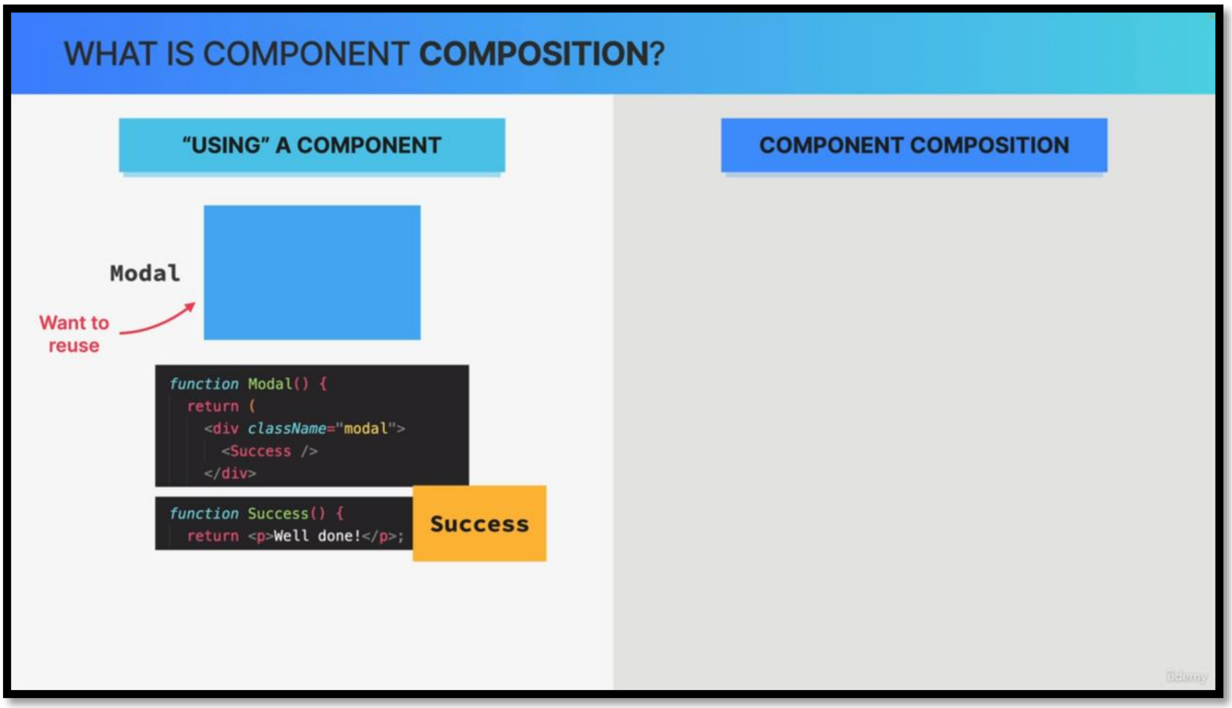


WHAT IS COMPONENT **COMPOSITION?**

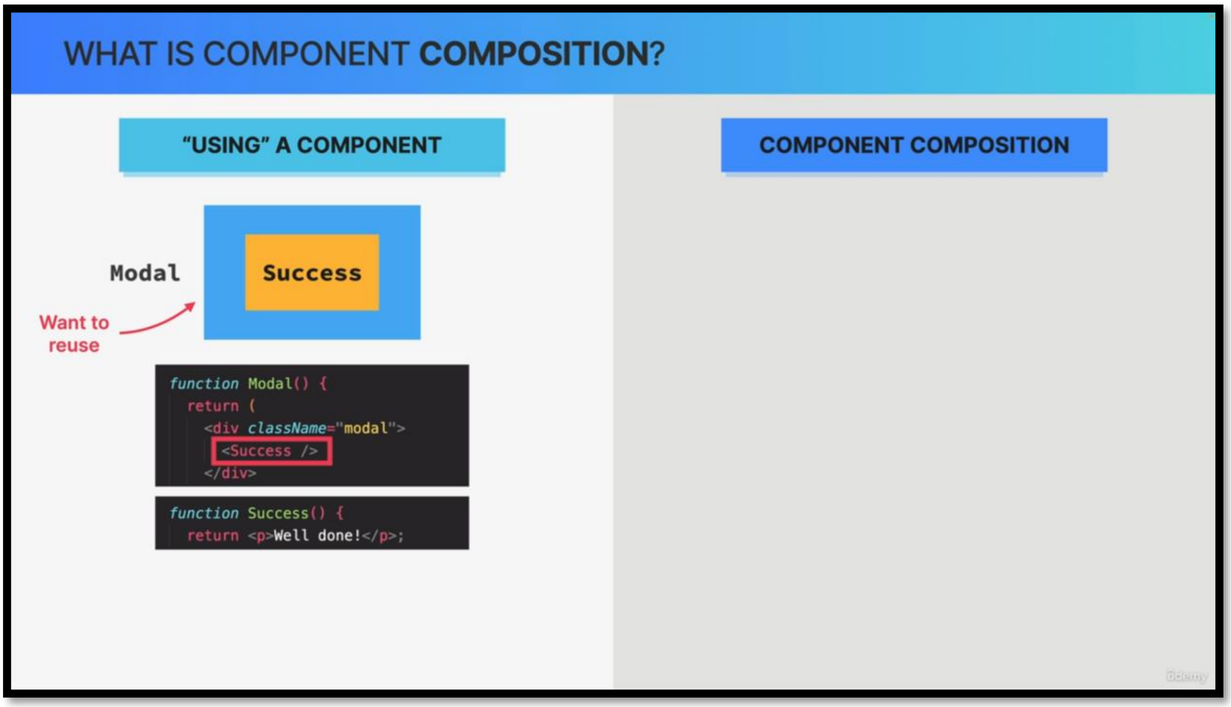
```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

What happens when we simply use or include a component in another component in JSX?

Let's say that we have this model component that we want to reuse, and also this success component which simply renders the message well done.

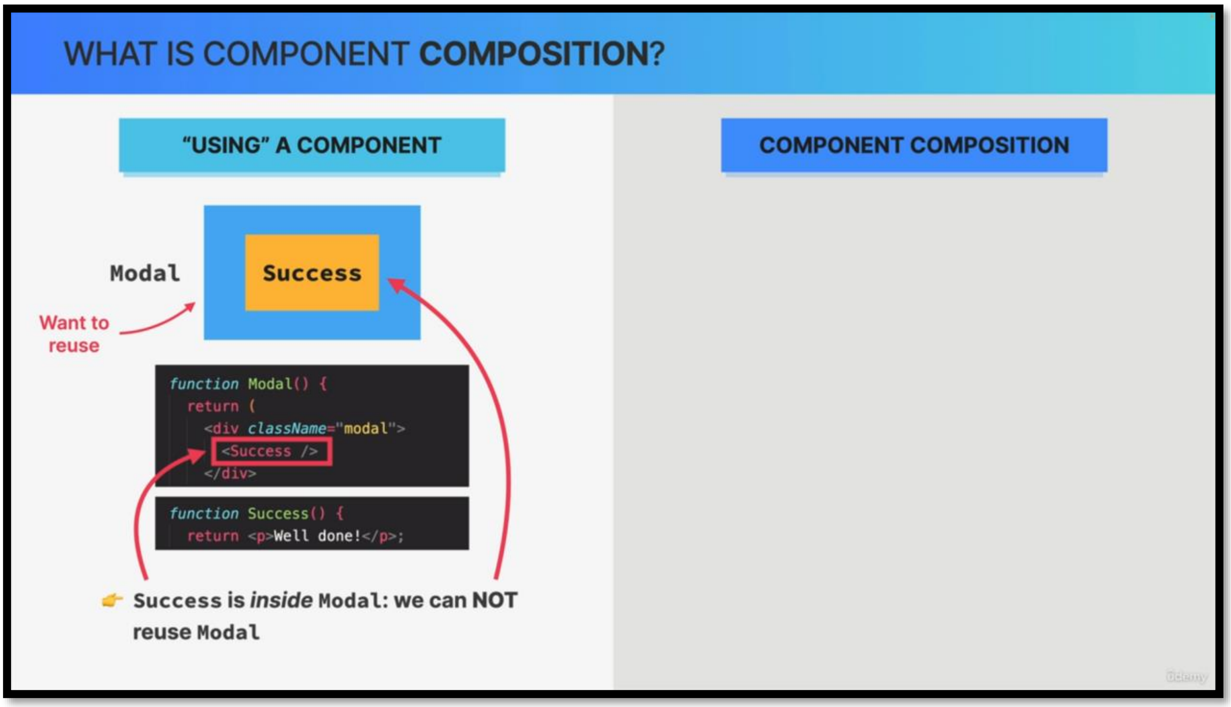
We just use the success component inside the modal component. This sort of thing is exactly what we have been doing with our components most of the time. So, we just use them inside of other components.



