**C10A – Strings y Arrays**

**.length** (PROPIEDAD):

Esta propiedad retorna la cantidad total de caracteres del string, incluido los espacios. (Como es una **propiedad** **al invocarla no necesitamos paréntesis).**

**.indexOf ( ) :**

Busca, en el string, el string que recibe como parámetro. **Recibe** un elemento a buscar en el array y **retorna** el primer índice que recibe si no, retorna -1.

* Un **objeto** es una colección de datos relacionados y/o funcionalidad.
* Cuando una **función** le pertenece a un **objeto**, en este caso nuestro string, la llamamos **método.**

.**slice()** :

Corta el string y devuelve una parte donde se aplica. Recibe 2 numeros como parámetros (pueden ser negativos) A: Desde y B: Hasta ej : frase.slice(7,12);

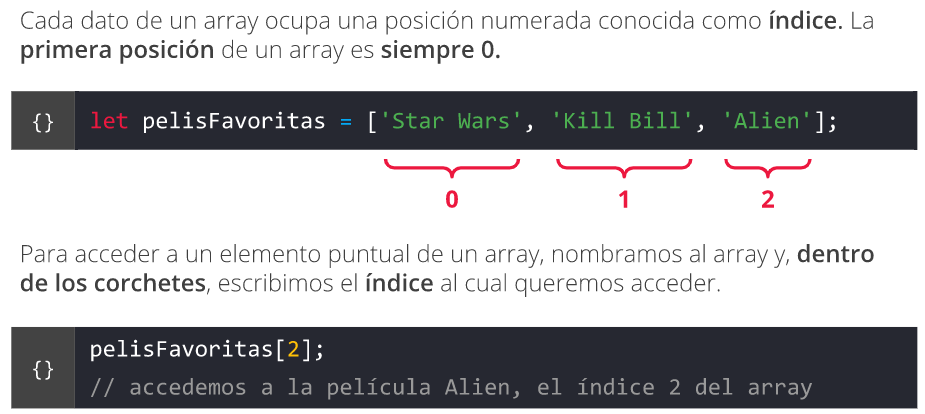
.**trim ()** :

Elimina los espacios que estén al principio y al final de un string, no recibe parámetros.

**.split ()** : Divide un string en partes, recibe un string(carácter) que usara como separador “ “ o , - . (¡ENTRE COMILLAS EL PARAMETRO!). Este separador debe estar en la frase.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2 / 3 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

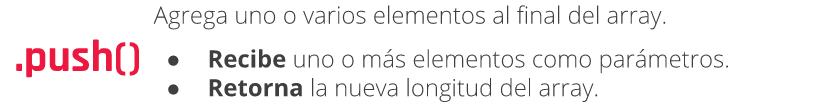
Arrays: Los **arrays** nos permiten generar una colección de datos en orden.

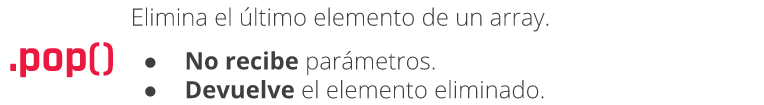


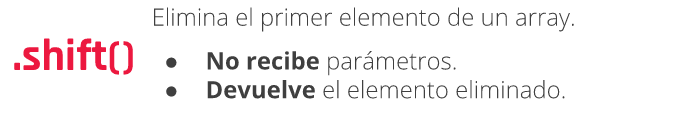
Longitud de un array: Podemos saberla con **.length** (cada índice cuenta por 1).

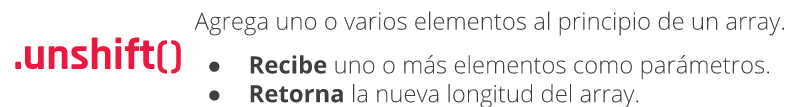
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3 / 3 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

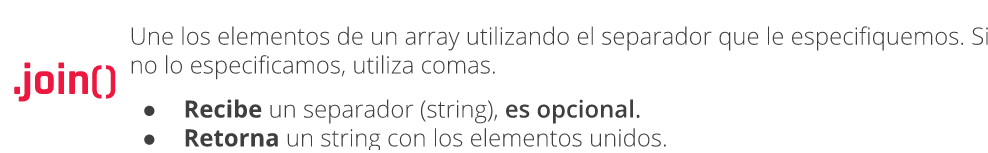
**Métodos de arrays:** Para JS los arrays son un tipo especial de objetos, por lo que tiene sus propios métodos.

