Environment setup For Mac and window

## Reinstall iMac

1. Press power button once and release.
2. Hold “command + R” until the option menu comes .
3. Goto Disk utility > select internal storage > press erase > close the window
4. Select the reinstall menu > select internal storage > continue

## 

## Environment setup for React-Native in Mac

1. Install JDK [link](https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html)
   1. Open Terminal
   2. Confirm you have JDK by typing “which java”. It should show something like /usr/bin/java.
   3. Check you have the needed version of Java, by typing “java -version”.
2. Install Python 2 [link](https://www.python.org/downloads/) (Only for window)
3. Install Xcode
4. Install android studio [link](https://developer.android.com/studio/)
5. Install Homebrew ( /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)" ) (For mac)
6. Install Node ( For install in window go to this [link](https://nodejs.org/en/download/))

$ brew update

$ brew install node

$ node -v

1. Install watchman (For mac)

$ brew update

$ brew install watchman

1. Install eslint

npm install -g eslint

1. Android environment variable setup for WINDOWS

##### JAVA\_HOME = C:\Program FIles\Java\jdk

Path =C:\Users\AppData\Local\Android\sdk\platform-tools

ANDROID\_HOME = c:\Users\YOUR\_USERNAME\AppData\Local\Android\Sdk

1. Android environment variable setup for MAC
   1. Open your Bash Profile with command below:

nano .bash\_profile

* 1. Edit .bash\_profile with below paths:

export ANDROID\_HOME=/Users/{Account}/Library/Android/sdk  
 export PATH=$ANDROID\_HOME/platform-tools:$PATH  
 export PATH=$ANDROID\_HOME/tools:$PATH

\*\*PC: please check your Android sdk directory path is installed correctly.

* 1. Save your Bash Profile with this buttons:

**Ctl+o > Enter > ctl+x**

1. Install react-native-cli globally

#### npm install -g react-native-cli

1. Create a new project

#### react-native init project\_name

1. Run your android project

#### react-native run-android

1. Run your ios project

#### react-native run-ios

Environment setup for React-Native in Ubuntu 18.04

1. sudo apt install openjdk-8-jdk
2. java -version
3. Set JAVA\_HOME
   1. Check Java version: which java
   2. Edit .bashrc by: gedit .bashrc
   3. End of the bashrc file add
      1. JAVA\_HOME=/usr/lib/jvm/default-java/bin
      2. export JAVA\_HOME
      3. PATH=$PATH:$JAVA\_HOME
      4. export PATH
      5. Save and close the terminal
   4. echo $JAVA\_HOME to check the home
   5. Details on [LInk](https://www.wikihow.com/Set-Up-Your-Java_Home-Path-in-Ubuntu)
4. Download and install Android Studio
   1. cd android-studio/bin
   2. ./studio.sh
5. Set the ANDROID\_HOME environment variable
   1. sudo gedit ~/.bashrc
   2. export ANDROID\_HOME=/home/user\_directory/Android/Sdk
   3. export PATH=${PATH}:$ANDROID\_HOME/tools:$ANDROID\_HOME/platform-tools
   4. export JAVA\_HOME=/usr/lib/jvm/java-8-oracle
   5. echo $ANDROID\_HOME to check
   6. Details [Link](https://medium.com/@aashimad1/install-android-studio-in-ubuntu-b8aed675849f)

Setting up React Native in Ubuntu 18:

1. Installing node [Link](https://tecadmin.net/install-latest-nodejs-npm-on-ubuntu/)
2. sudo npm install -g react-native-cli
3. react-native init ProjectName
4. Cd Project
5. react-native run-android

