# Theory: Boolean and logical operators



8 minutes 0 / 5 problems solved



You already know what the variables of numeric and string types are. However, programming requires more than that. In this topic, we will consider another important variable type and operators that work with it.

## §1. Boolean type

There are statements about which we can say whether they are true or not. Imagine that it is raining today, so the statement "it is raining" is true. The statement "5 < 3" is always false, and "the Moon orbits the Earth" is true. These statements have a special type – logical, or **boolean**.

There are only two possible values of boolean variables: `true` and `false`. Look at the example of setting a boolean variable and printing it:

```
let bool = true;
console.log(bool); // true
```

## §2. Logical operators

In order to perform operations with the boolean variables, **logical operators** are used. There are only three of them in JavaScript: logical AND (`&&`), logical OR (`||`) and NOT (`!`). The first two operators are **binary**, which means that they're used with two operands, and the `!` operator is **unary,** so it only takes one operand. `&&` returns *true* if both operands are true and *false* in all other cases:

```
console.log(true && true);   // true
console.log(true && false);  // false
console.log(false && true);  // false
console.log(false && false); // false
```

`||` returns *false* if both operands are false and *true* in all other cases:

```
console.log(true || true);   // true
console.log(true || false);  // true
console.log(false || true);  // true
console.log(false || false); // false
```

`!` returns *false* to true and *true* to false:

```
console.log(!false);  // true
console.log(!true);   // false
console.log(!!true);  // true
```

## §3. More capabilities of logical operators in JS

Actually, logical operators in JavaScript have much wider capabilities than the traditional usage described above. Their operands can be not only the logical variables but also variables of other types.

Among the numerical values, `0` is considered `false`, and all other numbers are `true`. All strings except the empty ones are considered `true`.

Current topic:

Boolean and logical operators

Topic depends on:

- ✕ Variables
- ✕ Strings and numbers

Topic is required for:

- Type conversion

Table of contents:

Expression is always calculated from left to right. `&&` returns *false* as soon as it finds the first occurring *false*, and the operator `||` returns *true* as soon as it sees the first *true*:

```
console.log(true || 0);      // true
console.log(false && "sun"); // false
console.log(1 || 0);         // 1
```

## §4. Priority

When working with complex expressions, you should take into account their **priority**. The priority of `!` is higher than that of `&&`, and the priority of `&&` is higher than that of `||`. If you need to change the priority, use parentheses:

```
console.log(!false && !true);   // false
console.log(!(false && !true)); // true
```

## §5. Conclusion

In this topic, we learned what booleans are, reviewed three logical operators in JavaScript and saw that JS logical operators can be applied not only to logical variables. Now it is time to move on to applying this knowledge in practice.

Report a typo

**129** users liked this theory. **2** didn't like it. **What about you?**

Start practicing

Comments (5) Hints (0) Useful links (1) Show discussion