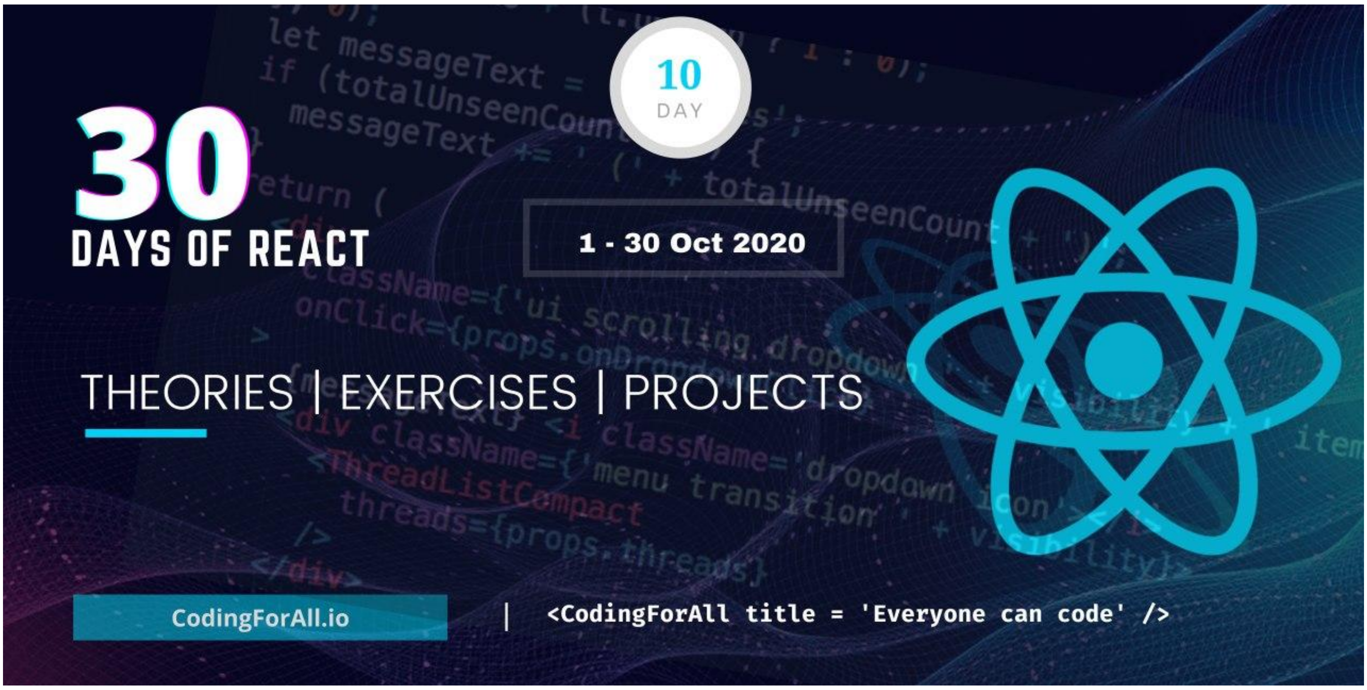
# 30 Days Of React: React Project Folder Structure

LinkedIn Follow @asabeneh 776

Author: Asabeneh Yetayeh

October, 2020

<< Day 9 | Day 11 >>


30
DAYS OF REACT
10
DAY
1 - 30 Oct 2020
THEORIES | EXERCISES | PROJECTS
CodingForAll.io
<CodingForAll title = 'Everyone can code' />

# React Project Folder Structure and File Naming

There is no strict way to use a single folder structure or file naming in React project. Most of the time, these kind of choice can be made by a team. Sometimes a company may have a developed guidelines about what code conventions to follow, folder structure and file naming. There is no right or wrong way of structuring a React project but some structures are better than the others for scalability,maintainability, ease of working on files and easy to understand structure. If you like to learn further about folder structure you may check the following articles.

- React Folder Structure by https://www.devaradise.com

- React Folder Structure by www.robinwieruch.de
- React Folder Structure by Faraz Ahmad
- React Folder Structure by https://maxrozen.com/

I use a mix of different conventions. If you like you can follow it but please stick in a structure which you think makes sense for you.

