mdn web docs

Expressions and operators

This chapter documents all the JavaScript language operators, expressions and keywords.

Expressions and operators by category

For an alphabetical listing see the sidebar on the left.

Primary expressions

Basic keywords and general expressions in JavaScript. These expressions have the highest precedence (higher than operators).

`this`

The `this` keyword refers to a special property of an execution context.

Literals

Basic `null`, boolean, number, and string literals.

`[]`

Array initializer/literal syntax.

`{}`

Object initializer/literal syntax.

`function`

The `function` keyword defines a function expression.

`class`

The `class` keyword defines a class expression.

`function*`

The `function*` keyword defines a generator function expression.

`async function`

The `async function` defines an async function expression.

`async function*`

The `async function*` keywords define an async generator function expression.

`/ab+c/i`

Regular expression literal syntax.

`` `string` ``

Template literal syntax.

`( _ )`

Grouping operator.

Left-hand-side expressions

Left values are the destination of an assignment.

Property accessors

Member operators provide access to a property or method of an object (`object.property` and `object["property"]`).

`?.`

The optional chaining operator returns `undefined` instead of causing an error if a reference is nullish (`null` or `undefined`).

`new`

The `new` operator creates an instance of a constructor.

`new.target`

In constructors, `new.target` refers to the constructor that was invoked by `new`.

`import.meta`

An object exposing context-specific metadata to a JavaScript module.

`super`

The `super` keyword calls the parent constructor or allows accessing properties of the parent object.

`import()`

The `import()` syntax allows loading a module asynchronously and dynamically into a potentially non-module environment.

Increment and decrement

Postfix/prefix increment and postfix/prefix decrement operators.

`A++`

Postfix increment operator.

`A--`

Postfix decrement operator.

`++A`

Prefix increment operator.

`--A`

Prefix decrement operator.

Unary operators

A unary operation is an operation with only one operand.

`delete`

The `delete` operator deletes a property from an object.

`void`

The `void` operator evaluates an expression and discards its return value.

`typeof`

The `typeof` operator determines the type of a given object.

`+`

The unary plus operator converts its operand to Number type.

`-`

The unary negation operator converts its operand to Number type and then negates it.

`~`

Bitwise NOT operator.

`!`

Logical NOT operator.

`await`

Pause and resume an async function and wait for the promise's fulfillment/rejection.

Arithmetic operators

Arithmetic operators take numerical values (either literals or variables) as their operands and return a single numerical value.

`**`

Exponentiation operator.

`*`

Multiplication operator.

`/`

Division operator.

`%`

Remainder operator.

`+` (Plus)

Addition operator.

`-`

Subtraction operator.

Relational operators

A comparison operator compares its operands and returns a boolean value based on whether the comparison is true.

`<` (Less than)

Less than operator.

`>` (Greater than)

Greater than operator.

`<=`

Less than or equal operator.

`>=`

Greater than or equal operator.

`instanceof`

The `instanceof` operator determines whether an object is an instance of another object.

`in`

The `in` operator determines whether an object has a given property.

> **Note:** `=>` is not an operator, but the notation for Arrow functions.

Equality operators

The result of evaluating an equality operator is always of type boolean based on whether the comparison is true.

`==`

Equality operator.

`!=`

Inequality operator.

`===`

Strict equality operator.

`!==`

Strict inequality operator.

Bitwise shift operators

Operations to shift all bits of the operand.

`<<`

Bitwise left shift operator.

`>>`

Bitwise right shift operator.

`>>>`

Bitwise unsigned right shift operator.

Binary bitwise operators

Bitwise operators treat their operands as a set of 32 bits (zeros and ones) and return standard JavaScript numerical values.

`&`

Bitwise AND.

`|`

Bitwise OR.

`^`

Bitwise XOR.

Binary logical operators

Logical operators implement boolean (logical) values and have short-circuiting behavior.

`&&`

Logical AND.

`||`

Logical OR.

`??`

Nullish Coalescing Operator.

Conditional (ternary) operator

`(condition ? ifTrue : ifFalse)`

The conditional operator returns one of two values based on the logical value of the condition.

Assignment operators

An assignment operator assigns a value to its left operand based on the value of its right operand.

`=`

Assignment operator.

`*=`

Multiplication assignment.

`/=`

Division assignment.

`%=`

Remainder assignment.

`+=`

Addition assignment.

`-=`

Subtraction assignment

`<<=`

Left shift assignment.

`>>=`

Right shift assignment.

`>>>=`

Unsigned right shift assignment.

`&=`

Bitwise AND assignment.

`^=`

Bitwise XOR assignment.

