



¿qué es el dom?

El DOM (Document Object Model) es el concepto que nos permite, desde Javascript, comunicarnos con la estructura que hemos programado en HTML.

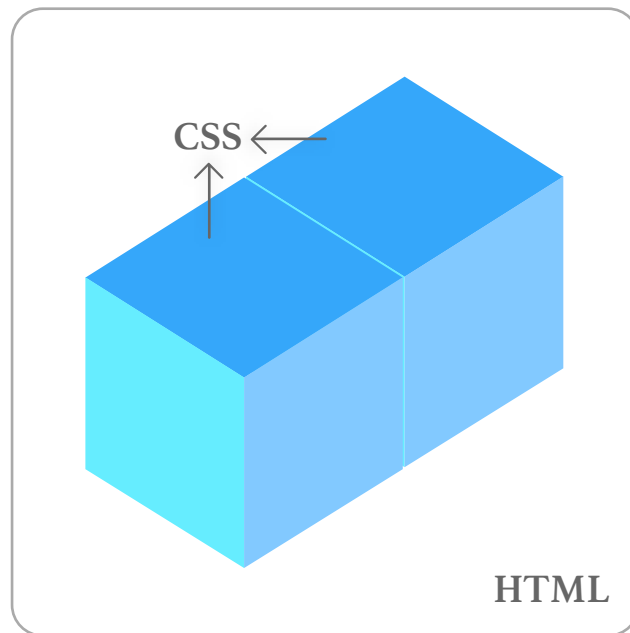
Pero viajemos un poco al pasado y revisemos los primeros apuntes.

Hace algunas semanas hemos diferenciado tres lenguajes que componen a una aplicación web y sus particulares roles.



¿por qué el dom?

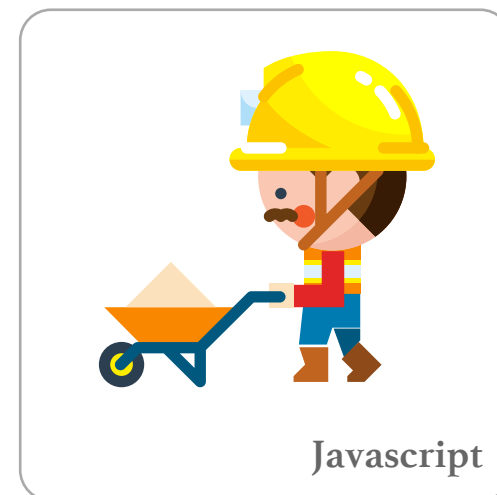
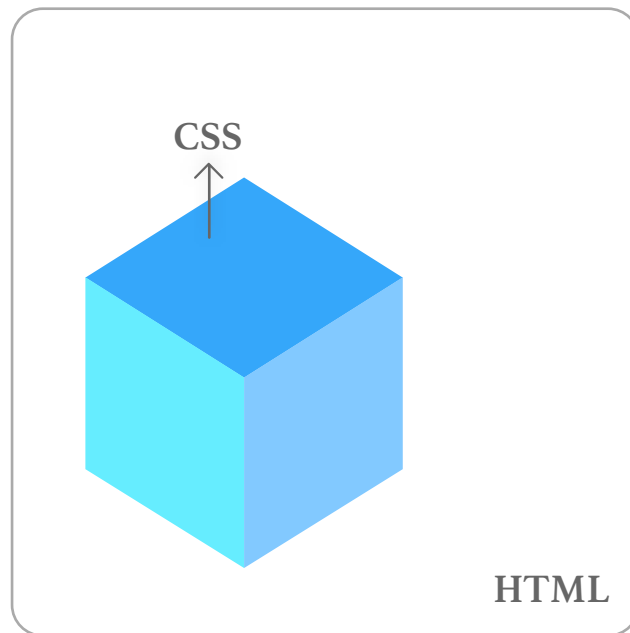
Con HTML definíamos una estructura, con CSS le dabíamos estilo y mediante JavaScript programábamos y definíamos flujos de usos. Nos resta *una pieza* para poder, desde Javascript acceder (y modificar) la estructura definida.





¿por qué el dom?

