JavaScript Arithmetic Operators

Summary: in this tutorial, you will learn how to use JavaScript arithmetic operators to perform arithmetic calculations.

Introduction to the JavaScript Arithmetic Operators

JavaScript supports the following standard arithmetic operators:

Operator	Sign
Addition	+
Subtraction	-
Multiplication	*
Division	/

An arithmetic operator accepts numerical values as operands and returns a single numerical value. The numerical values can be literals or variables.

Addition operator (+)

The addition operator returns the sum of two values. For example, the following uses the addition operator to calculate the sum of two numbers:

```
let sum = 10 + 20;
console.log(sum); // 30
```

You can also use the addition operator with two variables. For example:

```
let netPrice    = 9.99,
    shippingFee = 1.99;
let grossPrice  = netPrice + shippingFee;

console.log(grossPrice);
```

Output:

```
11.98
```

If either value is a string, the addition operator uses the following rules:

- If both values are strings, it concatenates the second string to the first one.
- If one value is a string, it implicitly converts the numeric value into a string and concatenates two strings.

For example, the following uses the addition operator to concatenate two strings:

```
let x = '10',
    y = '20';
let result = x + y;

console.log(result);
```

Output:

```
1020
```

The following example shows how to use the addition operator to calculate the sum of a number and a string:

```
let result = 10 + '20';

console.log(result);
```

Output:

```
1020
```

In this example, JavaScript converts the number `10` into a string `'10'` and concatenates the second string `'20'` to it.

The following table shows the result when using the addition operator with special numbers:

First Value	Second Value	Result	Explanation
NaN		NaN	If either value is NaN, the result is NaN
Infinity	Infinity	Infinity	Infinity + Infinity = Infinity
-Infinity	-Infinity	-Infinity	-Infinity + (-Infinity) = – Infinity
Infinity	-Infinity	NaN	Infinity + -Infinity = NaN
+0	+0	+0	+0 + (+0) = +0
-0	+0	+0	-0 + (+0) = +0
-0	-0	-0	-0 + (-0) = -0

Subtraction operator (-)

The subtraction operator (`-`) subtracts one number from another. For example:

```
let result = 30 - 10;
console.log(result); // 20
```

If a value is a string, a boolean, null, or undefined, the JavaScript engine will:

- First, convert the value to a number using the `Number()` function.
- Second, perform the subtraction.

The following table shows how to use the subtraction operator with special values:

First Value	Second Value	Result	Explanation
NaN		NaN	If either value is NaN, the result is NaN
Infinity	Infinity	NaN	Infinity – Infinity = NaN
-Infinity	-Infinity	-Infinity	-Infinity – (-Infinity) = NaN
Infinity	-Infinity	Infinity	Infinity
+0	+0	+0	+0 – (+0) = 0
+0	-0	-0	+0 – (-0) = 0
-0	-0	+0	-0 – (-0) = 0

Multiplication operator (*)

JavaScript uses the asterisk (*) to represent the multiplication operator. The multiplication operator multiplies two numbers and returns a single value. For example:

```
let result = 2 * 3;
console.log(result);
```

Output:

```
6
```

If either value is not a number, the JavaScript engine implicitly converts it into a number using the `Number()` function and perform the multiplication. For example:

```
let result = '5' * 2;

console.log(result);
```

Output:

```
10
```

The following table shows how the multiply operator behaves with special values:

First Value	Second Value	Result	Explanation
NaN		NaN	If either value is NaN, the result is NaN
Infinity	0	NaN	Infinity * 0 = NaN
Infinity	Positive number	Infinity	-Infinity * 100 = -Infinity
Infinity	Negative number	-Infinity	Infinity * (-100) = -Infinity
Infinity	Infinity	Infinity	Infinity * Infinity = Infinity

Divide operator (/)

Javascript uses the slash (`/`) character to represent the divide operator. The divide operator divides the first value by the second one. For example:

```
let result = 20 / 10;

console.log(result); // 2
```

If either value is not a number, the JavaScript engine converts it into a number for division. For example:

```
let result = '20' / 2;
console.log(result); // 10;
```

The following table shows the divide operators' behavior when applying to special values:

First Value	Second Value	Result	Explanation
NaN		NaN	If either value is NaN, the result is NaN
A number	0	Infinity	1/0 = Infinity

First Value	Second Value	Result	Explanation
Infinity	Infinity	NaN	Infinity / Infinity = NaN
0	0	NaN	0/0 = NaN
Infinity	A positive number	Infinity	Infinity / 2 = Infinity
Infinity	A negative number	-Infinity	Infinity / -2 = -Infinity

Using JavaScript arithmetic operators with objects

If a value is an object, the JavaScript engine will call the `valueOf()` method of the object to get the value for calculation. For example:

```
let energy = {
  valueOf() {
    return 100;
  },
};

let currentEnergy = energy - 10;
console.log(currentEnergy);

currentEnergy = energy + 100;
console.log(currentEnergy);

currentEnergy = energy / 2;
console.log(currentEnergy);

currentEnergy = energy * 1.5;
console.log(currentEnergy);
```

Output:

```
90
200
50
150
```

If the object doesn't have the `valueOf()` method but has the `toString()` method, the JavaScript engine will call the `toString()` method to get the value for calculation. For example:

```
let energy = {
  toString() {
    return 50;
```

```
  },
};

let currentEnergy = energy - 10;
console.log(currentEnergy);

currentEnergy = energy + 100;
console.log(currentEnergy);

currentEnergy = energy / 2;
console.log(currentEnergy);

currentEnergy = energy * 1.5;
console.log(currentEnergy);
```

Output:

```
40
150
25
75
```

More on the valueOf vs. toString methods of the object.

Summary

- Use the JavaScript arithmetic operators including addition (`+`), subtraction (`-`), multiply (`*`) and divide (`/`) to perform arithmetic operations.

Quiz