RECORDATORIOS DE Formularios en HTML

Un **formulario** web sirve para enviar, tratar y recuperar datos que son enviados y recibidos entre un cliente y un servidor web.

Los formularios se definen con **etiquetas**. La etiqueta principal es **<form></form>.** Para que sea funcional, la etiqueta <form> necesita inicializar dos atributos:

- **action** – Contiene la URL donde se redirigen los datos del formulario.
- **method** – Indica el método por el cual el formulario envía los datos. Puede ser POST o GET.

Ejemplo:

```
<html>
	<head><title>Ejemplo de formulario</title></head>
	<body>
		<h3>Formulario</h3>
		<form action="http://www.ceu.es" method="post">
		</form>
	</body>
</html>
```

Elementos de un formulario

El elemento principal del formulario se denomina con la etiqueta <**input**>

Atributos de la etiqueta input:

- **Type:** Indica el tipo de elemento que vamos a definir. De él dependen el resto de parámetros. Los valores posibles que acepta el atributo type son:
  - **text** (cuadro de texto).
  - **password** (cuadro de contraseña, los caracteres aparece ocultos tras asteriscos).
  - **checkbox** (casilla de verificación).
  - **radio** (opción de entre dos o más).
  - **submit** (botón de envío del formulario).
  - **reset** (botón de vaciado de campos).
  - **file** (botón para buscar ficheros).
  - **hidden** (campo oculto para, el usuario no lo visualiza en el formulario),
  - **image** (botón de imagen en el formulario).
  - **button** (botón del formulario).

- **Name**. El atributo name asigna un nombre al elemento. Si no le asignamos nombre a un elemento, el servidor no podrá identificarlo y por tanto no podrá tener acceso al elemento.
- **Value**. El atributo value inicializa el valor del elemento. Los valores dependerán del tipo de dato, en ocasiones los posibles valores a tomar serán verdadero o falso.
- **Size**. Este atributo asigna el tamaño inicial del elemento. El tamaño se indica en pixeles. En los campos text y password hace referencia al número de caracteres.
- **Maxlength**. Este atributo indica el número máximo de caracteres que pueden contener los elementos text y password. Es conveniente saber que el tamaño de los campos text y password es independiente del número de caracteres que acepta el campo.
- **Checked**. Este atributo es exclusivo de los elementos checkbox y radio. En el definimos que opción por defecto queremos seleccionar.
- **Disabled**. Este atributo hace que el elemento aparezca deshabilitado. En este caso el dato no se envía al servidor.
- **Readonly**. Este atributo sirve para bloquear el contenido del control, por tanto el valor del elemento no se podrá modificar.
- **Src**. Este atributo es exclusivo para asignar una URL a una imagen que ha sido establecida como botón del formulario.
- **Alt**. El atributo alt, incluye una pequeña descripción del elemento. Habitualmente y si no lo hemos desactivado cuando posicionamos el ratón (sin pulsar ningún botón) encima del elemento, podemos visualizar la descripción del mismo.

Ejemplos de Tipos de input

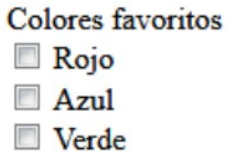
- **Cuadro de texto**: Este input muestra un cuadro de texto vacío en el que el usuario puede introducir un texto. Este es uno de los elementos más usados. La forma de indicar que es un campo de texto es: **type=”text”**

- **Cuadro de contraseña**: es como el cuadro de texto, con la diferencia que los caracteres que escribe el usuario no se ven en pantalla. En su lugar los navegadores muestran asteriscos o puntos.

```
<input type="password" name="contrasenia" />
```

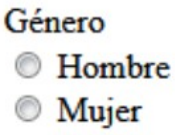
- **Casilla de verificación:** Estos elementos permiten al usuario activar o desactivar la selección de cada una de las casillas de forma individual.

  Colores favoritos
  </br><input name="rojo**" type="checkbox"** value="ro"/> Rojo
  </br><input name="azul" **type="checkbox"** value="az"/> Azul
  </br><input name="verde" **type="checkbox"** value="ve"/> Verde

- **Opción de radio**: Este tipo de elemento agrupa una serie de opciones excluyentes entre sí. De esta forma el usuario sólo puede coger una opción de entre todas las que tiene establecidas un grupo de botones radio.

  Género
  </br><input **type="radio" name="genero"** value="M"> Hombre
  </br><input **type="radio" name="genero"** value="F"> Mujer

- **Botón de envío**: Este elemento es el encargado de enviar los datos del formulario al servidor. En este caso el type toma el valor **submit**. El valor del atributo value se mostrará en este caso en el botón generado.

