More on React re-rendering

Re-rendering in React happens whenever there is a change in a component's state or props. React components are designed to be declarative, meaning they automatically update when their underlying data changes. The re-rendering process ensures that the user interface (UI) reflects the most up-to-date state of the application.

Specifically, re-rendering occurs under the following circumstances:

1. **State Change:** When a component's state is updated using the `setState()` method (for class components) or the state update function (for functional components using hooks like `useState`), React automatically triggers a re-render of that component and its child components.

2. **Props Change:** If a component receives new props from its parent, React compares the incoming props with the previous props and re-renders the component with the updated prop values if they differ.

3. **Context Change:** When the context API in React is used, a component can subscribe to changes in a specific context. If the context value changes, the component and its descendants will re-render to reflect the updated context.

4. **Force Re-render:** In certain scenarios, you may need to force a re-render of a component. This can be achieved by calling the `forceUpdate()` method (not recommended in most cases, as it bypasses the usual React update process).

It's important to note that React uses a virtual DOM to efficiently calculate the difference between the previous and current states of the components. This process is called "reconciliation" and allows React to minimize actual DOM updates, leading to improved performance and a smoother user experience.