WHAT IS COMPONENT **COMPOSITION?**

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```



WHAT IS COMPONENT **COMPOSITION?**

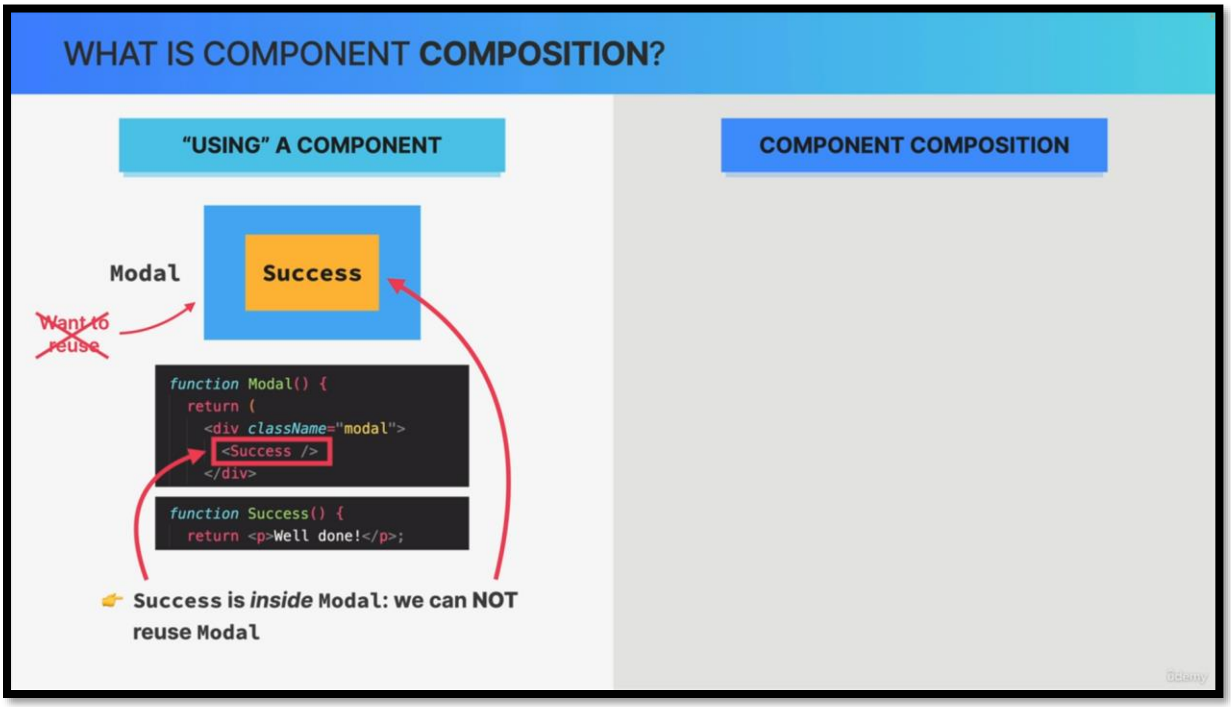
"USING" A COMPONENT

COMPONENT COMPOSITION

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

👉 `Success` is *inside* `Modal`: we can NOT reuse `Modal`



WHAT IS COMPONENT **COMPOSITION?**

"USING" A COMPONENT

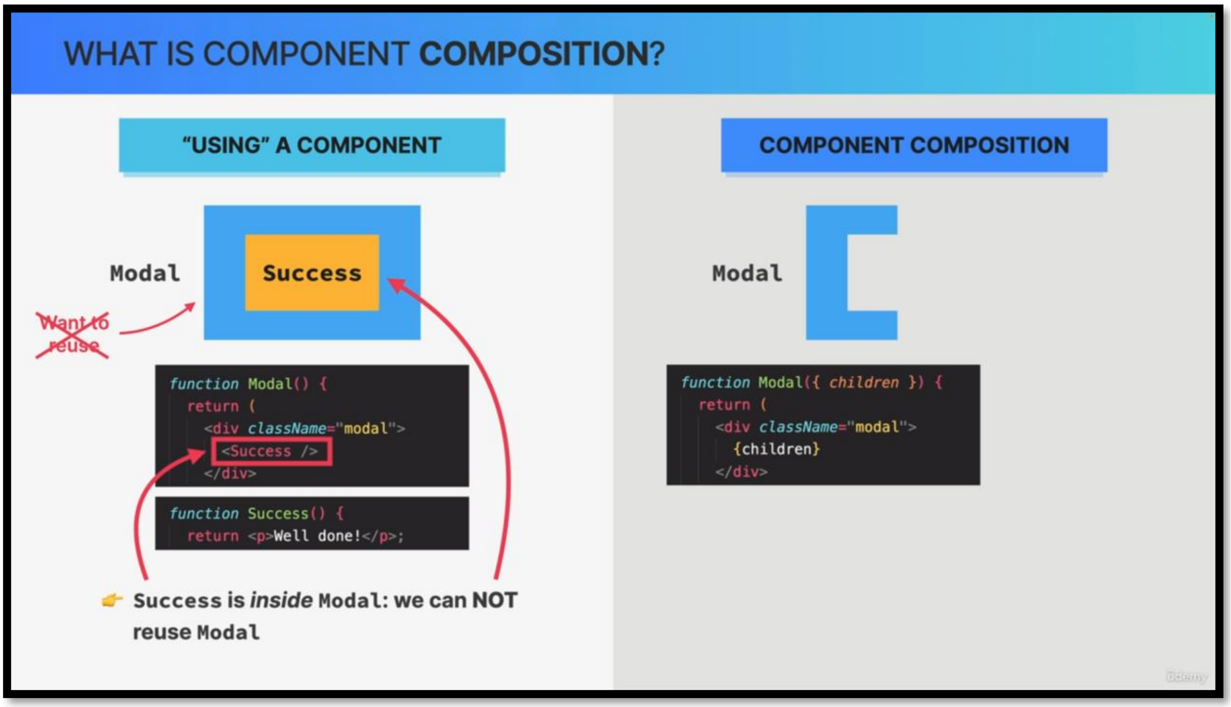
COMPONENT COMPOSITION

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

👉 `Success` is *inside* `Modal`: we can NOT reuse `Modal`

