# Theory: Break and continue

17 minutes 0 / 5 problems solved

Skip this topic Start practicing

508 users solved this topic. Latest completion was 1 day ago.

Current topic:

Break and continue

Topic depends on:

× Switch

× For loops

× While loops

Table of contents:

When writing loops in programs, you often have to change their standard behavior: finish their execution earlier or skip some iterations (one execution of a loop body is called iteration). This more flexible work with loops is possible due to two operators: `break` and `continue`. In this topic, we will consider how to work with them.

## §1. The break operator

Most often, the loop execution ends when its condition becomes `false`, but the `break` operator allows you to exit the loop at any moment you need. Consider the code sample where this operator is used:

```
for (let n = 1; n <= 50; n++) {
  if (n == 5) {
    break;
  }
  console.log(n); // 4
}
```

In the example above, we created a counter that had to return values between 1 and 50. However, this does not happen because we have set a condition to exit the loop using the `break` operator. It breaks the loop after 5 iterations.

Be careful: it only completes the loop it is in at the moment. If this loop is executed inside another loop, the external loop will not be stopped.

The `break` operator has two uses:

- it terminates the current loop of any type (`for`, `while`, `do-while`);
- it terminates a case in the `switch` statement;

The `break` operator is most often applied when a loop cannot be executed for some reason, such as when the application detects an error.

## §2. The continue operator

The `continue` operator allows you to stop the current loop iteration and start a new one.

This operator can be used inside any kind of loops:

- inside the `while` or `do-while` loop, it returns directly to the loop condition;
- Inside the `for` loop, it first calculates the increment expression and then returns to the condition.

It is used if it's clear that there is nothing else to do on the current loop iteration. Thus, the following code will print all the even numbers from 1 to 10 into the console:

```
for (let n = 1; n <= 10; n++) {
  if (n % 2 !== 0) {
    continue;
  }
  console.log(n); // 2 4 6 8 10
}
```

You cannot use break and continue with the operator `?` as such constructions lead to errors.

Now that you know it all, you will not be afraid of any loops: you can find a way even out of an infinite loop!

Report a typo

51 users liked this theory. 0 didn't like it. **What about you?**

😍 🙂 😐 🙁 😡

Start practicing

Comments (0) Hints (0) Useful links (0) Show discussion