

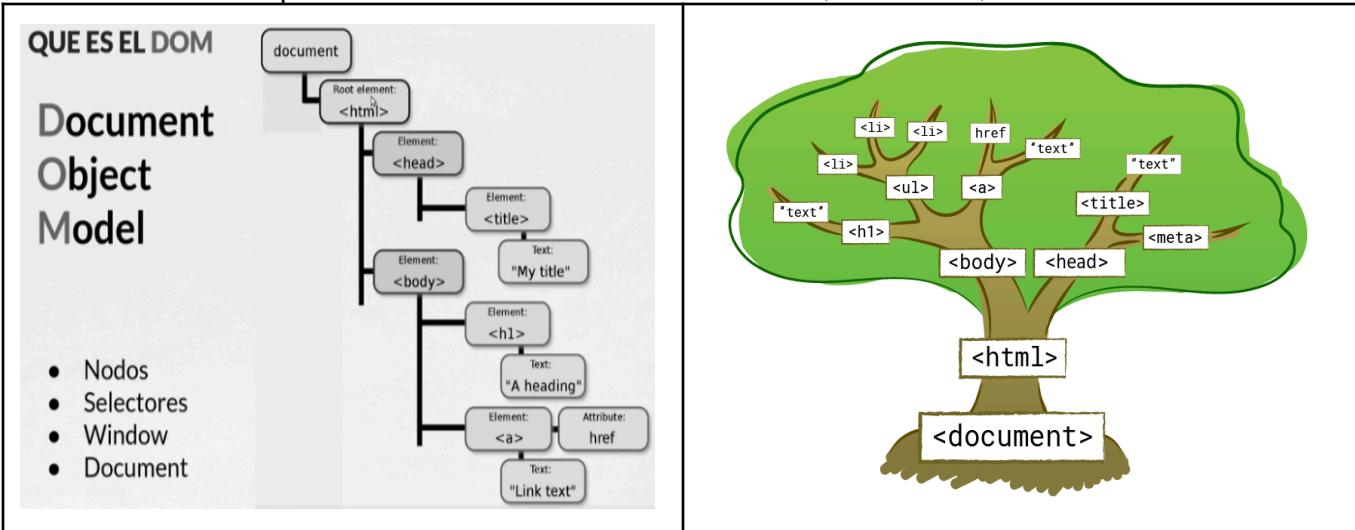
JAVASCRIPT- DOM (Document Object Model)

Extensiones recomendadas para VisualStudioCode

ESLint: para corregir la sintaxis en Javascript.

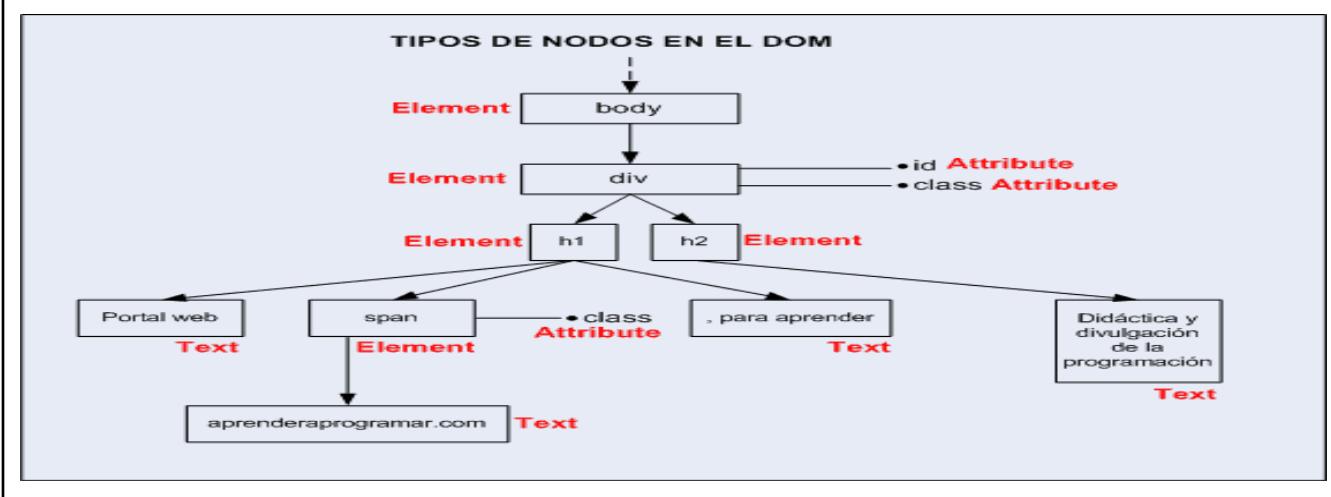
AutoClose: para cerrar las etiquetas no cerradas en html

El modelo de objeto de documento (DOM) es una interfaz de programación para los documentos HTML que se basa en que la página web es un documento dividido en nodos o propiedades a las cuales podemos acceder a través de selectores para realizar acciones como eliminar el nodo, modificarlo, etc.



Veamos un fragmento de código html y su árbol DOM:

```
<body>
  <div id="cabecera" class="brillante">
    <h1>Portal web <span class="destacado">aprenderaprogramar.com</span>, para aprender</h1>
    <h2>Didáctica y divulgación de la programación</h2>
  </div>
</body>
```



Los principales tipos de nodos en el DOM son:

- a) **Document**: el nodo document es el nodo raíz, a partir del cual derivan el resto de nodos.

- b) **Element**: son los nodos definidos por etiquetas html. Por ejemplo, una etiqueta **div** es un elemento que genera un nodo. Si dentro de ese **div** tenemos tres etiquetas **p**, dichas etiquetas son otros elementos/nodos hijos de la etiqueta **div**.
- c) **Text**: el texto dentro de un nodo **element** se considera un nuevo nodo hijo de tipo **text** (texto). Los navegadores también crean nodos tipo texto sin contenido para representar elementos como saltos de línea o espacios vacíos.
- d) **Attribute**: los atributos de las etiquetas definen nodos, aunque trabajando con JavaScript no los veremos como nodos, sino que lo consideraremos información asociada al nodo de tipo **element**.
- e) **Comentarios y otros**: los comentarios y otros elementos como las declaraciones **doctype** en cabecera de los documentos HTML generan nodos.

1. DOCUMENT

La palabra “**document**” representa el nodo superior y por lo tanto la página entera cargada en el navegador. Podemos ver su árbol DOM y todos sus nodos o propiedades:

<pre>1 console.log(document)</pre>	<pre>▼ #document <!DOCTYPE html> <html lang="es"> ▶ <head>...</head> ▶ <body>...</body> </html></pre>
<p>Podemos llamar a uno de sus nodos como por ejemplo al head o al title, etc.:</p> <pre>2 console.log(document.head)</pre>	<pre>▼ <head> <meta charset="UTF-8"> <meta http-equiv="X-UA-Compatible" content="IE=edge"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Document</title> </head></pre>
<p>Si ponemos un title al documento, también podemos ver el nodo title:</p> <pre>3 console.log(document.title)</pre>	<pre>Haciendo pruebas con javascript-DOM</pre>

2. SELECTORES de NODOS

TODO ESTO FUNCIONA SI `<script src="app.js"></script>` está al final del **body**, si está en el **head** no funciona ya que se ejecutaría el archivo javascript antes de que se cargue todo el DOM. (pero se podría hacer de varias maneras, una de ellas es poner `<script src="app.js" defer></script>` para que se cargue todo el dom antes del script)

Para interactuar con alguno de los elementos del “**document**” primero tenemos que seleccionarlo. Los nodos se pueden seleccionar de varias formas:

- Por el nombre de la etiqueta o tag. Por ejemplo `<p>`, `<h1>`, ``, etc.
- Por el nombre de su identificador id. Por ejemplo `<p id="parrafo">`
- Por el nombre de su clase class. Por ejemplo `<h1 class="titulos">`

En función de cómo tengamos identificado un nodo hay varias formas para seleccionarlo:

1. `getElementsByName()` (por nombre de la etiqueta)

Tenemos el "ejemplo1.html" que si picamos en el botón llama a una función del archivo "ejemplo1.js" donde tenemos el código javascript.

```
<body>
  <h1>Título de la página</h1>
  <p>Este es el primer párrafo</p>
  <p>Este es el segundo párrafo</p>
  <p>Este es el tercer párrafo</p>

<button onclick="cambiar()">Cambia la página</button>
<script src="ejemplo1.js"></script>
</body>
```

Una vez seleccionado un elemento podemos ver y/o modificar su contenido (`textContent`) y modificar su estilo (`style`) de la siguiente manera.

```
1 let titulo = document.getElementsByTagName("h1")
2 console.log(titulo)
3 console.log(titulo[0].textContent)
4
5 titulo[0].style.color = "white"
6 titulo[0].style.backgroundColor = "grey"
7 titulo[0].textContent = "CAMBIO DE TITULO"
```

En el archivo `ejemplo1.js` seleccionamos la etiqueta `h1`, la guardamos en una variable.

```
function cambiar(){
let titulo = document.getElementsByTagName("h1")
console.log(titulo)
}
```

Al pulsar en el botón no vemos nada en la página ya que lo mostramos en la consola donde vemos que la etiqueta la guarda como un array con valor [0] ya que solo hay un `h1` en la página.

```
▼ HTMLCollection [h1]
  ► 0: h1
    length: 1
  ► [[Prototype]]: HTMLCollection
```

CAMBIO DE TITULO

Esto en el primer párrafo

Esto en el segundo párrafo

Esto en el tercer párrafo

Ahora vamos a seleccionar la etiqueta `<p>`, pero vemos que hay tres etiquetas `<p>` por lo que creará un array con tres elementos [0,1,2]. Vamos a cambiar el contenido del párrafo segundo, el color a rojo de letra del tercero y el color de fondo a azul del primero.

```
let parrafos = document.getElementsByTagName("p")
console.log(parrafos)
parrafos[1].textContent = "HE MODIFICADO EL CONTENIDO"
parrafos[2].style.color = "red"
parrafos[0].style.backgroundColor = "blue"
}
```

En consola vemos que hay 3 elementos en el array

```
▼ HTMLCollection(3) [p, p, p]
  ► 0: p
  ► 1: p
  ► 2: p
    length: 3
  ► [[Prototype]]: HTMLCollection
Esto en el segundo párrafo
```

Título de la página

Este es el primer párrafo

HE MODIFICADO EL CONTENIDO

Este es el tercer párrafo

Pregunta: ¿qué crees que hace la siguiente sentencia?

```
var nombreVariable = document.getElementsByTagName("a");
```

Ejercicio1

Sobre el ejercicio anterior, si pulsamos en el botón se debe cambiar el texto del título a "EJERCICIO1 JAVASCRIPT", modificar el color de fondo de todos los párrafos a color amarillo y su texto a gris, el tamaño de texto del tercer párrafo a 40px y su estilo a cursiva.

`style.fontStyle= "italic"
style.fontSize= "XXpx"`

EJERCICIO1 JAVASCRIPT

Este es el primer párrafo

Este es el segundo párrafo

Este es el tercer párrafo

[Cambia la página](#)

2. getElementById() (por nombre del id)

Recordamos que "id" sirve para identificar un único nodo en una página web. Por ello en javascript al seleccionar un elemento por su id solo seleccionamos única y exclusivamente ese nodo. Al seleccionar solo un nodo **no crea un array**.

En el ejemplo anterior, agregamos un id al "h1"

```
<body>
  <h1 id="titulo1">Título de la página</h1>
  <p>Este es el primer párrafo</p>
  <p>Este es el segundo párrafo</p>
  <p>Este es el tercer párrafo</p>

  <button onclick="cambiar()">Cambia la página</button>
  <script src="ejemplo1.js"></script>
</body>
```

Código Javascript cambiamos el contenido, el color y el color de fondo del h1.

```
function cambiar(){
let titulo1 =document.getElementById("titulo1")
console.log(titulo1)
console.log(titulo1.textContent)

titulo1.style.textContent ="ESTE ES EL NUEVO TITULO"
titulo1.style.color = "white"
titulo1.style.backgroundColor = "grey"
}
```

La consola

```
<h1 id="titulo1">ESTE ES EL NUEVO TITULO</h1>
Título de la página
```

La página web

```
ESTE ES EL NUEVO TITULO
Este es el primer párrafo
Este es el segundo párrafo
Este es el tercer párrafo
Cambia la página
```

3. querySelector() (por nombre de la etiqueta, id o por class) (NO NECESARIO VER)

Este selector puede seleccionar un nodo por cualquier identificador, es decir por el nombre de la etiqueta, por su id o por su class. El problema es que solo selecciona el primer elemento encontrado, es decir que **NO crea un array**, por ello igual no es tan importante ya que se puede sustituir por otros selectores mejores que sí crean un array.

Seguimos con el ejemplo anterior, agregamos a nuestra página el siguiente código que tiene identificadores **id** y **class**:

```

<body>
    <h1 id="titulo1">Título de la página</h1>
    <p class="parrafos">Este es el primer párrafo</p>
    <p class="parrafos">Este es el segundo párrafo</p>
    <p class="parrafos">Este es el tercer párrafo</p>

    <button onclick="cambiar()">Cambia la página</button>
    <script src="ejemplo1.js"></script>
</body>

```

Para seleccionar por el **id** ponemos la # (cómo css). Se selecciona el primer elemento encontrado

```

function cambiar(){
let titulo1 =document.querySelector("#titulo1")
console.log(titulo1)
console.log(titulo1.textContent)

titulo1.style.color = "white"
titulo1.style.backgroundColor = "grey"
}

```

ESTE ES EL NUEVO TITULO

Este es el primer párrafo
Este es el segundo párrafo
Este es el tercer párrafo

[Cambia la página](#)

Para seleccionar por la **class** ponemos el punto. (cómo css). Se selecciona el primer elemento encontrado

```

function cambiar(){
let parrafos =document.querySelector(".parrafos")
console.log(parrafos)
console.log(parrafos.textContent)

parrafos.style.color = "white"
parrafos.style.backgroundColor = "grey"
}

```

Título de la página

Este es el primer párrafo
Este es el segundo párrafo
Este es el tercer párrafo

[Cambia la página](#)

4. **querySelectorAll** (por nombre de la etiqueta, id o por class)

Selecciona todos los elementos que encuentre con ese identificador y por ello **crea un array** con todos los elementos seleccionados.

