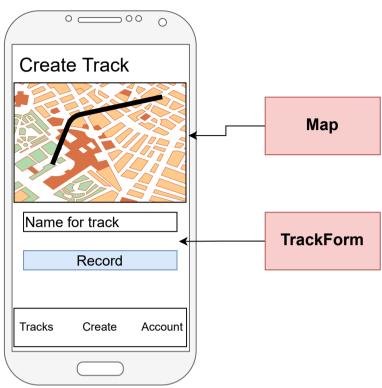
TrackCreateScreen



Expo-location

Before tracking the users location, we need to ask for permission

Two forms of location tracking - foreground and background

We need to be extremely aware of when we're tracking location - it consumes battery power!

Foreground Location Tracking

Gives us users location whenever our app is visible on the screen

Easy to set up + use

Background Location Tracking

Gives us the users location at all times, even if the app is not visible or device is locked

Much more complicated to set up

Uses considerably more battery power

TrackCreateScreen

Create Track

When a user comes to this screen, we need to start tracking their location right away, even if they are not actively recording a track

When a user navigates away, we need to stop tracking their location unless they are recording a track

Record

Tracks Create Account

npx expo-cli install expo-location

Installing with expo-cli makes sure we get a version of expo-location that is compatible with our React Native + Expo versions

Location Permissions

TrackCreateScreen appears

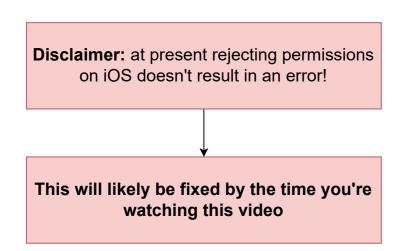
Prompt user for permission to track their location

Permissions request is async. If user rejects request, the call throws an error.

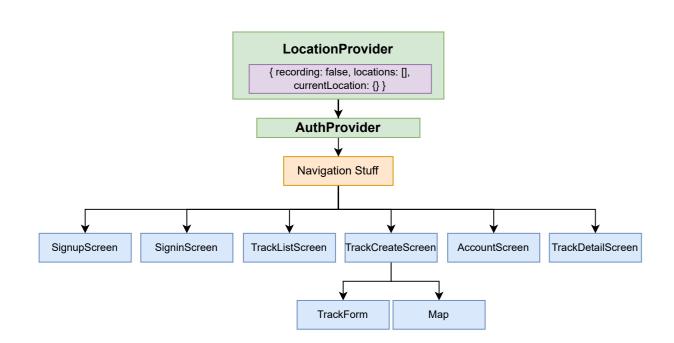
Show error if the user denies permission, otherwise start tracking

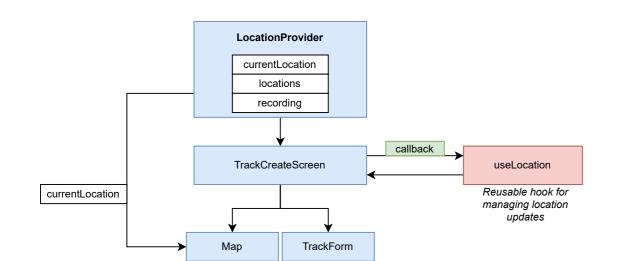
If user returns to TrackCreateScreen, we will rerun permissions code, but the OS will automatically approve/reject us

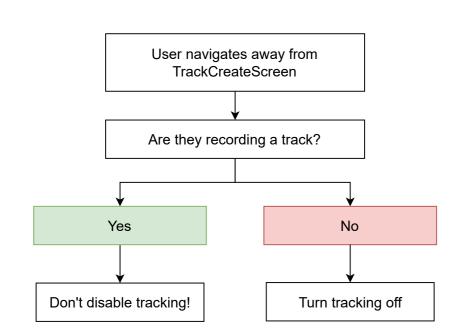
...later



Resetting Permissions for Testing Settings > General > Reset > Reset Location and Privacy 'adb shell pm reset-permissions' at the terminal Settings > Expo > Toggle permissions (note this doesn't actually *reset* permissions) Android Device Different based on Android version. Do a google search

