



```
var colorCorrecto=false;

do{
var color=prompt("Qué color quieres (R: rojo, V:verde, A:azul?");

if(color=='R'){
    color="red";
    document.getElementById("fondo").style.backgroundColor=color;
    colorCorrecto=true;
}

if(color=='V'){
    color="green";
    document.getElementById("fondo").style.backgroundColor=color;
    colorCorrecto=true;
}

if(color=='A'){
    color="blue";
    document.getElementById("fondo").style.backgroundColor=color;
    colorCorrecto=true;
}
}while(colorCorrecto!=true); dime por qué el fondo de la web no cambia de color
```

El fondo de la web **no** cambia de color** porque en tu código, la función `document.getEl`

Explicación detallada

- El `<div id="fondo"></div>` que tienes en tu HTML está vacío y, por defecto, su altura
- Lo que ves como "fondo blanco" es el fondo del `<body>`, no del `<div>`.

Soluciones prácticas

Opción 1: Cambiar el fondo del <body>

```
if(color=='R'){
    color="red";
    document.body.style.backgroundColor=color;
    colorCorrecto=true;
}
if(color=='V'){
    color="green";
    document.body.style.backgroundColor=color;
    colorCorrecto=true;
}
if(color=='A'){
    color="blue";
    document.body.style.backgroundColor=color;
    colorCorrecto=true;
}
```

De este modo, cambiarás **el fondo de toda la web**, y sí lo verás inmediatamente.

Opción 2: Dar tamaño al div

Si quieres cambiar solo el <div id="fondo">, añade CSS para que tenga anchura y altura:

```
#fondo {
    width: 100vw;
    height: 100vh;
}
```

Así, el elemento fondo ocupará toda la ventana y el color será visible.

Resumen: Tu código funciona, pero no es visible porque el div es invisible por defecto. Usa `document.body` para el fondo global, o dale tamaño al div si prefieres usarlo.

El comportamiento descrito se debe a los estilos CSS por defecto y la naturaleza del DOM.

Uncaught Error

```
at <anonymous> (c:\Users\espej\OneDrive\Documentos\FP\3AÑO\ENTORNO CLIENTE\color.js:10:26)
```

```
var colorCorrecto=false;
```

```
do{
```

```
var color=prompt("Qué color quieres (R: rojo, V:verde, A:azul?");
```

```
if(color=='R'){
```

```
    color="red";
```

```
    document.boody.style.backgroundColor=color;
```

```

    colorCorrecto=true;
}

if(color=='V'){
    color="green";
    document.body.style.backgroundColor=color;
    colorCorrecto=true;
}

if(color=='A'){
    color="blue";
    document.body.style.backgroundColor=color;
    colorCorrecto=true;
}
}while(colorCorrecto!=true);

```

El error en tu código se debe a un **error tipográfico**: escribiste `document.boody` en vez de `document.body`. La propiedad correcta es `document.body`.

Explicación del error

- JavaScript no reconoce `document.boody` porque **la propiedad "boody" no existe** en el objeto `document`.
- Esto causa un error tipo "Uncaught Error" cuando intentas asignar el color:

```
document.boody.style.backgroundColor=color; // ← aquí está el error
```

Solución

Corrige la línea que tienes mal para que ponga `body` y no `boody`:

```

if(color=='R'){
    color="red";
    document.body.style.backgroundColor=color;
    colorCorrecto=true;
}

```

Código corregido

```

var colorCorrecto=false;

do{
    var color=prompt("Qué color quieres (R: rojo, V:verde, A:azul?)");

    if(color=='R'){
        color="red";
        document.body.style.backgroundColor=color;
        colorCorrecto=true;
    }
}

```

```

    if(color=='V'){
        color="green";
        document.body.style.backgroundColor=color;
        colorCorrecto=true;
    }
    if(color=='A'){
        color="blue";
        document.body.style.backgroundColor=color;
        colorCorrecto=true;
    }
}while(colorCorrecto!=true);

```

Con esto ya no tendrás el error y el fondo se cambiará correctamente.