En este caso podemos hacer un **for** para recorrer todo el array y cambiar todos los elementos, ya que si son muchos hacerlo de uno en uno no es viable.

```

function cambiar(){
let parrafos =document.querySelectorAll(".parrafos")

for (let index = 0; index < parrafos.length; index++) {
    parrafos[index].style.color = "white"
    parrafos[index].style.backgroundColor = "grey"
}
}

```

En consola vemos el array con tres elementos

```

▼ NodeList(3) [p.parrafos, p.parrafos, p.parrafos]
  ▷ 0: p.parrafos
  ▷ 1: p.parrafos
  ▷ 2: p.parrafos
  length: 3
  ▷ [[Prototype]]: NodeList
  Esto en el primer párrafo

```

Título de la página

Este es el primer párrafo
Este es el segundo párrafo
Este es el tercer párrafo

[Cambia la página](#)

Nota: también se puede usar **getElementsByClassName()**

Ejercicio2 usando `GetElementsByClassName()`

Sobre el siguiente código `html`, al hacer click en el botón se quiere conseguir la página de abajo derecha usando arrays cuando haya varios elementos iguales y for para recorrerlos. Podéis agregar los selectores que queráis.

```
<h1>IMPLANTACIÓN DE APLICACIONES WEB</h1>
<h2>JAVASCRIPT</h2>
    <p>Este es el primer párrafo</p>
    <p>Este es el segundo párrafo</p>
    <p>Este es el tercer párrafo</p>
<h2>PHP</h2>
    <p>Este es el cuarto párrafo</p>
    <p>Este es el quinto párrafo</p>
    <p>Este es el sexto párrafo</p>
<button id="cambiar">Cambia la página</button>
```

IMPLANTACIÓN DE APLICACIONES WEB	IMPLANTACIÓN DE APLICACIONES WEB
JAVASCRIPT	JAVASCRIPT
Este es el primer párrafo	<i>Este es el primer párrafo</i>
Este es el segundo párrafo	<i>Este es el segundo párrafo</i>
Este es el tercer párrafo	<i>Este es el tercer párrafo</i>
PHP	PHP
Este es el cuarto párrafo	<i>Este es el cuarto párrafo</i>
Este es el quinto párrafo	<i>Este es el quinto párrafo</i>
Este es el sexto párrafo	<i>Este es el sexto párrafo</i>
<input type="button" value="Cambia la página"/>	<input type="button" value="Cambia la página"/>

3. NODOS/ELEMENTOS. CREACIÓN-MODIFICACIÓN-ELIMINACIÓN

Podemos agregar un nuevo nodo o elemento al final de nuestra página `html` que puede ser un párrafo, un nuevo valor de lista, una imagen, etc. Los pasos son los siguientes:

1. Primero creamos el elemento

```
var nuevoNodo = document.createElement("etiqueta_html_del_nodo");
```

2. Luego agregamos el contenido del elemento. Hay dos formas. Es mejor la primera ya que de esta manera podemos agregar html al contenido como luego veremos

```
nuevoNodo.innerHTML ="Nuevo <b>texto</b>" //nuevoNodo.textContent="Nuevo texto"
```

3. Por último agregamos el elemento como hijo de la página(body) al final de la misma.

```
document.body.appendChild(nuevoNodo);
```

Si es una imagen no habrá texto, pero sí hay atributo de la imagen:

```
var img = document.createElement("img");
img.src = "imagen.jpg";
```

```
document.body.appendChild(img);
```

Si es un enlace hay texto y atributo:

```
let enlace = document.createElement("a");
enlace.textContent="texto del enlace"
enlace.setAttribute("href", "url")
document.body.appendChild(enlace);
```

Ejercicio3-a

Al ejercicio anterior le creamos un segundo botón que si lo picamos agregamos 3 cosas nuevas: un título h2, un párrafo y una imagen.

```
function agregar(){
    //Creamos un nuevo nodo que es un h2
    var titulonuevo = document.createElement("h2");
    //Agregamos el contenido al nodo anterior. Hay dos formas:
    titulonuevo.innerHTML="SEGURIDAD" //titulonuevo.textContent="SEGURIDAD"
    //Agregamos si queremos estilos al elemento
    titulonuevo.style.TextAlign ="center"
    titulonuevo.style.color="red"
    //Añadimos el nuevo nodo(h2) a la página, se agrega al final
    document.body.appendChild(titulonuevo);

    //Agregamos ahora un párrafo
    var parrafonuevo = document.createElement("p");
    parrafonuevo.innerHTML="Nuevo <b><u>párrafo</u></b>" //parrafonuevo.textContent="Nuevo párrafo"
    document.body.appendChild(parrafonuevo);

    //agregamos una imagen
    var img = document.createElement("img");
    img.src = "imagen.jpg";
    document.body.appendChild(img);
}
```

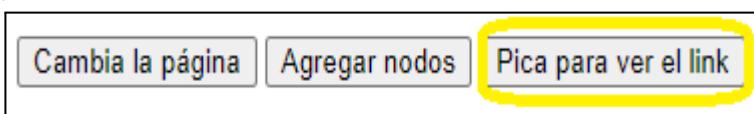
Cambia la página Agregar nodos

SEGURIDAD

Nuevo párrafo



Agrega ahora un tercer botón para si lo pulsamos que aparezca un enlace al final de la página que lleve a la página del centro Sanluis.



CREAR Y AGREGAR UN ELEMENTO EN CUALQUIER LUGAR DE LA PÁGINA

Hemos visto crear elementos o nodos al final de la página con `document.body.appendChild`, pero en muchos casos queremos crearlos en otro lugar de la página o incluso crear un elemento que sustituya a otro que ya está en la página.

En este caso los pasos son parecidos a los anteriores, pero en vez de agregar el nodo al body, crearemos un nuevo nodo vacío(`<div></div>`) y lo agregaremos ahí. Lo vemos...

Ejercicio3-b

En el ejercicio anterior queremos agregar a la página un cuarto párrafo en javascript y en PHP.

En la página HTML vamos a agregar dos div vacíos, pero identificados con un id cada uno.

En el JS, cuando agreguemos el nodo no lo haremos al body, sino a este elemento <div> vacío correspondiente.

En el HTML agregamos dos <div> vacíos con id

```
<body>
    <h1 id="titulo1">IMPLANTACIÓN DE APLICACIONES WEB</h1>
    <h2 id="titulo21">JAVASCRIPT</h2>
    <p class="parrafos1">Este es el primer párrafo</p>
    <p class="parrafos1">Este es el segundo párrafo</p>
    <p class="parrafos1">Este es el tercer párrafo</p>
    <div id="nuevoparrafoJS"></div>
    <h2 id="titulo22">PHP</h2>
    <p class="parrafos2">Este es el cuarto párrafo</p>
    <p class="parrafos2">Este es el quinto párrafo</p>
    <p class="parrafos2">Este es el sexto párrafo</p>
    <div id="nuevoparrafoPHP"></div>
<button onclick="cambiar()">Cambia la página</button>
<button onclick="agregar()">Agregar nodos</button>
<button onclick="enlace()">Pica para ver el link</button>
```

En JAVASCRIPT agregamos los dos nodos a esos <div> vacíos.

```
//Agregamos ahora un párrafo JS
var parrafonuevoJS = document.createElement("p");
parrafonuevoJS.innerHTML="Nuevo párrafo de <u>Javascript</u>"
//parrafonuevoJS.textContent="Nuevo párrafo de Javascript" Otra forma
parrafonuevoJS.style.color="white"
parrafonuevoJS.style.backgroundColor="red"
document.getElementById("nuevoparrafoJS").appendChild(parrafonuevoJS);
//Agregamos ahora el otro párrafo PHP
var parrafonuevoPHP = document.createElement("p");
parrafonuevoPHP.innerHTML="Nuevo párrafo de <u>PHP</u>"
//parrafonuevoPHP.textContent="Nuevo párrafo de PHP" Otra forma
parrafonuevoPHP.style.color="white"
parrafonuevoPHP.style.backgroundColor="red"
document.getElementById("nuevoparrafoPHP").appendChild(parrafonuevoPHP);
```

IMPLANTACIÓN DE APLICACIONES WEB

JAVASCRIPT

Este es el primer párrafo

Este es el segundo párrafo

Este es el tercer párrafo

Nuevo párrafo de Javascript

PHP

Este es el cuarto párrafo

Este es el quinto párrafo

Este es el sexto párrafo

Nuevo párrafo de PHP

[Cambia la página](#) [Agregar nodos](#) [Pica para ver el link](#)

Ejercicio3-b: ahora práctica tú agregando la imagen anterior entre el JAVASCRIPT Y EL PHP y que esté centrada (cuidado, hay que centrar el <div>, no la imagen "").

MODIFICACIÓN DE UN NODO

En este caso no hay que crear el nodo, solo es necesario seleccionar el nodo a modificar y luego utilizamos alguna de las dos sentencias ya vistas para agregar el contenido y se quita el anterior contenido.

```
elemento.innerHTML="nuevo <br>texto</br>";
elemento.textContent="nuevo texto";
```

ELIMINACIÓN DE UN NODO

Para eliminar un nodo se puede hacer de varias formas, una manera sencilla es seleccionar el nodo "padre" del hijo que queremos eliminar y el propio nodo "hijo", y luego usando `padre.removeChild("hijo");` eliminamos el nodo hijo. Sería así:

```
var padre = document.getElementById("padreEliminar");
var hijo = document.getElementById("nodoEliminar");
padre.removeChild(hijo)
```

Ejercicio3-c

Copiamos el siguiente código html con nodos que queremos modificar y eliminar.

```
<h1 id="titulo1">MODIFICAR/ELIMINAR NODOS</h1>
<table width="50%">
    <tr>
        <td>
            <ol>
                <li>Primer elemento de la lista</li>
                <li>Segundo elemento de la lista</li>
                <li>Tercer elemento de la lista</li>
                <li>Cuarto elemento de la lista</li>
                <li>Quinto elemento de la lista</li>
            </ol>
            <button onclick="cambiar()">Cambia nodo</button>
        </td>
        <td>
            <ul>
                <li>Sexto elemento de la lista</li>
                <li>Séptimo elemento de la lista</li>
                <li>Octavo elemento de la lista</li>
                <li>Noveno elemento de la lista</li>
                <li>Décimo elemento de la lista</li>
            </ul>
            <button onclick="eliminar()">Elimina nodo</button>
        </td>
    </tr>
</table>
```

La página antes

MODIFICAR/ELIMINAR NODOS

- 1. Primer elemento de la lista
- 2. Segundo elemento de la lista
- 3. Tercer elemento de la lista
- 4. Cuarto elemento de la lista
- 5. Quinto elemento de la lista

Cambia nodo

- Sexto elemento de la lista
- Séptimo elemento de la lista
- Octavo elemento de la lista
- Noveno elemento de la lista
- Décimo elemento de la lista

Elimina nodo

La página después

MODIFICAR/ELIMINAR NODOS

- 1. Nuevo primer elemento de la lista
- 2. Segundo elemento de la lista
- 3. Tercer elemento de la lista
- 4. Cuarto elemento de la lista
- 5. Nuevo quinto elemento de la lista

Cambia nodo

- Sexto elemento de la lista
- Octavo elemento de la lista
- Noveno elemento de la lista
- Décimo elemento de la lista

Elimina nodo

SE PIDE

Si pulsamos el botón "**Cambia nodo**" se quiere sustituir el primer y quinto elemento por otro con texto en rojo.
Si pulsamos el botón "**Elimina nodo**" se debe eliminar el séptimo nodo.

Solución (intentar hacerlo sin mirar)

Ejercicio3c ampliado repaso: Se quiere crear un nuevo botón que ponga "agregar nodos" y que agregue un nuevo nodo entre el 2 y el 3 y otro entre el 8 y 9

Ejercicio: Sorteo de las posiciones de los grupos Ethazi.

Tenemos cinco grupos de Ethazi y se quiere sortear como sentarse y ubicarse en el aula. Para ello hay cinco formas de sentarse que vemos en la siguiente imagen de la página.

Sorteo de las posiciones de los grupos de clase-Ethazi

Pica para el Sorteo

- 1.
- 2.
- 3.
- 4.
- 5.

En la siguiente distribución de clase se mostrará por colores la posición de los distintos equipos Ethazi. SUERTE!!!!

PROFESOR

Asiento1	Asiento2	Asiento3	Asiento4	Pasillo	Asiento5	Asiento6	Asiento7	Asiento8
----------	----------	----------	----------	---------	----------	----------	----------	----------

Asiento9	Asiento10	Asiento11	Asiento12	Pasillo	Asiento13	Asiento14	Asiento15	Asiento16
----------	-----------	-----------	-----------	---------	-----------	-----------	-----------	-----------

Asiento17	Asiento18	Asiento19	Asiento20	Asiento21	Asiento22	Pasillo	Asiento23	Asiento24	Asiento25
-----------	-----------	-----------	-----------	-----------	-----------	---------	-----------	-----------	-----------

Asiento25

Si picamos en el botón "Pica para el Sorteo" se sortean de forma aleatoria los cinco grupos, se indican al principio de la página como ha quedado el sorteo por orden y color y se colocan en las posiciones A,B,C,D y D. El resultado es el siguiente:

Sorteo de las posiciones de los grupos de clase-Ethazi

Pica para el Sorteo

1. Equipo3
2. Equipo5
3. Equipo4
4. Equipo1
5. Equipo2

En la siguiente distribución de clase se mostrará por colores la posición de los distintos equipos Ethazi. SUERTE!!!!