However, when it comes to re-usability this creates a big problem. That's because right now the success component really is inside of modal. They're deeply linked together in the JSX right now, and therefore we cannot reuse this modal component to display some other message inside of it, for example, an error message.



WHAT IS COMPONENT COMPOSITION?

"USING" A COMPONENT

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

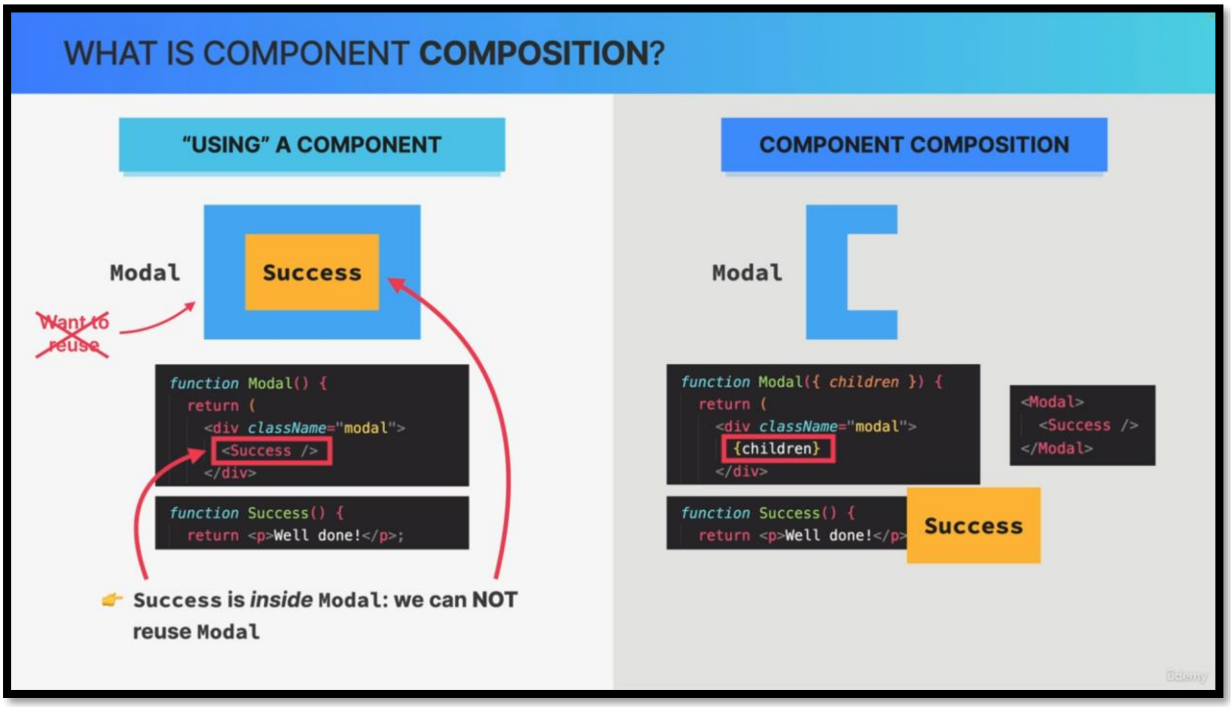
```
function Success() {
  return <p>Well done!</p>;
```

👉 Success is *inside* Modal: we can NOT reuse Modal