## Command for MAC

1. Homebrew Install: /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
2. brew update
3. brew install node
4. node -v // check node version
5. brew install watchman
6. npm install -g react-native-cli
7. react-native init ‘project name’ // Create a new project
8. react-native run-android // run android project
9. react-native run-ios // run ios project
   1. If Xcode not found then <Select “preference > Location > Command Line Tools > Xcode XX.X”> or [Link](https://stackoverflow.com/questions/39778607/error-running-react-native-app-from-terminal-ios)
10. npm install -g eslint // install eslint
11. npm install --save-dev eslint-config-rallycodding
12. adb devices // check running android device
13. react-native run-ios --simulator="iPhone 4s" // run a specific ios simulator
14. adb shell input keyevent 82
15. adb reverse tcp:8081 tcp:8081
16. React-native start
17. react-native run-android --variant=release // run a release build for android
18. Update react-native

npm install -g npm-check-updates

ncu -u react-native

npm install

1. In windows sdk.dir = C:\\Users\\USERNAME\\AppData\\Local\\Android\\sdk

in macOS sdk.dir = /Users/USERNAME/Library/Android/sdk

in linux sdk.dir = /home/USERNAME/Android/Sdk

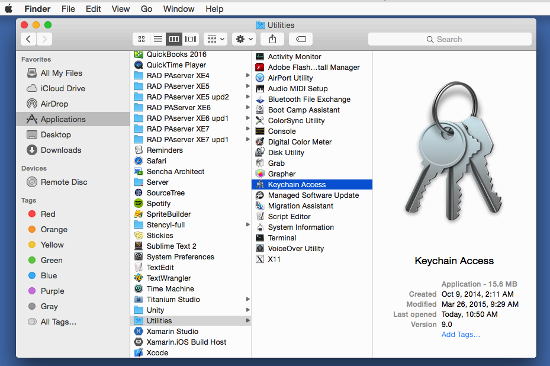
adding in android\local.properties for windows

14. ADB Android Device Unauthorized. When adb not found

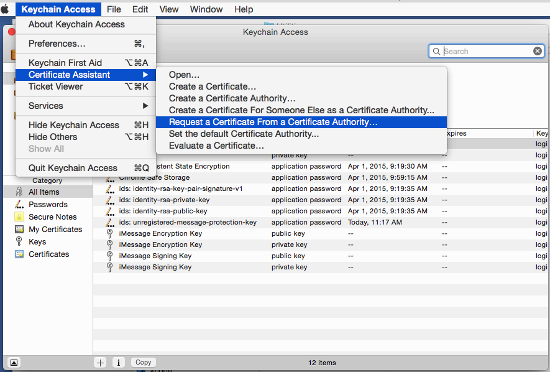
* unplug device
* adb kill-server
* adb start-server
* plug device

## Create Certificate Signing Request

Start by creating a .certSigningRequest (CSR) file on your Mac, using Keychain Access. Open Finder, and then open Keychain Access from the Utilities folder.

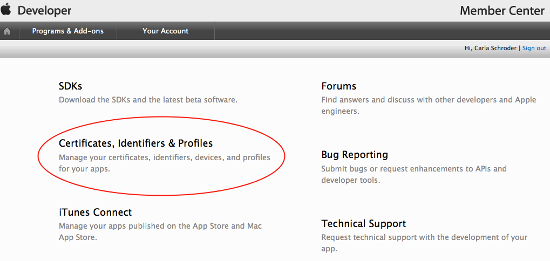


Next, open Keychain Access > Certificate Assistant > Request a Certificate From a Certificate Authority.



Enter the email address that you use in your Apple developer account, and enter a common name. The common name can be anything you want, for example a helpful descriptive name like "ios-mybiz". Check Saved to disk and Let me specify key pair information, then click Continue.

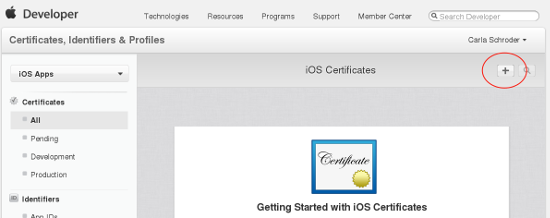
Now login to the Member Center on <https://developer.apple.com/>. Click Certificates, Identifiers, & Profiles.