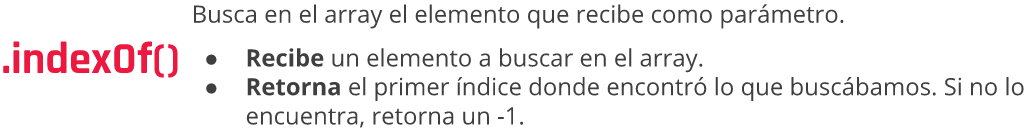


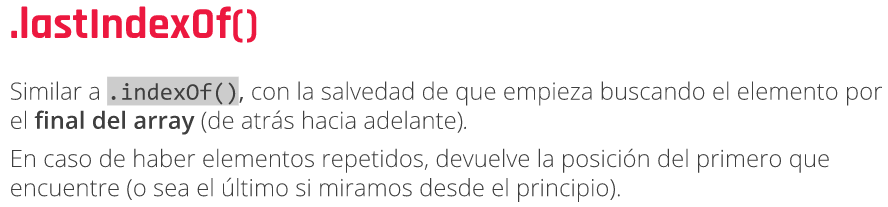


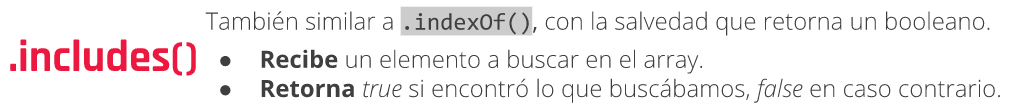






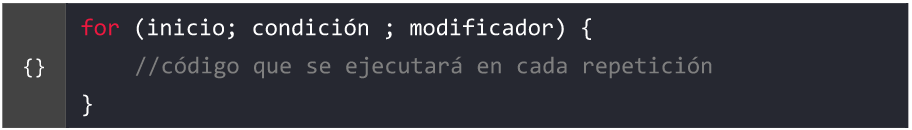




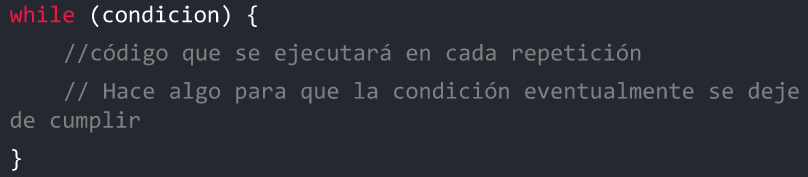


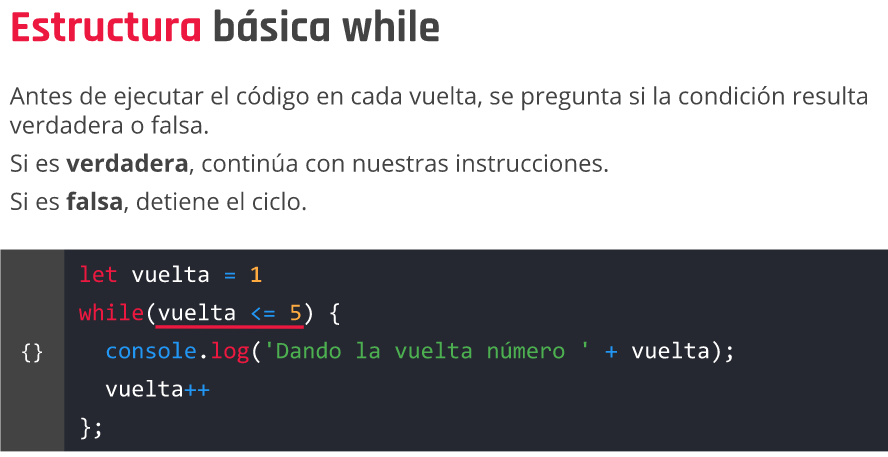
**C11A – For Loop / While-Do While**

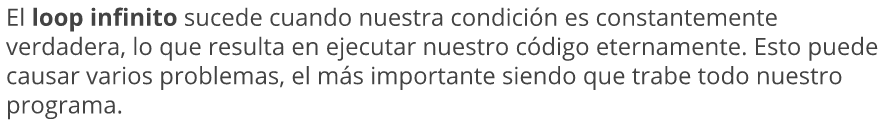
Ciclos **FOR:** Los ciclos nos permiten repetir instrucciones de manera sencilla. Podemos hacerlo una determina cantidad de veces, o mientras se cumpla una condición.

