Quiz

This quiz will test your knowledge on using React's context in a strongly typed manner in both function and class components.

Creating strongly-typed context

1

We are implementing a context for authentication. What is wrong with the implementation below and how can this be corrected?

```
type AuthContextType = {
  isAuthenticated: boolean;
  userName?: string;
  signIn: (userName: string) => void;
  signOut: () => void;
};
const AuthContext = React.createContext(undefined);
```

2

If we have a context called `AuthContext` that is of type `AuthContextType | undefined`, how can we create a strongly-typed custom hook to get access to the context?

3

If we have a function component and need to access a context in the components JSX, which approaches could we use? (tick all that apply).

4

We are implementing a class component and are accessing a context in its `componentDidMount` method as follows:

```
componentDidMount() {
  const isAuthenticated = this.context!.isAuthenticated;
  ...
}
```

The type of `isAuthenticated` is `any`. How should we resolve this?

Check Answers

Next up, we are going to learn how to strongly-type React component refs.