El error se produce cuando se accede a una propiedad inexistente en JavaScript, especialmente en el DOM.

cómo se crea un botón en h tml

Para crear un botón en HTML se utiliza la etiqueta `<button>`. Es muy sencillo y el botón aparece visible en la página con el texto que pongas entre las etiquetas de apertura y cierre.^[1] ^[2]

Ejemplo de botón básico

```
<button>Haz clic aquí</button>
```

Esto muestra un botón en la web con el texto "Haz clic aquí".^[2] ^[1]

Otros métodos

- También se puede usar `<input type="button" value="Texto">`, pero el método recomendado

- Para que el botón haga algo al hacer clic, necesitas JavaScript. Por ejemplo:

```
<button onclick="alert('¡Hola!')">Saludar</button>
```

El contenido entre `<button>` y `</button>` es el texto del botón y puede ser cualquier cosa.



PONME UN EJEMPLO de una funcion en javascript a la que se le da un parámetro de entrada

Aquí tienes un ejemplo sencillo de una función en JavaScript a la que se le da un parámetro de entrada:

```
function saludar(nombre) {  
    console.log("Hola, " + nombre + "!");  
}  
  
// Llamada a la función con un parámetro  
saludar("Juan"); // Imprime: Hola, Juan!
```

En este ejemplo, `nombre` es el parámetro de entrada que recibe la función `saludar`. Al llamar a la función con `"Juan"`, el mensaje se personaliza con ese valor. [\[9\]](#) [\[10\]](#)

Otro ejemplo, usando `return`:

```
function cuadrado(numero) {  
    return numero * numero;  
}  
  
var resultado = cuadrado(5); // resultado es 25
```

Aquí, la función `cuadrado` recibe un parámetro `numero` y devuelve el cuadrado de ese valor. [\[9\]](#)



Explícame este elemento y usos `document.getElementById`

`document.getElementById` es un método en JavaScript que permite buscar y obtener un elemento HTML de la página mediante su atributo único `id`. Devuelve el elemento correspondiente para poder manipularlo desde JavaScript.

Cómo funciona

- Recibe como parámetro una cadena (string) con el valor del `id` que tiene el elemento en el HTML.
- Retorna el objeto DOM del elemento con ese `id`.
- Si no encuentra ningún elemento con ese `id`, devuelve `null`.
- El `id` debe ser único dentro del documento HTML; no debe haber dos elementos con el mismo `id`.

Ejemplo básico

HTML:

```
<p id="mensaje">Texto inicial</p>
<button onclick="cambiarTexto()">Cambiar texto</button>
```

JavaScript:

```
function cambiarTexto() {
  var elemento = document.getElementById("mensaje");
  elemento.textContent = "El texto ha cambiado usando getElementById";
}
```

Al pulsar el botón, la función obtiene el párrafo con `id="mensaje"` y cambia su texto. [\[19\]](#) [\[20\]](#) [\[21\]](#)

Usos comunes

- Cambiar estilos: `elemento.style.color = "red";`

- Modificar contenido: ``elemento.innerHTML = "Nuevo contenido";``

- Obtener valores o atributos.
- Vincular eventos a botones u otros elementos.

Nota importante

Solo funciona a nivel de `document` (no existe `element.getElementById`) porque el `id` es único para toda la página.

En resumen, `document.getElementById` es clave para acceder rápidamente a un elemento HTML especificado para modificarlo o interactuar con él mediante JavaScript. [\[21\]](#) [\[22\]](#) [\[19\]](#)



`document.getElementById("botonR").addEventListener("click",function(event){`

`cambiarColor('ROJO');`
`});` por qué ponemos `function(event)`? por qué no es `function` y el parámetro de entrada de la función?

Se pone `function(event)` porque la función que se pasa como segundo argumento en `addEventListener` es un **manejador de evento** que recibe automáticamente un objeto `event` cuando se dispara el evento. Este objeto `event` contiene información sobre el evento que ocurrió (por ejemplo, qué botón se pulsó, la posición del ratón, etc.).