PROFESOR

Equipo3	Equipo3	Equipo3	Equipo3	Pasillo	Equipo3	Equipo1	Equipo1	Equipo1
---------	---------	---------	---------	---------	---------	---------	---------	---------

Equipo5	Equipo5	Equipo5	Equipo5	Pasillo	Equipo5	Equipo2	Equipo1	Equipo1
---------	---------	---------	---------	---------	---------	---------	---------	---------

Equipo4	Equipo4	Equipo4	Equipo4	Equipo4	Equipo2	Pasillo	Equipo2	Equipo2	Equipo2
---------	---------	---------	---------	---------	---------	---------	---------	---------	---------

Asiento25

4. EVENTOS

Un evento es una respuesta que se ejecuta ante una acción determinada. Un evento puede ser por ejemplo:

- Terminar de cargarse la página en el navegador.
- El usuario selecciona, hace clic o pasa el ratón por encima de cierto elemento.
- El usuario presiona una tecla del teclado, suelta la tecla, etc.
- El usuario redimensiona o cierra la ventana del navegador.
- Enviar un formulario.
- Reproducir un video, pausarlo o terminarlo.

Hay dos formas de crear un evento:

1. EVENTOS que se declaran en el archivo HTML (Empiezan con ON.....)

`onclick, onmouseover, onmouseout, onload, ondblclick, etc.`

Los eventos que empiezan con ON... los debemos de declarar dentro de la página html. Algunos son:

```
<!-- Despues de cargar toda la página buscar la función saludar-->
<body onload="saludar()">
<!-- al pulsar el botón, ejecutar la función cambiar(ya vista)-->
<button onclick="cambiar()">Cambia la página</button>
<!-- Otra forma de crear un botón y al pulsar ejecutar una función-->
<input type="Button" value="Compara números" onclick="calculoMayor()">
<!-- Eventos al pasar y al quitar el ratón sobre una imagen-->

```

Luego en el documento Javascript simplemente ponemos la función declarada y las acciones que realiza.

```
function calculoMayor(){
    //acciones a ejecutar
}
```

2. EVENTOS declarados en el archivo JAVASCRIPT. No se usa ON, se usa addEventListener

Se recomienda usar los eventos de esta forma para no mezclar el html con el javascript. Estos eventos se crean dentro del Javascript para asociarlo a algún elemento del HTML. En este caso para seleccionar el elemento del HTML debemos de identificarlo (`id, class, etc`).

Por ejemplo, si tuviéramos un ejercicio de comparar dos números al pulsar un botón, en la página HTML no pondríamos el evento `onclick="calculoMayor()`, en cambio usamos `id = "comparar"` para identificar el elemento y luego usarlo.

```
<input type="Button" value="Compara números" id="comparar">//no pongo onclick
<button id="comparar">Compara los números</button>//otra forma
```

En el documento Javascript hacemos lo siguiente:

```

let elemento=document.getElementById("comparar") //guardamos el elemento en una variable
elemento.addEventListener("tipoEvento", nombreFuncion) //le añadimos un evento de tal
forma que si se acciona se ejecuta una función
function nombreFuncion(){
    // acciones a ejecutar en la función del evento
}

```

- Elemento: es la variable donde hemos guardado el elemento seleccionado.
- TipoEvento: es lo que queremos hacer, por ejemplo: **click, mouseover, mouseout, dblclick**, etc. **No se usa ON (onclik, onmouseover,....)**
- NombreFuncion: es la función a la que llamamos que indicará lo que hay que hacer. Esta función puede tener o no parámetros. Si no tiene parámetros no se pone paréntesis.

Otro ejemplo: si queremos que cambie algo cuando pasamos el ratón por encima de un nodo, sería así:

```

var cambiar =document.getElementById("cambiar") //guardamos el elemento en una variable
cambiar.addEventListener("mouseover", cambio) //sobre La variable creamos el evento

function cambio(){
    // acciones a ejecutar en la función del evento
}

```

Eventos del ratón (click, mouseover, mouseout, dblclick)

Suelen ser los eventos más usados. Algunos de ellos son:

- **mousedown/mouseup**: mousedown cuando se pulsa el botón del ratón sobre un elemento sin soltarlo y mouseup cuando se suelta.
- **click**: se activa después de mousedown y mouseup seguido sobre el mismo elemento.
El orden de los eventos cuando pulsamos el ratón es **mousedown/mouseup/click**
- **dblclick**: se activa después de dos clicks seguidos sobre el mismo elemento. Se usa raramente.
- **mouseover/mouseout**: se activa cuando el puntero del mouse entra/sale de un elemento.
- **mousemove**: cualquier movimiento del mouse sobre un elemento activa el evento.
- **contextmenu**: Se activa al pulsar el botón derecho del ratón

Ejercicio4: vamos a hacer tres eventos en la siguiente página html:

- **Evento1**: que al cargarse toda la página que salude con una alerta.
- **Evento2**: que al situar el ratón sobre el título h1 que cambie el color del texto y fondo del h1 de forma aleatoria entre 10 colores que no coincidan, y al quitar el ratón que vuelva a la posición normal.
- **Evento3**: que al picar con el ratón sobre la imagen que cambie por otra imagen, y al dejar de picar el ratón vuelve a la imagen inicial.
- **Evento4**: un botón que si lo picamos cambia el color del fondo de la página de forma aleatoria

Copiamos la siguiente página html y las imágenes correspondientes:

```

<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejercicio4</title>

```

```

<style>
    body {text-align: center;}
    img{width: 300px; height: 200px;}
</style>
</head>
<body>
    <h1> Sitúa el ratón encima y quítalo!!!! </h1>
    <br>
    <h2>El gato cambia al picar con el mouse o quitarlo</h2>
    
    <br><br>
    <input type="button" value="Cambia el color de fondo">

</body>
</html>

```

La página se verá de la siguiente manera:

Sitúa el ratón encima y quítalo!!!!

El gato cambia al picar con el mouse o quitarlo



Evento1 solución: usamos en la página html el evento `onload="funcion()"` que se usa para cuando se carga la página, y en el archivo javascript ponemos la función.

<code><body onload="saludar()"></code>	<code>function saludar(){ alert("Hola, gracias por visitar la página") }</code>
Otra Solución: <code><body></code>	<code>window.addEventListener("load", saludo); function saludo(){ alert("Hola buenos días, ya se ha cargado la página") }</code>

Evento2 solución: usamos **mouseover** y **mouseout** para situar el ratón y quitarlo sobre un elemento (h1). No hay que hacer nada en la página html, excepto identificar el h1 con un id.

```
//Creamos una matriz de 10 colores para poder variar el color de forma aleatoria
let matrizColores = ["#000000", "#FF0000", "#00FF00", "#0000FF", "#FFFF00", "#00FFFF", "#FF00FF", "#999999", "#FF99FF", "#66FF99"];

//seleccionamos el h1 y le agregamos dos eventos con una función para cada evento
let titulo1 = document.getElementById("titulo1")
titulo1.addEventListener("mouseover", cambioColor)
titulo1.addEventListener("mouseout", noCambioColor)

function cambioColor(){
    //definimos la función ponerRaton para cuando ponemos el ratón encima
    indice = Math.floor(Math.random()*10); //aleatoriedad entre 0 y 10
    titulo1.style.color=matrizColores[indice];
    titulo1.style.fontSize="30";
    titulo1.style.backgroundColor=matrizColores[indice+1];
    titulo1.style.fontFamily="Courier";
    titulo1.style.textAlign = "center";
}

function noCambioColor(){
    //definimos la función quitarRaton para cuando quitamos el ratón
    indice = Math.floor(Math.random()*10); //aleatoriedad entre 0 y 10
    titulo1.style.color="";
    titulo1.style.fontSize="";
    titulo1.style.backgroundColor="#ffffff";
    titulo1.style.fontFamily="";
}
```

Evento3 solución: ahora los eventos son **mousedown** y **mouseup** para el nodo de la imagen.

```
//Ahora hacemos algo parecido con dos imágenes de un gato
let gato =document.getElementById("gato")
gato.addEventListener("mousedown", cambioImagen)
gato.addEventListener("mouseup", volverImagen)

function cambioImagen(){
    gato.src = "gatoalegre.jpg";
    gato.width = 300;
    gato.height = 200;
}

function volverImagen(){
    gato.src = "gatoenfadado.jpg";
    gato.width = 300;
    gato.height = 200;
}
```

Evento4 solución: si picamos en el botón cambia de color de fondo de la página. El evento es **click** o **mouseover**.

```
let colorFondo = document.getElementById("colorFondo");
colorFondo.addEventListener("click", cambioColorRGB);

function cambioColorRGB(){
    let red= Math.floor(Math.random()*256);
    let green= Math.floor(Math.random()*256);
    let blue= Math.floor(Math.random()*256);
    //La sintaxis para generar un color en rgb: rgb($red, $green, $blue);
    let rgbColor =`rgb(${red},${green},${blue})`;
    console.log(rgbColor); //por si lo queremos ver el rgb en consola
    document.body.style.backgroundColor = rgbColor;
}
```

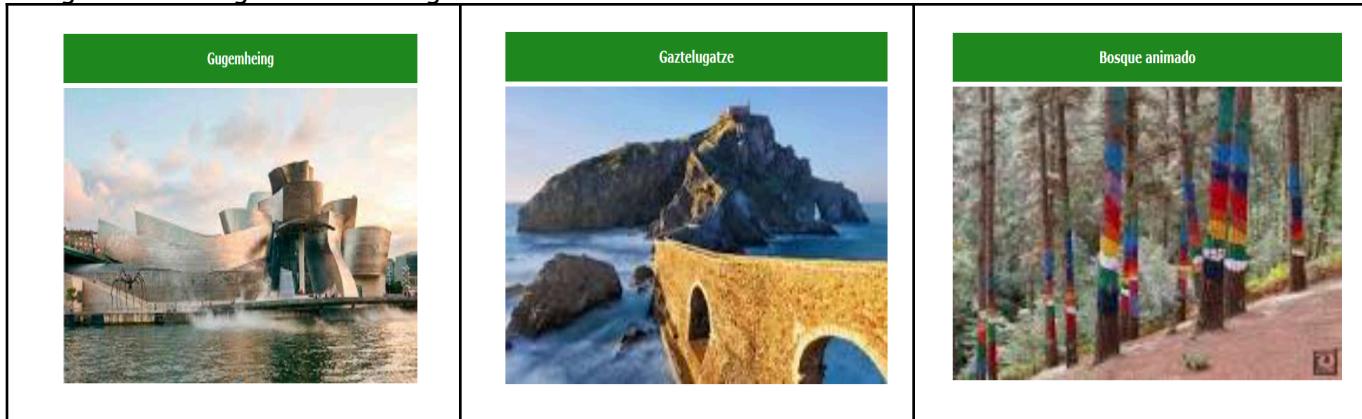
Ejercicio 5a: setInterval, clearInterval(idSetInterval) y setTimeout

Podemos jugar con los tiempos de ejecución de una función en Javascript usando estas sentencias

SetInterval(Carrusel de imágenes): permite ejecutar una función cada cierto tiempo(ms), es decir generar repetición o bucle en la ejecución de una función. Por ejemplo, creamos una función para mostrar una imagen aleatoria un tiempo y volver a ejecutar la misma función y así repetidamente consiguiendo un carrusel de imágenes que cambian cada cierto tiempo.

Ejemplo: **setInterval("nombreFuncion()", 5000);** Cada 5 sg ejecutamos la función indicada

Descargáros de Alexia el ejercicio de "Carrusel de imágenes de Bizkaia". Queremos cambiar el título y la imagen de esos lugares cada 2 segundos:



El archivo Javascript

```
// repetimos la función "cambiarFoto" cada 2 segundos
setInterval("cambiarFoto()", 2000);

function cambiarFoto() {
    let lugar = ["Gaztelugatxe", "Costa", "Plentzia", "Guggenheim", "Butrón", "Algorta", "Bosque animado"];
    let fotos = ["gaztelugatxe.jpg", "costa.jpg", "plentzia.jpg", "museo.jpg", "castillo.jpg", "algorta.jpg", "bosque.jpg"];
    // calculo de un numero aleatorio de tipo entero entre 0 y 9
    let numAleatorio = Math.floor(Math.random()*7);
    //seleccionamos el título y la imagen con sus id
    let titulo = document.getElementById("titulo");
    let imagen = document.getElementById("imagen");
    titulo.innerHTML = lugar[numAleatorio];
    imagen.src = "img/" + fotos[numAleatorio];
}
```

clearInterval(idSetInterval): lo usamos si por algún motivo quisiéramos detener la repetición. Por ejemplo en el caso anterior podríamos parar la repetición si picamos en un botón o si se cumple una condición .