  <input **type="submit"** name="enviar" value="Enviar">

- **Botón de reset**: Este elemento es un botón que establece el formulario a su estado original.
  <input type="reset" >

- **Ficheros adjuntos**: Este tipo de input permite adjuntar ficheros adjuntos. El elemento añade de forma automática un cuadro de texto que se dispondrá para almacenar la dirección del fichero adjunto seleccionado.

  Fichero adjunto
  <input **type="file"** name="fichero"/>

- **Campos ocultos**: Los campos ocultos no son visibles en el formulario por el usuario. Estos elementos son útiles para enviar información de forma oculta que no tenga que ser tratada por el usuario.

  <input **type="hidden"** name="campoOculto" value="cambiar"/>

- **Botón de imagen**: Este elemento es una personalización de un botón, cambiando el aspecto por defecto que tienen los botones de un formulario por una imagen.

  <input **type="image"** name="enviar" src="imagen_mundo.jpg"/>

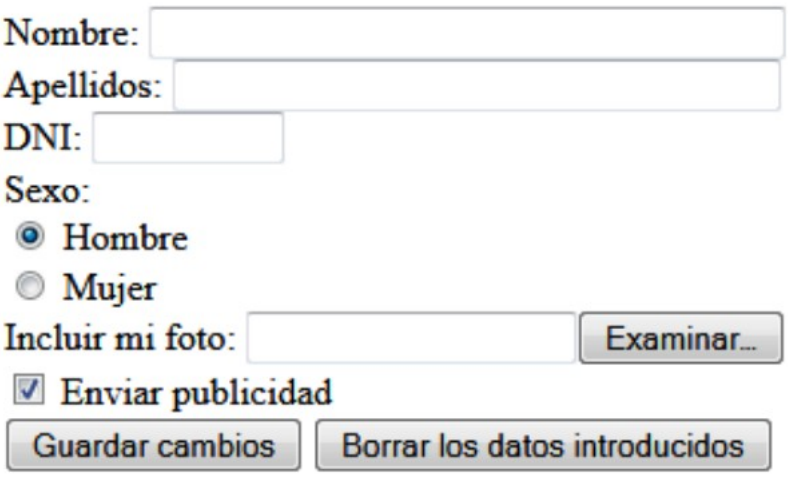
- **Botón**: Existe un elemento botón, al que podemos asociar diferentes funcionalidades. De esta forma no nos tenemos que ceñir los botones de submit o reset que nos ofrecen los formularios.

  <input **type="button"** name="opcion" value="Opcion validar"/>

Ejemplo completo:

```
<form action="pagina.php" method="post" enctype="multipart/form-data" /> <br/>
    Nombre:<input type="text" name= "nombre" value="" size="42 maxlength="30”/>
    Apellidos:<input type="text" name="ape" value="” size="40” maxlength="80”/>
    DNI:<input type="text" name="dni" value="" size="10” maxlength="9" />
    Sexo:
    <input type="radio" name="sexo" value="hombre” checked="checked”/>Hombre
    <input type="radio" name="sexo" value="mujer" />Mujer

    Incluir mi foto: <input type="file" name="foto" /> <br/>
    <input name="publ" type="checkbox" value="publicidad” checked="checked"/>
    Enviar publicidad
    <input type="submit" name="enviar" value="Guardar cambios” />
    <input type="reset" name="limpiar" value="Borrar los datos introducidos”/>
</form>
```



Formas de selección del objeto Form desde JavaScript

En JavaScript, puedes seleccionar un elemento <form> de varias maneras utilizando el DOM. A continuación te muestro algunas formas comunes de hacerlo:

Si partimos del siguiente ejemplo:

```
<div id="menulateral">
    <form id="contactar" name="contactar" action="...">...</form>
</div>
```

1.- Seleccionar el formulario por 'id': getElementById('id')

Si tu formulario tiene el atributo 'id', puedes seleccionarlo con getElementById. tendremos que tener la precaución de asignar id únicos a nuestros objetos, para evitar que tengamos objetos con id repetidos.

Ejemplo:

```
const formulario=document.getElementById("contactar");
```

2.- Seleccionar el formulario por 'name': document.forms

Si tu formulario tiene un atributo name, puedes seleccionarlo con **document.forms**.

