

Equality Operators: == vs. ===

Introduction to Equality Operators

In this lecture, we explore the use of equality operators in JavaScript, focusing on the differences between the strict (===) and loose (==) equality operators.

Let us begin by creating a variable named **age** and assigning it the value 18. The **age** variable is often used for simple calculations.

javascript Code Sample

```
age = 18
```

Now, let us create an if statement that logs to the console that the person just became an adult, only if the age is exactly 18. To check if the age is exactly 18, we use the triple equals (===).

javascript Code Sample

```
if (age === 18)

    console.log('You just became an adult.');
```

If the if block contains only one line, curly braces are not required. Testing this code will display the message when age is 18. If the value is changed to something else, such as 19, the message will not appear.

The strict equality operator (===) returns **true** only if both sides are exactly the same, including type. For example:

javascript Code Sample

```
18 === 18 // true

18 === 19 // false
```

Do not confuse the assignment operator (=) with the comparison operator (===). Besides the triple equals, there is also a double equals (==). The triple equals is called the strict equality operator because it does not perform type coercion. The double equals is the loose equality operator and does perform type coercion.

javascript Code Sample

```
18 == '18' // true

18 === '18' // false
```

The loose equality operator (==) converts the string '18' to the number 18 before comparison, resulting in **true**. The strict equality operator (===) does not perform this conversion, so the comparison is **false**.

As a general rule for clean code, avoid the loose equality operator as much as possible. Always use strict equality (===) when comparing values. If type conversion is needed, convert the value manually before comparison.

Using Prompt and Type Conversion

There is a simple way to get a value from any webpage using the **prompt** function. The value returned by **prompt** is always a string.

javascript Code Sample

```
favorite = prompt("What's your favorite number?");
```

```
console.log(favorite);
```

```
console.log(typeof favorite);
```

The console will display the value as a string. To check if the favorite number is 23 using the loose equality operator:

javascript Code Sample

```
if (favorite == 23)

    console.log('Cool. 23 is an amazing number!');
```

This works because the loose equality operator converts the string to a number. However, using the strict equality operator will not work unless the type is converted.

javascript Code Sample

```
if (favorite === 23)

    console.log('Cool. 23 is an amazing number!');
```

To use strict equality, convert the input to a number using the **Number** function:

javascript Code Sample

```
favorite = Number(prompt("What's your favorite number?"));

if (favorite === 23)

    console.log('Cool. 23 is an amazing number!');
```

Now, **favorite** is a number, and the strict equality comparison will work as expected.

Using Else If and Multiple Conditions

You can add more conditions to an if-else statement using **else if** blocks.

javascript Code Sample

```
if (favorite === 23) {

    console.log('Cool. 23 is an amazing number!');

} else if (favorite === 7) {

    console.log('7 is also a cool number!');

} else {

    console.log('Number is not 23 or 7.');
```

You can continue adding more **else if** blocks for additional conditions, such as checking if the number is 9.

javascript Code Sample

```
if (favorite === 23) {

    console.log('Cool. 23 is an amazing number!');

} else if (favorite === 7) {
```

```
    console.log('7 is also a cool number!');  
  } else if (favorite === 9) {  
    console.log('9 is also a cool number!');  
  } else {  
    console.log('Number is not 23, 7, or 9.');
```

This allows you to check multiple conditions in sequence before reaching the final else block.

The Inequality Operator

There is also an operator for checking if values are different. The strict inequality operator is `!==`, and the loose version is `!=`. Always use the strict version.

javascript Code Sample

```
if (favorite !== 23)  
  console.log('Why not 23?');
```

If the favorite number is not 23, the message will be logged. If it is 23, only the corresponding block will execute.

Sometimes you need the inequality operator, and sometimes the equality operator. Choose the one that fits your problem, but always use the strict version for reliability.

Key Takeaways

- The strict equality operator (`===`) checks for both value and type equality, while the loose equality operator (`==`) performs type coercion.
- It is recommended to use the strict equality operator to avoid unexpected bugs caused by type coercion.
- When comparing user input, convert strings to numbers before using strict equality.
- The strict inequality operator (`!==`) should be preferred over the loose version (`!=`) for clarity and reliability.