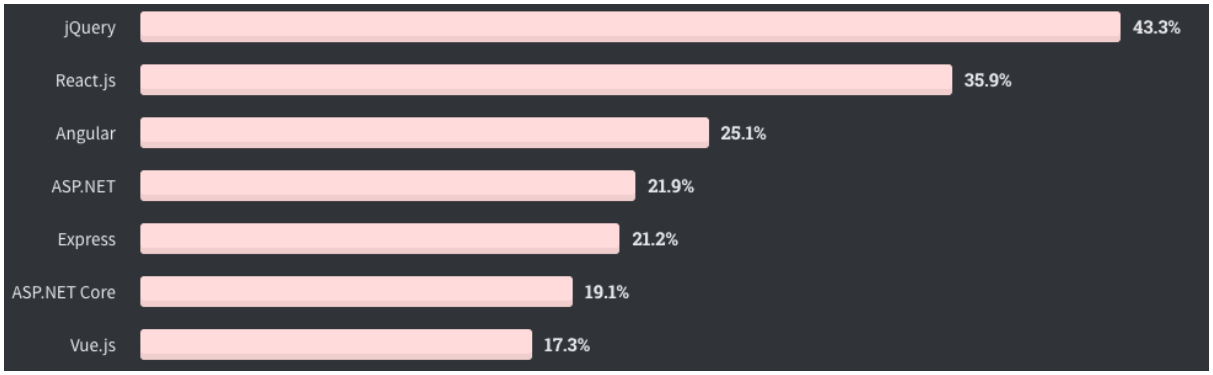
React Basics: Creating a Todo App
(step by step)

React has been all the rage in front-end development over the past 3-5 years and its popularity has still been growing strong.



https://insights.stackoverflow.com/survey/2020#most-popular-technologies

What is React?

- It is an open-source library built with JavaScript that allows you to build user interfaces more efficiently.
- It is a not a “framework” as it is minimal in its footprint and can work very nicely with other libraries.
- It allows the developer to break out the user interface into reusable pieces of code called “components”. This reusability is the power of React (and other similar tools) that will help scale the application more effectively.
- It keeps the data (state) and UI in sync and updates the UI when the data changes. This eliminates the need of writing custom functions that would interface with the browser DOM API.

When to Learn React?

- Don’t jump into learning React until you have a good understanding of the core concepts of JavaScript:
    - Primitive data types
    - Control flow
    - Loops

    - Arrays and objects
    - Scope and closure
    - Callbacks and higher-order functions (map, forEach, filter, etc.

- You don't have to be a JavaScript ninja, but the above concepts should be very familiar to you. ☺

Core Concepts of React:

JSX

- JavsScript and XML, syntax extension developed for React
- Is an abstraction for creating the UI of a component. React has a createElement method that always executes for a component, but using JSX allows developers an easier way to code React components.
- The term "elements" and "components" are referenced in React but are two separate entities. Elements are written in JSX (button, list, header, etc.) and make up one React component.
- JSX consists of HTML and JavaScript expressions. Anytime you need to include JavaScript inside JSX, you would use the bracket {} syntax.

Components

- Comes in two flavors: class components and function components. Class components are based of ES6 Classes and function components are just regular functions that can be written in declaration or expression syntax.
- The first letter needs to be **C**apitalized (because of how JSX works), otherwise it will be thought of as an element.
- Are re-rendered anytime data changes within it (state or sent props).
- Allows the developer to encapsulate pieces of code to be re-used throughout the application.
- Has either a .js or .jsx extension. Many developers prefer .jsx because it quickly identifies if a file is a component.

Props

- Stands for Properties and it is the data you send (i.e., function argument) to a component.
- Each component gets a Props object, and each time you send a “prop” to a component, it will be added to the Props object as a property.
- The prop data that is sent to a component is read only and cannot be changed in the component that receives it.

State

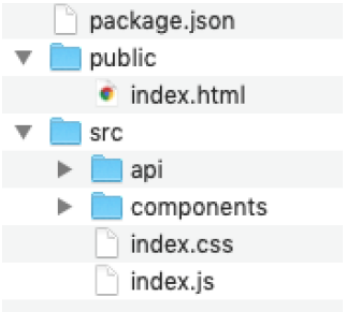
- Defined as the local data stored within a component.
- Class component have state built-in and are ready to use, but function components require hooks which are built-in functions that are part of the React library. The useState hook is the hook that gives function components the power of state.
- State can be passed to other “child” components via props to allow state to be lifted up. This pattern allows multiple components to share state where a child component will use state that is defined in a parent component.
- State management is a huge topic in React and takes a lot of practice to truly understand.

