UD 4 – Manipulación de documentos web con JS

BOLETÍN DE EJERCICIOS

1. Realiza un script que muestre 100 veces el texto "Hola mundo" en la consola.

2. Realiza un script que utilice un bucle para mostrar en la consola la tabla de multiplicar del 9 con el siguiente formato:

   9 x 0 = 0

   9 x 1 = 9

   …

3. Realiza un script que le haga al usuario tres preguntas para averiguar su nombre (XXX), su primer apellido (YYY) y su segundo apellido (ZZZ). Después debe imprimirse en consola el texto: "Te llamas XXX YYY ZZZ"

4. Realiza un script que use una ventana *alert* para mostrar el siguiente menú al usuario:

   MENU CALCULÍN

   1 – Calcular el cuadrado de un número

   2 – Calcular el 20% de un número

   3 – Calcular la mitad de un número

   Usa el carácter de escape \n para introducir un salto de línea en el texto

   A continuación, se debe hacer uso de un prompt para pedir la opción escogida por el usuario.

   Si el número está comprendido entre 1 y 3 entonces se pedirá al usuario que introduzca el número y se mostrará en la consola el resultado del cálculo correspondiente.

   Si el número no era ninguno de los del menú entonces se mostrará el mensaje "Opción incorrecta" y el programa terminará.

5. Convertir valores en JS siempre es una “fiesta sorpresa” ejecuta el siguiente código y analiza los resultados obtenidos:

   alert(Number(" 123 "));

   alert(Number("123z"));

   alert(Number(true));

   alert(Number(false));

   alert(Number(null));

   alert(Number(undefined));

   NOTA: *NaN es el acrónimo de Not a Number*

6. Crea una función *divideNumeros(a, b)* que compruebe que los parámetros a y b son números válidos antes de realizar la división de ambos. Si todo está correcto se debe devolver la división a/b. Si alguno de los parámetros no es un número o bien b = 0 entonces se debe devolver el valor NaN.

   Posteriormente en el script, llama a la función que has creado probando los 3 casos posibles.

7. Crea una función llamada *validaNombre* que el pregunte al usuario su nombre y compruebe que el valor introducido no está vacío ni tiene un número inferior de 3 caracteres (busca en Internet cómo se hace). En el caso de que sea un nombre válido se debe retornar dicho nombre. En caso contrario se debe imprimir una alerta que informe al usuario de su error y se debe devolver el valor "invalid".

   Posteriormente, llama a la función que has creado e imprime el valor devuelto en la consola.

8. Crea un objeto que represente a un *producto* a la venta que tenga por nombre "Auriculares BT", precio 23 € y con stock de 44 unidades. A continuación, imprímelo en la consola.

   Añade un método llamado *importeEnStock*() que devuelva el producto del precio por las unidades en stock. Invócalo y muestra el resultado en la consola.

   Ahora debes decrementar las unidades en stock tres veces y subirle el precio un 5%.

   Vuelve a invocar al método *importeEnStock*() e imprime el resultado en la consola.

9. Crea una función llamada *reponeUnidades* que reciba dos parámetros: un objeto producto como el del ejercicio anterior y un número entero que representará el número de unidades que se repondrán en el stock del producto. La función debe incrementar el número de unidades en stock del producto con el número entero que se reciba como parámetro.

   A continuación, llama la función con un objeto que tenga 10 unidades y se repongan 20. Imprime en consola el objeto antes y después de invocar la función.

10. Crea un array vacío y cárgalo con tres objetos producto como los que creaste en el ejercicio 8. A continuación, recorre el array con una estructura for…of e imprime en la consola mensajes con la siguiente estructura:

    "Nombre: NNN – Precio XXX € - Stock: XX unidades – Importe en stock: XXX €"

11. Realiza una página web que incruste en el <body> un script que realice dos acciones:
    a. Imprime en la página web el mensaje "Hola mundo" rodeado de etiquetas <h1>
    b. Abre una ventana usando la función "alert" en la que se muestra el mensaje "Hola usuario/a"

12. Modifica el ejercicio anterior para que el código JS se guarde en un fichero *script.js* y que el documento HTML haga un enlace a dicho fichero.

