页面性能：静态资源内联与压缩

静态资源内联

Webpack 的内联功能允许您将静态资源（如图像，字体或 CSS）内联到您的代码中，而无需将它们单独打包。这样做的好处是可以减少网络请求，提高应用程序的性能。

- 减少网络请求次数；
- 有利于缓存；
- 防止资源加载失败

内联资源

- 内联图片 （最常见）
- 内联 CSS
- 内联 html

Webpack5 资源模块

资源模块(asset module)是一种模块类型，它允许使用资源文件（字体，图标等）而无需配置额外 loader。

在 webpack 5 之前，通常使用：

- raw-loader 将文件导入为字符串
- url-loader 将文件作为 data URI 内联到 bundle 中
- file-loader 将文件发送到输出目录

资源模块类型(asset module type)，通过添加 4 种新的模块类型，来替换所有这些 loader：

- asset/resource 发送一个单独的文件并导出 URL。之前通过使用 file-loader 实现。
- asset/inline 导出一个资源的 data URI。之前通过使用 url-loader 实现。
- asset/source 导出资源的源代码。之前通过使用 raw-loader 实现。
- asset 在导出一个 data URI 和发送一个单独的文件之间自动选择。之前通过使用 url-loader，并且配置资源体积限制实现。

raw-loader 将文件导入为字符串
url-loader 将文件作为 data URI 内联到 bundle 中
file-loader 将文件发送到输出目录
资源模块类型(asset module type)，通过添加 4 种新的模块类型，来替换所有这些 loader：

asset/resource 发送一个单独的文件并导出 URL。之前通过使用 file-loader 实现。
asset/inline 导出一个资源的 data URI。之前通过使用 url-loader 实现。
asset/source 导出资源的源代码。之前通过使用 raw-loader 实现。
asset 在导出一个 data URI 和发送一个单独的文件之间自动选择。之前通过使用 url-loader，并且配置资源体积限制实现。

图片压缩

JavaScript

```
npm install image-webpack-loader
```

config/image-inline.js

JavaScript

```
module.exports = (base) => {

  let loader = base.module.rules.find(
    (v) => v.test.toString() === "/\\.(png|jpe?g|gif|webp)(\\?.*)?$/"
```

```
  );
  Object.assign(loader, {
    parser: {
      dataUrlCondition: {
        maxSize: 30 * 1024 // 4kb
      }
    },
    use: [
      {
        loader: 'image-webpack-loader',
        options: {
          mozjpeg: {
            progressive: true,
          },
          // optipng.enabled: false will disable optipng
          optipng: {
            enabled: false,
          },
          pngquant: {
            quality: [0.65, 0.90],
            speed: 4
          },
          gifsicle: {
            interlaced: false,
          },
          // the webp option will enable WEBP
          webp: {
            quality: 75
          }
        }
      },
    ]
  },);

  return base;
};
```