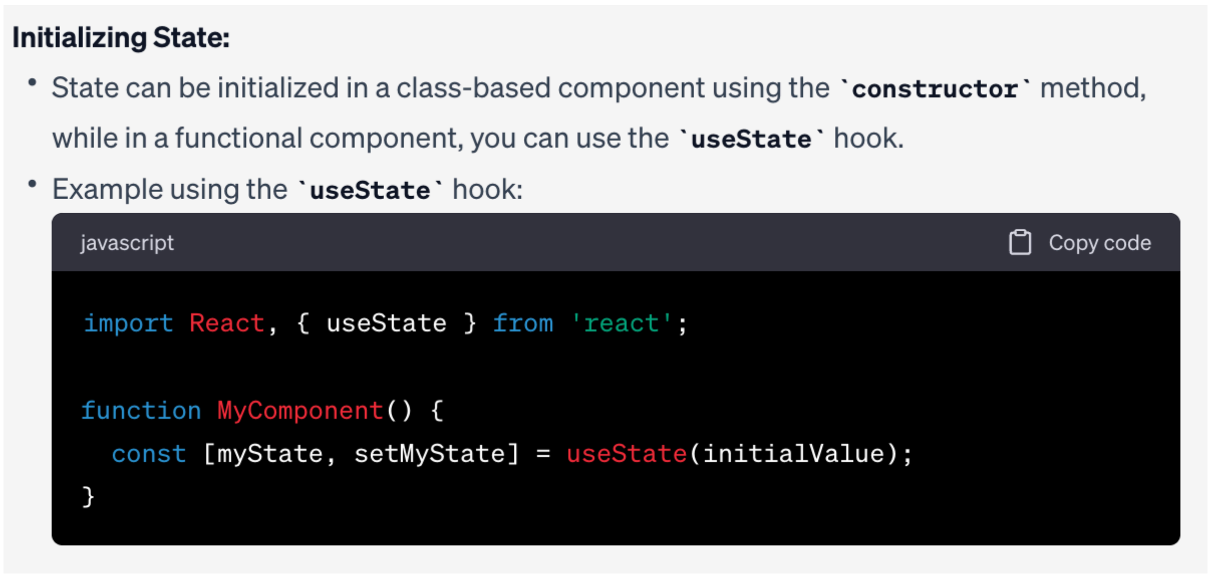
Sunday, 3 September 2023

States & Mapping

- States are the special variables which can hold any kind of values like integer, string, boolean, arrays & objects. States are known as the state of your component and whenever the state of component changes the whole component in which the state is created is reloaded again.



Initializing State:

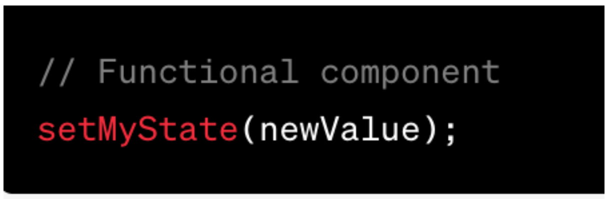
- State can be initialized in a class-based component using the `constructor` method, while in a functional component, you can use the `useState` hook.
- Example using the `useState` hook:

```javascript
import React, { useState } from 'react';

function MyComponent() {
  const [myState, setMyState] = useState(initialValue);
}
```

States are treated as variables but we cannot change their value directly as we do in the variables. The function associated with state only can change the state. As in above example setMyState function can only change myState.

We just have to call that function and in the parameter we have to pass a new value for state. Whenever the state changes the component will reload again.



```javascript
// Functional component
setMyState(newValue);
```