COMPONENT COMPOSITION

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

In order to solve this, we now bring in the technique of component composition where we can compose the model and success components together.



WHAT IS COMPONENT COMPOSITION?

"USING" A COMPONENT

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

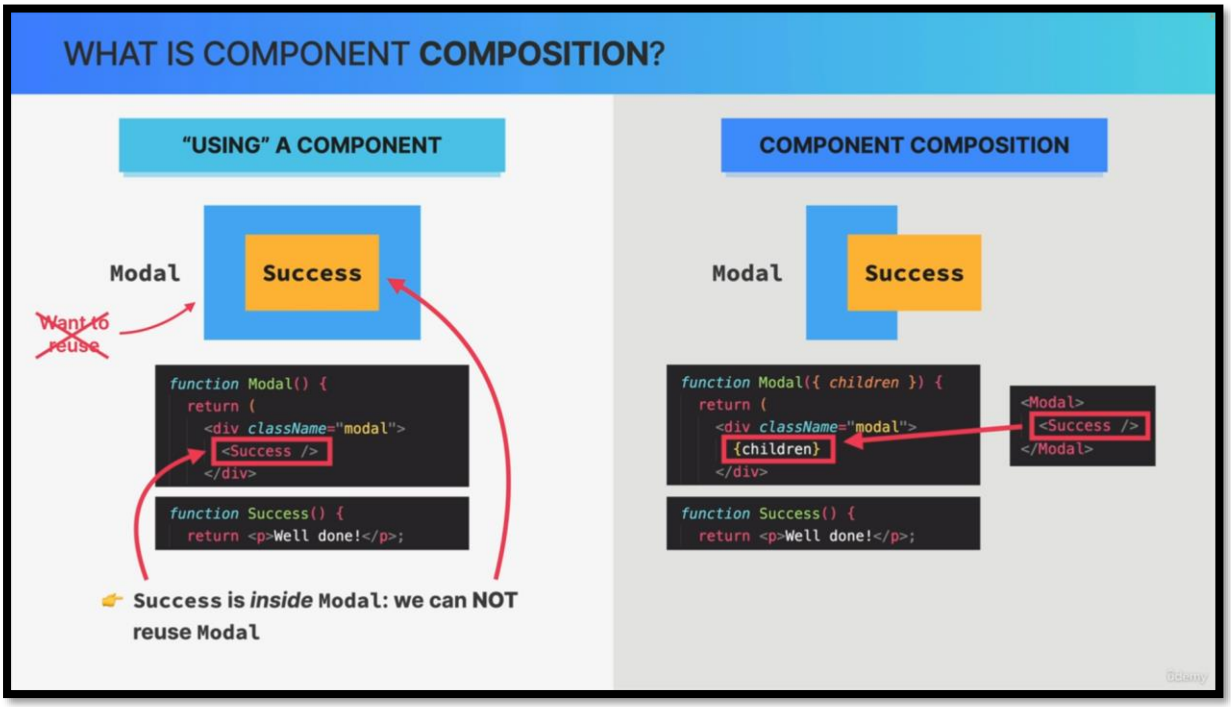
👉 Success is *inside* Modal: we can NOT reuse Modal

COMPONENT COMPOSITION

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
<Modal>
  <Success />
</Modal>
```

```
function Success() {
  return <p>Well done!</p
```



