**INDICE**

1. [**Intro, tipos de lenguaje**](#introleng)**.**
2. [**Algoritmos.**](#algoritmos)
3. [**JavaScript.**](#JS)
4. [**Variables.**](#variables)
5. **[Arrays, GetElementById.](#arrays)**
6. [**Estructura IF.**](#INTROIF)
7. [**Booleanos y Operadores.**](#Booleanos)
8. [**Estructura For/bucles**](#forbucle)
9. [**Funciones**](#funciones)
10. **. DOM**
11. [**. Formularios**](#events)

**Lenguaje de programación:**

Es un conjunto de símbolos y letras que se usan para crear aplicaciones.

Se llama lenguaje porque esta formado por un: léxico, sintaxis y una semántica. ( lo contienen TODOS los lenguajes de programación ).

Lexico: símbolos permitidos y vocabulario

Sintaxis: reglas que indican como se constituye el lenguaje.

Semantica: reglas que permiten saber el significado de las expresiones construidas.

**Lenguaje Maquina:**

Lenguaje de los ordenadores.

Estructura adaptada a los rodenadores

Programacion difícil que requiere conocer la estructura física del PC.

Se obtienen programas MUY eficientes.

Caracteristicas: Instrucciones expresadas en binario, hexadecimal u octal, datos referenciados por las direcciones de memoria donde están, instrucciones que hacen operaciones simples, instrucciones rigidas.



**Lenguaje ensamblador:**

Intento de hacer mas humano el lenguaje de los ordenadores.

Se traduce al lenguaje maquina con el ensamblador.

Programas muy eficientes.

El lenguaje continua estando ligado al procesador

Caracteristicas: se usa una notación simbólica para expresar los códigos, permite un direccionamiento simbólico de variables.



**Lenguaje de ALTO NIVEL:**

Se independiza el lenguaje de programación de la maquina.

Utilizan expresiones parecidas a los humanos.

Necesitan un compliador o interprete.

Caract: Independientes de la arquitectura física de la maquina, requiere un programa que traduzca a lenguaje maquina, una sentencia de alto nivel da lugar a varias instrucciones de lenguaje maquina, hay instrucciones potentes de uso frecuente como funciones matemáticas…, estos son MENOS eficientes.

**Compiladores e intérpretes:**

Un traductor es un programa que toma como entrada un programa escrito en lenguaje de alto nivel (código fuente) y proporciona como salida otro programa semánticamente igual llamado código objeto.

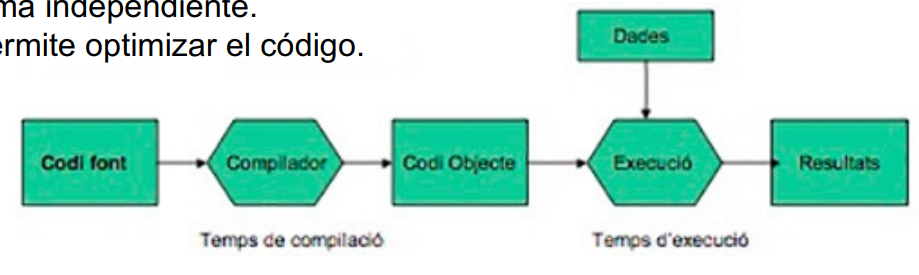
**COMPILADORES (lenguajes compilados)**:

Traduce el código fuente a un código objeto escrito en LM (lenguaje máquina).

Avisa al programador de errores durante el proceso y no se genera el código objeto hasta que el código fuente sea correcto.

Los códigos fuente y objeto se guardan en dos ficheros de forma independiente.

Todo esto permite optimizar el código.



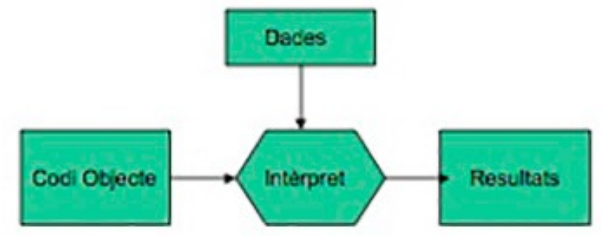
**INTERPRETES (lenguajes interpretados):**

Aquí se interpreta el código linea a línea (no todo el código de golpe como los compilados).

Se usa directamente el código fuente para ejecutar programas.

Cada vez que se ejecuta el programa se ejecuta el código.

Es más lento y la optimización del código es la del propio código (la optimización del código la tiene que hacer el propio programador) .Prácticamente la mayoría de los lenguajes ‘’famosos’’ hoy dia son interpretados. (JS, Python, PHP…).



**Algoritmos:**

Es una expresión del programa que yo quiero hacer a alto nivel.

Planteamiento del problema (qué es lo que queremos de nuestro código)

Analisis para generar un algoritmo.

Programación.

Prueba y depuración del programa.

Explotación de la aplicación.

Los lenguajes de programación se utilizan para solucionar los problemas planteados y ejecutar lo que nosotros queremos que haga nuestra aplicación.

Las acciones de un algoritmo se expresan en los programas con instrucciones, sentencias y proposiciones.

