Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Esteban Garces Alzate
Repositorio: EstebanGarcesA/act_web1_s9
Fecha de evaluación: 5/10/2025, 13:19:42
Evaluado por: Sistema de Evaluación Masiva

Resumen de Calificaciones

Calificación general: 4.4/5.0
Actividades completadas: 10/10
Porcentaje de completitud: 100.0%

Detalle de Actividades

#	Descripción	Archivo	Encontrado	Calificación
1	Información Básica del Documento - Crea ...	ejercicios/ejercicio_01.js	Sí	4.0
2	Seleccionar Elementos por ID - Crea una ...	ejercicios/ejercicio_02.js	Sí	4.0
3	Seleccionar Elementos por Clase - Crea e...	ejercicios/ejercicio_03.js	Sí	5.0
4	Cambiar Contenido de Elementos - Crea el...	ejercicios/ejercicio_04.js	Sí	4.0
5	Modificar Atributos - Crea elementos con...	ejercicios/ejercicio_05.js	Sí	5.0
6	Agregar y Quitar Clases CSS - Crea eleme...	ejercicios/ejercicio_06.js	Sí	4.0
7	Crear y Agregar Elementos - Crea nuevos ...	ejercicios/ejercicio_07.js	Sí	4.0
8	Eventos Básicos - Crea elementos interac...	ejercicios/ejercicio_08.js	Sí	5.0
9	Formularios y Validación Simple - Crea u...	ejercicios/ejercicio_09.js	Sí	4.0
10	Navegación entre Elementos - Crea una es...	ejercicios/ejercicio_10.js	Sí	5.0

Retroalimentación Detallada

Actividad 1: Información Básica del Documento - Crea una página HTML simple que muestre: Título de la página usando document.title, URL actual usando document.URL, Mostrar esta información en un div en la página

Archivo esperado: ejercicios/ejercicio_01.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Se podría mejorar utilizando template literals para concatenar strings de manera más legible y evitando la repetición de `div.innerHTML`.

Actividad 2: Seleccionar Elementos por ID - Crea una página con varios elementos que tengan ID y: Usa getElementById() para seleccionar elementos, Cambia el texto de los elementos seleccionados, Cambia el color de fondo de un elemento

Archivo esperado: ejercicios/ejercicio_02.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Falta cerrar correctamente la etiqueta h1 en la línea 2 (cambiar </h1> por </h1>) y considerar agregar un archivo HTML para una mejor demostración de la funcionalidad.

Actividad 3: Seleccionar Elementos por Clase - Crea elementos con la misma clase y: Usa getElementsByClassName() para seleccionarlos, Cambia el estilo de todos los elementos de esa clase, Cuenta cuántos elementos tienen esa clase

Archivo esperado: ejercicios/ejercicio_03.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es claro, funcional y resuelve correctamente el problema planteado. Se aplican buenas prácticas al iterar sobre la colección HTML.

Actividad 4: Cambiar Contenido de Elementos - Crea elementos con texto y: Usa textContent para cambiar el texto, Usa innerHTML para agregar HTML, Crea un botón que cambie el contenido al hacer clic

Archivo esperado: ejercicios/ejercicio_04.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La funcionalidad principal está implementada correctamente, pero falta la parte de la creación del botón y la vinculación de las funciones a él. Podrías mejorar la estructura del código separando la lógica de la manipulación del DOM.

Actividad 5: Modificar Atributos - Crea elementos con atributos y: Usa getAttribute() para leer atributos, Usa setAttribute() para cambiar atributos, Cambia el src de una imagen y el href de un enlace

Archivo esperado: ejercicios/ejercicio_05.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. El código cumple con los requisitos de la actividad y utiliza buenas prácticas al manipular los atributos de los elementos HTML.

Actividad 6: Agregar y Quitar Clases CSS - Crea elementos con estilos CSS y: Usa classList.add() para agregar clases, Usa classList.remove() para quitar clases, Usa classList.toggle() para alternar clases

Archivo esperado: ejercicios/ejercicio_06.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución es correcta y funcional. Se recomienda agregar validación para asegurar que el elemento 'texto' exista antes de manipular sus clases, para evitar errores en caso de que el elemento no se encuentre en el DOM.

Actividad 7: Crear y Agregar Elementos - Crea nuevos elementos dinámicamente: Usa createElement() para crear elementos, Usa appendChild() para agregarlos al DOM, Crea una lista de elementos con un botón

Archivo esperado: ejercicios/ejercicio_07.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La solución cumple con la funcionalidad descrita. Sin embargo, la variable `contador` debería declararse fuera de la función para mantener su valor entre llamadas y dentro de la función `agregarItem` debería actualizarse después de usar su valor actual. Podrías considerar usar `let` o `const` para declarar variables en lugar de `var` para mejorar la claridad y el alcance.

Actividad 8: Eventos Básicos - Crea elementos interactivos: Usa addEventListener() para eventos de click, Maneja eventos de mouseover y mouseout, Cambia elementos cuando ocurran los eventos

Archivo esperado: ejercicios/ejercicio_08.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

Excelente solución. El código es claro, conciso y cumple con todos los requisitos de la actividad. Bien aplicado el uso de addEventListener para los diferentes eventos.

Actividad 9: Formularios y Validación Simple - Crea un formulario simple y: Obtén valores de inputs con value, Valida que los campos no estén vacíos, Muestra mensajes de error o éxito

Archivo esperado: ejercicios/ejercicio_09.js

Estado: Archivo encontrado

Calificación: 4.0/5.0

Retroalimentación:

La validación básica funciona correctamente y muestra mensajes. Podrías mejorar la validación incluyendo un chequeo de formato de email y separando la lógica de actualización de la UI.

Actividad 10: Navegación entre Elementos - Crea una estructura HTML y demuestra: Usa parentElement para acceder al elemento padre, Usa children para acceder a elementos hijos, Usa nextElementSibling para el siguiente hermano, Muestra la información de navegación en la página

Archivo esperado: ejercicios/ejercicio_10.js

Estado: Archivo encontrado

Calificación: 5.0/5.0

Retroalimentación:

La solución es correcta y concisa. El código es legible y cumple con los requerimientos del ejercicio. Excelente trabajo.

Resumen General

Excelente trabajo. Completó 10/10 actividades (100%) con una calificación promedio de 4.4/5. Demuestra buen dominio de los conceptos.

Recomendaciones

- Continuar con el excelente trabajo y mantener la calidad del código