React Key

In Detail

Gagan Saini

React Key

Keys are a special prop that helps **identify** and track elements in lists or arrays of components. When you render a **dynamic** list of components in React, each component should have a **unique "key"** prop assigned to it. React uses these keys to efficiently update and **reorder elements** in the **virtual DOM** when the list changes, resulting in improved performance and a better user experience.

Why Key is important ?

Without keys, React may have difficulty efficiently updating the virtual DOM when a list changes. React's **reconciliation** process relies on keys to determine which components were added, removed, or changed in the list. The reconciliation process aims to **minimize unnecessary DOM manipulations** and optimize rendering.

Benefits of using React keys.

- Optimized Rendering
- Reordering Support
- Consistent Element Identity

Optimized Rendering

With keys, React can efficiently update the DOM only for the components that have changed, reducing unnecessary rendering and improving performance.

Reordering Support

If the order of the list changes, React can correctly reorder the elements based on their keys, making list reordering seamless.

Consistent Element Identity

Keys ensure that React can track elements across renders, maintaining their state and avoiding unexpected behavior.

Key Uniqueness

Keys should be **unique among siblings** in the list. However, React **does not enforce** this **uniqueness across different lists**. If you have components from different lists with the same keys, React may encounter issues during reconciliation. Therefore, it's best to ensure key **uniqueness** within the **same list**.

Reconciliation

Reconciliation in React is the process of **comparing** the **previous virtual DOM** with the **new virtual DOM** and determining the minimal set of changes required to update the actual DOM to reflect those changes. It ensures that React efficiently updates only the necessary parts of the user interface, **minimizing DOM manipulations** and improving performance.

Importance of Keys in React Reconciliation

- Efficient DOM Updates
- Element Identity
- Stable Component State
- Optimized Rendering
- List Reordering
- Avoiding Flickering

Example

```
// Object
const arr = [
    { id: 1, name: "Gagan" },
    { id: 2, name: "Rohan" },
    { id: 3, name: "Rahul" },
];

// Rendering
<div>
  {arr.map((item) => {
    return (
      <h1 key={item.id}>
        {item.name} has ID : {item.id}
      </h1>
    );
  })}
</div>
```

@Gagan Saini

Web Developer

Connect For More !