

[< Cómo llega un script al navegador](#)

18

Respuesta a: **Cómo llega un script al navegador**AugdiAugust  8105

## ¿Cómo llega un script al navegador?

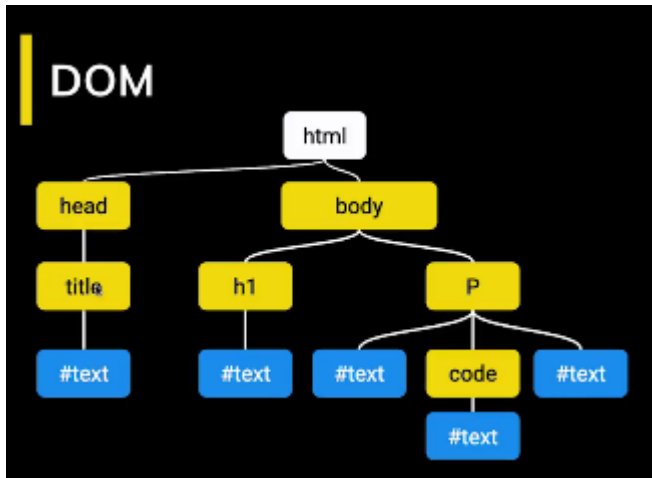
Anteriormente lo usamos con la etiqueta `<script>`, pero hay muchas más formas de traer un script.

## DOM

Es una representación de un documento HTML, por ejemplo:

```
<html>
<head><title>DOM</title></head>
<body>
  <h1>DOM</h1>
  <p>
    Cuando llega el HTML al browser, este lo empieza a parsear:
    va leyendo etiqueta por etiqueta y va creando el DOM.
    Cuando este proceso termina por completo, es cuando
    obtenemos el evento DOMContentLoaded.
  </p>
</body>
</html>
```

Cuando el navegador recibe este archivo lo tiene que convertir en una estructura parecida a un árbol.



Es una estructura fácil de procesar. Luego que el documento es leído y cargado ocurre el **DOMContentLoaded**.

## DOMContentLoaded

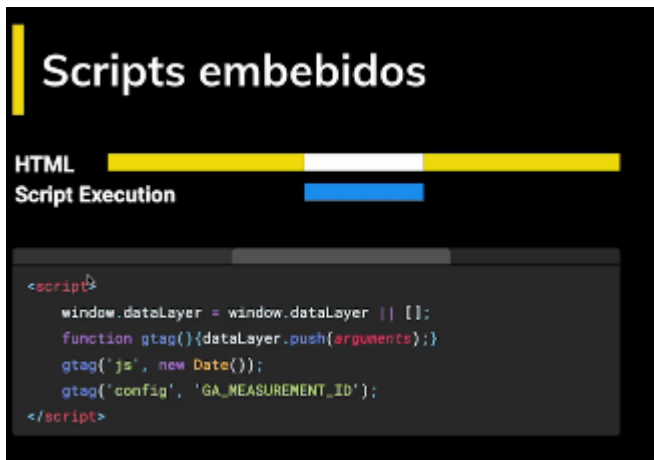
A partir de este punto tenemos la garantía de que todo nuestro documento se ha cargado.

## Scripts externos o embebido

Cuando abrimos etiquetas scripts lo hacemos como un elemento más del HTML, pero no todos son iguales. Veamos este ejemplo:

```
<html>
<head>
  <!-- Global site tag (gtag.js) - Google Analytics -->
  <script async
src="https://googletagmanager.com/gtag/js?id=U-123456-1"></script>
  <script>
    window.dataLayer = window.dataLayer || [];
    function gtag(){dataLayer.push(arguments);}
    gtag('js', new Date());
    gtag('config', 'GA_MEASUREMENT_ID');
  </script>
</head>
<body><!-- ... --></body>
</html>
```

Primero tenemos un **script** que trae a **google analytics** con una fuente externa, luego tenemos un **script** embebido de cuatro líneas. Cuando tenemos un código embebido provoca que el navegador detenga el proceso de carga hasta que todo el código que esté dentro de ese **script** se ejecute por completo. Pasaría algo así:



Nuestro navegador estaría leyendo el **HTML** y luego se encuentra con el **script**. En este caso el **script** es muy corto, el navegador se detendrá allí pero se cumplirá muy rápido.

