

JavaScript Array Methods

A developer quick reference for common array operations, return values, and pitfalls.

Choose the method for the job

Method	Use it when	Returns / mutates
map(fn)	Transform every item	New array
filter(fn)	Keep items matching a test	New array
find(fn)	Get the first matching item	Item or undefined
some(fn)	Check whether any item matches	Boolean
every(fn)	Check whether all items match	Boolean
reduce(fn, initial)	Combine items into one result	Accumulator result
flatMap(fn)	Map and flatten one level	New array
sort(compareFn)	Sort items	Mutates original array
slice(start, end)	Copy a section	New array
includes(value)	Check for a value	Boolean
forEach(fn)	Run an action per item	Undefined

Examples

Goal	Example
Double numbers	<code>const doubled = nums.map(n => n * 2);</code>
Keep active users	<code>const active = users.filter(u => u.active);</code>
Find by ID	<code>const item = items.find(x => x.id === id);</code>
Sum values	<code>const total = nums.reduce((sum, n) => sum + n, 0);</code>
Sort numbers ascending	<code>const sorted = [...nums].sort((a, b) => a - b);</code>
Group by key (modern JS)	Use Map or reduce; choose a clear, well-tested approach.

Common pitfalls

- [] Use an explicit numeric comparator for numeric sort.
- [] Remember `sort()` mutates; copy first if immutability matters.
- [] `find()` may return undefined; handle the missing case.
- [] Avoid side effects inside `map()` when `forEach()` is clearer.
- [] Use strict equality or an intentional matching function.