ReactJS Concepts - Detailed Explanation

# Define Props

Props (short for "properties") are read-only components used to pass data from one component to another, typically from a parent component to a child component.

Props allow components to be dynamic and reusable by changing their behavior and rendering based on the data passed.

Props are immutable, meaning a component cannot change the props it receives.

They are passed as attributes to components and can be accessed using `this.props` in class components or directly as arguments in function components.

Example:

<Cart itemname="Laptop" price={50000} />

In the Cart component, you can access it as:

this.props.itemname or this.props.price

# Explain Default Props

Default Props are used to ensure that a React component has default values for its props if they are not explicitly provided by the parent component.

This prevents undefined values and allows for more robust components.

In class components, default props are defined using the static `defaultProps` property.

Example:

class Cart extends React.Component {

// ... render code

}

Cart.defaultProps = {

itemname: "Unnamed Item",

price: 0

};

This ensures that if no `itemname` or `price` is passed to the Cart component, it will use these default values.

# Identify the differences between State and Props

| Feature | Props | State |

|------------------|---------------------------------------|--------------------------------|

| Definition | Used to pass data from parent to child | Used to manage data within a component |

| Mutability | Immutable (cannot be changed) | Mutable (can be changed using setState) |

| Accessibility | Passed from parent, read-only | Maintained inside component |

| Lifecycle Impact | Does not cause re-render if changed externally | Triggers re-render when state changes |

| Purpose | Configure child components | Store local, interactive data |

# Explain reactDOM.render()

The `ReactDOM.render()` method is used to render a React element or component into the DOM.

Syntax (React v17 and below):

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

In React v18 and above, this has been updated to:

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

root.render(<App />);

This function is typically used in the `index.js` file of a React project to render the root component of the application.