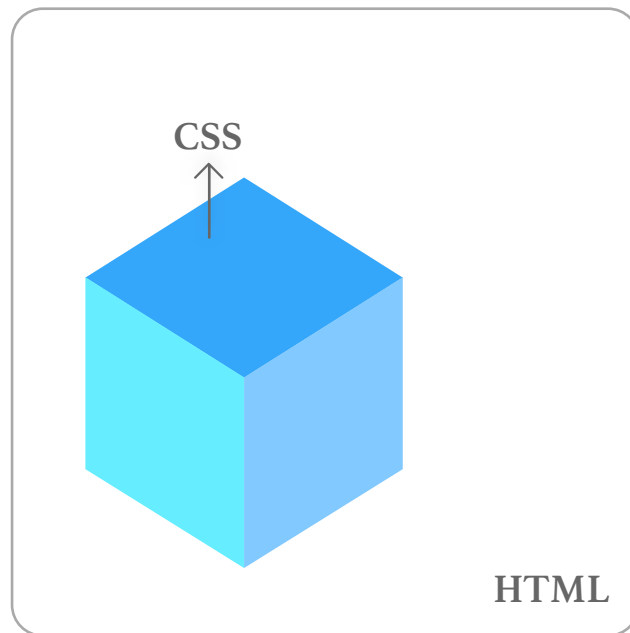
Con HTML definíamos una estructura, con CSS le dabíamos estilo y mediante Javascript programábamos y definíamos flujos de usos. Nos resta *una pieza* para poder, desde Javascript acceder (y modificar) la estructura definida.





¿por qué el dom?

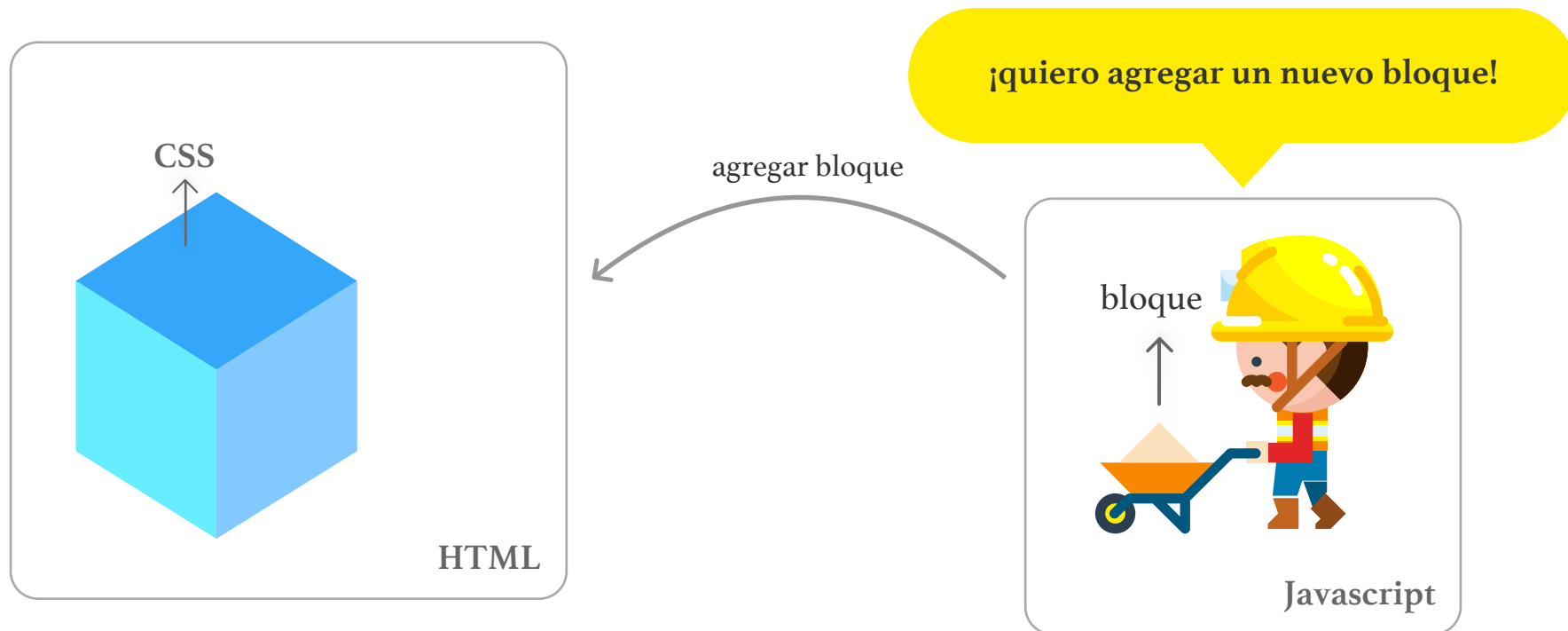
Con HTML definíamos una estructura, con CSS le dabámos estilo y mediante Javascript programábamos y definíamos flujos de usos. Nos resta *una pieza* para poder, desde Javascript acceder (y modificar) la estructura definida.