**INSTRUCCIONES BASICAS DE TODO LENGUAJE:**

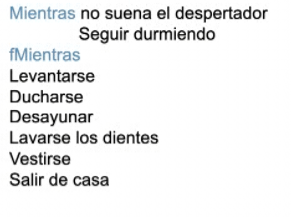
Instrucciones de entrada / salida. **Se necesita recibir datos para dar salida a un resultado.**

Instrucciones aritmético-lógicas**. Las mas famosas son la suma, resta, modulo…**

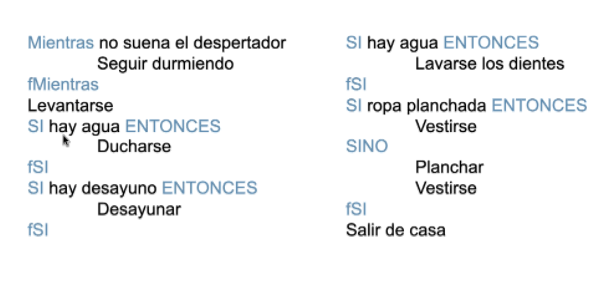
Instrucciones selectivas. **Son las que permiten diferencias de una acción u otra en el código.**

Instrucciones repetitivas. **Permiten hacer bucles (ejemplo básico: si yo estoy durmiendo // mientras no suene el despertador seguir durmiendo // Si suena el despertador, Despertarse // Eso es un bucle adaptado a la vida real.**

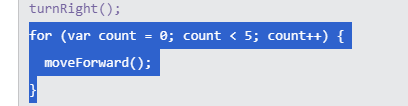
**Al final un algoritmo es un conjunto de pasos a seguir para resolver un problema concreto sin ambigüedad alguna en un tiempo finito.**

****

Estructuras algoritmo:



**Esto seria un ejemplo de bucle:**

****

**Estas creando una variable count y le dices que empieza en 0 y que mientas sea menor de 5 le sumes una unidad (count++) y lo ejecute con moveforward (moverse hacia delante).**

**If/else**

****

**En este juego vemos la función if y else (else = sino).**

**En este caso le estamos diciendo al juego que repita esto hasta encontrar la planta: Si en el camino puedes ir de frente, seguir andando hacia delante. SINO (else) girar a la izquierda.**

**JAVASCRIPT**

Nosotros usaremos JavaScript en nivel cliente (no servidor) aunque también se puede usar en entorno servidor (NodeJS).

Javascript es un lenguaje de guión para el desarrollo de paginas web, este va encajado en el HTML al igual que el CSS, JS nos permitirá tener comportamientos dinámicos en la pagina web.

**Normas Básicas**

No se tienen en cuenta los espacios en blanco y las nuevas líneas.

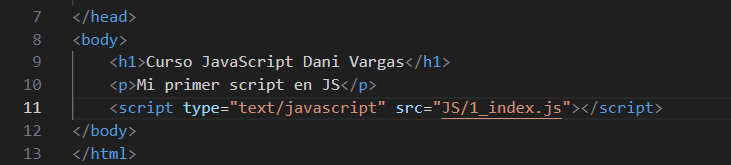
Se distinguen las mayúsculas y las minúsculas.

No se define el tipo de las variables.

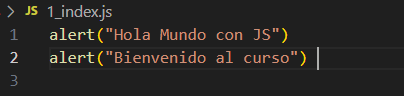
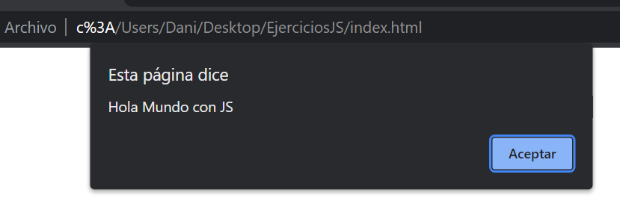
Se pueden incluir comentarios ( // monolinea /\* XXXX \*/ Multilinea ).

**Introducir hoja JS dentro de nuestro html:**

Lo codificaremos con la etiqueta script como último elemento del body, poniendole un type=”text/javascript” (esto es opcional ponerlo) y la url con la ruta donde tengamos el archivo.



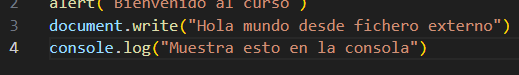
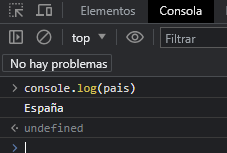
En este caso ponemos un script de tipo **alert** para que salga una ventana emergente cuando cargamos la página:

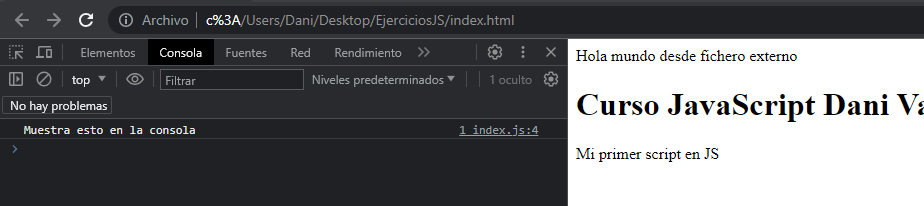


Esta ventana se cargará y al darle a aceptar a las dos ventanas emergentes, nos cargará la página.

Document write para añadir algo de texto al cuerpo de la página.

Console.log para mostrar algo en la consola de herramientas para desarrollador.





