

On this page



Getting Started

Create React App is an officially supported way to create single-page React applications. It offers a modern build setup with no configuration.

Quick Start

```
npx create-react-app my-app  
cd my-app  
npm start
```

If you've previously installed `create-react-app` globally via `npm install -g create-react-app`, we recommend you uninstall the package using `npm uninstall -g create-react-app` or `yarn global remove create-react-app` to ensure that `npx` always uses the latest version.

(*`npx` comes with `npm` 5.2+ and higher, see [instructions for older npm versions](#))*

Then open <http://localhost:3000/> to see your app.

When you're ready to deploy to production, create a minified bundle with `npm run build`.



Starts the development server.

`yarn build`

Bundles the app into static files for production.

`yarn test`

Starts the test runner.

```
yarn eject
  Removes this tool and copies build dependencies, configuration files
  and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

  cd my-app
  yarn start

Happy hacking!
λ cd my-app
λ npm start

> my-app@0.1.0 start ~/my-app
> react-scripts start
```

Get Started Immediately

You **don't** need to install or configure tools like webpack or Babel. They are preconfigured and hidden so that you can focus on the code.

Create a project, and you're good to go.

Creating an App

You'll need to have Node ≥ 14 on your local development machine (but it's not required on the server). You can use [nvm](#) (macOS/Linux) or [nvm-windows](#) to switch Node versions between different projects.

To create a new app, you may choose one of the following methods:

npx

```
npx create-react-app my-app
```

([npx](#) comes with [npm 5.2+](#) and higher, see [instructions for older npm versions](#))

npm

```
npm init react-app my-app
```

`npm init <initializer>` is available in [npm 6+](#)

Yarn

```
yarn create react-app my-app
```

`yarn create` is available in Yarn 0.25+

Selecting a template

You can now optionally start a new app from a template by appending `--template [template-name]` to the creation command.

If you don't select a template, we'll create your project with our base template.

Templates are always named in the format `cra-template-[template-name]`, however you only need to provide the `[template-name]` to the creation command.

```
npx create-react-app my-app --template [template-name]
```

You can find a list of available templates by searching for `"cra-template-*` on npm.

Our [Custom Templates](#) documentation describes how you can build your own template.

Creating a TypeScript app

You can start a new TypeScript app using templates. To use our provided TypeScript template, append `--template typescript` to the creation command.

```
npx create-react-app my-app --template typescript
```

If you already have a project and would like to add TypeScript, see our [Adding TypeScript](#) documentation.

Selecting a package manager

When you create a new app, the CLI will use [npm](#) or [Yarn](#) to install dependencies, depending on which tool you use to run `create-react-app`. For example:

```
# Run this to use npm
```

```
npx create-react-app my-app  
# Or run this to use yarn  
yarn create react-app my-app
```

Output

Running any of these commands will create a directory called `my-app` inside the current folder. Inside that directory, it will generate the initial project structure and install the transitive dependencies:

```
my-app  
├── README.md  
├── node_modules  
├── package.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
```

No configuration or complicated folder structures, only the files you need to build your app. Once the installation is done, you can open your project folder:

```
cd my-app
```

