What is create-react-app

Not required for React but very convenient

create-react-app (CRA) is a **program**

- Creates a directory for a new react application
- Creates/configures package.json
- Installs npm packages
- Common configs
- Webpack for bundling
- Eslint for linting
- Babel for transpiling
- Notable: fairly un-opinionated

Installing and using CRA

```
npx create-react-app cats
```

Replace "cats" with a directory name of your choice

- a temporary global install of `create-react-app`
- and it executes (runs) create-react-app

Creates the directory `cats/` (or whatever)

Why not install it locally?

- Running this creates local installs
- A tool to setup your tools

What did that do?

- Took a long time to install a lot
- Gave instructions (these are in the README)
- Installed a `package.json` and the dependencies
    - "react-scripts" package wraps webpack/babel/etc
    - "react-scripts" ALSO has the configs for those
    - Some config is in package.json directly
- Created a `/src` directory
- Created a `/public` directory

What now?

Some commands are defined in `package.json`:

- `npm run eject` - removes CRA, leaves code/configs
    - Don't use this, but try it out on a test project
- `npm run build` - creates static files in `/build`
    - NOT `public/`! Like `/src`, `/public` is **input** here
    - NO SERVER - you have to write or use one
- `npm start` - starts a dev server
    - Auto reloads/rebuilds when the code changes
    - Not for final use - use `npm run build` and use the `/build` results for a real server

What do we have?

- in `/public` the `index.html` file has no real content
    - but does have `<div id="root"></div>`
- `/src/index.js` loads `App` and injects into `#root`

But what is `App`?

To make ME happy, rename `src/App.js` to `src/App.jsx`

Build Process

All bundled using `webpack` and transpiled by `babel`

Run `npm start`

- This starts a dev-server (keeps terminal busy)
- This transpiles and bundles the code

This is just pre-configured work we've done before

Non-JS imports

`import` is a standard, despite requiring a bundler

Some **non-standard** transpiling extensions are often used to bring CSS and/or images into a web-page

- requires a **configured** transpiler/bundler to do
- CRA gives you that config

```
import React from 'react'; // standard library import
import ReactDOM from 'react-dom'; // standard library import
import logo from './logo.svg'; // NON-STANDARD, path to img
import './index.css'; // NON-STANDARD, adds CSS to page
import App from './App'; // standard local file import
```