May 13 · 6 min read

How To Build an App for the Oculus Go From Start To Finish (with Unity)

> This guide is for developing for the Oculus Go, on Windows with Unity 2018.1. It is current as of June 21st, 2018.

This week my company held it's first ever internal hackathon. As soon as I heard about it, I signed up and knew exactly what I wanted to build—a VR experience for my shiny new Oculus Go. Long story short, building for the Go wasn't as straightforward as I had hoped, but by the end of the 24 hours I had gotten it figured out and had my app built! So here's a thorough end to end guide of how to develop for the Go, which should cover and eliminate all the bugs I experienced so you don't need to worry about them :)

What You'll Need

1. Unity 3D (free)
2. Android Studio (free)
3. A computer with a decent graphics card (I have a 13" Surface Book 2 (~$2k))
4. And an Oculus Go, of course! ($199)

1) Download and Install Unity (& the Android Plugin)

- Install Unity 2018.1 Personal (it's free): https://store.unity.com/
- When installing make sure to add on Android Build support

2) Download and Install Android Studio

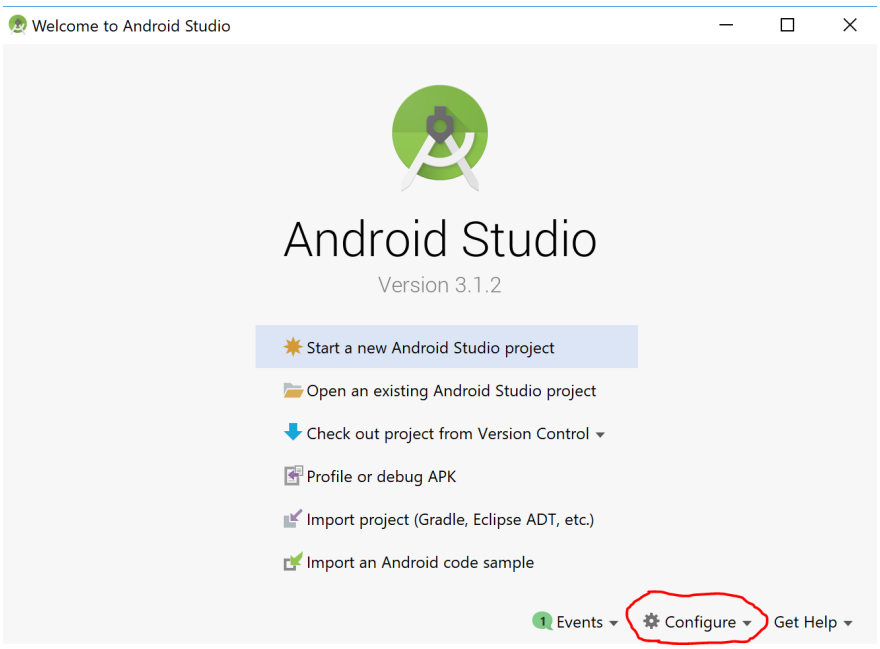
- https://developer.android.com/studio/

3) Grab Oculus Core Utilities for Unity

- https://developer.oculus.com/downloads/unity/

4) Android Development Software Setup

- Open Android Studio
- On the welcome screen, click **Configure** on the bottom right, and then **SDK Manager**



- Under the SDK Platforms tab, make sure API Levels 21 through 27 are checked.
- On the **SDK Tools** tab, check the box next to *Show Package Details* in the lower right corner of the window.

- Under **Android SDK Build-Tools 28-rc2**, make sure the highest numbered item is check ***that does not end with "-rc1" or "-rc2".*** As of this writing, that is **27.0.3.**

27.0.3 = GOOD. 28.0.0-rc1/2 = BAD.

- Also under **SDK Tools** make sure the follow are checked:

> *LLDB*

> *Android SDK Platform-Tools*

> *Android SDK Tools 25.2.5*

> *NDK*

- Click **Apply**. This will install everything you selected above on the SDK tabs.
- Back on the welcome screen, click **Configure** on the bottom right, and then **Project Defaults > Project Structure**
- Here you'll the file paths for the SDK, JDK and NDK. Copy these down in a text editor; we'll need these shortly

- Set your “environment variables”:
- On your Windows machine, search for “Environment Variables”. This should take you to Control Panel > System Properties. Go to the Advanced Tab, then to the **Environment Variables** button in the lower right.
- In the top section, set/modify/add the following variables:

> *Set the environment variable **JAVA_HOME** to the JDK location.*

> *Set the environment variable **ANDROID_HOME** to the Android SDK location.*

> *Set the environment variable **ANDROID_NDK_HOME** to the Android NDK location.*