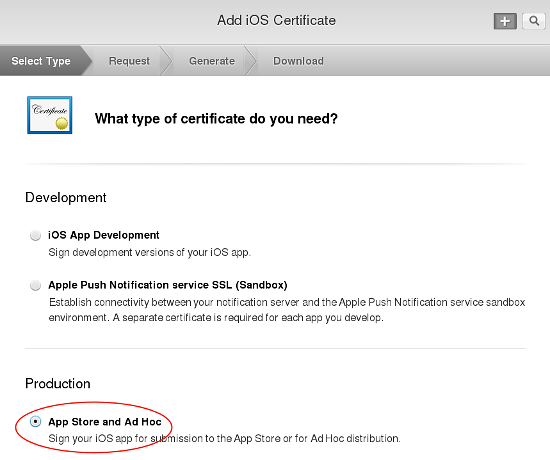
Then click iOS Apps > Certificates.



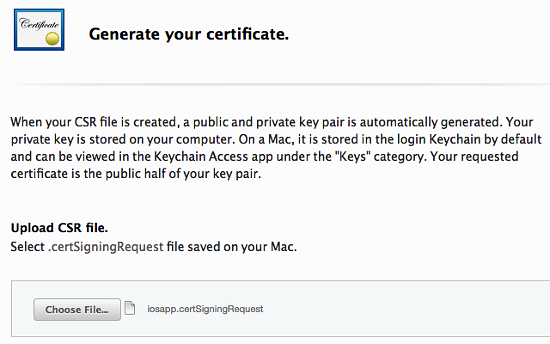
Next, click the add button (the little plus sign) in the top right corner of the iOS Certificate page.



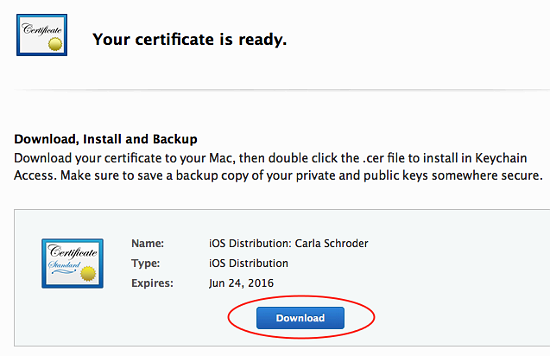
Under "What type of certificate do you need?" check App Store and Ad Hoc, then click the Continue button at the bottom of the page.



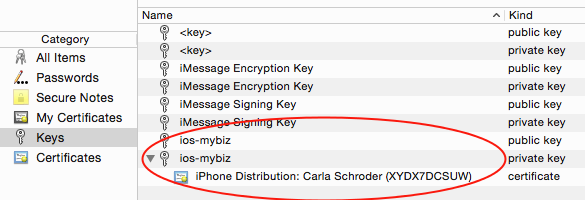
The next screen, About Creating a Certificate Signing Request (CSR) has information about creating a CSR in Keychain Access. You already did this, so go to the next screen. "Add iOS Certificate", to upload the CSR you already created, then click the Generate button.



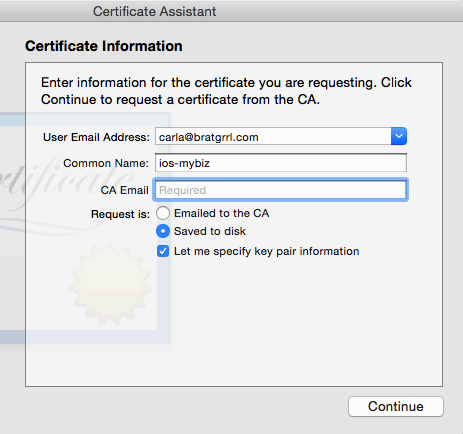
Your new certificate is named ios\_distribution.cer. Download it to your Mac; then find it and double-click on it to install it properly in Keychain.



After installing it, you should see it stored with its corresponding private key in Keychain.