## File Naming

In all my React project, I will use CamelCase file name for all components. I prefer to use descriptive long name.

## Folder

I found it easy to put all images, icons and fonts in the assets folder and all CSS style sheets in styles folder. All components will be in the components folder.

So far, we have been working on index.js file. We have lots of component on index.js. Today we will move every component to a single file and we will import all the files to App.js. In the process, you will see my folder structure. Currently, we are at src directory. All the folder structure will be inside the src directory. Let's start from the index.js file. In addition to index.js file, let's create an App.js file and move most of the components we had to App.js for the time being. The index.js is your getaway to connect the component with index.html.

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

const App = () => <h1>Welcome to 30 Days Of React</h1>

const rootElement = document.getElementById('root')
ReactDOM.render(<App />, rootElement)
```

In the above snippet of code, we have the App component. Let's create the App component to its own file, App.js

```
// src/App.js
import React from 'react
const App = () => <h1>Welcome to 30 Days Of React</h1>
```

We have to export the component to import it in another file. We can export it as a default or named export. In one file, we can make one default export and many named exports. First, let's implement it using name export and later in default export.

We just add the keyword export before *let* or *const* to make a named export.

```
// src/App.js
import React from 'react

// named export in arrow function
export const App = () => <h1>Welcome to 30 Days Of React</h1>
```

Exporting in a function declaration, a regular function

```
// src/App.js
import React from 'react
// named export in regular function, function declaration
export function App () {
return <h1>Welcome to 30 Days Of React</h1>
}
```

Now, let's import the App component from the App.js file to index.js.

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

const rootElement = document.getElementById('root')
ReactDOM.render(<App />, rootElement)
```

We saw a named export and now let's implement it with default export. We can do it in two ways but it is recommended to use the second way if we are exporting components because sometimes we may bind a component with another higher order component.

```
// src/App.js
import React from 'react
// export default in arrow function
export default const App = () => <h1>Welcome to 30 Days Of React</h1>
```

```
// src/App.js
import React from 'react
// export default in arrow function
export default function App () {
  return <h1>Welcome to 30 Days Of React</h1>
}
```

```
// src/App.js
// Recommended for most of the cases
import React from 'react
const App = () => <h1>Welcome to 30 Days Of React</h1>
export default App
```

If a component is exported as default we do not need curly bracket during importing.

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

const rootElement = document.getElementById('root')
ReactDOM.render(<App />, rootElement)
```

If you remember, we created the following components so far and we have been putting them together. It is not easy to work like this. Now we will move them all components to a separate file.

```
// index.js
import React from 'react'
import ReactDOM from 'react-dom'
import asabenehImage from './images'
import { countriesData } from './data/countries'

// Header component
class Header extends React.Component {
  render() {
    console.log(this.props.data)
    const {
      welcome,
      title,
      subtitle,
      author: { firstName, lastName },
      date,
    } = this.props.data

    return (
      <header>
        <div className='header-wrapper'>
          <h1>{welcome}</h1>
          <h2>{title}</h2>
          <h3>{subtitle}</h3>
          <p>
            {firstName} {lastName}
          </p>
          <small>{date}</small>
        </div>
      </header>
    )
  }
}

const Country = ({
  country: { name, capital, flag, languages, population, currency },
}) => {
  const formatedCapital =
    capital.length > 0 ? (
      <>
        <span>Capital: </span>
        {capital}
      </>
    ) : (
      ''
    )
  const formatLanguage = languages.length > 1 ? `Languages` : `Language`
  return (
    <div className='country'>
      <div className='country_flag'>
        <img src={flag} alt={name} />
      </div>
      <h3 className='country_name'>{name.toUpperCase()}</h3>
      <div class='country_text'>
        <p>{formatedCapital}</p>
        <p>
          <span>{formatLanguage}: </span>
          {languages.join(', ')}
        </p>
        <p>
          <span>Population: </span>
          {population.toLocaleString()}
        </p>
        <p>
          <span>Currency: </span>
          {currency}
        </p>
      </div>
    </div>
  )
}

