Custom App

Next.js uses the `App` component to initialize pages. You can override it and control the page initialization and:

- Persist layouts between page changes
- Keeping state when navigating pages
- Inject additional data into pages
- Add global CSS

To override the default `App`, create the file `pages/_app.js` as shown below:

pages/_app.js

```
export default function MyApp({ Component, pageProps }) {
  return <Component {...pageProps} />;
}
```

The `Component` prop is the active `page`, so whenever you navigate between routes, `Component` will change to the new `page`. Therefore, any props you send to `Component` will be received by the `page`.

`pageProps` is an object with the initial props that were preloaded for your page by one of our data fetching methods, otherwise it's an empty object.

The `App.getInitialProps` receives a single argument called `context.ctx`. It's an object with the same set of properties as the `context` object in `getInitialProps`.

Caveats

- If your app is running and you added a custom `App`, you'll need to restart the development server. Only required if `pages/_app.js` didn't exist before.

- Adding a custom `getInitialProps` in your `App` will disable Automatic Static Optimization in pages without Static Generation.
- When you add `getInitialProps` in your custom app, you must `import App from "next/app"`, call `App.getInitialProps(appContext)` inside `getInitialProps` and merge the returned object into the return value.
- `App` does not support Next.js Data Fetching methods like `getStaticProps` or `getServerSideProps`. If you need global data fetching, consider incrementally adopting the `app/` directory.

TypeScript

If you're using TypeScript, take a look at our TypeScript documentation.