`|=`

Bitwise OR assignment.

`**=`

Exponentiation assignment.

`&&=`

Logical AND assignment.

`||=`

Logical OR assignment.

`??=`

Nullish coalescing assignment.

`[a, b] = arr`, `{ a, b } = obj`

Destructuring assignment allows you to assign the properties of an array or object to variables using syntax that looks similar to array or object literals.

Yield operators

`yield`

Pause and resume a generator function.

`yield*`

Delegate to another generator function or iterable object.

Spread syntax

`...obj`

Spread syntax allows an iterable, such as an array or string, to be expanded in places where zero or more arguments (for function calls) or elements (for array literals) are expected. In an object literal, the spread syntax enumerates the properties of an object and adds the key-value pairs to the object being created.

Comma operator

`,`

The comma operator allows multiple expressions to be evaluated in a single statement and returns the result of the last expression.

Specifications

Specification
ECMAScript Language Specification # sec-addition-operator-plus
ECMAScript Language Specification # sec-assignment-operators
ECMAScript Language Specification # sec-async-function-definitions
ECMAScript Language Specification # sec-async-generator-function-definitions
ECMAScript Language Specification # prod-BitwiseANDExpression
ECMAScript Language Specification # sec-bitwise-not-operator
ECMAScript Language Specification # prod-BitwiseORExpression
ECMAScript Language Specification # prod-BitwiseXORExpression
ECMAScript Language Specification # sec-class-definitions

Specification
ECMAScript Language Specification # sec-comma-operator
ECMAScript Language Specification # sec-conditional-operator
ECMAScript Language Specification # sec-postfix-decrement-operator
ECMAScript Language Specification # sec-delete-operator
ECMAScript Language Specification # sec-destructuring-assignment
ECMAScript Language Specification # sec-destructuring-binding-patterns
ECMAScript Language Specification # sec-multiplicative-operators
ECMAScript Language Specification # sec-equality-operators
ECMAScript Language Specification # sec-exp-operator
ECMAScript Language Specification # sec-function-definitions
ECMAScript Language Specification # sec-generator-function-definitions
ECMAScript Language Specification # sec-relational-operators
ECMAScript Language Specification # sec-grouping-operator
ECMAScript Language Specification # sec-import-calls
ECMAScript Language Specification # prod-ImportMeta
HTML Standard # hostgetimportmetaproperties
ECMAScript Language Specification # sec-postfix-increment-operator
ECMAScript Language Specification # sec-left-shift-operator
ECMAScript Language Specification # prod-LogicalANDExpression

Specification
ECMAScript Language Specification # sec-logical-not-operator
ECMAScript Language Specification # prod-LogicalORExpression
ECMAScript Language Specification # sec-new-operator
ECMAScript Language Specification # sec-built-in-function-objects
ECMAScript Language Specification # sec-null-value
ECMAScript Language Specification # prod-CoalesceExpression
ECMAScript Language Specification # sec-object-initializer
ECMAScript Language Specification # prod-OptionalExpression
ECMAScript Language Specification # sec-property-accessors
ECMAScript Language Specification # sec-signed-right-shift-operator
ECMAScript Language Specification # prod-SpreadElement
ECMAScript Language Specification # prod-ArgumentList
ECMAScript Language Specification # prod-PropertyDefinition
ECMAScript Language Specification # sec-subtraction-operator-minus
ECMAScript Language Specification # sec-super-keyword
ECMAScript Language Specification # sec-this-keyword
ECMAScript Language Specification # sec-typeof-operator
ECMAScript Language Specification # sec-unary-minus-operator
ECMAScript Language Specification # sec-unary-plus-operator

Specification
ECMAScript Language Specification # sec-unsigned-right-shift-operator
ECMAScript Language Specification # sec-void-operator
ECMAScript Language Specification # prod-YieldExpression
ECMAScript Language Specification # sec-generator-function-definitions-runtime-semantics-evaluation

