images

If you want to use a cloud provider to optimize images instead of using the Next.js built-in Image Optimization API, you can configure `next.config.js` with the following:

next.config.js

```js
module.exports = {
  images: {
    loader: 'custom',
    loaderFile: './my/image/loader.js',
  },
};
```

This `loaderFile` must point to a file relative to the root of your Next.js application. The file must export a default function that returns a string, for example:

```js
export default function myImageLoader({ src, width, quality }) {
  return `https://example.com/${src}?w=${width}&q=${quality || 75}`;
}
```

Alternatively, you can use the `loader` prop to pass the function to each instance of `next/image`.

Example Loader Configuration

- Akamai
- Cloudinary
- Cloudflare
- Contentful
- Fastly

- Gumlet
- ImageEngine
- Imgix
- Thumbor

Akamai

```js
// Docs: https://techdocs.akamai.com/ivm/reference/test-images-on-demand
export default function akamaiLoader({ src, width, quality }) {
  return `https://example.com/${src}?imwidth=${width}`;
}
```

Cloudinary

```js
// Demo: https://res.cloudinary.com/demo/image/upload/w_300,c_limit,q_auto/turtles.jpg
export default function cloudinaryLoader({ src, width, quality }) {
  const params = ['f_auto', 'c_limit', `w_${width}`, `q_${quality || 'auto'}`];
  return `https://example.com/${params.join(',')}${src}`;
}
```

Cloudflare

```js
// Docs: https://developers.cloudflare.com/images/url-format
export default function cloudflareLoader({ src, width, quality }) {
  const params = [`width=${width}`, `quality=${quality || 75}`, 'format=auto'];
  return `https://example.com/cdn-cgi/image/${params.join(',')}/${src}`;
}
```

Contentful

```js
// Docs: https://www.contentful.com/developers/docs/references/images-api/
export default function contentfulLoader({ src, quality, width }) {
  const url = new URL(`https://example.com${src}`);
  url.searchParams.set('fm', 'webp');
  url.searchParams.set('w', width.toString());
  url.searchParams.set('q', quality.toString() || '75');
  return url.href;
}
```

Fastly

```js
// Docs: https://developer.fastly.com/reference/io/
export default function fastlyLoader({ src, width, quality }) {
  const url = new URL(`https://example.com${src}`);
  url.searchParams.set('auto', 'webp');
  url.searchParams.set('width', width.toString());
  url.searchParams.set('quality', quality.toString() || '75');
  return url.href;
}
```

Gumlet

```js
// Docs: https://docs.gumlet.com/reference/image-transform-size
export default function gumletLoader({ src, width, quality }) {
  const url = new URL(`https://example.com${src}`);
  url.searchParams.set('format', 'auto');
  url.searchParams.set('w', width.toString());
  url.searchParams.set('q', quality.toString() || '75');
  return url.href;
}
```

ImageEngine

```js
// Docs: https://support.imageengine.io/hc/en-us/articles/360058880672-Directives
export default function imageengineLoader({ src, width, quality }) {
  const compression = 100 - (quality || 50)
  const params = [`w_${width}`, `cmpr_${compression}`)]
  return `https://example.com${src}?imgeng=/${params.join('/')`
}
```

Imgix

```js
// Demo: https://static.imgix.net/daisy.png?format=auto&fit=max&w=300
export default function imgixLoader({ src, width, quality }) {
  const url = new URL(`https://example.com${src}`);
  const params = url.searchParams;
  params.set('auto', params.getAll('auto').join(',') || 'format');
  params.set('fit', params.get('fit') || 'max');
  params.set('w', params.get('w') || width.toString());
  params.set('q', quality.toString() || '50');
  return url.href;
}
```

Thumbor

```
// Docs: https://thumbor.readthedocs.io/en/latest/
export default function thumborLoader({ src, width, quality }) {
  const params = [`${width}x0`, `filters:quality(${quality || 75})`];
  return `https://example.com${params.join('/')}${src}`;
}
```