**VARIABLES**

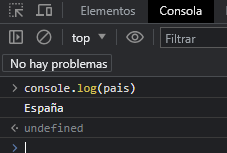
**Strings o Numeros:**

Al introducir el valor a una variable hay dos posibilidades, o poner un numero (sin comillas) o poner un string (con comillas). Sin comillas solo pueden ser números ya que es valor de tipo number, mientras entre comillas puede ser tanto números como palabras o incluso números decimales con comas.

Las variables nunca pueden empezar por un numero. Es decir nunca puedes poner var 3 = 3.

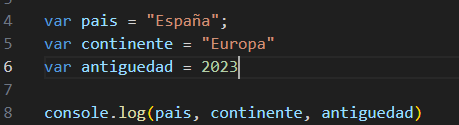
**Son como un almacenamiento / contenedor de información:**

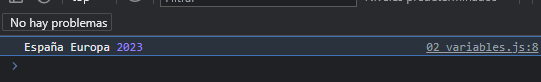
Se codifica mediante var más el nombre de la variable y su contenido entre comillas:



Al hacer console log en la consola de herramientas preguntándole por la variable país nos devolverá el nombre de la variable.

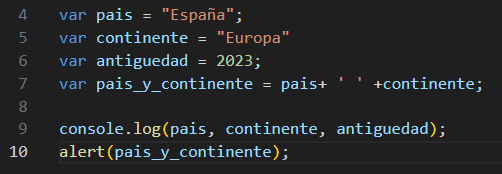
Se puede hacer que con el mismo archivo JS aparezca en la consola el mensaje al cargar la pagina con una cadena de texto:

El numero puede ir sin comillas y al mostrarse en la página se mostrara de un color distinto en la consola.

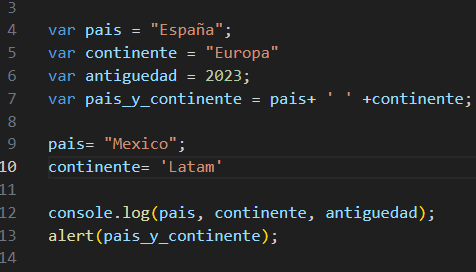


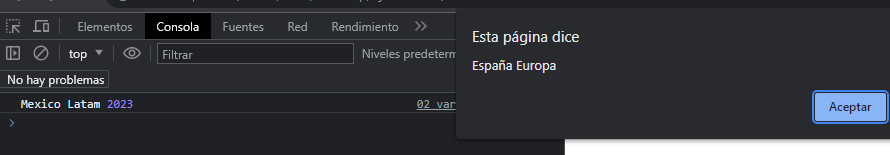
**Concatenacion de variables:**

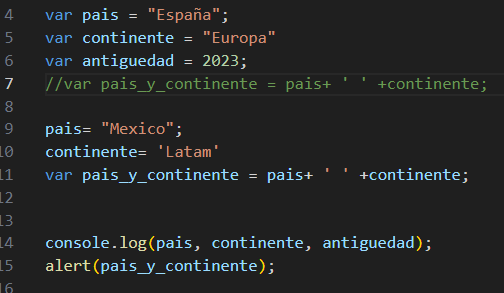
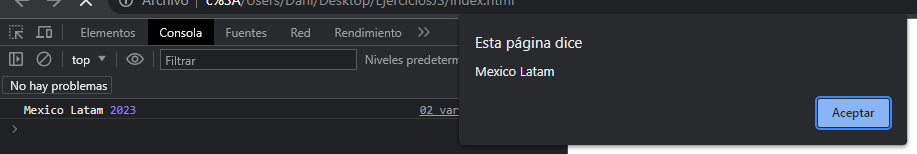
Podemos unir dos variables de esta forma con el símbolo + ‘ ‘ +





Las **variables** como su nombre indican **pueden cambiar** con líneas posteriores, es decir por ejemplo:

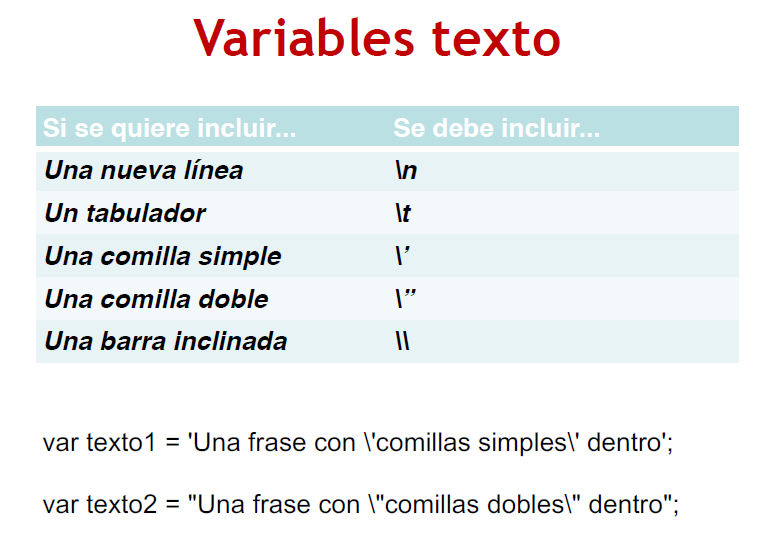
En este caso concreto, cambiamos las variables de país y continente a Mexico y Latam, pero al ser posterior el cambio a la variable pais\_y\_continente, esta concatenación sigue teniendo los valores antiguos por lo que la alerta de la concatenación sigue mostrando los valores antiguos mientras que el console.log mostrará los nuevos:

Si moviésemos la variable de pais\_y\_continente debajo de los nuevos valores de pais y continente, ahora la alerta también mostraría los nombres de las variables que hemos cambiado:

VARIABLES DE TEXTO:

Por ejemplo, no podríamos poner dobles comillas dentro de un string ya que nos daría error.

