JavaScript Ternary Operator

Summary: in this tutorial, you will learn how to use the JavaScript ternary operator to make your code more concise.

Introduction to JavaScript ternary operator

When you want to execute a block if a condition evaluates to `true`, you often use an if...else statement. For example:

```
let age = 18;
let message;

if (age >= 16) {
  message = 'You can drive.';
} else {
  message = 'You cannot drive.';
}

console.log(message);
```

In this example, we show a message that a person can drive if the age is greater than or equal to 16. Alternatively, you can use a ternary operator instead of the if-else statement like this:

```
let age = 18;
let message;

age >= 16 ? (message = 'You can drive.') : (message = 'You cannot drive.');

console.log(message);
```

Or you can use the ternary operator in an expression as follows:

```
let age = 18;
let message;

message = age >= 16 ? 'You can drive.' : 'You cannot drive.';

console.log(message);
```

Here's the syntax of the ternary operator:

```
condition ? expressionIfTrue : expressionIfFalse;
```

In this syntax, the `condition` is an expression that evaluates to a Boolean value, either `true` or `false`.

If the condition is `true`, the first expression (`expresionIfTrue`) executes. If it is false, the second expression (`expressionIfFalse`) executes.

The following shows the syntax of the ternary operator used in an expression:

```
let variableName = condition ? expressionIfTrue : expressionIfFalse;
```

In this syntax, if the `condition` is `true`, the `variableName` will take the result of the first expression (`expressionIfTrue`) or `expressionIfFalse` otherwise.

JavaScript ternary operator examples

Let's take some examples of using the ternary operator.

1) Using the JavaScript ternary operator to perform multiple statements

The following example uses the ternary operator to perform multiple operations, where each operation is separated by a comma. For example:

```
let authenticated = true;
let nextURL = authenticated
  ? (alert('You will redirect to admin area'), '/admin')
  : (alert('Access denied'), '/403');

// redirect to nextURL here
console.log(nextURL); // '/admin'
```

In this example, the returned value of the ternary operator is the last value in the comma-separated list.

2) Simplifying ternary operator example

See the following example:

```
let locked = 1;
let canChange = locked != 1 ? true : false;
```

If the `locked` is 1, then the `canChange` variable is set to `false`, otherwise, it is set to `true`. In this case, you can simplify it by using a Boolean expression as follows:

```
let locked = 1;
```

```
let canChange = locked != 1;
```

3) Using multiple JavaScript ternary operators example

The following example shows how to use two ternary operators in the same expression:

```
let speed = 90;
let message = speed >= 120 ? 'Too Fast' : speed >= 80 ? 'Fast' : 'OK';

console.log(message);
```

Output:

```
Fast
```

It's a good practice to use the ternary operator when it makes the code easier to read. If the logic contains many `if...else` statements, you should avoid using the ternary operators.

Summary

- Use the JavaScript ternary operator (`?:`)to make the code more concise.

Quiz