¿por qué el dom?

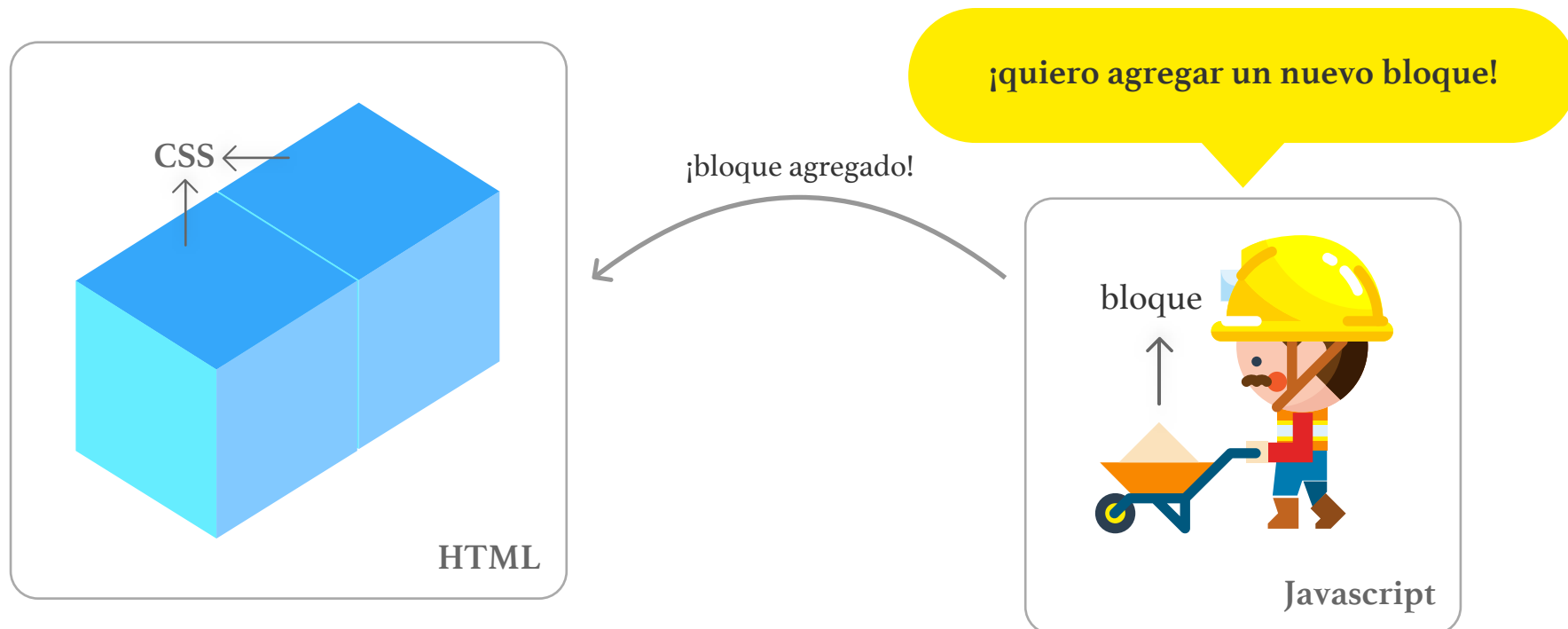
Con HTML definíamos una estructura, con CSS le dabámos estilo y mediante Javascript programábamos y definíamos flujos de usos. Nos resta *una pieza* para poder, desde Javascript acceder (y modificar) la estructura definida.





¿por qué el dom?

Con HTML definíamos una estructura, con CSS le dabámos estilo y mediante Javascript programábamos y definíamos flujos de usos. Nos resta *una pieza* para poder, desde Javascript acceder (y modificar) la estructura definida.

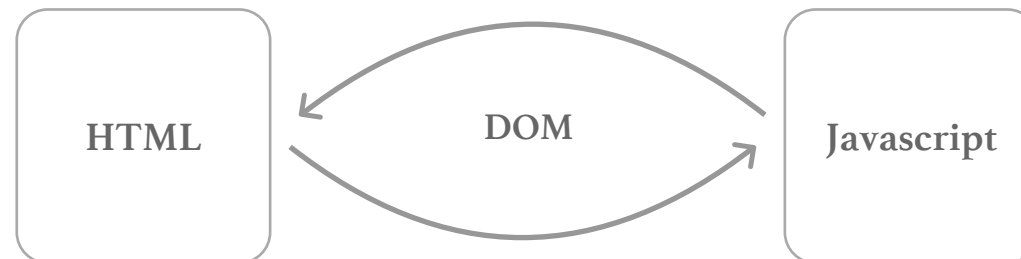




¿por qué el dom?