Var texto1 = ‘’ Una frase con ‘’comillas dobles’’ dentro’’ ; Esto ES ERROR

Var texto1 = “ Una frase con \”comillas dobles \” dentro; ESTO ESTÁ BIEN !

**ARRAYS:**

Son cadenas de strings (o valores) asignados en una variable o constante que luego se les asigna una numeración iniciando desde el 0.



**Pueden ser tan largas como queramos, siempre se asignaran entre comillas pues son strings.**

**Podemos hacer que uno de los meses concreto salga como alerta al entrar a la pagina.**

**Si por ejemplo quisiéramos que saliese Marzo, pondríamos :**

alert(meses[2]);

**Ponemos el 2 entre corchetes para decirle que es el valor 2 de nuestro array meses.**

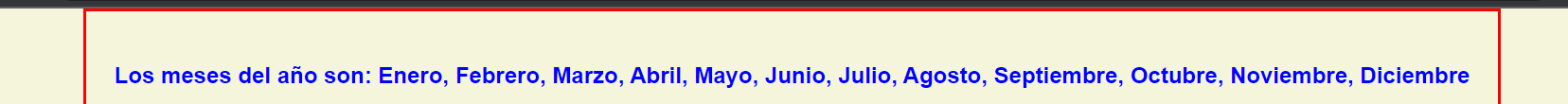
**Para mostrarlo en el HTML dentro de una caja especifica podemos utilizar el getElementByid:**

document.getElementById("meses").innerHTML = meses;

**Con esto le decimos al navegador que dentro del documento, coja un elemento ID con el ID=meses y le añada en el la caja con ese ID nuestra Array. Es decir, pondría dentro de esa caja nuestros meses del año.**

****

**El problema sería que entonces el contenido de esa caja desaparecería, para remediarlo bastaría con poner un signo + delante del = antes de dar nuestra array:**

document.getElementById("meses").innerHTML += meses;

**Aquí el contenido de nuestra caja era un párrafo con el texto ‘’ Los meses del año son:’’ y le hemos añadido detrás nuestra Array con el nombre meses.**

**ESTRUCTURA IF**

Es una estructura de condición, para que si esa condición se cumple, haga una cosa y si no se cumple, haga otra.

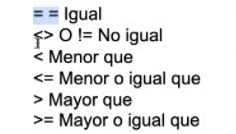


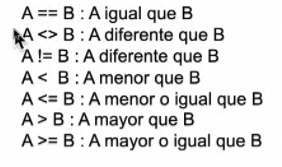
Cuando solo hay un =, es una asignación. Si queremos hacer una comparación se pone ==.

EXPRESIONES BOOLEANAS.

Se usan para determinar si un conjunto de una o más condiciones es verdadero o falso y el resultado de su evaluación es un valor de verdad.

2<3 ; A<3 ; A<B (son expresiones booleanas)

Expresiones booleanas:

**EJEMPLOS**

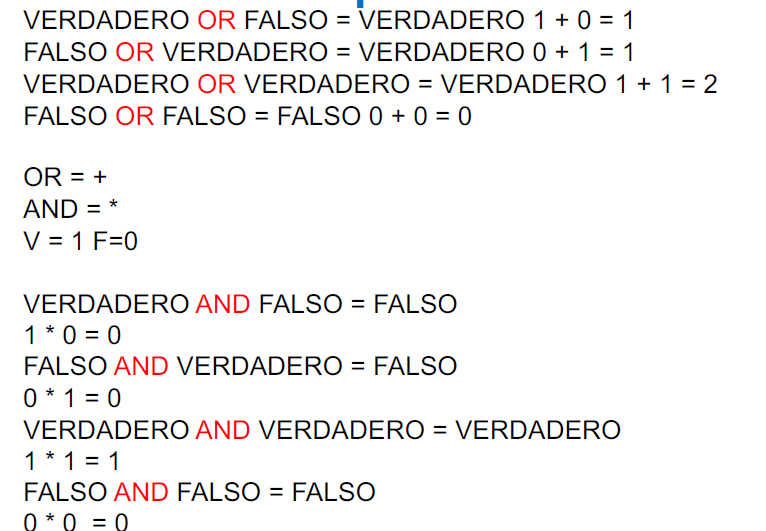
**Hay que tener cuidado con el Mayor o igual que: Si ponemos => no funcionara, se pone primero el mayor qué.**

**OPERADORES:**

NOT (NO) : Este operador produce un valor verdadero si su operando es falso.

AND: Este operador produce el valor verdadero si ambos operadores son verdadero ya que es un añadido (Y). Si cualquiera de los 2 es falso, dará falso

OR (O): Este operador realiza una operación O-inclusivo. El resultado es verdadero si cualquiera de los 2 o ambos son verdaderos, sino, es falso.

