The switch and with Statements

In this lesson, you will have an overview of all flow-control statements provided by the JavaScript language.

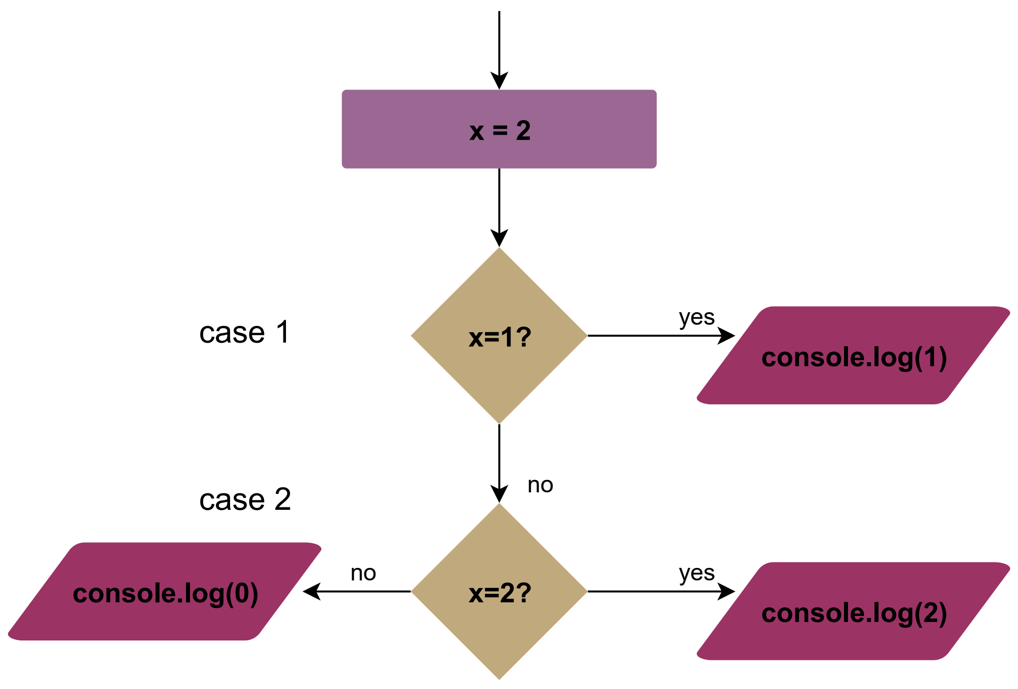
The `switch` statement

As an alternative to compound if statements, JavaScript defines the switch statement.

Illustration

Here is the concept explained in the form of an illustration:

Here is the concept explained in the form of an illustration.



Flowchart of switch case

Syntax

The `switch` statement has the following syntax:

```
switch (expression) {
    case value1:
        statement
        break;
    case value2:
        statement
        break;
        // "case" can be repeated
    case valueN:
        statement
        break;
    default:
        statement
}
```

switch statement syntax in JavaScript

The expression is evaluated, and its value is checked against values in case branches. If the value equals the expression, the statement belonging to the case branch is executed. The break statement automatically terminates the switch statement.

Example

Here is a short example:

```
let x = 3;
switch (x) {
    case 1:
        console.log(1);
        break;
    case 2:
        console.log(2);
        break;
    case 3:
        console.log(3);
        break;
    default:
        console.log("Other than 1, 2, or 3.");
}
```

You can omit the break statement, but in this case the execution falls through to the next case branches, unless a break is found.

Examples

The following sample demonstrates this situation:

```
var num = 1;
switch (num) {
  case 1:
  case 2:
    console.log(2);
  case 3:
    console.log(3);
    break;
  default:
    console.log("Other than 1, 2, or 3.");
} // logs 2 and 3
```

This example will create two log entries, “2” and “3”, although variable `num` is set to 1. When entering the `switch` statement, `num` equals 1, so **case 1** is executed.

It does not contain any statement or break, and execution flows to **case 2** that logs “2”. Because there is no `break` statement, the execution flows to **case 3**, and here “3” is logged. The next `break` terminates the `switch` statement. When the values in case branches are evaluated, you may use any kind of expressions. This lets you to use switch in such a way that is generally not allowed in other programming languages:

```
var num = 6;
switch (true) {
    case num >= 1 && num < 6:
        console.log("1-5");
        break;
    case num >= 6 && num < 10:
        console.log("6-10");
        break;
    default:
        console.log(">10");
} // logs 6-10
```

Here, the switch expression is true, and this is matched with the values of case branches. It means the first case branch is executed in which the branch value is evaluated to true. In this code snippet this is the second branch, so “6-10” will be logged.

The `with` statement

Beside flow-control statements, JavaScript provides the with statement that sets the scope of the code within a particular object. This statement was created as a shortcut when a single object was being coded to over and over again, such as in this example:

Example

```
console.log(document.title);
console.log(document.anchors.length);
console.log(document.links.length);
console.log(document.images.length);
console.log(document.forms.length);
```

The syntax of the `with` statement is the following:

```
with (expression) statement;
```

with statement syntax in JavaScript

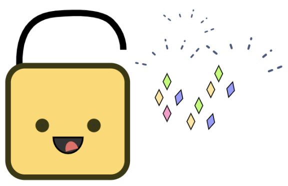
In this case, the statement is executed within the scope of the expression. So, you can rewrite the code above using the `with` statement:

```
with (document) {
  console.log(title);
  console.log(anchors.length);
  console.log(links.length);
  console.log(images.length);
  console.log(forms.length);
}
```

Show Useful Info

Achievement unlocked! 🎉

Congratulations! You've learned how to use the switch and with statements in JavaScript.



Great work! Give yourself a round of applause! :)

In the *next lesson*, we will summarize everything we've learned so far in this chapter.

See you there! :)