Ejercicio 5a anterior. Podemos ampliar el ejercicio para que si ha hecho un bucle repetitivo 10 veces que cambie a una foto y título estático. Debemos de identificar el setInterval con una variable y luego controlar con un contador en número de veces.

```
// repetimos la función "cambiarFoto" cada 2 segundos
let carrusel=setInterval("cambiarFoto()", 2000);

let i=0
function cambiarFoto() {
    let lugar = ["Gaztelugatze", "Costa", "Plentzia", "Gugemheing", "Butrón", "Algorta", "Bosque animado"];
    let fotos = ["gaztelugatxe.jpg", "costa.jpg", "plentzia.jpg", "museo.jpg", "castillo.jpg", "algorta.jpg", "bosque.jpg"];
    // calculo de un numero aleatorio de tipo entero entre 0 y 9
    let numAleatorio = Math.floor(Math.random()*7);
    //seleccionamos el título y la imagen con sus id
    let titulo = document.getElementById("titulo");
    let imagen = document.getElementById("imagen");
    titulo.innerHTML = lugar[numAleatorio];
    imagen.src = "img/" + fotos[numAleatorio];
    i++
    if (i==10) {
        titulo.innerHTML = "SE TERMINÓ EL BUCLE"
        imagen.src= "img/fin.jpg"
        clearInterval(carrusel)
    }
}
```

SE TERMINÓ EL BUCLE



Otro ejemplo práctico del caso anterior sería poner un contador hacia atrás de 10 imágenes de 10 a 0 y cuando llega a 0 que se cargue otra página, o que se vea una imagen o que ocurra otra cosa.

Ejercicio 5b: setTimeout: Juego trivial

setTimeout sirve para temporizar una ejecución o carga de la página, es decir permite esperar un tiempo antes de ejecutar una función, es decir que actúa como un temporizador.

setTimeout(función, tiempoEnMilisegundos);

En este ejercicio nos muestra la siguiente página (imagen1) donde si picamos en el botón "Ver Pregunta" nos muestra una pregunta y una imagen de un contador atrás de 10 segundos para pensar la respuesta (imagen2). A los 10 segundos se muestra un texto y un input para meter la respuesta y enviarla (imagen3). NO HAY QUE HACER MÁS, NO HAY QUE ENVIAR LA RESPUESTA!!!

Juego trivial

Vamos a jugar al trivial. Tienes que contestar a una pregunta bien y te llevas el premio. Tendrás 10 segundos para pensar la respuesta y luego contestar rápidamente. SUERTEE!!

[Ver Pregunta](#)

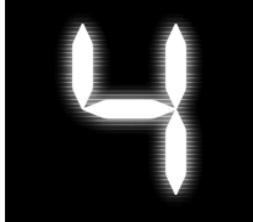
Juego trivial

Vamos a jugar al trivial. Tienes que contestar a una pregunta bien y te llevas el premio. Tendrás 10 segundos para pensar la respuesta y luego contestar rápidamente. SUERTEE!!

[Ver Pregunta](#)

Escribe en qué ciudad estás:

1. Bilbao
2. Basauri
3. Barakaldo



Juego trivial

Vamos a jugar al trivial. Tienes que contestar a una pregunta bien y te llevas el premio. Tendrás 10 segundos para pensar la respuesta y luego contestar rápidamente. SUERTEE!!

[Ver Pregunta](#)

Escribe en qué ciudad estás:

1. Bilbao
2. Basauri
3. Barakaldo



TU RESPUESTA ES: [Enviar Respuesta](#)

5. TRATAMIENTO DE LOS DATOS DE UN INPUT (NO FORMULARIO)

Ya sabemos que un input es una caja donde el usuario introduce un dato. Ese dato se puede tratar de diferentes formas dependiendo de la finalidad del input. Por ello vamos a distinguir entre los inputs de un formulario y los que no lo son. En ambos casos suele haber un botón para realizar algo con el contenido de dichos inputs.

INPUTS DE FORMULARIO

Los datos introducidos en los inputs de un formulario se suelen utilizar para enviarlos probablemente al servidor, para tratarlos en una base de datos, todo ello usando el lenguaje PHP. Por ejemplo dar de alta o registrar a un usuario en una página.

En este caso se tiene que usar las etiquetas `<form action="xx" method="xxx"> </form>` y enviar el formulario usando el atributo `action=" "` (ya se verá con PHP) y poniendo en el button un `<button type="submit">` y se envía el formulario al servidor. Pero antes de enviar los datos al servidor es importante validarlos en el cliente (navegador), y esto se puede hacer con Javascript.

INPUTS QUE NO SON DE UN FORMULARIO

Hay otros datos introducidos en inputs que no requieren ser tratados con PHP en una BBDD, es decir que no se tratan como un formulario, por lo tanto no es necesario usar las etiquetas `<form></form>`. Estos inputs recogen datos que se tratan dentro del navegador para obtener algún resultado e interactuar con la página sin que actúe el servidor. Por ejemplo suelen utilizarse para cuando picamos en un botón realizar algún evento o normalmente para realizar alguna operación con el contenido de los inputs, como por ejemplo alguna operación matemática, alguna validación como el ejercicio anterior del trivial, etc.

SELECCIÓN DE LOS DATOS, VALOR O CONTENIDO DE UN INPUT

No es lo mismo seleccionar la caja del input, que su contenido. Lo vemos con un ejemplo en el cual tenemos un input para introducir texto correspondiente al nombre de un usuario:

```
<input type="text" size="20" id="username" name="username" required placeholder="Ingrese su usuario">
```

No es lo mismo seleccionar el input para cambiarle el color de texto, color de fondo, etc., que seleccionar su contenido o valor para luego tratarlo como corresponda. Los dos casos son:

```
let inputUsuario = document.getElementById("username") // Seleccionamos el input  
let valorUsuario = document.getElementById("username").value // Seleccionamos su contenido
```

Ejercicio6 (NO formulario): Eventos en inputs (click, focus, blur)

Tenemos una página donde queremos obtener el resultado de la suma de dos números que inicialmente hemos introducido en dos inputs. Al picar en el botón sumar se calcula la suma y se pone en la celda del resultado, pero solo podemos introducir números, sino nos da una alerta del error.

```
isNaN(numero) // Devuelve true si el valor no es un número y false si lo es.
```

SUMAR DOS NÚMEROS

Número 1:	<input type="text" value="8"/>
Número 2:	<input type="text" value="5"/>
Resultado:	<input type="text" value="13"/>
	<input type="button" value="Sumar"/>

Además en este ejercicio aprovechamos a ver los eventos "focus" y "blur" que son eventos que se activan cuando se enfoca en algún elemento concreto del html como un input, textarea, button, checkbox, file, password, radio, reset y submit. Su funcionamiento es igual que el resto de eventos.

```
<body>
<center><h1>SUMAR DOS NÚMEROS</h1></center>
<table width="40%" align="center">
<tr>
  <td>Número 1:</td>
  <td><input type="text" size="20" maxlength="30" placeholder="Ingrese el número 1"/></td>
</tr>
<tr>
  <td>Número 2:</td>
  <td><input type="text" size="20" maxlength="30" placeholder="Ingrese el número 2"/></td>
</tr>
<tr>
  <td> </td>
  <td><button>Sumar</button> </td>
</tr>
<tr>
  <td>Resultado:</td>
  <td><input type="text" placeholder="Resultado" maxlength="9" disabled></td>
</tr>
</table>
</body>
```

El archivo Javascript sería:

Para el evento focus y blur (en el primer input solo)

```
//EVENTO FOCUS Y BLUR
let input1 = document.getElementById("numero1")
input1.addEventListener("focus", resaltar1)
input1.addEventListener("blur", noResaltar1)

function resaltar1(){
  input1.style.backgroundColor="pink"
}
function noResaltar1(){
  input1.style.backgroundColor=""
}
```

Para el evento de picar el botón y calcular la suma

```
function suma(){
  let num1 = parseFloat(document.getElementById("numero1").value)
  let num2 = parseFloat(document.getElementById("numero2").value)
  if (!isNaN(num1) && !isNaN(num2)) {
    // Realizar la suma y mostrar el resultado
    let lasuma = num1 + num2;
    document.getElementById("resultado").value = lasuma;
  } else {
    alert("Por favor, ingrese solo números.");
  }
}
```

Ejercicio7 (NO formulario): un Botón like

Tenemos un botón sobre un artículo que pone "Me gusta". Si picamos en él cambia a "No me gusta" y si viceversa.

Un botón ⇒ 

```
//Recogemos el botón
const boton = document.getElementById("miboton")
//al hacer clic en el botón cambiamos a no me gusta.
boton.addEventListener("click", cambio)
//Función que cambia el contenido del botón
function cambio() {
    if(boton.textContent == "Me gusta"){
        |   |   boton.textContent="No me gusta"
    }else{
        |   |   boton.textContent="Me gusta"
    }
}
```

Ejercicio7b (NO formulario): varios Botones like (NO HACER)

Podríamos tener varios botones en la página y ahora programarlo para todos los botones. En este caso hay que seleccionar todos los botones, y lo hacemos con el selector "class" por lo que se crea un array que habrá que recorrer para seleccionar cada botón.

Varios botones => 

```
//AHORA SI HAY VARIOS BOTONES
//Recogemos los botones
const arrayBotones = document.getElementsByClassName("miboton")

//al tener 4 botones o seleccionamos los 4 botones por separado con ids
// o hacemos un for para que se seleccionen los 4 a la vez con class.
for (let i = 0; i < arrayBotones.length; i++) {
    arrayBotones[i].addEventListener("click", cambio)
    //Función que cambia el contenido del botón
    function cambio() {
        if(arrayBotones[i].textContent == "Me gusta"){
            |   |   arrayBotones[i].textContent="No me gusta"
        }else{
            |   |   arrayBotones[i].textContent="Me gusta"
        }
    }
}
```

Ejercicio8: Ejercicio para convertir el valor de €uros en otra moneda. En un campo input nos pide los €uros a convertir, luego nos pide entre varias opciones, el tipo de moneda a convertir (dólares, Libras y Yenes) y si picamos en el botón "convertir" nos calcula la conversión en otro campo y lo queremos redondeado con dos decimales.

1 €uro = 1,09 dólares || 1 €uros = 0,86 Libras esterlinas || 1 €uro = 154,95 yenes

El código html:

```
<!DOCTYPE html PUBLIC>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Insert title here</title>
</head>
<body>
    <center><h1>CONVERSIÓN DE MONEDA</h1></center>
    <table width="40%" align="center">
        <tr>
            <td>€uros a convertir:</td>
            <td><input type="number" id="euros" size="20" maxlength="30" placeholder="Ingrese los €"/></td>
        </tr>
        <tr>
            <td>Elige la moneda:</td>
            <td><select name="select" id="opcion">
                <option value="dolar" selected>Dolares</option>
                <option value="libras">Libras</option>
                <option value="yen">Yenes</option>
            </select></td>
        </tr>
        <tr>
            <td></td>
            <td><button>Convertir</button></td>
        </tr>
        <tr>
            <td>Resultado:</td>
            <td><input id="resultado" type="text" placeholder="Resultado" maxlength="9"/></td>
        </tr>
    </table>
</body>
</html>
```

CONVERSIÓN DE MONEDA

€uros a convertir:	10
Elige la moneda:	<input type="button" value="Yenes ▾"/>
Resultado:	1549.50

window.location.href

La variable `window.location.href` es un método JavaScript utilizado para redirigir el navegador a una nueva página. Es tan sencillo como poner `window.location.href = "otrapagina.html"` cuando queramos acceder a otra página. Por ejemplo si pulsamos un botón y todo es correcto que nos mande a otra página. En el siguiente ejercicio lo practicamos.

`window.open("otrapagina.html", "_blank");` Si quisieramos abrirla en otra pestaña del navegador.

Ejercicio 9: Marcar checkbox para validar

En muchas ocasiones hemos tenido que marcar una casilla de verificación (checkbox) para poder seguir, por ejemplo hacia otra página, o para que se active un botón y poder seguir, etc. Pues vamos a hacer estos dos casos con un ejercicio con Javascript.

1. Tenemos una página (descargarla de Alexia) donde hay unos términos y condiciones a leer y abajo tenemos un botón para pasar a otra página. Si picamos en ese botón y no hemos marcado la casilla de "aceptar los términos" nos aparece una alerta indicándolo, en cambio si está marcada la casilla sí que podemos pasar a la segunda página.
2. En la segunda página en cambio el botón está desactivado y para que se active tenemos que marcar el checkbox. Una vez activado el botón, si picamos en él nos manda a la tercera y última página.

Página 1 (parte final de la página)

occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

He leído y acepto los términos y condiciones:

[Siguiente página](#)

Página 2 (parte final de la página)

occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

He leído y acepto los términos y condiciones:

[Finalizar](#)

Página 3

Página FINAL

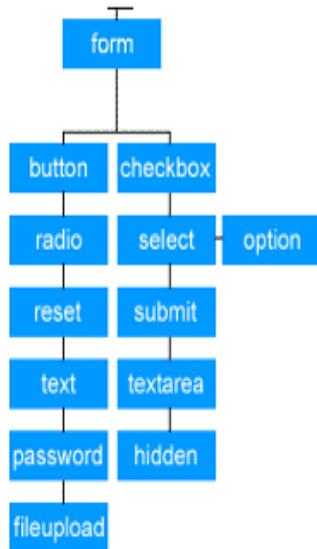
SI HAS LLEGADO HASTA AQUÍ ES QUE TODO ESTÁ BIEN.

6. FORMULARIOS- VALIDACIÓN

Casi todas las páginas tienen algún formulario ya sea para contacto, registro, blog, etc. Esos datos se suelen tratar con PHP para registrarse o consultarse en una Base de Datos en el lado del servidor, pero antes de enviarse al servidor hay que validar que esos datos sean correctos. Para validar los datos se suele usar Javascript por los siguientes motivos:

- **Respuesta inmediata:** El usuario recibe retroalimentación al instante si ha ingresado una edad no válida, evitando envíos innecesarios al servidor.
- **Mejor experiencia de usuario:** Se evita que el usuario envíe el formulario con errores, lo que puede ser frustrante.
- **Menor carga en el servidor:** Se reducen las solicitudes al servidor, optimizando el rendimiento.