WHAT IS COMPONENT **COMPOSITION?**

"USING" A COMPONENT

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

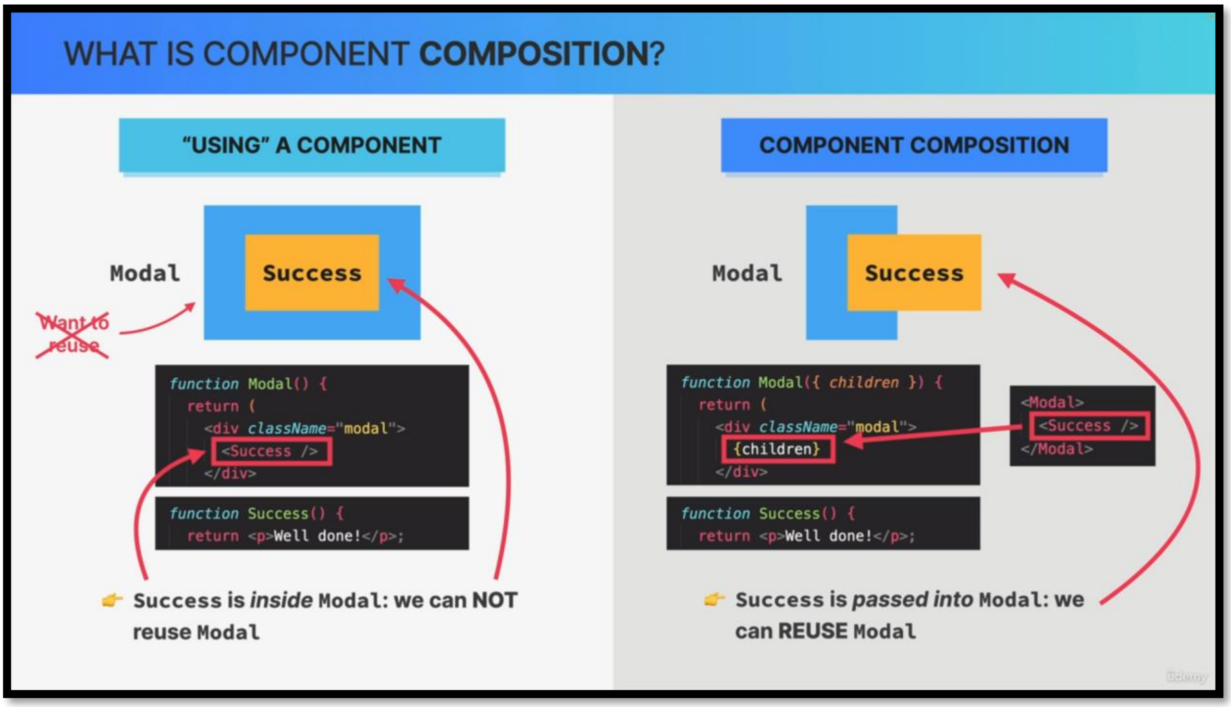
Success is *inside* Modal: we can NOT reuse Modal

COMPONENT COMPOSITION

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

```
<Modal>
  <Success />
</Modal>
```



WHAT IS COMPONENT **COMPOSITION?**

"USING" A COMPONENT

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

Success is *inside* Modal: we can NOT reuse Modal

COMPONENT COMPOSITION

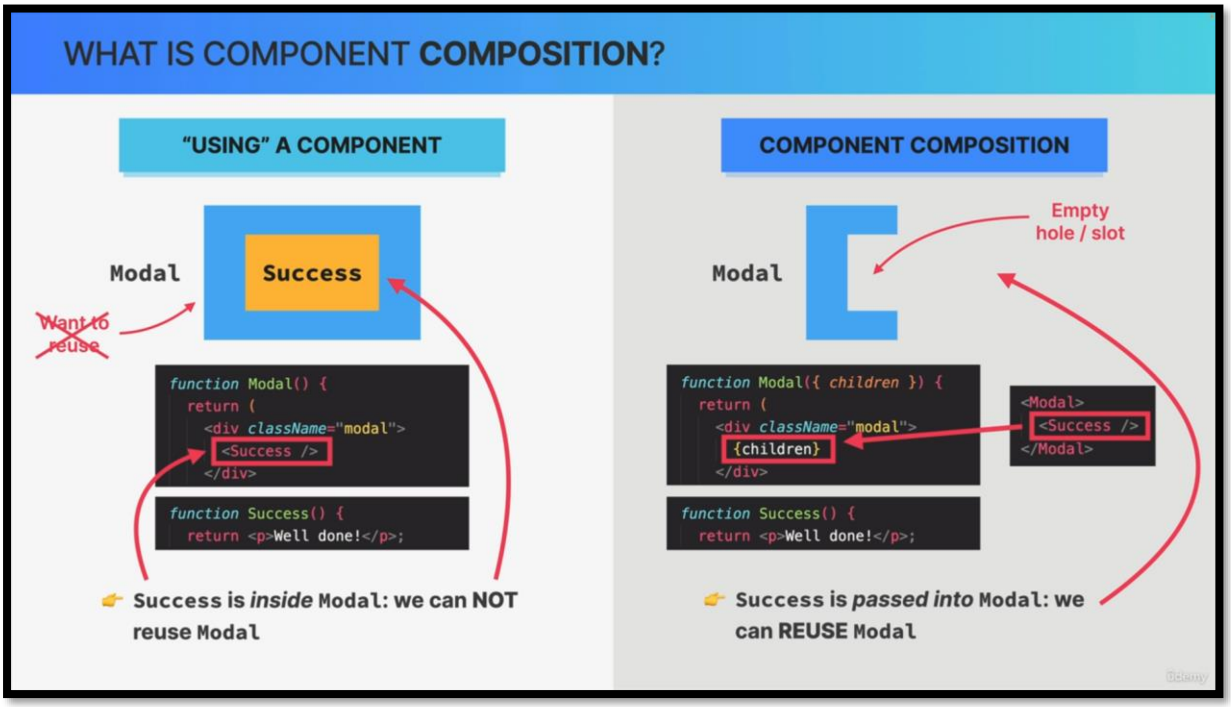
```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

