Day 16 of Training

JavaScript: Operators and Conditional Logic

Understanding Comments in Code

The session began with a practical but often overlooked aspect of programming: comments. We learned that comments are essential for explaining what our code does, making it understandable for ourselves later and for other developers. We explored both single-line comments (//) and multi-line comments (/* ... */), noting that anything within these markers is ignored by the JavaScript engine and doesn't affect the program's execution. This emphasized the importance of clear, descriptive documentation within our code.

Introduction to Operators

The main topic for the day was operators, which are symbols that perform operations on data. The instructor highlighted that once we grasp operators and conditional statements in JavaScript, the underlying logic can be applied to many other programming languages like C++, Java, or Python. We learned about:

- **Arithmetic Operators:** These are familiar mathematical operations like addition (+), subtraction (-), multiplication (*), and division (/). We also covered the less common but crucial modulo operator (%), which returns the remainder of a division, and the exponentiation operator (**), used for powers.
- **Unary Operators:** These are special arithmetic operators that work on a single operand. We focused on increment (++) and decrement (--) operators, which increase or decrease a variable's value by one. We also distinguished between "pre" and "post" increment/decrement, understanding how they affect when the value is updated in an expression.
- **Assignment Operators:** These operators are used to assign values to variables. The most basic is the simple assignment operator (=). We also learned shorthand assignment operators like +=, -=, *=, /=, %=, and **=, which combine an arithmetic operation with assignment (e.g., a += 4 is equivalent to a = a + 4).

Comparison Operators: Making Decisions

Next, we delved into comparison operators, which are used to compare two values and always return a boolean result (true or false). We covered:

- **Loose Equality (==) and Strict Equality (===):** A critical distinction was made between == (which compares only values and performs type coercion) and === (which compares both value and data type, preventing unexpected conversions). The strict equality operator (===) is generally preferred for more predictable comparisons.
- **Inequality (!=) and Strict Inequality (!==):** Similar to equality, these check if values are not equal, with !== being the strict version.
- **Greater than (>), Less than (<), Greater than or equal to (>=), Less than or equal to (<=):** These are standard relational operators used for numerical comparisons.

Logical Operators: Combining Conditions

The final set of operators introduced were logical operators, used to combine multiple boolean expressions. These also return a boolean result:

- **Logical AND (&&):** Returns true only if *all* conditions are true; otherwise, it returns false.
- **Logical OR (||):** Returns true if *at least one* condition is true; it only returns false if all conditions are false.
- **Logical NOT (!):** Reverses the boolean value of an expression (turns true to false and false to true).

Conditional Statements: If, Else If, and Else

With operators in hand, we moved on to conditional statements, which allow our code to execute different blocks of logic based on whether a condition is true or false.

- **if Statement:** Executes a block of code if a specified condition is true.
- **if-else Statement:** Provides an alternative block of code to execute if the if condition is false.
- **if-else if-else Statement:** Used for checking multiple conditions sequentially. If the if condition is false, it moves to the else if conditions, and if none of those are true, the final else block is executed.

The Ternary Operator

A special operator called the **Ternary Operator** was also introduced. It's a compact way to write simple if-else statements, taking three operands: a condition, a value if true, and a value if false.

Exercise:

1. **Check Multiples:** Write a simple JavaScript program that prompts the user to "Enter a number". Then, check if the entered number is a multiple of 5. Print a message indicating whether it is or isn't. (Hint: Use the modulo operator and an if-else statement)
2. **Student Grading System:** Create a program that takes a student's score (between 0 and 100) as input. Assign and print a grade based on the following scale using if-else if-else statements:
    - 90-100: A
    - 70-89: B
    - 60-69: C
    - 50-59: D
    - 0-49: F