En la imagen podemos ver los diferentes tipos de campos de un formulario.



Normalmente, la validación de un formulario consiste en llamar a una función de validación cuando el usuario pulsa sobre el botón de envío del formulario. En esta función, se hace lo siguiente:

SELECCIÓN DEL FORMULARIO Y DE SUS CAMPOS

Si queremos seleccionar el formulario entero podemos usar su "id" o por su name, pero se recomienda usar su id que si suponemos es "form1" lo seleccionamos así:

```
//UN FORMULARIO SE PUEDE SELECCIONAR POR SU "ID"
let form1 = document.getElementById("form1");
form1.addEventListener("submit", (event) =>{
})
```

Los campos de un formulario se pueden seleccionar también por "id" o por su "name", aunque se recomienda seleccionarlo por su id:

```
//SELECCIÓN DEL VALOR DE LA EDAD USANDO ID
let edad = document.getElementById("edad");
edad.value; //para obtener su valor
```

```
//SELECCIÓN DEL VALOR DE LA EDAD USANDO EN "NAME"
let edad = form1.edad.value //form1 es el name del <form>
```

NO USAR

VALIDACIÓN DEL FORMULARIO ANTES DE SU ENVÍO

Antes de enviar un formulario al servidor se debe validar para ver si es correcto y en caso afirmativo se envía a servidor PHP, pero si no es correcto no se debe de enviar. Hay dos formas de hacer esto:

- a. Forma más actual usando `event.preventDefault()`; en este caso seleccionamos el formulario con id, y le aplicamos la función "submit" para que si se envía el formulario se ejecute otra función que se la suele llamar `event` (pero se le puede poner cualquier nombre) donde controlaremos o

validaremos el formulario, para que no se envíe si no se cumple la validación, así no llegan los datos al servidor, y si no queremos que se envíen tendremos que poner la sentencia `event.preventDefault();` (de esta forma no hace falta poner submit en el botón de envío)

```
const formulario = document.getElementById("form1");
formulario.addEventListener("submit", function(event) {
    const nombre = formulario.nombre.value;
    if (nombre.trim() === "") {
        alert("Por favor, introduce tu nombre.");
        event.preventDefault();
    }
});
```

En vez de `function(event)` se podría haber puesto `(event) =>` (función flecha)

También se podría haber puesto la función de forma separada, pero es más lioso:

```
const formulario = document.getElementById("form1");
function validarFormulario(evento) {
    const nombre = formulario.nombre.value;
    if (nombre.trim() === "") {
        alert("Por favor, introduce tu nombre.");
        evento.preventDefault(); // evita el envío
    }
}

formulario.addEventListener("submit", validarFormulario);
```

Otra forma sería seria el siguiente:

```
form1.addEventListener("submit", (event) =>{
    if (!validarFormulario()) {
        event.preventDefault(); // Evita que se envíe el formulario
    }
})
```

En la cual dentro de la función event se pregunta si otra función (`validarFormulario`) ha devuelto `false` y en este caso no se envía el formulario al poner `event.preventDefault();`, y si envía `true` se envía. Tendremos que crear esta función `validarFormulario()` donde validamos los datos y si se cumplen las condiciones devolvemos "`return true`", pero si no se cumplen devolvemos "`return false`".

- b. Una forma no tan usada actualmente es poner dentro de la etiqueta `<form ... >` la función `onsubmit="return validar()"` y luego en el archivo javascript creamos la función validar para validar los datos que nos devolverá "`return true`" si todo es correcto o "`return false`" si algún dato de formulario no es correcto y así no enviaremos el formulario. En este caso no se necesita la función `event()`.

```
<form name="form1" action="tratarDatos.php" method="post" id="formulario" onsubmit="return validar()>
```

Ejercicio 10a: Validar formulario

Vamos a realizar la siguiente página con un formulario de dos campos y un botón de enviar.

```
<body>
    <center><h1>FORMULARIO</h1></center>
    <form name="form1" action="tratarDatos.php" method="post" id="formulario">
        <table width="40%" align="center">
            <tr>
                <td>Nombre: </td>
                <td><input type="text" name="nombre" size="20" maxlength="30" placeholder="Ingrese su nombre"/></td>
            </tr>
            <tr>
                <td>Edad: </td>
                <td><input type="text" name="edad" size="2" maxlength="2" placeholder="17"/></td>
            </tr>
            <tr>
                <td> </td>
                <td><button type="submit">Enviar datos</button> </td>
            </tr>
        </table>
    </form>
    <script src="xxxxxxxxx"></script>
</body>
```

Obtenemos esta página:

The screenshot shows a web page with a title 'FORMULARIO' at the top center. Below it is a table with two rows. The first row has two columns: 'Nombre:' followed by a text input field, and 'Edad:' followed by another text input field. The second row has two columns: an empty space followed by a submit button labeled 'Enviar datos'.

```
const formulario = document.getElementById("formulario"); // id del form

formulario.addEventListener("submit", function(event){
    // Validamos el nombre
    if (formulario.nombre.value=="") {
        alert("Tienes que introducir el nombre");
        event.preventDefault();
        return;
    }

    // Validamos la edad
    let edad = formulario.edad.value;
    let edadN = parseInt(edad);

    if (edadN=="") {
        alert("No has introducido tu edad");
        event.preventDefault();
        return;
    } else if (isNaN(edadN)) {
        alert("Tienes que introducir un número válido");
        event.preventDefault();
        return;
    } else if (edadN<18){
        alert("Se requiere ser mayor de edad");
        event.preventDefault();
        return;
    } else {
        alert("Todo validado y enviado");
    }
});
```

Ponemos `event.preventDefault()`; para que el formulario no se envíe ya que no se ha cumplido alguna condición, pero ponemos `return` para salir de la función ya que ya no se cumple una condición. Si no ponemos `return` se sigue ejecutando la función aunque el formulario no se envía.

Ejercicio1 Ob: Ampliación del formulario

Ahora vamos a ampliar el formulario con varios campos más como son la dirección de correo, DNI, etc y proceder a validarlos según un formato adecuado a cada campo. El campo DNI tiene que tener 8 dígitos y una letra y al campo de correo debe ser del formato xxxxxxxx@xxxxxx.xxx

- **required pattern="[0-9]{8}[A-Z]{1}"** (Es para requerir ese formato en el campo del formulario en html, es decir de se requieren 8 dígitos de 0 a 9, y 1 dígito de A a Z)
- En cambio para validar el correo debemos de usar Javascript como ahora veremos..

```
<tr><td>DNI:</td>
    <td><input name="dni" type="text" placeholder="Ingrese su DNI" required pattern="[0-9]{8}[A-Z]{1}" maxlenht="9"></td>
</tr>
<tr><td>Email:</td>
    <td><input name="email" type="text" placeholder="Ingrese su email" maxlenht="15"></td></tr>
<tr>
```

Nueva parte de código para validar mail.

```
//No hace falta validar el DNI ya que lo hemos hecho en el html
// Validados el mail con un formato válido
let mail = form1.mail.value
let mailValido = /^[a-z](\w\.)*[a-z](\w\.)*\.[a-z]{2,3}$/
if (mail == "") {
    alert("No has introducido un mail");
    return false
} else if (mailValido.test(mail) == false) {
    alert("La dirección de correo no tiene formato válido")
    return false
}
```

FORMULARIO

Nombre:	<input type="text" value="Ingrese su nombre"/>
Edad:	<input type="text" value=">17"/>
DNI:	<input type="text" value="Ingrese su DNI"/>
Email:	<input type="text" value="Ingrese su email"/>
<input type="button" value="Enviar datos"/>	

En Javascript validamos un email válido si tiene el siguiente formato:

```
let mailValido= /^[a-z](\w\.)*[a-z](\w\.)*\.[a-z]{2,3}$/
```

Ejercicio1a Usuario-Password sin ocultar

En una página se pide usuario y contraseña para poder acceder a otra página privada. Se pide lo siguiente y si no se cumple no se debe de enviar el formulario al servidor:

- Se quiere enfocar cualquiera de los dos campos cuando se sitúa el cursor y quitar el enfoque cuando quitamos el cursor.
- No se puede dejar ninguno de los dos campos en blanco, dándonos una alerta si lo hacemos y volvemos a introducirlos.
- En caso de no introducir bien el usuario y la password nos sale una alerta indicando que el usuario y/o la password no son correctos, volver a introducirlo.
- En caso de poner el usuario (alumno) y la password (sanluis) nos carga otra página "tratarDatos.php" que no hace falta hacer pero que sí se carga al picar en el botón.

INICIAR SESIÓN

Usuario:	<input type="text" value="Ingrese el usuario"/>
Contraseña:	<input type="password" value="Su contraseña"/>
<input type="button" value="Acceder"/>	

```
<!DOCTYPE html PUBLIC>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Insert title here</title>
</head>
<body>
<center><h1>INICIAR SESIÓN</h1></center>
<form action="procesarDatos.php" method="post">
<table width="40%" align="center">
<tr>
<td>Usuario: </td>
<td><input type="text" id="usuario" size="20" maxlength="30" placeholder="Ingrese el usuario"/></td>
</tr>
<tr>
<td>Contraseña: </td>
<td><input type="password" id="contrasena" size="20" maxlength="30" placeholder="Su contraseña"/></td>
</tr>
<tr>
<td> </td>
<td><button id="acceder">Acceder</button> </td>
</tr>
</table>
</form>
</body>
</html>
```

Ejercicio 11b: Usuario-Password con icono ojo (alexia)

Todos hemos introducido un usuario y una contraseña en alguna página web. En el momento de introducir una contraseña vemos que la contraseña no se puede ver, pero en ocasiones aparece un icono de un ojo donde si picamos en él podemos hacer legible la contraseña y si volvemos a picar vuelve a hacerse ilegible.

Vamos a realizar el ejercicio anterior con los siguientes cambios:

1. Si picamos en el input del "Usuario" se debe de remarcar con el fondo rosa.
2. Si picamos en el input de la "Contraseña" se debe de visualizar el ojo.
3. Si picamos en el ojo se debe de hacer legible la contraseña y el ojo cambia a un ojo tachado.
4. Si picamos en el ojo tachado se debe de volver a hacer ilegible la contraseña y el ojo vuelve a ponerse sin tachar.

5. Si introducimos bien el usuario (alumno) y la contraseña (sanluis) nos da una alerta indicándolo y accedemos a la página php, en caso contrario no se envia el formulario (preventdefault)

INICIAR SESION	
Usuario:	<input type="text" value="fruiz@centrosanluis.com"/>
Contraseña:	<input type="password" value="....."/> 
<input type="button" value="Iniciar sesión"/>	

INICIAR SESION	
Usuario:	<input type="text" value="fruiz@centrosanluis.com"/>
Contraseña:	<input type="password" value="sanluis"/> 
<input type="button" value="Iniciar sesión"/>	

Ejercicio12 (opcional): "aprendeEuskera"

Hacer una pequeña página para un programa infantil para que los niños/as aprendan los días de la semana en euskera. Se debe elegir entre un día de la semana en castellano y luego su traducción en euskera (los días en euskera están desordenados)

JUEGO PARA APRENDER EUSKERA EUSKARA IKASTEKO JOLASA

DIAS DE LA SEMANA ASTEEGUNAK

Si acertamos nos muestra la siguiente página

JUEGO PARA APRENDER EUSKERA EUSKARA IKASTEKO JOLASA

DIAS DE LA SEMANA	ASTEEGUNAK
<input type="button" value="Martes"/>	<input type="button" value="Astearteak"/>

GENIAL HAS ACERTADO



Si no acertamos nos muestra la siguiente

JUEGO PARA APRENDER EUSKERA EUSKARA IKASTEKO JOLASA

DIAS DE LA SEMANA	ASTEEGUNAK
<input type="button" value="Lunes"/>	<input type="button" value="Larunbata"/>

OHHH NO HAS ACERTADO!!



NOTA: si queremos que cada vez que se pique en el botón no se agregue una imagen nueva para tener siempre una imagen, ya sea de alegría o tristeza, **no debemos agregar un elemento a la página con createElement**, sino que debemos de cambiar el contenido del <div> con innerHTML poniendo el texto o la etiqueta de la imagen que corresponda.

`elemento.innerHTML = ""`

Ejercicio 13 (opcional): "juegoDados"

Se quiere hacer una página web de una apuesta del resultado de la tirada de un dado. Requisitos:

- El usuario debe apostar de forma obligatoria a un número de dado entre 1 y 6 y en caso contrario debe de salir una alerta que indique que el valor del nº no es correcto.
- Debe de poner el importe de la apuesta debiendo estar entre 5 y 10 € y sino sale una alerta indicando que la apuesta no está entre los valores requeridos.
- Ambas celdas deben remarcarse (color rosa) cuando ponemos el cursor dentro y desmarcarse cuando lo sacamos.
- Al picar en el botón "Apuesta" sale una imagen que es un gif de un dado girando durante 5 segundos. Al final de los 5 segundos sale el resultado(aleatorio) del dado.
- Si el usuario ha ganado la apuesta se duplica su importe y sino, se divide entre dos. En ambos casos sale un mensaje como el de la imagen inferior.

Se utiliza la función `setTimeout(resultado,5000);`



This screenshot shows the result of a failed bet. The user had bet on number 4 for 10 units. The message "OHH! NO HAS ACERTADO, ahora te queda 5€" (Oh! You didn't guess it, now you have 5€) is displayed at the bottom in red.

