

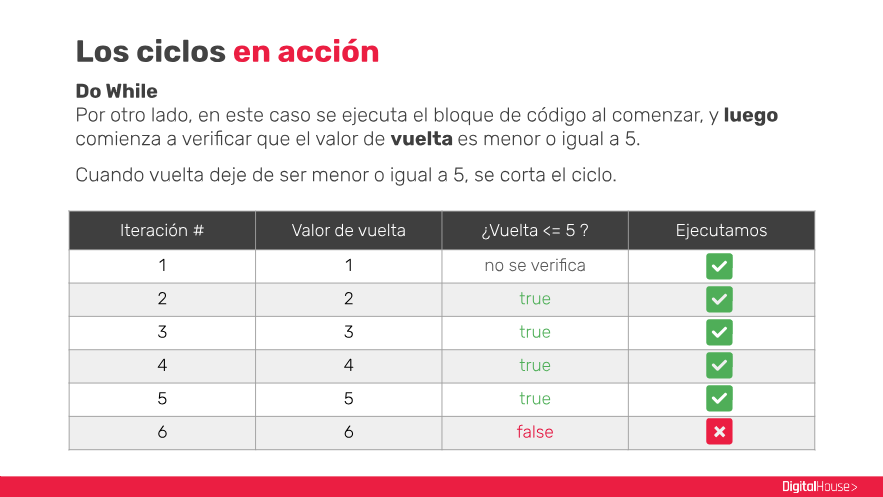
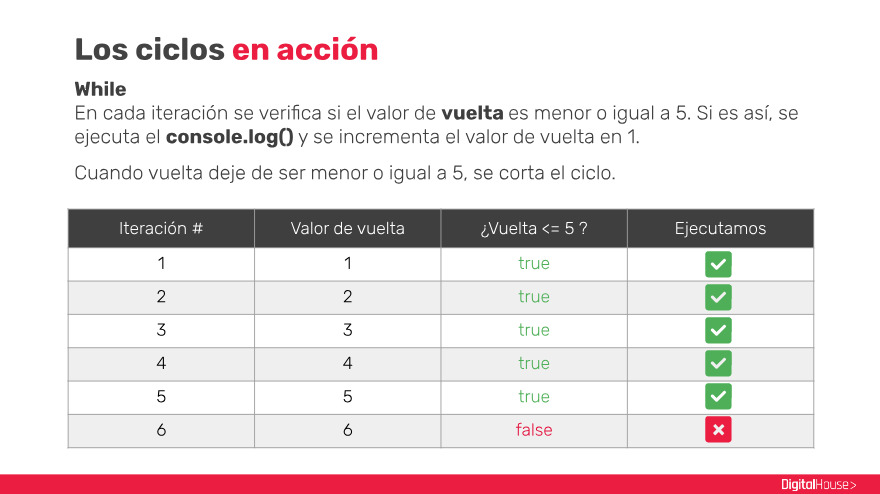
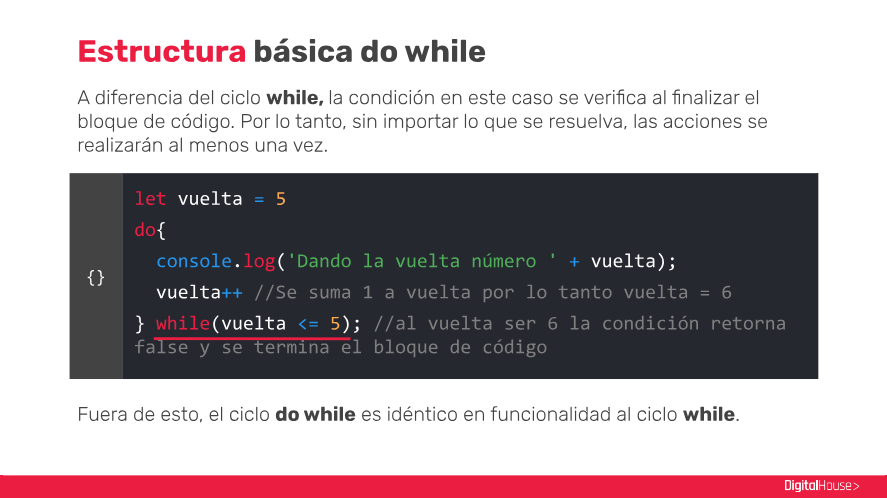
