GoogleVR & GearVR Camera Setup V1.4

Table of Contents

Introduction
1. What is GoogleVR & GearVR Camera Setup?
2. How to use
    Prerequisites
    Starting With A New Project
    Starting with Existing Google Cardboard Project
    Building A Multi-Scene Setup
    Platform Dependent Compilation
3. Demo
4. How It Works
5. Further Info

Introduction

Thanks for the purchase and support! We are a community of VR devs, working together to create games, experiences, development tools, and tutorials in an effort to empower emerging VR developers worldwide. Join us here:
https://www.youtube.com/nurfacegames/

For any questions or support, please email:
nurfacegames@gmail.com

1. What is GoogleVR & GearVR Camera Setup?

Developing for Mobile VR is great but porting between Google VR (previously Cardboard) and Gear VR can be frustrating! This asset allows you to easily switch the build target between Google VR and Gear VR, with a single click. Import the package and add the “VRMain” prefab, now simply click the “VR Build Settings” option on the toolbar and select your build target!

2. How to use

Prerequisites

GoogleVR 0.8+

- This asset relies on the Google VR SDK 0.8+ to work. Because we are developing for both Google VR and Gear VR, the Google VR SDK will be a part of the project. Unity is compatible out of the box with Gear VR using a camera and the *“Virtual Reality Supported”* option in the build settings.
- We recommend Google VR 1.01:
  https://github.com/googlevr/gvr-unity-sdk

Unity 5

- Unity 5 is required. We recommend Unity 5.4.2:
  https://unity3d.com/get-unity/download/archive

Tip:
If you are using Unity 5.4+, you may have issues interacting with objects when building with GearVR (or any native Unity VR build), please see this video for more details:
https://www.youtube.com/watch?v=Nrz3De7qRmI

Starting With A New Project

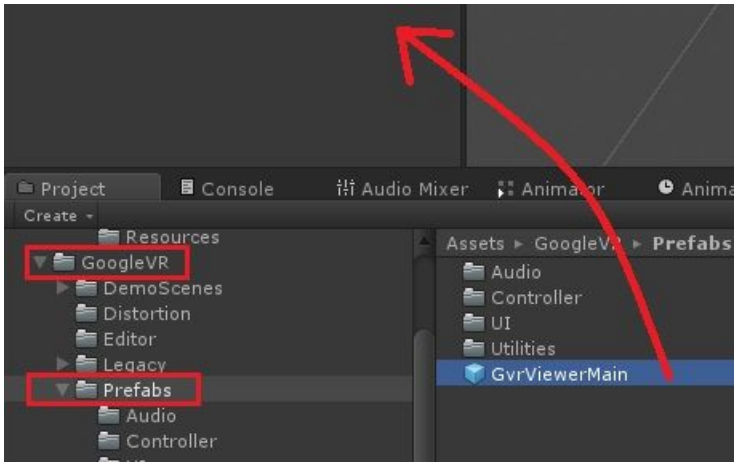
Video Tutorial: https://www.youtube.com/watch?v=Z4RVRwb8kGA

Step 1: Import the Asset
If you are starting with a new project, begin by importing this asset. This contains the necessary scripts and a Demo.

Step 2: Import the GoogleVR SDK into the project
This asset relies on the GvrViewer and GvrHead scripts, so GoogleVR version 0.8+ must be used. The Google Cardboard SDK can be downloaded at Github: https://github.com/googlevr/gvr-unity-sdk

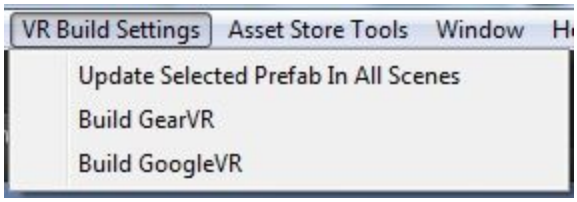
Previous to 0.8, the scripts were named differently with 'Cardboard' in the naming convention, for example *CardboardMain*. If your project is using an older version of Cardboard, please upgrade to 0.8+. See Starting with Existing Google Cardboard Project.

Step 3: Add the GvrViewerMain Prefab to the Scene
The GvrViewerMain script is required and is contained in a prefab located at */GoogleVR/Prefabs/GvrViewerMain.* Add this prefab to the scene:



Step 4: Select Main Camera and add the GvrHead script
In GoogleVR 1.0, the GvrHead script is added to Main Camera at runtime. We need to change this because we must change values on the script when changing between GoogleVR and GearVR. To fix this, simply select the gameobject "Main Camera"**,** click "Add Component", and add the GvrHead script.