13. Realiza una página web que inserte en el <body> un script que realice dos acciones:
    a. Le pida al usuario con un "prompt" su nombre y lo guarde en una variable *nombre.*
    b. A continuación, escriba en la página web el mensaje "Hola *nombre*" rodeado de etiquetas <h1>

14. Realiza una página web que inserte en el <body> un script que realice las siguientes acciones:
    a. Le pida al usuario con un "prompt" su nombre y lo guarde en una variable *nombre.*
    b. Le pida al usuario con un "prompt" sus apellidos y los guarde en *apellidos.*
    c. A continuación, escriba en la página web el mensaje "Hola *nombre apellidos*" rodeado de etiquetas <h1>

15. Realiza una página web que inserte en el <body> un script que realice las siguientes acciones:
    a. Le pida al usuario dos números con un "prompt"
    b. A continuación, escriba en la página web el mensaje "La suma de los números introducidos es *XXX* " rodeado de etiquetas <h1>

    OJO: Usa Number(…) para convertir los valores leídos al tipo number.

16. Partiendo del siguiente documento HTML

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8"/>
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Tu periódico</title>
</head>
<body>
  <h1 id="titular">Pon tu título aquí</h1>
  <p id="principal">Esto es un párrafo principal</p>
  <p class="secundario">Este es el primer párrafo secundario</p>
  <div>
    <p class="secundario">Este es el segundo párrafo secundario</p>
    <p class="secundario">Este es el tercer párrafo secundario</p>
  </div>
  <script></script>
</body>
</html>
```

Añade al script del final del body las instrucciones que permitan seleccionar e imprimir en consola los siguientes nodos:

- El primer párrafo con id “principal”

- El primer párrafo con clase “secundario”
- El primer párrafo con clase “secundario” contenido en un div
- El primer nodo title contenido en el head
- El primer nodo body

17. Tomando el HTML del ejercicio anterior, selecciona todos los nodos <p> del documento e imprímelos en la consola utilizando un bucle para ello.

18. Partiendo del siguiente documento HTML, modifícalo para que se añada un script que pregunte al usuario con un **prompt** qué texto quiere poner tanto en el titular como en el párrafo.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8"/>
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Tu periódico</title>
</head>
<body>
  <h1 id="titular">Pon tu título aquí</h1>
  <p id="parrafo">Esto es un párrafo de muestra</p>
</body>
</html>
```

Para ello debes saber que el texto introducido por el usuario para el titular debe interpretarse como texto puro. Sin embargo, el texto que suministre el usuario para el párrafo podrá contener etiquetas HTML que deberán ser interpretadas por el navegador.

19. Modifica el ejercicio anterior para que el contenido del párrafo que escriba el usuario se muestre en el documento HTML como un enlace a la siguiente dirección: https://developer.mozilla.org/es/

20. ¿Qué pasaría si el usuario introduce en el texto del párrafo el siguiente script?

<script> console.log(“Hola. Te estoy intentando hackear la página”) </script>

21. Inserta el siguiente <body> en la plantilla HTML que usamos en clase. A continuación, completa el script para que se rellene la tabla con los datos del array de objetos, creando una fila de encabezado con dos columnas tituladas: “Nombre” y “Teléfono”. *Debes usar la sintaxis de template strings.*

```
<body>
  <h1>Contactos telefónicos</h1>
  <table id="contact-table"></table>
  <script>
      let contacts = [
          { name: "Ángela", phone: 636021781 },
          { name: "Bea", phone: 654256755 },
          { name: "Juan", phone: 645888610 },
          { name: "Miguel", phone: 615992156 },
          { name: "Tomás", phone: 612415446 },
          { name: "Zaida", phone: 955359613 } ];
  </script>
</body>
```

