Increment Decrement operator in JS

In JavaScript, the increment and decrement operators are used to increase or decrease the value of a variable by 1. They are represented by the symbols "++" and "--" respectively. These operators can be applied to both numeric and string variables.

Let us understand increment decrement operators with examples:

1. Increment Operator (++): The increment operator increases the value of the operand by adding 1 to the current value of the operand.

Example:

```
let num = 15;
num++; // Incrementing num by 1
console.log(num); // Output: 16
```

In this example, the variable **num** is initially assigned the value of 15. When we use the increment operator **++**, it adds 1 to the current value of **num**, resulting in 16.

2. Decrement Operator (--): The decrement operator decreases the value of the operand by subtracting 1 from the current value of the operand.

Example:

```
let count = 10;
count--; // Decrementing count by 1
console.log(count); // Output: 9
```

Prefix form

When we place the ++ or -- operators before the operands, it is called prefix form. In the prefix form the ++ or – is executed first then the updated value is used in the expression.

Prefix Form Example:

```
let x = 15;
let y = ++x;
console.log(x); // Output: 16
console.log(y); // Output: 16
```

In the prefix form, **++x**, the value of **x** is incremented first, and then the updated value is assigned to **y**.

Postfix Form

When we place the ++ or -- operators after the operands, it is called the postfix form. In the postfix form the ++ or – is executed after the execution of other operators in the expression.

Postfix Form Example:

```
let a = 7;
let b = a--;
console.log(a); // Output: 6
console.log(b); // Output: 7
```

In the postfix form, **a--**, the current value of **a** is assigned to **b** first, and then the value of **a** is decremented.

Using increment decrement operators with strings

In JavaScript, the increment and decrement operators can be used with strings, but their behavior is slightly different compared to numeric variable.

When the increment or decrement operator is applied to a string, it converts the string into a numeric value and then performs the operation. The conversion of a string to a number is done using the JavaScript's built-in **parseInt()** function.

Here's an example to illustrate this:

```
let str = "5";
str++; // Incrementing the string value
console.log(str); // Output: 6
```

In this example, the variable **str** is initially assigned the value of "5" as a string. When we apply the increment operator **++** to it, JavaScript internally converts the string into a numeric value (using **parseInt()**), which becomes 5. Then it increments the numeric value by 1, resulting in 6.

Similarly, the decrement operator **--** works in the same way when applied to a string:

```
let str2 = "10";
str2--; // Decrementing the string value
console.log(str2); // Output: 9
```

Here, the string "10" is converted to the numeric value 10, and then the decrement operation is performed, resulting in 9.

It's important to note that if the string cannot be converted to a valid number, the result will be **NaN** (Not a Number):

```
let invalidStr = "Hello";
invalidStr++; // Invalid string, conversion to NaN
console.log(invalidStr); // Output: NaN
```

In this case, since "Hello" cannot be converted to a numeric value, the result is **NaN**.