Version 11

To upgrade to version 11, run the following command:

Terminal

```
npm install next@11

yarn add next@11
```

Webpack 5

Webpack 5 is now the default for all Next.js applications. If you did not have a custom webpack configuration, your application is already using webpack 5. If you do have a custom webpack configuration, you can refer to the Next.js webpack 5 documentation for upgrade guidance.

Cleaning the `distDir` is now a default

The build output directory (defaults to `.next`) is now cleared by default except for the Next.js caches. You can refer to the cleaning `distDir` RFC for more information.

If your application was relying on this behavior previously you can disable the new default behavior by adding the `cleanDistDir: false` flag in `next.config.js`.

`PORT` is now supported for `next dev` and `next start`

Next.js 11 supports the `PORT` environment variable to set the port the application runs on. Using `-p`/`--port` is still recommended but if you were prohibited from using `-p` in any way you can now use `PORT` as an alternative:

Example:

```
PORT=4000 next start
```

`next.config.js` customization to import images

Next.js 11 supports static image imports with `next/image`. This new feature relies on being able to process image imports. If you previously added the `next-images` or `next-optimized-images` packages you can either move to the new built-in support using `next/image` or disable the feature:

next.config.js

```js
module.exports = {
  images: {
    disableStaticImages: true,
  },
};
```

Remove `super.componentDidCatch()` from `pages/_app.js`

The `next/app` component's `componentDidCatch` was deprecated in Next.js 9 as it's no longer needed and has since been a no-op. In Next.js 11, it was removed.

If your `pages/_app.js` has a custom `componentDidCatch` method you can remove `super.componentDidCatch` as it is no longer needed.

Remove `Container` from `pages/_app.js`

This export was deprecated in Next.js 9 as it's no longer needed and has since been a no-op with a warning during development. In Next.js 11 it was removed.

If your `pages/_app.js` imports `Container` from `next/app` you can remove `Container` as it was removed. Learn more in the documentation.

Remove `props.url` usage from page components

This property was deprecated in Next.js 4 and has since shown a warning during development. With the introduction of `getStaticProps` / `getServerSideProps` these methods already disallowed the usage of `props.url`. In Next.js 11, it was removed completely.

You can learn more in the documentation.

Remove `unsized` property on `next/image`

The `unsized` property on `next/image` was deprecated in Next.js 10.0.1. You can use `layout="fill"` instead. In Next.js 11 `unsized` was removed.

Remove `modules` property on `next/dynamic`

The `modules` and `render` option for `next/dynamic` were deprecated in Next.js 9.5. This was done in order to make the `next/dynamic` API closer to `React.lazy`. In Next.js 11, the `modules` and `render` options were removed.

This option hasn't been mentioned in the documentation since Next.js 8 so it's less likely that your application is using it.

If your application does use `modules` and `render` you can refer to the documentation.

Remove `Head.rewind`

`Head.rewind` has been a no-op since Next.js 9.5, in Next.js 11 it was removed. You can safely remove your usage of `Head.rewind`.

Moment.js locales excluded by default

Moment.js includes translations for a lot of locales by default. Next.js now automatically excludes these locales by default to optimize bundle size for applications using Moment.js.

To load a specific locale use this snippet:

```
import moment from 'moment';
import 'moment/locale/ja';

moment.locale('ja');
```

You can opt-out of this new default by adding `excludeDefaultMomentLocales: false` to `next.config.js` if you do not want the new behavior, do note it's highly recommended to not disable this new optimization as it significantly reduces the size of Moment.js.

Update usage of `router.events`

In case you're accessing `router.events` during rendering, in Next.js 11 `router.events` is no longer provided during pre-rendering. Ensure you're accessing `router.events` in `useEffect`:

```
useEffect(() => {
  const handleRouteChange = (url, { shallow }) => {
    console.log(
      `App is changing to ${url} ${
        shallow ? 'with' : 'without'
      } shallow routing`,
    );
  };

  router.events.on('routeChangeStart', handleRouteChange);

  // If the component is unmounted, unsubscribe
  // from the event with the `off` method:
  return () => {
    router.events.off('routeChangeStart', handleRouteChange);
  };
}, [router]);
```

If your application uses `router.router.events` which was an internal property that was not public please make sure to use `router.events` as well.

React 16 to 17

React 17 introduced a new JSX Transform ↗ that brings a long-time Next.js feature to the wider React ecosystem: Not having to `import React from 'react'` when using JSX. When using React 17 Next.js will automatically use the new transform. This transform does not make the `React` variable global, which was an unintended side-effect of the previous Next.js implementation. A codemod is available to automatically fix cases where you accidentally used `React` without importing it.

Most applications already use the latest version of React, with Next.js 11 the minimum React version has been updated to 17.0.2.

To upgrade you can run the following command:

```
npm install react@latest react-dom@latest
```

Or using `yarn`:

```
yarn add react@latest react-dom@latest
```