Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: Carolina Bolivar Rios
Repositorio: CarolinaBolivar5/act_web1_s9
Fecha de evaluación: 5/10/2025, 13:20:42
Evaluado por: Sistema de Evaluación Masiva

Resumen de Calificaciones

Calificación general: 4.8/5.0
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	5.0
2	Seleccionar Elementos por ID - Crea una ...	ejercicios/ejercicio_02.js	Sí	5.0
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	5.0
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	5.0
7	Crear y Agregar Elementos - Crea nuevos ...	ejercicios/ejercicio_07.js	Sí	5.0
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	4.0

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: 5.0/5.0

Retroalimentación:

Excelente solución. El código es conciso, funcional y cumple con los requisitos de la actividad utilizando correctamente las propiedades `document.title` y `document.URL`.

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: 5.0/5.0

Retroalimentación:

La solución es correcta y cumple con los requisitos de la actividad. El código es conciso y funcional, demostrando una comprensión clara de getElementById y la manipulación del DOM.

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:

La solución es correcta y eficiente. Utiliza `getElementsByClassName()` y un bucle for de manera adecuada para modificar los elementos y actualizar el contador.

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: 5.0/5.0

Retroalimentación:

Excelente solución. El código resuelve el problema de manera eficiente y clara, utilizando correctamente textContent e innerHTML para manipular el contenido del elemento. Bien hecho.

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. Cumple con todos los requisitos del ejercicio, utilizando correctamente getAttribute y setAttribute para modificar los atributos del enlace y la imagen.

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: 5.0/5.0

Retroalimentación:

La solución es correcta y completa. El código es limpio, fácil de entender y utiliza las funciones `add`, `remove` y `toggle` de `classList` según lo solicitado.

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: 5.0/5.0

Retroalimentación:

Excelente solución. El código es conciso, funcional y sigue las buenas prácticas para la creación y adición de elementos al DOM. El uso de un contador es una solución elegante.

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 conciso, claro y cumple con todos los requisitos de la actividad. Bien hecho.

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 solución es correcta y funcional. Sería bueno agregar validación del formato del correo electrónico para una solución más robusta.

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: 4.0/5.0

Retroalimentación:

La solución es correcta y cumple con la actividad. Sin embargo, sería mejor validar si `nextElementSibling` existe antes de acceder a `textContent` para evitar errores si es el último elemento.

Resumen General

Excelente trabajo. Completó 10/10 actividades (100%) con una calificación promedio de 4.8/5. Demuestra buen dominio de los conceptos.

Recomendaciones

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