Por qué se escribe `function(event)`

- `addEventListener` llama a la función cuando sucede el evento, y automáticamente **le pasa el objeto del evento como argumento**.
- Ese parámetro `event` es opcional en la función, pero si quieres acceder a detalles del evento deberías incluirlo.
- No se pone solo `function` sin paréntesis porque la sintaxis de función en JavaScript requiere definir paréntesis para los parámetros, incluso si están vacíos.

Ejemplo para comparar

```
// Función que recibe el evento
document.getElementById("botonR").addEventListener("click", function(event) {
    console.log(event.type); // Muestra 'click'
    cambiarColor('ROJO');
});

// Función sin usar el parámetro event
document.getElementById("botonR").addEventListener("click", function() {
    cambiarColor('ROJO');
});
```

Ambas son correctas, pero si quieres acceder a información del evento, debes poner `event` como parámetro.

En resumen

- `function(event)` define que la función espera un parámetro, que es el objeto evento.
- No se pone simplemente `function` porque hace falta declarar los paréntesis, y si necesitas el objeto evento, debes declararlo dentro.
- El evento es pasado automáticamente y ofrece detalles internos del evento que disparó la función. [\[29\]](#) [\[30\]](#) [\[31\]](#)

¿Quieres un ejemplo concreto usando el parámetro `event`?



`colorEscogido=document.getElementById("boton");`

```
function cambiarColor(colorEscogido){
```

```
var colorEscogido;
```

```
if(colorEscogido=='ROJO'){
    color="red";
    document.body.style.backgroundColor=color;
```

```

    }

    if(colorEscogido=='VERDE'){
        color="green";
        document.body.style.backgroundColor=color;

    }

    if(colorEscogido=='AZUL'){
        color="blue";
        document.body.style.backgroundColor=color;

    }

}

document.getElementById("botonR").addEventListener("click",function(event){
    cambiarColor('ROJO');
});

document.getElementById("botonV").addEventListener("click",function(event){
    cambiarColor('VERDE');
});

document.getElementById("botonA").addEventListener("click",function(event){
    cambiarColor('AZUL');
});

```

ESCOGE TU COLOR

ROJO

VERDE

AZUL

analiza este código explícame cómo se comunican ambos, y cómo sabe js qué botón pertenece a cada color, y que botonR es Red, etc explícamelo como si fuera una estudiante de bachillerato

Este código HTML y JavaScript trabajan juntos para permitir al usuario escoger un color (rojo, verde o azul) pulsando un botón, y cambiar el color de fondo de la página en función del botón pulsado.

Cómo se comunican HTML y JavaScript

- En el HTML hay tres botones: cada uno tiene un atributo id diferente: botonR, botonV y botonA. Estos identificadores únicos permiten a JavaScript seleccionar cada botón fácilmente.

- El archivo JavaScript `color2.js` (que se carga desde la etiqueta `<script>`) contiene la lógica que "escucha" cuándo se hace clic en esos botones y cambia el color.

Cómo sabe JavaScript qué botón es cada color

- Usando `document.getElementById("botonR")`, JavaScript recupera el botón con id `"botonR"`, que corresponde al botón rojo.
- Luego, a ese botón se le añade un "escuchador" o "listener" usando `.addEventListener("click", function(event) { ... })`. Esto significa que cuando se haga clic en ese botón, se ejecutará la función que cambia el color a `'ROJO'`.
- Lo mismo ocurre con los botones `"botonV"` (verde) y `"botonA"` (azul), que ejecutan `cambiarColor('VERDE')` y `cambiarColor('AZUL')` respectivamente.

Cómo funciona la función `cambiarColor`

- `cambiarColor` recibe una cadena con el color deseado (por ejemplo, `'ROJO'`).
- Según el valor recibido, cambia la propiedad `document.body.style.backgroundColor` a `"red"`, `"green"` o `"blue"`, que es el color de fondo del `<body>`.
- Esto hace que todo el fondo de la página cambie a ese color.

