@wordpress/is-shallow-equal

- Usage
- Rationale
- Benchmarks
- Contributing to this package

A function for performing a shallow comparison between two objects or arrays. Two values have shallow equality when all of their members are strictly equal to the corresponding member of the other.

Usage

The default export of `@wordpress/is-shallow-equal` is a function which accepts two objects or arrays:

```
import isShallowEqual from '@wordpress/is-shallow-equal';

isShallowEqual( { a: 1 }, { a: 1, b: 2 } );
// ⇒ false

isShallowEqual( { a: 1 }, { a: 1 } );
// ⇒ true

isShallowEqual( [ 1 ], [ 1, 2 ] );
// ⇒ false

isShallowEqual( [ 1 ], [ 1 ] );
// ⇒ true
```

You can import a specific implementation if you already know the types of values you are working with:

```
import { isShallowEqualArrays } from '@wordpress/is-shallow-equal';
import { isShallowEqualObjects } from '@wordpress/is-shallow-equal';
```

Shallow comparison differs from deep comparison by the fact that it compares members from each as being strictly equal to the other, meaning that arrays and objects will be compared by their *references*, not by their values (see also *Object Equality in JavaScript.*) In situations where nested objects must be compared by value, consider using `fast-deep-equal` instead.

```
import isShallowEqual from '@wordpress/is-shallow-equal';
import fastDeepEqual from 'fast-deep-equal/es6'; // deep comparison

let object = { a: 1 };

isShallowEqual( [ { a: 1 } ], [ { a: 1 } ] );
// ⇒ false

fastDeepEqual( [ { a: 1 } ], [ { a: 1 } ] );
// ⇒ true

isShallowEqual( [ object ], [ object ] );
// ⇒ true
```

Rationale

Shallow equality utilities are already a dime a dozen. Since these operations are often at the core of critical hot code paths, the WordPress contributors had specific requirements that were found to only be partially satisfied by existing solutions.

In particular, it should…

1. …consider non-primitive yet referentially-equal members values as equal.
   - Eliminates `is-equal-shallow` as an option.
2. …offer a single function through which to interface, regardless of value type.
   - Eliminates `shallow-equal` as an option.
3. …be barebones; only providing the basic functionality of shallow equality.
   - Eliminates `shallow-equals` as an option.
4. …anticipate and optimize for referential sameness as equal.
   - Eliminates `is-equal-shallow` and `shallow-equals` as options.
5. …be intended for use in non-Facebook projects.
   - Eliminates `fbjs/lib/shallowEqual` as an option.

6. …be the most performant implementation.
    - See Benchmarks.

Benchmarks

The following results are historical: they were produced under Node v10.15.3 (LTS) on a MacBook Pro (Late 2016) 2.9 GHz Intel Core i7, and the script that generated them has since been removed.