// User Card Component
const UserCard = () => (
  <div className='user-card'>
    <img src={asabenehImage} alt='asabeneh image' />
    <h2>Asabeneh Yetayeh</h2>
  </div>
)

// Hexadecimal color generator
const hexaColor = () => {
  let str = '0123456789abcdef'
  let color = ''
  for (let i = 0; i < 6; i++) {
    let index = Math.floor(Math.random() * str.length)
    color += str[index]
  }
  return '#' + color
}

const HexaColor = () => <div>{hexaColor()}</div>

const Message = ({ message }) => (
  <div>
    <h1>{message}</h1>
  </div>
)
const Login = () => (
  <div>
    <h3>Please Login</h3>
  </div>
)
const Welcome = (props) => (
  <div>
    <h1>Welcome to 30 Days Of React</h1>
  </div>
)

// A button component
const Button = ({ text, onClick, style }) => (
  <button style={style} onClick={onClick}>
    {text}
  </button>
)

// TechList Component
// class base component
class TechList extends React.Component {
  render() {
    const { techs } = this.props
    const techsFormatted = techs.map((tech) => <li key={tech}>{tech}</li>)
    return techsFormatted
  }
}

// Main Component
// Class Component
class Main extends React.Component {
  render() {
    const {
      techs,
      greetPeople,
      handleTime,
      loggedIn,
      handleLogin,
      message,
    } = this.props
    console.log(message)

    const status = loggedIn ? <Welcome /> : <Login />
    return (
      <main>
        <div className='main-wrapper'>
          <p>Prerequisite to get started react.js:</p>
          <ul>
            <TechList techs={this.props.techs} />
          </ul>
          {techs.length === 3 && (
            <p>You have all the prerequisite courses to get started React</p>
          )}
          <div>
            <Button
              text='Show Time'
              onClick={handleTime}
              style={buttonStyles}
            />{' '}
            <Button
              text='Greet People'
              onClick={greetPeople}
              style={buttonStyles}
            />
            {!loggedIn && <p>Please login to access more information about 30 Days Of React challenge</p>}
          </div>
          <div style={{ margin: 30 }}>
            <Button
              text={loggedIn ? 'Logout' : 'Login'}
              style={buttonStyles}
              onClick={handleLogin}
            />
            <br />
            {status}
          </div>
          <Message message={message} />
        </div>
      </main>
    )
  }
}

// CSS styles in JavaScript Object
const buttonStyles = {
  backgroundColor: '#61dbfb',
  padding: 10,
  border: 'none',
  borderRadius: 5,
  margin: 3,
  cursor: 'pointer',
  fontSize: 22,
  color: 'white',
  margin: '0 auto',
}

// Footer Component
// Class component
class Footer extends React.Component {
  constructor(props) {
    super(props)
  }
  render() {
    return (
      <footer>
        <div className='footer-wrapper'>
          <p>Copyright {this.props.date.getFullYear()}</p>
        </div>
      </footer>
    )
  }
}

class App extends React.Component {
  state = {
    loggedIn: false,
    techs: ['HTML', 'CSS', 'JS'],
    message: 'Click show time or Greet people to change me',
  }
  handleLogin = () => {
    this.setState({
      loggedIn: !this.state.loggedIn,
    })
  }
  showDate = (time) => {
    const months = [
      'January',
      'February',
      'March',
      'April',
      'May',
      'June',
      'July',
      'August',
      'September',
      'October',
      'November',
      'December',
    ]

    const month = months[time.getMonth()].slice(0, 3)
    const year = time.getFullYear()
    const date = time.getDate()
    return `${month} ${date}, ${year}`
  }
  handleTime = () => {
    let message = this.showDate(new Date())
    this.setState({ message })
  }
  greetPeople = () => {
    let message = 'Welcome to 30 Days Of React Challenge, 2020'
    this.setState({ message })
  }

  render() {
    const data = {
      welcome: '30 Days Of React',
      title: 'Getting Started React',
      subtitle: 'JavaScript Library',
      author: {
        firstName: 'Asabeneh',
        lastName: 'Yetayeh',
      },
      date: 'Oct 9, 2020',
    }
    const techs = ['HTML', 'CSS', 'JavaScript']

    return (
      <div className='app'>
        {this.state.backgroundColor}
        <Header data={data} />