Browser compatibility

Report problems with this compatibility data on GitHub

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android
Addition (`+`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Addition assignment (`x += y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Assignment (`x = y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
`async function` expression	✓ Chrome 55	✓ Edge 15	✓ Firefox 52	✓ Opera 42	✓ Safari 10.1	✓ Chrome Android 55	✓ Firefox for Android 52
`async function*` expression	✓ Chrome 63	✓ Edge 79	✓ Firefox 55	✓ Opera 50	✓ Safari 12	✓ Chrome Android 63	✓ Firefox for Android 55
`await`	✓ Chrome 55	✓ Edge 14	✓ Firefox 52	✓ Opera 42	✓ Safari 10.1	✓ Chrome Android 55	✓ Firefox for Android 52
Use at module top level	✓ Chrome 89	✓ Edge 89	✓ Firefox 89	✓ Opera 75 ⋯	✓ Safari 15	✓ Chrome Android 89	✓ Firefox for Android 89
Bitwise AND (`a & b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise AND assignment (`x &= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise NOT (`~a`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise OR (`a | b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android
Bitwise OR assignment (`x |= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise XOR (`a ^ b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise XOR assignment (`x ^= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
`class`	✓ Chrome 42	✓ Edge 13	✓ Firefox 45	✓ Opera 29	✓ Safari 7	✓ Chrome Android 42	✓ Firefox for Android 45
Comma operator	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Conditional operator (`c ? t : f`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Decrement (`--`)	✓ Chrome 2	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 4	✓ Chrome Android 18	✓ Firefox for Android 4
`delete`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 9	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Destructuring assignment	✓ Chrome 49	✓ Edge 14	✓ Firefox 41 *	✓ Opera 36	✓ Safari 8	✓ Chrome Android 49	✓ Firefox for Android 41 *
Computed property names	✓ Chrome 49	✓ Edge 14	✓ Firefox 41	✓ Opera 36	✓ Safari 10	✓ Chrome Android 49	✓ Firefox for Android 41
Rest in arrays	✓ Chrome 49	✓ Edge 16	✓ Firefox 41	✓ Opera 36	✓ Safari 9.1	✓ Chrome Android 49	✓ Firefox for Android 41
Rest in objects	✓ Chrome 60	✓ Edge 79	✓ Firefox 55	✓ Opera 47	✓ Safari 11.1	✓ Chrome Android 60	✓ Firefox for Android 55
Division (`/`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Division assignment (`x /= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Equality (`a == b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Exponentiation (`**`)	✓ Chrome 52	✓ Edge 14	✓ Firefox 52	✓ Opera 39	✓ Safari 10.1	✓ Chrome Android 52	✓ Firefox for Android 52
Exponentiation assignment (`x **= y`)	✓ Chrome 52	✓ Edge 14	✓ Firefox 52	✓ Opera 39	✓ Safari 10.1	✓ Chrome Android 52	✓ Firefox for Android 52
`function` expression	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Trailing comma in parameters	✓ Chrome 58	✓ Edge 14	✓ Firefox 52	✓ Opera 45	✓ Safari 10	✓ Chrome Android 58	✓ Firefox for Android 52
`function*` expression	✓ Chrome 49	✓ Edge 12	✓ Firefox 26	✓ Opera 36	✓ Safari 10	✓ Chrome Android 49	✓ Firefox for Android 26
Trailing comma in parameters	✓ Chrome 58	✓ Edge 79	✓ Firefox 52	✓ Opera 45	✓ Safari 10	✓ Chrome Android 58	✓ Firefox for Android 52

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android
Greater than (`a > b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Greater than or equal (`a >= b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Grouping operator `()`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
`import`	✓ Chrome 63	✓ Edge 79	✓ Firefox 67	✓ Opera 50	✓ Safari 11.1	✓ Chrome 63 Android	✓ Firefox for Android 67
The `options` parameter ⚠	✓ Chrome 91	✓ Edge 91	✗ Firefox No *	✗ Opera No	✓ Safari 15	✓ Chrome 91 Android	✗ Firefox for Android No *
Available in workers	✓ Chrome 80	✓ Edge 80	✓ Firefox 114	✓ Opera 67	✓ Safari 15	✓ Chrome 80 Android	✓ Firefox for Android 114
`import.meta`	✓ Chrome 64	✓ Edge 79	✓ Firefox 62	✓ Opera 51	✓ Safari 11.1	✓ Chrome 64 Android	✓ Firefox for Android 62
`import.meta.resolve`	✓ Chrome 105	✓ Edge 105	✓ Firefox 106	✓ Opera 91	✓ Safari 16.4	✓ Chrome 105 Android	✓ Firefox for Android 106
`in`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Increment (`++`)	✓ Chrome 2	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 4	✓ Chrome 18 Android	✓ Firefox for Android 4
Inequality (`a != b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
`instanceof`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Bitwise left shift (`a << b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Left shift assignment (`x <<= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Less than (`a < b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Less than or equal (`a <= b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Logical AND (`&&`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome 18 Android	✓ Firefox for Android 4
Logical AND assignment (`x &&= y`)	✓ Chrome 85	✓ Edge 85	✓ Firefox 79	✓ Opera 71	✓ Safari 14	✓ Chrome 85 Android	✓ Firefox for Android 79