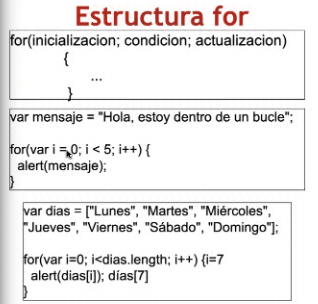


**Este es un ejemplo para entenderlo mejor, Si es verdadero es = 1 si es falso es = 0.**

**OR es una suma y AND es una multiplicación.**

**Solo hay que coger los valores asignados y cambiarlos por 0 o 1 si son verdaderos o falsos.**

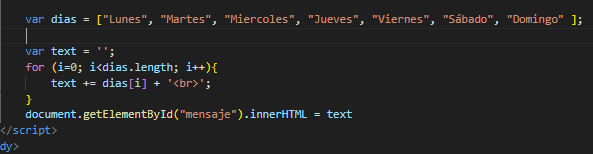
**Estructura FOR**

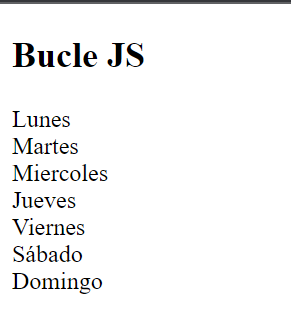
Tambien llamada Bucle.

Estructura: primero va el valor inicial (i=0), luego la condición (for) y luego la actualización (para cuando se acabe el bucle

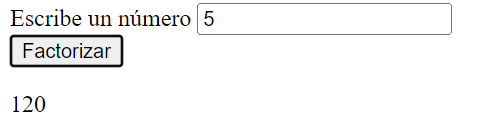
Asignamos la inicialización en 0. Decimos que se repita mientras i sea menor que 7 (numero de elementos del array días y que se sume de 1 en 1 (i++).

El i=7 y días[7] son en realidad comentarios

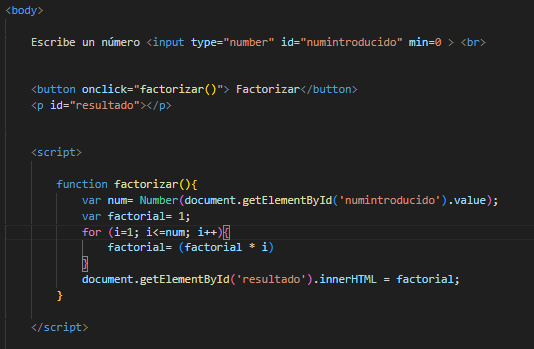




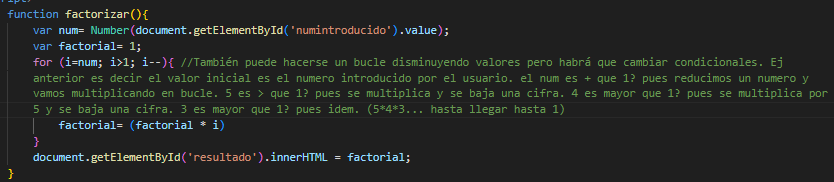
**Podemos complicarlo y hacer un formulario con un bucle. Por ejemplo un bucle en el que mediante un input number calcularemos el factorial del numero introducido por el usuario.**

****

Esto quedaría así en código:



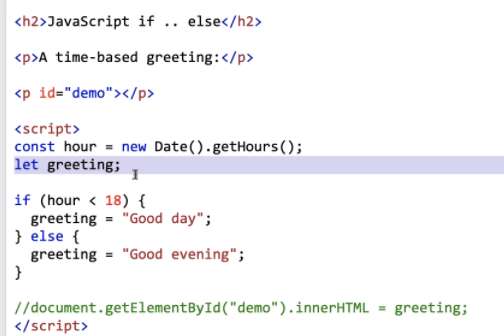
También se puede hacer de forma inversa y calcularlo de forma que vayamos restando 1 unidad en cada bucle al numero introducido por el usuario :



Este método en este caso parece mas correcto desde el punto de vista ‘’matemático’’ ya que una factorización de un numero es multiplicar un numero por sus anteriores .

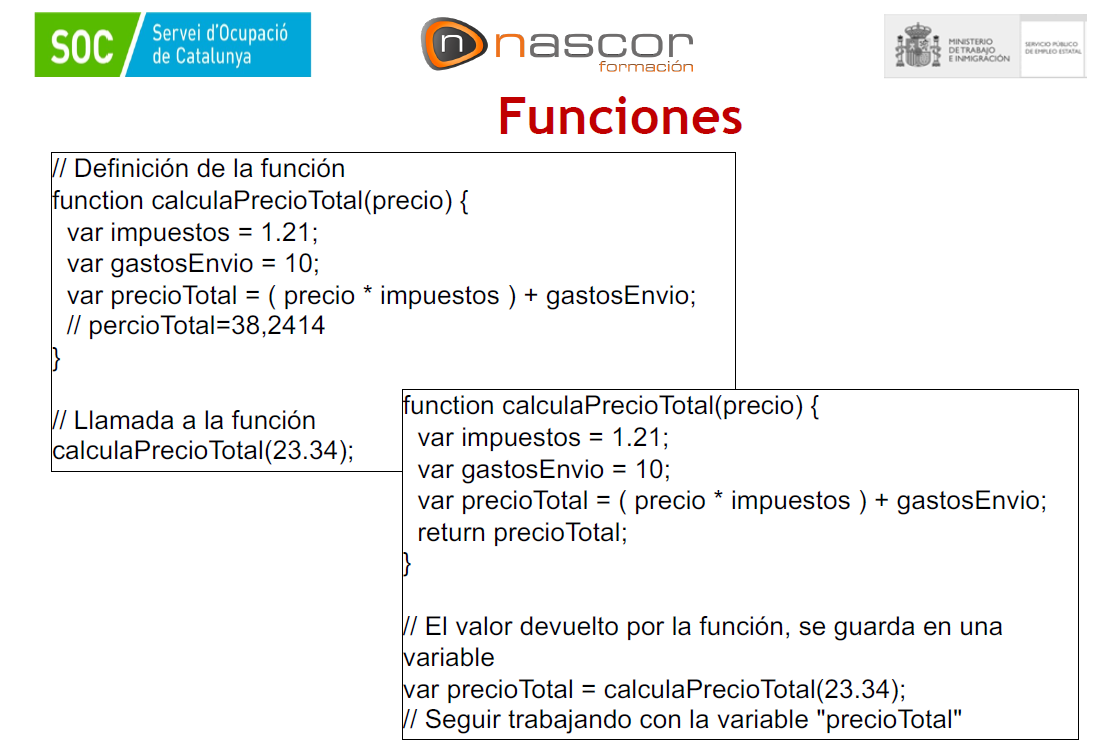
(num\*(num-1)\*(num-1-1)\*(num-1-1-1)…hasta que el numero sea 1).

