Logical Operators

**Introduction to Logical Operators in JavaScript**

Let's now return to our code and learn how logical operators work in JavaScript. We will use the Boolean variables from the last lecture, specifically the ones about having a driver's license and having good vision.

**Defining Boolean Variables**

We will create a variable for each condition. The first one will be **hasDriversLicense**, and let's start by setting it to **true**. Remember, this corresponds to variable A from the last lecture. Then, let's create **hasGoodVision** and also initialize it with **true**. These two variables, A and B, were used as examples previously, so it's a good idea to use them here as well.

**Using the AND Operator**

Let's log to the console the result of using the AND operator on these two variables. The AND operator in JavaScript is represented by **&&**. It combines two logical values. For example:

**hasDriversLicense && hasGoodVision**

Since both variables are **true**, the result should be **true**. Let's verify this.

**javascript Code Sample**

console.log(hasDriversLicense && hasGoodVision);

**Changing One Variable to False**

Now, let's change one of the variables to **false**. Remember, the result of **true && false** will be **false** because it is enough for one variable to be false for the entire AND operation to be false. Let's confirm this.

**javascript Code Sample**

hasGoodVision = false;

console.log(hasDriversLicense && hasGoodVision);

**Using the OR Operator**

Next, let's see how the OR operator works. We will use the OR operator, represented by **||**, instead of AND. For example:

**hasDriversLicense || hasGoodVision**

What do you think the result will be? Since one of the variables is **true**, the result should be **true**. Let's check.

**javascript Code Sample**

console.log(hasDriversLicense || hasGoodVision);

**Using the NOT Operator**

Finally, we can use the NOT operator to invert a Boolean value. The NOT operator in JavaScript is the exclamation mark **!**. For example:

**!hasDriversLicense**

This will change **true** to **false** and vice versa.

**javascript Code Sample**

```javascript
console.log(!hasDriversLicense);
```

**Decision Making with Boolean Variables**

Now that we understand how logical operators work, let's use these variables to make a decision. Suppose we want to determine whether Sarah should drive or not. We will create a new Boolean variable **shouldDrive** based on the conditions that Sarah has a driver's license and good vision.

**javascript Code Sample**

```javascript
let shouldDrive = hasDriversLicense && hasGoodVision;
```

**javascript Code Sample**

```javascript
if (shouldDrive) {
  console.log('Sarah is able to drive');
} else {
  console.log('Someone else should drive');
}
```

**Explanation**

If **shouldDrive** is **true**, the console will log "Sarah is able to drive". Otherwise, it will log "Someone else should drive". This condition checks that both **hasDriversLicense** and **hasGoodVision** are true for Sarah to drive.

**Testing Different Conditions**

If **hasGoodVision** is **false**, the condition fails, and the else block executes. If both variables are **true**, Sarah is able to drive.

**javascript Code Sample**

```javascript
hasGoodVision = true;
shouldDrive = hasDriversLicense && hasGoodVision;


if (shouldDrive) {
  console.log('Sarah is able to drive');
} else {
  console.log('Someone else should drive');
}
```

**Adding a Third Variable: isTired**

Let's add another Boolean variable **isTired** and set it to **true**. This will be variable C. We can use more than two variables with logical operators by chaining them together. For example:

**hasDriversLicense && hasGoodVision && !isTired**

This means Sarah should drive only if she has a driver's license, has good vision, and is not tired.

**javascript Code Sample**

```javascript
let isTired = true;

shouldDrive = hasDriversLicense && hasGoodVision && !isTired;
```

**javascript Code Sample**

```javascript
if (shouldDrive) {

  console.log('Sarah is able to drive');

} else {

  console.log('Someone else should drive');

}
```

**Explanation**

Since **isTired** is **true**, the expression **!isTired** evaluates to **false**. Therefore, the entire condition is false, and the else block executes, indicating someone else should drive. To fix this, set **isTired** to **false**.

**javascript Code Sample**

```javascript
isTired = false;

shouldDrive = hasDriversLicense && hasGoodVision && !isTired;


if (shouldDrive) {

  console.log('Sarah is able to drive');

} else {

  console.log('Someone else should drive');

}
```

**Summary**

With these three Boolean operators—AND (**&&**), OR (**||**), and NOT (**!**)—we can model complex decision-making scenarios. Although it may seem confusing at first, practicing these concepts will help you understand how to use logical operators effectively in programming.

**Practice Challenge**

To reinforce your understanding, try applying these logical operators in your own coding exercises. Experiment with different Boolean variables and conditions to see how the operators affect the outcomes.

**Key Takeaways**

- Logical operators in JavaScript include AND (**&&**), OR (**||**), and NOT (**!**).
- The AND operator returns true only if both operands are true.
- The OR operator returns true if at least one operand is true.
- The NOT operator inverts the Boolean value of its operand.

- Boolean variables can be combined to model complex decision-making conditions.