**ReactJS Notes**

**What is React js**

* a javascript library for building user interfaces
* client-side (runs on the user browser)
* the page doesn’t need to refresh as it does not need to request a new html page, making it super fast
* reduces code needed
* uses components which has dedicated, but small task
* we use react mainly for single page applications. Sometimes when we click a new tab, it looks like we requested a new html page, but we just changed what is visible
* there is html code in js code, which browser cannot understand which is why there is a ‘build’ process which makes the code readable from the browser, so the code we write is not exactly the same as the code the browser reads

**Installing React**

* first we need to install node js <https://nodejs.org/en/> and download the latest version
* check nodejs is installed by typing node-v into the cmd
* open cmd and cd into a folder where you want the react app to be (such as desktop)
* run in cmd: npx create-react-app INSERT\_APP\_NAME
* once it says happy hacking, you are done
* run in cmd: cd INSERT\_APP\_NAME
* run in cmd: npm start (as long as you keep this up, the website will update depending on code changes)
* this should open a tab on your browser at <http://localhost:3000/>

A picture containing logo

Description automatically generated

* open the react app in a code editor
* Text

  Description automatically generated
* Open src folder, and delete app.test.js, logo.svg, reportWebVitals.js, setupTests.js, and App.css
* You should be left with the following: Graphical user interface, application

  Description automatically generated
* In src, go to index.js and make it look like the following

Text

Description automatically generated

* In src, go to index.css and remove all of it

Graphical user interface, application, Teams

Description automatically generated

* In src, go to app.js and make it the following

Graphical user interface, application

Description automatically generated

* Open in vscode, go to view, extensions, and type prettier
* Graphical user interface, text

  Description automatically generated
* Install it, then go to file, preferences, keyboard shortcuts, type format document, and edit the first one to make a shortcut u want

**Diving into the project**

* We have two js files, index and app. Index is the starting point of our react application, the code in here will be the first code that will be executed in our browser.
* Notice we imported the ReactDOM object from the ‘react-dom’ library
* On the ReactDOM object, we call a render method which allows us to add html code. This html code works because of the build step. This html in js code is called JSX. JSX is not understood by browsers and is converted behind the scenes.



* Notice this code renders our own html element, the <App/> element which is imported from the App file.
* We tell the App element to be placed where the element of id of root is. (you can find the root in public/index.html)
* Index.html is the single html page in the project since react is mainly used for single paged applications
* Notice if you inspect the page, not ctrl U, and open the div with id root, we see code added there even tho we didn’t add it directly on the index.html page.

Text

Description automatically generated with medium confidence

**Components**

* Looking at App.js, App is a react component, which is just a standard js function, but it returns JSX code.
* Text

  Description automatically generated with medium confidence
* We export the function using ‘export default AppNAME;’ to make it accessible outside the file
* Also, notice when we inspect the page, we don’t see a div called App, we just see the return value of the App function. This is because these custom elements like App is not understood by the browser.
* We can change the app js code to look like the following
* We just write the html code we want to have appear on the screen

Text

Description automatically generated

* Notice the browser auto reloads to detect the changes
* Go to <https://github.com/academind/react-complete-guide-code/blob/zz-reactjs-summary/extra-files/index.css> and copy the css code and put it into index.css
* Go back to app.js and add the card class to a div using the className. Note we don’t use class like html since class is a keyword in javascript.
* Text

  Description automatically generated

**Building and Reusing Components**

* However, right now, all our code is in one component, and we might want to split that up in the future. For example, if we want a second todo, we have to replicate the entire code block. As well, if we want to change all the code blocks, you have to go to each of the code blocks and edit it manually instead of making the change in one location. That’s why react has components
* Create a new folder in the src folder called components
* Create a new file in the components folder which will be the name of your app. The name has to start with a capital to differentiate it from built-in html components.

Graphical user interface, application

Description automatically generated

* Now make app.js and todo.js look like the following
* Note the Todo app is self closing, we could also write <Todo></Todo>, but since there is not content between it, we can self close it.

Text

Description automatically generatedText

Description automatically generated

* If we want multiple Todo, we just have to Todo multiple Todo elements and one change in the Todo.js file will change all the Todo elements.

**Props and Dynamic Content in Components**

* We want to pass arguments into the components
* In app.js, add the following

Text

Description automatically generated

* In the component, we can add a parameter called props(we can name it whatever we want, but the idea is called props so we’ll use props)
* Props is a js object and all the arguments are key value pairs. So the attribute names such as ‘text’ would be keys will the value would be something like ‘Learn React’
* Text

  Description automatically generated
* Notice we use {} around props.text to tell react this is js and not html, since otherwise, the h2 name would be props.text. Inside the {}, we can have things like 2+2 and it’ll evaluate to 4. You can have any single line expression inside the {}, but you can’t have block expressions like if statements.

Graphical user interface, application

Description automatically generated