







 lukeed /  
clsx










[Code](#) [Issues](#) 3 [Pull requests](#) 2 [Actions](#) [Security](#) [Insights](#)



A tiny (239B) utility for constructing `className` strings conditionally.

 MIT license

 7.1k stars  123 forks  17 watching  1 Branch  14 Tags  Activity

 Public repository

master	1 Branch	14 Tags	Go to file	Go to file	+	Add file	Code	...
lukeed	chore: add licenses badge	✓	2 months ago					
.github	fix(ci): replace nyc -> c8	3 months ago						
bench	feat: add clsx/lite module	3 months ago						
bin	feat: add clsx/lite module	3 months ago						
src	feat: add clsx/lite module	3 months ago						
test	feat: add clsx/lite module	3 months ago						
.editorconfig	initial commit	6 years ago						
.gitignore	chore: import Action script	4 years ago						
clsx.d.mts	break: include "exports" map w/ "types" ...	8 months ago						
clsx.d.ts	break: include "exports" map w/ "types" ...	8 months ago						
license	initial commit	6 years ago						
package.json	2.1.0	3 months ago						
readme.md	chore: add licenses badge	2 months ago						

README MIT license

# clsx

CI passing coverage 100% licenses MIT

A tiny (239B) utility for constructing `className` strings conditionally.  
Also serves as a [faster](#) & smaller drop-in replacement for the `classnames` module.

This module is available in three formats:

- **ES Module:** `dist/clsx.mjs`
- **CommonJS:** `dist/clsx.js`
- **UMD:** `dist/clsx.min.js`

## Install

```
$ npm install --save clsx
```

## Usage

```
import clsx from 'clsx';
// or
import { clsx } from 'clsx';

// Strings (variadic)
clsx('foo', true && 'bar', 'baz');
//=> 'foo bar baz'

// Objects
clsx({ foo:true, bar:false, baz:true })
```

```
    // => 'foo baz'

    // Objects (variadic)
    clsx({ foo:true }, { bar:false }, null, { '--foobar':'hello' });
    // => 'foo --foobar'

    // Arrays
    clsx(['foo', 0, false, 'bar']);
    // => 'foo bar'

    // Arrays (variadic)
    clsx(['foo'], ['', 0, false, 'bar'], [['baz', [['hello'], 'there']]]);
    // => 'foo bar baz hello there'

    // Kitchen sink (with nesting)
    clsx('foo', [1 && 'bar', { baz:false, bat:null }, ['hello', ['world']]], 'cya');
    // => 'foo bar hello world cya'
```

## API

### clsx(...input)

Returns: String

#### input

Type: Mixed

The `clsx` function can take **any** number of arguments, each of which can be an Object, Array, Boolean, or String.

**Important:** Any falsy values are discarded!  
Standalone Boolean values are discarded as well.

```
clsx(true, false, '', null, undefined, 0, NaN);
// => ''
```



## Modes

There are multiple "versions" of `clsx` available, which allows you to bring only the functionality you need!

### clsx

**Size (gzip):** 239 bytes

**Availability:** CommonJS, ES Module, UMD

The default `clsx` module; see [API](#) for info.

```
import { clsx } from 'clsx';
// or
import clsx from 'clsx';
```



### clsx/lite

**Size (gzip):** 140 bytes

**Availability:** CommonJS, ES Module

**CAUTION:** Accepts **ONLY** string arguments!

Ideal for applications that **only** use the string-builder pattern.

Any non-string arguments are ignored!

```
import { clsx } from 'clsx/lite';
// or
import clsx from 'clsx/lite';

// string
clsx('hello', true && 'foo', false && 'bar');
// => "hello foo"

// NOTE: Any non-string input(s) ignored
clsx({ foo: true });
//=> ""
```



## Benchmarks

For snapshots of cross-browser results, check out the [bench](#) directory~!

## Support

All versions of Node.js are supported.

All browsers that support [Array.isArray](#) are supported (IE9+).

**Note:** For IE8 support and older, please install `clsx@1.0.x` and beware of [#17](#).

## Tailwind Support

Here some additional (optional) steps to enable classes autocompletion using `clsx` with Tailwind CSS.

### ► Visual Studio Code

You may find the [clsx/lite](#) module useful within Tailwind contexts. This is especially true if/when your application only composes classes in this pattern:

```
clsx('text-base', props.active && 'text-primary', props.className);
```



## Related

- [obj-str](#) - A smaller (96B) and similiar utility that only works with Objects.

## License

MIT © [Luke Edwards](#)

### Releases 11

 **v2.1.0** Latest  
on Dec 29, 2023

[+ 10 releases](#)

### Sponsor this project



lukeed Luke Edwards

 Sponsor

---

[Learn more about GitHub Sponsors](#)

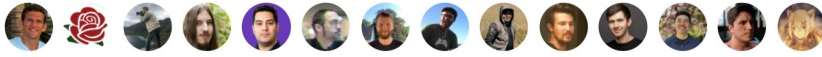
---

**Used by** 2.8m




---

**Contributors** 14




---

**Languages**

● JavaScript 97.6%
 ● TypeScript 2.4%