

```
----- AYUDAS -----
FORMULARIO
Programar el botón de submit del formulario
document.getElementById().addEventListener("submit", function(event){
     event.preventDefault(); //para no hacer el submit
});
Resetar un formulario FORMULARIO.reset()
CADENAS
ver si una cadena contiene espacios cadena.includes (" ")
ver si una cadena contiene "Azul" cadena.includes("Azul")
ARRAY
añadir un elemento array.push(elemento)
comprobar si un array contiene un elemento
                                               array.includes(elemento)
Se puede imprimir todo un array console.log(array)
Recorrer un array llamado palabras
                                  for (const palabra of palabras){ . . . }
DOM
Añadir/quitar clase CSS elemento.classList.add("CLASE")
                   elemento.classList.remove("CLASE")
Ver si un elemento tiene una clase elemento.classList.contains("CLASE")
Obtener los hijos de un elemento let hijos = elemento.children
```



EJERCICIO1 - FORMULARIO DE ALTA

- Los usuarios dados de alta se guardarán en un array. De partida, hay dos creados: zipi y zape; y se pueden ver debajo. Cuando consigas el alta de un nuevo usuario, lo añades al array y lo muestras.
- En los campos de la contraseña, hay un (botón no puede ser porque hace el submit sin quererlo) para ver en claro la contraseña.
- Los errores los muestras debajo de cada input, como te de la gana.
- Si todo ha ido bien, NO HACER EL SUBMIT; se resetea el formulario y se muestra el nuevo nombre.
- Nombre: no puede estar vacío y no se puede repetir un nombre dado ya de alta.
- Password1: sin espacios y mínimo de 4 caracteres
- Password2: debe coincidir con el primero

PARA EL SPAN DE VER CONTRASEÑA: para que se vea la contraseña, cambiar el tipo del input, de password a text, y viceversa. Se puede acceder y cambiar el tipo con INPUT.type = 'TIPO'









EJERCICIO2 - PARSEAR UN JSON

- Dado el JSON proporcionado, mostrarlo en el HTML de la forma indicada
- Añadir cada empleado como hijo de un div proporcionado en el html <div id="lista-empleados>
- <u>NO SE PUEDE USAR innerHTML</u>. Hay que crear cada una de las etiquetas. La estructura para mostrar cada empleado y que hay que crear, toda ella dinámicamente es:

Datos del JSON

Ana García

• edad: 30

• email: ana.garcia@example.com

Pablo Fernández

• edad: 25

• email: luis.fern@example.com

María López

• edad: 40

• email: maria.lopez@example.com

Anuk Garo

• edad: 50

• email: anuk@example.com



EJERCICIO3 - JUEGO DE PALABRAS

Juego que consiste es buscar palabras.

- Dado un texto, primero indicaremos una palabra a buscar y se nos indicaran las repeticiones de dicha palabra.
- Una vez que sabemos las repeticiones, buscamos dicha palabra en el texto, haciendo click (<u>se pondrá con fondo amarillo</u>). Cuando encuentres todas las repeticiones, HAS GANADO!

Para hacer el ejercicio, lo primero que tienes que hacer es desglosar todo el texto en palabras (usa split) y cada palabra meterla en un para poder crearle el evento del click. Dicho evento comprobará si la palabra clicada es una de las buscadas.

```
const lista_palabras = CADENA.split(" ");
SPAN.addEventListener("click",() => { . . . });
```

```
<div id="div-texto"> == .
    <span>Clara </span>
    <span>miró </span>
    <span>la </span>
    <span>ventana. </span>
    <span>El </span>
    <span>cielo </span>
    <span>estaba </span>
    <span>gris. </span>
    <span>llovía </span>
```

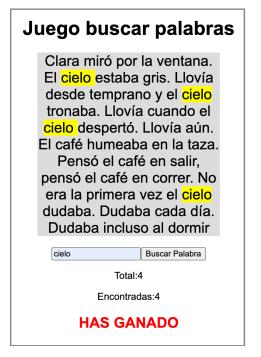
NOTA: no volverse loco con las palabras que tienen una coma junta. No solucionar el problema, salvo que vayas muy muy sobrado de tiempo.

PRUEBA: prueba a buscar la palabra cielo y café

Juego buscar palabras

Clara miró por la ventana.
El cielo estaba gris. Llovía desde temprano y el cielo tronaba. Llovía cuando el cielo despertó. Llovía aún.
El café humeaba en la taza.
Pensó el café en salir, pensó el café en correr. No era la primera vez el cielo dudaba. Dudaba cada día.
Dudaba incluso al dormir







EJERCICIO4 - DESPLEGABLE

Hacer que los párrafos cambien de color según el color escogido en el desplegable.

- El HTML del select esta ya creado, con sus opciones.
- Para simplificar el ejercicio, solo hay 3 colores: violeta, azul y naranja. Usa la propiedad CSS color:violet color:blue y color:orange
- Usar el evento onchange en el desplegable
- Los tres párrafos también están ya creados

Hay un botón que simplemente resetea y lo pone todo como estaba originalmente (quita todas las clases a los párrafos y deja el desplegable con la primera opción).

Para resetear el SELECT y que tenga la primera opción, usar:

ELEMENTO SELECT.selectedIndex=0