## ¿Donde colocar el script?

Es muy importante saber esto, por eso vamos a ver un ejemplo:

```
<body>
  <script>
    let formEl = document.querySelector("#loginForm");
    formEl.addEventListener("submit", function (event) {
      // ...
    });
  </script>
  <form id="loginForm" action="/login" method="POST">
    <!-- ... -->
  </form>
</body>
```

Tenemos primero el **script** con las funcionalidades que quisiéramos tener en un formulario. Luego tenemos el formulario.

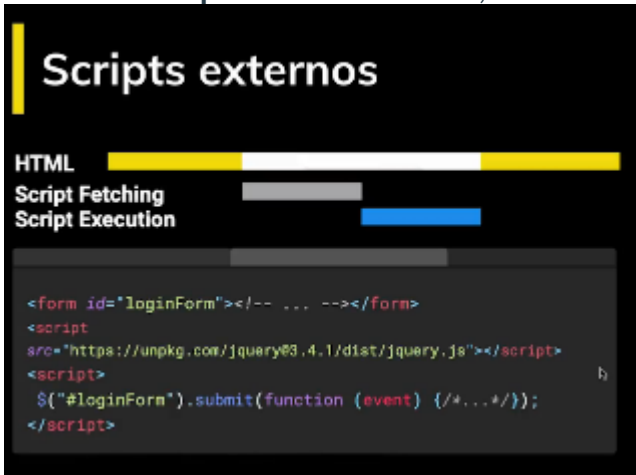
¿Qué pasaría si ejecutamos este código? No dará el famoso error:



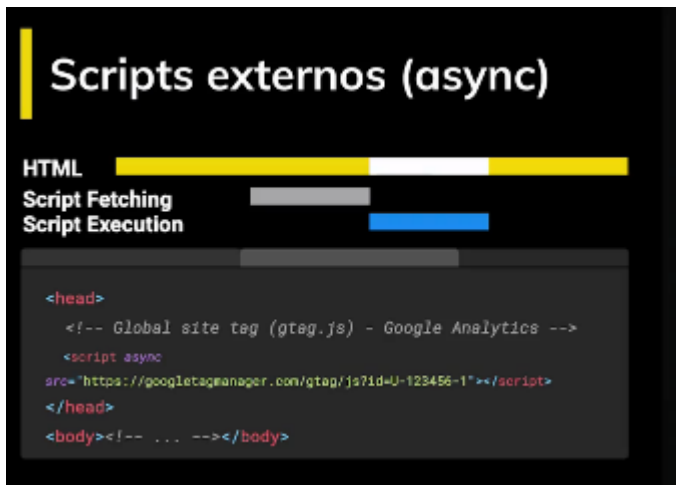
Cuando tenemos el **script** por arriba del formulario sucede que nuestro navegador detiene la lectura y creación de elemento cuando ve la etiqueta **script**. En nuestro navegador no existe la ID que usamos en el JavaScript, tampoco existen nuestro formulario. La solución es colocar elemento que vamos a tratar con el **script** en la parte de arriba para que se cargue primero. Por esto es que siempre nos han aconsejado colocarla antes de cerrar la etiqueta `</body>`, esto para que el navegador encuentre todos los elementos antes de ejecutar el script.

## Scripts externos

Con estos **scripts** sucede lo mismo, nuestro navegador detendrá el proceso hasta que la petición a la fuente y el **script** lleguen a manos de el.



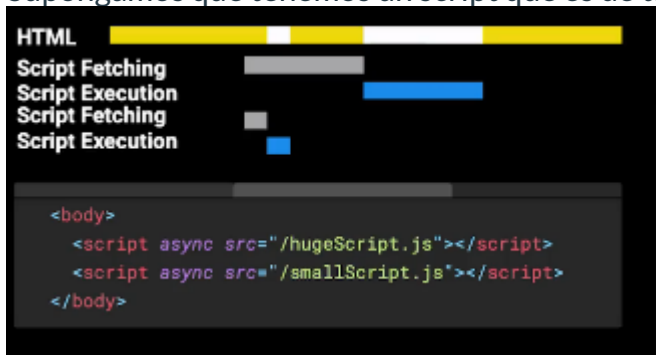
Ahora existe un atributo que le podemos dar a los scripts externos.



