**Event Listener**

**👋 Introducción**

¡Hola! 👋 ¡Bienvenido al fascinante mundo de los Event Listeners en JavaScript! Este módulo te sumergirá en el poderoso concepto de escuchar y responder a las acciones del usuario en tus aplicaciones web. Los Event Listeners son esenciales para crear experiencias interactivas, permitiéndote llevar tus proyectos al siguiente nivel.

Durante este módulo, explorarás cómo los Event Listeners actúan como puentes entre el usuario y tu aplicación, capturando eventos como clics, teclas presionadas y movimientos del mouse. Profundizarás en la comprensión de cómo estos eventos desencadenan respuestas específicas, proporcionando una interactividad dinámica que mejora significativamente la experiencia del usuario.

**📖 Material Descargable**

Aquí es donde encontrarás el material descargable que complementará los ejercicios prácticos que vamos a ver en los siguientes pasos.

👉[**Teoría Event Listener**](https://drive.google.com/file/d/1uIqSWcT_C7x6mv9roA7NtEBoRqJbKxDs/view?usp=drive_link)

**⏭️ En el paso siguiente** veremos cómo manejar los eventos en javascript.

# Practicando con Event Listener

A continuación comenzaremos con las actividades que te ayudarán a manejar eventos en javascript.

## ✏️Actividad Controlador de eventos

### 1. Generación de otra etiqueta <script></script>

Haga otra etiqueta <script></script>, aparte de la que ya tenía. Puede ir antes o después de la existente.

### 2. Añadir event listener

Pegue el siguiente código dentro de la etiqueta <script></script>:

document.addEventListener("DOMContentLoaded", (event) => {

    console.log(event);

});

1. **document.addEventListener**: Este código agrega un controlador de eventos al objeto document, que representa toda la página web. El evento en cuestión (DOMContentLoaded) se ejecuta cuando el documento HTML inicial ha sido completamente cargado y analizado.
2. **event:** El segundo parámetro es una función anónima que registra el objeto event en la consola. Este objeto contiene información sobre el evento.
3. **F12:** Abra la consola de desarrollador (F12) y verifique el contenido de la variable event

### 3. document.addEventLister("click")

document.addEventListener("click", (event) => {

    console.log(event);

    const bodyColor = document.body.style.background;

    document.body.style.background =

        bodyColor === "white" ? "red" : "white";

});

#### 1. Añadir un Controlador de Eventos:

* **document.addEventListener("click", (event) => {...}):** El evento que se está escuchando es el click. Cada vez que se hace clic en cualquier lugar de la página, se ejecuta la función anónima proporcionada.

#### 2. Función del Controlador de Eventos:

* **console.log(event)**: Registra el objeto event en la consola. Este objeto contiene información sobre el evento de clic, como la ubicación del mismo, el elemento que fue clickeado, etc.
* **const bodyColor = document.body.style.background**: Obtiene el color actual de fondo del body de la página y lo almacena en la variable bodyColor.

#### 3. Cambio de Color de Fondo:

* Luego, se cambia el color de fondo del body de la página con esta línea:

document.body.style.background = bodyColor === "white" ? "red" : "white";

* Esta es una expresión condicional (ternaria). Si bodyColor es igual a "white", entonces se cambia el color de fondo a "red". Si no, se cambia a "white".
* En resumen, este script hace que el fondo de la página web cambie entre rojo y blanco con cada clic, demostrando una manera interactiva y dinámica de modificar el estilo de un elemento en respuesta a eventos del usuario.

### 🧰Material Complementario

Si deseas profundizar más en lo aprendido en este paso, te recomendamos que visites el siguiente recurso:

* [**Variable event**](https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener#the_event_listener_callback)

**Change Event**

En esta sección nos enfocaremos en el evento change de cada producto (modificación del stock), para ir actualizando el total del pedido.

El evento change se utiliza comúnmente en conjunción con Event Listeners para detectar cambios en el valor de un elemento de formulario, como un campo de entrada (<input>), un elemento de selección (<select>), o un área de texto (<textarea>).

Cuando un usuario interactúa con un elemento de formulario y realiza algún cambio, como seleccionar una opción diferente en un menú desplegable o ingresar texto en un campo de entrada, se dispara el evento change. Este evento captura la acción de cambio y proporciona una oportunidad para ejecutar funciones o realizar acciones específicas en respuesta a dicho cambio.

**✏️Actividad Validación y cálculo del precio**

**1- Añadir el addEventListener**

En la función populate, dentro del bucle del carrito, hay una variable que representa la cantidad de ese producto que existe en el carrito.

La variable se llama productInput.