22. Inserta el siguiente <body> en la plantilla HTML que usamos en clase. A continuación, completa la función *changeFighter* para que se genere un número aleatorio que permita seleccionar uno de los luchadores del array. A continuación, se actualizará el texto de las etiquetas h1 y h2 para que se correspondan con las propiedades *name* y *country* del luchador seleccionado y se actualizará la propiedad *src* de la imagen para que se corresponda con la URL que aparece en la propiedad *image* del objeto.

```
<body>
    <h1>Tu próximo oponente aparecerá en 3, 2, 1</h1>
    <h2></h2>
    <img src=""/>

    <script>
        function changeFighter() {
            let fighters = [
                {name: "Blanka", country: "Brazil", image:
"https://isilionangel.files.wordpress.com/2012/10/blanka.png?w=584"},
                {name: "Chun-Li", country: "China", image:
"https://i.pinimg.com/originals/a5/05/01/a50501331601d0936ce3b8db826ac148.jpg
"},
                {name: "Zangief", country: "Russia", image:
"https://upload.wikimedia.org/wikipedia/en/7/70/Super_Zangief.png"},
                {name: "Vega", country: "Spain", image:
"https://upload.wikimedia.org/wikipedia/en/4/45/SSFVega.png"},
                {name: "Ken", country: "U.S.A.", image:
"https://upload.wikimedia.org/wikipedia/en/2/2f/Ken_Masters_%28SF3_-
_Third_Strike%29.png"}
            ];

            // Escribe tu código aquí
        }
        setTimeout(changeFighter, 3000);
    </script>
</body>
```

23. Inserta el siguiente <body> en la plantilla HTML que usamos en clase. A continuación,

completa la función *rotateElements* para que los elementos del array roten cíclicamente de forma que se retire el primer elemento y se inserte al final del array.

Completa también la función *replaceLinks* que debe modificar cada uno de los enlaces <a> del documento de modo que se emparejen con los elementos del array, actualizando el texto mostrado con la propiedad *name* de cada uno de los objetos y se actualice el atributo *href* con las respectivas propiedades *url* de los objetos del array.

```
<body>
    <h1>Enlaces ascendentes</h1>
    <ul>
        <li><a href="#" target="_blank"></a></li>
        <li><a href="#" target="_blank"></a></li>
        <li><a href="#" target="_blank"></a></li>
        <li><a href="#" target="_blank"></a></li>
        <li><a href="#" target="_blank"></a></li>
    </ul>

    <script>
        let pictures = [
            { name: "La joven de la perla", url:
"https://upload.wikimedia.org/wikipedia/commons/thumb/d/d7/Meisje_met_de_pare
l.jpg/800px-Meisje_met_de_parel.jpg" },
            { name: "El beso", url:
"https://upload.wikimedia.org/wikipedia/commons/thumb/f/f2/El_beso%28Gustav_K
limt%29.jpg/640px-El_beso%28Gustav_Klimt%29.jpg" },
            { name: "Mujer asomada a la ventana", url:
"https://static3.museoreinasofia.es/sites/default/files/obras/AS02157.jpg" },
            { name: "Idilio en el mar", url:
"https://www.ecured.cu/images/8/80/Idilio-mar-sorolla_1.jpg" },
            { name: "La gran ola", url:
"https://ichef.bbci.co.uk/news/640/cpsprodpb/11D93/production/_121270137_gett
yimages-584047706.jpg" }
        ];

        function rotateElements(array) {}
        function replaceLinks(array) {}

        setInterval(function() {
                    replaceLinks(pictures);
                    rotateElements(pictures);
                }, 200); // Ejecuta la función cada 0.2 segundos

    </script>
</body>
```

NOTA: repasa el uso de los métodos push, pop, shift y unshift de los arrays

24. Descarga el fichero *cuadro-tsunami-alumnado.html* y completa las funciones llamadas poneXXX y quitaXXX de forma que se añadan o quiten las clases CSS correspondientes.