```
<Modal>
  <Success />
</Modal>
```

Success is *passed into* Modal: we can REUSE Modal

So here we have our modal component again, but with a fundamental difference. So, this component does not include a predefined component but instead it accepts children with the children prop.



WHAT IS **COMPONENT COMPOSITION?**

"USING" A COMPONENT

COMPONENT COMPOSITION

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

Success is *inside* Modal: we can NOT reuse Modal

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

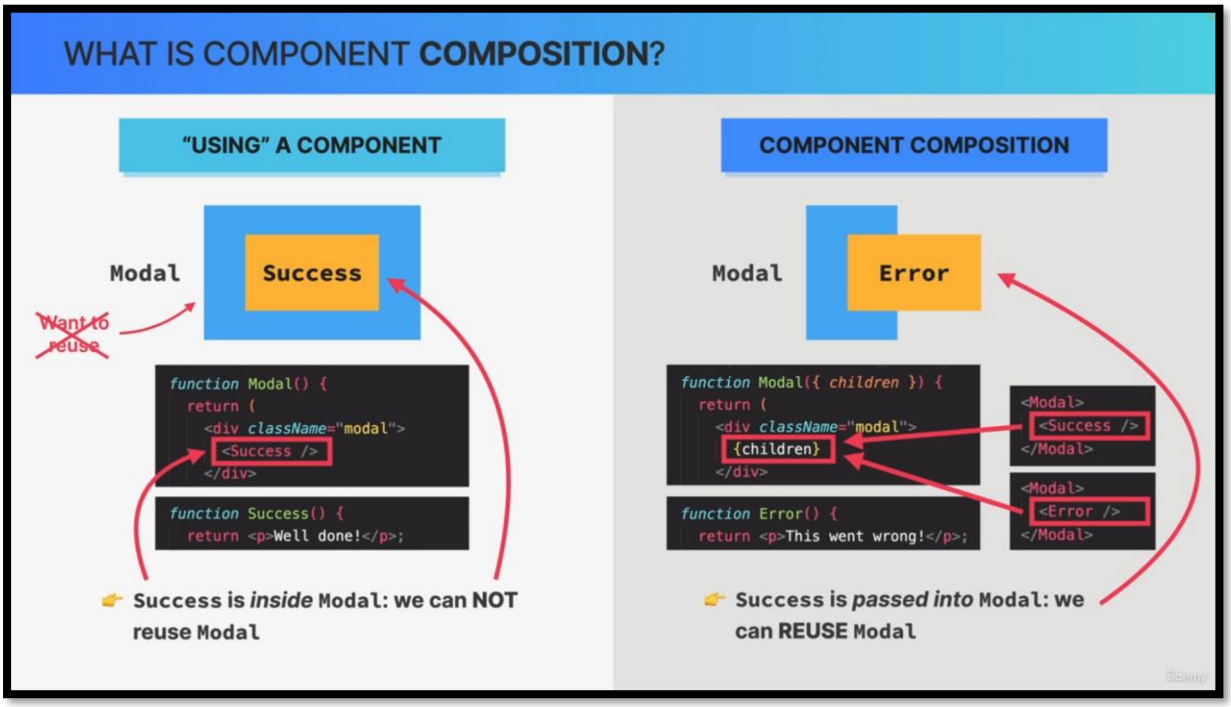
```
function Success() {
  return <p>Well done!</p>;
```

```
<Modal>
  <Success />
</Modal>
```

Success is *passed into* Modal: we can REUSE Modal

If we get our success component again, we can now basically just pass it into the modal by placing it between the opening and closing tags when we use modal.

We could have passed in any other component which makes the model component highly reusable. So essentially when we do component composition, we leave this hole or this empty slot in the component ready to be filled with any other component that we want.



WHAT IS **COMPONENT COMPOSITION?**

"USING" A COMPONENT

COMPONENT COMPOSITION

```
function Modal() {
  return (
    <div className="modal">
      <Success />
    </div>
```

```
function Success() {
  return <p>Well done!</p>;
```

Success is *inside* Modal: we can NOT reuse Modal

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
function Error() {
  return <p>This went wrong!</p>;
```

