TypeScript for

Summary: in this tutorial, you will learn about the TypeScript `for` loop statement that executes a piece of code repeatedly.

Introduction to the TypeScript for statement

The following shows the syntax of the TypeScript `for` loop statement:

```
for(initialization; condition; expression) {
    // statement
}
```

The `for` loop statement creates a loop. It consists of three optional expressions separated by semicolons (`;`) and enclosed in parentheses:

- `initialization` : is an expression evaluated once before the loop begins. Typically, you use the `initialization` to initialize a loop counter.
- `condition` – is an expression that is evaluated at the end of each loop iteration. If the `condition` is `true` , the statements in the loop body execute.
- `expression` – is an expression that is evaluated before the `condition` is evaluated at the end of each loop iteration. Generally, you use the `expression` to update the loop counter.

All three expressions in the `for` loop statement are optional. It means that you can use the `for` loop statement like this:

```
for(;;) {
    // do something
}
```

In practice, you should use a `for` loop if you know how many times the `loop` should run. If you want to stop the loop based on a condition other than the number of times the loop executes, you should use a `while` loop.

TypeScript allows you to omit the loop body completely as follows:

```
for(initialization; condition; expression);
```

However, it is rarely used in practice because it makes the code more difficult to read and maintain.

TypeScript for examples

Let's take some examples of using the TypeScript `for` loop statement.

1) Simple TypeScript for example

The following example uses the `for` loop statement to output 10 numbers from 0 to 9 to the console:

```
for (let i = 0; i < 10; i++) {
    console.log(i);
}
```

Output:

```
0
1
2
3
4
5
6
7
```

```
8
9
```

How it works:

- First, declare a variable `i` and initialize it to 0.
- Then check if `i` is less than `10`. If it is, output `i` to the console and increment the variable `i` by one.
- Finally, repeat the second step until `i` equals `10`.

2) TypeScript for example: optional block

The following example shows the same output as the above example. However, the `for` doesn't have the `initialization` block:

```
let i = 0;
for (; i < 10; i++) {
    console.log(i);
}
```

Like the `initialization` block, you can omit the `condition` block.

However, you must escape the loop when a condition is met by using the `if` and `break` statements. Otherwise, you will create an infinite loop that causes the program to execute repeatedly until it is crashed.

```
for (let i = 0; ; i++) {
    console.log(i);
    if (i > 9) break;
}
```

The following example illustrates a `for` loop that omits all three blocks:

```
let i = 0;
for (; ;) {
    console.log(i);
    i++;
```

```
    if (i > 9) break;
}
```

Output:

```
0
1
2
3
4
5
6
7
8
9
```

How it works:

- First, declare a loop counter `i` and initialize it to `0` before entering the for.
- Then, in each loop iteration, output `i` to the console, increment it by one, and break out of the loop if `i` is greater than 9.

Summary

- Use the TypeScript `for` statement when you want to repeatedly execute a piece of code a number of times.