

Condicionales

DEV.F
DESARROLLAMOS(PERSONAS);

dev

Condicionales



Cuando en programación hablamos de condicionales, hablamos de una estructura sintáctica que sirve para tomar una decisión a partir de una condición.

```
Si <condición>  
entonces <operación>
```

Condicionales en la vida cotidiana

En la vida diaria, actuamos de acuerdo a la evaluación de condiciones, de manera mucho más frecuente de lo que en realidad creemos:



Si el semáforo está en verde, cruzar la calle. Si no, esperar a que el semáforo se ponga en verde.



¿Comeré una galleta? o
¿Comere 2 galletas?

Operadores relacionales

DEV.F
DESARROLLAMOS(PERSONAS);

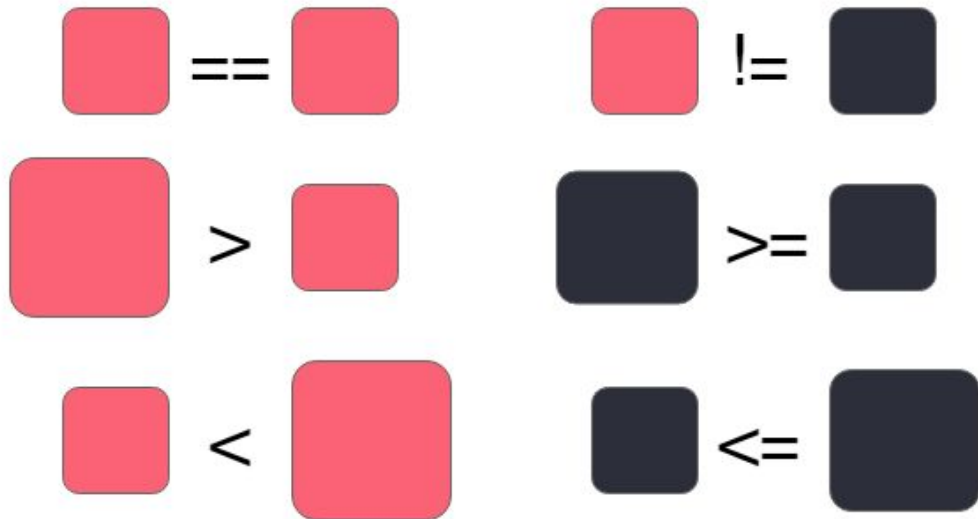
dev

Operadores relacionales

Los operadores relacionales son símbolos que se usan para comparar dos valores.

Si el resultado de la comparación es correcto la expresión considerada es verdadera, en caso contrario es falsa.

“Verde” != “Verde” => false



<

>

<=

>=

==

===

!=

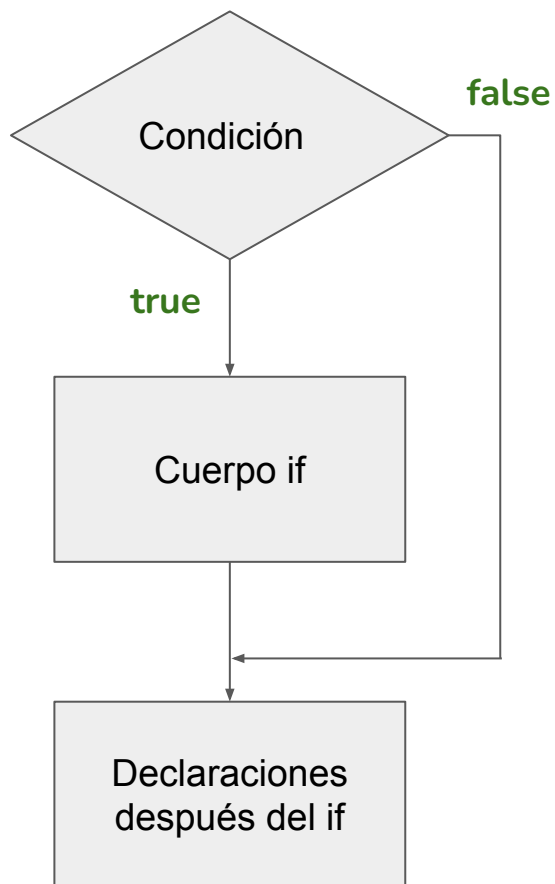
Tablas de Operadores de Comparación Comunes

| Simbolo | Significado | Ejemplo | Resultado |
|---------|--------------|--------------|-----------|
| == | Igual que | true == true | true |
| != | Distinto que | 10 != 10 | false |
| < | Menor que | 10 < 4 | false |

| Simbolo | Significado | Ejemplo | Resultado |
|---------|-------------------|---------|-----------|
| > | Mayor que | 3 > 1 | true |
| <= | Menor o igual que | 10 <= 9 | false |
| >= | Mayor o igual que | 4 >= 10 | false |

Tomando decisiones: Declaraciones If / else





Declaraciones If / else

if ejecuta una sentencia si una condición especificada es evaluada como verdadera. Si la condición es evaluada como falsa, otra sentencia puede ser ejecutada (**if else** o **else**).

