

## Hito 1 PRIMER TRIMESTRE 2021

### CUESTIÓN 1.

#### *Declaración de variables en JavaScript*

- **Explica qué tipado (débil o fuerte) se aplica en JavaScript**

JavaScript se caracteriza por estructuras de tipado débil donde la declaración de variables no exige la asociación con un tipo de datos de forma implícita y unívoca, a cualquier variable se le puede asignar (y reasignar) valores de todos los tipos.

- **Realiza un ejemplo con declaración de diferentes variables: texto, numéricas, booleanas y de listas. Identifica correctamente el tipo de dato de cada una.**

Declaro las variables y las imprimo por consola, con el tipo de variable que es cada una.

```
<script>
  var a = 2; // Number (int)
  var b = 2.5; // Number (float)
  var c = "Hola"; // String
  var d = true; // Boolean
  var array = ["Soy String", 5, true]; // Array literal

  console.log("La variable a: " + a + ", es de tipo: " + typeof(a));
  console.log("La variable b: " + b + ", es de tipo: " + typeof(b));
  console.log("La variable c: " + c + ", es de tipo: " + typeof(c));
  console.log("La variable d: " + d + ", es de tipo: " + typeof(d));
  console.log("La variable array: " + array + ", es de tipo: " + typeof(array));
</script>
```

La consola muestra:

```
La variable a: 2, es de tipo: number
La variable b: 2.5, es de tipo: number
La variable c: Hola, es de tipo: string
La variable d: true, es de tipo: boolean
La variable array: Soy String,5,true, es de tipo: object
```

Nota: el array es de tipo Objeto porque un objeto es una colección de propiedades.

- **Explica la diferencia entre let, var y const con un ejemplo.**

**let:** es una variable la cual puedes cambiar su valor, solo funciona dentro del bloque donde está declarada. Las variables let son block-scope (su ámbito es su bloque).

```
let a = "hola";
let a = "adios";
console.log(a);
```

Identifier 'a' has already been declared

let no permite redefinir la misma variable en el mismo bloque.

En diferentes bloques es posible definir múltiples variables con el mismo nombre siendo cada una de ellas una variable diferente.

```
let a = "hola";
if(true){
  let a = "adios";
  console.log("Dentro del if: " + a);
}
console.log("Fuera del if: " + a);
```

```
Dentro del if: adios
Fuera del if: hola
```

**var:** es una variable la cual puedes cambiar su valor, el ámbito de una variable declarada con var, es su contexto de ejecución. El ámbito de una variable var declarada fuera de la función es global. En el caso de var las variables pueden ser de global-scope, local-scope y block-scope, es decir que su ámbito dependerá de donde las declaremos.

```
var b = "hola";
var b = "adios";
console.log(b);
```

var permite redefinir la misma variable en el mismo bloque.

```
var b = "hola";
if(true){
  var b = "adios";
  console.log("Dentro del if: " + b);
}
console.log("Fuera del if: " + b);
```

```
Dentro del if: adios
Fuera del if: adios
```

**const:** es una variable la cual no puedes cambiar su valor una vez definida, las variables const son block-scope (su ámbito es su bloque).

```
const a = "hola";
console.log(a);
a = "adios";
console.log(a);
```

```
hola
✖ ▶ Uncaught TypeError: Assignment to constant variable.
   at Ejercicio1.html:55
```

No te permite cambiar su valor, ni redefinir la variable

```
const a = "hola";
const a = "adios";
```

```
Identifier 'a' has already been declared
```

## CUESTIÓN 2.

### DOM vs BOM en JavaScript

- Diseña una aplicación web con un formulario, imágenes y tablas.

He creado un formulario muy simple en el que se pide el nombre, el email y el sexo todos ellos requeridos para enviar el formulario.

```
<form action="">
  <label for="nombre">Nombre: </label>
  <input type="text" name="nombre" id="nombre" required />
  <br><br>
  <label for="email">Email: </label>
  <input type="email" name="email" id="email" required />
  <br><br>
  <label>Sexo: </label>
  <input type="radio" name="radiob" id="r1" value="Masculino" required />
  <label for="r1">Masculino</label>
  <input type="radio" name="radiob" id="r2" value="Femenino" required />
  <label for="r2">Femenino</label>
  <input type="radio" name="radiob" id="r3" value="Otro" required />
  <label for="r3">Otro</label>
  <br><br>
  <button onclick="enviar(event)">Enviar</button>
</form>
```

Ya que los datos no se van a enviar a ningún lado y solo se van a tratar en funciones simples de JavaScript, no he puesto ningún action, ni el método, y no he puesto un input type submit, en su lugar he puesto un botón (los required no funcionan si no tiene un input type submit).

