Uso de sentencias if

La sentencia if permite separar la ejecución del código en función de la evaluación de una comparación. Las siguientes líneas de código muestran la sintaxis. Los operadores condicionales se escriben entre paréntesis, y el código a ejecutar, si el condicional se evalúa como true, se indica entre corchetes ({}):

```
if(x==5){
  hacer_algo();
}
```

Además de ejecutar el código que se encuentra dentro del bloque **if** de instrucciones, se puede especificar un bloque **else** que se ejecute solo si la condición es false.

```
if(x==5){
  hacer_algo();
} else {
  hacer_algo_else();
}
```

Uso de sentencias if

También es posible encadenar ó anidar sentencias if. Para hacer esto, se agrega una declaración condicional junto con una declaración else.

```
if(x<5){
  hacer_algo();
} else if(x<10) {
  hacer_algo_else();
} else {
  hacer_algo();
}
```

Uso de sentencias switch

Otro tipo de lógica condicional es la sentencia switch. Permite evaluar una expresión una vez y luego, según el valor, ejecutar una de las muchas secciones diferentes de código.

La instrucción switch evalúa la expresión por completo y obtiene un valor. Ese valor puede ser una cadena, un número, un valor booleano o incluso un objeto. Luego, la expresión switch se compara con cada valor especificado por la instrucción case. Si el valor coincide, se ejecuta el código de la sentencia. Si ningún valor coincide, se ejecuta el código predeterminado.

```
switch(expresión){
  case value1:
    <código a ejecutar>
    break;
  case value2:
    <código a ejecutar>
    break;
  default:
    <código a ejecutar si no se cumplen value1 o value2>
}
```

Implementación de bucles

El bucle es un medio que permite ejecutar el mismo segmento de código varias veces. Esto es extremadamente útil cuando se necesitan realizar las mismas tareas en una matriz o conjunto de objetos. JavaScript proporciona funcionalidad para realizar bucles for y while

Bucle while

Es el bucle más básico en JavaScript. Un ciclo while prueba una expresión y continúa ejecutando el código contenido entre {} corchetes hasta que la expresión se evalúe como false.
Por ejemplo, el siguiente bucle while se ejecuta hasta que el valor de i es igual a 5:

```
var i = 1;
while (i<5){
  print("Iteración" + i + "\n");
  i++;
}
```

Salida ->

```
Iteración 1
Iteración 2
Iteración 3
Iteración 4
```

Bucle do / while

Se usa cuando se desea ejecutar el código contenido dentro del ciclo, al menos, una vez. Es decir, la expresión no se puede probar hasta que el código se haya ejecutado al menos una vez.
Por ejemplo, el siguiente bucle do while se ejecuta mientras que el valor de dia sea distinto a miercoles:

```
var dias = ["lunes", "martes", "miércoles", "jueves", "viernes"];
var i=0;
do{
    var dia=dias[i];
    print("Es " + dia + "\n");
    i++
} while (dia != "miércoles");
```

Salida ->

```
Es lunes

Es martes

Es miércoles
```

Bucle for

Permite ejecutar código una cantidad específica de veces mediante una declaración for que combina tres declaraciones en un solo bloque de ejecución utilizando la siguiente sintaxis:

```
for (asignación; condición; actualización;){
  código a ejecutar;
}
```

La sentencia **for** utiliza esas tres declaraciones, al ejecutar el ciclo.

Bucle for

- **Asignación**: se ejecuta una única vez, antes de que comience el bucle. De esta manera se inicializan las variables utilizadas en el ciclo como condicionales.
- **Condición**: se evalúa antes de cada iteración del ciclo. Si la expresión se evalúa como true, se ejecuta el ciclo; de lo contrario, for finaliza la ejecución del bucle.
- **Actualización**: Valor de incremento de la variable de iteración. Su uso típico es incrementar un contador usado en la instrucción 2.

```
for (var i=1; i<=3; i++){
    print("El valor de i es ", i, "\n");
}
```

Salida ->

```
El valor de i es  1

El valor de i es  2

El valor de i es  3
```

Bucle for in

Otro tipo de bucle for es el bucle for/ in, que se ejecuta en cualquier tipo de datos que se pueda iterar. En su mayor parte, se usa el bucle for/ in en matrices y objetos.

```
var dias = ["lunes", "martes", "miércoles",
            "jueves", "viernes"];
for (var idx in dias){
  print("Es " + dias[idx] + "\n");
}
```

Salida ->

```
Es lunes
Es martes
Es miércoles
Es jueves
Es viernes
```

Bucle con método foreach

Este método entra dentro del grupo de Iterables sin devolver una nueva matriz, lo que hace el forEach es ejecutar una función por cada elemento del arreglo. En cada iteración se tendrá acceso a 3 variables: valor (del elemento), índice (del elemento) y arreglo (que se está recorriendo).

```
var dias = ["lunes", "martes", "miércoles", "jueves", "viernes"];

dias.forEach(function(element) {
    print("Día : " + element);
}
);
```

Salida ->

```
Día : lunes
Día : martes
Día : miércoles
Día : jueves
Día : viernes
```