What Is Rendering in React, and How Are Components Displayed on the Screen?

In React, rendering is the process by which components appear in the user interface (UI), usually the browser.

React takes all your JavaScript, JSX, and CSS code, figures out how it should look, and then displays it in the user interface.

The rendering process consists of three stages: trigger, render, and commit. Let's take a look at these in more detail.

The trigger stage occurs when React detects that something has changed and that the user interface might need to be updated. This change is often due to an update in the state or props.

For instance, noticing that it's time for dinner can trigger you to go into the kitchen to start cooking.

In the `Counter` example below, clicking the increment or decrement button triggers React to show the new `count` value:

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

function Counter() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <h1>{count}</h1>
      <button onClick={() => setCount(count - 1)}>Decrement</button>
      <button onClick={() => setCount(count + 1)}>Increment</button>
    </div>
  );
}

export default Counter;
```

Once the trigger happens, React enters the render stage. Here, React re-evaluates your components and figures out what to display.

To do this, React uses a lightweight copy of the "real" DOM. This is called the virtual DOM. With the virtual DOM, React can quickly check what needs to change in the component.

Think of this stage as the point where you're in the kitchen, you've gathered your ingredients, and you cook your dinner.

For the `Counter` component, the render stage is the point where React runs the functions again with the new `count` value. React recalculates what the `<h1>{count}</h1>` part of the component should look like based on the updated `count` value, but you won't see any changes on the screen until the next stage – commit.

The commit stage is where React takes the prepared changes from the virtual DOM and applies them to the real DOM. In other words, this is the stage where you see the final result on the screen.

To make this happen, React compares the virtual DOM to the actual DOM, identifies only the parts that need updates, and applies those changes to the real DOM to update the user interface.

You can think of this stage as the point where you serve the food you cooked, making it visible just like React does when it commits updates to the actual DOM.

As for the `Counter` component, the commit stage is the point in which the new `count` value is applied to the `h1` element, and you can see the change on the page.

These three processes are extremely fast because React minimizes direct DOM manipulation by calculating changes in the virtual DOM first, then it updates only the parts that need to change in the real DOM.

Questions

What are the three stages of the rendering process in React?

Start, load, and finish.

Initialize, update, and complete.

Trigger, render, and commit.

Begin, process, and end.

Which technology does React use to perfect the render stage by efficiently calculating changes?

Shadow DOM

Service Workers

Virtual DOM

Web Components

What happens during the commit stage in React's rendering process?

React prepares changes in the virtual DOM.

React applies changes from the virtual DOM to the real DOM.

React decides which components need re-rendering.

React only checks for component errors.