Create React App

(https://github.com/learn-co-curriculum/react-hooks-create-react-app) (https://github.com/learn-co-curriculum/react-hooks-create-react-app/issues/new)

Learning Goals

- Understand how to create a React app using `create-react-app`

Introduction

So far, through this React course, a basic framework of files has been provided in the labs. In order to make React easier to work with, a specific file structure and set of *dependencies* is required. Having to set all that up every time can be a bit of a pain and is also prone to error. On top of this, copying and pasting old React projects means you may miss out on the most up-to-date React features.

Fortunately, the creators of React have also set up a handy tool for rapidly creating the barebones file structure we need for React apps. In this lesson, we will be discussing how to use the `create-react-app` node package to get our own projects off the ground.

Keeping Up To Date

Before we continue, it is recommended you run `npm install -g npm` . This will make sure you have the newest version of `npm` . Once this is installed, you should be able to run the `npx` command. Instead of having to globally install a node package using `npm` , with `npx` , we can provide a node package name as an argument and use remote node packages as though they were installed.

Since you're grabbing the package remotely, you will be getting the most up to date version of it by default!

Creating A React App From Scratch

In your terminal, navigate to a location where you would like your React app directory to be located.

Decide on a name for your app. Once you've got one, run the following with your app's name in place of `your-app-name` :

```
$ npx create-react-app your-app-name --use-npm
```

The `create-react-app` package sets up the basic file structure. Using the `--use-npm` flag will also instruct Create React App to run `npm install` to install the dependencies.

Create React App Features

Let's take a tour of some of the key features of our newly created React app. Using Create React App version 4 (the latest version at the time of writing), our file structure looks like this:

```
your-app-name
├── README.md
├── node_modules
├── package.json
├── package-lock.json
├── .gitignore
├── public
|   ├── favicon.ico
|   ├── index.html
|   ├── logo192.png
|   ├── logo512.png
|   ├── manifest.json
|   └── robots.txt
└── src
    ├── App.css
    ├── App.js
    ├── App.test.js
    ├── index.css
    ├── index.js
    ├── logo.svg
    ├── serviceWorker.js
    └── setupTests.js
```

Here's a quick rundown of what the key files/folders are used for:

- `/public` : Used for static assets, most importantly our `index.html` file. If you look at the HTML in this file, you'll see a `<div>` with the ID of "root" — this is where our React components will go.
- `/src` : All our Javascript and CSS code must go in this folder. This is where our React components live!
- `package.json` : Our project configuration, including npm scripts and our project's list of dependencies.

Even though our `package.json` only lists a few key dependencies (namely `react` , `react-dom` , and `react-scripts`), Create React App provides a number of other dependencies under the hood that make our lives as developers easier! For example:

- **Babel**: a *transpiler* for converting modern JavaScript, and JSX, into something all browsers can understand (more on that in the next lessons)
- **Webpack**: an asset manager that bundles and minifies our JavaScript code, CSS files, and other assets like images (this is what lets us `import` node modules and CSS files in our JavaScript files!)
- **ESLint**: a highly customizable tool for checking our code's syntax in the editor — think of it like a spell checker with superpowers

You can find much more detail in the **create-react-app documentation** (https://create-react-app.dev/) .

Other Tools For Creating React Apps

While Create React App is a great choice for your projects at Flatiron, there are some excellent alternatives that might make sense for other projects you build later on. The **React Docs** (https://reactjs.org/docs/create-a-new-react-app.html) list a few of the most popular ones. A couple of highlights:

- **Gatsby**: a static-site generator that turns your application into separate pages by building HTML files from JSX components. It's a popular choice for personal blogs, including Dan Abramov's **overreacted** (https://github.com/gaearon/overreacted.io)
- **Next.js**: a framework for creating static and server-side generated React applications. Read more about the differences between Client-Side Rendering and Server-Side Rendering **here** (https://developers.google.com/web/updates/2019/02/rendering-on-the-web) .

Conclusion

That's it! With Create React App, it's fast and easy to get a React app up and running. The app is ready to run with `npm start` , and will display some default content. The `README.md` file provided also has a very detailed breakdown of all the additional features that come with `create-react-app` .

While it is perfectly fine to set up your own React files, `create-react-app` is a handy solution to quickly get past any setup and get straight to designing your app. Since it is actively maintained by Facebook, you're also always getting a nicely polished, up-to-date base for your React applications!

Resources

- **create-react-app documentation (https://create-react-app.dev/)**