Ejemplo sencillo para explicar el flujo

1. El usuario ve 3 botones en pantalla: ROJO, VERDE, AZUL.
2. El usuario pulsa el botón que dice "ROJO".
3. JavaScript detecta el clic en el botón con id `"botonR"`.
4. Se ejecuta la función anónima que llama a `cambiarColor('ROJO')`.
5. La función `cambiarColor` cambia el fondo de la página a rojo.
6. La pantalla muestra la página con fondo rojo.

En resumen

- El enlace entre HTML y JS es el atributo `id` en los botones y el uso de `document.getElementById` para seleccionarlos.
- Los eventos `"click"` detectan cuando el usuario pulsa un botón.
- Al evento le sigue la llamada a una función que recibe un parámetro (`"ROJO"`, `"VERDE"`, `"AZUL"`) para definir qué color poner de fondo.
- Así JavaScript "sabe" qué botón corresponde a cada color gracias a los `id` distintivos y la función que pasa el color como parámetro. [\[39\]](#) [\[40\]](#) [\[41\]](#)

¿Quieres ayuda para mejorar o modificar este programa?



colorEscogido=document.getElementById("boton"); esta frase hace algo en el código?

La línea `colorEscogido = document.getElementById("boton");` en tu código hace lo siguiente:

- Usa `document.getElementById("boton")` para buscar en el documento HTML un elemento que tenga el atributo `id="boton"`.
- Si existe un elemento con ese id, esa línea guarda una referencia a ese elemento en la variable `colorEscogido`.
- Si no hay ningún elemento con id `"boton"` en el HTML, esa variable quedará con valor `null`.

En tu código, esta línea

```
colorEscogido = document.getElementById("boton");
```

no hace nada visible ni útil porque:

- En el HTML que mostraste no existe ningún elemento con `id="boton"`, los botones tienen ids `botonR`, `botonV` y `botonA`.
- Entonces `colorEscogido` quedará con valor `null` y no se usa después en tu script.
- Además, dentro de la función `cambiarColor` tienes declarado otra variable con el mismo nombre, lo que crea confusión y no usa el valor obtenido fuera de la función.

Conclusión

Esta línea no tiene efecto práctico porque hace referencia a un id inexistente y no se usa en ningún otro lugar de tu código. Sería mejor eliminarla para evitar confusión.

Para seleccionar los botones que tienes, debes usar sus ids reales, por ejemplo:

```
var botonRojo = document.getElementById("botonR");
```

y así con cada botón.

En resumen, la línea `colorEscogido=document.getElementById("boton");` no hace nada en tu caso, no aporta nada ni afecta la funcionalidad del programa porque no existe ese id en el HTML y la variable no se usa luego. [\[49\]](#) [\[50\]](#) [\[51\]](#)



Entiendo que le introduzco el parámetro, porque primero se ejecuta lo que va a abajo de la función y luego la función es así?

La ejecución funciona de la siguiente forma:

- Primero el navegador carga todo el código JavaScript línea a línea, de arriba hacia abajo.
- Cuando llega a esta instrucción

```
document.getElementById("botonR").addEventListener("click", function(event) {
  cambiarColor('ROJO');
});
```

no ejecuta inmediatamente la función `cambiarColor`.

- En vez de eso, esta línea **asocia un "escuchador" (listener)** para el evento "click" en el botón con id "botonR".
- Eso significa que la función solo se ejecutará **cuando se haga realmente clic en ese botón**.
- Por lo tanto, no importa que esta línea esté antes o después de la definición de la función `cambiarColor` en el código.
- Cuando el usuario pulse el botón, en ese momento se llama a `cambiarColor('ROJO')` con ese parámetro.