Es el DOM la pieza que nos faltaba: un concepto mediante el cual podemos interactuar, desde Javascript, con la estructura de nuestra aplicación (HTML).

El DOM es, técnicamente, la representación (el modelo) de esta estructura (o documento), expuesto en un objeto de Javascript. De acá el porqué de su nombre (Document Object Model, modelo del objeto documento).





¿qué son los elementos?

No. Esto no es un error. *No viajamos al pasado*. No es una diapositiva de HTML. Simplemente empezamos a cerrar el círculo del desarrollo de una aplicación web.

Un elemento, sabemos que es un “*ladrillo*” de nuestra estructura, de nuestro código HTML, y como lo que queremos es acceder a esta estructura, nuestro objetivo a cumplir mediante el DOM es acceder a los elementos.



¿qué son los eventos?

Un concepto fundamental en el desarrollo de aplicaciones web es el de eventos.

Un evento es un hecho que ocurre en el navegador producido por el usuario.

Pensémoslo como una acción, algo que el usuario hace: *un click, apretar una tecla, mover el mouse*.

¿Qué nos interesa respecto a estos eventos? Dos cosas: saber cuándo ocurren y saber cómo ocurren. Nos interesa saber cuándo un usuario aprieta una tecla y nos interesa saber qué tecla apretó. Cuándo y cómo.



¿cómo interactuar con el dom?

Bien. El DOM nos sirve para interactuar con la estructura. Tenemos dos conceptos fundamentales: *elementos* (ladrillos de la estructura) y *eventos* (hechos que suceden en la estructura). Pero, ¿cómo interactuamos con el DOM?

Hablamos ya de funciones globales/nativas (*alert*, por ejemplo) y de variables globales/nativas (*console*, el cual tiene una propiedad *log* que nos permite imprimir en pantalla).

Ahora nos toca hablar de la variable global que representa al DOM, de *document*.



¿qué es *document*?

document es una variable global (existe como variable sin que la hayamos definido nosotros) de tipo objeto, mediante la que nos vamos a comunicar con la estructura.

En programación, esto es a lo que se llama *interfaz*, un puente entre dos cosas: en este caso, entre *Javascript* y *HTML*, o entre *el código* y *la estructura*.



¿cómo usar *document*?

Recordemos que el DOM es la representación de la estructura en Javascript. Esta representación tarda un momento en construirse y al hacer uso del DOM tenemos que estar seguros que esto ya ocurrió, ya que sino intentaremos acceder a elementos que todavía no existen en esta representación.

Para estar seguros de estos, existe otra variable global llamada *window*, que cuenta con una propiedad llamada *onload*. El valor de esta propiedad será una función que se va a ejecutar una vez que el DOM sea construido, lo cual nos asegura que podemos usarlo y acceder a los elementos correctamente.



¿cómo usar *document*?



```
window.onload = function() {  
    // acá podemos usar el DOM  
}
```



¿qué hacer con *document*?

Ahora sí. *Podemos empezar a usar el DOM.* Y para lo primero que lo vamos a usar, es para identificar elementos.

