

Conectando JavaScript con HTML

Web Development

Conectando JavaScript con HTML

Para agregar JavaScript a un documento HTML se utiliza la etiqueta `<script>`. Generalmente se coloca dentro del elemento `<head>` o al final del `<body>`. En lugar de tener contenido dentro de la etiqueta, se usa el atributo `src` para indicar la ruta del archivo JavaScript que se va a cargar.

```
<script src="script.js"></script>
```

<https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Elements/script>

Comportamiento por defecto

Por defecto, los navegadores detienen la carga del HTML hasta que el script haya sido descargado y ejecutado.

Esto puede hacer que la página tarde más en mostrarse.

Para evitar este bloqueo, existen dos atributos importantes:

- Atributo defer
- Atributo async

Atributo defer

Descarga el script de forma asíncrona, sin bloquear la carga de la página.

El script se ejecuta automáticamente después de que el DOM haya sido completamente procesado.

```
<script src="script.js" defer></script>
```

Ideal para scripts que necesitan acceder al contenido del DOM.

Es la opción más recomendada actualmente.

Atributo async

Descarga el script mientras el HTML sigue cargando.

Se ejecuta tan pronto como el script termina de descargarse, incluso antes de que se haya cargado todo el DOM.

```
<script src="script.js" async></script>
```

Útil para scripts que no dependen del DOM, como análisis o publicidad.

Su ejecución puede ser impredecible si depende del contenido de la página.

Ubicación tradicional

Antes, los scripts se colocaban al final del <body>:

```
<body>
```

```
...
```

```
<script src="script.js"></script>
```

```
</body>
```

Esto aseguraba que el DOM estuviera listo antes de ejecutar el script.

Sin embargo, usar el atributo defer logra el mismo efecto de forma más eficiente.

Ejemplo

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Connecting JS To HTML</title>
  </head>
  <body>
    <button>Click Me</button>
  </body>
</html>
```

Click Me

```
const button = document.querySelector('button');
button.addEventListener('click', setBackgroundColor);

function setBackgroundColor() {
  document.body.style.backgroundColor = '#00334C';
}
```

Ejemplo JS dentro del HTML, no buena practica

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Connecting JS To HTML</title>
    <script>
      console.log('hello world');
    </script>
  </head>
  <body>
    <button>Click Me</button>
  </body>
</html>
```


Ejemplo JS dentro del HTML, no buena practica

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Connecting JS To HTML</title>
    <script src="script.js"></script>
  </head>
  <body>
    <button>Click Me</button>
  </body>
</html>
```

Justo ahora esto no va a funcionar porque el código lo lee de arriba hacia abajo, al momento de leer el script el botón no existe entonces no podemos interactuar con el

Ejemplo JS dentro del HTML, no buena practica

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Connecting JS To HTML</title>
  </head>
  <body>
    <button>Click Me</button>
    <script src="script.js"></script>
  </body>
</html>
```

Si lo ponemos en el fondo del body “funciona” pero semánticamente no es bueno y además si tenemos un HTML muy largo el JS no va a cargar hasta el final y eso consume tiempo/performance.

Ejemplo JS dentro del HTML, no buena practica

```
window.addEventListener('DOMContentLoaded', main);  
  
function main() {  
  const button = document.querySelector('button');  
  button.addEventListener('click', setBackgroundColor);  
  
  function setBackgroundColor() {  
    document.body.style.backgroundColor = '#00334C';  
  }  
}
```

Podemos manejar dentro del mismo JS un event listener para que primero se cargue el dom y después se llamen las funciones del JS

Ejemplo JS dentro del HTML, no buena practica

```
window.addEventListener('load', main);
```

Podemos usar load tambien, solo que eso espera a que se cargue todo, las imágenes, estilos y todo, el otro método solo espera a que se cargue la estructura básica del DOM tree

Ejemplo JS dentro del HTML, no buena practica

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Connecting JS To HTML</title>
    <script src="script.js" defer></script>
  </head>
  <body>
    <button>Click Me</button>
  </body>
</html>
```

Usando defer podemos conseguir el mismo resultado de una manera más limpia y eficiente, normalmente es el método que usaremos.