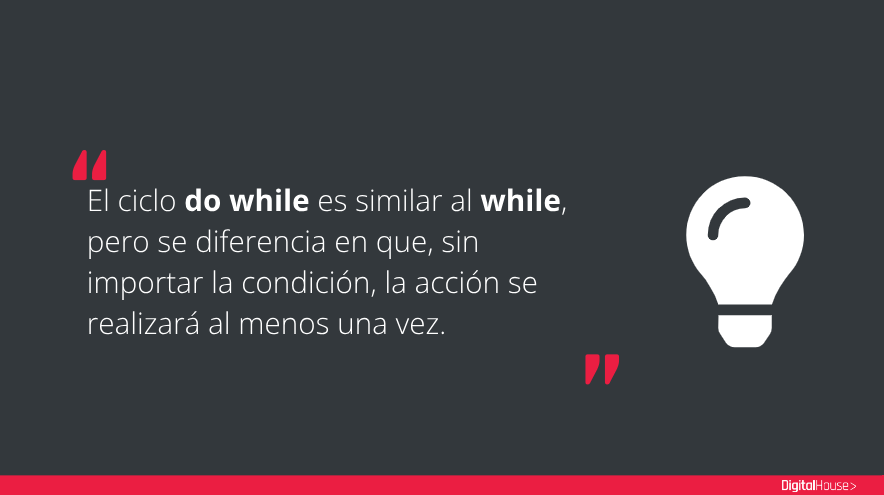
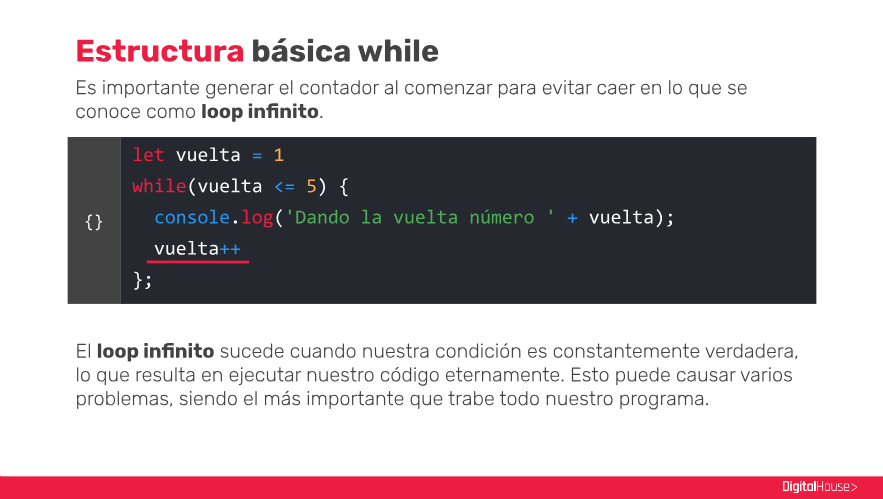
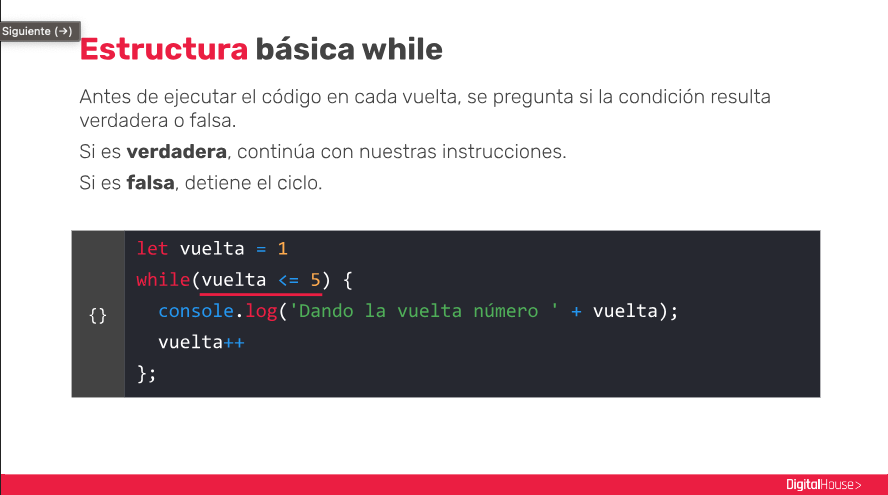