Agrega el siguiente listener después de declararla. Éste es el elemento al cual le vamos a aplicar el listener.

//Start Código existente

const productInput = document.createElement("input");

productInput.type = "number";

productInput.value = product.quantity;

productInput.min = 1;

//End Código existente

// Agregar `addEventListener`

productInput.addEventListener("change", (event) => {

    const value = event.target.value;

    const index = cart.findIndex(

        ({ title }) => product.title === title

    );

    if (index < 0) return console.log("Error");

    cart[index].quantity = value;

    renderTotalPrice();

});

**2-  Asignación de un Controlador de Eventos:**

* **productInput.addEventListener("change", (event) => {...}):** Este código añade un controlador de eventos al elemento productInput. El evento que se está escuchando es el change, que ocurre, por ejemplo, cuando un usuario cambia el valor en un campo de entrada (input) y luego sale del campo (en este caso es un campo numérico para la cantidad del producto).

**3- Dentro del Controlador de Eventos:**

* **const value = event.target.value:** Cuando se produce un cambio, event.target se refiere al elemento específico que desencadenó el evento, y value es el valor actual del elemento de entrada después del cambio.

**4- Buscar el Producto en el Carrito:**

* **const index = cart.findIndex(({ title }) => product.title === title);:** Esta línea busca en el array cart para encontrar el índice del producto que se está modificando. Utiliza findIndex para buscar en cada objeto del carrito (cart) y compara el título del producto actual (product.title) con cada título en el carrito. El índice del producto correspondiente se almacena en index.

**5- Validación y Actualización de la Cantidad:**

* Si index < 0, significa que el producto no se encontró en el carrito, y en este caso, se registra un error en la consola y se sale de la función.
* Si se encuentra el producto (index >= 0), se actualiza la propiedad quantity del producto en el carrito con el nuevo valor obtenido del campo de entrada.

Nótese que al final se llama a la función renderTotalPrice. Esto es simplemente mover la lógica del renderizado del total a una función y llamarla. Quedaría de la siguiente manera:

function renderTotalPrice() {

    const priceElement = document.querySelector(".resume .price");

    priceElement.textContent = formatPrice(

        cart.reduce(

        (total, { price, quantity }) => total + price \* quantity,

        0

        )

    );

}

el código completo quedaría similar a esto:

const cart = [...]

document.addEventListener("DOMContentLoaded", populate);

function populate() {

  const cartContainer = document.querySelector(".cart-container");

  document.querySelector(".products")?.remove();

  if (!cartContainer) return console.log("Error");

  const productContainer = document.createElement("div");

  productContainer.classList.add("card", "products");

  cart.forEach((product) => {

    const productElement = document.createElement("article");

    productElement.classList.add("product-cart");

    const productImage = document.createElement("img");

    productImage.src = product.image;

    productImage.alt = product.title;

    const productMiddle = document.createElement("div");

    productMiddle.classList.add("middle");

    const productInfo = document.createElement("div");

    productInfo.classList.add("info");

    const productTitle = document.createElement("strong");

    productTitle.classList.add("title");

    productTitle.textContent = product.title;

    const productDescription = document.createElement("span");

    productDescription.classList.add("description");

    productDescription.textContent = product.description;

    const productInput = document.createElement("input");

    productInput.type = "number";

    productInput.value = product.quantity;

    productInput.min = 1;

    productInput.addEventListener("change", (event) => {

      const value = event.target.value;

      const index = cart.findIndex(({ title }) => product.title === title);

      if (index < 0) return console.log("Error");

      cart[index].quantity = value;

      renderTotalPrice();

    });

    const productPrice = document.createElement("strong");

    productPrice.classList.add("price");

    productPrice.textContent = `AR$ ${formatPrice(product.price)}`;

    productInfo.appendChild(productTitle);

    productInfo.appendChild(productDescription);

    productMiddle.appendChild(productInfo);

    productMiddle.appendChild(productInput);

    productElement.appendChild(productImage);

    productElement.appendChild(productMiddle);

    productElement.appendChild(productPrice);

    productContainer.appendChild(productElement);

  });

  cartContainer.insertAdjacentElement("afterbegin", productContainer);

  renderTotalPrice();

}

function renderTotalPrice() {

  const priceElement = document.querySelector(".resume .price");

  priceElement.textContent = formatPrice(

    cart.reduce((total, { price, quantity }) => total + price \* quantity, 0)

  );

}

function formatPrice(price) {

  return new Intl.NumberFormat("es-AR", {

    style: "currency",

    currency: "ARS",

  }).format(price);

}

**⏭️ En el paso siguiente** podrás centrarte en el comportamiento de un evento muy utilizado, el evento “click”.