## **示例**

例如，你可以使用 loader 告诉 webpack 如何加载 CSS 文件，或者如何将 TypeScript 转为 JavaScript。为此，首先安装相对应的 loader：

npm install --save-dev css-loader

npm install --save-dev ts-loader

然后指示 webpack 对每个 .css 使用 [css-loader](https://www.webpackjs.com/loaders/css-loader)，以及对所有 .ts 文件使用 [ts-loader](https://github.com/TypeStrong/ts-loader)：

**webpack.config.js**

module.exports = {

module: {

rules: [

{ test: /\.css$/, use: 'css-loader' },

{ test: /\.ts$/, use: 'ts-loader' }

]

}};

## **使用 loader**

在你的应用程序中，有三种使用 loader 的方式：

* [配置](https://www.webpackjs.com/concepts/loaders/" \l "configuration)（推荐）：在 webpack.config.js 文件中指定 loader。
* [内联](https://www.webpackjs.com/concepts/loaders/" \l "inline)：在每个 import 语句中显式指定 loader。
* [CLI](https://www.webpackjs.com/concepts/loaders/" \l "cli)：在 shell 命令中指定它们。

### **配置[Configuration]**

[module.rules](https://www.webpackjs.com/configuration/module/" \l "module-rules) 允许你在 webpack 配置中指定多个 loader。

module: {

rules: [

{

test: /\.css$/,

use: [

{ loader: ['style-loader'](https://www.webpackjs.com/loaders/style-loader) },

{

loader: ['css-loader'](https://www.webpackjs.com/loaders/css-loader),

options: {

modules: true

}

}

]

}

]

}

## **loader 特性**

loader 支持链式传递。能够对资源使用流水线(pipeline)。一组链式的 loader 将按照相反的顺序执行。

## **解析 loader**

loader 遵循标准的[模块解析](https://www.webpackjs.com/concepts/module-resolution/)。

按照约定，loader 通常被命名为 xxx-loader（例如 json-loader）。