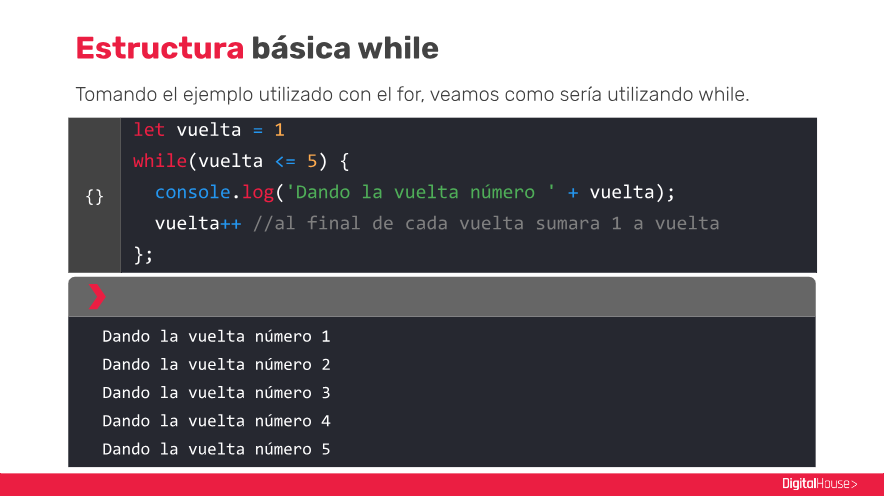
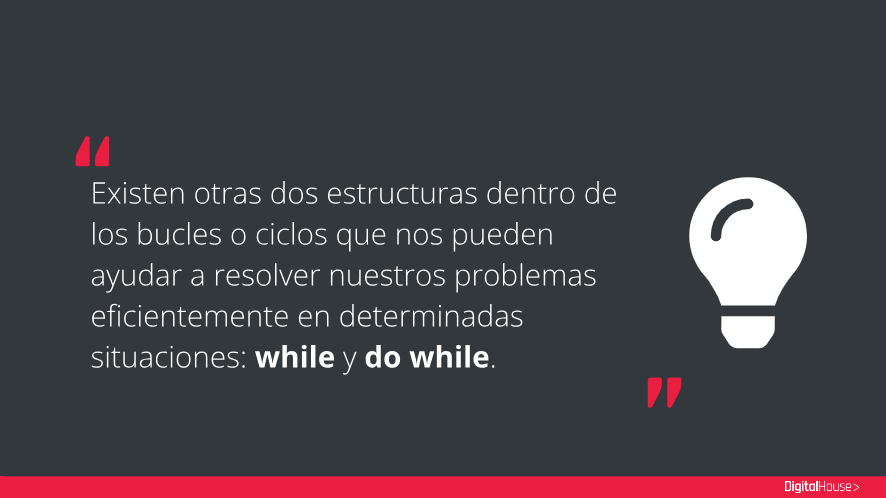
Ciclos: while y do while

