# Child Components Update Their Parents' state

In the last lesson, you passed information from a stateful, parent component to a stateless, child component.

In this lesson, you'll be expanding on that pattern. The stateless, child component will update the state of the parent component.

Here's how that works:

1

The parent component class defines a method that calls this.setState().

For an example, look in **Step1.js** at the .handleClick() method.

2

The parent component binds the newly-defined method to the current instance of the component in its constructor. This ensures that when we pass the method to the child component, it will still update the parent component.

For an example, look in **Step2.js** at the end of the constructor() method.

3

Once the parent has defined a method that updates its state and bound to it, the parent then passes that method down to a child.

Look in **Step2.js**, at the prop on line 28.

4

The child receives the passed-down function, and uses it as an event handler.

Look in **Step3.js**. When a user clicks on the <button></button>, a click event will fire. This will make the passed-down function get called, which will update the parent's state.

Click through the three files in order, and try to follow their chronology. Whenever you're ready, click Next and we'll build an example!

# Define an Event Handler

To make a child component update its parent's state, the first step is something that you've seen before: you must define a state-changing method on the parent.

# Pass the Event Handler

In the last exercise, you defined a function in Parent that can change Parent's state.

Parent must pass this function down to Child, so that Child can use it in an event listener on the dropdown menu.

# Receive the Event Handler

You just passed a function down to Child that can change Parent's state!

# Automatic Binding

Great work! Stateless components updating their parents' state is a React pattern that you'll see more and more. Learning to recognize it will help you understand how React apps are organized.

Click Next to move on to the final version of our programming pattern!