**return**

The **return** statement ends function execution and specifies a value to be returned to the function caller.

[**Try it**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#try_it)

[**Syntax**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#syntax)

return [expression]

Copy to Clipboard

expression

The expression whose value is to be returned. If omitted, undefined is returned instead.

[**Description**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#description)

When a return statement is used in a function body, the execution of the function is stopped. If specified, a given value is returned to the function caller. For example, the following function returns the square of its argument, x, where x is a number.

function square(x) {

return x \* x;

}

const demo = square(3);

// demo will equal 9

Copy to Clipboard

If the value is omitted, undefined is returned instead.

The following return statements all break the function execution:

return;

return true;

return false;

return x;

return x + y / 3;

Copy to Clipboard

[**Automatic Semicolon Insertion**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#automatic_semicolon_insertion)

The return statement is affected by [automatic semicolon insertion (ASI)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Lexical_grammar#automatic_semicolon_insertion). No line terminator is allowed between the return keyword and the expression.

return

a + b;

is transformed by ASI into:

return;

a + b;

Copy to Clipboard

The console will warn "unreachable code after return statement".

**Note:** Starting with Firefox 40, a warning is shown in the console if unreachable code is found after a return statement.

To avoid this problem (to prevent ASI), you could use parentheses:

return (

a + b

);

Copy to Clipboard

[**Examples**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#examples)

[**Interrupt a function**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#interrupt_a_function)

A function immediately stops at the point where return is called.

function counter() {

for (let count = 1; ; count++) { // infinite loop

console.log(`${count}A`); // until 5

if (count === 5) {

return;

}

console.log(`${count}B`); // until 4

}

console.log(`${count}C`); // never appears

}

counter();

// Logs:

// 1A

// 1B

// 2A

// 2B

// 3A

// 3B

// 4A

// 4B

// 5A

Copy to Clipboard

[**Returning a function**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/return#returning_a_function)

See also the article about [Closures](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Closures).

function magic() {

return function calc(x) { return x \* 42; };

}

const answer = magic();

answer(1337); // 56154

Copy to Clipboard