Sintaxis if básica

Puedo ejecutar un código en el caso de que se cumpla una condición, solamente utilizando if:

```
if (condición) sentencia1
```

If cuando ejecuta una sola línea de código (sentencia)

```
if (condición) {  
    sentencia1;  
    sentencia2;  
    sentencia3;  
}
```

Si el if ejecuta múltiples sentencias, debe llevar {}.

Sintaxis if-else básica

Con la estructura if else, puedo ejecutar un código si se cumple la condición o en caso contrario ejecutar otro en su lugar:

```
if (condición) {  
    //código a ejecutar si la condición es verdadera  
} else {  
    //ejecuta este otro código si la condición es falsa  
}
```

Sintaxis if-else básica: Ejemplo

Con la estructura if else, puedo ejecutar un código si se cumple la condición o en caso contrario ejecutar otro en su lugar:

```
> var edad = 20;
```

```
< undefined
```

```
> if (edad >= 18) console.log("Eres mayor de edad")  
   else console.log("Eres menor de edad");
```

```
Eres mayor de edad
```

Sintaxis if - else if

Con la estructura **if - else if**, puedo ejecutar un código si se cumple con la condición especificada en cada caso:

```
if (condición1) {  
    //código a ejecutar si la condición1 es verdadera  
} else if (condición2) {  
    //ejecuta este otro código si la condición2 es verdadera  
} else if (condición3) {  
    //ejecuta este otro código si la condición3 es verdadera  
} else if (condición4) {  
    //ejecuta este otro código si la condición4 es verdadera  
}
```

Sintaxis if - else if: Ejemplo

Con la estructura **if - else if**, puedo ejecutar un código si se cumple con la condición especificada en cada caso:

```
> var semaforo = "amarillo";  
  if (semaforo == "verde") {  
    console.log("Avanza")  
  } else if (semaforo == "amarillo") {  
    console.log("Comienza a frenar")  
  } else if (semaforo == "rojo") {  
    console.log("Detente")  
  }
```

Comienza a frenar

Sintaxis if - else if - else

Con la estructura **if - else if**, puedo ejecutar un código si se cumple con la condición especificada en cada caso y al final con **else** ejecutar un código si no se cumplio ninguna condición anterior.

```
if (condición1) {  
    //código a ejecutar si la condición1 es verdadera  
} else if (condición2) {  
    //ejecuta este otro código si la condición2 es verdadera  
} else if (condición3) {  
    //ejecuta este otro código si la condición3 es verdadera  
} else {  
    //ejecuta este otro código si ninguna condición se cumple  
}
```