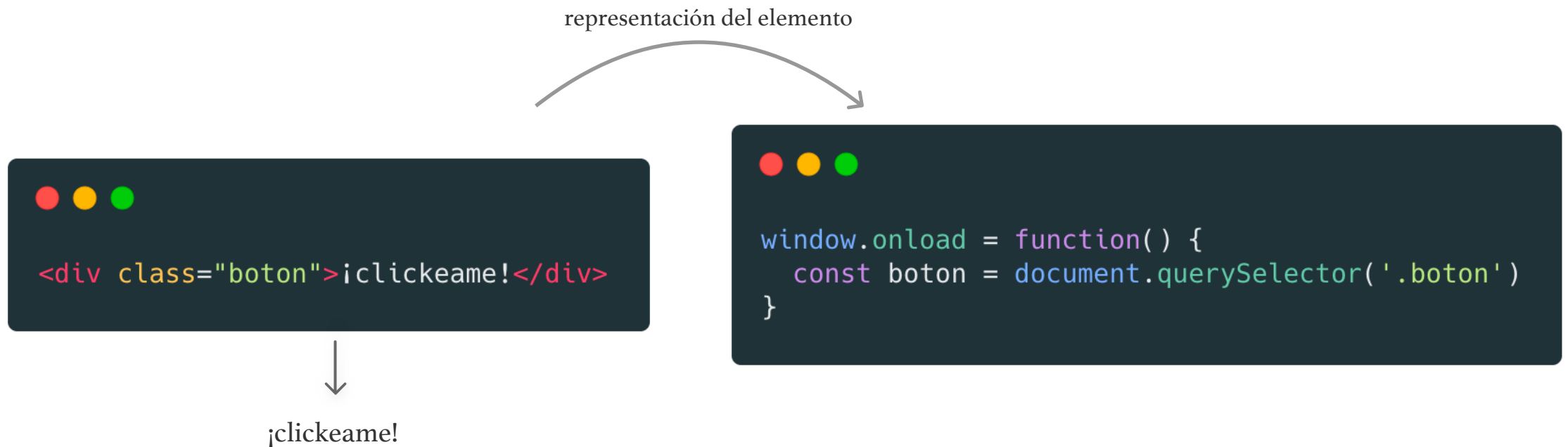
document presenta entre sus propiedades algunas funciones que nos van a servir para identificar los ladrillos de nuestra estructura.

Si hacemos memoria, en CSS tuvimos un problema idéntico, y es allí donde incorporamos el uso de selectores (*.boton* para seleccionar todos los elementos con la clase *boton*). Traigámoslos de vuelta para solucionar este nuevo problema.



¿cómo identificar elementos?

Entre sus propiedades, *document* tiene a la que probablemente más usemos, la función *querySelector* que recibirá como parámetro un selector. El resultado de la ejecución de esta función será, en caso de existir, un elemento que cumpla con el selector del parámetro.





¿qué hacer con un elemento?

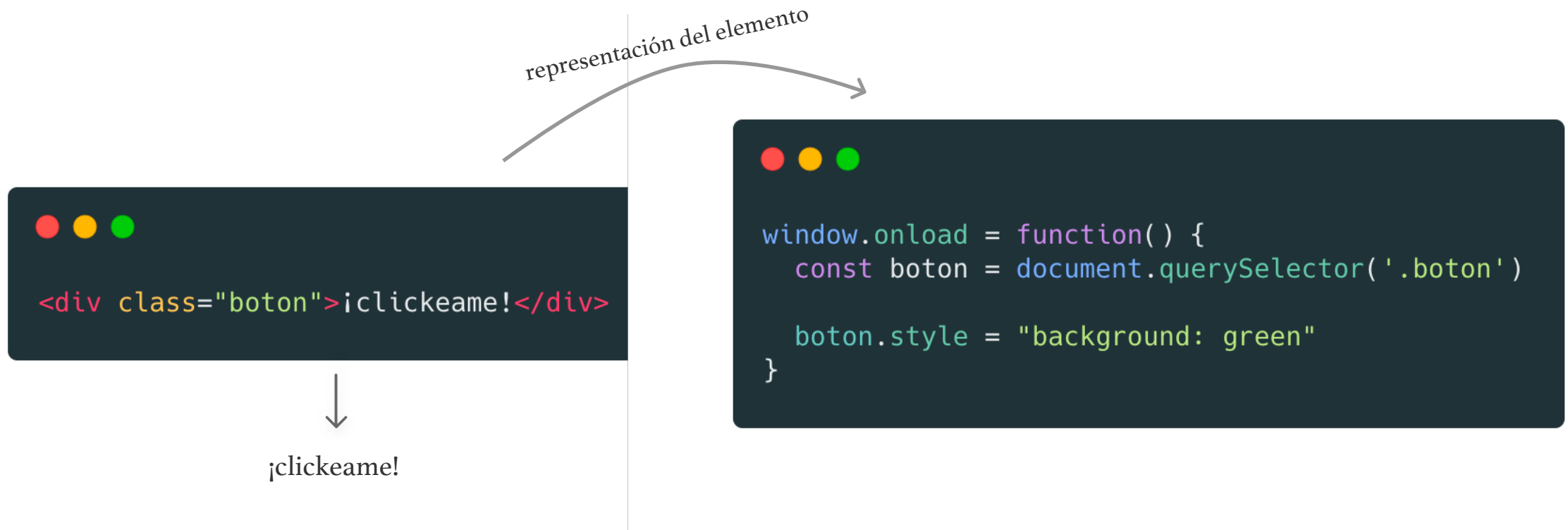
Lo más probable es que hayamos obtenido un elemento persiguiendo un fin en particular. ¿Cuál? Quizás *cambiarle el color de fondo, achicar la letra, hacerlo desaparecer*. Cualidad, casualmente de estilo. Dar estilo sabemos que era el rol de CSS. Una de las cosas que podemos hacer mediante el DOM es cambiar el estilo de un elemento en particular escribiendo código CSS. Pero, ¿cómo?