Con este atributo logramos que la carga del **HTML** no se interrumpa cuando el **script fetching** está sucediendo, esto quiere decir que la petición va a ocurrir asíncronamente. Cuando ya tenemos el script y pasamos al **script execution** sí nos detenemos.

## ¿Qué sucede si tenemos dos script que son asíncronos?

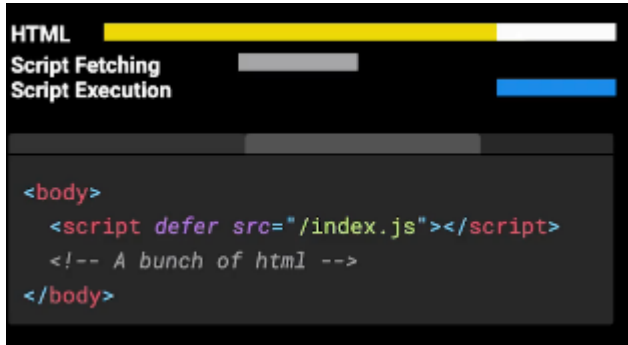
Supongamos que tenemos un script que es de tamaño inmenso y el otro sí es un poco reducido.



Ambas peticiones salen pero no necesariamente se contestan en el mismo orden, en este caso primero responde el primer **script**, cuando se termina de cargar inmediatamente se ejecuta. Cuando llega el segundo **script** pasa lo mismo, se ejecuta inmediatamente interrumpiendo al HTML.

## Scripts externos con defer

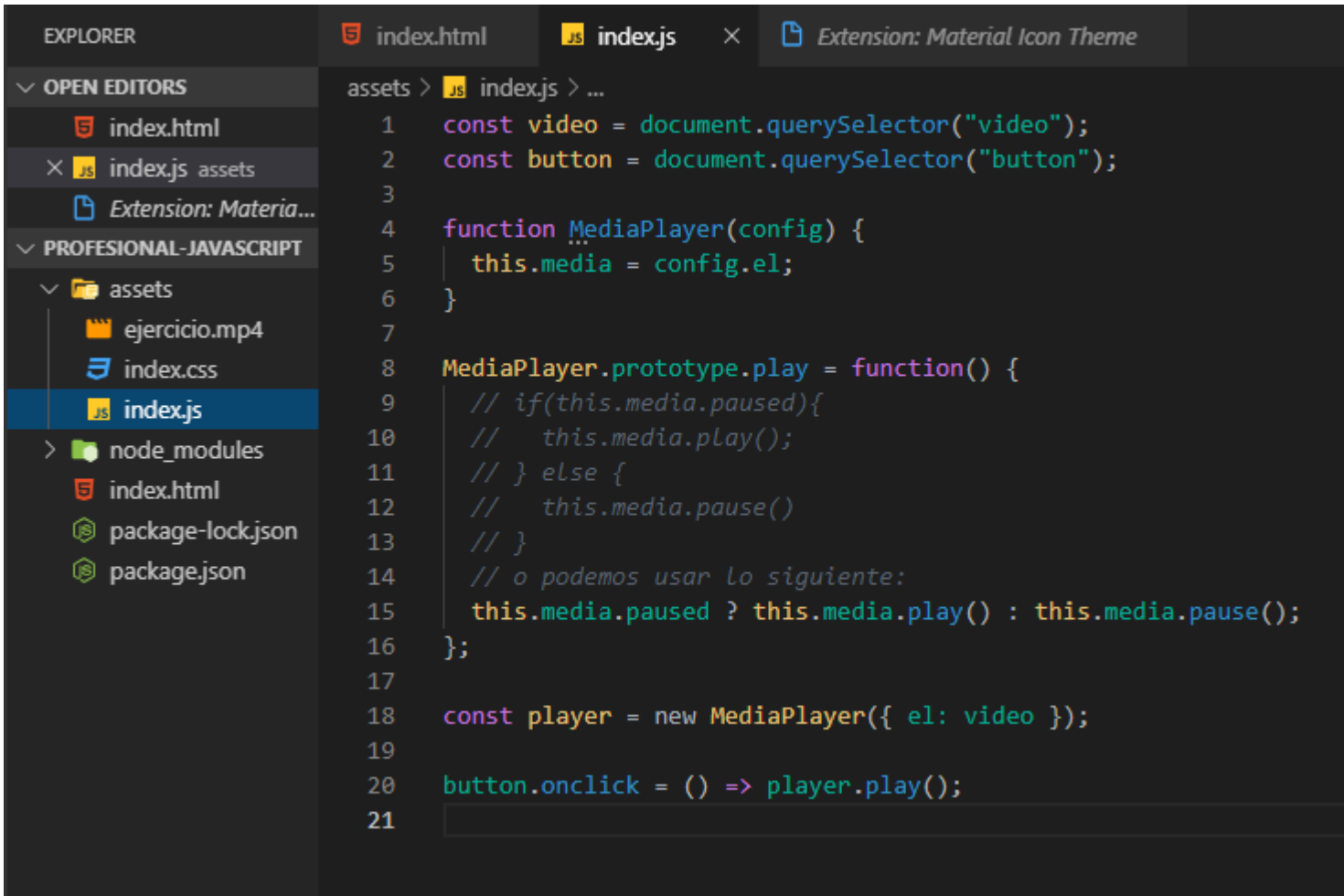
Tenemos esta tercera forma de traer scripts externos. Esto lo que haces es posponer la ejecución del JavaScript hasta el final de la carga del HTML.