En resumen, el parámetro se pasa **en el momento de ejecutar la función al hacer clic**, no cuando se carga el código. El código carga las funciones y asigna qué hacer cuando sucede un evento, y solo en ese momento (al clic) la función se ejecuta con el parámetro que se le pasa explícitamente. [\[59\]](#) [\[60\]](#) [\[61\]](#)

¿Quieres ejemplo paso a paso visual?



ya pero cuando hace click primero se ejecuta esa línea, y luego que ya sabemos cuáles el parámetro de entrada se ejecuta la función

Exacto, cuando haces clic en el botón, primero se ejecuta la línea que tiene el `addEventListener` relacionado con ese botón, pero en realidad, esa línea ya había sido ejecutada cuando se cargó el código para "escuchar" el evento.

Lo que sucede en el momento del clic es:

1. **El navegador detecta el clic** y busca qué funciones están escuchando ese evento ("click") en ese botón.
2. Se ejecuta la función que pasaste al `addEventListener`, es decir, la función anónima `function(event) { cambiarColor('ROJO'); }`.

3. En esa función, se llama a `cambiarColor('ROJO')`, pasando 'ROJO' como parámetro.
4. La función `cambiarColor` recibe ese parámetro, sabe qué color usar, y cambia el fondo.

Por tanto, el orden es:

- El código ya tiene registrado a qué botones escucha.
- Cuando haces clic, se **dispara la ejecución**, donde se llama a la función con el parámetro que definiste.
- La función recibe ese parámetro y actúa según él.

No es que primero se ejecute la línea del `addEventListener` en el clic, sino que esa línea ya existía, y en el clic solo se llama la función que tú has asociado con ese evento pasando el parámetro. [\[69\]](#) [\[70\]](#) [\[71\]](#)

¿Quieres que te lo explique con un esquema o paso a paso visual?



la variables `var colorEscogido`; que he puesto dentro de la función hace algo? Debería estar fuera, puedo no declarar esa variables?

Declarar la variable `var colorEscogido`; dentro de la función hace que esa variable sea una **variable local** de la función. Esto significa que:

- Solo existe y se puede usar dentro de esa función.
- Cada vez que se llama la función, se crea una nueva variable local `colorEscogido`.
- No afecta ni puede usarse fuera de la función.

¿Debería estar fuera de la función?

No, en tu caso **no es necesario declararla dentro de la función** porque:

- La función `cambiarColor` ya recibe un parámetro que se llama `colorEscogido`.
- La línea `var colorEscogido`; dentro de la función crea una variable local nueva y vacía que enmascara (oculta) el parámetro que recibes.
- Esto puede causar confusión o errores porque estás **sobrescribiendo el parámetro** con una variable local sin valor.

¿Puedes no declararla?

- Si usas el parámetro que te pasan, no necesitas declarar otra variable con el mismo nombre.
- Puedes simplemente eliminar `var colorEscogido`; dentro de la función para usar directamente el parámetro que entra.

Ejemplo correcto

```
function cambiarColor(colorEscogido){
  if(colorEscogido=='ROJO'){
    color="red";
    document.body.style.backgroundColor=color;
  }
  // resto del código...
}
```

Resumen

- Declarar `var colorEscogido`; dentro de la función crea una variable local que oculta el parámetro que entra.
- No es necesario ni recomendable redeclarar esa variable dentro de la función si ya es parámetro.
- Lo ideal es usar el parámetro directamente sin redeclararlo, para evitar confusión y errores. [\[79\]](#) [\[80\]](#) [\[81\]](#)

¿Quieres que te ayude a corregir el código para que sea más limpio?



Entonces aunque le meto el parámetro a la función, no lo declaro en ningún sitio, excepto en los `addEventListener` donde le especifico que al pinchar en `botonR` me aplique 'Rojo' y y que rojo es el valor escogido, es así?

Exacto, el parámetro que recibe la función `cambiarColor` no se declara en ningún sitio global, sino que se le pasa directamente dentro del `addEventListener` en el momento en que el evento ocurre.

Así funciona:

- En el código tienes algo como:

```
document.getElementById("botonR").addEventListener("click", function(event){
  cambiarColor('ROJO');
});
```

- Aquí cuando el botón con id `botonR` es clicado, se ejecuta esa función anónima.
- Esa función anónima llama a `cambiarColor` y le pasa la cadena `'ROJO'` como parámetro.
- Dentro de `cambiarColor(colorEscogido)`, el parámetro `colorEscogido` tendrá el valor `'ROJO'`.

