**react** - framework (refer)

**it is library**

**props - details**

**heading - state**

now- function based components

class based components

bebel compiles jsx down to

react.createElement() calls

**npm install - create new node module**

react function based component - **rfc**

import PropTypes from 'prop-types'

---------------------------

(pts)

Navbar.prototype = {

title: PropTypes.string.isRequired,

About: PropTypes.string

}

Navbar.defaultProps = {

title: 'Set title here',

About: 'About text here'

}

**HOOK** - bina class banae class ke features use karne ki madat

**{useState}** - help to make a state variable

**const [text , setText] = useState(‘Enter the text here’);**// inside func based component upside of return statement

**HANDLE ON CHANGE EVENT:**

<textarea

className="form-control"

value={text}

onChange={handleOnChnage}

id="MyBox"

rows="8"

></textarea>

const handleOnChnage = (event) => {

console.log("OnChnage");

setText(event.target.value);

};

**another way for typeScript ...**

minalchhatre@Minals-MacBook-Air react % npm create vite@4.1.0

Need to install the following packages:

create-vite@4.1.0

Ok to proceed? (y) y

✔ Project name: … react-app

✔ Select a framework: › React

✔ Select a variant: › TypeScript

Scaffolding project in /Users/minalchhatre/Documents/Minal/react/react-app...

Done. Now run:

cd react-app

npm install

npm run dev

minalchhatre@Minals-MacBook-Air react % cd react-app

minalchhatre@Minals-MacBook-Air react-app % npm install

added 84 packages, and audited 85 packages in 18s

9 packages are looking for funding

run `npm fund` for details

found 0 vulnerabilities

minalchhatre@Minals-MacBook-Air react-app %

minalchhatre@Minals-MacBook-Air react % cd react-app

minalchhatre@Minals-MacBook-Air react-app % npm run dev

**LIBRARY :** Tool to provide functionality

**Framework:** set of tools and guidelines for building app

**npm i [bootstrap@5.2.3](mailto:bootstrap@5.2.3)**

**//in main.tsx**

import 'bootstrap/dist/css/bootstrap.css'

**we dont have for loop so the approach:**

const items =['India','France','Itali','Germany','USA'];

{items.map(item => <li key={item}>{item}</li>)}

//key={item} is unique id for item

{items.map((item, index) => (

<li

className="list-group-item"

key={item}

onClick={() => console.log("Clicked " + item, index)}

>

{item}

</li>

))}

**js -**

true && ‘Minal’

ans : ‘Minal’

false && ‘Minal’

ans : false

true && 1

ans: 1

{/\* {items.length === 0 ? <p>No Item Found</p> : null}; - conditional statement , below is another way \*/}

{items.length === 0 && <p>No Item Found</p>}

**Event Handling in typescript react -**

calling

// onClick={() => console.log("Clicked " + item, index)}

//onClick={(event) => console.log(event)}

onClick={handleClick}

Importing

import { MouseEvent } from "react";

function

//event handler

const handleClick = (event: MouseEvent) => console.log(event);

**Hook useState**

import

import { useState } from "react";

declare

const [selectedIndex, setselectedIndex] = useState(0);

handle

onClick={() => {

setselectedIndex(index);

}}

**Props**

//{items: [], heading: string}

//typescript interface

interface Props {

items: string[];

heading: string;

}

function ListGroup({ items, heading }: Props) {

App.tsx

unction App() {

let items = ["India", "France", "Itali", "Germany", "USA"];

return (

<div>

<ListGroup items={items} heading="Cities" />

</div>

);

}

ListGroup.tsx

import { useState } from "react";

//import { MouseEvent } from "react";

//{items: [], heading: string}

//typescript interface

interface Props {

items: string[];

heading: string;

onSelectItem: (item: string) => void;

}

function ListGroup({ items, heading, onSelectItem }: Props) {

//items = [];

//const message = items.length === 0 ? <p>No Item Found</p> : null; - by using variable

//by using function

// const getMessage = () => {

// return items.length === 0 ? <p>No Item Found</p> : null;

// };