> *Add the JDK tools directory to your **PATH**, ie C:\Program Files\Android Studio\jre\bin*

- One of the errors I kept running into when building for Android in Unity was “Unable to list target platforms”. It’s vague and frustrating. Here’s how to fix it (taken from this stackoverflow post):

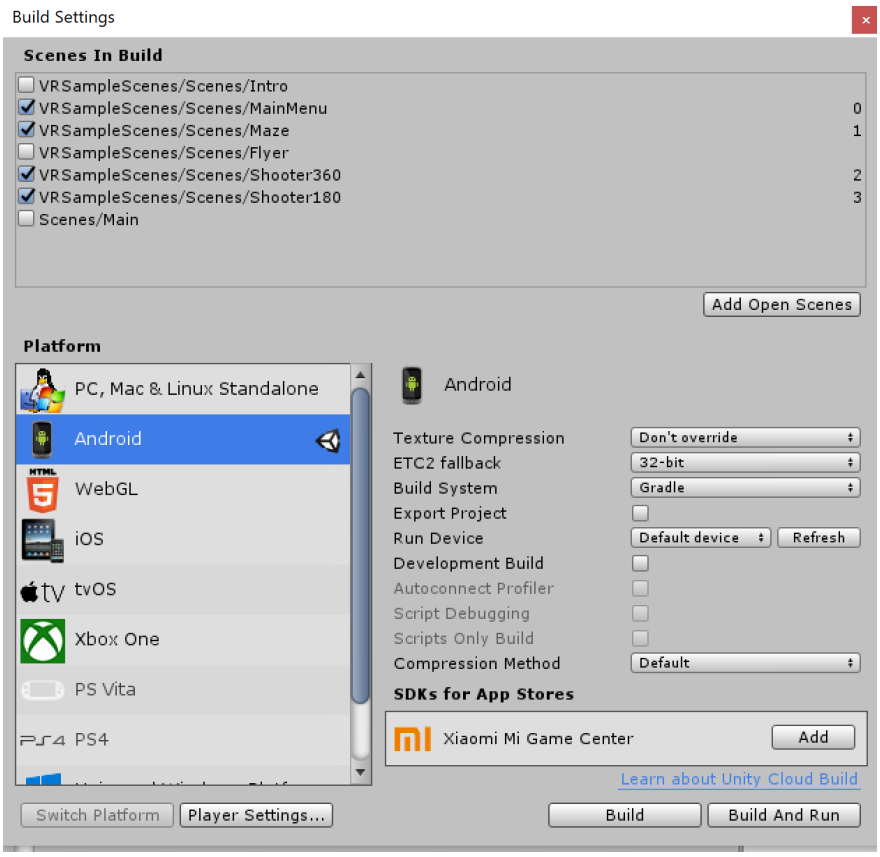
> *Delete android sdk "tools" folder : [Your Android SDK root]/tools -> tools*

> *Download SDK Tools: http://dl-ssl.google.com/android/repository/tools_r25.2.5-windows.zip*

> *Extract that to Android SDK root*

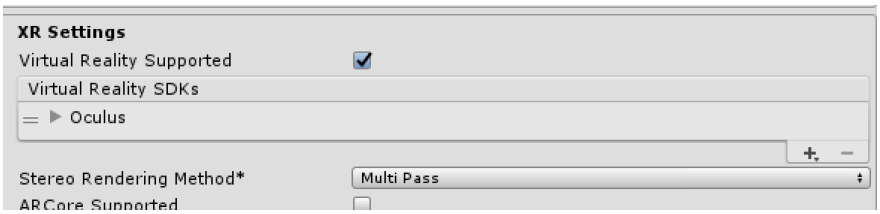
5) Setting up Unity to Build for Android

- Open Unity and create a new project
- Once your new (or existing) project opens, we need to set it to build for Android.
- Go to **File > Build Settings**
- Select Android and then **Switch Platform.** (If you did not add Android support when you first installed Unity, you will have to do so now, then restart Unity).



- Close the Build Settings window.

- Go to **Edit > Preferences**
- Click on the **External Tools** tab
- Scroll down to the **Android** section
- Set the SDK, JDK and NDK paths to what you copied to the text file in Step 4.
- Close the Preferences window.
- Go to **Edit > Project Settings > Player**
- Scroll down to **XR Settings.** Click the box next to **Virtual Reality Supported**.