He creado un div que contiene 3 imágenes y un botón cuya función se explicara en el siguiente apartado.

```
<div id="fotos">
  
  
  
  <button onclick="mostrar()">Mostrar imágenes de nuevo</button>
</div>
```

Por último, he creado otro formulario en el que insertando un numero te generará una tabla de multiplicar de dicho número.

```
<div>
  <p>Generar Tabla de Multiplicar</p>
  <form action="">
    <label for="numero">Numero: </label>
    <input type="number" min="-1000" max="1000" name="numero" id="numero" placeholder="Numero" required>
    <input type="submit" value="Enviar" onclick="tabla(event)">
  </form>
</div>
```


Resultado final de la página.

Nombre:

Email:

Sexo: ☐ Masculino ☐ Femenino ☐ Otro

Enviar



Mostrar imágenes de nuevo

Generar Tabla de Multiplicar

Numero:

Enviar

- *Modifica los contenidos de cada uno de los elementos utilizando JavaScript mediante acceso al Document object model.*

Para la tabla lo que he hecho a sido recoger los datos y mostrarlos debajo del propio formulario una vez pulsado el botón enviar que llama a una función específica para realizar esto.

```
function enviar(event) {
    event.preventDefault();
    let div = document.getElementById("resultadoFormulario");
    //eliminamos todos los hijos del div en el caso que los tuviera
    //por si se reenvia el formulario borrar el anterior
    while (div.firstChild) {
        div.removeChild(div.lastChild);
    }
    //obtenemos los datos del formulario
    let nombre = document.getElementById("nombre").value;
    let email = document.getElementsByTagName("input");
    let sexo = document.querySelector('input[name="radiob"]:checked');
    //Validacion del formulario
    if (nombre.length != 0 && email[1].value.length != 0 && sexo != null) {
        //Creamos un text node con la informacion, lo añadimos al p, y el p lo añadimos al div
        let nombreTxt = document.createTextNode("Nombre: " + nombre);
        var p = document.createElement("p");
        p.appendChild(nombreTxt);
        div.appendChild(p);

        let emailTxt = document.createTextNode("Email: " + email[1].value);
        var p = document.createElement("p");
        p.appendChild(emailTxt);
        div.appendChild(p);

        sexo = sexo.value;
        let sexoTxt = document.createTextNode("Sexo: " + sexo);
        var p = document.createElement("p");
        p.appendChild(sexoTxt);
        div.appendChild(p);
    } else {
        let errorTxt = document.createTextNode("No puedes Enviar un Formulario vacio");
        var p = document.createElement("p");
        p.appendChild(errorTxt);
        div.appendChild(p);
    }
}
```

Esta función valida el formulario para que no se pueda dejar ningún campo vacío, recoge los datos del DOM y los muestra en un div creado debajo del formulario.

Resultado:

Nombre:

Email:

Sexo: ☒ Masculino ☐ Femenino ☐ Otro

Nombre: raul

Email: raul.medina@campusfp.es

Sexo: Masculino

Para las fotos he creado una función que la oculta la imagen, esta función se activa cuando el ratón pasa por encima de la imagen. Esta función recoge el elemento del DOM mediante el event.srcElement.

```
function ocultar(event) {  
    let target = event.srcElement;  
    target.style.visibility = "hidden";  
}
```

Luego con un clicando un botón se vuelven a mostrar todas las fotos que se hubieran ocultado.

```
function mostrar() {  
    let imagenes = document.getElementsByTagName("img");  
    //Conversion de collection a array  
    let imagenesArray = Array.from(imagenes);  
  
    imagenesArray.forEach(element => {  
        if(element.style.visibility == "hidden"){  
            element.style.visibility = "visible";  
        }  
    });  
}
```

Aquí el ratón estaba sobre la segunda imagen la cual se oculta.



