

JS | Conditionals and Loops

Objetivos:

- ✓ Comprender qué son las declaraciones
- ✓ Entender por qué son necesarios los condicionales
- ✓ Utilizar *if*, *else*
- ✓ Comprender y usar *switch*
- ✓ Identificar cuándo es mejor usar *switch*, *if .. else*
- ✓ Entender por qué son necesarios los bucles y las iteraciones
- ✓ Comprender y usar la declaración *while* y *for*

Declaraciones - resumen general

Las declaraciones son fragmentos de código que realizan alguna acción.

Por ejemplo:

- if...else
- switch
- do...while
- for
- forEach
- break
- while

Declaraciones condicionales

if ejecuta una declaración si la condición especificada es verdadera.

Si la condición es falsa, se puede ejecutar otra declaración.

```
if (condition) {  
    // code to execute if the condition is true  
} else {  
    // code to execute if the condition is false  
}
```


Si tenemos más de una condición para verificar:

```
if (condition1) {  
    // code to execute if condition1 is true  
} else if (condition2) {  
    // code to execute if condition2 is true  
} else {  
    // code to execute if condition1 and condition2 are false  
}
```

Condiciones anidadas

```
if (condition) {  
    if (nestedCondition) {  
        // The code will be executed if  
        // condition === true && nestedCondition === true  
    } else {  
        // The code will be executed if  
        // condition === true && nestedCondition === false  
    }  
} else {  
    // The code will be executed if  
    // condition === false  
}
```


Hora de practicar - Custom Hello, world!

Preguntémosle al usuario en qué idioma quiere ver el mensaje. Debe tener, al menos, tres idiomas diferentes.

- Si el usuario quiere el mensaje en español, debe mostrar en la consola "Hola, mundo!".
- Si el usuario quiere el mensaje en francés, debe mostrar en la consola "Bonjour tout le monde".
- Finalmente, si no tenemos el idioma indicado, mostraremos "¡Hello, world!".

Use las declaraciones `if ... else` y `else if` que acabamos de aprender

Si tenemos muchas condiciones,
aunque funcione bien, no es lo más
correcto. Lo correcto, en estos casos,
es utilizar *switch*

switch

```
switch (expression) {  
  case value1:  
    // executed code when the expression === value1  
    break;  
  case value2:  
    // executed code when the expression === value2  
    break;  
  case value3:  
    // executed code when the expression === value3  
    break;  
  default:  
    // executed code when none of the values match the expression  
}
```

Comprueba la condición y ejecuta declaraciones asociadas con ese caso

break

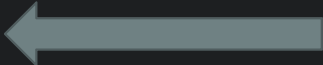
La declaración *break* termina el caso actual.

```
switch (expresión a evaluar){  
    case valor1 :  
        instrucción a ejecutar;  
        break;  
    case valor2 :  
        instrucción a ejecutar;  
        break;  
    case valor3 :  
        instrucción a ejecutar;  
        break;  
    default :  
        instrucción si no hay ninguna coincidencia;
```


Podemos configurar el mismo código para que se ejecute para un caso diferente.

```
const name = prompt("Favorite Game of Thrones main character:");
let house = "";

switch (name) {
  case "Khal Drogo":
    house = "Dothraki Horselord";
    break;
  case "Daenerys":
    house = "Targaryen";
    break;
  case "Jon Snow":
  case "Sansa":
  case "Arya":
    house = "Stark";
    break;
  default:
    house = "other";
}
```



Hora de practicar

Vamos a mejorar el saludo personalizado "Hello, World!" utilizando switch en vez de if..else
Agrega algunos idiomas más en tu programa.

Bucles e iteraciones

Se utilizan para ejecutar tareas repetitivas.

Por ejemplo, si queremos imprimir números del 1 al 100, no los escribiremos todos. Usaremos *while* o la declaración *for* para hacer eso. Ambos nos ofrecen una manera rápida y fácil de hacer algo repetidamente.

while statement

```
while (condition) {  
    // code to be executed while the condition is true  
}
```

Crea un bucle que ejecuta un código especificado siempre que la condición evalúe verdadero

do while statement

crea un ciclo que ejecuta un bloque de código una vez, antes de verificar si la condición es verdadera, y luego repetirá el ciclo siempre que la condición sea verdadera.

```
do {  
    // block of code to be executed  
} while (condition);
```

Práctica

- Iteración 1: crear un script que cuente de 0 a 30. Resultado esperado:

1
2
...
29
30

- Iteración 2: cambiar el script anterior para escribir "diez" cuando el valor de i sea 10 y "veinte" cuando el valor sea 20. Resultado esperado:

1
...
9
diez
11
...
19
veinte
21
...
30

Resolución

- Iteración 1:

```
let i = 1;

while (i <= 30) {
  console.log (i);
  i++;
}
```

- Iteración 2:

```
let i = 1;

while (i <= 30) {
  switch(i) {
    case 10:
      console.log("diez");
      break;
    case 20:
      console.log("veinte");
      break;
    default:
      console.log(i);
  }
  i++;
}
```

for statement

Crea un bucle con tres valores diferentes separados por punto y coma: inicialización, condición y expresión final.

```
for (initialization;  
condition; finalExpression) {  
    // code to execute  
}
```


Break statements

Termina la declaración switch o el bucle actual (for, for...in, while, do...while)

Cuando se usa en un bucle, lo corta y continúa ejecutando el código después del bucle (si lo hay).

```
var text = "";

for (var i = 1; i < 5; i++) {
  if (i === 3) {
    break;
  }
  text += "The number is " + i + '\n';
}

console.log(text);
```

Qué va a imprimir?

Continue Statement

Corta una iteración (en el ciclo) si ocurre una condición específica, y continúa con la siguiente iteración en el ciclo.

```
var text = "";

for (var i = 1; i < 5; i++) {
    if (i === 3) {
        continue;
    }
    text += "The number is " + i;
}

console.log(text);
//'The number is 1 The number is 2 The number is 4'
```


Práctica

Escribamos un ciclo que iterará de 0 a 20. Para cada iteración, verificará si el número actual es par o impar, e imprimirá eso en la pantalla. Necesitará el operador % para hacer el ejercicio. Resultado esperado:

1 es impar

2 es par

...

19 es impar

20 es par

Posible solución:

```
for (let i = 1; i < 21; i++) {  
  if (i % 2 === 0) {  
    console.log(i + " es par");  
  } else {  
    console.log(i + " es impar");  
  }  
}
```


Resumiendo

- ✓ Usamos valores booleanos (verdadero o falso) en las declaraciones condicionales (*if..else* o *switch*), y dependiendo de la condición sabemos qué bloque de código se ejecutará.
- ✓ Si estás evaluando una condición y tiene demasiadas opciones para verificar, la mejor solución es cambiar *if..else* por *switch*.
- ✓ Finalmente, aprendimos a iterar con *while* y *for*.