Programming JS

Margit Tennosaar

Basic function sum up

```
function sum(x, y) {
  return x + y;
}
```

```
sum(25, 35)
```

```
function multiply(x, y) {
  return x * y;
  console.log('Hello World'); // this will NEVER run
}
```

```
const divide = (x, y) => x / y;
```

```
divide(25, 35)
```

Conditions

Conditional statements enable your code to make decisions based on certain conditions.

Execute different code blocks depending on whether a condition is true or false.

If statement

- Checks if something is true.

- If true, executes the code inside the if block.

```
if (age >= 18) {
  console.log("You are an adult");
}
```

The if...else Statement

Provides two options: one if the condition is true, another if false.

```
if (age >= 18) {
  console.log("You are an adult");
} else {
  console.log("You are a minor");
}
```

The if...else if...else Statement

- Allows checking more than one condition.

- Provides multiple options.

```
if (score >= 90) {
  console.log("Grade: A");
} else if (score >= 80) {
  console.log("Grade: B");
} else {
  console.log("Grade: F");
}
```

Logical Operators

Combine multiple conditions into a single statement.

&& (AND) Operator

- All conditions must be true.
- **Example:** if (age >= 18 && hasID)

|| (OR) Operator

- At least one condition must be true.
- **Example:** if (age >= 18 || hasPermission)

! (NOT) Operator

- Inverts the condition.
- **Example:** if (!isRaining)

Ternary Operator

- Shorthand for if...else statements.

- Useful for simple conditions.

```
condition ? expressionIfTrue : expressionIfFalse
```

```
let message = age >= 18 ? "Adult" : "Minor";
console.log(message); // Outputs: "Adult”
```

Identifying Even and Odd Numbers

Use the modulus operator (%) to determine if a number is even or odd.

```
// Even
8 % 2; // 0, hence even

// Odd
7 % 2; // 1, hence odd
```

The switch Statement

Alternative to if...else when comparing one variable to multiple values.

```
switch (expression) {
  case value1:
    // Code to execute
    break;
  case value2:
    // Code to execute
    break;
  default:
    // Code to execute if no match
}
```

```
switch (day) {
  case 1:
    console.log('Monday');
    break;
  case 2:
    console.log('Tuesday');
    break;
  default:
    console.log('Invalid day');
}
```

Simplifying Conditions

```
function isAdult(age) {
  if (age >= 18) {
    return true;
  }
  return false;
}
```

```
function isEligibleForVote(age) {
  return age >= 18;
}
```

When to use switch vs if

- **Use switch:** When comparing the same variable to many constant values.

- **Use if:** When evaluating different conditions or ranges.

Best practices

- Keep code readable; avoid too many nested `if...else` blocks.
- Combine conditions with `&&` (AND) and `||` (OR).
- Use `else` to catch all other cases not covered by `if` or `else if`.
- Use the ternary operator for simple `if...else` conditions.
- switch` is great for handling multiple possible values of a single variable.
- Consider unexpected inputs like `null`, `undefined`, or unusual values.
- Return the condition directly when possible.
- Use a consistent coding style to avoid errors.
- Add comments to explain complex logic in your conditions.