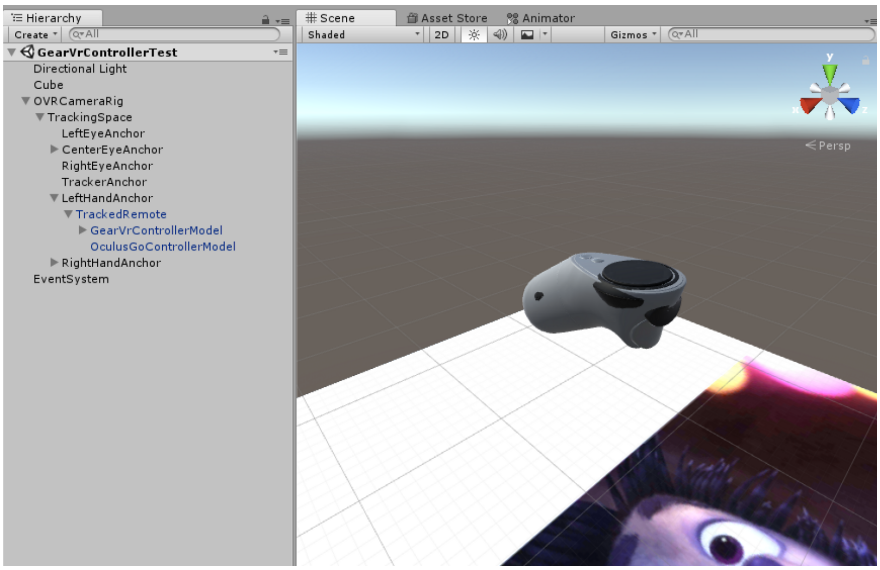
- Verify that Oculus appears in the SDK's list. If not, click the "**+**" to the right, and select **Oculus**.
- Scroll back up to the top and set your Company Name and Product Name.
- Then scroll down to **Other .** Set the Package Name field to **com.[CompanyName].[ProductName]**
- For Minimum API Level—I would set this **API Level 21**

6) Import Oculus Samples Scenes

To get started with your app, we'll import some premade sample scenes that Oculus was nice enough to make for us.

- Grab the OculusUtilities Unity package that you downloaded in step 3
- Drag and drop OculusUtilities.unity into the Asset folders of your project, in the Unity window. Import when prompted.
- Now, in your Assets folder, go to **Oculus > VR > Scenes**. Double click **GearVrControllerTest.**

- In the hierarchy of the scene, dive down within the OVRCameraRig object to find the **OculusGoControllerModel** within the scene. Double click it to focus on it in the scene view (see below).



7) Enable Developer Mode on the Go

(Taken from the Oculus Developer documentation)

To begin development locally for Oculus Go, you must enable Developer Mode in the companion app. Before you can put your device in Developer Mode, you need to have created (or belong to) a developer organization on the Oculus Dashboard.

To join an existing organization:

1. You'll need to request access to the existing Organization from the admin.
2. You'll receive an email invite. Once accepted, you'll be a member of the Organization.

To create a new organization:

1. Go to: https://dashboard.oculus.com/organization/create
2. Fill in the appropriate information.

- **Enable Developer Mode**

To put your Oculus Go in developer mode:

1. Open the Oculus app on your mobile device.
2. In the Settings menu, select your Oculus Go headset that you're using for development.
3. Select **More Settings**.
4. Toggle **Developer Mode** on.

8) Set up Android Debug Bridge

We're almost there (I promise)! Before we can actually move our Unity app from our computer to our Go, we need to install Android Debug Bridge (ADB) and the drivers for the Oculus Go.

- Download and install the Oculus Go ADB driver:

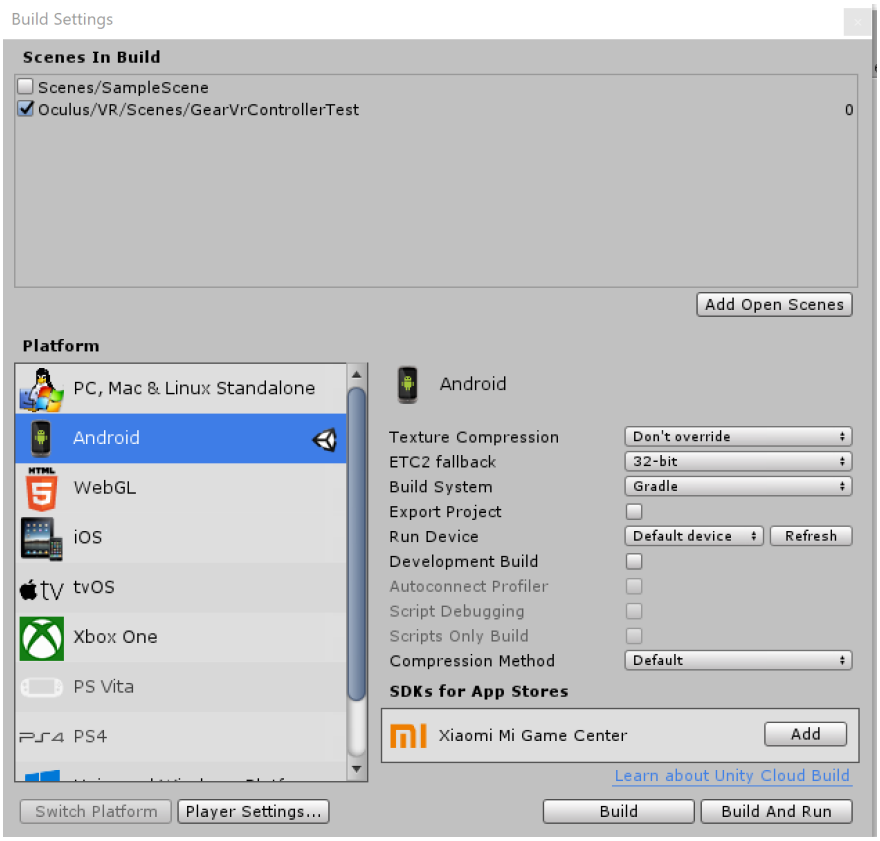
1. Download the zip file containing the driver.
2. Unzip the file.
3. Right-click on the .inf file and select **Install**.

- Download and install ADB: https://forum.xda-developers.com/showthread.php?t=2588979 (See steps starting at 5:40 mark in video below)

9) Connect your Go and Build!

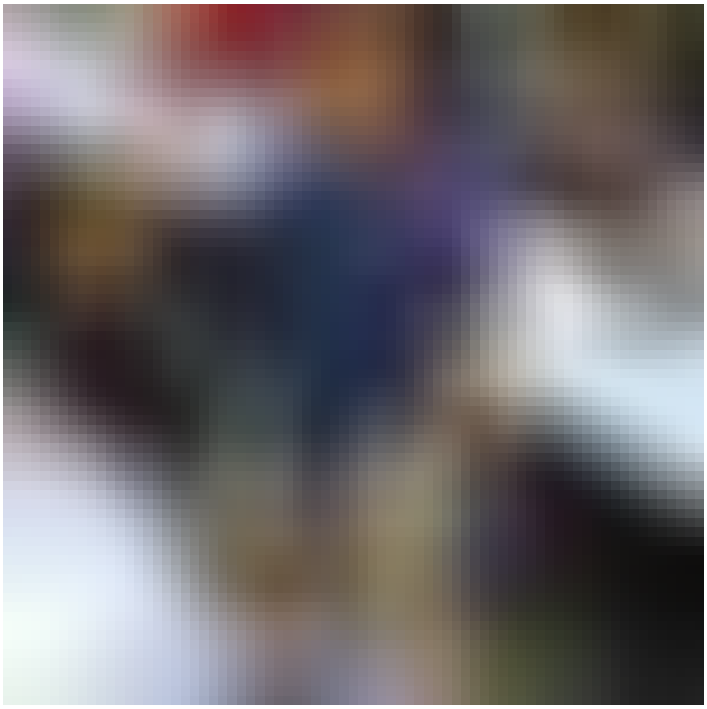
- Connect your go to your PC via micro-USB.

- In the ADB window from the last step, run *adb devices* to confirm that your Go has been detected.
- Back in Unity, go to **File > Build Settings**
- Click “Add Open Scenes”, and ensure only the scene *GearVrControllerTest* has the check mark next to it checked.
- Click **Build**



- Select the folder where you would like to place the .APK file that’s generated. I like to create a “Builds” folder in the project. **Save**.
- After the .APK is generated, navigate to it in File Explorer. Copy it to the same directory where you installed ADB in Step 8.
- Being the ADB window back up.
- Run *adb install [NameOfYourApp].apk*
- You should now see the file get transferred to your Oculus Go. In a minute or two, it should then show the message **Success**.
- Congrats—you did it!!

- Now, to run your app, put on your Go and go to **Library > Unknown Sources.** You should see the Bundle ID of your app on the last page of the Unknown Apps list. Click on it to launch it, and enjoy!
- To uninstall run *adb uninstall *package name**. For example, *adb uninstall com.headjack.myapp.*



Let me know if this worked for you (it does for me!), and if you have any feedback.