This screenshot shows the result of a successful bet. The user had bet on number 2 for 7 units. The message "HAS ACERTADO, AHORA TIENES 14€" (You guessed it, now you have 14€) is displayed at the bottom in green.

Ejercicio14 (opcional): "juegoPoker"

Tenemos una página de poker donde un usuario juega con la banca. Se pica en el botón "Repartir" y se sortean 4 cartas de poker para el usuario y otras cuatro para la banca y se colocan en cada baraja boca arriba. Se suman los puntos de cada uno y a ver quien gana!!!

Las imágenes de las cartas y los puntos de cada carta están establecidos en dos arrays en el archivo javascript. A JUGAR!!!!!

<https://drive.google.com/file/d/1R3OvTLocNv2CSAXhIT97AODxmeUtkvXs/view?usp=sharing>



Si alguien termina puede cambiar el mensaje diciendo si ha ganado o perdido o agregar un mensaje nuevo.

Extensiones recomendadas para Visual Studio Code

ESLint: para corregir la sintaxis en Javascript.

JQuery snippets: permite crear códigos JQuery de manera automática.

Un Framework o librería es una ayuda que los programadores pueden usar para ayudarles a programar ciertas operaciones que les simplifica la tarea. En nuestro caso JQuery es un **framework** para JavaScript de código abierto que agrega interactividad a un sitio web simplificando la tarea de programación con javascript. Cada plugin tiene un sitio web desde donde se pueden descargar sus archivos, con demos, instrucciones para su implementación, etc. En la web hay cientos de blogs que recopilan y analizan los **plugins** según sus funcionalidades. Cada plugin tiene sus instrucciones de instalación propias.

Descarga

Los **pluggins** de JQuery se pueden usar de dos maneras: (usaremos la opción 1)

- Descargar JQuery desde el sitio oficial <https://jquery.com/>, elegimos por ejemplo el archivo comprimido que es la primera opción(botón derecho, guardar como) y lo descargamos como archivo .js y luego lo vinculamos al final de la página html para dar tiempo a que se carguen todos los contenidos de la página. Lo vinculamos así:

```
<script src="js/code.jquery.com_jquery-3.7.0.min.js"></script>
```

 code.jquery.com_jquery-3.7.0.min.js
NOTA: hay que poner el enlace a jquery antes del enlace al js para que funcione!!!!!!

- Otra forma de vincular la librería es utilizando el servidor de Google. De esta manera no necesitamos descargarla ni subirla a nuestro servidor. En ese caso, el código es el mismo, pero con ruta absoluta:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
```

Sintaxis

Para insertar el código jquery usamos la siguiente sintaxis:

- Empezamos con el carácter \$ que significa jquery
- Entre paréntesis ponemos el elemento a seleccionar. Se **selecciona** según los identificadores usados en **CSS**, que son la **etiqueta html**, el **class** que se usa el punto . y el **id** se usa #
- Luego el evento a realizar sobre dicho elemento que definimos a través de una función

Ejemplo

```
$("elemento").evento(function(){
  //código de la función
});
```

Eventos sencillos (algunos ya vistos con Javascript)

click() hacer click sobre el elemento.

dblclick() hacer doble-click sobre un elemento

mouseenter() al pasar el ratón por encima del elemento

mouseleave() El ratón, que estaba situado encima de un elemento, sale de él

hide() ocultar un elemento.

show() Muestra un elemento

mousedown() cuando picamos con el botón izquierdo del ratón

mouseup() cuando soltamos el botón izquierdo del ratón

toggle() si el elemento está oculto lo muestra, y si está visible lo oculta

fadeIn() aparece un elemento poco a poco.

fadeOut() se desvanece un elemento poco a poco.

fadeToggle() si el elemento está desvanecido aparece y viceversa.

slideDown() desliza un elemento hacia abajo, es decir despliega un elemento tipo cortinilla.

slideUp() desliza un elemento hacia arriba, es decir pliega un elemento tipo cortinilla.

slideToggle() si el elemento está se pliega, si no está se despliega (cortinilla)

focus() cuando situamos el ratón en algún elemento

keydown() El evento se produce en el momento que se presiona una tecla

keyup() El evento se produce en el momento de dejar de presionar una tecla

append() agrega nuevos elementos

Ejemplo1

Tenemos el siguiente código con un título **h1** y sin **id** ni **class**, por lo que seleccionaremos el elemento h1 por el nombre de la etiqueta.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <h1>Título de la página</h1>
  <script src="../code.jquery.com_jquery-3.7.0.min.js"></script>
  <script src="ejemplo.js"></script>
</body>
</html>
```

Queremos que al hacer click en el título nos da una alerta y nos cambie el color a rojo

```
$( "h1" ).click( function() {
  alert( "HAS HECHO CLICK" )
  let titulo = document.getElementById("titulo")
  titulo.style.color="red"
})
```

Ahora al pasar el ratón por encima

```
1  $("h1").mouseenter(function(){
2    alert("HAS PASADO CON EL RATON")
3  })
```

Ejemplo2

Ahora en vez de una alerta queremos actuar sobre un elemento de tal forma que si picamos sobre él se oculte. Para ello usamos una caja con un div y lo identificamos en un identificador id.

```
<body>
  <h1>Título de la página</h1>
  <p>Vamos a realizar efectos con el siguiente cuadro</p>
<br>
<div id="caja" style="background:#98bf21;height:100px;width:100px;position:absolute">hola
</div>

<script type="text/javascript" src="../code.jquery.com_jquery-3.7.0.min.js"> </script>
<script type="text/javascript" src="ejemplo2.js"></script>
</body>
```

Queremos que cuando pulsemos con el ratón encima del div que desaparezca y cuando soltemos el ratón que aparezca de nuevo.

Para ello usamos los eventos mousedown y hide sobre el mismo elemento, pero luego no podemos usar el evento mouseup sobre el elemento al estar oculto, por lo que usamos el identificador universal que representa toda la página que es el asterisco *

```
$("#caja").mousedown(function(){
  $("#caja").hide()
})

$(".*").mouseup(function(){
  $("#caja").show()
})
```

Ejemplo3:

Las acciones a realizar cuando realizamos un evento también pueden ser modificar su estilo o realizar un movimiento, etc.

Por ejemplo si picamos en el div cambiamos el color y si soltamos vuelve a cambiar.

```
$("#caja").mousedown(function(){
  $(this).css("background-color", "gold");
})

$("#caja").mouseup(function(){
  $(this).css("background-color", "salmon");
})
```

En vez de seleccionar el elemento podemos usar el que ya tenemos seleccionado con **THIS** que actúa sobre el elemento seleccionado en ese momento, en el caso de que hubiese varios elementos que coincidan con la selección. En este caso se podría haber puesto también "#caja" en lugar de "this" ya que solo hay un elemento identificado con el id=caja.

Ejemplo4: eventos usando botones

Los eventos también se pueden hacer pulsando en un botón. Sobre el ejercicio anterior creamos tres botones, uno para ocultar el div usando **slideUp()**, otro para mostrarlo usando **slideDown()** y otro para cambiar el estilo.

El código html de los botones

Las tres funciones jquery son:

```
<button id="ocultar">Ocultar</button>
<button id="mostrar">Mostrar</button>
<button id="estilo">estilo</button>
```

Título de la página

Vamos a realizar efectos con el siguiente cuadro

Ocultar Mostrar estilo

Hola

```
$("#ocultar").click(function(){
    $("#caja").slideUp(2000)
})

$("#mostrar").click(function(){
    $("#caja").slideDown(4000)
})

$("#estilo").click(function(){
    $("#caja").css({'color':'purple','font-size':'50px','background':'pink'});
});
```

En los eventos `slideDown` y `slideUp` hemos puesto entre paréntesis el tiempo en que ocurre el evento en milisegundos.

- También se podía haber usado `$("#caja").fadeIn(4000)` `$("#caja").fadeOut(4000)` Probarlo
- También se podría haber mostrado u ocultado el elemento picando en un único botón usando el efecto `toggle`, `slideToggle` o `fadeToggle` dependiendo de si el elemento está presente o no está. Probarlo.

Efectos de Animaciones `animate()`

Ya hemos visto que podemos cambiar el estilo de un elemento, pero ahora vemos como también podemos animarlo cambiando las propiedades CSS, pero solo se pueden animar las propiedades css que tienen valores numéricos o con unidades. Para ello hay que indicar los parámetros de la animación. `animate()`

ANIMABLES	NO ANIMABLES	
width / height top / left / right / button opacity borderWidth / borderXWidth padding / paddingX margin / marginX fontSize lineHeight	color background-color display float	<pre>\$(“elemento”).animate({ prop1: valor1, prop2: valor2, ... propN: valorN }, duración_milisegundos); });</pre>
Ejemplo donde se aumenta el tamaño de una imagen		Se aumenta el tamaño de la letra de la celda de la tabla en 3.
<pre>\$(“img”).animate({ height: “+=50px”, //50 pixels más width: “+=50px” //50 pixels más },3000);</pre>		<pre>\$(“td”).animate({ fontSize : “3rem” },2000);</pre>

Ejemplo5a

Vamos a crear un nuevo botón que ponga mover y que si picamos nos deja un espacio a la izquierda de 500px

Si quisiéramos moverlo aumentando el espacio en la izquierda sería.

```
$("#mover").click(function(){
    $("#caja").animate({left: '500px'});
});
```

```
$("#caja").animate({left: "+=50px"});
```

Ejemplo5b

Igual que el anterior, pero si picamos en el botón, además de desplazarse, aumenta de tamaño

```
$("#mover").click(function(){
    $("#caja").animate({left: "500px", height: "200px", width: "200px"});
});
```

Ejemplo5c

Igual que el anterior, pero además le cambiamos el color del texto, de fondo, el tamaño del texto y cambiamos el texto a la caja div.

```
$("#mover").click(function(){
    $("#caja").animate({left: "500px", height: "200px", width: "200px"});
    $("#caja").css({'color': 'purple', 'font-size': '50px', 'background': 'yellow'});
    $("#caja").text('HOLA CLASE');
});
```

Ejemplo5d. Sin usar botón

También se podría haber hecho cualquiera de las animaciones al cargarse la página sin usar ningún botón, para ello usamos el siguiente código.

También se podría haber hecho el evento sin botón, al situar el cursor encima del elemento o picando en el elemento.

```
$document.ready(function(){
    $("#caja").animate({left: "500px"}, 5000);
})
```

```
$("#caja").mouseenter(function(){
    $("#caja").animate({left: "500px"}, 5000);
});
```

hover() y this()

hover() : Este evento muy útil tiene realmente dos eventos (funciones), uno es cuando el ratón se sitúa encima del elemento seleccionado y el otro cuando se separa. Por ello tiene dos funciones, la primera para cuando el ratón entra, y la segunda cuando sale. (parecido a **mouseenter()** y **mouseleave()** pero en un solo evento)

this() : esta función que ya hemos comentado anteriormente se usa mucho, indica el elemento seleccionado o activo en ese momento.

Vemos la estructura como hay dos funciones, la primera es cuando pasa el ratón y la segunda cuando se quita:

```
$( "elemento" ).hover(function(){
    $(this).evento(); //si pasa el ratón
}, function(){
    $(this).evento(); //si quita el ratón
})
```

Ejemplo 6 CV:

El profesor os entrega un archivo html y css correspondiente a una página de un currículum vitae y queremos que cada vez que el ratón pase por uno de los párrafos que se remarque solo ese párrafo y cuando se quite el ratón que vuelva a su estado original, usando para ello **hover()** y para que lo haga solo en el párrafo activo usamos **this()**

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Currículum Vitae (corto)</title>
    <charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="CV.css"/>
</head>
<body>
<h1>Currículum Vitae de Xabier Moreno López</h1>
<p>Este es un ejemplo de currículum vitae. Cualquier parecido con la realidad es pura coincidencia. </p>
<h2>Datos personales</h2>
<p>Nombre y Apellidos: Xabier Moreno López<br/>
    Dirección: C./ Licenciado Poza nº 31 - 48027 Bilbao<br />
    D.N.I.: 27.182.818S</p>
<h2>Educación y formación</h2>
<p>2021-2023: Centro San Luis(Bilbao). Ciclo Formativo de Grado Superior Administración de Sistemas Informáticos en Red (modalidad semipresencial)<br />
    2019-2021: Instituto de Formación Profesional San Luis (Bilbao). Ciclo Formativo de Grado Superior Desarrollo de Aplicaciones Multiplataforma.</p>
<h2>Capacidades y aptitudes personales</h2>
<h3>Idiomas</h3>
<p>Inglés: Nivel Alto. Título Superior de la Escuela Oficial de Idiomas.<br />
    Euskera:EGA</p>
```

```
</body>
```

```
</html>
```

El archivo CSS

```
*{margin:0px;}
```



```
body {  
    padding: 50px;  
    background-color: lightblue;  
    font-family: calibri;  
    text-align: justify;  
}  
  


```
h1 {
 background-color: brown;
 color: white;
 margin: 10px;
 border: 15px solid rgb(66, 62, 62);
 text-align: center;
 text-transform: uppercase;
 word-spacing: 15px;
}


```
h2 {  
    padding: 15px;  
    border: 10px solid #456547;  
    border-style: outset;  
    background-color: gray;  
    color: rgb(8, 8, 228);  
    font-style:italic;  
}  
  


```
h3 {
 margin-left: 20px;
 border: 1px solid rgb(136, 158, 37);
 background-color: #dddddd;
 color: rgb(87, 82, 82);
}


```
p {  
    margin: 30px;  
    padding: 5px;  
    border: 2px solid rgb(128, 127, 202);  
    border-radius: 20px;  
    background-color: #dddddd;}
```


```


```


```


```

```

span{
    color: #20B2AA;
    font-weight: bold;
}

```

Se quiere que cuando pasemos el ratón por encima de un párrafo cambie el estilo solo de ese párrafo, cuando quitemos el ratón que vuelva a su estado normal.

