

A) Esquina inferior izquierda

B) Esquina superior izquierda 

C) Centro del lienzo

D) Esquina superior derecha

Explicación: En un elemento **canvas** de HTML5, el sistema de coordenadas comienza en la esquina superior izquierda, que es el punto (0, 0).

¿Cuál de los siguientes ejemplos es una API web nativa?

A) DOM ☒

B) Node.js

C) Bootstrap

D) React

Explicación: El **DOM** (Document Object Model) es una API web nativa proporcionada por el navegador para interactuar con documentos HTML y XML.

En una validación de nota (0–10), ¿cuál sería la forma correcta de gestionar un valor fuera de rango?

A) Usar solo console.log

B) Ignorar el valor

C) Lanzar un Error con throw ☒

D) Mostrar un warning por consola

Explicación: Un valor fuera de rango rompe la lógica esperada, por lo que lo adecuado es **lanzar** una excepción con `throw` para poder gestionarla explícitamente.

¿Qué atributo debe indicarse en `<script>` para trabajar con módulos?

A) module="true"

B) async="module"

C) type="module" ☒

D) lang="js"

Explicación: Para usar módulos ES6 en el navegador, el `<script>` debe declararse con el atributo **type** igual a `"module"`.

¿Qué instrucción se utiliza para lanzar manualmente un error en JavaScript?

A) new Error()

B) try

C) throw ☒

D) catch

Explicación: La palabra clave **throw** permite lanzar manualmente un error (por ejemplo, `throw new Error("mensaje")`).

¿Qué método inicia un nuevo trazado en canvas?

A) startPath()

B) newPath()

C) openPath()

D) beginPath() 

Explicación: En la API de canvas, `beginPath()` inicia un **nuevo** trazado, separándolo del anterior.

¿Qué propiedad establece el color de relleno en canvas?

A) backgroundColor


B) colorFill

C) fillStyle 

D) strokeStyle

Explicación: En la API de canvas, la propiedad **fillStyle** define el color (o gradiente/patrón) con el que se rellenan las figuras.

¿Cuál es la diferencia clave entre localStorage y sessionStorage?

A) Persistencia tras cerrar el navegador 

B) Capacidad de almacenamiento

C) Tipo de datos

D) Velocidad de acceso

Explicación: Los datos en **localStorage** persisten aunque se cierre el navegador, mientras que los de **sessionStorage** se eliminan al cerrar la pestaña o ventana.

¿Qué ocurre con una función declarada en un módulo sin exportarla?

- A) Provoca un error
- B) Se convierte en global
- C) Solo es accesible desde el mismo módulo ☒
- D) Es accesible desde cualquier archivo

Explicación: En los módulos ES6, lo que no se **exporta** queda encapsulado y solo puede usarse dentro de ese mismo archivo.

¿Cuál es la conversión correcta de grados a radianes?

- A) $(\text{Math.PI} / 180) * \text{grados}$ ☒
- B) $\text{grados} * 180$
- C) $\text{grados} / \text{Math.PI}$
- D) $\text{grados} * \text{Math.PI}$

Explicación: Para pasar de grados a radianes se multiplica por $\pi/180$, es decir, `grados * (Math.PI / 180)`.

¿Qué define correctamente un módulo en JavaScript?

- A) Un conjunto de funciones globales
- B) Un objeto JSON externo
- C) Un archivo JS con código exportable ☒
- D) Un archivo HTML reutilizable

Explicación: Un módulo es un **archivo** JavaScript cuyo contenido puede exportarse e importarse desde otros archivos mediante `export` e `import`.

Si no se especifican los atributos width y height en un canvas, ¿qué tamaño tendrá por defecto?

- A) 500×500
- B) Depende del navegador
- C) 400×200
- D) 300×150 ☒

Explicación: El lienzo **canvas** tiene por defecto un tamaño de 300 píxeles de ancho por 150 de alto si no se indican atributos de tamaño.

¿Cuál es la principal diferencia práctica entre un Error y una Exception en JavaScript?

- A) El Error nunca detiene la ejecución del programa
- B) La Exception está prevista y gestionada por el programador ☒
- C) El Error se lanza manualmente y la Exception automáticamente
- D) La Exception solo se produce en tiempo de desarrollo

Explicación: En la práctica, se habla de **exception** cuando el error se contempla y maneja con `try/catch`, mientras que un **Error** no controlado detiene el programa.

¿Qué método solicita permiso al usuario para mostrar notificaciones?

- A) `requestPermission()` ☒
- B) `showPermission()`
- C) `allowNotification()`
- D) `notify()`

Explicación: La API de **Notifications** usa `Notification.requestPermission()` para pedir al usuario autorización antes de mostrar notificaciones.

Si dentro de un try se produce una excepción, ¿qué ocurre con el resto del código del bloque try?

- A) Se ejecuta parcialmente
- B) Se ignora completamente
- C) Se ejecuta después del catch
- D) No se ejecuta ☒

Explicación: Cuando se lanza una excepción dentro de `try`, la ejecución del bloque se interrumpe y el flujo pasa directamente al bloque `catch` o `finally`.


¿Cuál es el propósito principal del bloque finally?

- A) Capturar errores no controlados
- B) Evitar que se ejecute el bloque catch
- C) Liberar recursos independientemente de que haya error ☒

D) Ejecutarse solo si hay error

Explicación: El bloque **finally** se ejecuta siempre, haya o no excepción, y se usa típicamente para cerrar conexiones, liberar recursos o hacer limpieza final.

¿Qué elemento HTML se utiliza para dibujar gráficos mediante JavaScript?

- A) ``
- B) `<figure>`
- C) `<svg>`
- D) `<canvas>` 

Explicación:

El elemento `<canvas>` permite dibujar gráficos dinámicos usando JavaScript a través de su API.

¿Qué método se utiliza para obtener el contexto 2D del canvas?

- A) `getCanvas("2d")`
- B) `drawContext()`
- C) `getContext("2d")` 
- D) `render2D()`


Explicación: El método `getContext("2d")` devuelve el contexto necesario para dibujar en dos dimensiones dentro del canvas.

¿Qué método de la API Canvas permite descargar la imagen creada?

- A) `saveImage()`
- B) `toDataURL()` 
- C) `downloadCanvas()`
- D) `exportCanvas()`

Explicación: `toDataURL()` convierte el contenido del canvas en una imagen codificada que puede descargarse o guardarse.

¿Para qué se utiliza `moveTo(x, y)` ?

- Dibujar una línea
- Cambiar el color
- Cerrar el trazado
- **Mover la pluma sin dibujar** 

Explicación: `moveTo(x,y)` es un método del contexto de dibujo de canvas en JavaScript que mueve el "lápiz" a las coordenadas especificadas sin trazar ninguna línea. Se usa para establecer el punto inicial de un nuevo trazo o para reposicionar el punto de dibujo.