Day 3 - Setting Up

In real React projects, instead of CDN, we use the create-react-app package to generate a React project started (boilerplate).

- create-react-app makes it so you only need to worry about developing the product, not manually install all the necessary packages.

Node

This is a JavaScript runtime environment that lets JavaScript run on the server. The React application starts by default on localhost 3000. create-react-app configures a node server for the React application.

Module

A single or multiple functions, that can be exported and imported when needed, can be included in a project. In React, we don't use link to access module or packages. Instead, we import the module.

```
// math.js
export const addTwo = (a, b) +> a + b
export const multiply = (a, b) => a * b
export const subtract = (a, b) => a - b

export default (function doSomeMath() {
  return {
    addTwo,
    multiply,
    subtract,
  }
})()

// Importing math.js modules to a different file:

// index.js
// to import doSomeMath from the math.js with or without extension
import doSomeMath from './math.js'

// To import the other modules
// since these modules were not exported as default, we have to destructure
import { addTwo, multiply, subtract } from './math.js'

import * as everything from './math.js' // to import everything remaining
console.log(addTwo(5, 5))
console.log(doSomeMath.addTwo(5, 5))
console.log(everything)
```

After seeing this, when you see import React from 'react' or import ReactDOM from 'react-dom' you won't be surprised.

Package

A package is a module or collection of modules. For instance, React and ReactDOM are packages.

Node Package Manager (NPM)

Installed with Node. This is the default package manager for Node.js. It allows users to consume and distribute JavaScript modules that are available in the registry. NPM allows users to create packages, use packages, and distribute packages.

Create React App

To create a React project, we can use one of the following ways:

- Assuming you installed node, open the CLI and run the following command:

```
npx create-react-app name-of-your-project
# Always use this one
```

- If you don't like to write npx every time you create a project, you may install the create-react-app package globally in your computer using the below command.

```
npm install -g create-react-app
```

  - After create-react-app has been installed globally, you can create a React application as follows:

```
create-react-app name-of-project
```

Your first React App

First, you'll cd into the directory that will contain your project. Next, create the react app. Then, cd into the react app. Finally, run:

```
npm start
```

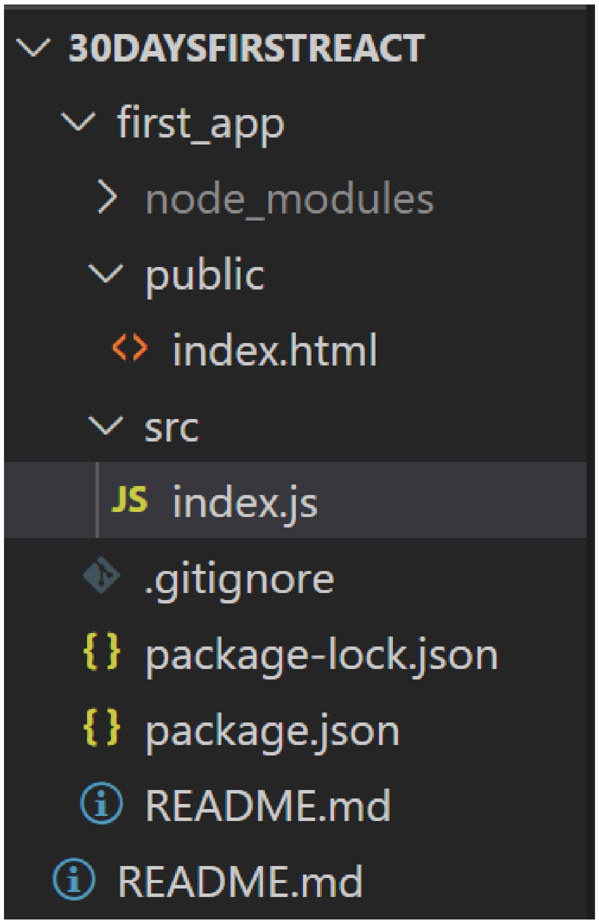
This starts our react application on localhost 3000. We can modify the content by accessing App.js. To stop the server, we press Ctrl + C in the CLI.

React Boilerplate

There are three folders in the React boilerplate: node_modules, public and src. There is also a .gitignore, README.md, package.json, and yarn.lock (or package-lock.json)

- node_modules - Stores all the necessary node packages of the React applications.
- public
    - index.html - the only HTML file we have in the entire application
    - favicon.ico - an icon file
    - manifest.json - Used to make the application a progressive web app
    - other images - open graph images (open graph images are images that are visible when a link is shared on social media)
    - robots.txt - information, if the website allows web scraping
- src
    - App.css, index.css - different CSS files
    - index.js - A file that allows you to connect all the components with index.html
    - App.js - This is the file where we usually import most of the presentational components.
    - serviceWorker.js: Used to add progressive web app features.
    - setupTests.js - To write testing cases
- package.json - List of packages the application uses
- .gitignore - Allows files and folders not to be pushed to GitHub.
- README.md - Markdown file to write documentation
- yarn.lock or package-lock.json - Means to lock the version of the package.

