

# Desarrollo Backend con NodeJS

Condicionales y bucles

# Condicionales

## Condicionales

En todo lenguaje de programación el código necesita realizar condiciones y ejecutar acciones dependiendo de distintas entradas.

- Ejemplo si un contador llega a 0 que muestre un mensaje en la pantalla
- Si las vidas de un jugador llegan a 0 que le notifique en la pantalla que perdió

Todo esto se hace mediante condicionales que en el caso de Javascript son **if**, **else** y **else if**

## if ... else



```
if (condition) {  
    // si se cumple la condición se ejecuta este código  
} else {  
    // si la condición no se cumple se ejecuta este código  
}
```

## if ... else if ... else



```
if (condition1) {  
    // si se cumple la primer condición se ejecuta este código  
} else if (condition2) {  
    // si se cumple la segunda condición se ejecuta este código  
} else {  
    // si no se cumple ninguna condición se ejecuta este código  
}
```

# if ... else if ... else



```
if (condition1) {  
    // si se cumple la primera condición se ejecuta este código  
} else if (condition2) {  
    // si se cumple la segunda condición se ejecuta este código  
} else if (condition3) {  
    // si se cumple la tercera condición se ejecuta este código  
} else if (condition4) {  
    // si se cumple la cuarta condición se ejecuta este código  
} else if (condition5) {  
    // si se cumple la quinta condición se ejecuta este código  
} else {  
    // si no se cumple ninguna condición se ejecuta este código  
}
```

# Operadores de comparación

## Operadores de comparación

Los operadores de comparación son usados para probar las condiciones dentro de nuestra declaración condicional

- **operadores == y !=**
  - ◆ comprueba que dos valores sean iguales o distintos
- **operadores === y !==** — prueba si dos valores son iguales o distintos pero además chequea el **tipo**
- **< y >**
  - ◆ prueban si un valor es menor o mayor que otro
- **<= y >=** prueba si un valor es menor e igual o mayor e igual que otro.





```
console.log( 7 > 9 ) // false
console.log( 7 < 9 ) // true
console.log( "CourseIt" === "CourseIt" ) // true
console.log( "Frontend" !== "Backend" ) // true
console.log( 1 == "1" ) // true
console.log( 1 === "1" ) // false
```

# Operadores lógicos

## Operadores lógicos

Los operadores lógicos sirven para relacionar condiciones

### → Operador && (AND)

- ◆ Si ambas condiciones se cumplen el resultado será verdadero, sino sera falso


### → Operador || (OR)

- ◆ Si al menos una de las condiciones se cumple el resultado será verdadero, si ninguna se cumple será falso



```
const name = "CourseIt"
const address = "25 de Mayo 564"

// queremos saber si estamos en CourseIt
if (name === "CourseIt" && address === "25 de Mayo 564") {
    // VERDADERO, estamos en CourseIt
} else {
    // FALSO, estamos en otro lugar
}
```



```
const drinks = ["café", "te", "jugo de naranja", "agua"]

// queremos diferenciar bebidas calientes y frías
if ( drink[0] === "café" || drink[0] === "te" ) {
  // VERDADERO, nuestra bebida cumple con al menos una condición
} else {
  // FALSO, nuestra bebida no cumple la condición
}

// queremos diferenciar bebidas calientes y frías
if ( drink[2] === "café" || drink[2] === "te" ) {
  // FALSO, nuestra bebida es jugo y no cumple la condición
} else {
  // En este caso se ejecuta este código
}
```

# Bucles

## Bucles

Un bucle es la manera en que le vamos a decir a nuestro código que ejecute una acción múltiples veces.


→ Ejemplos

- ◆ Hacer un contador de 0 a 10
- ◆ Recorrer un array para mostrar su contenido.

# For

Un bucle for se repite hasta que la condición especificada se evalúa como false. Una sentencia for se escribe de la siguiente manera:

**for (inicio ; condición ; incremento)**



```
for (let i = 0; i < 10; i++ ) {  
  console.log( i )  
  // 0, 1, 2, 3, 4, 5, 6, 7, 8, 9  
}
```



# while

Una sentencia while ejecuta sus sentencias mientras la condición sea evaluada como verdadera

**while (condicion) { sentencia }**



```
let i = -5
```

```
while (i < 10) {  
  console.log(i)  
  i++  
}
```

```
// -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
```

## for ... in

La sentencia **for...in** itera una variable especificada sobre todas las propiedades enumerables de un objeto. Se escribe de la siguiente manera:

```
for ( variable in objeto ) { sentencia }
```



```
const user = {  
  name: 'Bel',  
  lastname: 'Rey',  
  age: 33  
}  
  
for (property in user) {  
  console.log('clave:', property)  
  console.log('valor:', user[property])  
}  
  
// 'clave': name  
// 'valor': 'Bel'  
// 'clave': lastname  
// 'valor': 'Rey'  
// 'clave': age  
// 'valor': 33
```