Sintaxis if - else if - else: Ejemplo

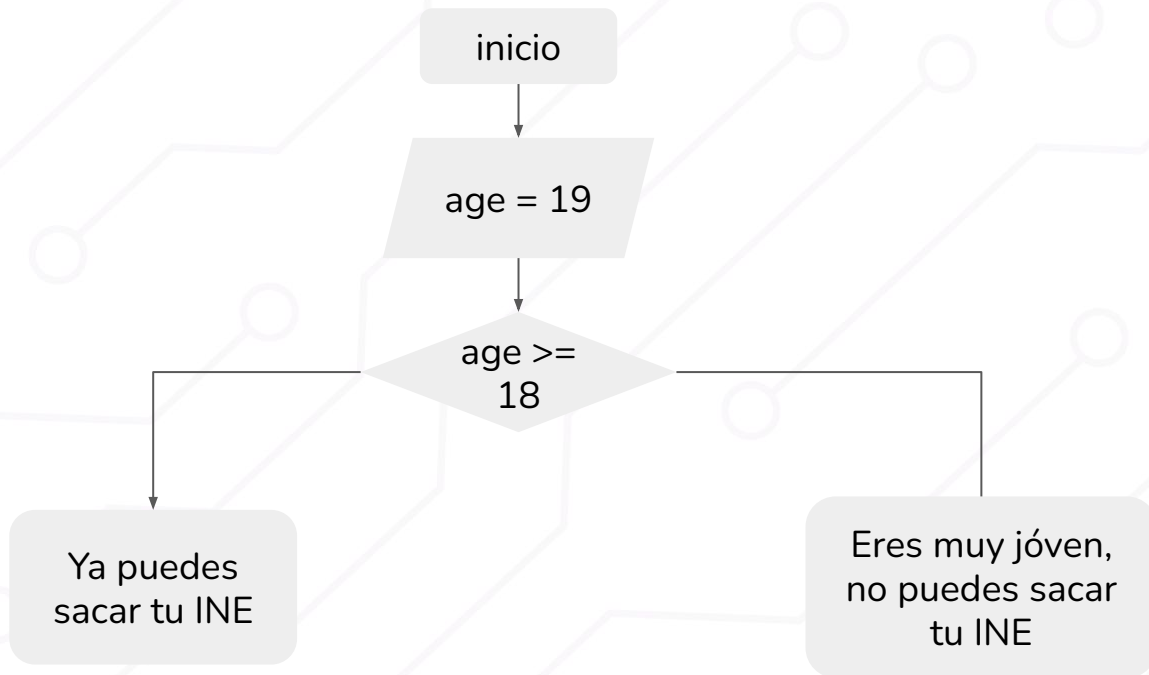
Con la estructura **if - else if**, puedo ejecutar un código si se cumple con la condición especificada en cada caso y al final con **else** ejecutar un código si no se cumplió ninguna condición anterior.

```
> var semaforo = "morado";  
  if (semaforo == "verde") {  
    console.log("Avanza")  
  } else if (semaforo == "amarillo") {  
    console.log("Comienza a frenar")  
  } else if (semaforo == "rojo") {  
    console.log("Detente")  
  } else {  
    console.log("Por favor introduce un color de semáforo válido");  
  }
```

Por favor introduce un color de semáforo válido

Demostración: ¿Sarah puede sacar su INE?

Queremos escribir un programa que imprima en consola si Sarah puede ir a sacar su INE.



Template Literals | Template Strings

Podemos usar 'backticks' `` para crear un valor de tipo string.

En qué nos ayudan?

Podemos imprimir los valores de variables sin tener que concatenar múltiples valores (string + números).

Dentro de \${ } podemos escribir expresiones

```
1  const firstName = 'Yaxche';  
2  
3  const greeting1 = 'Hola ' + firstName + ', como estás?'  
4  console.log(greeting1);
```

```
1  const firstName = 'Yaxche';  
2  
3  const greeting2 = `Hola, ${firstName}, como estás?`;   
4  console.log(greeting2);
```

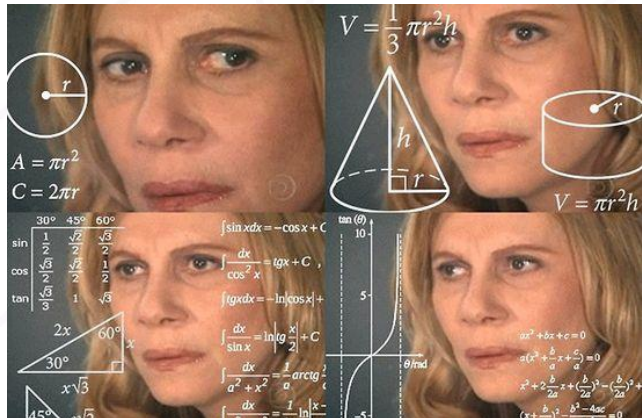
Ejercicio en clase: ¿Este número es par?

Escribe un programa donde se le pida al usuario un número e imprima en consola si el número es par o impar.

Hint:

1. Los números son pares si al dividirse entre 2, el residuo es igual (===) a 0 (cero)

05:00



Otros operadores

DEV.F
DESARROLLAMOS(PERSONAS);

dev

And (&&)

Con el operador **and** (&&), **solo si ambas condiciones se cumplen** (son verdaderas), el resultado **es verdadero**.

| 22<67 && -5<0 | | Resultado final |
|---------------|-------|-----------------|
| true | true | true |
| true | false | false |
| false | true | false |
| false | false | false |

Or (||)

Con el operador **or(||)**, si una sola condición se cumple (es verdaderas), el resultado es **verdadero**.

| Or | | |
|-------|-------|-------|
| true | true | true |
| true | false | true |
| false | true | true |
| false | false | false |

Hoy es jueves ó está lloviendo.