Para la tabla, una vez enviado un numero mediante el formulario, se genera una tabla de multiplicar de dicho número en un div debajo del formulario con la función siguiente.

```
function tabla(event) {
    event.preventDefault();

    let numero = document.getElementById("numero").value;
    let divtabla = document.getElementById("tabla");
    //validacion de formulario
    if (numero.length != 0) {
        //eliminamos todos los hijos del div en el caso que los tuviera
        //por si se reenvia el formulario borrar el anterior
        while (divtabla.firstChild) {
            divtabla.removeChild(divtabla.lastChild);
        }
        // Creo los elementos <table> y <tbody>
        var tabla = document.createElement("table");
        var tblBody = document.createElement("tbody");

        for (var i = 0; i <= 10; i++) {
            // Creo las hileras y las celdas de la tabla
            var hilera = document.createElement("tr");

            var celda = document.createElement("td");

            var textoCelda = document.createTextNode(i + " x " + numero + " = " + (i * numero));

            celda.appendChild(textoCelda);
            hilera.appendChild(celda);

            // agrega la hilera al final de la tabla
            tblBody.appendChild(hilera);
        }
        // agregamos el tbody a la tabla
        tabla.appendChild(tblBody);
        // metemos la tabla dentro de divtabla
        divtabla.appendChild(tabla);
        // modifica el atributo "border" de la tabla
        tabla.setAttribute("border", "1");
    } else {
        while (divtabla.firstChild) {
            divtabla.removeChild(divtabla.lastChild);
        }

        let errorTxt = document.createTextNode("No puedes Enviar un Formulario vacio");
        var p = document.createElement("p");
        p.appendChild(errorTxt);
        divtabla.appendChild(p);
    }
}
```

Como en la función del formulario anterior, primero comprueba si hay otra tabla creada y la borra en el caso de que la hubiera para no añadir otra tabla, también verifica si el formulario está vacío. Luego se crean los elementos necesarios (table, tbody, tr...) y se van agregando y rellendo en el orden correspondiente para la creación de la tabla y por ultimo se modifica el atributo de borde de la tabla.

Resultado:

Generar Tabla de Multiplicar

Numero: <input type="text" value="6"/> <input type="button" value="Enviar"/>
------------------------------------------------------------------------------

0 x 6 = 0
1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60

- Realiza al menos tres ejemplos de manejo de BOM. Por ejemplo, identificar navegador, consultar dimensiones de ventana, mostrar listado de páginas visitadas...

Para identificar el navegador he hecho una función que identifica el navegador y lo filtra mediante unos if para sacar el nombre del navegador, solo lo he hecho con los principales navegadores, ya que hay que hacerlo a mano si no quieres descargar ninguna librería externa.

```
function identificarNavegador() {  
    let navegador = navigator.userAgent.toLowerCase();  
  
    let body = document.getElementsByTagName("body");  
    let p = document.createElement("p");  
    let texto = "";  
    if (navegador.indexOf('chrome') > -1) {  
        texto = document.createTextNode('El navegador es Chrome');  
    }  
    if (navegador.indexOf('firefox') > -1) {  
        texto = document.createTextNode('El navegador es Firefox');  
    }  
    if (navegador.indexOf('opera') > -1) {  
        texto = document.createTextNode('El navegador es Opera');  
    }  
    if (navegador.indexOf('MSIE') > -1) {  
        texto = document.createTextNode('El navegador es Internet Explorer');  
    }  
    p.appendChild(texto);  
    body[0].appendChild(p);  
}
```

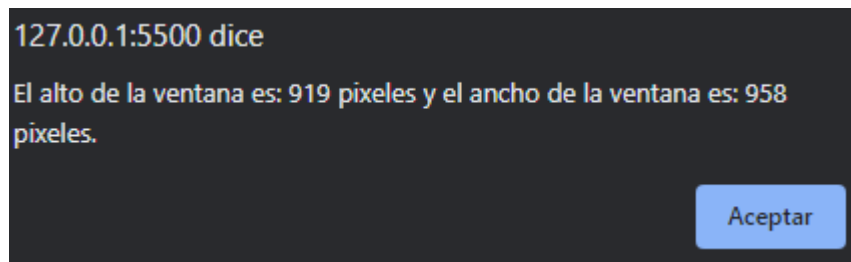
El resultado se añade al cuerpo del body en forma de párrafo.