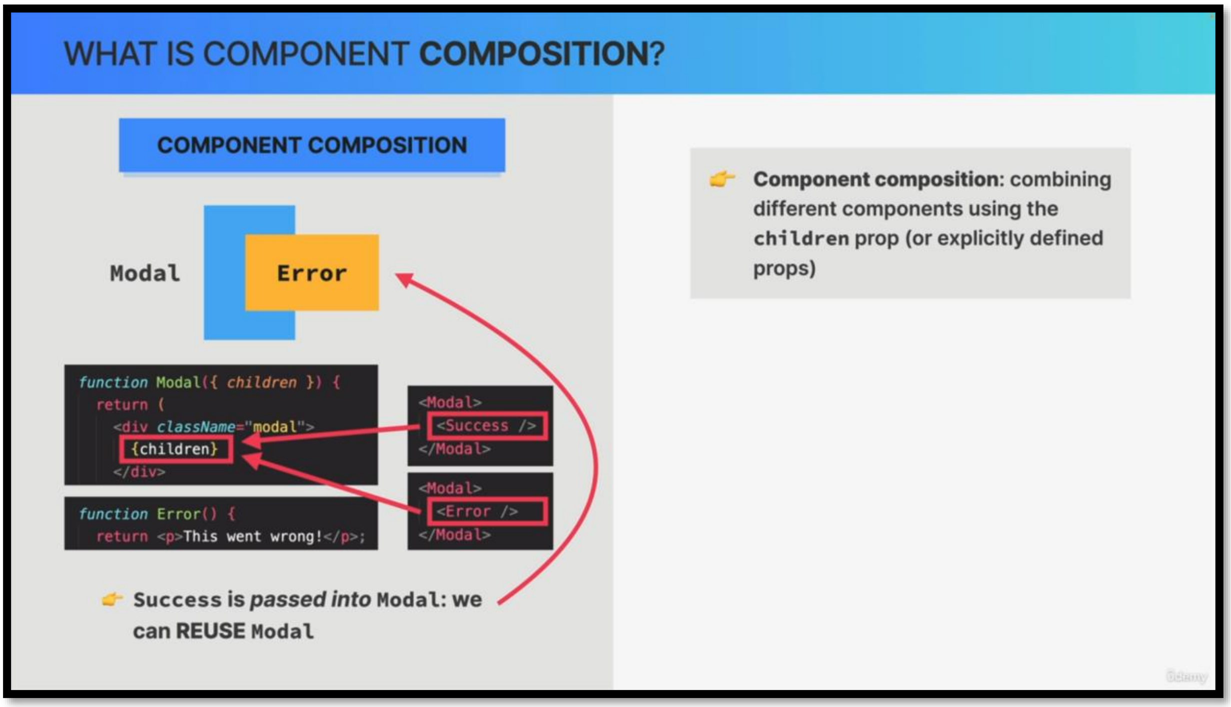
```
<Modal>
  <Success />
</Modal>
```

```
<Modal>
  <Error />
</Modal>
```

Success is *passed into* Modal: we can REUSE Modal

Let's say that later we needed another model window somewhere else in the app but one that renders this error message. Well, that's pretty easy now.

We just used the model component again but this time we pass in the error component as a children. With this, we have also successfully composed these two components together as well.



WHAT IS COMPONENT **COMPOSITION?**

COMPONENT COMPOSITION

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
function Error() {
  return <p>This went wrong!</p>;
```

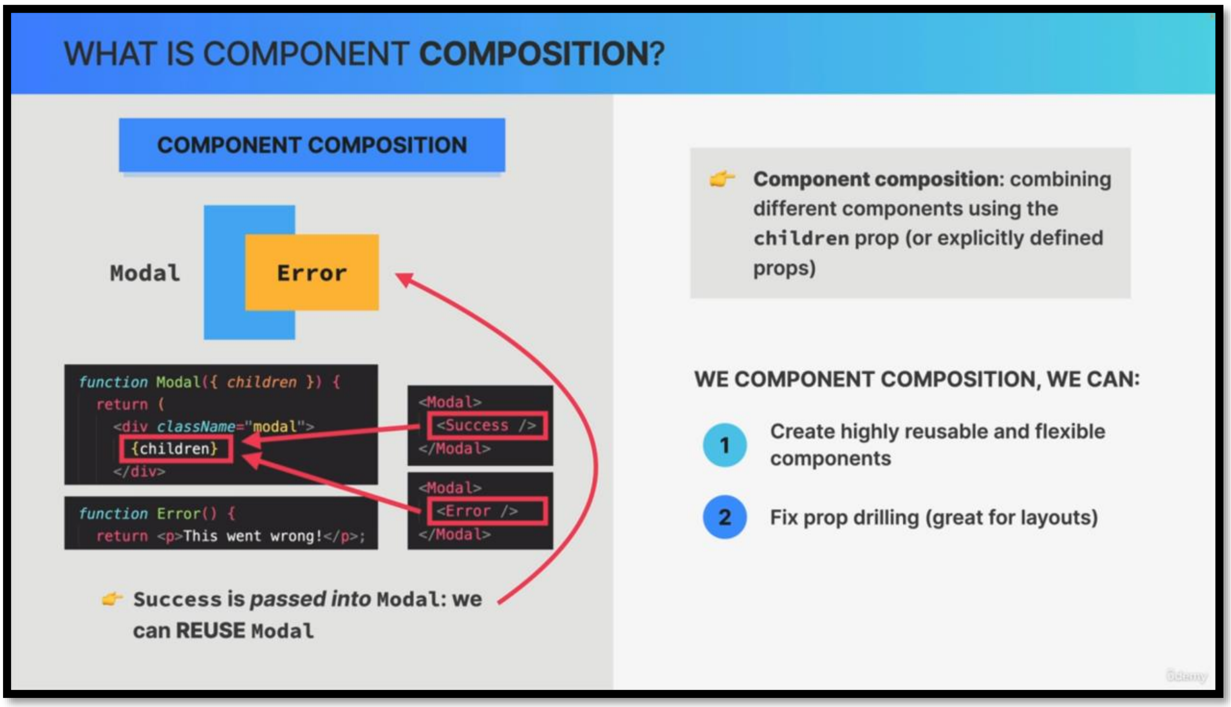
```
<Modal>
  <Success />
</Modal>
```

