Webpack CSS Modules: Breaking Change in `css-loader` **v7—Watch Out for** `namedExport`**!**

Migrating to `css-loader@7`? Here's what you *must* know - or if you're migrating from Vite

Kabui, Charles

2025-04-07

Table of Contents

🔗 Read at [ToKnow.ai](#)

TL;DR:

If you're upgrading or migrating from Vite and want to keep your old import style:

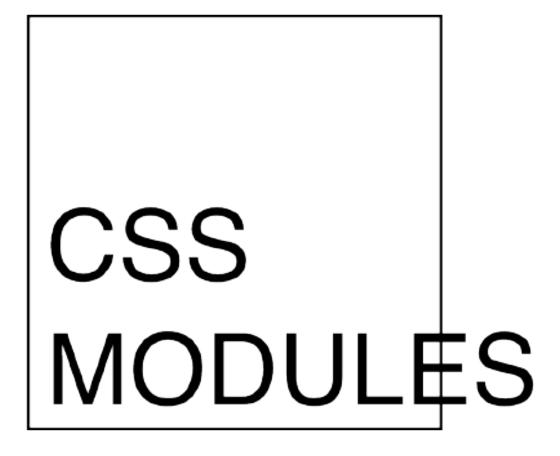**Just set `namedExport: false` in your `css-loader` config.**

Figure 1: Webpack CSS Modules

If you're upgrading from `css-loader` v6 to **v7.0.0** (released April 4, 2024), there's a **breaking change** that can cause silent failures in your builds—particularly when importing CSS modules like this:

```
import css from './styles.module.css'; //  This breaks in v7 by default
```

This pattern worked in **v6**, and is also common in tools like **Vite**, which default to default ex-

ports for CSS modules. But in `css-loader@7`, **the default behavior has changed**. Named exports are now **enabled by default**, which breaks that familiar syntax.

Fix it by explicitly setting:
`namedExport: false`

Here's how your `webpack.config.js` should look:

```
module.exports = {
  module: {
    rules: [
      {
        test: /\.css$/i,
        loader: 'css-loader',
        options: {
          modules: {
            namedExport: false, // <-- This is the key fix
            exportLocalsConvention: 'as-is',
            // or:
            // exportLocalsConvention: 'camel-case-only'
          },
        },
      },
    ],
  },
};
```

Why This Matters:

In `css-loader@7`, if `namedExport` is `true` (the new default), the import syntax must change to:

```
import * as css from './styles.module.css'; //  Works in v7 with namedExport: true
```

Without this change, you'll run into confusing issues where your CSS imports are `undefined`.

Reference:

- [css-loader Changelog - v7.0.0 (Breaking Changes)](#)

- [css-loader Docs: `modules.namedExport`](#)