25. Descarga el fichero *wikipedia-dark-alumnado.zip*. Descomprímelo y abre el fichero index.html. Vamos a crear el efecto de pasar del *light-mode* al *dark-mode* y viceversa. Para

ello debes completar la función *toggleDarkMode* de modo que se alterne la aplicación de la clase “dark” cada vez que se llame a la función. Además, se debe seguir el siguiente orden:

a. Aplicar/desaplicar la clase “dark” a todos los
b. Aplicar/desaplicar la clase “dark” a todos los <a>
c. Aplicar/desaplicar la clase “dark” a todos los <h1>, <h2>…, <h6>

26. Crea un documento HTML desde cero que sea capaz de pintar una rejilla 2x2 con imágenes cuya altura sea de 300px. Las imágenes no deben aparecer inicialmente, lo que se busca es que vayan apareciendo gradualmente cada segundo. Cada imagen debe realizar una transición de 2 segundos de duración que modifique su altura desde 0px a 300px.

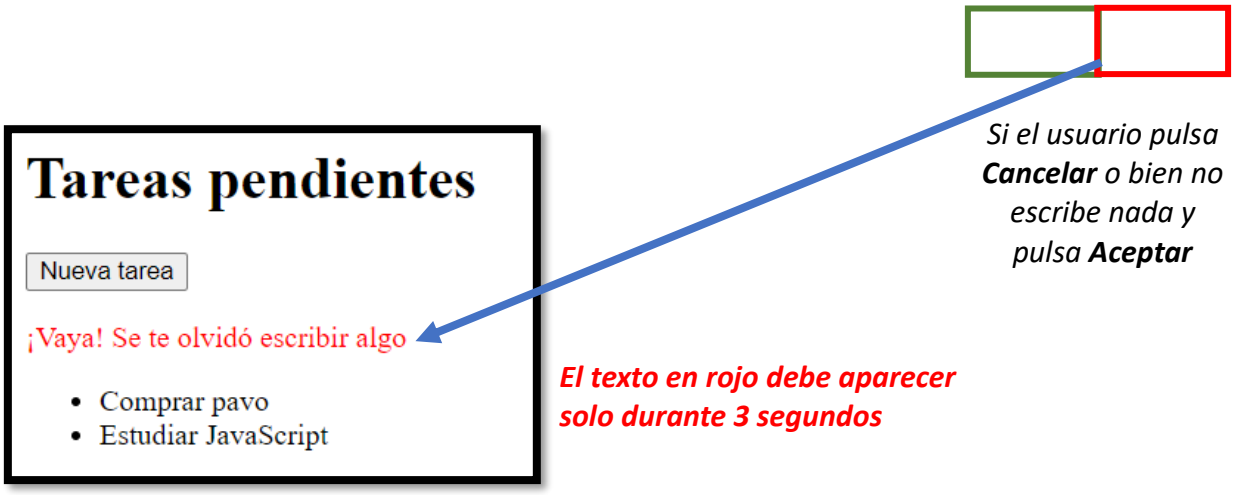
    Para programar que las imágenes aparezcan cada segundo se aconseja usar una estructura como la siguiente:

```
interval = 1000;
for(let img of images) {
    setTimeout(function() {
            makeBigger(img)
        }, interval);
    interval += 1000;
}
```

    Sabiendo que *images* será un array con los nodos <img> del documento y la función *makeBigger* simplemente añade una clase a la imagen que recibe como parámetro para que crezca hasta los 300px.

27. Haz una pequeña aplicación que registre tareas pendientes de realizar. El funcionamiento debe ser el siguiente:

(continúa)

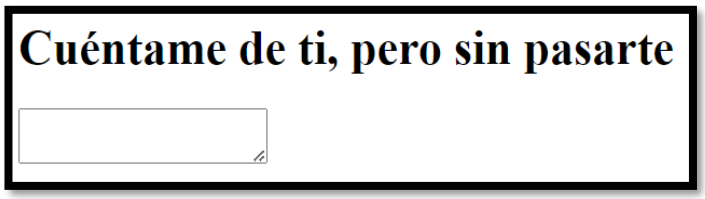


28. Partiendo del documento estudiado 'event-parameter.html' hazle las modificaciones necesarias para que ahora la selección de un elemento produzca la deselección del resto de elementos. Una vez lo hayas conseguido, modifícalo de nuevo para que se permita la selección múltiple de elemento, si la tecla CTRL está presionada.

