Group 21 – Cycling Route Marker Application(s) (Mobile & Desktop)

2023 Nov 18

Build Activity Document - Version 1.0

Introduction

Hi there,

Most of our functional code is written in JavaScript. However, the Framework we use to build versions that port to both iOS and Android requires some configuring and further steps before we can deploy actual apps.

This document covers the basic requirements of setting up preview and development builds and the activities necessary to ensure those builds work. Production builds are not ready and will not be included in this document version.

Development & Preview

Expo is the React-based framework used to code the JS portions of our mobile application. It provides a mobile app for quick and easy testing of the application on iOS and Android devices or emulators. However, Expo's mobile app receives a JS package for deployment from the framework's dev server, and requires a continued connection to operate (console logs, debugging, etc.). In order to test the app without a direct connection to the dev server (on a developer's machine), we need to build a version that is installed as an app for iOS or Android.

However, an installed app version does not have the same console and debugging capabilities as the Expo testing version, and has some increased dependencies when building. Therefore, installed versions are split in three types: preview, development, and production. Production builds are a future topic.

Development builds use a dev-client dependency to package the Expo dev client connection just like Expo Go's testing, but install the entire app directly on the device instead of going through Expo Go. This means that the build is more stable and won't respond to code changes, so development can continue without messing up the working version. In addition, a working development build can be used to determine which project commits can be turned into working preview builds.

Preview builds are similar to production in that they are solely for testing the working features on the device without a dev server. Debugging is harder due to less feedback and no console, but the focus is on testing working features instead of fixing issues.

Procedures

To build installed versions of an app, we need native code. Expo's prebuild function is the easy way to create native code for iOS and Android apps. Prebuilding is usedful because our the project currently does not contain native code, so it does not yet need custom update procedures for the native versions.

This is our current build procedure necessary to building a native mobile app version:

1. Prebuild native code for the target version (or all versions).

    a. If the /ios and /android folders are already in the same directory as package.json, you may want to delete them.
    b. Run `expo prebuild` in the same directory as package.json, targeting the version using `--platform` if desired.
    c. NOTE: the online EAS build can do this automatically if your commit and gitignore exclude the /ios and /android folders in this directory. We have elected to include it with the possibility of future modifications to native code in mind.

2. Commit the project contents. This means that the build process can see which commit was used to build this version.
3. Run `eas build`
    a. Use `--platform` to select `android` `ios` or `all`
    b. Use `--profile` to select `preview` or `development`
    c. NOTE: iOS builds will require a developer account to build. Because Apple.
4. Download the build file (QR codes that last 13 days will be provided in the console if left open, and expo.dev will save a link in "Builds") and add it to your target device/emulator.
5. Open the developer server and run the application.
    a. This step is only for the `development` builds for testing whether the project compiles to native code correctly.

A few final notes are important here. First, due to Apple's app installation and signing requirements, in-development apps require Developer Version to be run (app signatures are not yet set up). This requires a device restart to turn on in the first place, but does not appear in Settings unless the app is already downloaded. Second, due to the installation requirements, the build configuration provided to EAS Build requires a target device UDID to be included in the Apple build. This means that even for the sponsor, testing an iOS preview version requires them to provide device data.