Quiz

In this quiz, we will test what we have learned about strongly-typed function component props with TypeScript

Creating strongly-typed function props

1

Below is a component for a red button:

```
const RedButton = ({ children }) => (
  <button style={{ backgroundColor: "red" }}>{children}</button>
);
```

What type annotation can we use to strongly type the props of the button? (There is more than one correct answer).

2

The component below renders a label in a specified color. How can we use strongly-typed props so that only `"blue"` or `"red"` can be passed into the color prop?

```
const Label = ({ children, color }) => (
  <label style={{ color }}>{children}</label>
);
```

3

We have a button component below that has a `size` prop to specify the height of the button. How can we allow consumers of the component not to specify the `size` prop and have it default to `"small"`?

```
type Props = { size: "small" | "large", children: React.ReactNode };
const Button = ({ children, size }: Props) => (
  <button style={{ height: size === "small" ? "20px" : "30px"
 }}>
    {children}
  </button>
);
```

4

We have a card component below that shows the name and email address of a person:

```
const PersonCard = ({ color, person }) => (
  <div style={{ border: `1px solid ${color}` }}>
    <h3>{person.name}</h3>
    <a href={`mailto: ${person.email}`}>{person.email}</a>
  </div>
);
```

How can we strongly-type the props for this component?

5

We have a `List` component that can be consumed as follows:

```
<List
  data={["Fred", "Bob", "Jane"]}
  renderItem={item => (
    <div>
      <span>{item}</span>
      <button>Do something</button>
    </div>
  )}
/>
```

What would strongly-typed props for this component be?

Check Answers

Excellent, that’s another chapter complete!

Next, we’ll learn how to strongly-type React function component states.