Web Programming 2x1

JavaScript Operators

BELGIUM CAMPUS iTversity

Outcomes

Students should understand the following outcomes, upon successful completion of this module:

- Operators’ precedence
- Assignment Operators
- Comparison Operators
- Arithmetic operators
- Concatenation Operators
- Logical Operators
- Combining Operators

Operators in JavaScript

- Operators are symbols that perform operations on operands (values and variables).
- They are categorized into different types based on their functionality
- Operators precedence is essential for writing efficient and correct code.
- JavaScript contains:

Binary Operator

A binary operator operates on two operands. It appears between two operands, which can be variables, literals, or expressions.

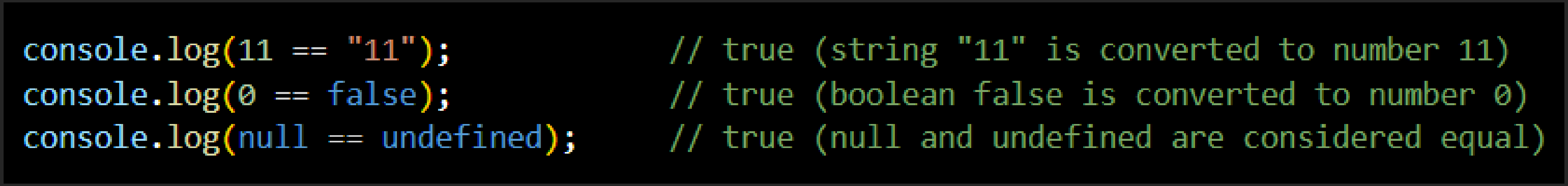
- Arithmetic Operators: +, -, *, /, %
- Comparison Operators: ==, !=, >, <, >=, <=

Equality Operators

- Both == and === are used for comparison, but they differ in terms of strictness and type coercion.

Loose Equality Operator ==

- The == operator checks for equality between two operands after performing type coercion if necessary.
- It tries to convert the operands to the same type before making the comparison.



```
console.log(11 == "11");          // true (string "11" is converted to number 11)
console.log(0 == false);          // true (boolean false is converted to number 0)
console.log(null == undefined);   // true (null and undefined are considered equal)
```

Comparison Operators

Strict Equality Operator ===

- The === operator checks for equality between two operands without performing type coercion. It compares both the values and the types of the operands.

```
console.log(7 === "7");            // false (string "7" is not equal to number 7)
console.log(0 === false);          // false (number 0 is not equal to boolean false)
console.log(null === undefined);   // false (null and undefined are not strictly equal)
```

Key Differences

- Type Coercion: == performs type coercion, meaning it may convert operands to a common type before comparison.

- Strictness: === checks both value and type, ensuring both operands are exactly the same.

- === does not perform type coercion.

Operators in JavaScript

Unary Operator

A unary operator operates on a single operand. It appears before or after its operand.

- Increment and Decrement Operators: ++, --
- Logical NOT Operator: !

Ternary Conditional Operator

The ternary conditional operator (? :) is unique in JavaScript as it takes three operands.

It evaluates a condition and returns one of two expressions, depending on whether the condition is true or false.

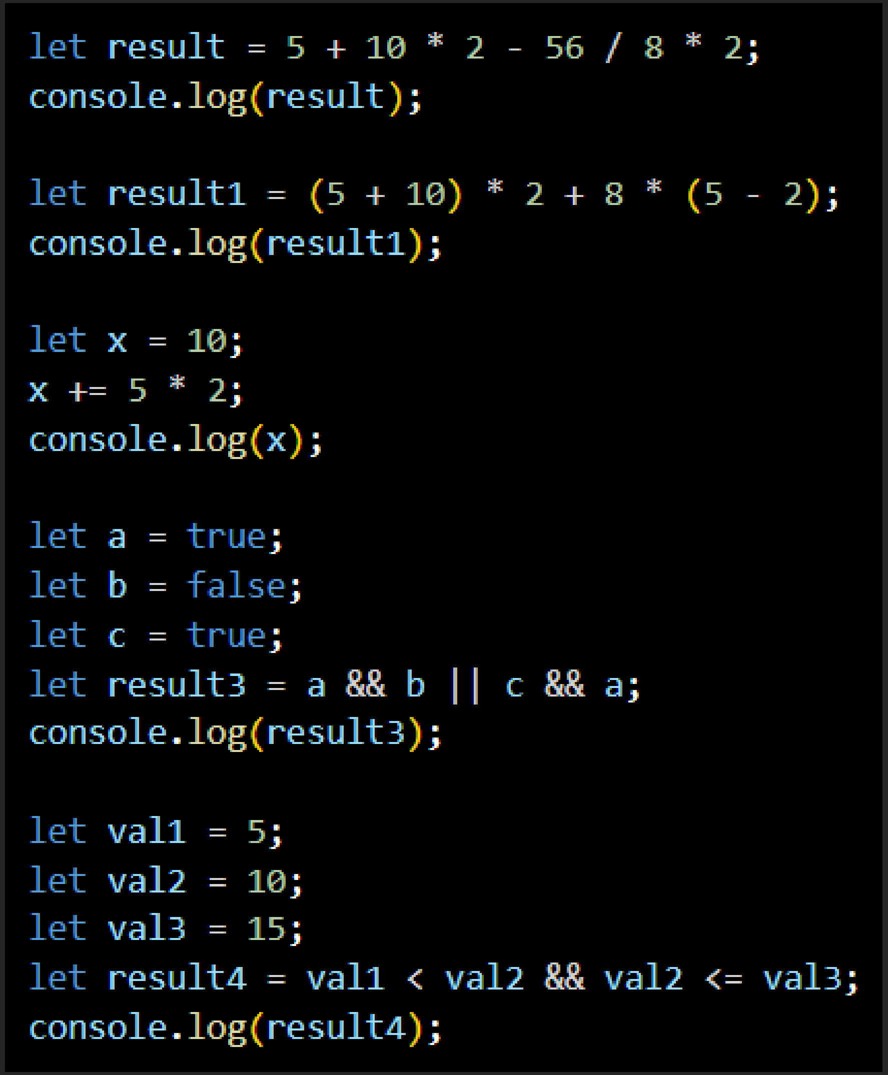
Syntax:

```
condition ? expressionIfTrue : expressionIfFalse
```

BELGIUM CAMPUS iTversity

Operator Precedence

- Operator precedence in JavaScript determines the order in which operators are evaluated when multiple operators are used in an expression.
- Operators with higher precedence are evaluated first.
- If operators have the same precedence, they are evaluated from left to right.



```
let result = 5 + 10 * 2 - 56 / 8 * 2;
console.log(result);

let result1 = (5 + 10) * 2 + 8 * (5 - 2);
console.log(result1);

let x = 10;
x += 5 * 2;
console.log(x);

let a = true;
let b = false;
let c = true;
let result3 = a && b || c && a;
console.log(result3);

let val1 = 5;
let val2 = 10;
let val3 = 15;
let result4 = val1 < val2 && val2 <= val3;
console.log(result4);
```

Thank You!

The End

info@belgiumcampus.ac.za +27 10 593 53 68

/belgiumcampusSA

#Belgium Campus

/belgiumcampus