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android
Logical NOT (`!`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Logical OR (`||`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Logical OR assignment (`x ||= y`)	✓ Chrome 85	✓ Edge 85	✓ Firefox 79	✓ Opera 71	✓ Safari 14	✓ Chrome Android 85	✓ Firefox for Android 79
Multiplication (`*`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Multiplication assignment (`x *= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
`new`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
`new.target`	✓ Chrome 46	✓ Edge 13	✓ Firefox 41	✓ Opera 33	✓ Safari 11	✓ Chrome Android 46	✓ Firefox for Android 41
`null`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Nullish coalescing operator (`??`)	✓ Chrome 80	✓ Edge 80	✓ Firefox 72	✓ Opera 67	✓ Safari 13.1	✓ Chrome Android 80	✓ Firefox for Android 79
Nullish coalescing assignment (`x ??= y`)	✓ Chrome 85	✓ Edge 85	✓ Firefox 79	✓ Opera 71	✓ Safari 14	✓ Chrome Android 85	✓ Firefox for Android 79
Object initializer	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Computed property names	✓ Chrome 47	✓ Edge 12	✓ Firefox 34	✓ Opera 34	✓ Safari 8	✓ Chrome Android 47	✓ Firefox for Android 34
Shorthand method names	✓ Chrome 47	✓ Edge 12	✓ Firefox 34	✓ Opera 34	✓ Safari 9	✓ Chrome Android 47	✓ Firefox for Android 34
Shorthand property names	✓ Chrome 47	✓ Edge 12	✓ Firefox 33	✓ Opera 34	✓ Safari 9	✓ Chrome Android 47	✓ Firefox for Android 33
Spread properties	✓ Chrome 60	✓ Edge 79	✓ Firefox 55	✓ Opera 47	✓ Safari 11.1	✓ Chrome Android 60	✓ Firefox for Android 55
Optional chaining operator (`?.`)	✓ Chrome 80	✓ Edge 80	✓ Firefox 74	✓ Opera 67	✓ Safari 13.1	✓ Chrome Android 80	✓ Firefox for Android 79
Property accessors	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Remainder (`%`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android
Remainder assignment (`x %= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise right shift (`a >> b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Right shift assignment (`x >>= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Spread syntax (...)	✓ Chrome 46	✓ Edge 12	✓ Firefox 16	✓ Opera 37	✓ Safari 8	✓ Chrome Android 46	✓ Firefox for Android 16
Spread in array literals	✓ Chrome 46	✓ Edge 12	✓ Firefox 16	✓ Opera 37	✓ Safari 8	✓ Chrome Android 46	✓ Firefox for Android 16
Spread in function calls	✓ Chrome 46	✓ Edge 12	✓ Firefox 27	✓ Opera 37	✓ Safari 8	✓ Chrome Android 46	✓ Firefox for Android 27
Spread in object literals	✓ Chrome 60	✓ Edge 79	✓ Firefox 55	✓ Opera 47	✓ Safari 11.1	✓ Chrome Android 60	✓ Firefox for Android 55
Strict equality (`a === b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Strict inequality (`a !== b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Subtraction (`-`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Subtraction assignment (`x -= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
`super`	✓ Chrome 42	✓ Edge 13	✓ Firefox 45	✓ Opera 29	✓ Safari 7	✓ Chrome Android 42	✓ Firefox for Android 45
`this`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 9.5	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
`typeof`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Unary negation (`-`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Unary plus (`+`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Bitwise unsigned right shift (`a >>> b`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4
Unsigned right shift assignment (`x >>>= y`)	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 3	✓ Safari 1	✓ Chrome Android 18	✓ Firefox for Android 4

	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android
`void`	✓ Chrome 1	✓ Edge 12	✓ Firefox 1	✓ Opera 4	✓ Safari 3.1	✓ Chrome 18 Android	✓ Firefox for 4 Android
`yield`	✓ Chrome 39	✓ Edge 12	✓ Firefox 26 ✱	✓ Opera 26	✓ Safari 10	✓ Chrome 39 Android	✓ Firefox 26 ✱ for Android
`yield*`	✓ Chrome 39	✓ Edge 12	✓ Firefox 27 ✱	✓ Opera 26	✓ Safari 10	✓ Chrome 39 Android	✓ Firefox 27 ✱ for Android

Tip: you can click/tap on a cell for more information.

✓ Full support ◆ Partial support ⊗ No support ⚠ Non-standard. Check cross-browser support before using. ✱ See implementation notes.

⚑ User must explicitly enable this feature. ··· Has more compatibility info.

See also

- Operator precedence

This page was last modified on Apr 5, 2023 by MDN contributors.