You will need to build React Native from source if you want to work on a new feature/bug fix, try out the latest features which are not released yet, or maintain your own fork with patches that cannot be merged to the core.

Android

Prerequisites

Assuming you have the Android SDK installed, run `android` to open the Android SDK Manager.

Make sure you have the following installed:

1. Android SDK version 30 (compileSdkVersion in `build.gradle`)
2. Android NDK 21.4.7075529 (From in `gradle.properties`, download links and installation instructions below)

Point Gradle to your Android SDK:

Step 1: Set environment variables through your local shell.

Note: Files may vary based on shell flavor. See below for examples from common shells.

- bash: `.bash_profile` or `.bashrc`
- zsh: `.zprofile` or `.zshrc`
- ksh: `.profile` or `$ENV`

Example:

```
export ANDROID_SDK=/Users/your_unix_name/android-sdk-macosx
export ANDROID_NDK=/Users/your_unix_name/android-ndk/android-ndk-r20b
```

Step 2: Create a `local.properties` file in the `android` directory of your react-native app with the following contents:

Example:

```
sdk.dir=/Users/your_unix_name/android-sdk-macosx
ndk.dir=/Users/your_unix_name/android-ndk/android-ndk-r20b
```

Download links for Android NDK

1. Mac OS (64-bit) - https://dl.google.com/android/repository/android-ndk-r20b-darwin-x86_64.zip
2. Linux (64-bit) - https://dl.google.com/android/repository/android-ndk-r20b-linux-x86_64.zip
3. Windows (64-bit) - https://dl.google.com/android/repository/android-ndk-r20b-windows-x86_64.zip
4. Windows (32-bit) - https://dl.google.com/android/repository/android-ndk-r20b-windows-x86.zip

You can find further instructions on the official page. Or follow instructions to download appropriate NDK from SDK manager.

Building the source

1. Installing from the fork

First, you need to clone `react-native` from the repo you want to use locally inside the `node_modules` folder. For example, if you wish to build against `main` of this repo:

```
mkdir node_modules
git clone git@github.com:facebook/react-native.git node_modules/react-native
```

2. Adding gradle dependencies

Update the `android/build.gradle` to make sure you're using AGP 7.x and add the highlighted lines.

```
buildscript {
    // ...
    dependencies {
        // Make sure that AGP is at least at version 7.x
        classpath("com.android.tools.build:gradle:7.0.4")

+       classpath("com.facebook.react:react-native-gradle-plugin")
+       classpath("de.undercouch:gradle-download-task:5.0.1")
    }
}
```

3. Adding the `:ReactAndroid` project

Add the `:ReactAndroid` project in `android/settings.gradle`:

```
pluginManagement {
    repositories {
        gradlePluginPortal()
        mavenLocal()
        google()
    }
}

// ...

include ':ReactAndroid'
project(':ReactAndroid').projectDir = new File(
    rootProject.projectDir, '../node_modules/react-native/ReactAndroid')

// Includes React Native Gradle Plugin into Gradle project. Required by
`:ReactAndroid` project configuration.
includeBuild('../node_modules/react-native/packages/react-native-gradle-plugin')

...
```

Modify your `android/app/build.gradle` to use the `:ReactAndroid` project instead of the pre-compiled library, e.g. - replace `implementation 'com.facebook.react:react-native:+'` with `implementation project(':ReactAndroid')`:

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
```

```
    implementation project(':ReactAndroid')

    // ...
}
```

4. Making 3rd-party modules use your fork

If you use 3rd-party React Native modules, you need to override their dependencies so that they don't bundle the pre-compiled library. Otherwise you'll get an error while compiling - `Error: more than one library with package name 'com.facebook.react'` .

Modify your `android/build.gradle` , and add:

```
allprojects {
    repositories { /* ... */ }

    configurations.all {
        resolutionStrategy {
            dependencySubstitution {
                substitute module("com.facebook.react:react-native:+") with
project(":ReactAndroid")
            }
        }
    }
}
```

Building from Android Studio

From the Welcome screen of Android Studio choose "Import project" and select the `android` folder of your app.

You should be able to use the *Run* button to run your app on a device. Android Studio won't start the packager automatically, you'll need to start it by running `npm start` on the command line.

Additional notes

Building from source can take a long time, especially for the first build, as it needs to download ~~200 MB of artifacts and compile the native code. Every time you update the `react-native` version from your repo, the build directory may get deleted, and all the files are re-downloaded. To avoid this, you might want to change your build directory path by editing the `~~/.gradle/init.gradle` file:

```
gradle.projectsLoaded {
    rootProject.allprojects {
        buildDir = "/path/to/build/directory/${rootProject.name}/${project.name}"
    }
}
```

Troubleshooting

Gradle build fails in `ndk-build` . See the section about `local.properties` file above.

Publish your own version of React Native

There is a docker image that helps you build the required Android sources without installing any additional tooling (other than Docker, which can be committed to a git branch as a fully functional React Native fork release.

Run this from a fork of the React Native repo.

```
git checkout -d release/my-react-native-release
docker run --rm --name rn-build -v $PWD:/pwd -w /pwd reactnativecommunity/react-native-android /bin/sh -c "./gradlew installArchives"
git add android --force
git commit -a -m 'my react native forked release'
git push
```

Install it in your app project package.json.

```
"dependencies": {
    ...
    "react-native": "myName/react-native#release/my-react-native-release"
}
```

Rationale

The recommended approach to working with React Native is to always update to the latest version. No support is provided on older versions and if you run into issues the contributors will always ask you to upgrade to the latest version before even looking at your particular issue. Sometimes, though, you are temporarily stuck on an older React Native version, but you require some changes from newer versions urgently (bugfixes) without having to do a full upgrade right now. This situation should be short lived by definition and once you have the time, the real solution is to upgrade to the latest version.

With this goal of a shortlived fork of React Native in mind, you can publish your own version of React Native. The facebook/react-native repository contains all the dependencies required to be used directly as a git dependency, except for the Android React Native library binary (.aar).

Building

This binary needs to become available in your project's `node_modules/react-native/android` folder or directly in your gradle dependency of your Android app. You can achieve this in one of two ways: Git dependency branch, Android binary dependency through Maven.

To build the .aar React Native library, you can follow the steps to build from source first to install all required tooling. Then to build the actual library, you can run the following in the root of your react-native checkout:

```
./gradlew :ReactAndroid:installArchives --no-daemon
```

If you don't want to install the required toolchain for building from source, you can use a prebuilt docker image to create a react native binary;

```
docker run --rm --name rn-build -v $PWD:/pwd -w /pwd reactnativecommunity/react-native-android /bin/sh -c "./gradlew installArchives"
```

If you haven't used the Android NDK before or if you have a NDK version not exactly matching the required version for building React Native, this is the recommended approach.

The resulting binary can be made available to app projects in one of the two ways described below.

Publishing to Maven/Nexus

Upload the binaries from the `android` folder to maven and point your Android app project gradle dependency for React Native to your Maven/Nexus dependency.

Publishing to a git fork dependency

Instead of uploading to Maven/Nexus, you can add the binaries built in the previous steps to git, by changing the .gitignore and committing the binaries to your forked branch. This allows you to make your fork into a functioning git dependency for React Native app projects.

If you have changes that you want to actually merge to React Native, make them on another branch first and open a PR. To start making your dependency branch, make sure you are on a 'release/my-forked-release' branch, then merge any commits that you need from yourself or others into this branch. This release branch should never be merged into any other branch.

```
# create .aar, then:
git add android --force
git commit -m 'my release commit'
git push
```

Now you can use this branch as a git dependency in your app project, by pointing your package.json dependency to this branch:

```
"dependencies": {
    ...
    "react-native": "my-name/react-native#release/my-forked-release,
    ...
}
```

No other modifications to your dependencies should be necessary for your native changes to be included in your project.