> `@wordpress/is-shallow-equal (type specific) (object, equal) x 4,519,009 ops/sec ±1.09% (90 runs sampled)` > `@wordpress/is-shallow-equal (type specific) (object, same) x 795,527,700 ops/sec ±0.24% (93 runs sampled)` > `@wordpress/is-shallow-equal (type specific) (object, unequal) x 4,841,640 ops/sec ±0.94% (93 runs sampled)` > `@wordpress/is-shallow-equal (type specific) (array, equal) x 106,393,795 ops/sec ±0.16% (94 runs sampled)` > `@wordpress/is-shallow-equal (type specific) (array, same) x 800,741,511 ops/sec ±0.22% (95 runs sampled)` > `@wordpress/is-shallow-equal (type specific) (array, unequal) x 49,178,977 ops/sec ±1.99% (82 runs sampled)`
>
> `@wordpress/is-shallow-equal (object, equal) x 4,449,367 ops/sec ±0.31% (91 runs sampled)` > `@wordpress/is-shallow-equal (object, same) x 796,677,179 ops/sec ±0.23% (94 runs sampled)` > `@wordpress/is-shallow-equal (object, unequal) x 4,989,529 ops/sec ±0.30% (91 runs sampled)` > `@wordpress/is-shallow-equal (array, equal) x 44,840,546 ops/sec ±1.18% (89 runs sampled)` > `@wordpress/is-shallow-equal (array, same) x 794,344,723 ops/sec ±0.24% (91 runs sampled)` > `@wordpress/is-shallow-equal (array, unequal) x 49,860,115 ops/sec ±1.73% (85 runs sampled)`
>
> `shallowequal (object, equal) x 3,702,126 ops/sec ±0.87% (92 runs sampled)` > `shallowequal (object, same) x 796,649,597 ops/sec ±0.21% (92 runs sampled)` > `shallowequal (object, unequal) x 4,027,885 ops/sec ±0.31% (96 runs sampled)` > `shallowequal (array, equal) x 1,684,977 ops/sec ±0.37% (94 runs sampled)` > `shallowequal (array, same) x 794,287,091 ops/sec ±0.26% (91 runs sampled)` > `shallowequal (array, unequal) x 1,738,554 ops/sec ±0.29% (91 runs sampled)`
>
> `shallow-equal (type specific) (object, equal) x 4,669,656 ops/sec ±0.34% (92 runs sampled)` > `shallow-equal (type specific) (object, same) x 799,610,214 ops/sec ±0.20% (95 runs sampled)` > `shallow-equal (type specific) (object, unequal) x 4,908,591 ops/sec ±0.49% (93 runs sampled)` > `shallow-equal (type specific) (array, equal) x 104,711,254 ops/sec ±0.65% (91 runs sampled)` > `shallow-equal (type specific) (array, same) x 798,454,281 ops/sec ±0.29% (94 runs sampled)` > `shallow-equal (type specific) (array, unequal) x 48,764,338 ops/sec ±1.48% (84 runs sampled)`

```
is-equal-shallow (object, equal) x 5,068,750 ops/sec ±0.28% (92 runs sampled) >
is-equal-shallow (object, same) x 17,231,997 ops/sec ±0.42% (92 runs sampled) >
is-equal-shallow (object, unequal) x 5,524,878 ops/sec ±0.41% (92 runs sampled)
>is-equal-shallow (array, equal) x 1,067,063 ops/sec ±0.40% (92 runs sampled)
>is-equal-shallow (array, same) x 1,074,356 ops/sec ±0.20% (94 runs sampled) >
is-equal-shallow (array, unequal) x 1,758,859 ops/sec ±0.44% (92 runs sampled)

shallow-equals (object, equal) x 8,380,550 ops/sec ±0.31% (90 runs sampled) >
shallow-equals (object, same) x 27,583,073 ops/sec ±0.60% (91 runs sampled) >
shallow-equals (object, unequal) x 8,954,268 ops/sec ±0.71% (92 runs sampled) >
shallow-equals (array, equal) x 104,437,640 ops/sec ±0.22% (96 runs sampled) >
shallow-equals (array, same) x 141,850,542 ops/sec ±0.25% (93 runs sampled) >
shallow-equals (array, unequal) x 47,964,211 ops/sec ±1.51% (84 runs sampled)

fbjs/lib/shallowEqual (object, equal) x 3,366,709 ops/sec ±0.35% (93 runs
sampled) >fbjs/lib/shallowEqual (object, same) x 794,825,194 ops/sec ±0.24%
(94 runs sampled) >fbjs/lib/shallowEqual (object, unequal) x 3,612,268 ops/sec
±0.37% (94 runs sampled) >fbjs/lib/shallowEqual (array, equal) x 1,613,800
ops/sec ±0.23% (90 runs sampled) >fbjs/lib/shallowEqual (array, same) x
794,861,384 ops/sec ±0.24% (93 runs sampled) >fbjs/lib/shallowEqual (array,
unequal) x 1,648,398 ops/sec ±0.77% (92 runs sampled)
```

Contributing to this package

This is an individual package that's part of the Gutenberg project. The project is organized as a monorepo. It's made up of multiple self-contained software packages, each with a specific purpose.

The packages in this monorepo are published to npm and used by WordPress as well as other software projects.

To find out more about contributing to this package or Gutenberg as a whole, please read the project's main contributor guide.

□□□ #2
□ □□□ □□□ 2026-10-31 18:19:25 UTC
□ □□□ □□□ 2026-10-31 18:19:26 UTC