Falsy and Truthy Values

DEV.F
DESARROLLAMOS(PERSONAS);

dev

Evaluando condiciones

- Cuando evaluamos “condicionales” ya sea dentro de un bloque **if** o **while** o **do-while**, Javascript puede trabajar de una manera diferente de lo que esperamos, Sigamos el siguiente ejemplo:

```
if(condicion){}
```

- Es decir, JavaScript comprueba si el valor es verdadero cuando se convierte a booleano. Este tipo de verificación es tan común que se introdujeron los siguientes nombres:

Falsy

Los siguientes valores se evalúan como falso (también conocido como valores Falsy)

- **false**
- **undefined**
- **null**
- **0**
- **NaN**
- **the empty string ("")**

Falsy



```
1 if( undefined ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ false



```
1 if( null ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ false

Falsy



```
1 if( 0 ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ false



```
1 if( NaN ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ false

Falsy



```
1 if( ' ' ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ false

Truthy

Los siguientes valores se evalúan como verdadero (también conocido como valores Trythy). *Todos los que no son falsy*

- **true**
- **15**
- **{}**
- **Strings**
- **[]**

Truthy



```
1 if( {} ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ true



```
1 if( [] ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ true

Truthy



```
1 if( 42 ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ true



```
1 if( 'helou!' ) {  
2     console.log('true');  
3 } else {  
4     console.log('false');  
5 }
```

→ true

Actividad

Ejercicios if-else

Challenge 1:

Escribe un programa donde se le pida al usuario (por medio de un prompt) 2 números diferentes e imprima el más grande.

Prueba con los siguientes datos:

numero 1: 2

numero 2: 7

→ resultado esperado : El número más grande es el 7

numero1 : 5

numero 2: 10

→ resultado esperado: El número más grande es el 10

Actividad

Ejercicios if-else

Challenge 2:

Tenemos dos equipos de fútbol, los Patriots y Broncos. Ellos juegan 3 veces. El ganador será aquel que tenga el promedio de puntos más alto.

Escribe un programa que

1. Calcule el promedio de puntos de cada equipo.
2. Compare los promedios y determine quién es el ganador imprimiéndolo en consola. No olvides que existen los empates (que ambos equipos tengan la misma puntuación promedio)
3. **Bonus 1:** Incluye un requerimiento de al menos 20 puntos, es decir que para que el ganador pueda ser premiado debe de hacer al menos 20 puntos promedio.
4. **Bonus 2:** Qué pasa si queremos agregar el mismo requerimiento de los 20 puntos a para los empates, es decir, si ambos equipos llegan a un empate pero su puntaje promedio es menor a 30 puntos no ganarán trofeo.

Actividad

Ejercicios if-else

Prueba los siguientes datos:

Caso 1.

Puntos para Patriots: 18, 12, 20

Puntos para Broncos: 29, 13, 25

→ resultado esperado: Broncos ganan

Caso 1 - Bonus 1:

Patriots: 18, 20, 27

Broncos: 19, 13, 25

→ resultado esperado: Patriots ganan

Caso 2 - Bonus 1:

Patriots: 8, 3, 14

Broncos: 9, 10, 3

→ resultado esperado: No cumplen con la regla de los 20 puntos

Caso 1 - Bonus 2:

Patriots: 19, 21, 32

Broncos: 18, 22, 32

→ resultado esperado: Hay un empate, ambos ganan

Caso 2 - Bonus 2:

Patriots: 14, 16, 27

Broncos: 13, 17, 27

→ resultado esperado: No cumplen con la regla de los 20 puntos

Tomando decisiones: Declaraciones switch - case

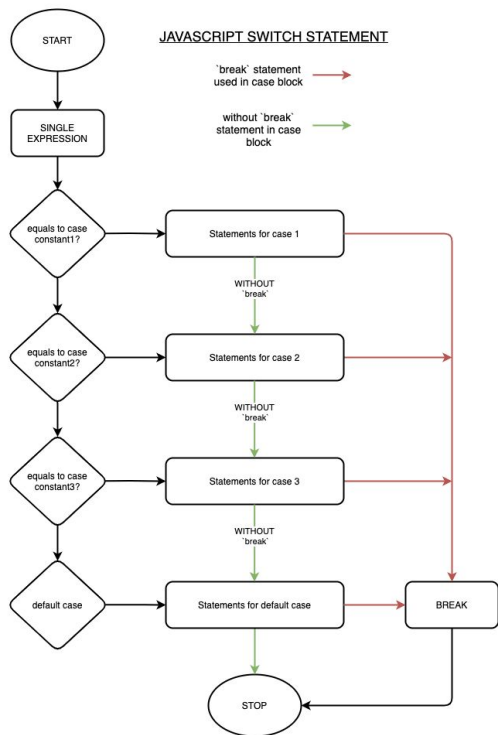


Switch - Case

Es una sentencia que evalúa si una expresión es igual a los casos que se le presentan.