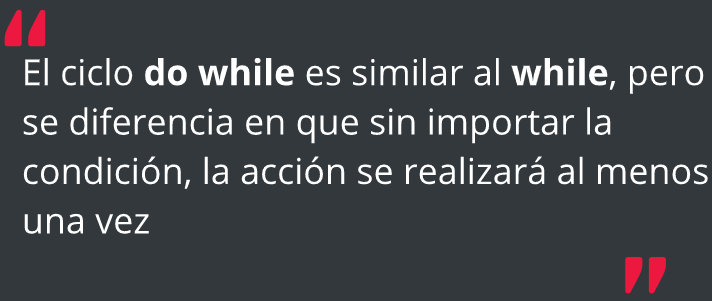


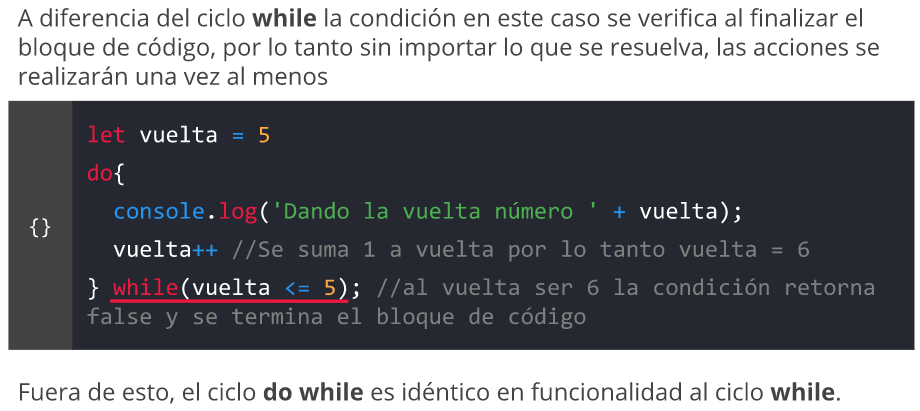
**While / Do While :**







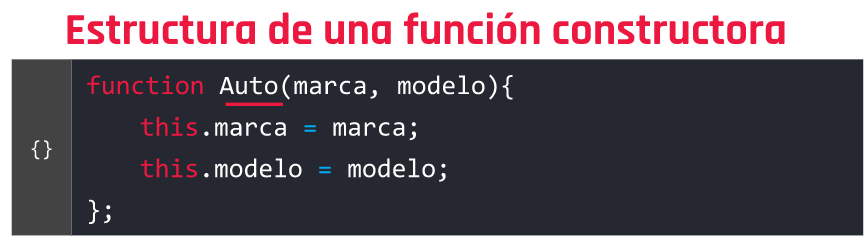




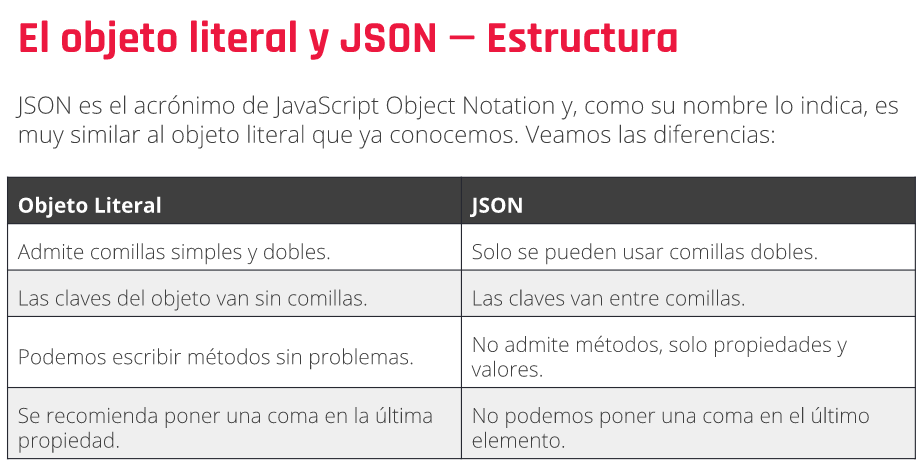
**C13A – Objetos literales y JSON**

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







**JSON (JavaScript Object Notation)  
  
**

****

****

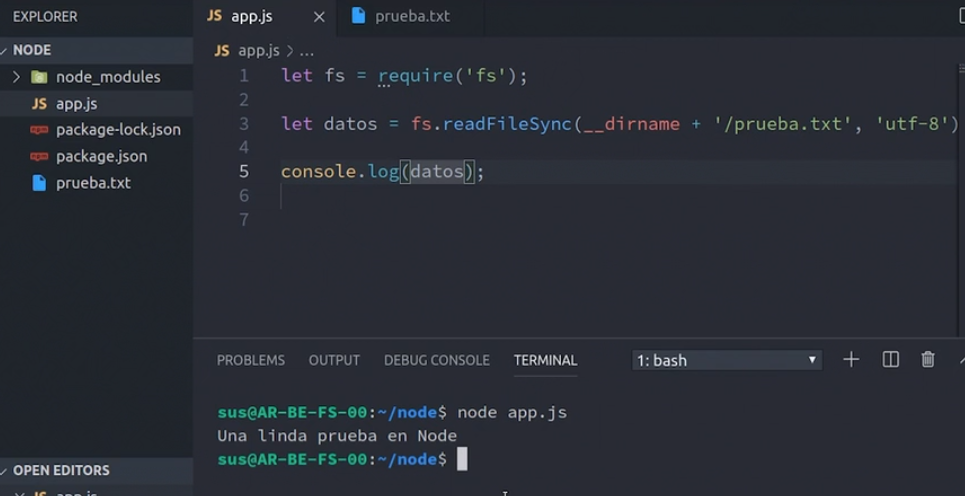
**C14 A Sistema de módulos (modularización)**

La modularización es poder separar nuestro código en diferentes archivos para hacerlo más fácil de leer, más corto, y mejor optimizado

Podemos dividir los módulos en tres categorías:

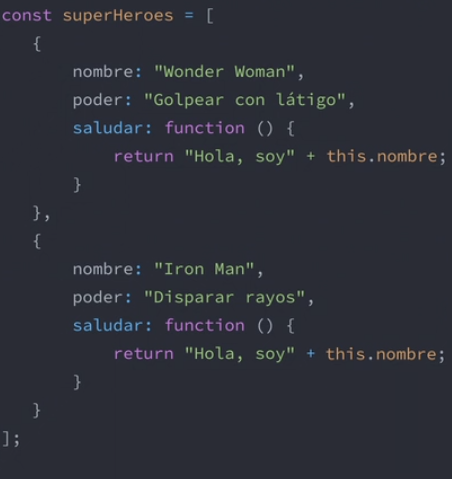
* **Módulos externos** son aquellos creados por otras personas o empresas, como lo son comúnmente las librerías y frameworks. Se deben instalar en la carpeta de nuestro trabajo.
* **Módulos nativos** similares a los externos, son funciones y métodos que nos ayudan a crear nuestras aplicaciones, con la diferencia de que vienen incluidos en el lenguaje (o en el caso de Node.js incluidos en el mismo entorno).
* **Módulos creados** son nuestras creaciones. Pueden ser variables, funciones, arreglos, casi cualquier cosa que nosotros queramos exportar.

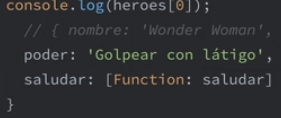
Un modulo es un bloque de código



