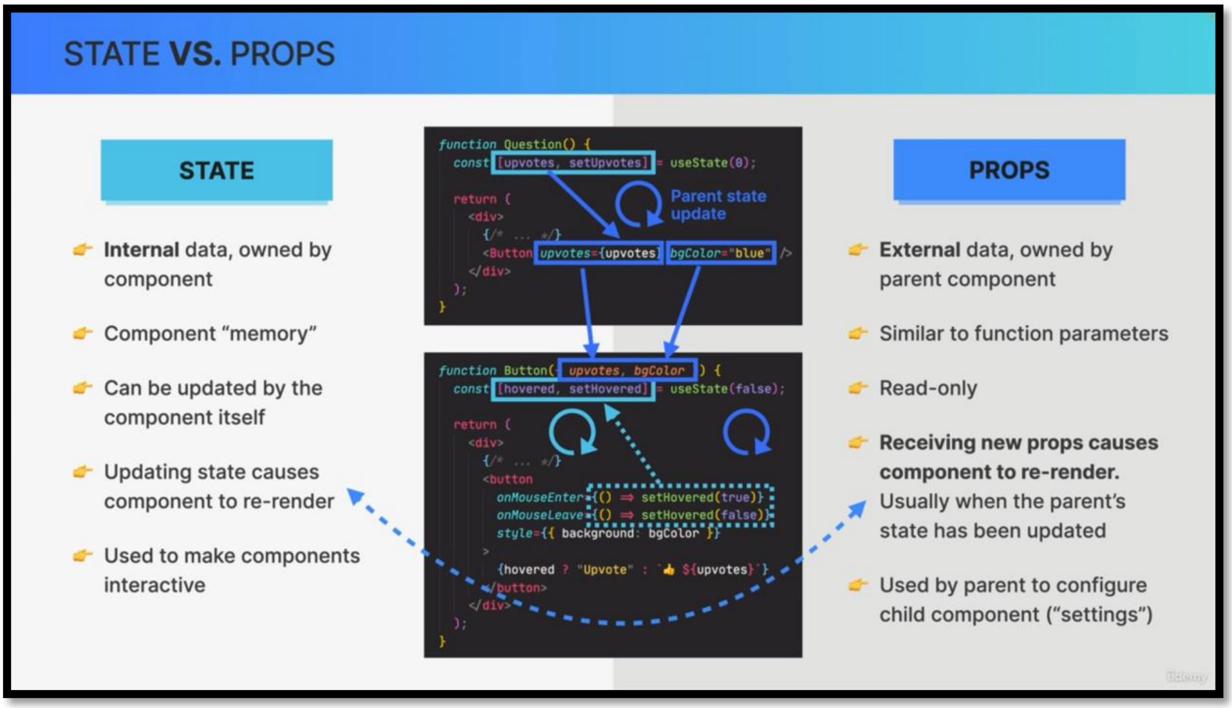

STATE VS. PROPS

STATE

- **Internal** data, owned by component
- Component "memory"
- Can be updated by the component itself
- Updating state causes component to re-render
- Used to make components interactive

```
function Question() {
  const [upvotes, setUpvotes] = useState(0);

  return (
    <div>
      {/* ... */}
      <Button upvotes={upvotes} bgColor="blue" />
    </div>
  );
}
```

Parent state update

```
function Button({ upvotes, bgColor }) {
  const [hovered, setHovered] = useState(false);

  return (
    <div>
      {/* ... */}
      <button
        onMouseEnter={() => setHovered(true)}
        onMouseLeave={() => setHovered(false)}
        style={{ background: bgColor }}
      >
        {hovered ? "Upvote" : `👍 ${upvotes}`}
      </button>
    </div>
  );
}
```

PROPS

- **External** data, owned by parent component
- Similar to function parameters
- Read-only
- **Receiving new props causes component to re-render.** Usually when the parent's state has been updated
- Used by parent to configure child component ("settings")

1. State is internal data that is owned by the component in which it is declared. Props is external data that is owned by the parent component.

2. State can be thought of as the component's memory because it can hold data over time and across multiple re-renders. We can think of props as function parameters or as a communication channel between parent and child components where parents can pass data into children.

3. State can be updated by the component itself. Props are read only. So, they cannot be modified by the component that is receiving them.

4. Updating state cause the component to be re-rendered by React. When the child component receives new updated props, that will actually also cause the component to re-render.

5. The mechanism of state is used to make components interactive. Props are used to give the parent component the ability to configure their child components. Props can be seen as settings in child components, which the parent component can define.