

Javascript

Es un lenguaje de programación utilizado por los navegadores web.

¿Cómo usar JavaScript?

- Agregamos en nuestro archivo html (En <head> o <body>)

```
<script>
  // Esto es un comentario, dentro de la etiqueta script podremos escribir código JS
</script>
```

Mensajes

- Alertas: Mensajes de información que pueden ser visualizados en los navegadores web.

```
<script>
  window.alert('Esto es un mensaje');
</script>
```

- Alerta de Confirmación: Mensaje de información que muestra una pregunta y entrega un verdadero o falso.

```
<script>
  var respuesta = window.confirm('¿Texto de pregunta?');
  // respuesta será "si" (true) o "no" (false)
</script>
```

- Alerta de Escritura: Mensaje con campo de entrada de texto.

```
<script>
  var nombre = window.prompt('Ingresa su nombre');
</script>
```

Operadores

- Asignación de variables

```
name = 'Hola Mundo' // name tiene el valor de 'Hola Mundo'
```

- Operadores algebraicos
 - Suma [+]
 - Resta [-]
 - División [/]
 - Multipliación [*]
 - Resto [%]
 - AND [&&] - Y (Operador lógico, representa la multiplicación)
 - OR [||] - O (Operador lógico, representa la suma)
 - NOT [!] - Negación (Valor negado de algo)
 - Igualdad en valor [==]
 - Igualdad en valor y tipo de dato [===]
 - Distinto en valor [!=]
 - Distinto en valor y tipo [!==]
 - Menor [<]
 - Mayor [>]
 - Menor o Igual [<=]
 - Mayor o Igual [>=]

```
var num = 6 % 3; // num es igual a 0
```

- Tablas de operadores Booleanos:

AND = &&		Resultado
V	V	V
V	F	F
F	V	F
F	F	F

OR =		Resultado
V	V	V
V	F	V
F	V	V
F	F	F

Variables

- Las variables en javascript son espacios de memoria que contienen datos de distintos tipos y para poder usarlos debemos declararlas de la siguiente manera.

- Descripción:

```
// var -> palabra reservada para definir una variable
// var_name -> es el nombre que tendrá nuestra variable
// = -> es el operador para asignar el valor inicial
// value -> es el valor de la variable
// ; -> punto y coma cierra la expresión
var var_name = value ;
```

- Ejemplos

```
var x = 5; // numerico entero
var y = 8.0; // numerico decimal
var flag = true; // booleano sólo 2 valores true o false
var name = 'Pedro'; // String (cadena de caracteres)
```

Estructuras de control

- Condición:** Es una expresión de la cual se puede determinar su verdad, generalmente son comparaciones por ejemplo:
 - `3 < 5 => true`
 - `5 < 3 => false`
- if:** Es una estructura de control que evalúa el valor de una condición separando 2 bloques de código (en caso de existir un else).

```
if (condition) {
    // Condition true
} else {
    // Condition false
}
```

- switch:** Es una estructura de control que evalúa el valor de una condición pero presenta varios casos.

```
switch (option) {
    case a:
        // option === a
        break;
    case b:
        // option === b
        break;
    case c:
        // option === c
        break;
    default:
        // option !== a && option !== b && option !== c
}
```

Ciclos

Son bloques de código que se repetirán hasta que la condición de termino deje de cumplirse.

- do-while:** Ciclo que ingresa al bloque de código y al final del bloque evalúa la condición de término.

```
do {
    // code here
} while (condition)
```

- while:** Ciclo que evalúa la condición de término al comienzo.

```
while (condition) {
    // code here
}
```

- for:** Ciclo que inicia una variable (contador), evalúa condición de término, incrementador (para contador).

```
for (var index = 0; condition; index++) {
    // code here
}
```

Arreglos

- Son listas o colecciones de datos que pueden ser contenidos dentro de una variable como lo muestra el siguiente ejemplo:

```
var nombres = ['Pedro', 'Juan', 'Diego'];
```

- Poseen **índices** que comienzan del cero.

```
nombres[0] // Estamos accediendo al primer valor del arreglo 'Pedro'
```

- El **largo** de un arreglo se obtiene con la propiedad **length** lo cual siempre será un número entero

```
nombres.length // Obtenemos un 3
```

- Para **añadir** elementos a un arreglo usamos el método **push**

```
nombres.push('Karen') // Ingresamos el nombre 'Karen' en nuestro arreglo
```

- Para **eliminar** elementos a un arreglo usamos el método **splice** y recibe por parámetro el índice del elemento a borrar

```
nombres.splice(1) // Borramos el nombre 'Juan'
```

- Para **eliminar el último elemento** a un arreglo usamos el método **pop** el método retorna el elemento y luego lo elimina.

```
var ultimoNombre = nombres.pop() // Extrae 'Karen' y luego lo Borra de la lista
```

- Para **ordenar** el arreglo usamos el método **sort** y nos entregará el arreglo ordenado

```
nombres.sort() // Ordena el arreglo según el tipo de dato
```

Funciones

Las funciones en general son bloques de código que estarán dentro de la declaración de la función, estas tienen un nombre (el cual usaremos para invocar el bloque de código) y parámetros. Los parámetros son variables que podremos utilizar en nuestro bloque de código.

```
function function_name(params1, params2....paramsN) {  
  // code here  
}  
// function es la palabra reservada para crear una nueva función  
// function_name es el espacio donde escribiremos el nombre de nuestra función  
// () los paréntesis son el espacio donde escribiremos los parametros en caso de tener  
// { } las llaves nos indican el inicio y fin de nuestro bloque de código
```

- Ejemplos de funciones:

- Función sin parámetros:

```
function mostrarAlerta() {  
  window.alert('Esto es un mensaje');  
}  
// para utilizar la función la invocamos por su nombre  
mostrarAlerta();
```

- Función con parámetros:

```
function mostrarAlerta(msg) {  
  window.alert(msg);  
}  
var mensaje = 'Esto es un mensaje';  
// para utilizar la función la invocamos por su nombre y escribimos parametros en paréntesis  
mostrarAlerta(mensaje);
```


- Función con retorno:

```
function sumar(x, y) {  
  return x + y; // utilizamos la palabra reservada return para entregar un valor  
}  
var numero = sumar(2, 3); // el valor será guardado en la variable numero  
window.alert('La suma es : ' + numero); // mostramos el valor en el alert concatenado el texto
```

Objetos JavaScript

Los objetos (`{property: value }`) también son contenidos en variables pero estas pueden contener más de un valor, al igual que los objetos reales tienen características (Variables) y acciones (Métodos) de la siguiente forma:

```
var objectName = {  
  propertyName: value,  
  methodName: function () {  
    // function code here  
  }  
};
```

Object	Properties	Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()

- Acceso a las propiedades de un objeto

```
// Forma genérica
objectName.propertyName;
objectName['propertyName'];
```

- Ejemplo:

```
var person = {
  name: 'Pedro'
};
person.name; // Para acceder con nombre de propiedad
person['name']; // Forma equivalente
```