        <Main
          techs={techs}
          handleTime={this.handleTime}
          greetPeople={this.greetPeople}
          loggedIn={this.state.loggedIn}
          handleLogin={this.handleLogin}
          message={this.state.message}
        />

        <Footer date={new Date()} />
      </div>
    )
  }
}

const rootElement = document.getElementById('root')
ReactDOM.render(<App />, rootElement)
```

## Components Folder

Inside the src directory will pull all the components

```
src
  App.js
  index.js
  components
```

```
     -auth
      -Signup.js
      -Signin.js
      -Forgotpassword.js
      -Resetpassord.js
    header
     -Header.js
    footer
     -Footer.js
    assets
     -images
     -icnons
     - fonts
    styles
     -button.js
     -button.scss
   utils
    -random-id.js
    -display-time.js
    -generate-color.js
   shared
    -Button.js
    -InputField.js
    -TextAreaField.js
```

Let's create components directory inside src and inside components let's create header director. Create Header.js inside the header directory.

```
// src/components/header/Header.js
import React from 'react'

const Header = (props) => {
  return (
    <header>
      <div className='header-wrapper'>
        <h1>{welcome}</h1>
        <h2>{title}</h2>
        <h3>{subtitle}</h3>
        <p>
          {firstName} {lastName}
        </p>
        <small>{date}</small>
      </div>
    </header>
  )
}

export default Header
```

Similar to the Header let's move all the components to their respective files. All the CSS files on index.html will moved into styles folder and after that each part has been split its respective file, try to check the styles folder.

## Fragments

Fragments are a way to avoid unnecessary parent element in JSX. Let's implement a fragment. We import fragment from react module. As you can see below, we imported React and fragment together by use a comma separation.

```
import React, { Fragment } from 'react'

const Skills = () => {
  return (
    <Fragment>
      <li>HTML</li>
      <li>CSS</li>
      <li>JavaScript</li>
    </Fragment>
  )
}
const RequiredSkills = () => {
  return (
    <ul>
      <Skills />
    </ul>
  )
}
```

It is also possible to just extract the fragment module from React as shown below.

```
import React from 'react'

const Skills = () => {
  return (
    <React.Fragment>
      <li>HTML</li>
      <li>CSS</li>
      <li>JavaScript</li>
    </React.Fragment>
  )
}

const RequiredSkills = () => {
  return (
    <ul>
      <Skills />
    </ul>
  )
}
```

In latest version of Reacts it also possible to write without extracting or importing using this signs(<> </>)

```
import React from 'react'

// Recommended
const Skills = () => {
  return (
    <>
      <li>HTML</li>
      <li>CSS</li>
      <li>JavaScript</li>
    </>
  )
}

const RequiredSkills = () => {
  return (
    <ul>
      <Skills />
    </ul>
  )
}
```

When we make a class-based component we have been using React.Component instead we can just import the component and the code will look more clean. Let's see an example.

```
import React from 'react'

// without importing the Component
// Not recommended
class App extends React.Component {
  render() {
    return <h1> 30 Days of React </h1>
  }
}
```

```
import React, { Component } from 'react'

// This is recommended
class App extends Component {
  render() {
    return <h1> 30 Days of React </h1>
  }
}
```

Well done. Time to do some exercises for your brain and muscles.

# Exercises

## Exercises:Level 1

1. What is the importance of React Folder Structure and File Naming
2. How do you export file
3. How do you import file
4. Make a component of module and export it as named or default export
5. Make a component or module and import it
6. Change all the components you have to different folder structure

## Exercises:Level 2

1. Make a simple portfolio using the components we have created so far. Implement a dark mode by using the function we wrote on day 8 challenge.

## Exercises: Level 3

Coming

🎉 CONGRATULATIONS ! 🎉

<< Day 9 | Day 11 >>