También he creado una función para sacar el tamaño de la ventana del navegador en píxeles.

```
function tamañoVentana() {  
    let ancho = window.innerWidth;  
    let alto = window.innerHeight;  
  
    let boton = document.createElement('button');  
    boton.innerText = 'Tamaño de ventana';  
    boton.setAttribute('onclick', 'alertaTamanios("ventana",' + alto + ',' + ancho + ');');  
    document.body.appendChild(boton);  
}
```

Se mostrará en forma de alerta pulsando el botón creado y añadido al body.

La alerta:

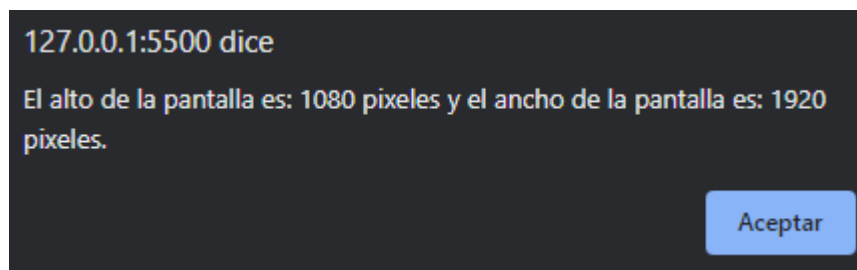


Al igual que el tamaño de la pantalla.

```
function tamañoPantalla() {  
    let ancho = screen.width;  
    let alto = screen.height;  
    let boton = document.createElement('button');  
    boton.innerText = 'Tamaño de pantalla';  
    boton.setAttribute('onclick', 'alertaTamanios("pantalla",' + alto + ',' + ancho + ');');  
  
    document.body.appendChild(boton);  
}
```

Este también se mostrará en forma de alerta pulsando el botón creado y añadido al body.

La alerta:



Estas 2 funciones utilizan una función para mostrar la alerta, que recoge los parámetros: tipo, alto y ancho.

```
function alertaTamanios(tipo, alto, ancho) {  
    window.alert("El alto de la " + tipo + " es: " + alto + " píxeles y el ancho de la " + tipo + " es: " + ancho + " píxeles.");  
}
```



Y todas estas funciones se cargan en el onload del body, agrupadas en una función.

```
function todo() {  
    identificarNavegador();  
    tamañoVentana();  
    tamañoPantalla();  
}
```

El resultado es el siguiente:

El navegador es Chrome

Tamaño de ventana

Tamaño de pantalla

### CUESTIÓN 3.

#### *Almacenamiento local*

- *En una página web almacena información utilizando cookies, LocalStorage y SessionStorage y explica brevemente sus diferencias.*

Las cookies sirven para almacenar poca información, al crearlas puedes establecer un tiempo de “vida”, el dominio, la ruta... la seguridad de estas es limitada.

La información de LocalStorage perdura hasta que se limpia el caché e información local del navegador, se puede almacenar mucha información. La seguridad de estas es bastante fuerte ya que es la del propio navegador.

La información de SessionStorage perdura hasta que se cierra el navegador o la pestaña donde se encuentra, estas pueden almacenar más información que una cookie.

```
<script>  
    //creacion de una cookie  
    document.cookie = "value = cookie";  
  
    //Creacion de una localStorage  
    localStorage.setItem("value","localStorage");  
  
    //Creacion de una sessionStorage  
    sessionStorage.setItem("value","sessionStorage");  
  
</script>
```

- *¿En qué carpeta de Windows se almacena cada dato?*

Depende del navegador las cookies se almacenan en distintas rutas:

**Internet Explorer** C:\Users\user\AppData\Local\Microsoft\Windows\INETCookies

**Google Chrome** En el archivo llamado cookies en la ruta:

C:\Users\user\AppData\Local\Google\Chrome\User Data\Default\Cookies

**Mozilla Firefox** En el archivo llamado cookies.sqlite en la ruta:

C:\Users\user\AppData\Roaming\Mozilla\Firefox\Profiles\usuario\cookies.sqlite

**Opera** En el archivo llamado cookies en la ruta:

C:\Users\usuario\AppData\Roaming\Opera Software\Opera Stable\cookies

- *Consultar los datos de cookies, LocalStorage y SessionStorage de la web*

Para consultar cada una de estas he creado una función distinta.

