Reporte de Evaluación - Fork de GitHub

Información General

Estudiante: paula andrea gil vargas
Repositorio: GGP113/act_web1_s9
Fecha de evaluación: 5/10/2025, 13:13:24
Evaluado por: Sistema de Evaluación Masiva

Resumen de Calificaciones

Calificación general: 4.0/5.0
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	3.0
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	4.0
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	4.0
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 funciona, pero sobreescribe el contenido del div en lugar de concatenarlo correctamente la primera vez. Se recomienda usar `+=` para añadir los valores al div en una sola operación.

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 cumple con los requisitos, pero no es necesario seleccionar todos los elementos con ID para luego iterar. Podrías acceder directamente a los elementos específicos que quieres modificar con `getElementById` para mejorar la eficiencia. Buen trabajo en general.

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: 3.0/5.0

Retroalimentación:

La solución intenta abordar el problema, pero la lógica para identificar clases repetidas es compleja e ineficiente. Además, solo cambia el estilo de las dos primeras clases encontradas y no generaliza para todas. Se recomienda simplificar la lógica de identificación y aplicar el estilo a todas las clases con el método `getElementsByClassName`.

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 actividad cumple con los requisitos. Sin embargo, la lógica para encontrar clases es innecesaria para la solución del problema planteado. Podrías mejorar la organización del código separando la manipulación del DOM en funciones más pequeñas.

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: 4.0/5.0

Retroalimentación:

La solución cumple con los requisitos del ejercicio. Podrías mejorar la generalización del código para que funcione con múltiples elementos en lugar de indexar directamente el primer elemento.

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 utilizar nombres más descriptivos para las variables y considerar la posibilidad de refactorizar para mayor legibilidad.

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 los requisitos principales, creando y agregando elementos al DOM. Considera agregar el botón como se indica en la descripción para una solución más completa y mejorar la organización del código, separando la lógica de la presentación (CSS en un archivo aparte o clases).

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: 4.0/5.0

Retroalimentación:

La solución cumple con los requisitos de la actividad. Podrías mejorar la estructura del código separando la lógica en funciones para mayor claridad.

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 cumple con los requisitos de la actividad, validando la existencia de valores en los campos del formulario. Se sugiere mejorar la forma en que se muestra el mensaje al usuario (usar un elemento existente en el HTML en vez de crear uno nuevo cada vez) y usar un enfoque más declarativo en la validación (ej: `campos.every(valor => valor)`).

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:

Excelente solución. Demuestra correctamente la navegación entre elementos utilizando parentElement, children y nextElementSibling. El código es claro, conciso y funcional.

Resumen General

Excelente trabajo. Completó 10/10 actividades (100%) con una calificación promedio de 4.0/5. Demuestra buen dominio de los conceptos.

Recomendaciones

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