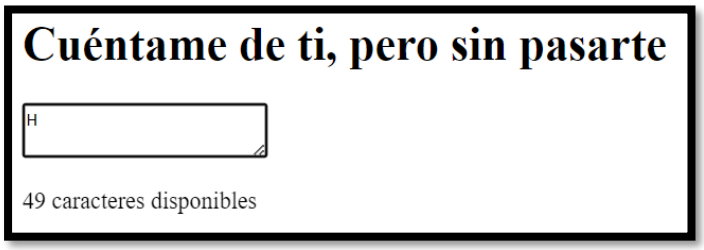
    NOTA: el objeto Event tiene una propiedad llamada ctrlKey que valdrá true si la tecla CTRL estaba presionada en el momento que se produjo el evento.

EJERCICIOS EXTRA DEL ANEXO

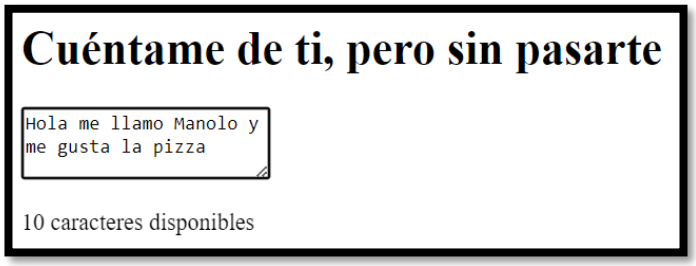
29. Realiza una interfaz de usuario que permita al usuario escribir en un elemento de tipo *textarea,* pero informando de los caracteres que le quedan disponibles del siguiente modo:



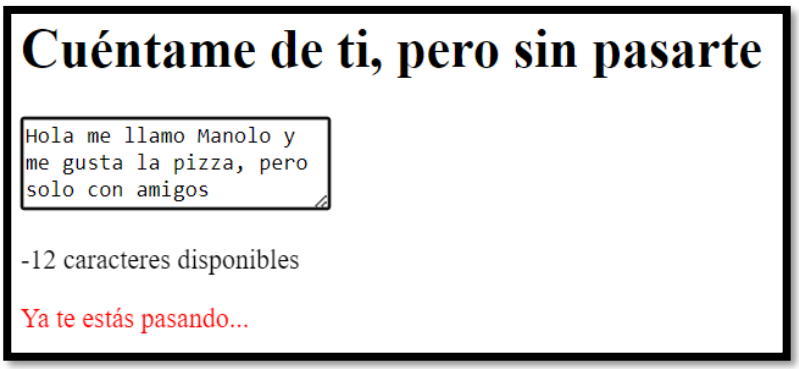
Tecleamos un carácter y aparece un mensaje con los caracteres disponibles restantes



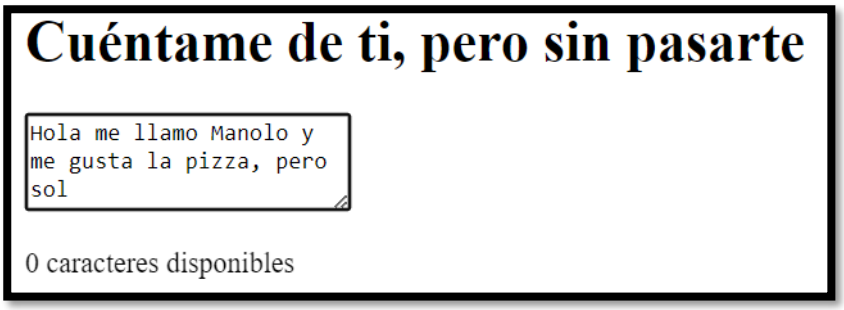
El mensaje se actualiza con cada pulsación de teclado



Cuando nos pasamos del límite aparece otro mensaje, pero nos permite seguir tecleando.



