Chapter 05 - Let's get Hooked!

Date : 08/01/2023

Ashraya_KK

React File Structure

* React doesn't have opinions on how you put files into folders
* Some devs group their files by features
* While moving files into folders, we need to export it so that we can import it wherever necessary.

There are 2 ways of exporting :

(i) export default

→ This is the default way.

→ This means you want to export only one value.

A module is a self contained unit that can expose assets to other modules using export, and acquire assets from other modules using import

→ There can be only one default export.

→ Default export is the value that will be imported from the module, if we use the simple import statement :-

```
import Title from "./components/Title";
```

'Title' is the name that will be given locally to the variable assigned to contain the value, and it doesn't have to be named like the origin export.

(ii) Named exports

Eg :- `export const Title = () => { }`

→ This is a named export with a name 'Title'

→ It can be imported like :-

```
import {Title} from "./components/Header";
```

→ If we created a new file which have 2 components and I want to export both of these components.

(i) I can wrap these components into single object and can export.

(ii) Or I can export it seperately

→ If Header.tsx have 2 components :-
Header & Title.
Then, either we can ~~each~~ export each using names,
or wrap it into single component and use default export. So, import {Title, Header} from "./components/Header"; will work.
Or we can use :-

```
import * as Obj from "./components/Header";
```

Then, we can use
'Obj.Title' in our code

Header.js

```
export const Title = () => {..}
~~export~~ ~~const~~
const Header = () => {..}
export Default Header
```

App.js

```
import Header,
{Title} from
"./comp/Header"
```

When you are using the component in same file, you don't have to export.

CONFIG FILE

- Create a config file in your project.
- I put all the hardcoded things into my config file. (config.js)
- Also called as constants file (constants.js)
- Config file ~~should~~ will be like :-

```
const IMG_CDN_URL = "https://someurl";
```

- make this as named export

```
export const IMG_CDN_URL = "https//";
```

- Then import it like :-

```
import { IMG_CDN_URL } from "./config";
```

- Put all hardcoded data. So, put restaurantList also in config file.

Building Search Functionality

→ Search bar - inside Body.

→ const Body = () => {

```
return (
  <>
    <div classname = "search-container" >
     <input
        type="text"
        className = "search-input"
        placeholder = "Search"
        value = " "

      />
    <button classnname = "search-btn">
      Search
    </button>

  </>
); };
```

→ We've got search input and search button with us. But if I try to write inside my input box, it's not working (because it's controlled by React.

[If I write the same code inside my HTML file, it will work.]

ONE-WAY DATA BINDING IN REACT

* Const Body = () => {

```
const SearchTxt = "KFC";
return (
  <>
  <div>
   <input type="text"
          placeholder="Search"
          value={searchTxt}
  </>      />
  );};
```

I have a variable 'SearchTxt' and if I put that here, Then, the value "KFC" will go inside my input box.

→ I'm not able to edit the value "KFC" because it is a hardcoded value.

→ To change the value in the input box, we need to modify the variable SearchTxt

But in input box, If we write something, it won't change SearchTxt.

This is called One-way data binding.

* How will I change the value of SearchTxt ?

Ans) :- Write an onChange method

onChange = {(e) => onChangeInput}

Create an onChangeInput function - It takes a function (which is a call back fn) which have a e event.

So, whenever input is changed, this function will be called.

→ If you need to maintain a variable that changes itself, then you need to maintain a 'React-kinda' variable.

→ React Variable.

It's like a state variable.

* [Every component in React maintains a state. So, you can put some variables on to that state.]

[Every time you have to create a local variable, you use state in it.]

→ In react, If I want to create a local variable like SearchTxt, I will create it using useState Hook

useState Hook

→ New way of creating variables

```
const [searchTxt] = useState();
```

If we have to create local variables in react, you need to use state variables.

→ State variables are created using **useState hook**

What is Hooks ?

- Hooks are normal functions.
- I get useState hook from 'react' library. (imported using named export)

What is the function of useState ?

- It's to create state variables.

```
const [SearchTxt] = useState();
```

(Second element is a set function to update the variable)

This function returns an array

The first element of this array is variable name.

'SearchTxt' is a local state variable.

→ To give a default value to my useState variable, do this ↓

```
Const [searchTxt] = useState("KFC");
```

→ This is how we create a local state variable in react.

> In javascript, we create a variable searchTxt having a value "KFC" like this ↓
>
> ```
> const searchTxt = "KFC";
> ```

→ In react, to modify the variable 'searchTxt' I have to use function.

useState() gives us that function.

Let us call that function 'setSearchTxt()'

```
const [searchTxt, setSearchTxt] = useState("KFC");
```

```
onChange = {(e) => {
    setSearchTxt("e.target.value");
}}
```

(e) → from this event property I can read whatever I'm typing

```
* const Body = () => {
    const [searchTxt, setSearchTxt] = useState("KFC");
    return (
      <>
      <div classname = "search-container">
      <input type = "text"
             className = "search-input"
             placeholder = "Search"
             value = {searchTxt}
             onchange = {(e) => {
               setSearchTxt(e.target.value);
             }}
      />
      <button className = "search-btn">
        Search - {searchTxt}
      </button>
    </>
    );};
```

TWO-WAY BINDING

Here, I'm reading as well as writing searchTxt. ~~Both~~

※ We have local variables. Why do we need state variables?

A):- Becoz, react has no idea what's happening to your local variables. So, react won't re-render any updates happening on that variable. Everytime, the variable wants to be in sync with the UI. For that, we need to use State variables.

React keeps track of State variable.

※ Whenever my variable is updated, my whole 'Body' (here) component re-renders. ie, React destroy the 'Body' component and create it again. Reconciliation (Diff algm) is happening behind the scenes.

※ But it just re-renders that updated portion. It is very quick.

[Interesting Section → 01:43:00].

Search Functionality

① We need to filter the data.

→ data here is restaurantList.

→ Create a function filterData();

② Update restaurantList when we filter the data.

→ I cannot update it directly, we've to make a State variable for this.

```
Const [restaurants, setRestaurants] = useState(
                                      restaurantList);
```

③ Suppose I updated my restaurant with filterdata, then I must modify this local variable, restaurant with the filter data.

FilterData function

```
filterData (searchTxt, restaurants)
```

searchTxt → my input. This the text we have to search in

restaurants → I'll search my searchTxt inside restaurants and filter data.

```
const data = filterData(searchTxt, restaurants);
setRestaurant(data);
```

Filter Algorithm (normal js).

```
function filterData(searchTxt, restaurants) {
  return (
    restaurants.filter((restaurant) =>
      restaurant.data.name.includes(searchTxt)
    );
}
```

```
<div className="restaurant-List">
  {restaurants.map((restaurant) => {
    return (
      <RestaurantCard {...restaurant.data}
        key={restaurant.data.id} />
    );
  })};
</div>
```

After one search, the state got updated and again when we search keywords, it won't work.