```
<Modal>
  <Error />
</Modal>
```

👉 Success is *passed into* Modal: we can REUSE Modal

👉 **Component composition:** combining different components using the children prop (or explicitly defined props)

Formally component composition is the technique of combining different components by using the children prop or by explicitly defining components as props.



WHAT IS COMPONENT **COMPOSITION?**

COMPONENT COMPOSITION

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
function Error() {
  return <p>This went wrong!</p>;
```

```
<Modal>
  <Success />
</Modal>
```

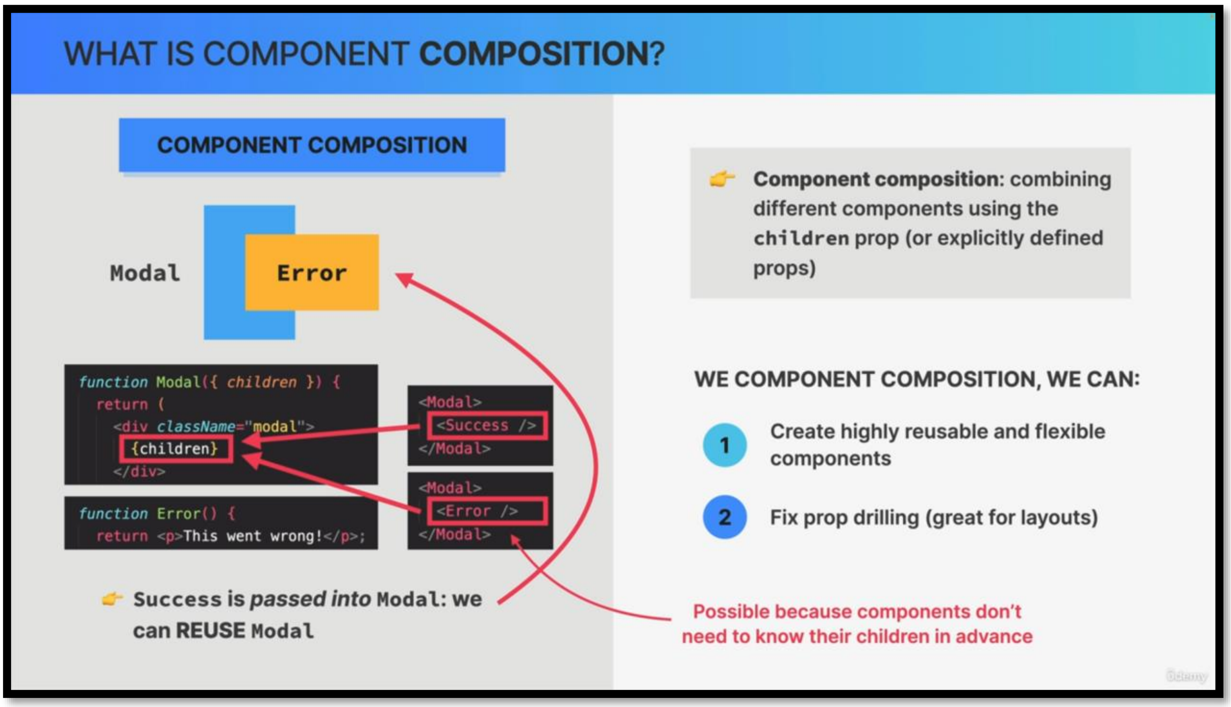
```
<Modal>
  <Error />
</Modal>
```

👉 Success is *passed into* Modal: we can REUSE Modal

👉 **Component composition:** combining different components using the children prop (or explicitly defined props)

WE COMPONENT COMPOSITION, WE CAN:

1. Create highly reusable and flexible components
2. Fix prop drilling (great for layouts)



WHAT IS COMPONENT **COMPOSITION?**

COMPONENT COMPOSITION

```
function Modal({ children }) {
  return (
    <div className="modal">
      {children}
    </div>
```

```
function Error() {
  return <p>This went wrong!</p>;
```

```
<Modal>
  <Success />
</Modal>
```

```
<Modal>
  <Error />
</Modal>
```

👉 Success is *passed into* Modal: we can REUSE Modal

👉 **Component composition:** combining different components using the children prop (or explicitly defined props)

WE COMPONENT COMPOSITION, WE CAN:

1 Create highly reusable and flexible components

2 Fix prop drilling (great for layouts)

Possible because components don't need to know their children in advance

We use composition for two big reasons or in two important situations.

First, when we want to create highly reusable and flexible components such as the modal window or really a million other reusable components that we can think of.

The second situation in which we can use composition is in order to fix a prop drilling problem.