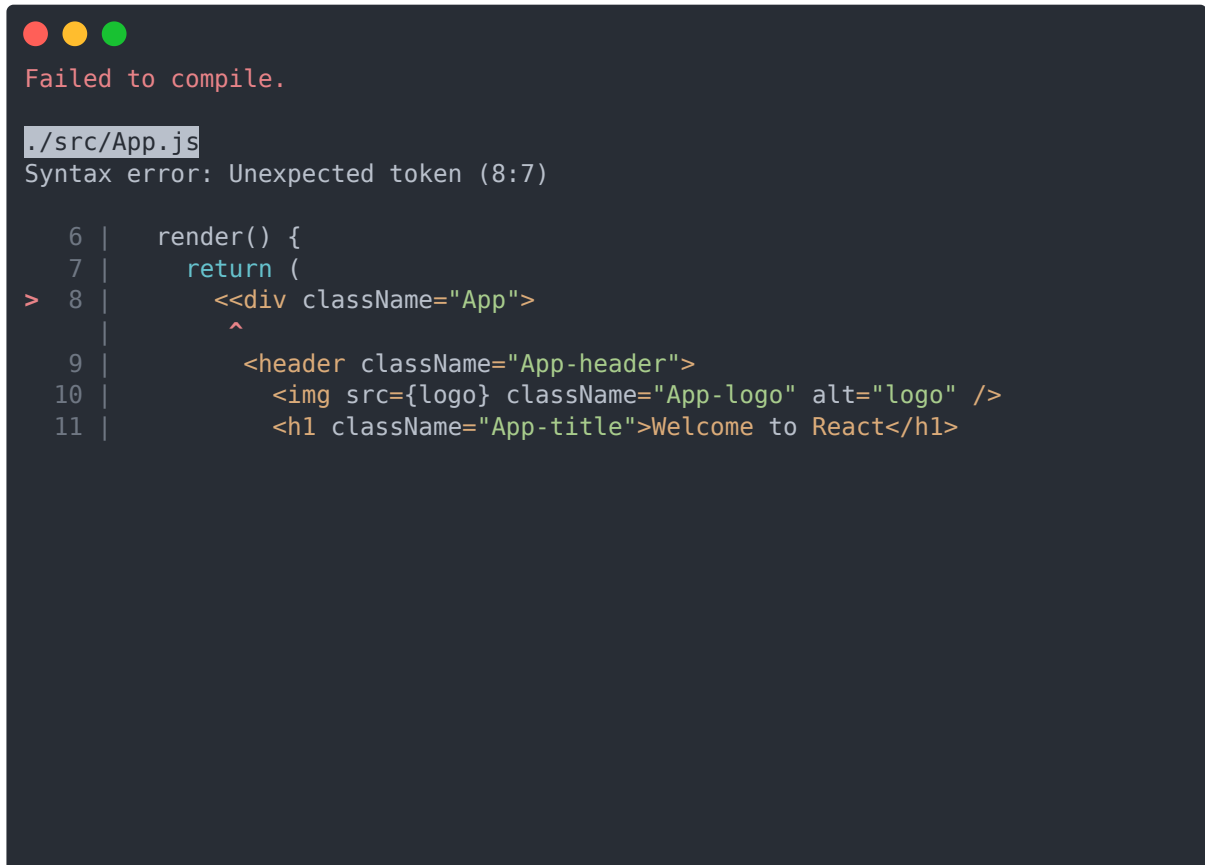
Scripts

Inside the newly created project, you can run some built-in commands:

npm start or yarn start

Runs the app in development mode. Open <http://localhost:3000> to view it in the browser.

The page will automatically reload if you make changes to the code. You will see the build errors and lint warnings in the console.

A terminal window with a dark background and light-colored text. At the top, there are three colored circles (red, yellow, green) and the text "Failed to compile." in red. Below that, the file path "./src/App.js" is highlighted in blue. The error message "Syntax error: Unexpected token (8:7)" is displayed. The code snippet shows a render function with a return statement containing JSX. On line 8, there is a closing tag for a div, but the opening tag is not properly closed, leading to the error. The code is as follows:

```
6 |   render() {  
7 |     return (  
> 8 |       <<div className="App">  
   |         ^  
9 |         <header className="App-header">  
10 |           <img src={logo} className="App-logo" alt="logo" />  
11 |           <h1 className="App-title">Welcome to React</h1>
```

npm test or yarn test

Runs the test watcher in an interactive mode. By default, runs tests related to files changed since the last commit.

[Read more about testing.](#)

npm run build or yarn build

Builds the app for production to the `build` folder. It correctly bundles React in production mode and optimizes the build for the best performance.

The build is minified and the filenames include the hashes.

Your app is ready to be deployed.

 [Edit this page](#)

*Last updated on **9/1/2021** by **Luke Karrys***