Si borramos caracteres y quedan 0 o más caracteres se ocultará el mensaje en rojo:



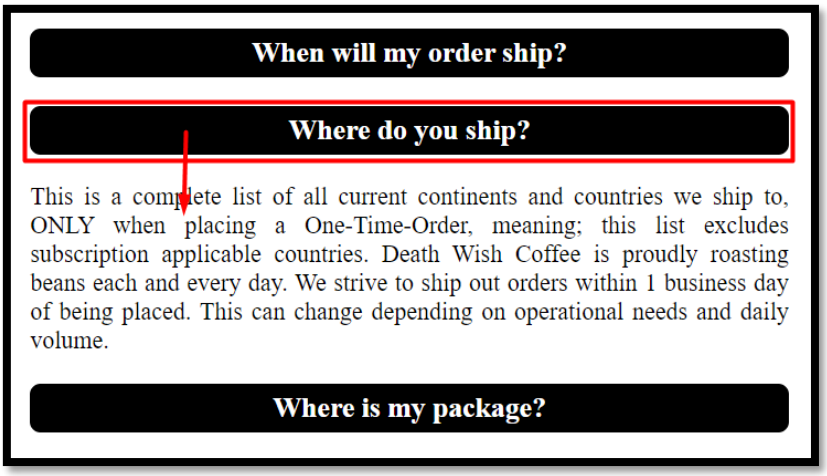
NOTAS:

- ***Debe escuchar el evento "input" del área de texto.***
- ***La propiedad value del nodo del elemento textarea contiene el texto tecleado por el usuario.***
- ***El manejador del evento debe definir una constante con valor 50 que representa el número de caracteres máximo disponible.***

30. Descarga el fichero ej30-FAQ-alumnado.html y completa el script para que inicialmente se cargue la página con todas las listas dinámicas ocultas. Se debe ver así:

Además, el botón “Expand All” debe mostrar todos los elementos con clase “faq-answer” y el botón “Reduce All” debe ocultarlos (hacerlos invisibles).

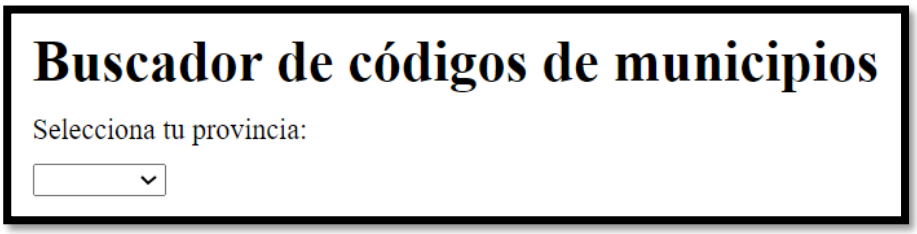
Por otro lado, cuando se pulse un elemento cuya clase sea “faq-question” se deben ocultar todas las respuestas con clase “faq-answer” excepto la asociada a la que provocó el evento.



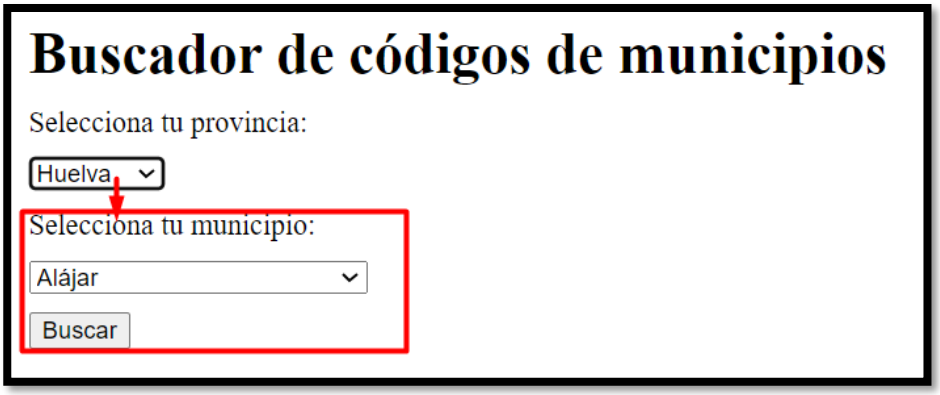
NOTA: para recuperar el elemento con clase “faq-answer”, partiendo del nodo con clase “faq-question” que provoca el evento, se debe usar la propiedad ***nextElementSibling*** *que nos devuelve el siguiente nodo hermano de uno dado.*

