

So we learned how to use event handlers, but now we want them to actually do something useful. Right. So we want to make the component interactive. And for that, as I have mentioned already, we need state. Now, without a doubt, state is the most important concept in React. So everything basically revolves around state in React. And so we will keep learning about state throughout the entire course. Therefore, let's start with an overview of what exactly you will learn about State while going through this course. First, we will learn what state actually is, what it does and why we need it, which is what this section is all about. Then we need to learn how to actually use state in practice using the use state or Usereducer hooks, the context API or even external tools like Redux. We will also need to deeply understand how to think about State in React. And so these are topics for future sections. Okay. And with this out of the way, we're now ready to learn what state actually is. So we have learned how to pass data into a component by using props, which remember is data that's coming from outside the component. But what if a component needs to actually hold its own data and also hold it over time? Also, what if we actually want to make our app interactive, changing the UI as a result of an action? Well, that's where finally State comes into play. So state is basically data that a component can hold over time, and we use it for information that the component needs to remember throughout its lifecycle. Therefore, we can think of state as being the memory of a component. So that can be quite a helpful analogy, I think. Now, examples of state can be simple things like a notification count, the text content of an input field or the active tab in a tabbed component. It can also be a bit more complex data, for example, the content of a shopping cart. Now what all these pieces of state have in common is that in the application the user can easily change these values. For example, when they read a notification, the count will go down by one, or when they click on another tab, that tab will become active. And therefore, each of these components needs to be able to hold this data over time. So over the lifecycle of the application. And for that reason, each of these pieces of information is a piece of state. And notice how I use the term piece of state here, because just the term state itself is more of a general term. So a piece of state or a state variable is just one single actual variable in the component that we can define in our code. On the other hand, the term state itself is more about the entire state that the component is in, like the entire condition at a certain point in time. So basically the general term state is all the pieces of state together. And if this sounds confusing, don't worry, these are just some minor differences in terminology. In practice, we usually use the terms state piece of state and state variable quite interchangeably. But anyway, let's now move on to the most important aspect of state, which is the fact that updating state triggers react to rerender the component. So again, whenever we update a piece of state in a component, this will make react rerender that component in the user interface. So it will create a new updated view for that component. And a component view is basically just the component visually rendered on the screen. So in the user interface. Now, up until this point, I have always just used the generic term user interface, but now we are actually talking about a single component. And when one single component is rendered, we call that a view. And so all the views combined together then make up the final user interface. Now, do you remember this small diagram that we saw right at the beginning of the course when we first talked about how React automatically keeps data in sync with the UI? Well, state is how react does that. So state is how react keeps the user interface in sync with data. We change the state. We change the UI. So summarizing state allows developers to do two important things. First, state allows us to update the component's view by rerendering the component. So it gives us a way to change parts of the UI. Second State allows developers to persist local variables between multiple renders and rerenders. So if you think about this state is basically a tool and in fact it's the most powerful tool that we have in the world of React. So understanding how state works and what it does. So understanding the mechanics of state will unlock the power of react development for you. But before we go, understand the mechanics of state, let's actually first go back to our code and use this powerful tool in practice for the first time.