React create-react-app

Starting a new React project is very complicated, with so many build tools. It uses many dependencies, configuration files, and other requirements such as Babel, Webpack, ESLint before writing a single line of React code. Create React App CLI tool removes all that complexities and makes React app simple. For this, you need to install the package using NPM, and then run a few simple commands to get a new React project.

The **create-react-app** is an excellent tool for beginners, which allows you to create and run React project very quickly. It does not take any configuration manually. This tool is wrapping all of the required dependencies like **Webpack**, **Babel** for React project itself and then you need to focus on writing React code only. This tool sets up the development environment, provides an excellent developer experience, and optimizes the app for production.

Requirements

The Create React App is maintained by **Facebook** and can works on any **platform**, for example, macOS, Windows, Linux, etc. To create a React Project using create-react-app, you need to have installed the following things in your system.

1. Node version >= 8.10
2. NPM version >= 5.6

Let us check the current version of **Node** and **NPM** in the system.

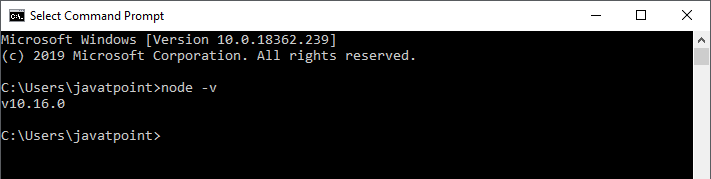
52.3M

1K

C++ vs Java

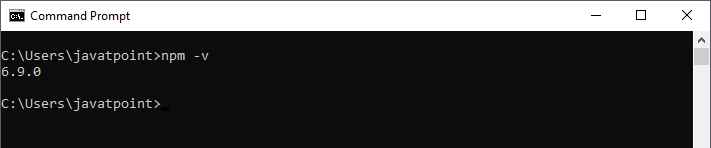
Run the following command to check the Node version in the command prompt.

1. $ node -v



Run the following command to check the NPM version in the command prompt.

1. $ npm -v



Installation

Here, we are going to learn how we can install React using ***CRA*** tool. For this, we need to follow the steps as given below.

Note :

npm install (in a package directory): Install the dependencies in the local node\_modules folder. In global mode (ie, with -g or --global appended to the command), it installs the current package context (ie, the current working directory) as a global package.

Local Installation of Packages: Local packages are installed in the directory where you run npm install <package-name> and they are put in the node\_modules folder under this directory.

Install React

We can install React using npm package manager by using the following command. There is no need to worry about the complexity of React installation. The create-react-app npm package manager will manage everything, which needed for React project.

1. C:\Users\javatpoint> npm install -g create-react-app

(Note : if we install create-react-app locally then it will not create react app it will happen in my system. So install globally)

Create a new React project

Once the React installation is successful, we can create a new React project using create-react-app command. Here, I choose "reactproject" name for my project.

1. C:\Users\javatpoint> create-react-app reactproject

**NOTE:** We can combine the above two steps in a single command using **npx**. The npx is a package runner tool which comes with npm 5.2 and above version.

**NPX: The npx stands for Node Package Execute and it comes with the npm, when you installed npm above 5.2.0 version then automatically npx will installed. It is an npm package runner that can execute any package that you want from the npm registry.****npx is also a CLI tool whose purpose is to make it easy to deal with dependencies hosted in the npm registry.**

**Note: The index.html is the main root file in your front-end & index.js is the main starting point of your backend. When you install Node on machine there is an npm which is node package manager which can install the project libraries and packages using 'npm install' based on the information on the package.json file. If you look into this file there is script part such as this:**