Ejemplo:

```
const formulario = document.forms['miFormulario'];
```

También podrías acceder al formulario directamente por su índice en document.forms si tienes más de uno en la página:

```
const formulario = document.forms[0];
```

IMPORTANTE: document.forms devuelve un array siempre

3.- Seleccionar el formulario por su tag 'form': getElementsByTagName(tag)

A través del método getElementsByTagName() del DOM, el cual nos permite acceder a un objeto a través de la **etiqueta** HTML que queramos. **Devuelve un array de elementos**. Por ejemplo para acceder a los objetos con etiqueta form haremos:

```
let formularios = document.getElementsByTagName("form");
let primerFormulario = formularios[0]; // primer formulario del documento
```

o también todo en una única línea:

```
let primerFormulario = document.getElementsByTagName("form")[0];
```

Otra posibilidad interesante que te permite el método anterior, es la de buscar objetos con un padre determinado, por ejemplo:

```
let menu=document.getElementById("menulateral");
let formularios=menu.getElementsByTagName("form"); // formularios contenidos en el menú lateral
let primerFormulario= formularios[0]; // primer formulario en el menú lateral
```

IMPORTANTE: getElementsByTagName() devuelve un array siempre

Acceso a propiedades y métodos del formulario desde JS

Un formulario en HTML tiene varias propiedades a las que puedes acceder desde JavaScript, como por ejemplo:

- **elements**: Colección de todos los elementos del formulario (inputs, selects, textareas, etc.).
- **length**: Cantidad de controles (inputs, selects, etc.) en el formulario.
- **action**: URL a la que se enviarán los datos cuando se envíe el formulario.
- **method**: Método HTTP usado para enviar los datos (GET o POST).
- **name**: Nombre del formulario.
- **id**: ID del formulario.

```
const formulario = document.getElementeById('miFormulario ');
```

Accediendo a propiedades:

```
console.log(formulario.action); // URL de envío
console.log(formulario.method); // Método HTTP
console.log(formulario.length); // Número de elementos dentro del formulario
console.log(formulario.elements); // Colección de todos los elementos
```

Desde código JavaScript, podemos modificar estas propiedades mediante asignaciones, por ejemplo:

```
formulario.action = "http://www.educacion.gob.es/recepcion.php";
document.forms[0].action = "http://www.educacion.gob.es/recepcion.php";
```

Accediendo a los elementos: form.elements[]

La propiedad **elements[]** de un formulario es una colección, que contiene todos los objetos input dentro de un formulario. Esta propiedad es otro array, con todos los campos input en el orden en el cual aparecen en el código fuente del documento.

Generalmente, es mucho más eficaz y rápido referenciar a un elemento individual usando su ID, pero a veces, los scripts necesitan recorrer cada elemento del formulario, para comprobar que se han introducido sus valores correctamente.

```
const campoNombre = formulario.elements['nombre']; // Acceso por atributo name
const campoEmail = formulario.elements[1]; // Acceso por índice
```

Cada uno de estos elementos tiene sus propias propiedades, como value, checked, etc.

```
console.log(campoNombre.value); // Obtener el valor del campo "nombre"
```

Empleando la propiedad elements[], podemos hacer un bucle que recorra un formulario. Por ejemplo, si los campos son de tipo texto, los ponga en blanco:

```
let miFormulario = document.getElementById("contactar"); // guardamos la referencia del formulario en una vble.

if (! miFormulario) return false; // Si no existe ese formulario devuelve false.
        for (var i=0; i< miFormulario.elements.length; i++){
         if (miFormulario.elements[i].type == "text"){
                miFormulario.elements[i].value = "";
        }
}
```

Accediendo a los métodos del formulario

Los formularios también tienen métodos útiles, como:

- submit(): Envía el formulario programáticamente.
- reset(): Restablece todos los campos del formulario a sus valores iniciales.

formulario.submit(); // Enviar el formulario programáticamente
formulario.reset(); // Restablecer todos los campos del formulario

Ejemplos:

```
<form id="miFormulario" action="/enviar" method="post">
        <input type="text" name="nombre" value="Juan">
        <input type="email" name="email" value="juan@example.com">
        <input type="submit" value="Enviar">
</form>
```

Puedes interactuar con él en JavaScript de la siguiente manera:

```
const formulario = document.getElementById('miFormulario');
```

formulario.action = '/nuevo-destino'; // Cambiar la acción del formulario
formulario.method = 'GET'; // Cambiar el método de envío

const nombre = *formulario*.elements['nombre']; // Acceder a un campo específico
console.log(nombre.value); // "Juan"
nombre.value = 'Carlos'; // Cambiar el valor de un campo
formulario.submit(); // Enviar el formulario

Identificación de objetos de un formulario desde Js

Para poder trabajar con los objetos de un formulario, lo primero que necesitas saber es, cómo referenciar a ese objeto. Eso puedes hacerlo directamente a través de su ID, o bien con su nombre de etiqueta, o a través del formulario como hemos explicado antes.