- No es necesario declarar la variable `colorEscogido` en otro sitio, porque se recibe directamente como argumento al ejecutar la función.

Así que sí, el parámetro lo defines explícitamente justo en el momento de la llamada dentro del `addEventListener`, y es la forma correcta de pasarle valores a la función usada en un evento. [\[89\]](#) [\[90\]](#) [\[91\]](#)

Si no pasaras el parámetro aquí, la función no sabría qué color usar.

¿Quieres que te prepare un ejemplo simple con comentarios para mostrar esto?

✱

addEventListener hazme un listado de los elementos más comunes como click y qu

Aquí tienes un listado de los eventos más comunes que se usan con `addEventListener` en JavaScript:

- `click`: Se dispara cuando el usuario hace clic en un elemento (botón, enlace, etc.).
- `dblclick`: Se dispara cuando el usuario hace doble clic en un elemento.
- `mouseover`: Cuando el puntero del ratón entra en un elemento.
- `mouseout`: Cuando el puntero del ratón sale de un elemento.
- `mousedown`: Cuando se presiona el botón del ratón sobre un elemento.
- `mouseup`: Cuando se suelta el botón del ratón en un elemento.
- `keydown`: Cuando se presiona una tecla del teclado.
- `keyup`: Cuando se suelta una tecla del teclado.
- `keypress`: Cuando se presiona y se mantiene una tecla (menos usado hoy).
- `focus`: Cuando un elemento gana el foco (ej. un input).
- `blur`: Cuando un elemento pierde el foco.
- `load`: Cuando un documento o recurso (imagen, script) termina de cargar.
- `unload`: Cuando el documento o ventana se cierra o cambia.
- `scroll`: Cuando se hace scroll en la página u otro elemento.
- `change`: Cuando cambia el valor de un elemento de formulario (input, select).
- `submit`: Cuando se envía un formulario.
- `input`: Cuando cambia el valor de un input en tiempo real.
- `resize`: Cuando se cambia el tamaño de la ventana (window).

Estos son algunos de los eventos más frecuentes y útiles para interactuar con el usuario y la página web. [\[99\]](#) [\[100\]](#) [\[101\]](#) [\[102\]](#)

¿Quieres ejemplos de alguno en particular?



1. <https://developer.mozilla.org/es/docs/Web/HTML/Reference/Elements/button>
2. https://developer.mozilla.org/es/docs/Learn_web_development/Core/Structuring_content/HTML_forms
3. <https://www.youtube.com/watch?v=zbdtdgPqQA4>
4. <https://mailchimp.com/es/resources/html-button-as-links/>
5. <https://www.youtube.com/watch?v=L5JtpnmyMcc>
6. <https://www.youtube.com/watch?v=jgEa1E74oqs>
7. <https://www.freecodecamp.org/espanol/news/ejemplos-de-codigo-de-enlaces-de-botones-html-como-crear-hipervinculos-html-usando-el-atributo-href-en-las-etiquetas/>
8. https://www.reddit.com/r/learnprogramming/comments/rmtysh/programming_trick_1_how_to_make_html_buttons_links/
9. <https://es.javascript.info/function-basics>
10. <https://basicos.codica.la/languages/javascript/lessons/define-functions-parameters>
11. <https://lenguajejs.com/fundamentos/funciones/parametros/>
12. <https://didacticode.com/curso/curso-javascript/leccion/javascript-1-17-funciones-parametros-argumentos/>
13. <https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Functions>
14. https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Functions/Default_parameters
15. <https://www.tutorialesprogramacionya.com/javascriptya/temarios/descripcion.php?cod=16&inicio=>
16. <https://www.youtube.com/watch?v=e3EyqGnb6XM>
17. https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=800%3Afunciones-javascript-concepto-parametros-argumentos-y-tipos-paso-por-valor-return-ejemplo-cu01122e&catid=78&Itemid=206
18. <https://es.stackoverflow.com/questions/614088/cómo-puedo-especificar-el-tipo-de-dato-necesario-de-un-parámetro-de-entrada-par>
19. <https://developer.mozilla.org/es/docs/Web/API/Document/getElementById>
20. <https://www.digitallearning.es/intro-programacion-js/getelementbyid.html>
21. <https://keepcoding.io/blog/nodo-getelementbyid/>
22. https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=805%3Ajavascript-acceder-a-elementos-por-id-getelementbyid-documentall-cambiar-imagen-img-src-cu01127e&catid=78&Itemid=206
23. <https://www.youtube.com/watch?v=9RrhCXtBiwA>
24. <https://lenguajejs.com/dom/buscar/getelementbyid/>
25. http://www.codexemplar.org/curso/curso_4_3_a.php
26. <https://www.esthersola.com/getelementbyelementid-queryselector/>
27. <https://www.youtube.com/watch?v=h9Z66loFZdY>
28. <https://www.arsys.es/blog/dom-document-object-model-que-es-y-como-funciona-en-javascript>
29. <https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener>
30. https://www.w3schools.com/js/js_htmlDOM_eventlistener.asp
31. <https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener>