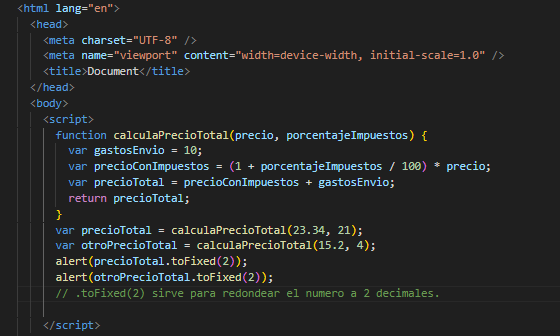
Para coger valores en JS:



El new date() genera el dia y el .getHours() coge la hora.

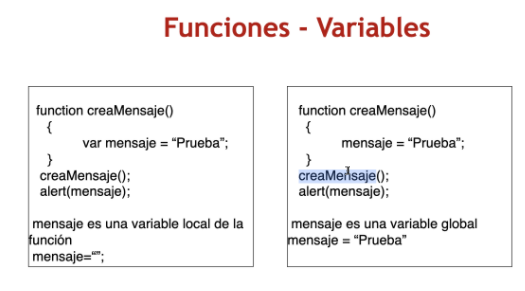
**FUNCIONES**

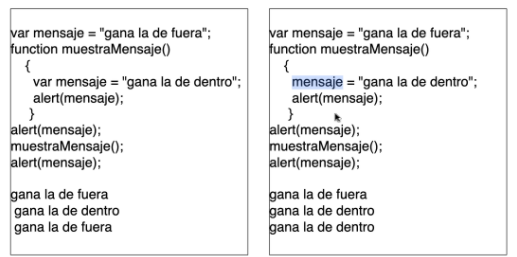




El ‘’return’’ es para que devuelva el valor de una variable (en este caso precioTotal.

El .toFixed(2) lo que hace es coger el resultado y redondearlo a 2 decimales!

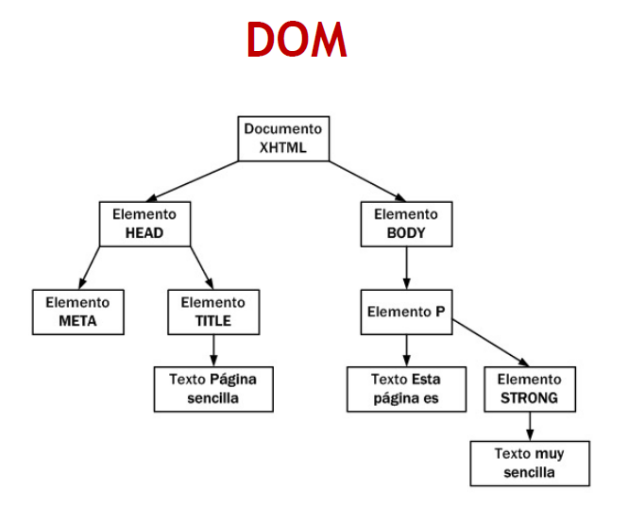
Para crear una variable global dentro de una función, crearemos la variable sin la palabra var.