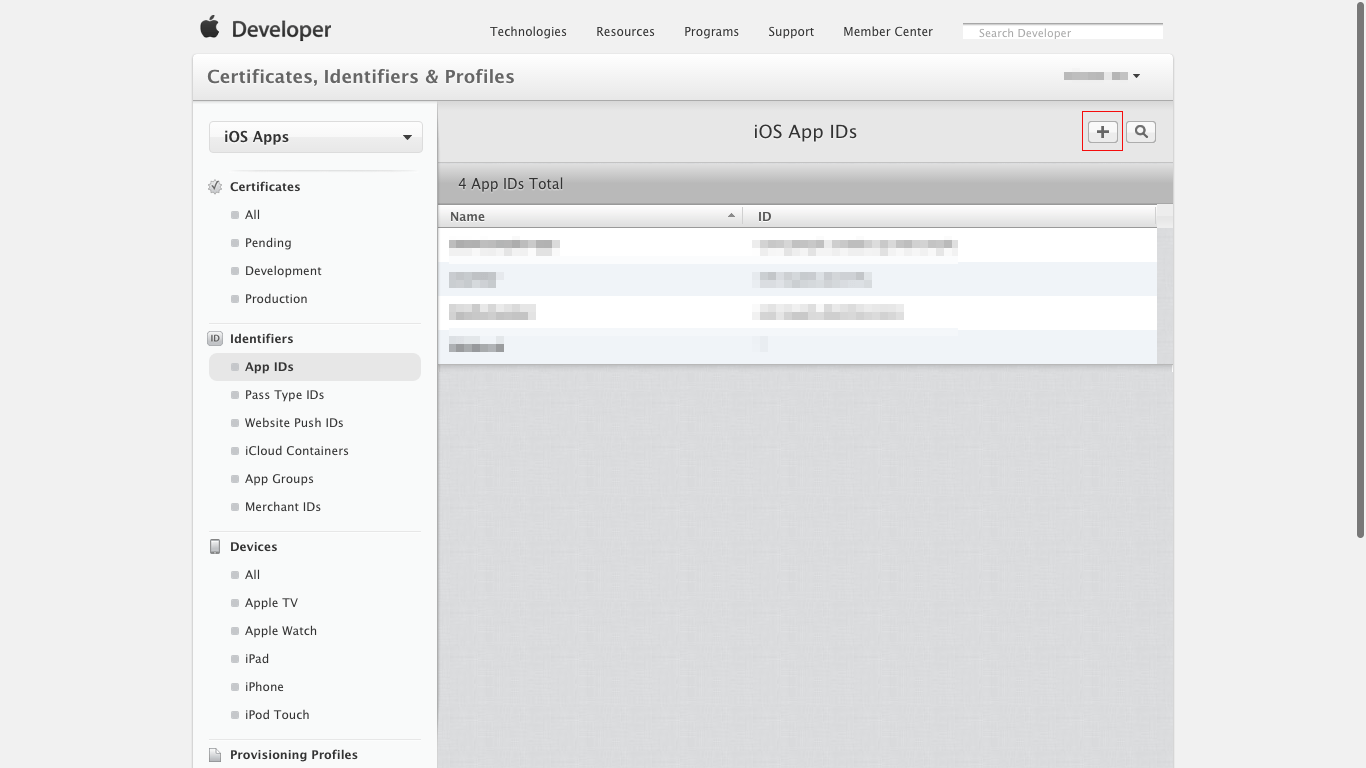
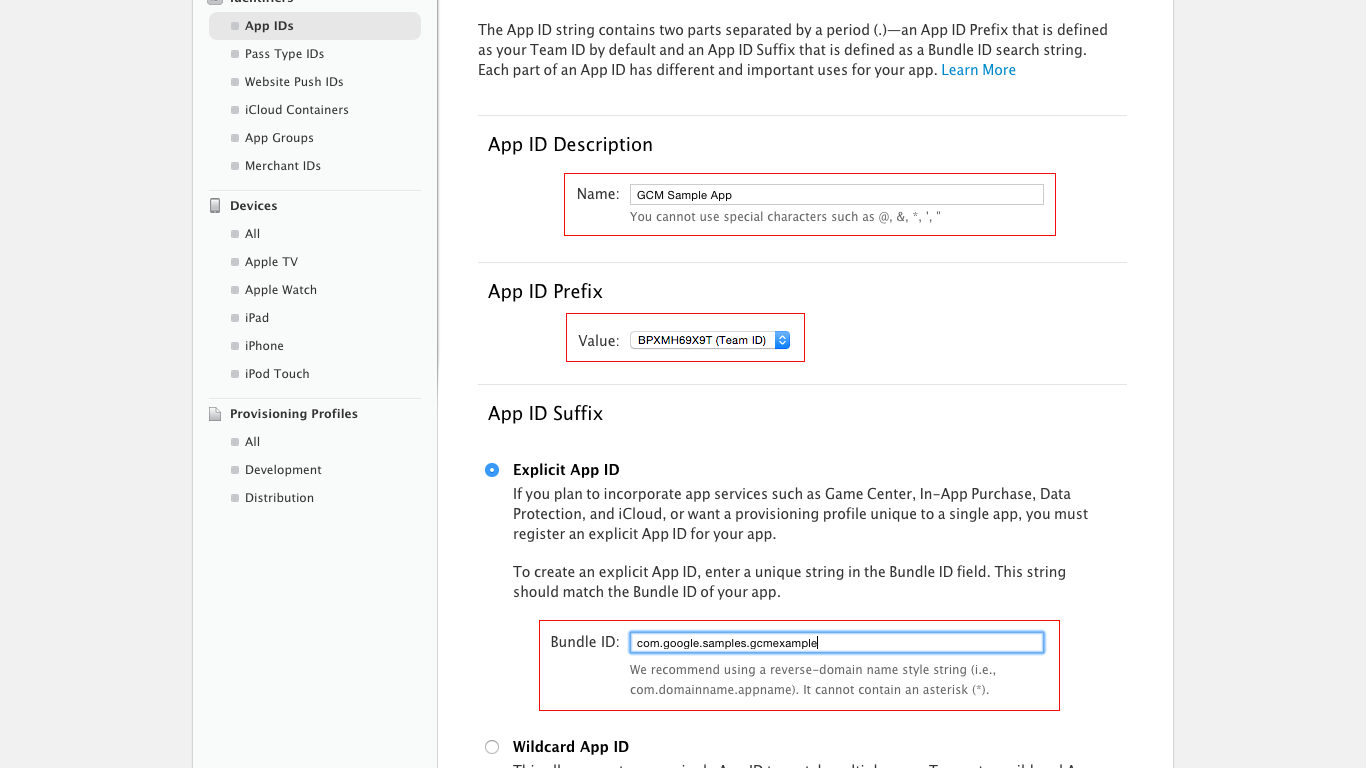
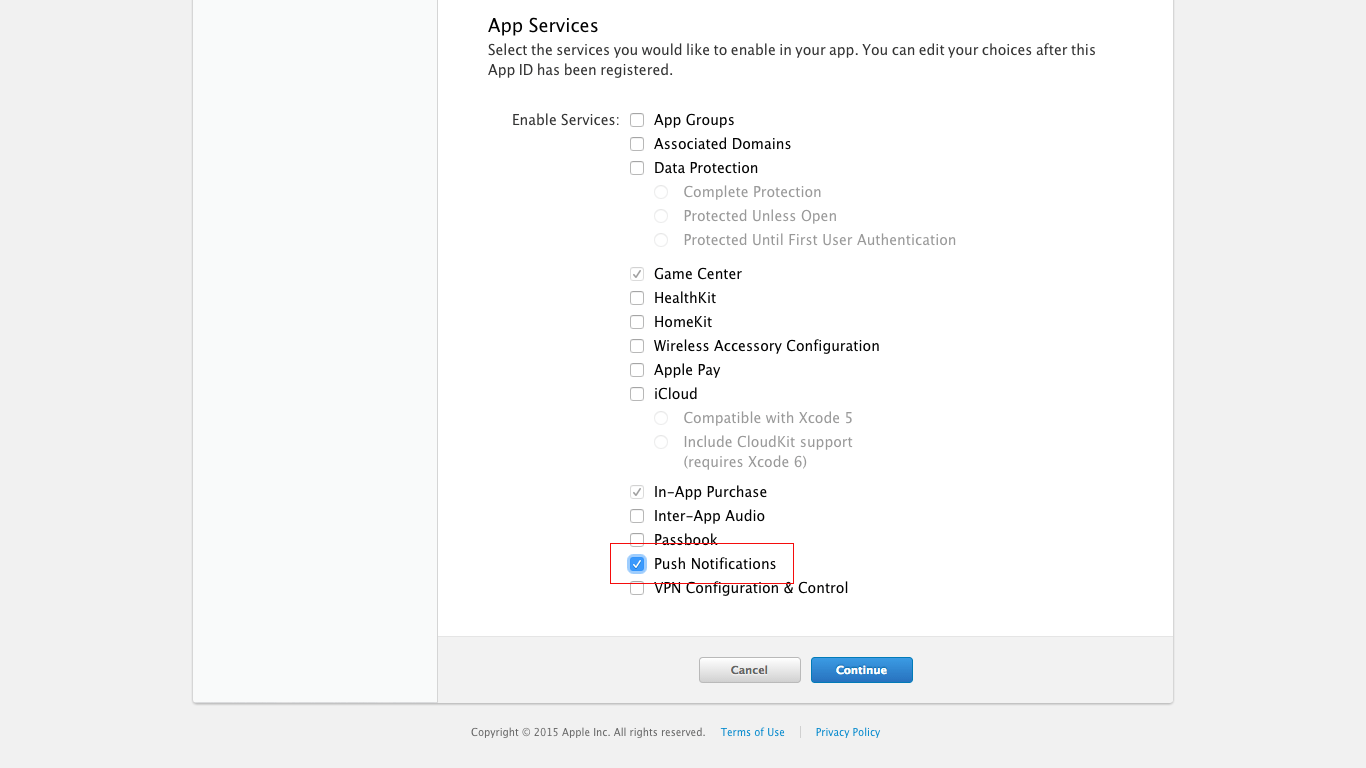
Remember to make backups of your keys and certificates and keep them in a safe place.



