Logical Operators: !, ||, &&

Learn operators that will make our conditional code much more interesting and powerful. Learn how to combine operators for more complex logic.

We'll cover the following

Let's introduce some logical operators to make if-statements more powerful.

!

The `!` operator is called the "not"-operator. It does two things to a value: it coerces it to either `true` or `false`, and then gives the *opposite* value. Falsey values become `true` and truthy values become `false`.

For example, `!true` will become `false`. `!0` will become `true`. `!''` will become `true`. Think of this operator as saying "This item is NOT true" or "This item is NOT false".

```
// Falsey values become true
console.log(!false);      // -> true
console.log(!'');         // -> true
console.log(!0);          // -> true
console.log(!null);       // -> true
console.log(!undefined);  // -> true
console.log(!NaN);        // -> true

```

```
// Truthy values become false
console.log(!true);      // -> false
console.log(!1);         // -> false
console.log(!'abc');     // -> false
console.log(!9999999);   // -> false
console.log(!'ha*UIHJ'); // -> false
```

`!` in if-statements#

We can use this in if-statements. If we want some code to run when something is falsey, it comes in handy.

```
let falseyValue = false;

if(!falseyValue) {
    console.log('This will run!');
} else {
    console.log('This will not run :(');
}
```

`||`#

The vertical bar that makes up this symbol is found on the right side of the keyboard. Double vertical bars is referred to as the "or"-operator.

To use `||`, we put an item on the left of it and another on the right. It follows these rules:

- If both items are truthy, the whole thing will be truthy.
- If one is truthy and the other is falsey, the whole thing will be truthy.
- If both items are falsey, the whole thing will be falsey.

In other terms:

- `true || true -> true`
- `true || false -> true`

- `false || true -> true`
- `false || false -> false`

A common use case is if-statements. We can use it in an if-statement when we want the code block to run if either one of the statements is “truthy”.

Play around with the code block below to test out the operator.

```javascript
let firstItem = true;
let secondItem = false;

if(firstItem || secondItem) {
    console.log('One or both of the items is truthy!');
} else {
    console.log('Neither of the items is truthy.');
}
```

&&

`&&` is another logical operator. To use it, we put an item on the left of it and another on the right. It follows these rules:

- If both items are truthy, the whole thing will be truthy.
- If one is truthy and the other is falsey, the whole thing will be falsey.
- If both items are falsey, the whole thing will be falsey.

In other terms:

- `true && true -> true`
- `true && false -> false`
- `false && true -> false`
- `false && false -> false`

We use it in an if-statement when we want the code block to run only if both items are truthy.

Play around with this code block to test it out.

```
let firstItem = true;
let secondItem = false;

if(firstItem && secondItem) {
    console.log('Both of the items are truthy!');
} else {
    console.log('One or both of the items is falsey.');
}
```

Combining Operators#

We can set up complex logic with these operators. If we have two items, how do we set up an if-statement such that it runs if both items are falsey? We can use both `!` and `&&`.

```
let firstItem = true;
let secondItem = false;

if(!firstItem && !secondItem) {
    console.log('Both of the items are falsey!');
} else {
    console.log('One or both of these items is truthy.');
}
```

These operators aren't read from left to right. There's an order to how the engine reads them.

Order of Operators#

1. Not: `!`
2. Logical AND `&&`
3. Logical OR `||`

This means that

```
!a || b && c
```

is evaluated like:

```
(!a) || (b && c)
```

Note that we can put pretty much any code inside parentheses. The parentheses take precedence over all operators, meaning anything we group in parentheses will be evaluated first, just like in math.

Quiz#

Feel free to test your understanding.

1 What will the following code print?

```
if ('abc' || 19) {
        console.log('A');
} else {
        console.log('B');
}
```

○ **A)** A

○ **B)** B

Submit Answer

Question 1 of 8
0 attempted

Reset Quiz

Back
if-statements, continued

Next
Comparison Operators: <, >, <=, >=, =...

✓ Completed

Completed

Report an Issue