32. <https://stackoverflow.com/questions/256754/how-to-pass-arguments-to-addeventlistener-listener-function>
33. <https://plainenglish.io/blog/passing-arguments-to-event-listeners-in-javascript-1a81bc397ecb>
34. <https://forum.freecodecamp.org/t/creating-an-event-listener-with-custom-parameters/476014>
35. https://www.reddit.com/r/learnjavascript/comments/twljtu/how_do_i_pass_arguments_to_a_function_inside/
36. <https://www.smashingmagazine.com/2025/07/handling-javascript-event-listeners-parameters/>
37. <https://discuss.codecademy.com/t/javascript-event-listeners-arguments/747768>
38. https://www.w3schools.com/js/tryit.asp?filename=tryjs_addeventlistener_parameters
39. <https://es.stackoverflow.com/questions/351312/cambiar-background-color-según-value-html-con-javascript>
40. https://www.w3schools.com/jsref/prop_style_backgroundcolor.asp
41. <https://www.freecodecamp.org/news/how-to-change-background-color-with-javascript/>
42. <https://developer.mozilla.org/es/docs/Web/API/Document/bgColor>
43. <https://stackoverflow.com/questions/70146994/how-to-change-in-javascript-background-color-from-css>
44. <https://developer.mozilla.org/es/docs/Web/CSS/background-color>
45. <https://htmlmasters.tech/cambiar-background-color-javascript/>
46. <https://www.youtube.com/watch?v=NgXLaC6VNjo>
47. <https://www.forosdelweb.com/f13/cambiar-backgroundcolor-del-body-1110926/>
48. <https://developer.mozilla.org/es/docs/Web/API/HTMLStyleElement>
49. <https://developer.mozilla.org/es/docs/Web/API/Document/getElementById>
50. https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=805%3Ajavascript-acceder-a-elementos-por-id-getelementbyid-documentall-cambiar-imagen-img-src-cu01127e&catid=78&Itemid=206
51. <https://www.luisllamas.es/seleccionar-elementos-del-dom-javascript/>
52. <https://lenguajejs.com/dom/buscar/getelementbyid/>
53. http://www.codexemplar.org/curso/curso_4_3_a.php
54. <https://www.youtube.com/watch?v=h9Z66loFZdY>
55. <https://www.esthersola.com/getelementbyelementid-queryselector/>
56. <https://www.digitallearning.es/intro-programacion-js/getelementbyid.html>
57. <https://www.arsys.es/blog/dom-document-object-model-que-es-y-como-funciona-en-javascript>
58. <https://www.youtube.com/watch?v=9RrhCXtBiwA>
59. http://www.mywonderland.es/curso_js/addevent/index.html
60. <https://lenguajejs.com/eventos/introduccion/addeventlistener/>
61. https://developer.mozilla.org/es/docs/Learn_web_development/Core/Scripting/Events
62. <https://es.stackoverflow.com/questions/373691/se-puede-definir-el-orden-de-ejecución-de-eventos>
63. <https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener>
64. http://www.codexemplar.org/curso/curso_4_3_e.php?mc=false
65. <https://sospnt.com/blog/191-eventos-con-javascript>