Let’s first delete all the files that we don’t need at the moment:



Now that we’ve deleted the files we won’t be going over (yet), we can write code in index.js. First, we should import React and ReactDOM. React lets us write JSX and ReactDOM lets us render JSX to the DOM. ReactDOM also has a render method that takes two parameters, a JSX or component and the root.

```
//index.js
// importing the react and react-dom package

import React from 'react'
import ReactDOM from 'react-dom'

const jsxElement = <h1>This is a JSX element</h1>
const rootElement = document.getElementById('root')

ReactDOM.render(jsxElement, rootElement)
```

```
<!-- index.html -->
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link
      href="https://fonts.googleapis.com/css?family=Montserrat:300,400,500|Roboto:300,400,500&display=swap"
      rel="stylesheet"
    />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />

    <title>30 Days Of React App</title>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

Let’s write more JSX elements and render them on the browser:

```
import React from 'react'
import ReactDOM from 'react-dom'

const jsxElement = <h1>This is a JSX element.</h1>
const rootElement = document.getElementById('root')

const tech = 'React'

function Title() {
  return <h2>Getting Started with {tech}</h2>
};

const header = (
  <header>
    <h1>Welcome to 30 Days of {tech}</h1>
    <Title />
    <h3>JavaScript Library</h3>
    <p>Chris Guerrero</p>
    <small>March 7, 2023</small>
  </header>
)

const technologies = ['HTML', 'CSS', 'JavaScript']
const formattedTechs = technologies.map((tech) => <li key={tech}>{tech}</li>)

const main = (
  <main>
    <p>Prerequisites to get start with React.js</p>
    <ul>
      {formattedTechs}
    </ul>
  </main>
)

function Footer() {
  return (
    <footer>
      <p>How does a website kick a soccer ball?</p>
      <small>Using its footer!</small>
    </footer>
  )
}

const app = (
  <div>
    {header}
    {main}
    <Footer />
  </div>
)

ReactDOM.render(app, rootElement)
```

Styles in JSX

We can style JSX elements using either inline, internal, or external CSS styles.

```
// index.js
import React from 'react'
import ReactDOM from 'react-dom'

const headerStyles = {
  backgroundColor: '#61DBFB',
  fontFamily: 'Helvetica Neue',
  padding: 25,
  lineHeight: 1.5,
}

// JSX element, header
const header = (
  <header style={headerStyles}>
    <div className='header-wrapper'>
      <h1>Welcome to 30 Days Of React</h1>
      <h2>Getting Started React</h2>
      <h3>JavaScript Library</h3>
      <p>Asabeneh Yetayeh</p>
      <small>Oct 2, 2020</small>
    </div>
  </header>
)

// JSX element, main
const mainStyles = {
  backgroundColor: '#F3F0F5',
}
const main = (
  <main style={mainStyles}>
    <p>Prerequisite to get started react.js:</p>
    <ul>
      <li>HTML</li>
      <li>CSS</li>
      <li>JavaScript</li>
    </ul>
  </main>
)

const footerStyles = {
  backgroundColor: '#61DBFB',
}
```

```
// JSX element, footer
const footer = (
  <footer style={footerStyles}>
    <p>Copyright 2020</p>
  </footer>
)

// JSX element, app
const app = (
  <div className='app'>
    {header}
    {main}
    {footer}
  </div>
)

const rootElement = document.getElementById('root')
// we render the JSX element using the ReactDOM package
ReactDOM.render(app, rootElement)
```

We can also apply internal styling by placing styling in the header of index.html.

```
// index.js
import React from 'react'
import ReactDOM from 'react-dom'
// JSX element, header
const header = (
  <header>
    <div className='header-wrapper'>
      <h1>Welcome to 30 Days Of React</h1>
      <h2>Getting Started React</h2>
      <h3>JavaScript Library</h3>
      <p>Instructor: Asabeneh Yetayeh</p>
      <small>Date: Oct 1, 2020</small>
    </div>
  </header>
)

// JSX element, main
const main = (
  <main>
    <div className='main-wrapper'>
      <p>
        Prerequisite to get started{' '}
        <strong>
          <em>react.js</em>
        </strong>
        :
      </p>
      <ul>
        <li>HTML</li>
        <li>CSS</li>
        <li> JavaScript</li>
      </ul>
    </div>
  </main>
)

// JSX element, footer
```

```
const footer = (
  <footer>
    <div className='footer-wrapper'>
      <p>Copyright 2020</p>
    </div>
  </footer>
)

// JSX element, app
const app = (
  <div className='app'>
    {header}
    {main}
    {footer}
  </div>
)

const rootElement = document.getElementById('root')
// we render the JSX element using the ReactDOM package
ReactDOM.render(app, rootElement)
```

Injecting Data to JSX Elements