31. Descarga el fichero ej31-municipios-alumnado.html y completa el script para que se comporte como se describe a continuación.

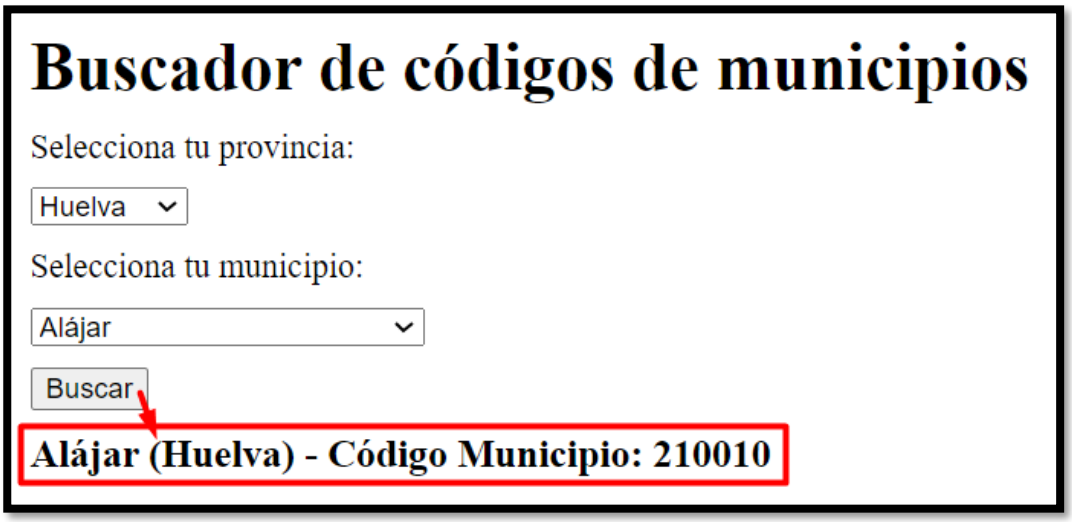
    La pantalla de bienvenida mostrará lo siguiente:



    Al seleccionar una provincia se comprobará que no se ha seleccionado la provincia "vacía" y, en caso contrario, se mostrará el identificado con "municipio-container". Además, se cargarán los municipios asociados a la provincia elegida.



    En el caso de que se pulse el botón "Buscar" se mostrará un mensaje justo debajo del botón:



    Si se selecciona una provincia distinta a la anterior entonces se recargan los municipios de la lista desplegable.

    En el caso de que se seleccione de nuevo la provincia "vacía" se ocultará el identificado con "municipio-container", mostrándose la pantalla de bienvenida.

32. Descarga el fichero ej32-only-numbers-alumnado.html y completa el script para que el

<input> solo admita números, impidiendo (prevent) que aparezcan otros caracteres. Se debe seguir el siguiente comportamiento:

Si se introducen números, aparecerán en el input con normalidad:

Sin embargo, si se teclean letras o signos de puntuación, estos no deben aparecer en el input, pero sí que debe mostrarse un mensaje de error como el siguiente:

Por último, el mensaje de error debe desaparecer automáticamente a los 2 segundos.

NOTA: el evento que nos interesa procesar es keydown. El objeto Event asociado a dicho evento tiene una propiedad key que contiene el valor del carácter asociado a la tecla presionada.