**Hands-on: 7. ReactJS-HOL**

**Introduction**

React is designed to build dynamic and reusable UI components. One of its core features is the ability to pass data between components using props and manage dynamic data using state. Understanding how props, default props, state, and ReactDOM.render() work is key to mastering React.

1. **Define Props**

Props (short for properties) are read-only attributes used to pass data from a parent component to a child component in React.

* **Key Points:**
* Props are passed as HTML-like attributes.
* Props are immutable inside the child component.
* Props allow component reusability with different values.
* **Example:**

function Greeting(props) {

return <h1>Hello, {props.name}!</h1>;

}

// Usage:

<Greeting name="Anshika" />

1. **Explain Default Props**

Default Props provide default values for props when none are passed by the parent component.

* **Purpose:**
* Avoid undefined values when props are missing
* Ensure component renders with fallback data
* **Example:**

function Greeting(props) {

return <h1>Hello, {props.name}!</h1>;

}

Greeting.defaultProps = {

name: "Guest"

};

If name is not passed, it will default to "Guest".

1. **Identify the Differences Between State and Props**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Props** | **State** |
| **Definition** | External data passed to a component | Internal data managed within the component |
| **Mutability** | Immutable (read-only) | Mutable (can be changed using setState) |
| **Ownership** | Set by the parent component | Managed by the component itself |
| **Purpose** | Configures component behavior | Tracks dynamic UI or logic changes |
| **Usage** | Used in both functional and class components | Used in both, but often initialized in class or with useState in functions |

1. **Explain ReactDOM.render()**

The ReactDOM.render() method is used to render React elements or components into the actual DOM.

* **Syntax:**

ReactDOM.render(<App />, document.getElementById('root'));

* **How It Works:**
* Takes a React element or component as the first argument
* Takes a DOM node (usually a div with id root) as the second argument
* Renders the React virtual DOM into the actual browser DOM
* **In modern React (v18+), it's replaced by:**

import { createRoot } from 'react-dom/client';

const root = createRoot(document.getElementById('root'));

root.render(<App />);

**Conclusion**

Understanding props, default props, state, and ReactDOM.render() is essential for mastering React's component-driven architecture. While props help in customizing and reusing components, state allows them to be dynamic. The ReactDOM.render() (or createRoot) method serves as the entry point to render the entire app into the DOM. Mastery of these concepts is foundational to building responsive and efficient React applications.