**"scripts": {**

**"start": "node ./index.js",**

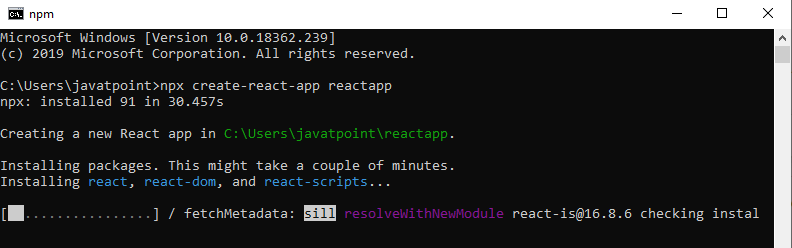
**},**

**The project command starting with npm such as npm run start or npm run build ... can be found and define here.**

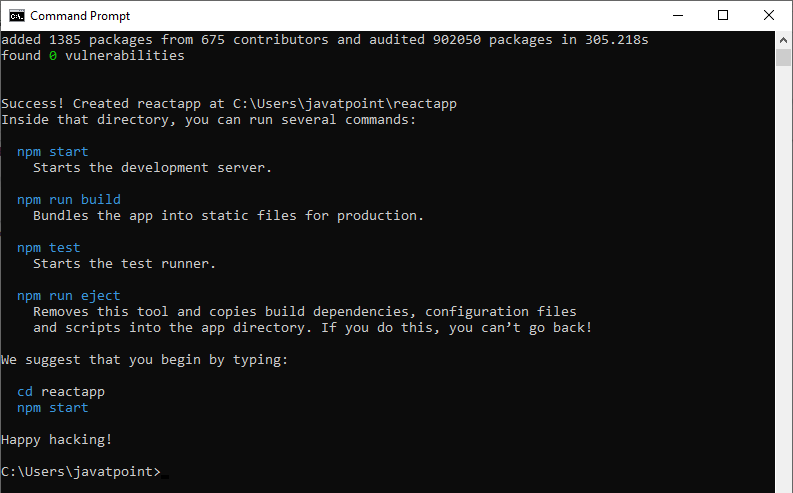
**When you start with your backed it initiated your project and load required templates. When you using IDE such as Webstorm, Visual Studio, or ... they build and run your entire project. So, by initiating and serving the index.html just you can see a part of front-end application and this running that you can see is what 'IDE's provide for you.**

1. C:\Users\javatpoint> npx create-react-app reactproject

**NOTE :**

**REMEMBER TO COME IN CREATED FOLDER BY CD COMMAND TO RUN THE PROJECT. DO NOT FORGET OTHERWISE IT GIVES ERROR.**   


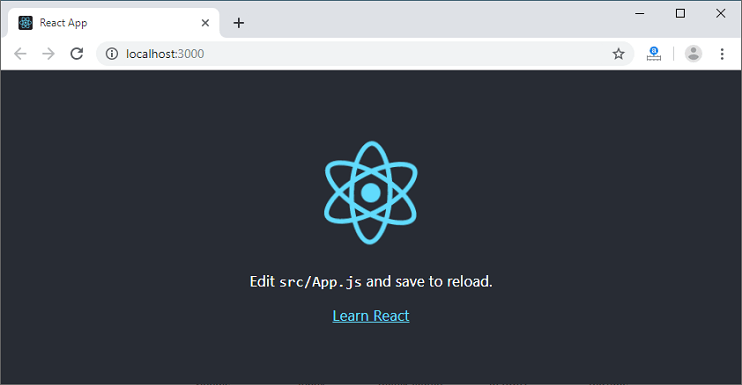
The above command will take some time to install the React and create a new project with the name "reactproject." Now, we can see the terminal as like below.



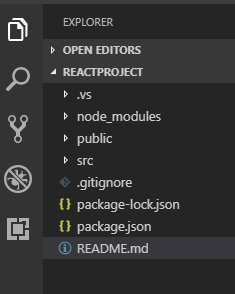
The above screen tells that the React project is created successfully on our system. Now, we need to start the server so that we can access the application on the browser. Type the following command in the terminal window.

1. $ cd reactapp
2. $ npm start

NPM is a package manager which starts the server and access the application at default server [http://localhost:3000](http://localhost:3000/). Now, we will get the following screen.



Next, open the project on Code editor. Here, I am using Visual Studio Code. Our project's default structure looks like as below image.



In React application, there are several files and folders in the root directory. Some of them are as follows:

1. **node\_modules:** It contains the React library and any other third party libraries needed.
2. **public:** It holds the public assets of the application. It contains the index.html where React will mount the application by default on the <div id="root"></div> element.
3. **src:** It contains the App.css, App.js, App.test.js, index.css, index.js, and serviceWorker.js files. Here, the App.js file always responsible for displaying the output screen in React.
4. **package-lock.json:** It is generated automatically for any operations where npm package modifies either the node\_modules tree or package.json. It cannot be published. It will be ignored if it finds any other place rather than the top-level package.
5. **package.json:** It holds various metadata required for the project. It gives information to npm, which allows to identify the project as well as handle the project?s dependencies.
6. **README.md:** It provides the documentation to read about React topics.

React Environment Setup

Now, open the **src >> App.js** file and make changes which you want to display on the screen. After making desired changes, **save** the file. As soon as we save the file, Webpack recompiles the code, and the page will refresh automatically, and changes are reflected on the browser screen. Now, we can create as many components as we want, import the newly created component inside the **App.js** file and that file will be included in our main **index.html** file after compiling by Webpack.

Next, if we want to make the project for the production mode, type the following command. This command will generate the production build, which is best optimized.

$ npm build

Note : <https://www.freecodecamp.org/news/npm-vs-npx-whats-the-difference>/

**Differences between npm and npx:**

|  |  |
| --- | --- |
| **npm** | **npx** |
| If you wish to run package through npm then you have to specify that package in your package.json and install it locally. | A package can be executable without installing the package. It is an npm package runner so if any packages aren’t already installed it will execute through npm registry. |
| To use `create-react-app` in npm the commands are `npm install create-react-app` then `create-react-app myApp` (Installation required). | In npx you can create a react app without installing the package: `npx create-react-app myApp` This command is required in every app’s life cycle only once. |
| Npm is a tool that use to install packages. | Npx is a tool that use to execute packages. |
| Packages used by npm are installed globally. You have to care about pollution in the long term. | Packages used by npx are not installed globally. You don’t have to worry about for pollution in the long term. |