Sintaxis:

```
1
2  switch (expression) {
3      case valor1:
4          sentencia;
5          break;
6
7      case valor2:
8          sentencia;
9      // ...
10
11     case valorN:
12         sentencia;
13
14     default:
15         sentencia;
16 }
```



Switch Case

```
1  let day = 'lunes'
2
3  switch (day) {
4    case 'lunes':
5      console.log('Sumergirme en mi propia miseria');
6      break;
7
8    case 'martes':
9      console.log('Contemplar el abismo');
10     break;
11
12    case 'miercoles':
13      console.log('Solucionar la hambruna mundial (sin decirle a nadie)');
14      break;
15
16    case 'jueves':
17    case 'viernes':
18      console.log('Danza y ejercicio');
19
20    case 'sabado':
21    case 'domingo':
22      console.log('Cena conmigo');
23
24    default:
25      console.log('Ese no es un día de la semana');
26  }
```

OJITO: Compara usando ===

break termina la ejecución.

Se pueden usar { } para separar los bloques de código (se crea un nuevo scope)

break y continue

break termina la ejecución del ciclo o switch que se esté ejecutando.

Sintaxis:

```
break;
```

continue termina la ejecución de la iteración actual del ciclo que se esté ejecutando. Continúa con el resto de las iteraciones

Sintaxis:

```
continue;
```

Más de éste cuando lleguemos a ciclos

Ejercicio en clase: Mini calculadora

Vamos a crear una calculadora de suma y resta.

1. Pide al usuario que escoja un operador (+ o -)
2. Pide al usuario dos números.
3. Dependiendo del operador que se haya elegido, imprime en consola la suma o resta de los dos valores

Hint:

¿Qué vamos a usar como comparador?
→ el operador.

05:00

Actividad

Ejercicio Switch Case

Challenge:

Crea un programa que exprese cuántos días tiene cada mes de éste año.

Pide que el usuario introduzca un mes a través de un prompt y compáralo a través de un switch

Tomando decisiones: Operador ternario

DEV.F
DESARROLLAMOS(PERSONAS);

dev

Operador Ternario | Operador Condicional

Abreviación de un if-else sencillo.

👉 No es una sentencia, es una expresión (regresa un valor)

Fun Fact: Es el único operador en JS que tiene 3 operandos.

Sintaxis:

Condición ? expresión1 : expresión2

```
If(condición) { } else { }
```

Contras?

Solo podemos ejecutar una sola línea de código

Operador Ternario | Operador Condicional



```
1  const age = 20;  
2  let drink;  
3  if (age ≥ 18) {  
4      drink = 'vino';  
5  } else {  
6      drink = 'agua';  
7  }  
8
```



```
1  const age = 20;  
2  let drink;  
3  let drinkTernario = age ≥ 18 ? 'vino' : 'agua';
```



```
1  let drinkTernario = `Me gusta tomar ${age ≥ 18 ? 'vino' : 'agua'}`;
```

Ejercicio en clase: Población en México

Usa el operador ternario para imprimir en consola un string que indique si

“México tiene una población por encima del promedio” si es que la población es mayor a 33 Millones de personas o

“México tiene una población por debajo del promedio” en el caso contrario.

HINT: Toma en cuenta que solo está cambiando una sola palabra entre estas dos frases.

Prueba con los datos:

mexicoPopulation: 23

→ resultado esperado: “Mexico tiene una población por debajo del promedio”

mexicoPopulation: 130

→ resultado esperado: “Mexico tiene una población por encima del promedio”

Actividad

Ejercicio Operador Ternario

Challenge:

Crea un programa que calcule propinas. En México es costumbre que la propina sea del 15% si la cuenta está en el rango de \$100 a \$800. Si el consumo no está en ese rango, la propina es del 20%.

1. Calcula la propina dependiendo del valor del consumo. Crea una variable `tip` (*propina*) que discierna la cantidad a pagar de propina usando el operador ternario. No uses un if-else (Si te es necesario construye una sentencia if-else y luego transfórmala a un operador ternario).
2. Imprime en consola el valor de la cuenta/consumo, la propina y el total (consumo + propina)
Por ejemplo: **“El consumo fue de \$275, la propina es de 41.25 y el total a pagar es de \$316.25”**

Actividad

Ejercicio Operador Ternario

HINT: Para comparar contra el rango de \$100 a \$800 vas a tener que usar el operador lógico &&

Es decir, **cuenta \geq 100 && cuenta \leq 800**