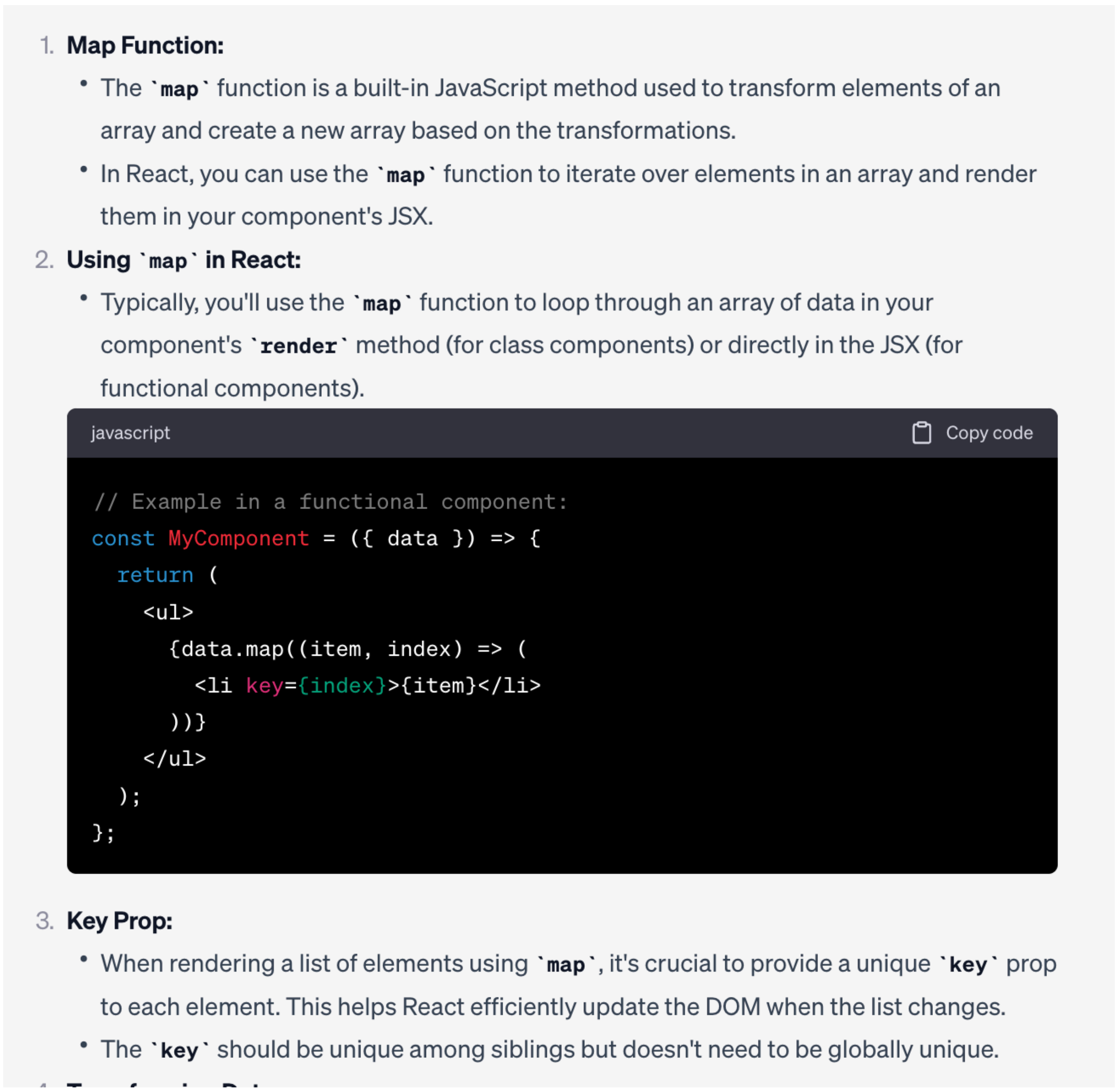
Mapping

Sometimes we have some set of values in the state (that is an array) and we want to render a single component for all the values in the array.

For example let us suppose that we have an state “todo” that is array. Todo is containing all the todos in the form of object. And we have component TodoItem that will take all the data in props and show a single item to you.

Now you wanted to render this component for all the elements in the array. If 10 todos are in array then you want to show TodoItem component 10 times for each element of array.

To achieve this we use “map” function on the array. Map is higher order function that will pass “item”,”index” to provide function and it will map the given component with each element of array.



1. **Map Function:**
   - The `map` function is a built-in JavaScript method used to transform elements of an array and create a new array based on the transformations.
   - In React, you can use the `map` function to iterate over elements in an array and render them in your component's JSX.
2. **Using `map` in React:**
   - Typically, you'll use the `map` function to loop through an array of data in your component's `render` method (for class components) or directly in the JSX (for functional components).

javascript Copy code

```javascript
// Example in a functional component:
const MyComponent = ({ data }) => {
  return (
    <ul>
      {data.map((item, index) => (
        <li key={index}>{item}</li>
      ))}
    </ul>
  );
};
```

3. **Key Prop:**
   - When rendering a list of elements using `map`, it's crucial to provide a unique `key` prop to each element. This helps React efficiently update the DOM when the list changes.
   - The `key` should be unique among siblings but doesn't need to be globally unique.