Give your CSR a helpful descriptive name, such as iosapp.certSigningRequest, and choose a location to save it on your hard drive, then click Save.

## 

## Create an app ID for ios

1. Navigate to the [Apple Developer Member Center](https://developer.apple.com/membercenter/index.action) and sign in.
2. Navigate to Certificates, Identifiers and Profiles.
3. In the drop down menu on the top left corner, select iOS, tvOS, watchOS if it's not already selected, then navigate to Identifiers > App IDs.
4. Click the + button to create a new App ID. 
5. To create the new App ID:
   1. Input a Name for your App ID (e.g. Firebase Sample App)
   2. Input a Team ID. This value must match the Team ID in the Membership tab.
   3. In the App ID Suffix section, select Explicit App ID, then input your Bundle ID (e.g. com.google.samples.example). The value of the Bundle ID should match the value that you are using in your app's Info.plist and the value that you are using to get a configuration for FCM. 
   4. If any service need to active example(Push Notifications) In the App Services section, make sure that Push Notifications is checked. 
6. Click Continue and check that your input is correct:
   1. The value of Identifier should match the concatenation of the values of the Team ID and of the Bundle ID
   2. Push Notifications should be Configurable
7. Click Register to create the App ID.

## Create a provisioning profile for ios

1. Navigate to the [Apple Developer Member Center](https://developer.apple.com/membercenter/index.action) and sign in.
2. Navigate to Certificates, Identifiers and Profiles.
3. In the drop down menu on the top left corner, select iOS, tvOS, watchOS if it's not already selected, then navigate to Provisioning Profiles > All.
4. Click the + button to create a new Provisioning Profile.
5. Select iOS App Development as provisioning profile type, then click Continue.
6. In the drop down menu, select the App ID you want to use, then click Continue.
7. Select the iOS Development certificate of the App ID you have chosen in the previous step, then click Continue.
8. Select the iOS devices that you want to include in the Provisioning Profile, then click Continue. Make sure to select all the devices you want to use for your testing.
9. Input a name for this provisioning profile (e.g. Firebase Sample App Development Profile), then click Generate.
10. Click Download to save the Provisioning Profile to your Mac.
11. Double-click the Provisioning Profile file to install it.

