**Setup React.js in 5 minutes or Less!**

1. Create GitHub Repo and clone it down into a local directory.
2. Ensure Auto-Save is **OFF**in VS Code Preferences.
3. Open VS Code at the local directory (you will be able to see both the repo's directory and any other sister directories we generate at this higher level).
4. In the VS Code terminal, type **npx create-react-app test-app** to download the create-react-app library, set up the React.js environment, and remove the create-react-app library from our computer (we only need it to set up our app.
   * The **npx**tells Node.js that we'll download, use, and remove the following library.
   * The **create-react-app**library is a user-friendly way to develop React.js applications in a learning environment.
   * The **test-app**is the name we wish to call our React.js application
5. Once create-react-app has completed its setup in a new directory (you'll see 2 directories in your VS Code explorer and even in your Window's Folder/Mac Finder now -- the one with **your repo's name** and another with **test-app**).
6. At the bottom of your terminal are 2 lines we need to execute: a way to navigate our terminal into the new directory and then a way to start our React.js server. For us, that will look like:
   * **cd test-app**
   * **npm start**
7. The **npm start** command compiles our code and spools up a local React.js server on **localhost:3000** in the browser. You should see a smoothly-spinning React.js logo on a dark background with information linking you to the React.js documentation.
8. Now to clean up our local items since we won't be doing any testing and we can remove some of the clutter involved with it. In your VS Code explorer, navigate to the **src** directory and *remove*the following files:
   * **App.css**
   * **App.test.js**
   * **index.css**
   * **logo.svg**
   * **reportWebVitals.js**
   * **setupTests.js**
9. In **index.js**,remove the import statements for **index.css** and **reportWebVitals** on lines 3 and 5, as well as the function call on line 17 for **reportWebVitals()**. Save the file.
10. In **App.js**, remove the import statements for **logo**and **App.css**. You may save the file if you wish, but just be aware your application will attempt to compile and throw a few errors since we aren't importing these items anymore but are still attempting to use them elsewhere in the file.
11. In **App.js**, import React on line 1 like so: **import React from 'react';**
12. While in **App.js**, remove the **<header>**element and its content (lines 6 - 19 in the video). In its place as the content of the **<div>**, write **Hello World!** and when you save the file, your server will re-compile the new user interface for you (no refreshing needed!). Ensure your content has changed before moving on.
13. In your VS Code explorer, create a new folder inside of **src**called **components** (folders should be *camelCase*and React.js files should be *PascalCase*, like **App.js**, **NamePresenter.js**, etc.). Your file structure should now resemble this:
    * **public**
    * **src**
      + **components**
      + **App.js**
      + **index.js**
    * .gitignore
    * package-lock.json
    * package.json

You are now ready to start creating new custom components and using them in **App.js**! Practice these steps and follow along with the video at least 2x per day to become more comfortable with setting up your React.js projects in the future. Feel free to brag about your setup times in Slack -- shoot for *5 minutes or less!*

*NB: The****.js****and****.jsx****file extensions are interchangeable (at this level of learning) when writing****COMPONENTS****(****index.js****is an entry point, not a component, and****shouldn't****be using the****.jsx****extension). The key differences you may concern yourself with are:*

* *You may receive more "React-y" Intellisense help in VS Code with components using the****.jsx****extension.*
* *Your GitHub will reflect this project as containing React.js instead of just JavaScript.*