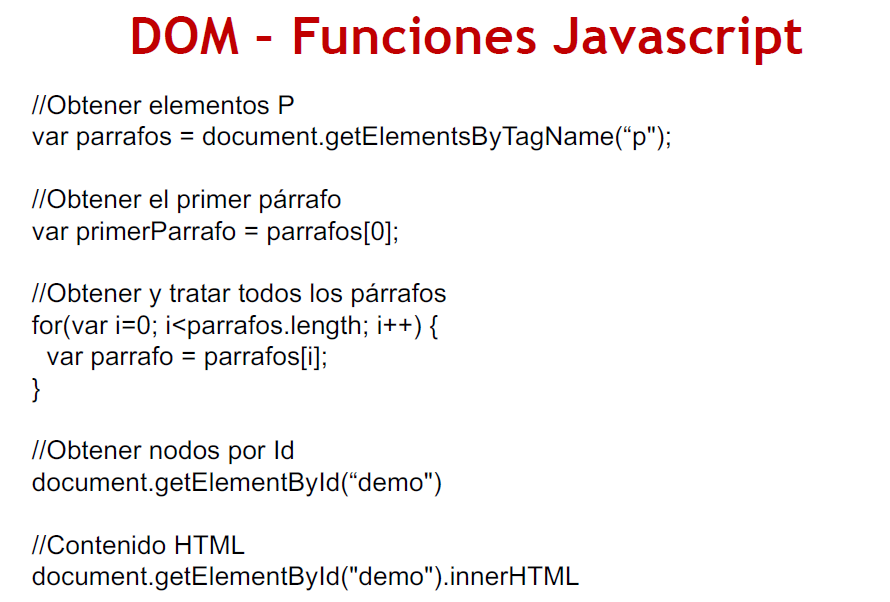


Aquí un ejemplo donde se puede ver mejor.



**DOM**

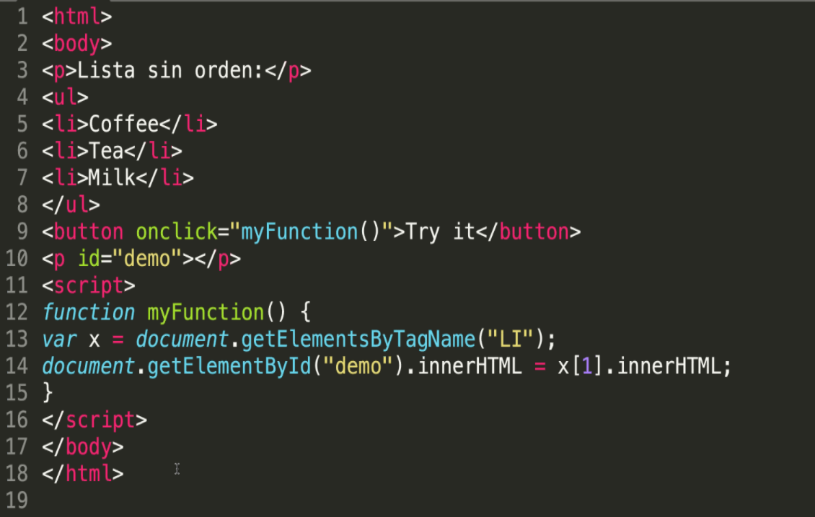
Elementos padres e hijos de todo el documento HTML.



El primer caso elementSSSS por el tag P, coge a todas las etiquetas P y las convierte en un array y podemos luego verlas (el primer P es [0]) para recorrer todos los párrafos al ser un array utilizaremos for…

En cambio al buscarlo por una id es sin la S pues el ID debe ser único.

El innerHTML como ya vimos es para añadir contenido al HTML.

En este caso dentro de la función creamos una variable X que coja los elementos LI del HTML y los convierte en un array. A continuación cogemos el elemento con id = demo que es la etiqueta p y le añadimos al html un texto con el texto del elemento 1 del array en este caso seria la LI con el texto Tea.

Formularios:

Puede haber mas de uno

Cada formulario se cumple cuando hay un action mediante el submit(normalmente).

Para acceder a los formularios : document.forms[0].elements[0] para acceder al primer elemento del formulario.

Para acceder al ultimo : document.forms[0].elements[document.forms[0].elements.length-1]

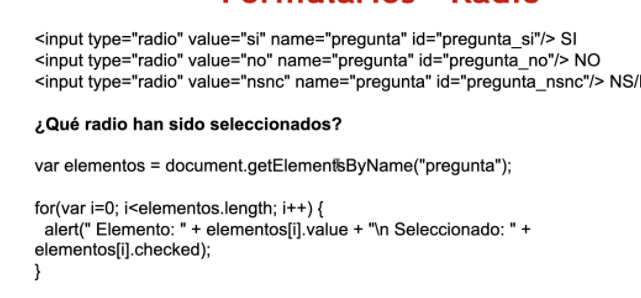
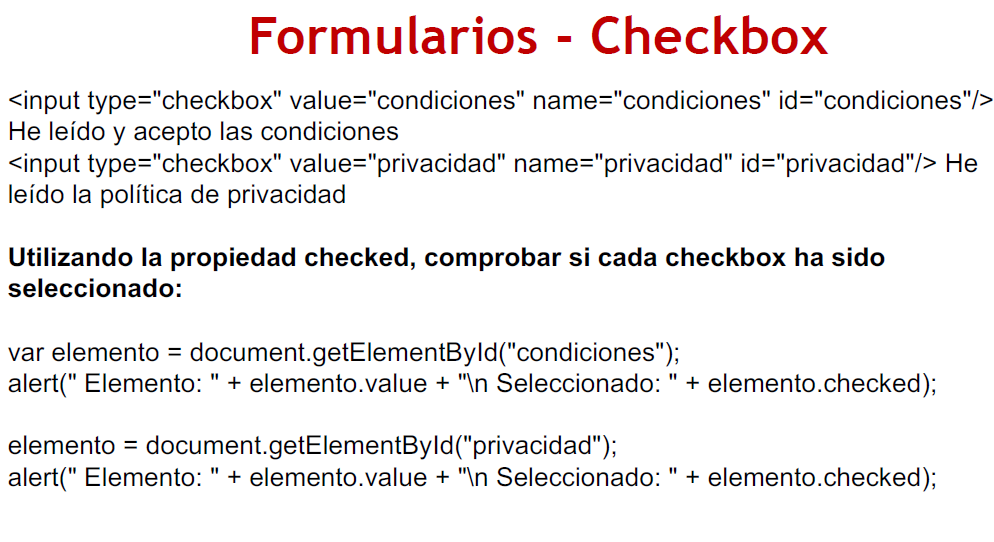
Eventos:  
onclick: Cuando pinchamos con el ratón sobre un elemento (cualquiera) pese a que se usa con botones.