## 

## 

## 

## 

## 

## Generating Signed APK (Android) from “Console”

[Link](https://facebook.github.io/react-native/docs/signed-apk-android) for details.

### Generate keytool:

* On Windows:

You can generate a private signing key using keytool. On Windows keytool must be run from C:\Program Files\Java\jdkx.x.x\_x\bin.

$ keytool -genkeypair -v -keystore my-release-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000

* On Mac:

if you're not sure where your jdk bin folder is, then perform the following command to find it:

$ /usr/libexec/java\_home

It will output the directory of the jdk, which will look something like this:

$ /Library/Java/JavaVirtualMachines/jdkX.X.X\_XXX.jdk/Contents/Home

Navigate to the directory by using the command $ cd /your/jdk/path and use the keytool command with sudo permission as shown below.

$ sudo keytool -genkey -v -keystore my-release-key.keystore -alias my-key-alias -keyalg RSA -keysize 2048 -validity 10000

Setting up gradle variables:

1. Place the my-release-key.keystore file under the android/app directory in your project folder.
2. Edit the file ~/.gradle/gradle.properties or android/gradle.properties, and add the following (replace \*\*\*\*\* with the correct keystore password, alias and key password).

|  |
| --- |
| MYAPP\_RELEASE\_STORE\_FILE=my-release-key.keystore  MYAPP\_RELEASE\_KEY\_ALIAS=my-key-alias  MYAPP\_RELEASE\_STORE\_PASSWORD=1234  MYAPP\_RELEASE\_KEY\_PASSWORD=1234 |

Edit the file android/app/build.gradle in your project folder, and add the signing config,

|  |
| --- |
| ...  android {  ...  defaultConfig { ... }  signingConfigs {  release {  if (project.hasProperty('MYAPP\_RELEASE\_STORE\_FILE')) {  storeFile file(MYAPP\_RELEASE\_STORE\_FILE)  storePassword MYAPP\_RELEASE\_STORE\_PASSWORD  keyAlias MYAPP\_RELEASE\_KEY\_ALIAS  keyPassword MYAPP\_RELEASE\_KEY\_PASSWORD  }  }  }  buildTypes {  release {  ...  signingConfig signingConfigs.release  }  }  }  ... |

Generating the release APK:

Simply run the following in a terminal

$ cd android

$ ./gradlew assembleRelease

Testing the release build of your app:

#### $ react-native run-android --variant=release

For optimization follow this [Link](https://facebook.github.io/react-native/docs/signed-apk-android#adding-signing-config-to-your-app-s-gradle-config).

### Find the apk:

#### YourApplication\app\build\outputs\apk

## 

## IOS build configuration

1. Download xcode.
2. Go to your project folder > ios
3. Double click .xcodeproj file . It will open by xcode
4. Click on Project folder name (ex: Test) in Xcode
5. Click on “General”
6. Sign in with your team account
   1. Select project name from “TARGETS”
   2. Click “Add Account” from “Signing” section
   3. Providing signing credential and sign in
   4. Select Team from signing section
7. Select “Project\_Name+Tests” from “TARGET”, usually the second option from the “TARGET” section.
   1. Select Team from signing section
8. Select “File > Project Settings > Shared Project Settings > Build System > Legacy Build System”
9. Select “Product > Schema > Edit Schema > Run > Build Configuration > Release”
10. Select “preference > Location > Command Line Tools > Xcode XX.X” or [Link](https://stackoverflow.com/questions/39778607/error-running-react-native-app-from-terminal-ios)
11. In command react-native run-ios you should have
    1. Create Certificate Signing Request
    2. Create an app ID for ios
    3. Create a provisioning profile for ios (download & install it)

PC: check upper section for a,b & c

## Generating Release Build for Testflight (IOS)

1. Xcode > Product > Archive
2. After appearing new screen
   1. Validate App > next > next > Validate = success > close
3. Distribute App > iOS App Store > Upload > Check all > Upload
4. For checking
   1. <https://appstoreconnect.apple.com/login>
   2. My Apps > Select App > Test Flight > Missing Compliance