Como en muchos lenguajes de programación —y JavaScript no podría ser menos—, existe más de una forma de realizar las cosas. Varios casos pueden diferir en determinados aspectos y, por eso, el lenguaje nos provee de diversas estructuras para realizar acciones determinadas, que si bien comparten su función core difieren en pequeñas cuestiones, ya sean de lógica, de estructura o de código en sí.

En este caso veremos otras dos formas de representar y recrear acciones cíclicas: el **while loop** y el **do while**.

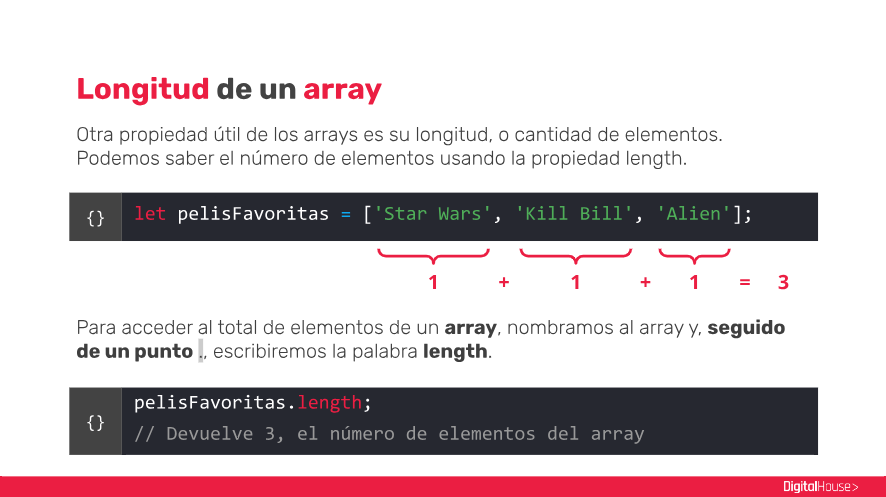
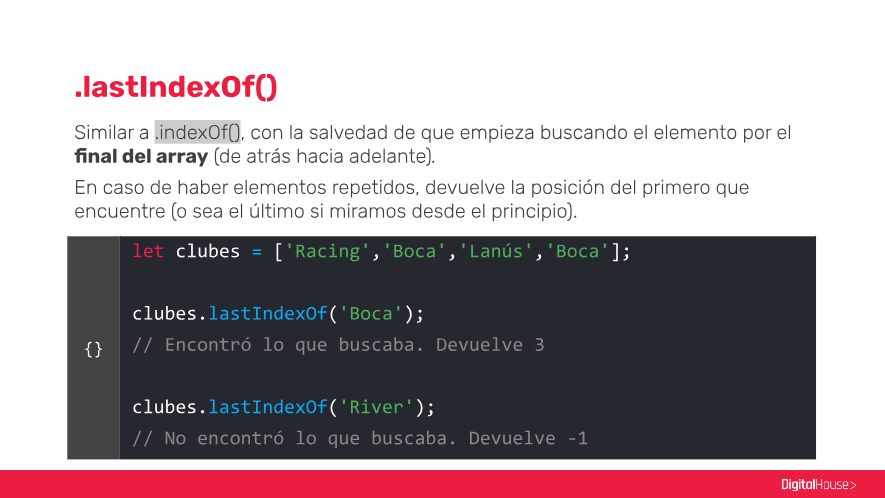
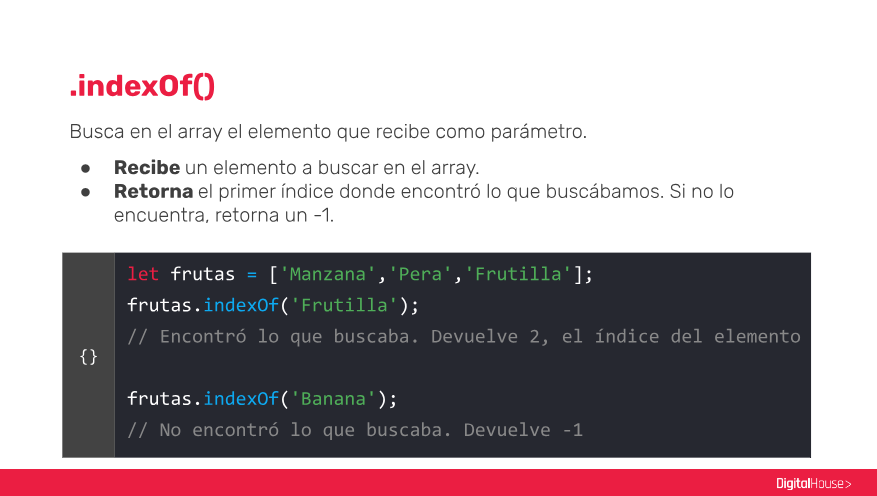
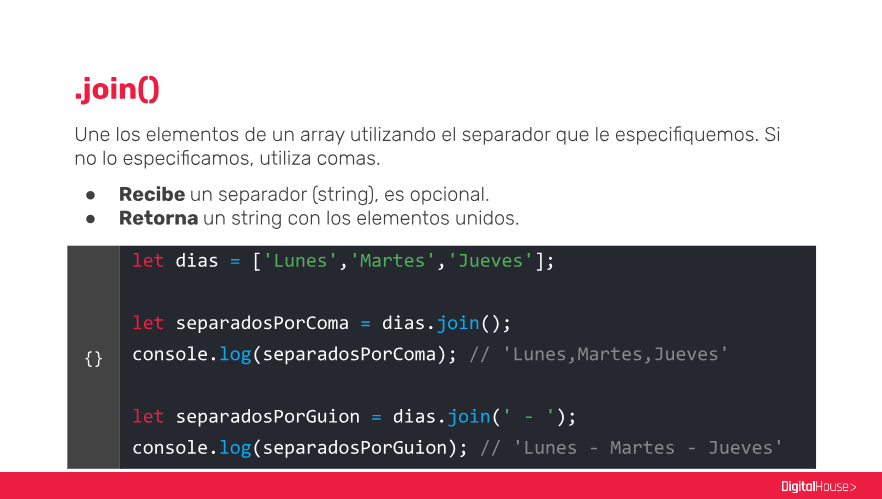
