Title: What are hooks in React?

Answer:

Hooks are a new feature added in React v16.8. It allows to use all React features without writing class components. For example, before version 16.8, we need a class component to manage state of a component. Now we can keep state in a functional component using `useState` hook.

React Hooks are in-built functions that allow to use **state** and **lifecycle** methods inside functional components, they also work together with existing code, so they can easily be adopted into a codebase.

Built-in Hooks

Basic Hooks

- useState()
- useEffect()
- useContext()

Additional Hooks

- useReducer()
- useCallback()
- useMemo()
- useRef()
- useImperativeHandle()
- useLayoutEffect()
- useDebugValue()

React Hooks advantages

- Hooks are easier to work with and to test (as separated functions from React components*) and make the code look cleaner, easier to read — a related logic can be tightly coupled in a custom hook.
- Hooks allow to do by breaking the logic between components into small functions and using them inside the components.
- Improved code reuse
- Better code composition
- Better defaults
- Sharing non-visual logic with the use of custom hooks
- Flexibility in moving up and down the components tree.

Let's see an example of useState hook example,

```
import { useState } from 'react';

function Example() {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

Tags: react, react-hooks