Operator precedence

see all contributors

Operator precedence determines the order in which operators are evaluated. Operators with higher precedence are evaluated first.

A common example:

```
3 + 4 * 5 // returns 23
```

The multiplication operator ("*") has higher precedence than the addition operator ("+") and thus will be evaluated first.

Associativity

Associativity determines the order in which operators of the same precedence are processed. For example, consider an expression:

```
a OP b OP c
```

Left-associativity (left-to-right) means that it is processed as `(a OP b) OP c`, while right-associativity (right-to-left) means it is interpreted as `a OP (b OP c)`. Assignment operators are right-associative, so you can write:

```
a = b = 5;
```

with the expected result that a and b get the value 5. This is because the assignment operator returns the value that it assigned. First, b is set to 5. Then the a is also set to 5, the return value of `b = 5`, aka right operand of the assignment.

Table

The following table is ordered from highest (20) to lowest (0) precedence.

Precedence	Operator type	Associativity	Individual operators

20	Grouping	n/a	(…)
19	Member Access	left-to-right	… . …
	Computed Member Access	left-to-right	… […]
	new (with argument list)	n/a	new … (…)
18	Function Call	left-to-right	… (…)
	new (without argument list)	right-to-left	new …
17	Postfix Increment	n/a	… ++
	Postfix Decrement	n/a	… --
16	Logical NOT	right-to-left	! …
	Bitwise NOT	right-to-left	~ …
	Unary Plus	right-to-left	+ …
	Unary Negation	right-to-left	- …
	Prefix Increment	right-to-left	++ …
	Prefix Decrement	right-to-left	-- …
	typeof	right-to-left	typeof …
	void	right-to-left	void …
	delete	right-to-left	delete …
15	Exponentiation	right-to-left	… ** …
14	Multiplication	left-to-right	… * …
	Division	left-to-right	… / …
	Remainder	left-to-right	… % …
13	Addition	left-to-right	… + …
	Subtraction	left-to-right	… - …
12	Bitwise Left Shift	left-to-right	… << …
	Bitwise Right Shift	left-to-right	… >> …
	Bitwise Unsigned Right Shift	left-to-right	… >>> …
11	Less Than	left-to-right	… < …
	Less Than Or Equal	left-to-right	… <= …
	Greater Than	left-to-right	… > …

	Greater Than Or Equal	left-to-right	… >= …
	in	left-to-right	… in …
	instanceof	left-to-right	… instanceof …
10	Equality	left-to-right	… == …
	Inequality	left-to-right	… != …
	Strict Equality	left-to-right	… === …
	Strict Inequality	left-to-right	… !== …
9	Bitwise AND	left-to-right	… & …
8	Bitwise XOR	left-to-right	… ^ …
7	Bitwise OR	left-to-right	… | …
6	Logical AND	left-to-right	… && …
5	Logical OR	left-to-right	… || …
4	Conditional	right-to-left	… ? … : …
3	Assignment	right-to-left	… = …
			… += …
			… -= …
			… **= …
			… *= …
			… /= …
			… %= …
			… <<= …
			… >>= …
			… >>>= …
			… &= …
			… ^= …
			… |= …
2	yield	right-to-left	yield …
	yield*	right-to-left	yield* …
1	Spread	n/a	... …

0	Comma / Sequence	left-to-right	... , ...