- 1ª forma: Lo mejor es identificar cada uno de los objetos con un **atributo id** que sea único, y que no se repita en el documento, así para acceder a cualquier objeto dentro de nuestro documento o formulario lo haremos con: **document.getElementById("id-del-control")**

- 2ª forma: Otra forma es acceder a un objeto a través de su **atributo name:**

  document.name-Formulario.name-del-control

- 3ª forma: A través de su tag con: document.*getElementsByTagName(“p”);*

  CUIDADO: DEVUELVE UN ARRAY SIEMPRE

Por ejemplo, si consideramos un ejemplo sencillo de formulario:

```
<form id="formularioBusqueda" name=”formularioBusqueda” action="cgi-bin/buscar.pl">
        <p>
                <input type="text" id="entrada" name="cEntrada">
                <input type="submit" id="enviar" name="enviar" value="Buscar...">
        </p>
</form>
```

Todas las siguientes instrucciones obtienen el objeto entrada:

```
document.getElementById("entrada");
document.formularioBusqueda.cEntrada;
document.formularioBusqueda.elements[0];
document.forms["formularioBusqueda"].elements["cEntrada"];
document.forms["formularioBusqueda"].cEntrada;
```

Objeto input de tipo text

- Primero debes seleccionar el elemento <input> en tu documento HTML: con document.getElementById, a través del formulario,etc..

  ```
  <input type="text" id="miInput" name="nombre" value="Juan">
  const input = document.getElementById('miInput');
  ```

- Una vez que tienes el input, puedes acceder a varias **propiedades**:
  - **value**: Obtiene o establece el valor del input.
  - **defaultValue** Ajusta o devuelve el valor por defecto de un campo de texto.
  - **placeholder**: Texto que aparece como sugerencia cuando el campo está vacío.
  - **name**: Nombre asociado al input.
  - **disabled**: Si es true, deshabilita el input.
  - **readonly**: Si es true, el input es de solo lectura.
  - **maxLength**: Establece el número máximo de caracteres permitidos.
  - **size**: Establece el tamaño del input (número de caracteres visibles).
  - **type** Devuelve el tipo de un campo de texto

- **IMPORTANTE**: el contenido de un value es siempre una cadena de texto, y quizás puedas necesitar realizar conversiones numéricas si quieres realizar operaciones matemáticas con esos textos. Se usan para definir constantes, valores que no pueden modificarse.
- **Ejemplos**

  ```
  // Obtener el valor actual del input
  console.log(input.value); // "Juan"
   // Cambiar el valor del input
  input.value = 'Carlos';
  // Cambiar el texto de placeholder
  input.placeholder = 'Escribe tu nombre';
  // Establecer el input como solo lectura
  input.readOnly = true;
  // Establecer el tamaño visible del input
  ```

```
input.size = 30;
```

Métodos del objeto input de tipo texto

- **select**() Selecciona el contenido de un campo de texto
- **focus** ( ) : Establece el foco en el input.
- **blur** ( ) : Elimina el foco del input.

Ejemplo:

```
// Establecer el foco en el input
input.focus();

// Eliminar el foco del input
input.blur();

// Seleccionar todo el texto dentro del input
input.select();
```

Objeto input de tipo checkbox

Propiedades del objeto checkbox

Checked: Ajusta o devuelve el estado checked de un checkbox. Si está o no marcado.

DefaultChecked: Devuelve el valor por defecto del atributo checked. Si está o no marcado por defecto.

Name: Ajusta o devuelve el valor del atributo name de un checkbox.

disabled: Si es true, desactiva el checkbox (no se puede interactuar con él).

Type: Nos indica que tipo de elemento de formulario es un checkbox.

Value: Ajusta o devuelve el valor del atributo value de un checkbox.

Métodos del checkbox

- **focus**(): Establece el foco en el input.
- **blur**(): Elimina el foco del input.