```

$("p").hover(function(){
    $(this).css({'color':'purple','font-size':'50px','background':'pink'});
}, function(){
    $(this).css({
        "margin": "30px",
        "padding": "5px",
        "border": "2px solid rgb(128, 127, 202)",
        "border-radius": "20px",
        "background-color": "#dddddd",
        'font-size': '20px'})
})

```

THIS

Continuando con This, también se usa mucho por ejemplo para desplegar sólo una opción de un menú que tiene varias opciones, en vez de desplegar todas las opciones de menú.

Ejemplo 7a

Tenemos el siguiente código html y queremos ocultar los hijos del elemento `<div>` que son los `` identificados con la clase "listado", pero cuando pasemos el ratón por un elemento solo queremos ocultar los hijos de ese elemento `` no de todos los ``

```

$(".asignaturas").hover(function() {
    $("ul", this).hide();
}, function(){
    $("ul", this).show();
});

```

Con `hover()` cuando el ratón se posiciona sobre algún elemento con `id="asignaturas"`, que son los dos `<div>` ejecutamos `$(“ul”, this)` que lo que hace es que al elemento hijo `` del `<div>` que tenemos seleccionado (`this`) lo ocultamos con `hide()`, y la segunda función es si quitamos el ratón se vuelve a ver con `show()`.

Gracias a `this()` solo interactuamos con el `<div>` dónde está el ratón, pero no sobre los dos `<div>`.

```


# Seguimos usando THIS y HOVER



## Asignaturas de ASIR



1ASIR (acerca y aleja el ratón)


- Redes
- Sistemas
- Marcas
- Hardware



2ASIR (acerca y aleja el ratón)


- SEGURIDAD
- SERVICIOS
- SISTEMAS
- IMPLEMENTACIÓN


```

Ejercicio 7b:

Ahora lo hacemos al revés, es decir ponemos **show** en la primera función y **hide** en la segunda para que se despliegue cuando situemos el ratón y se pliegue cuando lo quitemos.

```
$(".asignaturas").hover(function() {  
    $("ul", this).show();  
}, function(){  
    $("ul", this).hide();  
});
```

Vemos que funciona, lo que pasa es que al cargarse la página los **** están desplegados y nos gustaría que estuviesen plegados. Para ello en **css** pondremos "**display: none**" según la imagen de la derecha

```
<div class="asignaturas">1ASIR (acerca y aleja el ratón)<br>  
    <ul id="listado" style="display: none">  
        <li>Redes</li>  
        <li>Sistemas</li>  
        <li>Marcas</li>  
        <li>Hardware</li>  
    </ul>  
  
<div class="asignaturas">2ASIR (acerca y aleja el ratón)<br>  
    <ul id="listado" style="display: none">  
        <li>SEGURIDAD</li>  
        <li>SERVICIOS</li>  
        <li>SISTEMAS</li>  
        <li>IMPLANTACIÓN</li>  
    </ul>
```

Ejemplo8: Menú desplegable

En el ejemplo anterior hemos visto cómo ocultar un elemento y sus hijos, ahora podemos hacer lo contrario, es decir desplegar una opción entre varias obteniendo el efecto que hacemos sobre un menú en el cual solo se despliega la opción sobre la que hacemos click o pasa el ratón.

Nos descargamos de los siguientes links los dos archivos html y css sobre un menú para usar la función **hover** y **this** para desplegar solo la opción del menú seleccionado.

<https://drive.google.com/file/d/1xAcMjVHoluotu5VD09Q2bOCeHXh2vh0-/view?usp=sharing>
<https://drive.google.com/file/d/1M3uvkGM-rBsTlAdrVi818N-lmGFJsNP/view?usp=sharing>

El ejercicio debe quedar así:



EJERCICIOS DE REPASO

1. Con un array de los ingredientes para hacer una receta de pasta carbonara, si pulsamos en el botón nos muestra debajo con una lista todos los ingredientes necesarios. Podrías poner otro botón para que nos lo oculte o que nos remarque(estilo) los ingredientes imprescindibles, etc.

Ingredientes Pasta carbonara:

Si quieres hacer esta receta pica para ver los ingredientes

Los ingredientes necesarios son:

- Pasta
- Queso
- Bacon
- Pimienta
- Aceite

2. Tenemos una lista de muchos países, unos con costa marítima y otros no. Si picamos en el botón nos muestra con fondo azul y color rosa los que tienen costa.

Paises del mundo y si tienen costa: <p>Te mostramos varios paises, piensa cuales tienen costa marítima y luego pica para verlos, a ver si has acertado!!</p> <ul style="list-style-type: none">• Japon• Suiza• Bolivia• Zimbabue• Australia• Francia• Dinamarca• Hungria• Croacia• Espana• Liechtenstein• Serbia• Canada• Islandia• Austria• Nueva Zelanda• Afganistán• Luxemburgo• Lituania• Serbia• Nepal• Nigeria• Chile• Egipto <p><input type="button" value="Paises con costa"/></p>	Paises del mundo y si tienen costa: <p>Te mostramos varios paises, piensa cuales tienen costa marítima y luego pica para verlos, a ver si has acertado!!</p> <ul style="list-style-type: none">• Japon• Suiza• Bolivia• Zimbabue• Australia• Francia• Dinamarca• Hungria• Croacia• Espana• Liechtenstein• Serbia• Canada• Islandia• Austria• Nueva Zelanda• Afganistán• Luxemburgo• Lituania• Serbia• Nepal• Nigeria• Chile• Egipto <p><input type="button" value="Paises con costa"/></p>
--	--

3. Cálculo de la suma de dos números: se introducen dos números en dos inputs y al pulsar en un botón, el resultado de la suma nos lo muestra en otro input.

Introduzca dos números:

Número 1:

Número 2:

La suma de estos números es:

4. Pasar la temperatura de grados celsius a fahrenheit. El usuario introduce unos grados celsius en un input y la página se lo devuelve en grados farenheit en otro input.

Conversión de grados centígrados a grados Fahrenheit:

Introduce grados centígrados:

Conversión a grados Fahrenheit:

5. Tenemos un menú superior que si ponemos el cursor encima de cada opción se despliega su correspondiente submenú y si quitamos el botón se recoge el menú.

6. Hacer un carrusel de imágenes con los nombres de los equipos de la liga y sus escudos para que se vayan cambiando cada 3 segundos.

7. En un formulario se nos pide usuario(tunombre) y contraseña (abreteSesamo) para acceder. Cuando el usuario pulse en el botón de "validar" se comprueba la contraseña de tal forma que si se introduce de forma correcta aparece una alerta en pantalla diciendo que NO es correcta, y sino se accede a otra página.

Introduce el usuario y contraseña para entrar:

Usuario:
Contraseña:

Esta página dice

La contraseña SI es correcta, SI puedes entrar

8. Tabla que si pasamos el ratón por encima de una celda que cambie el color de fondo de la celda, el color del texto y el tamaño del texto. Y cuando quitemos el ratón que vuelva a sus valores.

Pasa el ratón por las celdas a ver que pasa!!!!!!

Pasa el ratón por aquí!!!!	
	Mejor pasa el ratón por aquí
Acercate con el ratón, te va a gustar	No pases por aquí por favor

9. Al pasar el ratón por encima de una imagen que se agrande y se mueva al centro de la pantalla. También se podría hacer picando un botón.

Pica con el ratón en la imagen a ver que pasa



10. Al pasar el ratón por encima: de una imagen cambia por otra y si hacemos doble click cambia por otra y si quitamos el ratón vuelve a la imagen inicial. (el del pato donald)

Prueba de cambio de imagen

La imagen cambia cuando el puntero del ratón está encima, se hace click o se hace doble click. Y si se aparta el ratón vuelve a la imagen original



11. Explica que el siguiente código Javascript en la página web

```

<body>
  <h1 id="titulo1">Tituto de la página</h1>

  <button id="botonCambiar">Cambia color</button>

  <p class="parrafos">Esto en el primer párrafo</p>
  <p class="parrafos">Esto en el segundo párrafo</p>
  <p class="parrafos">Esto en el trecer párrafo</p>

  <script src="app.js"></script>
</body>

```

La página queda:

Tituto de la página

[Cambia color](#)

- Inicio
- Nosotros
- Servicios
- Contacto

Esto en el primer párrafo

Esto en el segundo párrafo

Esto en el trecer párrafo

```

1 //creamos la variable con el elemento html
2 var boton = document.getElementById("botonCambiar")
3 // agregamos el evento click al boton
4 boton.addEventListener("click", evento)
5
6 // creamos la función del evento con un array de los numeros rgb y lo metemos en el body
7 function evento (){
8   let valoresRgb = [aleatorio(255), aleatorio(255), aleatorio(255)]
9   document.body.style.backgroundColor = "rgb(" + valoresRgb[0] + "," + valoresRgb[1] + "," + valoresRgb[2] + ")"
10 }
11
12 //creamos la función de generar un numero de 0 a maximo para el rgb
13 function aleatorio(maximo){
14   return Math.round(Math.random() * maximo)

```

ESTOS NO HACE FALTA HACER, ALGUNOS SON REPETIDOS Y OTROS NO SON IMPORTANTES.

12. Cálculo del mayor: igual que el anterior, pero nos calcula el mayor de los dos números y que nos dé una alerta por pantalla y el resultado lo guarde en otro input

Introduzca dos números: Número 1: <input type="text" value="8"/> Número 2: <input type="text" value="5"/> El máximo de estos números es: <input type="text" value="8"/> <input type="button" value="Compara números"/>	Esta página dice el numero uno es mayor <input type="button" value="Aceptar"/>
---	--

13. Nos piden introducir el número del DNI y el programa nos tiene que calcular la letra y mostrar un mensaje en pantalla con el DNI con la letra

Cálculo de la letra del DNI DNI: <input type="text"/> <input type="button" value="Calcula la letra"/>	Cálculo de la letra del DNI DNI: <input type="text" value="25654789"/> <input type="button" value="Calcula la letra"/> El DNI completo es: 25654789-Z
--	---

14. Programa que nos muestre la fecha actual por pantalla.

Mostrar fecha/hora En este ejercicio vamos a generar código JavaScript que escriba la fecha actual: Hoy es Tue Jun 21 2022 23:39:42 GMT+0200 (hora de verano de Europa central)
--

15. Explica que hace los siguientes códigos HTML y Javascript.

<pre><body> <h1>Calculadora</h1> <input type="text" id="num1" size="1"> X <input type="text" id="num2" size="1"> = <button id="result">??????</button> <input type="button" value="Calcular!" class="calcular"></pre>	Calculadora <input type="text"/> X <input type="text"/> = <input type="text"/> <input type="button" value="Calcular!"/>
---	---

```
let calcular = document.querySelector(".calcular")
calcular.addEventListener("click", evento)
function evento(){
    var num1=document.getElementById('num1').value;
    var num2=document.getElementById('num2').value;
    var resultado=(num1*num2);
    document.getElementById('result').innerHTML= resultado;
}
```

Calculadora

8 X = 56

DATATABLES/Bootstrap

Datatables

Es un plugin de jQuery para el desarrollo de tablas html, con el cual podemos mejorar la presentación de las tablas, realizar búsquedas, ordenaciones, filtros y todo lo que se nos ocurra. Vamos a la página de [datatables](#) donde nos explica como hacerlo y nos muestra ejemplos de tablas para hacer.

Bootstrap

Es un framework front-end utilizado para desarrollar aplicaciones web usando CSS y JS. Está constituido por una serie de archivos CSS y JavaScript responsables de asignar características específicas a los elementos de la página.

CDN

CDN es una red de distribución de contenido que consiste en un grupo de servidores repartidos en distintas zonas geográficas que aceleran la entrega del contenido web al acercarlo a los usuarios. Por ejemplo si está en Nueva York y desea consultar el sitio web de su tienda favorita en Londres, que está hospedado en un servidor en el Reino Unido, experimentará tiempos de carga de contenido lentos si la solicitud debe cruzar todo el océano Atlántico. Para remediar este problema, las CDN almacenan una versión en caché del contenido del sitio web de Londres en varias ubicaciones geográficas alrededor del mundo, que se conocen también como "puntos de presencia" (POP). Estos POP contienen sus propios servidores de almacenamiento en caché y son responsables de distribuir ese contenido cerca de su ubicación en Nueva York. El contenido que se distribuye desde un servidor más cercano a su ubicación física le ofrece una experiencia web más rápida y de alto rendimiento.

Ejercicio de DataTables: Vamos a construir esta Datatables.

DATATABLES

Ejemplo de DATATABLES

Buscar:

Name	Position	Office	Age	Start date	Salary
Airi Satou	Accountant	Tokyo	33	2008-11-28	\$162,700
Angelica Ramos	Chief Executive Officer (CEO)	London	47	2009-10-09	\$1,200,000
Ashton Cox	Junior Technical Author	San Francisco	66	2009-01-12	\$86,000
Bradley Greer	Software Engineer	London	41	2012-10-13	\$132,000
Brenden Wagner	Software Engineer	San Francisco	28	2011-06-07	\$206,850

Mostrando 1 a 5 de 57 registros
Mostrar 5 registros

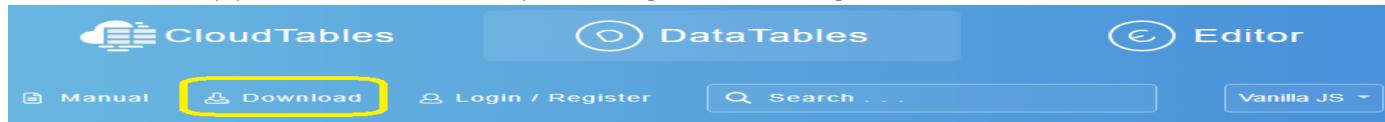
Anterior 1 2 3 4 5 ... 12 Siguiente

CONFIGURACIÓN DE DATATABLES

Vamos a la página de [datatables](#) en primer lugar a "Ejemplos" y nos muestra muchos ejemplos de tablas, podemos elegir el que más nos guste o uno parecido y luego lo modificamos. Por cada ejemplo que seleccionemos nos aparece abajo el código html, css, javascript y ajax si lo requiere.

Los pasos son los siguientes.

1. Elegimos la tabla que nos guste y vemos que abajo nos muestra el código html, javascript y css que vamos a necesitar, aunque luego podemos cambiar los datos e incluso añadir o quitar alguna columna. Elegimos la tabla "Zero configuration", [descargamos el código html de la tabla y la pegamos en nuestra página html](#). Si hubiera código css también lo tendríamos que crear. El código JS luego al final lo hacemos.
2. Vamos arriba y picamos en "Download" para configurar la descarga de datatables que queremos.



CloudTables

DataTables

Editor

Manual

Download

Login / Register

Search ...

Vanilla JS

- En el paso1 elegimos Bootstrap5
- En el paso2 elegimos DataTases

En el paso2 de extensiones seleccionamos "buttons", "Column visibility", "HTML5 export", "JSZip", "pdfmake" y "Print view". Para poder exportar la tabla a excel, zip, pdf e imprimir.

- En el paso 3 tenemos que elegir el método para usar esta configuración. Elegimos Minify para que la descarga y uso sea más rápido.

Podemos copiar los link CDN o descargarnos los archivos a usar en local. Mejor usamos los link CDN para no descargarnos los archivos que pesa más y hace la página más lenta. CDN ya hemos visto que es y da rapidez a la carga de la página.

Minify Use minified files for smaller file sizes and faster downloads.

Concatenate Combine files together for a, typically, faster download.

