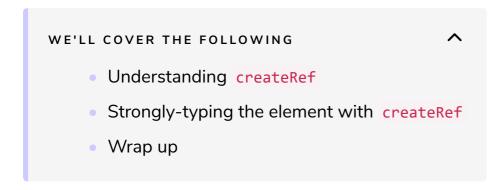
Creating strongly-typed refs in class components

In this lesson, we learn how to get a strongly-typed reference to an element in a class component.



Understanding createRef

In this lesson, we are going to implement the Search component we implemented in the last lesson as a class component. Below is a first attempt:

```
class Search extends React.Component {
  private input = React.useRef<HTMLInputElement>(null);

componentDidMount() {
  if (this.input.current) {
    this.input.current.focus();
  }
}

render() {
  return (
    <form>
        <input ref={this.input} type="type" />
        </form>
    );
}
```

What is wrong with this implementation?



A. Sllow Allswei

In class components, we can get a reference to an element using createRef. We can use this to revise our implementation of the Search component:

What do you think the type of the input.current property has been inferred
as?



Strongly-typing the element with createRef

We can explicitly define the type of the element returned from createRef by passing a generic type parameter:

```
React.createRef<ElementType>();
```

A revised, more strongly-typed version of the Search component is below. If you run it, you will see that the focus is set on the input after it renders.

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
class Search extends React.Component {
 private input = React.createRef<HTMLInputElement>();
 componentDidMount() {
   if (this.input.current) {
     this.input.current.focus();
 }
 render() {
   return (
     <form>
       <input ref={this.input} type="type" />
   );
 }
ReactDOM.render(
   <Search />,
   document.getElementById("root")
);
```

Wrap up

When using the createRef function, the type of the element being referenced should always be passed into its generic parameter. This will ensure the reference is strongly-typed.

Next up we'll check our understanding of how to get strongly-typed references to elements within function and class components in a quiz.