AMP

▶ **Examples**

With Next.js you can turn any React page into an AMP page, with minimal config, and without leaving React.

You can read more about AMP in the official amp.dev ↗ site.

Enabling AMP

To enable AMP support for a page, and to learn more about the different AMP configs, read the API documentation for `next/amp`.

Caveats

- Only CSS-in-JS is supported. CSS Modules aren't supported by AMP pages at the moment. You can contribute CSS Modules support to Next.js ↗ .

Adding AMP Components

The AMP community provides many components ↗ to make AMP pages more interactive. Next.js will automatically import all components used on a page and there is no need to manually import AMP component scripts:

```
export const config = { amp: true };

```

```
function MyAmpPage() {
  const date = new Date();

  return (
    <div>
      <p>Some time: {date.toJSON()}</p>
      <amp-timeago
        width="0"
        height="15"
        datetime={date.toJSON()}
        layout="responsive"
      >
        .
      </amp-timeago>
    </div>
  );
}

export default MyAmpPage;
```

The above example uses the `amp-timeago` component.

By default, the latest version of a component is always imported. If you want to customize the version, you can use `next/head`, as in the following example:

```
import Head from 'next/head';

export const config = { amp: true };

function MyAmpPage() {
  const date = new Date();

  return (
    <div>
      <Head>
        <script
          async
          key="amp-timeago"
          custom-element="amp-timeago"
          src="https://cdn.ampproject.org/v0/amp-timeago-0.1.js"
        />
      </Head>

      <p>Some time: {date.toJSON()}</p>
      <amp-timeago
        width="0"
        height="15"
        datetime={date.toJSON()}
        layout="responsive"
      >
        .
      </amp-timeago>
    </div>
  );
}

export default MyAmpPage;
```

AMP Validation

AMP pages are automatically validated with amphtml-validator during development. Errors and warnings will appear in the terminal where you started Next.js.

Pages are also validated during Static HTML export and any warnings / errors will be printed to the terminal. Any AMP errors will cause the export to exit with status code `1` because the export is not valid AMP.

Custom Validators

You can set up custom AMP validator in `next.config.js` as shown below:

```
module.exports = {
  amp: {
    validator: './custom_validator.js',
  },
};
```

Skip AMP Validation

To turn off AMP validation add the following code to `next.config.js`

```
experimental: {
  amp: {
    skipValidation: true;
  }
}
```

AMP in Static HTML Export

When using Static HTML export statically prerender pages, Next.js will detect if the page supports AMP and change the exporting behavior based on that.

For example, the hybrid AMP page `pages/about.js` would output:

- `out/about.html` - HTML page with client-side React runtime

- `out/about.amp.html` - AMP page

And if `pages/about.js` is an AMP-only page, then it would output:

- `out/about.html` - Optimized AMP page

Next.js will automatically insert a link to the AMP version of your page in the HTML version, so you don't have to, like so:

```
<link rel="amphtml" href="/about.amp.html" />
```

And the AMP version of your page will include a link to the HTML page:

```
<link rel="canonical" href="/about" />
```

When `trailingSlash` is enabled the exported pages for `pages/about.js` would be:

- `out/about/index.html` - HTML page
- `out/about.amp/index.html` - AMP page

TypeScript

AMP currently doesn't have built-in types for TypeScript, but it's in their roadmap (#13791).

As a workaround you can manually create a file called `amp.d.ts` inside your project and add these custom types.