Version: 6.x

Custom Android back button behavior

By default, when user presses the Android hardware back button, react-navigation will pop a screen or exit the app if there are no screens to pop. This is a sensible default behavior, but there are situations when you might want to implement custom handling.

As an example, consider a screen where user is selecting items in a list, and a "selection mode" is active. On a back button press, you would first want the "selection mode" to be deactivated, and the screen should be popped only on the second back button press. The following code snippet demonstrates the situation. We make use of `BackHandler` which comes with react-native, along with the `useFocusEffect` hook to add our custom `hardwareBackPress` listener.

Returning `true` from `onBackPress` denotes that we have handled the event, and react-navigation's listener will not get called, thus not popping the screen. Returning `false` will cause the event to bubble up and react-navigation's listener will pop the screen.

```
function ScreenWithCustomBackBehavior() {
  // ...

  useFocusEffect(
    React.useCallback(() => {
      const onBackPress = () => {
        if (isSelectionModeEnabled()) {
          disableSelectionMode();
          return true;
        } else {
          return false;
        }
      };

      const subscription = BackHandler.addEventListener('hardwareBackPress',
onBackPress);

      return () => subscription.remove();
    }, [isSelectionModeEnabled, disableSelectionMode])
  );
```

```
  // ...
}
```

Try this example on Snack

The presented approach will work well for screens that are shown in a `StackNavigator`. Custom back button handling in other situations may not be supported at the moment (eg. A known case when this does not work is when you want to handle back button press in an open drawer. PRs for such use cases are welcome!).

If instead of overriding system back button, you'd like to prevent going back from the screen, see docs for preventing going back.

Why not use component lifecycle methods

At first, you may be inclined to use `componentDidMount` to subscribe for the back press event and `componentWillUnmount` to unsubscribe, or use `useEffect` to add the listener. This approach will not work - learn more about this in navigation lifecycle.

Edit this page