Crear propio modulo:

1. Crear una carpeta, dentro de esa carpeta el modulo deseado. Ej . SUPERHEROES
2. Adentro de la carpeta SUPERHEROES, guardar el archivo, que normalmente recibe el nombre de **index.js;** su contenido podría ser el siguiente:



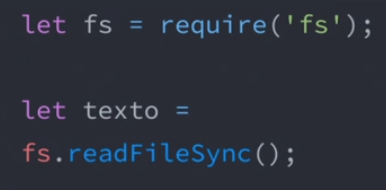
1. Como usar esta info ahora? > Necesitas exportar la variable que te interesa compartir.
2. Utilizar **module.exports = superHeroes**
3. **Escribir export , siempre al final del archivo .**
4. **En App.js hacer el require correspondiente, como esta en la imagen de modulo arriba.**
5. **Es fundamental respetar el nombre que le pusimos a la carpeta:**
6. 
7. Usar la variable, como se haría normalmente.
8. 

Ej playground : let perro = require ('./perro') ;

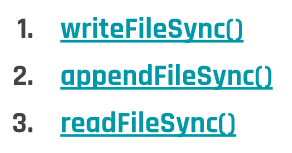
console.log("El perro se llama " + perro.nombre + " y tiene " + perro.edad+ " años")

## File system

El módulo nativo de Node.js llamado **file system (fs)** nos provee de la posibilidad de trabajar con el sistema de archivos de nuestra computadora, que en términos simples sería acceder al disco duro (o SSD si estamos un poquito más actualizados) para poder realizar algunas lecturas de archivos, modificarlos o mismo crearlos







<https://playground.digitalhouse.com/course/2848/unit/41118/lesson/90115/topic/395346>