Tool Chains

- To use React, you need a set of tools that will build the necessary folders and files. These “group” of tools is called a tool chain.
- There are many pre-built tool chains such as create-react-app, Gatsby, Next.js, Parcel, etc. **OR** you can create your own.
- Creating your own tool chain is the best way to learn how to build a react application with the most flexibility.
- The main categories of tools you will need in a tool chain will be:
    - package manager
    - bundler
    - compiler

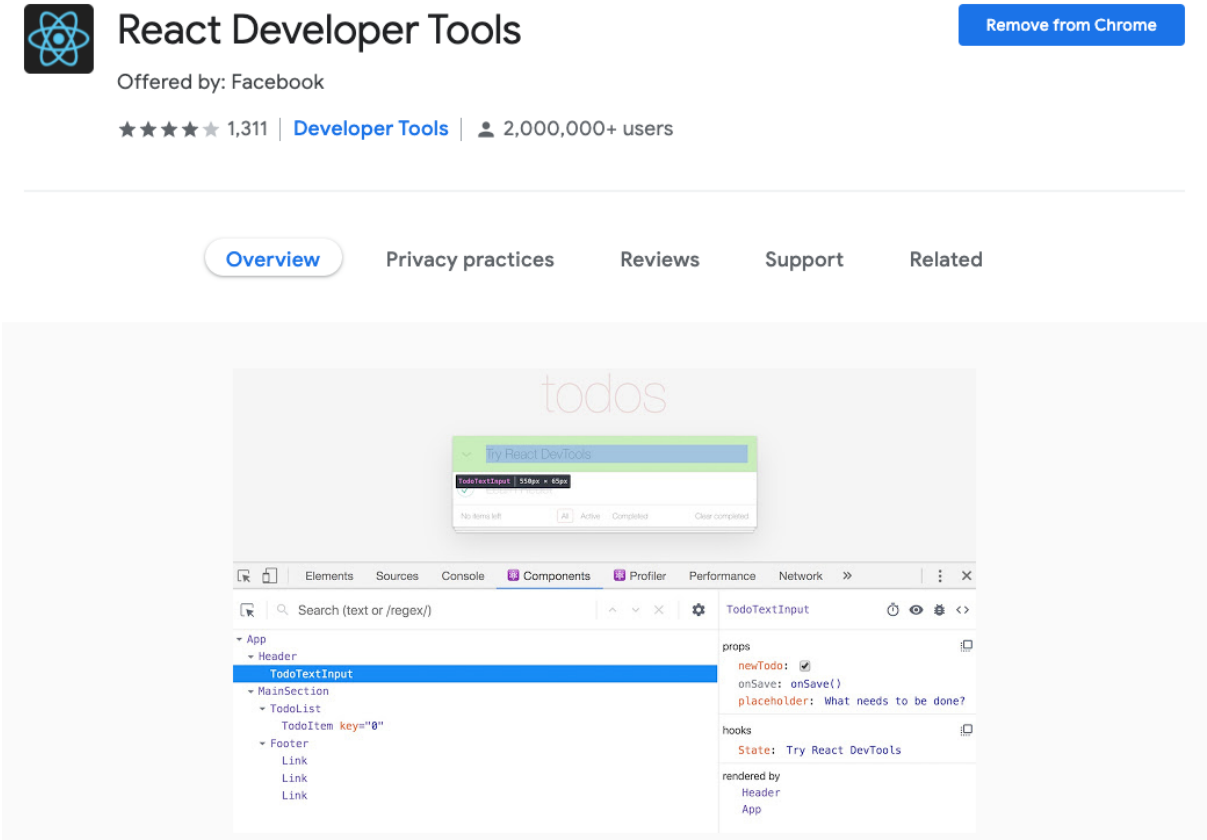
- library/packages (making your React project work)

- Whichever build tool you use, they will all scaffold a folder and file structure to be used as you build your React project. Each will be a little different, but will normally have a **src** and **public** folder as well as an **index.js** and **package.json** and other configuration files.



React Dev Tools

The most widely used developer tool to help you troubleshoot and debug you React app is the [React Developer Tools]



React Ecosystem

There is a never-ending list of related patterns, topics, technologies and tools that surround React and it will impossible to know if not event be aware of them all. So do not try.

Buzzwords like “redux, cypress, axios, apollo, fiber, formik, portals” all sound like they came out of science fiction movie script.

Understand the basics, then slowly build your understanding, little by little, and never worry about knowing everything, because you never will. ☺