9.REACTJS-HOL

**9.REACTJS-HOL**

Codes:

import React, { useState } from 'react';

// --- Data for the Application ---

// An array of 11 players with their names and scores.

const players = [

{ name: 'Jack', score: 50 },

{ name: 'Michael', score: 70 },

{ name: 'John', score: 40 },

{ name: 'Ann', score: 61 },

{ name: 'Elisabeth', score: 61 },

{ name: 'Sachin', score: 95 },

{ name: 'Dhoni', score: 100 },

{ name: 'Virat', score: 84 },

{ name: 'Jadeja', score: 64 },

{ name: 'Raina', score: 75 },

{ name: 'Rohit', score: 80 },

];

// A separate list of players for the destructuring example.

const IndianTeam = ['Sachin1', 'Dhoni2', 'Virat3', 'Rohit4', 'Yuvraj5', 'Raina6'];

// Two separate arrays of players for the merge example.

const T20Players = ['First Player', 'Second Player', 'Third Player'];

const RanjiTrophyPlayers = ['Fourth Player', 'Fifth Player', 'Sixth Player'];

// --- Reusable Components ---

// A simple styled list item component.

function PlayerListItem({ children }) {

return <li className="py-1 px-2 rounded-md bg-gray-50">{children}</li>;

}

// A styled heading component.

function SectionHeading({ children }) {

return <h1 className="text-2xl font-bold text-gray-800 mt-8 mb-4">{children}</h1>;

}

// --- Components for "List of Players" View (flag=true) ---

// 1. ListofPlayers Component: Displays all players using the .map() feature of ES6.

function ListOfPlayers({ playerList }) {

return (

<div>

<SectionHeading>List of All Players</SectionHeading>

<ul className="space-y-2">

{playerList.map((player, index) => (

<PlayerListItem key={index}>

Mr. {player.name} <span className="font-semibold float-right">{player.score}</span>

</PlayerListItem>

))}

</ul>

</div>

);

}

// 2. ScoreBelow70 Component: Filters players with scores <= 70 using an arrow function.

function ScoreBelow70({ playerList }) {

// Here we use the .filter() method with an ES6 arrow function to create a new array.

const playersBelow70 = playerList.filter(player => player.score <= 70);

return (

<div>

<SectionHeading>List of Players having Scores Less than or equal to 70</SectionHeading>

<ul className="space-y-2">

{playersBelow70.map((player, index) => (

<PlayerListItem key={index}>

Mr. {player.name} <span className="font-semibold float-right">{player.score}</span>

</PlayerListItem>

))}

</ul>

</div>

);

}

// --- Components for "Indian Players" View (flag=false) ---

// 3. OddPlayers Component: Uses array destructuring to get players at odd positions.

function OddPlayers({ team }) {

// ES6 Destructuring: We extract elements by their position in the array.

// The commas are used to skip elements we don't need.

const [first, , third, , fifth] = team;

return (

<div>

<SectionHeading>Odd Players</SectionHeading>

<ul className="space-y-2">

<PlayerListItem>First : {first}</PlayerListItem>

<PlayerListItem>Third : {third}</PlayerListItem>

<PlayerListItem>Fifth : {fifth}</PlayerListItem>

</ul>

</div>

);

}

// 4. EvenPlayers Component: Uses array destructuring to get players at even positions.

function EvenPlayers({ team }) {

// ES6 Destructuring: We skip the first element to access the even-indexed ones.

const [, second, , fourth, , sixth] = team;

return (

<div>

<SectionHeading>Even Players</SectionHeading>

<ul className="space-y-2">

<PlayerListItem>Second : {second}</PlayerListItem>

<PlayerListItem>Fourth : {fourth}</PlayerListItem>

<PlayerListItem>Sixth : {sixth}</PlayerListItem>

</ul>

</div>

);

}

// 5. ListOfIndianPlayers Component: Merges two arrays using the ES6 Spread operator.

function ListOfIndianPlayers() {

// ES6 Spread Syntax (...): We merge the two arrays into a single new array.

const mergedPlayers = [...T20Players, ...RanjiTrophyPlayers];

return (

<div>

<SectionHeading>List of Indian Players Merged:</SectionHeading>

<ul className="space-y-2">

{mergedPlayers.map((player, index) => (

<PlayerListItem key={index}>Mr. {player}</PlayerListItem>

))}

</ul>

</div>

);

}

// --- Main App Component ---

export default function App() {

// useState hook to manage the flag state. The default is true.

const [showListOfPlayers, setShowListOfPlayers] = useState(true);

// This function toggles the state variable when the button is clicked.

const toggleView = () => {

setShowListOfPlayers(!showListOfPlayers);

};

return (

<div className="bg-gray-100 min-h-screen font-sans">

<div className="container mx-auto p-4 md:p-8">

<div className="bg-white p-6 rounded-xl shadow-lg">

{/\* --- Toggle Button --- \*/}

<div className="text-center mb-6">

<button

onClick={toggleView}

className="bg-blue-600 text-white font-bold py-2 px-6 rounded-lg hover:bg-blue-700 transition duration-300 ease-in-out shadow-md"

>

Toggle View (Switch Flag)

</button>

<p className="text-gray-600 mt-2">

Currently showing: <span className="font-semibold">{showListOfPlayers ? "List of Players (flag=true)" : "Indian Players (flag=false)"}</span>

</p>

</div>

<hr className="my-6"/>

{/\* --- Conditional Rendering based on the flag --- \*/}

{showListOfPlayers ? (

// This block is rendered when showListOfPlayers (the flag) is true

<div>

<ListOfPlayers playerList={players} />

<ScoreBelow70 playerList={players} />

</div>

) : (

// This block is rendered when showListOfPlayers (the flag) is false

<div>

<OddPlayers team={IndianTeam} />

<EvenPlayers team={IndianTeam} />

<ListOfIndianPlayers />

</div>

)}

</div>

<footer className="text-center mt-8 text-gray-500">

<p>React Cricket App Assignment</p>

</footer>

</div>

</div>

);

}