Ejemplo:

```
const checkbox = document.getElementById('miCheckbox');

// Verificar si el checkbox está marcado
        if (checkbox.checked) {
          console.log('El checkbox está marcado');
        } else {
          console.log('El checkbox no está marcado');
        }

// Cambiar el estado del checkbox
checkbox.checked = true; // Ahora estará marcado


// Acceder al valor del checkbox

console.log(checkbox.value); // "1"

//Establecer el foco en el checkbox

checkbox.focus();
```

// Eliminar el foco del checkbox

checkbox.blur();

Objeto input de tipo radio

El manejo de inputs de tipo radio en JavaScript es similar al de los checkboxes, pero hay diferencias clave debido a que los radios se agrupan para permitir la selección de solo una opción dentro de un conjunto. Para ello deberemos asignar el **mismo atributo name** a cada uno de los botones del grupo.

1.- Si tienes un solo botón de radio, puedes seleccionarlo de la misma manera que cualquier otro input:

<input type="radio" id="opcion1" name="grupo" value="1">

const radio = document.getElementById('opcion1');

2.- Si tienes un grupo de botones de radio con el mismo atributo `name`, puedes seleccionarlos y trabajar con ellos en conjunto. Este es el caso más común:

<input type="radio" name="grupo" value="1"> Opción 1

<input type="radio" name="grupo" value="2"> Opción 2

<input type="radio" name="grupo" value="3"> Opción 3

*const radios = **document.getElementsByName**('grupo');*

Propiedades de los inputs de tipo radio

Los radios comparten muchas propiedades con los checkboxes:

- **checked**: Indica si el botón de radio está seleccionado o no (true o false).
- **value**: El valor del radio que se enviará si está seleccionado.
- **disabled**: Deshabilita el radio.
- **name**: Agrupa los radios, lo que permite seleccionar solo uno de ellos.

Métodos del input de tipo radio

Puedes usar métodos como focus(), blur()

Ejemplo:

```
let seleccionado;
for (let i = 0; i < radios.length; i++) {
  if (radios[i].checked) {
    seleccionado = radios[i].value;
    break;
  }
}

if (seleccionado) {
```

```
    console.log(`El radio seleccionado tiene el valor: ${seleccionado}`);
}
```

Objeto select

Acceder y manipular un <select> (lista desplegable) en JavaScript es muy común para manejar formularios. El proceso tiene algunas particularidades debido a que los <select> pueden tener múltiples opciones (<option>), y pueden ser de selección simple o múltiple.

```
<select id="miSelect">
  <option value="1">Opción 1</option>
  <option value="2">Opción 2</option>
  <option value="3">Opción 3</option>
</select>
```

Propiedades de select

Las propiedades más importantes de un <select> son:

- **value**: Devuelve el valor de la opción seleccionada.
- **selectedIndex**: Devuelve el índice (posición) de la opción seleccionada. *Los índices comienzan en*
- *la posición 0*
- **options**: Una colección de todas las opciones (<option>) dentro del <select>.
- **length**: Devuelve el número de opciones en el <select>.
- **multiple**: Indica si el <select> permite la selección múltiple (true o false).

Ejemplos:

```
// Obtener el valor de la opción seleccionada
console.log(select.value); // "1"

// Obtener el índice de la opción seleccionada
console.log(select.selectedIndex); // 0 (la primera opción)

// Verificar cuántas opciones hay en el select
console.log(select.length); // 3

// Verificar si el select permite múltiples selecciones
console.log(select.multiple); // false
```

Acceder a las opciones (<option>)

```
const opciones = select.options;

// Recorrer las opciones
for (let i = 0; i < opciones.length; i++) {
  console.log(`Opción ${i + 1}: ${opciones[i].text} (valor: ${opciones[i].value})`);
}
```

Propiedad innerHTML y textContent : propiedades en JavaScript que sirven para manipular el contenido de

un elemento HTML:

- innerHTML incluye etiquetas HTML y su contenido
- textContent: NO interpreta HTML, solo interpreta texto puro.

```
const elemento = document.getElementById("miElemento")

elemento.innerHTML = "<strong>Hola</strong> mundo!"; //mostrará en negrita: "Hola mundo!"

elemento.textContent = "<strong>Hola</strong> mundo!"; //mostrará el texto exacto
                                                       //<strong>Hola</strong> mundo!
```

Actualización de atributos

Podemos actualizar los valores de los atributos de elementos accediendo a ellos a través de un punto o bien con el método setAttribute().