El resultado de la ejecución de la propiedad/función *querySelector* del objeto global *document* es un objeto que representa al elemento de la estructura HTML. Como tal, tiene propiedades que podemos modificar, y así **modificar al elemento**.



¿cómo cambiarle el estilo a un elemento?

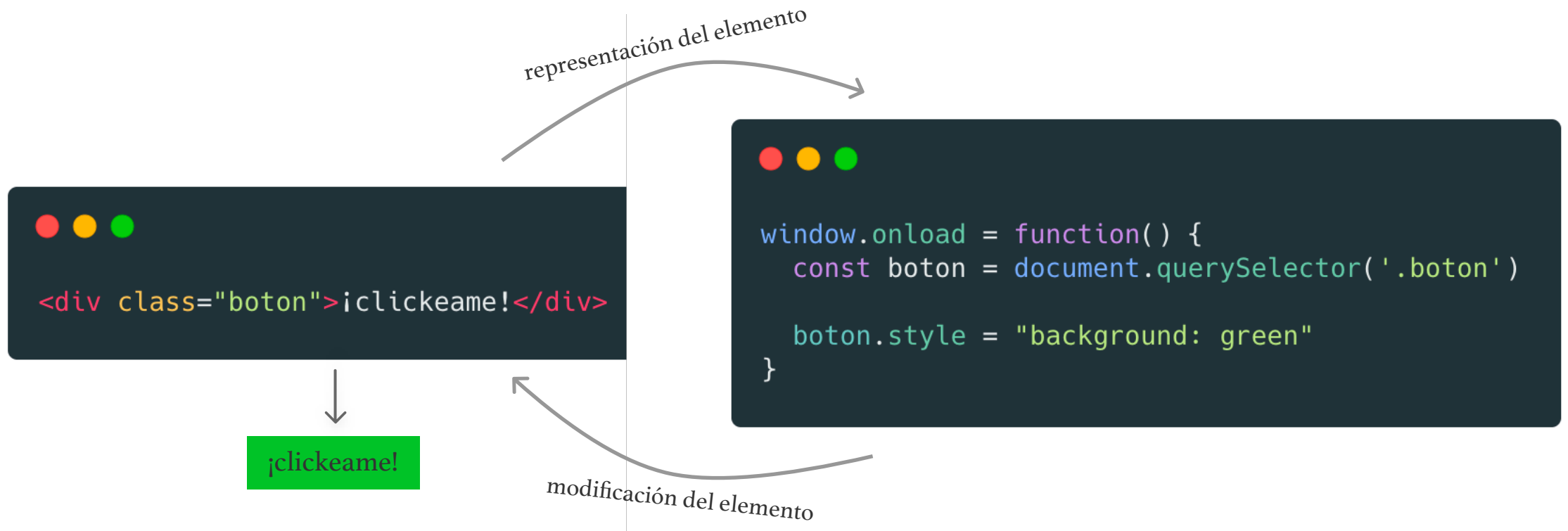
Los elementos son representados como objetos. Estos objetos/elementos tienen entre sus propiedades a, por ejemplo, *style*, la cual podemos modificar.





¿cómo cambiarle el estilo a un elemento?

Los elementos son representados como objetos. Estos objetos/elementos tienen entre sus propiedades a, por ejemplo, *style*, la cual podemos modificar.






ejercicio de dom

```
<h1>Manuscritos digitales</h1>

<p>Este es un proyecto tan nuevo como genial.</p>

<div id="nombre">Luchito</div>
```

1. Sin modificar la estructura, crear un programa que identifique el título, el párrafo, las dos imágenes y el nombre.
2. Modificar el programa para ocultar el título, agrandar el párrafo y cambiar el color del texto a violeta y ponerle bordes redondeados.
-  3. Si el usuario clickea en la firma, hacer que desaparezca el nombre. Si vuelve a clickear, que vuelva a aparecer. 