JavaScript continue

Summary: in this tutorial, you will learn how to use the JavaScript `continue` statement to skip the current iteration of a loop.

Introduction to the JavaScript continue statement

The `continue` statement terminates the execution of the statement in the current iteration of a loop such as a for, while, and do…while loop and immediately continues to the next iteration.

Here's the syntax of the `continue` statement:

```
continue [label];
```

In this syntax, the label is optional. It is a valid identifier associated with the label of a statement. Read the `break` statement tutorial for more information on the label statement.

Typically, you use the `continue` with an `if` statement like this:

```
// inside a loop
if(condition){
  continue;
}
```

In this syntax, the `if` statement specifies a condition to execute the `continue` statement inside a loop.

Using the continue statement in a for loop

When using the `continue` statement in a `for` loop, it doesn't terminate the loop entirely. Instead, it jumps to the `iterator` expression.

The following flowchart illustrates how the `continue` statement works in `a` `for` loop:

JavaScript
continue
with for
loop

The following example uses a `continue` in a `for` loop to display the odd number in the console:

```
for (let i = 0; i < 10; i++) {
  if (i % 2 === 0) {
    continue;
  }
  console.log(i);
}
```

Output:

```
1
3
5
7
9
```

In this example, the `for` loop iterates over the numbers from `0` to `9`.

The `i%2` returns the remainder of the division of the current value of `i` by `2`.

If the remainder is zero, the `if` statement executes the `continue` statement that skips the current iteration of the loop and jumps to the iterator expression `i++`. Otherwise, it outputs the value of `i` to the console.

Using the continue statement in a while loop

When using the `continue` statement in a `while` loop, it doesn't terminate the execution of the loop entirely. Instead, it jumps back to the condition.

The following flowchart show hows the continue statement works in a `while` loop statement:

The following example uses the `continue` statement in a `while` loop to display the odd numbers from 1 to 10:

```
let i = 0;
while (i < 10) {
  i++;
  if (i % 2 === 0) {
    continue;
  }
  console.log(i);
}
```

Output:

```
1
3
```

```
5
7
9
```

Using the continue statement with a label example

The `continue` statement can include an optional label like this:

```
continue label;
```

The following nested loop displays pairs of numbers from 1 to 2:

```
for (let i = 1; i < 3; i++) {
  for (let j = 1; j < 3; j++) {
    console.log(i, j);
  }
}
```

Output:

```
1 1
1 2
2 1
2 2
```

The following shows how to use the `continue` statement with a label:

```
outer: for (let i = 1; i < 4; i++) {
  for (let j = 1; j < 4; j++) {
    if (i + j == 3) continue outer;
    console.log(i, j);
  }
}
```

Output:

```
1 1
3 1
3 2
3 3
```

Summary

- Use the JavaScript `continue` statement to skip the current iteration of a loop and continue the next one.

Quiz