Ejemplo:

```
<img src=”imagen.png” id =”i1”>
```

Forma 1:

```
document.getElementByd(“i1”). src=”nuevaImagen.png” ;
```

Forma 2:

```
let mi =  document.getElementByd(“i1”);
mi.setAttribute(“src” , “nuevaImagen.png”);
```

Estilos

Hay distintas formas de modificar estilos en javascript:

```
mi.style.backgroundColor = “red”;

mi.setAttribute(“style”, “background-color:red”);

mi.setAttribute(“class”, “text-rojo”);

mi.className = 'texto-rojo';
```

Ejercicios

1. Crea un archivo HTML con un párrafo (<p>) con el texto: “Texto Original”. Preguntar al usuario si quiere cambiar texto y en caso afirmativo, cambiar su texto a “Texto Cambiado”. En caso negativo, preguntar si quiere cambiar el color de la letra. En caso afirmativo, cambiar la letra a color naranja.

2. Crea un archivo HTML con un input de tipo ‘text’ con un valor, y un botón. Cuando se pulse el botón, preguntar al usuario vuestro nombre y modificar el valor del input con el valor introducido. Cambiar también el texto del botón a ‘pulsado’.

3. Crea un archivo HTML con 1 checkbox, desmarcado por defecto, y un botón. Al hacer clic en el botón, marcarlo automáticamente.

4. Crear una página HTML con 4 checkbox, con los valores: 1,2,3 y 5 y un botón. Al pulsar el botón, mostrar en un alert la suma obtenida sólo de los checkbox marcados. Después todos los check volverán a estar desmarcados. Crear el html sin formulario.

   Ejemplo, si marcamos los dos primeros, la suma es 3. Hacerlo de dos formas:
   - 1º: Los checkbox tienen un id para poder acceder a cada uno individual.
   - 2º: Los checkbox no tienen un id propio, tienen un name común.

5. Crea un grupo de 4 input de tipo radio con 4 valores y un botón sin formulario. Al pulsar el botón, indicar con un alert cuál fue la opción seleccionada. No tienen id.

6. Crear el mismo ejercicio con los input y el botón dentro de un formulario. ¿Se te ocurre forma de hacerlo a través del formulario?

7. Crear un formulario con 6 checkbox y un botón ‘validar’ . Cuando pulsemos el botón, validará que hay 3 o más checkbox marcados. Mostrar por alert un mensaje que indique si hay o no 3 o más de 3 checkbox marcados.

   Una vez realizado, crear otro checkbox independiente con dos botones nuevos: marcar y desmarcar. Cuando pulsemos el botón marcar, marcaremos el checkbox y cuando pulsemos el botón desmarcar, desmarcará el checkbox.

8. Crear un formulario con un input de tipo fecha (date) y 3 input de tipo text: nombre, apellido1 y apellido2. Asignarle un valor por defecto a los textos. Recuperar en javascript (en un bucle), **todos** los nombres (atributo name) de los elementos del formulario y mostrarlos con alert. Después mostrar **con alert sólo los input de tipo text**.

   Añadir al ejercicio anterior, un alert para obtener el nombre completo, (concatenando el valor del nombre y del apellido 1 y apellido 2), a través del **id** de cada campo.

   Añadir al ejercicio anterior, un alert para obtener el nombre completo, (concatenando el valor del nombre y del apellido 1 y apellido 2), a través del atributo **name** de cada campo.

9. Crear una página html con un campo de texto (input), poner vuestro nombre y al pulsar un botón, mostrareis el contenido del input debajo del botón.
   Pista: usar etiqueta div
   Añadir al ejercicio que después de pulsar el botón, limpie el cuadro de texto y ponga el foco en él.
   El nombre lo mostrareis en color azul y en negrita. Usar una clase de estilo llamado ‘resultado’.

10. Crear una página html con un formulario y con un objeto select provincias con 4 provincias (4 options).Crear un botón tal que al pulsarlo muestre en un alert la información de la provincia seleccionada: índice del option seleccionado, descripción y valor.

    Añadir otro objeto select **múltiple,** pero que permita la selección multiple, y otro botón. Al pulsarlo, mostrar por alert la descripción de las provincias seleccionadas. Deben aparecer separadas por coma.

11. Crear un formulario con un cuadro de texto ‘edad’ y un botón. Al pulsar el botón, comprobar que la edad insertada es mayor o menor de edad. Poner el resultado en un div debajo del botón. Si es mayor de edad el color del texto es verde, y en caso contrario rojo. En caso de que el valor introducido no sea un número indicarlo con un mensaje en rojo en el mismo div.

    1º: Utilizar estilos individuales

    2º: Crear dos clases de estilo, uno para el texto en rojo y otro para el texto en verde.