Es similar al `async` ya que funciona asíncronamente sin detener el HTML. La gráfica muestra que la petición ocurre y se responde muy temprano, pero no se ejecuta hasta el final.

## ¡Vamos al código

Ahora en nuestro código lo que haremos es crear un archivo `index.js` en nuestra carpeta **assets**. Luego cortaremos el script que teníamos en nuestro HTML y lo pegamos en el archivo creado, no tenemos que pasar las etiquetas `</script>`. Quedará así:



The screenshot shows the VS Code editor interface. On the left, the Explorer sidebar displays the file structure with 'index.js' selected under the 'assets' folder. The main editor area shows the content of 'index.js' with the following code:

```
1  const video = document.querySelector("video");
2  const button = document.querySelector("button");
3
4  function MediaPlayer(config) {
5    this.media = config.el;
6  }
7
8  MediaPlayer.prototype.play = function() {
9    // if(this.media.paused){
10     //  this.media.play();
11    // } else {
12     //  this.media.pause()
13    // }
14    // o podemos usar lo siguiente:
15    this.media.paused ? this.media.play() : this.media.pause();
16  };
17
18  const player = new MediaPlayer({ el: video });
19
20  button.onclick = () => player.play();
21
```

Ahora a las etiquetas `<script>` que quedaron en nuestro HTML le agregaremos el atributo `src` con el **path** de nuestro nuevo archivo `index.js`.

```
<scriptsrc="./assets/index.js"></script>
```

No usaremos `async` ni `defer` por que ya lo tenemos al final y no interrumpirá con el HTML.

5 hace 8 meses

Frontend con React.JS

Curso Profesional de JavaScript • Cómo llega un script al navegador



Escribe tu comentario

<https://platzi.com/comentario/707398/>

+ 2

**crstnmln** 2057 Puntos

🕒 7 meses

Fue la solución a una gran parte de mis dudas, muchas gracias por tomarse el trabajo de hacer esto (si fuiste tú quien lo hizo). Excelente aporte.

**christian04velzquez** 16059 Puntos

🕒 7 meses

Gracias.

**AugdiAugust** 8105 Puntos

🕒 7 meses

¡Mira el curso en versión escrita acá! Está diseñado para tí.

**jsconestilo** 12704 Puntos

🕒 8 meses

#platzi Propongo que esté sea un modelo a seguir para el resumen de cada tema. Gracias compañero

**AugdiAugust** 8105 Puntos

🕒 8 meses

¡Mira el curso en versión escrita acá! Está diseñado para tí.