```
// index.js
import React from 'react'
import ReactDOM from 'react-dom'
// To get the root element from the HTML document

// JSX element, header
const welcome = 'Welcome to 30 Days Of React'
const title = 'Getting Started React'
const subtitle = 'JavaScript Library'
const author = {
  firstName: 'Asabeneh',
  lastName: 'Yetayeh',
}
const date = 'Oct 2, 2020'

// JSX element, header
const header = (
  <header>
    <div className='header-wrapper'>
      <h1>{welcome}</h1>
      <h2>{title}</h2>
      <h3>{subtitle}</h3>
      <p>
        Instructor: {author.firstName} {author.lastName}
      </p>
      <small>Date: {date}</small>
    </div>
  </header>
)

const numOne = 3
const numTwo = 2

const result = (
  <p>
```

```
    {numOne} + {numTwo} = {numOne + numTwo}
  </p>
)

const yearBorn = 1820
const currentYear = new Date().getFullYear()
const age = currentYear - yearBorn
const personAge = (
  <p>
    {' '}
    {author.firstName} {author.lastName} is {age} years old
  </p>
)

// JSX element, main
const techs = ['HTML', 'CSS', 'JavaScript']
const techsFormatted = techs.map((tech) => <li>{tech}</li>)

// JSX element, main
const main = (
  <main>
    <div className='main-wrapper'>
      <p>
        Prerequisite to get started{' '}
        <strong>
          <em>react.js</em>
        </strong>
        :
      </p>
      <ul>{techsFormatted}</ul>
      {result}
      {personAge}
    </div>
  </main>
)

const copyRight = 'Copyright 2020'

// JSX element, footer
const footer = (
  <footer>
    <div className='footer-wrapper'>
      <p>{copyRight}</p>
    </div>
  </footer>
)

// JSX element, app
const app = (
  <div className='app'>
    {header}
    {main}
    {footer}
  </div>
)

const rootElement = document.getElementById('root')
// we render the JSX element using the ReactDOM package
ReactDOM.render(app, rootElement)
```

Importing Media Objects in React

We can import images, video and audio into react. In our example, we'll create an images folder in the src folder and save an image inside. We'll then import this image to index.js and inject it to a JSX expression.

```
// index.js
import React from 'react'
import ReactDOM from 'react-dom'
import asabenehImage from './images/asabeneh.jpg'

const user = (
  <div>
    <img src={asabenehImage} alt='asabeneh image' />
  </div>
)

const rootElement = document.getElementById('root')
// we render the JSX element using the ReactDOM package
ReactDOM.render(user, rootElement)
```

Applying image to main component:

```
// index.js
import React from 'react'
import ReactDOM from 'react-dom'
// To get the root element from the HTML document
import asabenehImage from './images/asabeneh.jpg'
// JSX element, header
const welcome = 'Welcome to 30 Days Of React'
const title = 'Getting Started React'
const subtitle = 'JavaScript Library'
const author = {
  firstName: 'Asabeneh',
  lastName: 'Yetayeh',
}
const date = 'Oct 2, 2020'

// JSX element, header
const header = (
  <header>
    <div className='header-wrapper'>
      <h1>{welcome}</h1>
      <h2>{title}</h2>
      <h3>{subtitle}</h3>
      <p>
        Instructor: {author.firstName} {author.lastName}
      </p>
      <small>Date: {date}</small>
    </div>
  </header>
)

const numOne = 3
const numTwo = 2
const result = (
  <p>
    {numOne} + {numTwo} = {numOne + numTwo}
  </p>
)

const yearBorn = 1820
const currentYear = new Date().getFullYear()
const age = currentYear - yearBorn
const personAge = (
  <p>
    {' '}
    {author.firstName} {author.lastName} is {age} years old
  </p>
)

// JSX element, main
const techs = ['HTML', 'CSS', 'JavaScript']
const techsFormatted = techs.map((tech) => <li>{tech}</li>)

const user = (
  <div>
    <img src={asabenehImage} alt='asabeneh image' />
  </div>
)

// JSX element, main
const main = (
  <main>
    <div className='main-wrapper'>
      <p>
        Prerequisite to get started{' '}
        <strong>
          <em>react.js</em>
        </strong>
        :
      </p>
      <ul>{techsFormatted}</ul>
      {result}
      {personAge}
      {user}
    </div>
  </main>
)

const copyRight = 'Copyright 2020'

// JSX element, footer
const footer = (
  <footer>
    <div className='footer-wrapper'>
      <p>{copyRight}</p>
    </div>
  </footer>
)

// JSX element, app
const app = (
  <div className='app'>
    {header}
    {main}
    {footer}
  </div>
)

const rootElement = document.getElementById('root')
// we render the JSX element using the ReactDOM package
ReactDOM.render(app, rootElement)
```