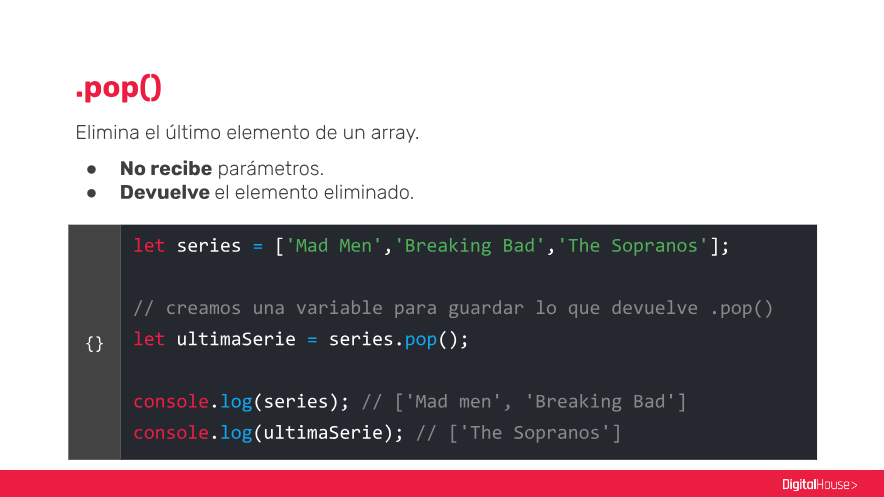
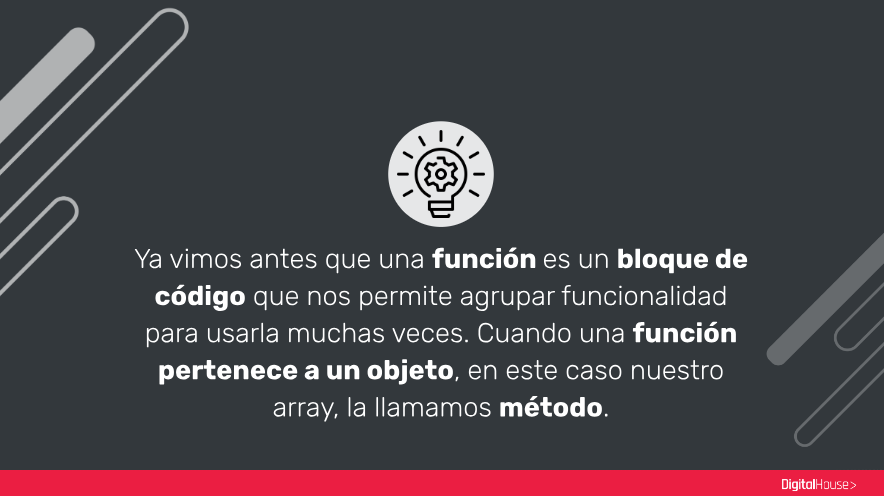
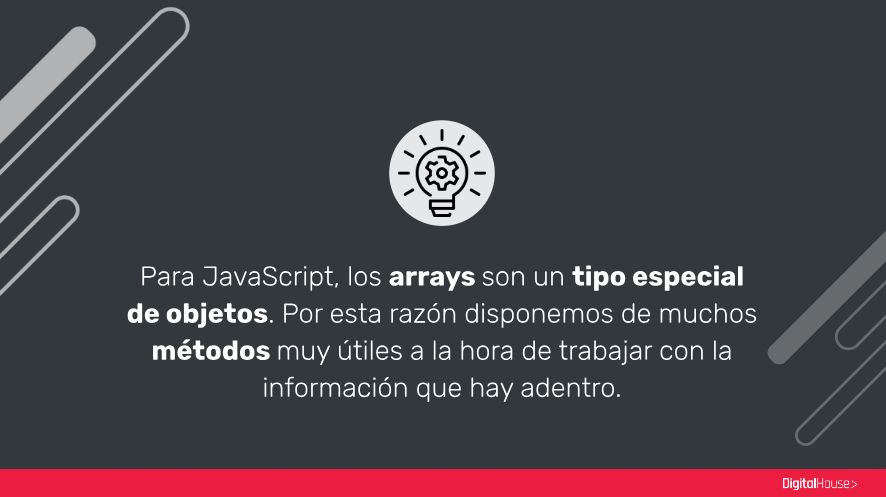
Si bien la estructura que más vamos a utilizar probablemente sea el ciclo **for**, existen casos y situaciones donde estas versiones pueden representar una mejora en nuestro código por la estructura que poseen.