66. <https://apuntes.de/javascript-programacion-asincrona/eventos-del-dom-tipos-de-eventos-y-manejadores/>
67. <https://kaboomeventos.com.ar/evento-addeventlistener-javascript/>
68. <https://programadorwebvalencia.com/cursos/javascript/eventos/>
69. <https://lenguajejs.com/eventos/introduccion/addeventlistener/>
70. https://developer.mozilla.org/es/docs/Learn_web_development/Core/Scripting/Events
71. <https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener>
72. <https://es.stackoverflow.com/questions/373691/se-puede-definir-el-orden-de-ejecución-de-eventos>
73. <https://es.javascript.info/dispatch-events>
74. http://www.codexemplar.org/curso/curso_4_3_e.php?mc=false
75. http://www.mywonderland.es/curso_js/addevent/index.html
76. <https://www.jaimetr.dev/posts/event-handler-vs-addeventlistener>
77. <https://somoospnt.com/blog/191-eventos-con-javascript>
78. <http://nachoiborraies.github.io/javascript/md/es/02b.html>
79. <https://es.stackoverflow.com/questions/241527/cómo-acceder-a-una-variable-que-esta-dentro-de-una-función-js>
80. <https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Functions>
81. <https://es.javascript.info/function-basics>
82. <https://desarrolloweb.com/articulos/517.php>
83. <https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/let>
84. <https://es.javascript.info/closure>
85. <https://web.dev/articles/global-and-local-scope?hl=es-419>
86. https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Grammar_and_types
87. <https://bootcamp.laboratoria.la/es/topics/javascript/basics/variables>
88. <https://jscodez.com/javascript-scope/>
89. <https://es.stackoverflow.com/questions/465749/pasar-parámetros-de-una-función-para-un-addeventlistener>
90. <https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener>
91. <https://lenguajejs.com/eventos/introduccion/addeventlistener/>
92. https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=865%3Athisstyle-is-undefined-funcion-manejadora-de-eventos-addeventlistener-con-parametros-ejemplos-cu01177e&catid=78&Itemid=206
93. https://www.reddit.com/r/learnjavascript/comments/twljtu/how_do_i_pass_arguments_to_a_function_inside/
94. <https://www.esdeerre.com/ejemplos/flash/pasar-parametros-en-funcion-asociada-a-un-addeventlistener/>
95. <https://lenguajejs.com/eventos/introduccion/addeventlistener-handleevent/>
96. <https://lineadecodigo.com/dom/pasar-datos-a-un-listener/>
97. <https://www.youtube.com/watch?v=2oHVJLrnRmY>
98. <https://www.forosdelweb.com/f13/addeventlistener-pasar-parametros-573253/>

99. <https://lenguajejs.com/eventos/introduccion/addeventlistener/>
100. <https://developer.mozilla.org/es/docs/Web/API/EventTarget/addEventListener>
101. http://www.mywonderland.es/curso_js/addevent/index.html
102. <https://www.luisllamas.es/javascript-eventos-dom-mas-usados/>
103. <https://programadorwebvalencia.com/cursos/javascript/eventos/>
104. https://developer.mozilla.org/es/docs/Learn_web_development/Core/Scripting/Events
105. <https://www.jaimetr.dev/posts/event-handler-vs-addeventlistener>
106. <https://www.youtube.com/watch?v=PRUiwhjkGbc>
107. <https://www.q2bstudio.com/nuestro-blog/27563/addeventlistener-en-javascript-guia-para-principiantes>
108. <https://www.chucksacademy.com/es/topic/javascript-events/tipos-de-eventos-en-javascript>