Oninput : Se ejecuta cuando el usuario escribe algo en el input. Tambien aplica a desplegables.

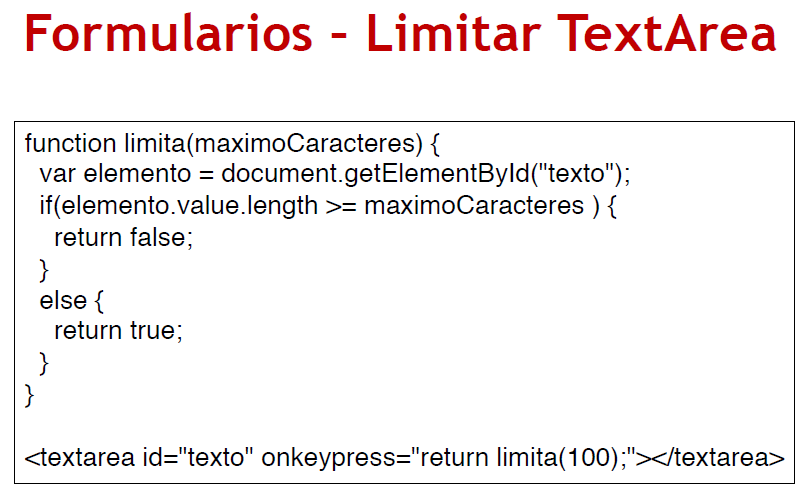
Onchange: Cuando el usuario cambia el valor de un elemento input y sale de este input. Tambien cuando el usuario señala una opción de una lista desplegable.

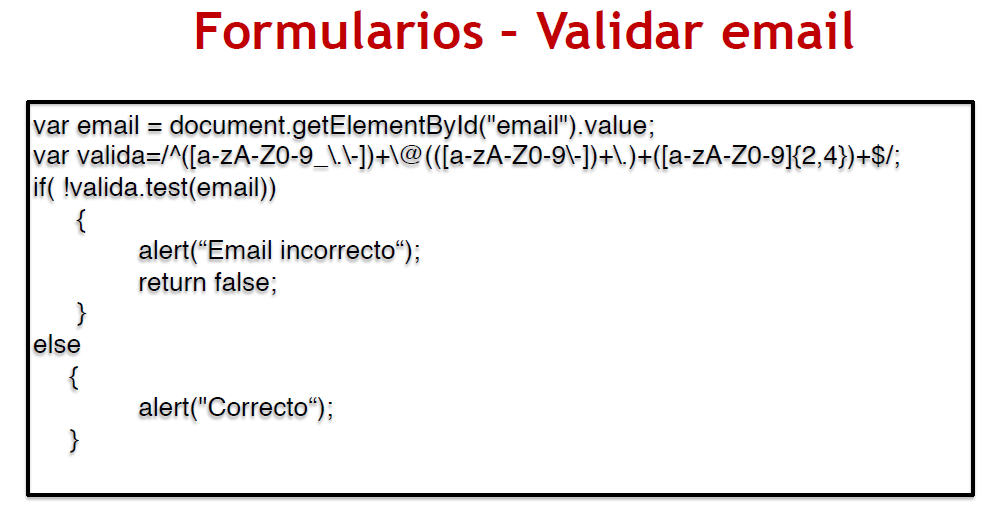
Onfocus: cuando el usuario selecciona un elemento del formulario.

Onblur: evento complementario de onfocus, ya que se produce cuando el usuario ha deseleccionado un elemento por haber seleccionado otro elemento del formulario.

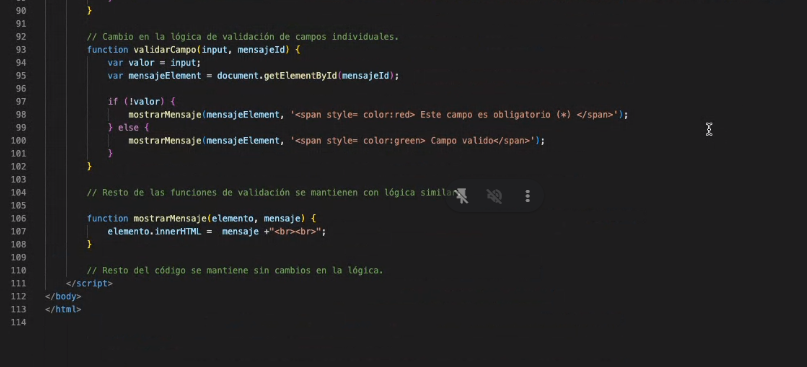
**Aquí averiguamos el botón que el user ha seleccionado.**

**Aquí para ver si el usuario ha aceptado los términos y condiciones**

****

**Con este limitamos un text área a 100 caracteres.**

**Validación de un correo electrónico.**

****