Javascript

Se ejecuta en local, no en un servidor, aporta respuesta inmediata, es interpretado no compilado. Agrega interactividad y responde al usuario. Proporciona efectos visuales.

Donde no llegues con HTML y CSS llegaras con JS.

Jquery: Librería de Javascript de código abierto, lenguaje simplificado de Javascript. Librería restada en todos los navegadores, permite agregar funcionalidades avanzadas sobre todo en animación.

Agregar codigo JS:

Dentro de HTML con etiqueta <script>, puede ir en el head o en el body. Suele estar mejor cuanto mas abajo por temas de ejecucion de codigo. Si no carga antes los elementos, puede dar error.

En archivos externos del tipo \*.js, aporta reutilización y cambios más flexibles.

Todos los comandos predefinidos se escriben asi = comando() , dentro de los () se puede escribir variables o introducir datos o frases o transformar números en texto con las strings.

**document.write()** sirve para escribir en el documento lo que quieras imprimir.

**alert()** es un popup con un mensaje y un boton aceptar.

**prompt()** es un popup para introducir datos.

**var()** es para indicar variables

**//** Comentarios lineales

/\* \*/ Comentarios en varias lineas

Estructuras basicas:

Al final de cada linea ; SIEMPRE.

Funciones predefinidas: vienen con el lenguaje y se pueden usar libremente. Ejemplos, el alert() , document.write(),...etc se diferencian SIEMPRE de otros elementos porque siempre llevan ().

Tipos de datos: Numeros con o sin decimales, Strings (textos con comillas (“” \ ‘’) y Booleanos (si o no, true or false)

Variables: Espacio de la memoria donde se almacenara un valor que puede cambiar:

var nombreVariable;

VARIABLES:

Reglas a la hora de nombrar variables:

Deben empezar por letra, $ o \_

Solo pueden contener letras, numeros, $ y \_

Son case sensitive, Distinguen mayusculas de minusculas.

No se pueden usar palabras reservadas (alert)

Recomendado que sean descriptivas a lo que afecten.(pantalla\_inicial, pantalla\_final,...)

Se recomienda utilizar el ingles.

Camel case -> miVariable (empieza en minus y segundo nombre en mayus);

Se pueden declarar muchas variables en la misma linea separada con comas:

**var pantalla = 1**, **idUsuario = 100**, **nombreUsuario=”Albert”;** Una variable que no sea numero, con comillas.

El símbolo **=** es ASIGNAR, no igual a.. cuando decimos **X = Y** siendo X 10 y Y 18, el resultado es 18, porque estamos diciendo que la variable X se le asigna el contenido de Y, sustituyendo el anterior contenido X.

El símbolo **==** es que el resultado sea similar o aproximado. Comparando.

El símbolo **===** es que el resultado, OBLIGATORIAMENTE, es el mismo que...

Si cometemos error en el código, el navegador no lo ejecuta pero no indica donde esta el fallo, tanto el VSC como la consola dell navegador nos avisan del error y en que linea se produce.

Operadores basicos:

* **suma +**
* **Resta -**
* **Multiplicacion \***

**/ Division /**

Las variables se deben de crear antes. Evitar poner - y \* porque se utilizan para hacer operaciones matemáticas. Para saber qué palabras se utiliza en JS para no usarlas en nombres, buscar algo así como “reserved words js”.

Para hacer saltos de línea “cutres”: **document.write(“<br/>”);**

El **prompt** es una ventana de recojida de datos mediante popup.

Matrices:

Son conexiones de datos.

Arrays - > Son colecciones de datos, en un array reservas espacio para almacenar datos, se puede entender como una gran variable con varios valores. Si una variable se puede ver como una carpeta , un array se debería ver como un archivador de carpetas. Muy útiles cuando no sabes el número de datos que vas a almacenar (carrito de compra )

rojo verde blanco violeta } arrayColores

Sintaxis:

Existen 2 formas de crear un array:

* var cestaCompra = [“llavero”, “mochila”, “zapatos”, “cordones”];
* var cestaCompra = newArray[“llavero”, “mochila”, “zapatos”, “cordones”];

Se pueden almacenar valores de distinto tipo.

cada valor va referenciado por un indice, empezando por el 0.

rojo verde blanco violeta } arrayColores

0 1 2 3

alert(arrayClientes[2]); -> “blanco”

Propiedades y métodos:

Javascript dispone de propiedades y métodos para los Array. Estos metodos y propiedades tienen funcionalidades determinadas.

**Lenght:** Cuenta la cantidad de elementos.

**Push:** Añade elementos al array en el ultimo sitio.

**Unshift:** Añade elementos al array en el primer sitio.

**Shift:** Elminia el primer elemento.

**Pop:** Elimina el ultimo elemento.

**Sort:** Combina aleatoriamente los elementos de strings, tanto textos como numeros.

Para ordenacion numerica: variable.sort(function(a,b){a-b});

Para eliminar elementos se utiliza **Shift** y **Pop**, para añadir **Unshift** y **Push**.

**Concat:** Union de dos arrays.

**Slice:** extrae elementos de la matrix (consultar) pero sin modificar la misma. Se utiliza para consultar datos para luego modificarlos y ponerlos otra vez.

CONDICIONALES:

Deciden el flujo del programa.

Se basa en condiciones que se deben de cumplir o no. Las aplicaciones parecen inteligentes. En los formularios los datos no se envian si los campos no se han rellenado correctamente.

* if: Si… es la traducion de la condicion, siempre tiene que ser o true o false.

Sintaxis if:

if(condicion) {

codigo a ejecutar cuando sea true

}

if(condicion) {

true

} else {

false

}

|  |  |
| --- | --- |
| == | Igual que |
| != | Diferente que |
| > | Mayor que |
| < | Menor que |
| >= | Mayor o igual que |
| <= | Menor o igual que |
| === | Estrictamente igual |
| !== | Estrictamente diferente |
| && | Y Logico |
| || | O Logico |

Métodos de parseo:

Cuando se comparan dos valores, deben de seguir una logica. En muchas ocasiones necesitamos convertir un string en number o viceversa. Existen diferentes metodos de conversion o parseo. parseInt(numero);

alert(parseInt(num1) + parseInt(num2));

El metodo typeof() es muy util para saber con que tipos de datos estamos trabajando.