Creating strongly-typed events

Often components need to expose event props. In this lesson, we'll learn how to implement event props that are strongly-typed.

WE'LL COVER THE FOLLOWING

Creating an event prop

We are going to continue the implementation of a `Searchbox` component we were working on in the last lesson.

Click the link below to open the exercise in CodeSandbox:

CodeSandbox project

The `Searchbox` component is at the point where we finished in the last lesson. We are going to create an optional event prop called `onSearch` that is triggered when the search criteria are changed. This will be a function prop that has the criteria as a parameter.

An example consumption of the `onSearch` prop is as follows:

```
<Searchbox onSearch={criteria => console.log(criteria)} />
```

Implement the `onSearch` prop in the `Searchbox` component. This is a function prop, so, remind yourself how we implemented types for function props in the *Creating strongly-typed function components props* lesson.

Show Answer

So, the type for event props is:

```
(param1: type1, param2: type2, ...) => void
```

Notice that the return type is `void,` which signifies nothing is returned from the function.

Wrap up

Events are exposed in a component using a function prop. We can type the function parameters appropriately, and generally the function will have a return type of `void`.

Excellent, we can now consume and create events in a strongly-typed manner!

Next, let's double-check what we have learned from the last couple of lessons with a quiz.