//event handler

//const handleClick = (event: MouseEvent) => console.log(event);

const [selectedIndex, setselectedIndex] = useState(0);

return (

<>

<h1>{heading}</h1>

{/\* {getMessage()} \*/}

{/\* {items.length === 0 ? <p>No Item Found</p> : null}; - conditional statement , below is another way \*/}

{items.length === 0 && <p>No Item Found</p>}

<ul className="list-group">

{items.map((item, index) => (

<li

className={

selectedIndex === index

? "list-group-item active"

: "list-group-item"

}

key={item}

// onClick={() => console.log("Clicked " + item, index)}

//onClick={(event) => console.log(event)}

//onClick={handleClick}

onClick={() => {

setselectedIndex(index);

onSelectItem(item);

}}

>

{item}

</li>

))}

</ul>

</>

);

}

export default ListGroup;

App.tsx

//import Message from "./Message";

import ListGroup from "./Components/ListGroup";

function App() {

let items = ["India", "France", "Itali", "Germany", "USA"];

const handleSelectItem = (item: string) => {

console.log("Clicked " + item);

};

return (

<div>

<ListGroup

items={items}

heading="Cities"

onSelectItem={handleSelectItem}

/>

</div>

);

}

export default App;

**shortcut - rafce**

import React from 'react'

const Alert = () => {

return (

<div>

</div>

)

}

export default Alert

**Passing children: pass children to a component**

Alert.tsx

//import React from "react";

import { ReactNode } from "react";

interface Props {

children: ReactNode;

}

const Alert = ({ children }: Props) => {

return (

<div className="alert alert-success" role="alert">

{children}

</div>

);

};

export default Alert;

App.tsx

//import Message from "./Message";

import Alert from "./Components/Alert";

import ListGroup from "./Components/ListGroup";

function App() {

let items = ["India", "France", "Itali", "Germany", "USA"];

const handleSelectItem = (item: string) => {

console.log("Clicked " + item);

};

return (

<>

<ListGroup

items={items}

heading="Cities"

onSelectItem={handleSelectItem}

/>

<Alert>

Hello <span>World</span>

</Alert>

</>

);

}

export default App;

Button.tsx

interface Props {

children: string;

//color?: string; //to specify the property is optional to typscript we add ?

color?:

| "primary"

| "secondary"

| "danger"

| "success"

| "warning"

| "info"

| "light"

| "dark"

| "link";

onClick: () => void;

}

const Buttons = ({ children, onClick, color = "warning" }: Props) => {

return (

<>

<button type="button" className={"btn btn-" + color} onClick={onClick}>

{children}

</button>

</>

);

};

export default Buttons;

Alert.tsx

//import React from "react";

import { ReactNode } from "react";

interface Props {

children: ReactNode;

onClose: () => void;

}

const Alert = ({ children, onClose }: Props) => {

return (

<div

className="alert alert-success alert-dismissible fade show"

role="alert"

>

{children}

<button

type="button"

className="btn-close"

data-bs-dismiss="alert"

aria-label="Close"

onClick={onClose}

></button>

</div>

);

};

export default Alert;

Alertbox.tsx

import React, { useState } from "react";

import Alert from "./Alert";

function Alertbox() {

const [alertVisible, setAlertVisibility] = useState(false);

return (

<>

<div>

{alertVisible && (

<Alert onClose={() => setAlertVisibility(false)}>

<strong>Clicked!</strong>

</Alert>

)}

<button

type="button"

className="btn btn-dark"

onClick={() => setAlertVisibility(true)}

>

ClickMe

</button>

</div>

</>

);

}

export default Alertbox;

App.tsx

//import Message from "./Message";

import Alert from "./Components/Alert";

import Alertbox from "./Components/Alertbox";

import Buttons from "./Components/Buttons";

import ListGroup from "./Components/ListGroup";

function App() {

let items = ["India", "France", "Itali", "Germany", "USA"];

const handleSelectItem = (item: string) => {

console.log("Clicked " + item);

};

return (

<>

<ListGroup

items={items}

heading="Cities"

onSelectItem={handleSelectItem}

/>

{/\* adding color property is optional \*/}

<Buttons color="warning" onClick={() => console.log("Button Clicked")}>

Button

</Buttons>

<Alertbox></Alertbox>

</>

);

}

export default App;

show hidden files in folder

command + shift + .