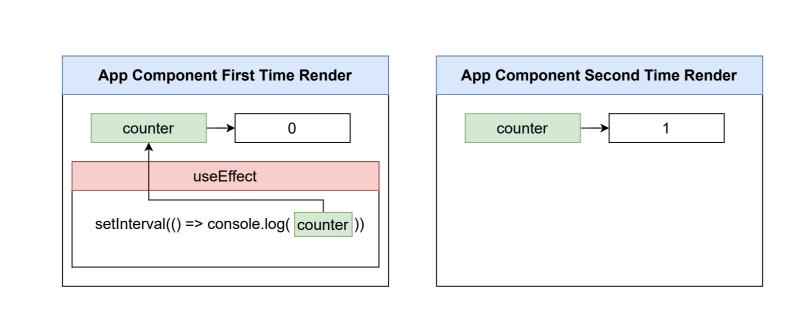


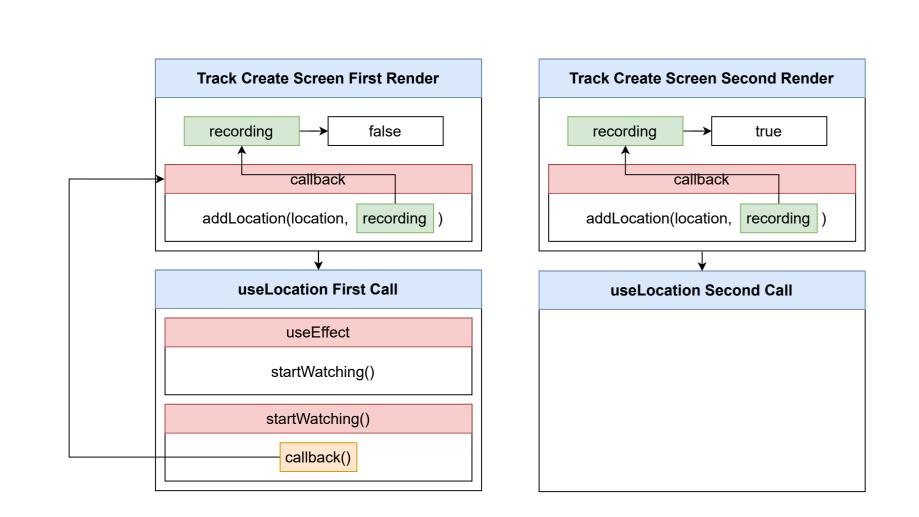


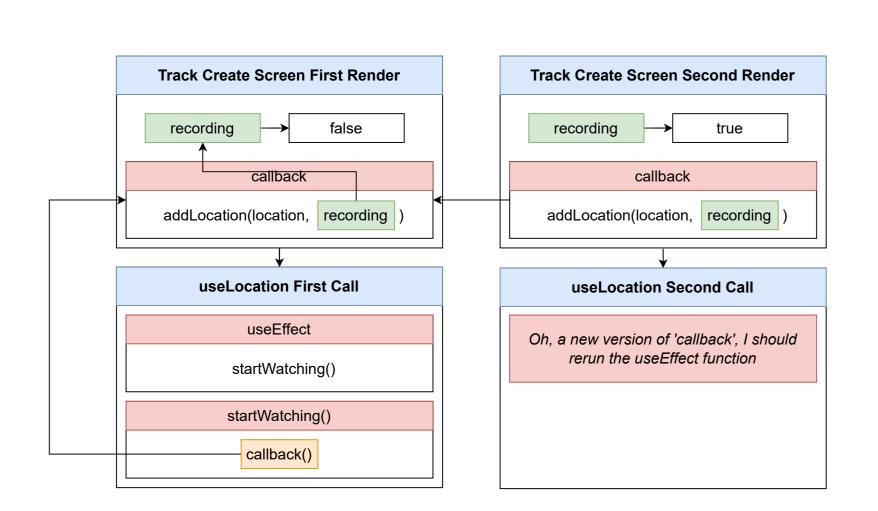


Codepen.io

Quick example with React on the web to understand 'useEffect'







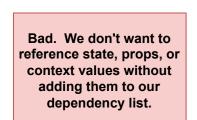
useCallback	() => { // callback v1 }	['hi there']	() => { // callback v1 }
		<u>, </u>	_
useCallback	() => {	['hi there']	() => { // callback v1 }
		,	
useCallback	() => { // callback v3 }	['bye there']	() => { // callback v3 }

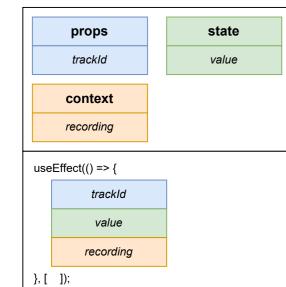
 useLocation First Call
 useLocation Second Call
 useLocation Third Call

 useEffect
 no change to callback, useEffect not called
 startWatching()

startWatching()

useLocation First Call	useLocation Second Call	useLocation Third Call
useEffect startWatching() return cleanupFunction	no change to callback, useEffect not called	wseEffect Ah, I am running a 2nd time. I will call the 'cleanup' function from the first time! cleanupFunction() startWatching()





Good!

Side note: 'setter' functions for state values are OK to reference

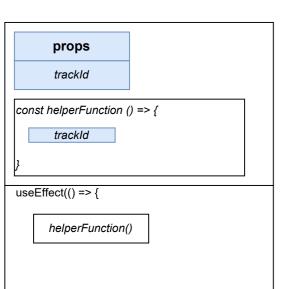
props
trackId
value

context
recording

useEffect(() => {
trackId
value
recording

}, [trackId value recording]);

Kind of Bad! Don't define helper functions that reference props, state, or context **outside** of **useEffect**



Good! Define helper functions that reference state/context/props inside useEffect. If you see state/context/props referenced in there, add it to the dependency array

props
trackId

useEffect(() => {

const helperFunction () => {

trackId
}

helperFunction()

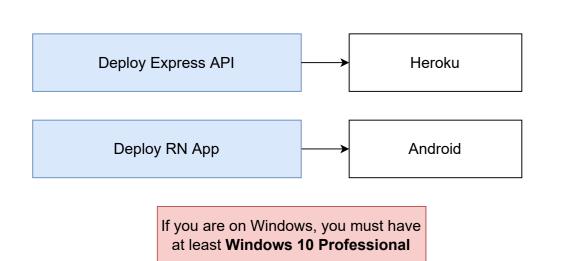
trackId

All about tracking the users location + storing points

Track

All about tracking the retrieving tracks fi

All about saving + retrieving existing tracks from our API



docs.expo.io/versions/latest/distribution/building-standalone-apps/

Deploying the app

play.google.com/apps/publish/signup

Create a new Play Store Dev

Account

devcenter.heroku.com/articles/heroku-cli

Heroku CLI for deploying
our Express API