

TypeScript

Estructuras de control

CertiDevs

êndice de contenidos

1. Estructuras de control	1
2. if else	1
3. switch	1
4. for	2
4.1. for cl�sico	2
4.2. for of	2
4.3. for in	2
4.4. forEach	3
5. while	3
6. do while	4

1. Estructuras de control

Las estructuras de control en TypeScript son prácticamente las mismas que en JavaScript, ya que TypeScript es una extensión tipada de JavaScript.

Las estructuras de control permiten controlar el flujo de la ejecución del código basándose en condiciones y ciclos.

Aquí se detallan las estructuras de control más comunes en TypeScript.

2. if else

if else: La declaración **if else** se utiliza para ejecutar un bloque de código si una condición es verdadera, y otro bloque de código si la condición es falsa.

```
let age: number = 18;

if (age >= 18) {
  console.log('You can vote.');
```

```
} else {
  console.log('You cannot vote.');
```

```
}
```

3. switch

switch: La declaración **switch** se utiliza para ejecutar diferentes bloques de código en función del valor de una expresión.

```
let day: number = 3;
let dayName: string;

switch (day) {
  case 1:
    dayName = 'Monday';
    break;
  case 2:
    dayName = 'Tuesday';
    break;
  case 3:
    dayName = 'Wednesday';
    break;
  // ... otros casos
  default:
    dayName = 'Invalid day';
}
```

```
console.log(dayName);
```

4. for

4.1. for clásico

for: se utiliza para repetir un bloque de código un número específico de veces.

```
for (let i: number = 0; i < 5; i++) {  
  console.log(i);  
}
```

¥ Start: `let i: number = 0;` - Se establece una variable de control (generalmente `i`) que actúa como contador.

¥ Stop: `i < 5;` - Mientras la condición sea verdadera, el bucle continuará ejecutándose.

¥ Step: `i++` - Después de cada iteración, la variable de control se actualiza.

4.2. for of

for of: se utiliza para iterar sobre los elementos de una colección iterable, como un array o un conjunto.

El bucle **for of** simplifica la iteración de elementos en una colección sin tener que preocuparse por la variable de control y las condiciones.

```
let numbers: number[] = [1, 2, 3, 4, 5];  
  
for (let num of numbers) {  
  console.log(num);  
}
```

En este caso, `num` toma el valor de cada elemento en el array `numbers` en cada iteración del bucle.

4.3. for in

for in: se utiliza para iterar sobre las claves de un objeto y puede ser útil para extraer información de un objeto.

El bucle **for in** se utiliza para iterar sobre las claves (propiedades) de un objeto.

Aunque este bucle se puede utilizar con colecciones iterables, no se recomienda, ya que no garantiza un orden específico de iteración.

```
let person = {
  name: 'John Doe',
  age: 30,
};

for (let key in person) {
  console.log(`${key}: ${person[key]}`);
}
```

4.4. forEach

El método `forEach()` es una función incorporada en los arreglos (Estructuras de datos) de TypeScript y JavaScript que permite iterar sobre cada elemento de un arreglo y ejecutar una función de callback para cada uno de ellos.

```
const numbers = [1, 2, 3, 4, 5];

numbers.forEach((number, index) => {
  console.log(`Número ${number} en la posición ${index}`);
});
```

En este ejemplo, el método `forEach` recorre cada elemento del arreglo `numbers` y ejecuta la función de flecha proporcionada en cada iteración, pasando el elemento actual ('number') y su índice ('index') como argumentos.

La salida será:

```
Número 1 en la posición 0
Número 2 en la posición 1
Número 3 en la posición 2
Número 4 en la posición 3
Número 5 en la posición 4
```

El método `forEach` es útil cuando deseas realizar operaciones en cada elemento de un arreglo sin preocuparte por el control del flujo, como en un bucle `for` clásico.

Sin embargo, ten en cuenta que 'forEach' no se puede interrumpir mediante una declaración 'break' o 'return', por lo que si necesitas un control de flujo más preciso, es posible que desees utilizar uno de los bucles 'for' mencionados anteriormente.

5. while

while: se utiliza para repetir un bloque de código mientras una condición sea verdadera.

```
let i: number = 0;
```

```
while (i < 5) {  
  console.log(i);  
  i++;  
}
```

6. do while

do while: es similar al bucle while, pero garantiza que el bloque de código se ejecute al menos una vez, incluso si la condición es falsa desde el principio.

```
let i: number = 0;  
  
do {  
  console.log(i);  
  i++;  
} while (i < 5);
```