```
1 <link href="https://cdn.datatables.net/1.13.5/css/dataTables.bootstrap5.min.css" rel="stylesheet"/>
2 <link href="https://cdn.datatables.net/buttons/2.4.1/css/buttons.bootstrap5.min.css" rel="stylesheet"/>
3
4 <script src="https://cdnjs.cloudflare.com/ajax/libs/jszip/3.10.1/jszip.min.js"></script>
5 <script src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.2.7/pdfmake.min.js"></script>
6 <script src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.2.7/vfs_fonts.js"></script>
7 <script src="https://cdn.datatables.net/1.13.5/js/jquery.dataTables.min.js"></script>
8 <script src="https://cdn.datatables.net/1.13.5/js/dataTables.bootstrap5.min.js"></script>
9 <script src="https://cdn.datatables.net/buttons/2.4.1/js/dataTables.buttons.min.js"></script>
10 <script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.bootstrap5.min.js"></script>
11 <script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.colVis.min.js"></script>
12 <script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.html5.min.js"></script>
13 <script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.print.min.js"></script>
```

Bootstrap 5

DataTables

Extensions

AutoFill

Buttons

Column visibility

HTML5 export

JSZip

pdfmake

Print view

3. Copiamos los dos primeros links y los pegamos en el head de la página html

```
<title>DataTables</title>
<!-DataTables con Bootstrap-->
<link href="https://cdn.datatables.net/1.13.5/css/dataTables.bootstrap5.min.css" rel="stylesheet"/>
<link href="https://cdn.datatables.net/buttons/2.4.1/css/buttons.bootstrap5.min.css" rel="stylesheet"/>
```

4. Copiamos el resto de links y los pegamos al final de la página antes del cierre del body de la página html.

```
<!-DataTables-->
<script src="https://cdnjs.cloudflare.com/ajax/libs/jszip/3.10.1/jszip.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.2.7/pdfmake.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.2.7/vfs_fonts.js"></script>
<script src="https://cdn.datatables.net/1.13.5/js/jquery.dataTables.min.js"></script>
<script src="https://cdn.datatables.net/1.13.5/js/dataTables.bootstrap5.min.js"></script>
<script src="https://cdn.datatables.net/buttons/2.4.1/js/dataTables.buttons.min.js"></script>
<script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.bootstrap5.min.js"></script>
<script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.colVis.min.js"></script>
<script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.html5.min.js"></script>
<script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.print.min.js"></script>
```

5. Necesitamos el **CDN de Bootstrap**. Vamos a la página de [Bootstrap](#), picamos en download, bajamos y copiamos los dos link según la imagen, uno es de CSS y otro de JS.

CDN a través de jsDelivr

Omita la descarga con [jsDelivr](#) para entregar la versión en caché de CSS y JS compilados de Bootstrap a su proyecto.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="...<br/><script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js" integrity="...">
```

Copiamos el primer link (css) y lo pegamos en el head de la página.

```
<!--DataTables con Bootstrap--><link href="https://cdn.datatables.net/1.13.5/css/dataTables.bootstrap5.min.css" rel="stylesheet"/><link href="https://cdn.datatables.net/buttons/2.4.1/css/buttons.bootstrap5.min.css" rel="stylesheet"/><!--Bootstrap--><link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="...">
```

Copiamos el segundo link (js) y lo pegamos en el body (después del resto de links)

```
<script src="https://cdn.datatables.net/buttons/2.4.1/js/buttons.print.min.js"></script><!--Bootstrap--><script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js" integrity="...">
```

6. Debemos de **usar JQuery**. Podemos tenerlo descargado o usar un **CDN**. Pero como ya lo tenemos descargado de otros ejercicios vamos a usarlo, en vez de descargar un **CDN**. (Si quisiéramos usar un **CDN** lo podríamos buscar por Internet o podemos ir a la página de [JQuery](#), downloads y luego buscamos **CDN** y podemos descargar un **CDN** de google, de jsdelivr, etc.)

Debemos de pegar el link antes del resto de link para que el resto de script puedan usarlo

```
</table><!--JQuery--><script src=".../JQuery/code.jquery.com_jquery-3.7.0.min.js"></script><!--DataTables--><script src="https://cdnjs.cloudflare.com/ajax/libs/jszip/3.10.1/jszip"></script><script src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.2.7/pdfmake"></script><script src="https://cdnjs.cloudflare.com/ajax/libs/pdfmake/0.2.7/vfs_fonts.js"></script><script src="https://cdn.datatables.net/1.13.5/js/dataTables.bootstrap5.min.js"></script>
```

7. Por último nos falta el **Javascript**. Podemos ponerlo en la propia página html o en un archivo independiente.

En la propia página html con el siguiente código que es el que nos viene en la página de DataTables. Se pone al final de la página, antes del body

```
<script>new DataTable('#example');</script></body></html>
```

En un archivo independiente. En este caso hay que enlazarlo en la página html al final del body. USAMOS MEJOR ESTE.

```
JS main.js > ready() callback
1   $(document).ready(function(){
2     $('#example').DataTable();
3   });

```

Otras configuraciones del Datatables

Por defecto nos viene ciertos parámetros para ver la tabla, como el idioma, el número de registros por página, etc. Para poder agregar o cambiar una configuración que trae por defecto tenemos que agregar un objeto al código de la DataTable del archivo Javascript.

Registros por página

Por defecto nos pone 10,20.... Lo podemos cambiar poniendo el objeto `LengthMenu: []`

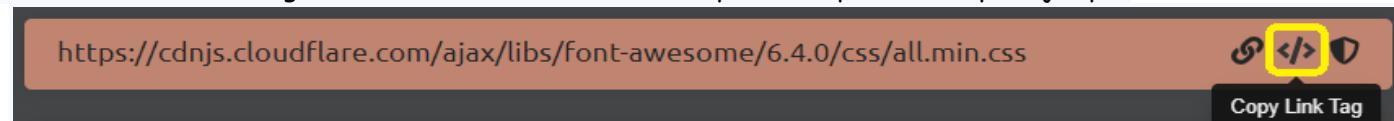
```
$document).ready(function(){
    $("#example").DataTable([
        // Para inicializar datatables
        lengthMenu: [5,10,15,20], //Las filas que vemos de la tabla
    ]);
});
```

Objeto nuevo

Iconos

Cuando nos hemos descargado la configuración de DataTables, hemos marcado la opción de descargarnos también la posibilidad de poner iconos para exportar la tabla a excel, pdf o imprimirla. Para poder conseguir estos iconos hacemos lo siguiente:

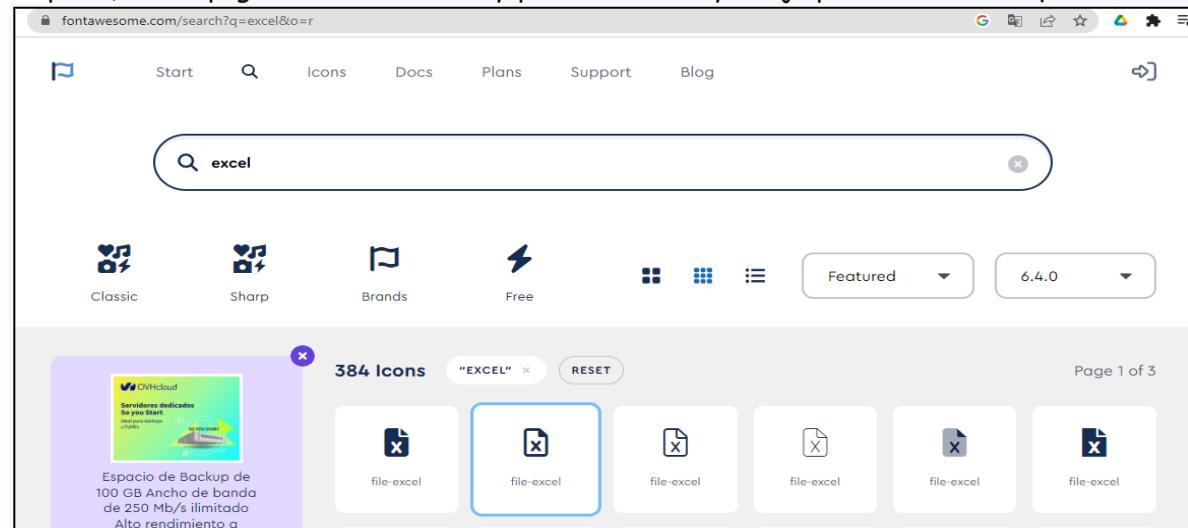
- Primero nos descargamos el [CDN de Font Awesome](#). Copiamos el primer link por ejemplo.



y lo pegamos en nuestra página html en el body.

```
<!--Font awesome-->
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css"
</head>
```

- Vamos a la página de [Font Awesome](#) para buscar los links de los iconos que queremos usar (excel, pdf, print). En la página vamos a iconos y ponemos excel y abajo picamos el icono que más nos guste.



- Luego nos muestra el ícono y su link (Imagen1) y lo copiamos y en el archivo de javascript agregamos el objeto `dom:"Bfrtilp"` y el objeto `buttons:[]` y dentro de los corchetes entre llaves la configuración de cada botón que pongamos, de la siguiente manera (Imagen2):

The screenshot shows the DataTables website's configuration interface. On the left, there's a large 'CSV' logo. The main area has tabs for 'HTML', 'React', 'Vue.js', and 'SVG'. The 'HTML' tab is selected, displaying a snippet of code: `<i class="fa-solid fa-file-csv"></i>`. To the right, a block of JavaScript code is shown:

```

$(document).ready(function() {
    $("#example").DataTable({
        // Para inicializar datatables
        lengthMenu: [5,10,15,20], //Las filas que vemos de la tabla
        dom: "Bfrtilp", //B es button, etc...
        buttons: [
            {
                extend: "excelHtml5",
                text: "<i class='fa-solid fa-file-csv'></i>",
                titleAttr: "Exportar a Excel",
                className: "btn btn-success",
            },
            {
                extend: "pdfHtml5",
                text: "<i class='fa-solid fa-file-pdf'></i>",
                titleAttr: "Exportar a PDF",
                className: "btn btn-danger",
            },
            {
                extend: "print",
                text: "<i class='fa-solid fa-print'></i>",
                titleAttr: "Imprimir",
                className: "btn btn-info",
            },
        ],
    });
});

```

Idioma

Podemos cambiar el idioma a castellano de la siguiente forma:

- Vamos a la página de datables, download, Plug-in, Internationalisation, y abajo buscamos "spanish" y picamos en el primero que es el genérico.

Language	Code	Completion	
Spanish (Changes Pending Moderation)	es-ES	99%	Log in to contribute
Spanish - Argentina	es-AR	67%	Log in to contribute
Spanish - Chile (Changes Pending Moderation)	es-CL	79%	Log in to contribute

- Bajamos a **Plug-in Code** y copiamos todo el código y lo tenemos que pegar como otro objeto en el archivo javascript.

Plug-in code

```

1  {
2      "processing": "Procesando...",
3      "lengthMenu": "Mostrar _MENU_ registros",
4      "zeroRecords": "No se encontraron resultados",
5      "emptyTable": "Ningún dato disponible en esta tabla",
6      "infoEmpty": "Mostrando registros del 0 al 0 de un total de 0 registros",

```

Javascript

Seguimos en el archivo javascript

Al final del archivo Javascript aparece así y tenemos que poner una coma.

```
language: {
    "processing": "Procesando...",  
    "lengthMenu": "Mostrar _MENU_ re
```

```
        "removeTitle": "Remover Estado",  
        "renameTitle": "Cambiar Nombre Estado"
    }
}
```

Fuente:

<https://www.youtube.com/watch?v=LlanZ4RdXBY>