Como pequeño aviso legal, durante las instancias evaluativas, no se pedirá ni evaluará el uso de la estructura While ni Do/While.



## Introducción a arrays

Ya sabemos cómo generar distintos tipos de datos en distintas variables, ya sean **números, booleanos, o cadenas de caracteres (strings)**… Pero ¿cómo podemos agrupar mucha información en una sola variable? Para ello contamos con un tipo de dato un poco más estructurado llamado **array** —también conocido como lista o arreglo—, que no es más que una “colección de elementos”.

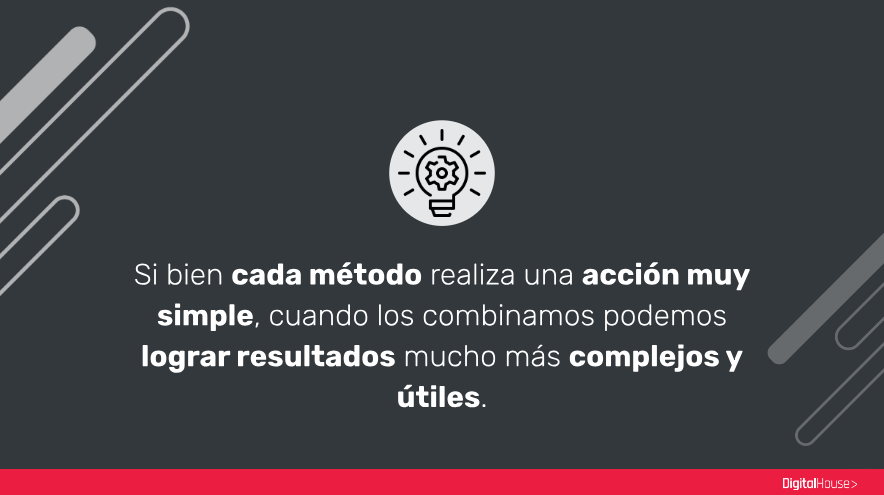
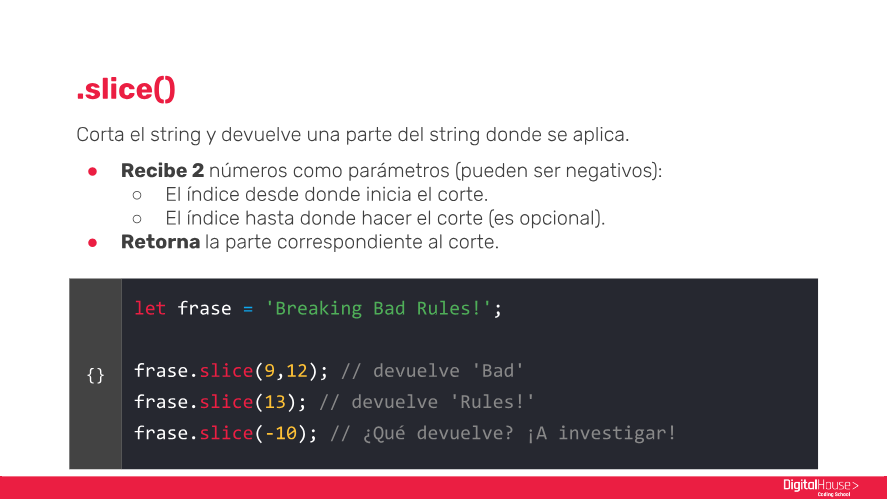
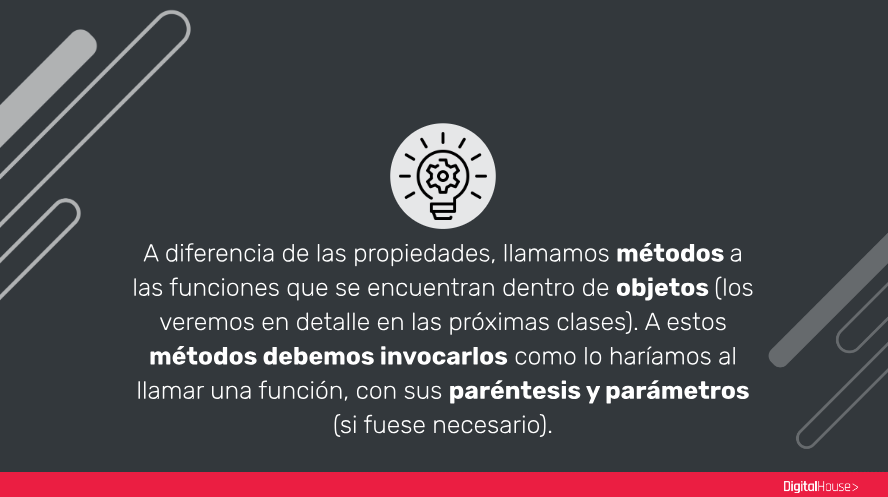
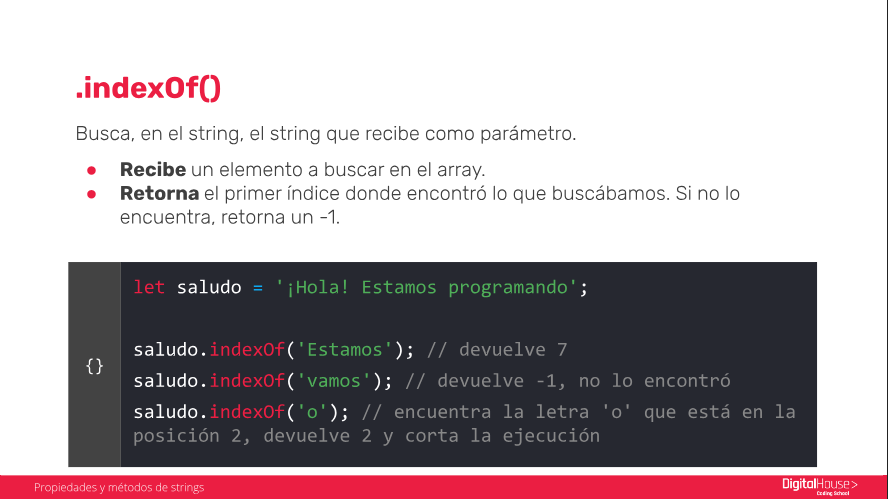
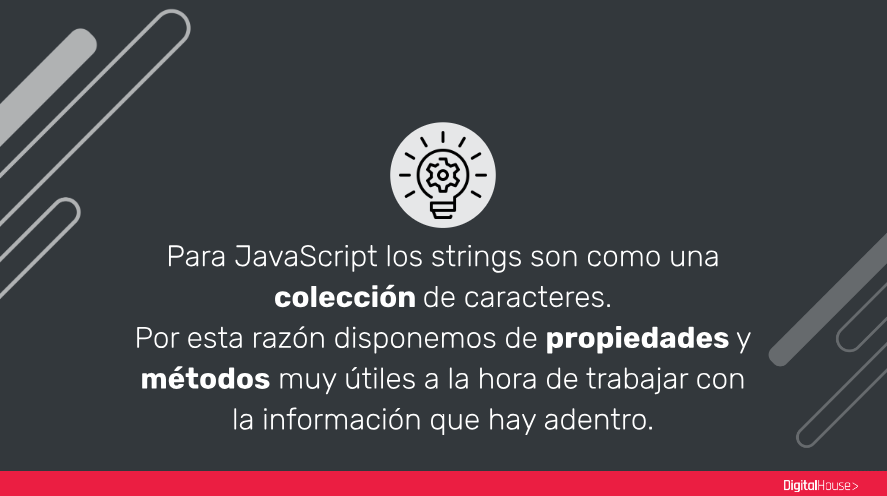


## Un poco más sobre strings

Ahora que conocemos los **arrays** vemos que podemos definirlos como listas o colecciones de elementos, organizados por un índice que comienza con el 0. Si lo analizamos, los strings son algo similares. Veamos un poco más sobre ellos.

Antes de meternos a fondo con los métodos de **string**, recordemos que —en muchos sentidos— para JavaScript un string no es más que una colección de caracteres. Esas cadenas de texto que parecen ser tan sencillas, pero que tienen un alto potencial.

En el siguiente video veremos que podemos hacer mucho más con un string que simplemente generar textos.



## Objetos literales

En JavaScript, un objeto literal es una entidad independiente con propiedades. A su vez, esas propiedades tienen valores.

El concepto de objetos puede compararse con entidades de la vida real. Por ejemplo, un país representaría un objeto literal con ciertas propiedades: su nombre, cantidad de habitantes, capital, etc. Del mismo modo, los objetos literales en JavaScript pueden tener propiedades que definan sus características.

Imaginemos entonces que queremos crear el modelo de un país en JavaScript. Mirá este video para aprender cómo hacerlo a través de un objeto literal.