Para las cookies la función que se activa con un botón recoge todas las cookies, las separa y guarda en un array, y posteriormente recorre ese array e imprime en un párrafo en la web.

```
function consultarCookies() {  
  let cookies = document.cookie;//obtenemos todas las cookies  
  
  let listaCookies = cookies.split(";");//array list de todas las cookies  
  
  let valores;  
  let parrafo;  
  let txtParrafo;  
  h2 = document.createElement("h2");  
  txth2 = document.createTextNode("Cookies");  
  h2.appendChild(txth2);  
  document.body.appendChild(h2);  
  listaCookies.forEach(element => {  
    valores = element.split("=");  
    console.log(valores[0] + " " + valores[1]);  
    parrafo = document.createElement("p");  
    txtParrafo = document.createTextNode("Name: " + valores[0] + ", Value: " + valores[1]);  
    parrafo.appendChild(txtParrafo);  
    document.body.appendChild(parrafo);  
  });  
}
```

Consultar cookies

Consultar localStorage

Consultar sessionStorage

## Cookies

Name: value, Value: cookie

Para las LocalStorage la función recoge un array de las keys de localStorage y mediante un bucle, se rellena un array con los datos de las localStorage, y como en las cookies, este array se imprime en un párrafo en la web.

```
function consultarLocalStorage() {
  let listaLocalStorage = [];
  let keys = Object.keys(localStorage)//array de keys de local storage
  let numeroKeys = keys.length;//longitud del array

  while (numeroKeys--) {//bucle para guaydar los valores del localStorage en el array
    listaLocalStorage.push("Key: " + keys[numeroKeys] + ", Value: " + localStorage.getItem(keys[numeroKeys]));
  }

  let parrafo;
  let txtParrafo;
  h2 = document.createElement("h2");
  txth2 = document.createTextNode("LocalStorage");
  h2.appendChild(txth2);
  document.body.appendChild(h2);
  listaLocalStorage.forEach(element => {//recorremos el array y lo ponemos en un parrafo
    console.log(element);
    parrafo = document.createElement("p");
    txtParrafo = document.createTextNode(element);
    parrafo.appendChild(txtParrafo);
    document.body.appendChild(parrafo);
  });
}
```

Consultar cookies

Consultar localStorage

Consultar sessionStorage

## LocalStorage

Key: value, Value: localStorage

Para las SessionStorage la función es muy similar a la de localStorage, recoge un array de las keys de sessionStorage y mediante un bucle, se rellena un array con los datos de las sessionStorage, y se imprime en un párrafo en la web.

```
function consultarSessionlStorage() {
  let listaSessionStorage = [];
  let keys = Object.keys(sessionStorage)//array de keys de session storage
  let numeroKeys = keys.length;//longitud del array

  while (numeroKeys--) {//bucle para guaydar los valores del sessionStorage en el array
    listaSessionStorage.push("Key: " + keys[numeroKeys] + ", Value: " + sessionStorage.getItem(keys[numeroKeys]));
  }

  let parrafo;
  let txtParrafo;
  h2 = document.createElement("h2");
  txth2 = document.createTextNode("SessionStorage");
  h2.appendChild(txth2);
  document.body.appendChild(h2);
  listaSessionStorage.forEach(element => {//recorremos el array y lo ponemos en un parrafo
    console.log(element);
    parrafo = document.createElement("p");
    txtParrafo = document.createTextNode(element);
    parrafo.appendChild(txtParrafo);
    document.body.appendChild(parrafo);
  });
}
```

Consultar cookies

Consultar localStorage

Consultar sessionStorage

## SessionStorage

Key: value, Value: sessionStorage

Key: IsThisFirstTime\_Log\_From\_LiveServer, Value: true

---