Step 5: Select *Build GearVR* or *Build GoogleVR*
Now you are ready to use- no code needed. View the demo scene or just drag the Prefab **"VRMain"** into your project as the camera. To change between Cardboard and Gear VR, simply click on "VR Build Settings > Build GearVR" or "VR Build Settings > Build GoogleVR":



Tips:

- If you are building for GearVR, make sure to include your device signature file in /Plugins/Android/assets
- If you are building for GoogleVR, remember to set orientation Landscape Left in the build settings.
- If you are using Google's Spatial Audio, remember to set the spatializer in Edit > Project Settings > Audio.

Starting with Existing Google Cardboard Project

To use this asset with an existing Cardboard project, it needs to be updated to the GoogleVR SDK.

Video Tutorial: https://www.youtube.com/watch?v=6_EA2ckX1bI

Upgrade Instructions from Google:
https://developers.google.com/vr/unity/release-notes#upgrade_instructions

Because the *Cardboard* and *Plugins* folders must be deleted, we need to "Break Prefab Instance" for any prefab that is contained in the *Assets/Cardboard/Prefabs/* folder, such as *CardboardMain*.

Step 1: Locate Cardboard Prefabs
Search all of your Scenes and locate any Prefab that is saved in the *Cardboard/Prefabs/* folder- CardboardMain, etc.

Step 2: Break Prefab Instances
Select the prefab GameObject in the Hierarchy window, click on the toolbar *"GameObject > Break Prefab Instance".* Repeat the process for every scene in the project. Now that all the prefabs such as *CardboardMain* are no longer Prefab Instances, we can delete the *Cardboard* and *Plugins* folders.

Step 3: Determine if other assets are using the Plugins/ folder
Because updating Cardboard requires deleting the *Plugins/* folder, and it's possible that other assets in your project are using the *Plugins/* folder, you should determine the impact of deleting this folder. If other assets are using *Plugins/* then you could re-import those assets after deleting the folder to restore the deleted files.

Step 4: Close Unity
The Cardboard SDK files are most likely in use by Unity.exe. Close Unity completely before deleting any folders. For example, if you set Cardboard Audio Spatializer under *Edit* > *Project Settings Audio* > *Spatialzier Plugin,* then the audio DLL will be in use and cannot be deleted.

Step 5: Delete Cardboard and Plugins Folders
Delete the **/Cardboard/** and **/Plugins/** folders in your root Asset folder of your project. This removes the Cardboard SDK. See Google's link above for details.

Step 6: Import the GoogleVR & GearVR Camera Setup Asset
Now GoogleVR is ready to be imported, which is contained in this asset. Import this asset and proceed with Starting With A New Project.

Building A Multi-Scene Setup

If your project has multiple scenes, with a "VRMain/CardboardMain" prefab in each scene, you will need to take additional steps to update those prefabs before building your project.

Tip: Consider moving the VRMain prefab into it's OWN scene, Scene 0 in the build order, and then loading content & levels additively. In this manner the VR Head will always be loaded, even in between levels, so values such as head rotation will not be lost between levels.

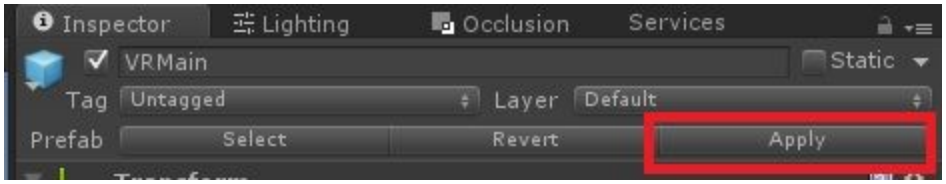
- **Asset Store Tool:** https://www.assetstore.unity3d.com/en/#!/content/73891
- **Free tutorial:** https://www.youtube.com/watch?v=lhCPBILR5mg

Switching between GearVR and GoogleVR requires changing values on the VRMain Prefab, specifically the GVRViewer and GVRHead scripts. When you use *"VR Build Settings* > *Build GearVR"*, it changes values on the VRMain gameobject but the prefab is not updated. If you are using VRMain in multiple scenes, you will need to update VRMain in each scene. I've included a tool to help with this.

Step 1: Select *Build GearVR* or *Build GoogleVR*
To change between Cardboard and Gear VR, simply click on "VR Build Settings > Build GearVR" or "VR Build Settings > Build GoogleVR"

Step 2: Update Prefab Instance

Select both the *VRMain and GvrViewerMain* GameObjects in the scene and click “Apply” in the Inspector after the word *Prefab:*



When you update a prefab instance, it updates all instances in the currently opened scene. In other scenes the prefab will not update, until you open the scene and select the Gameobject and click "Revert" to revert the prefab to it's newly updated state.

Step 3: Update Selected Prefab In All Scenes

I've added an option under *VR Build Settings* called *"Update Selected Prefab In All Scenes"*. This button will take the GameObject which is currently selected, and then open every single scene (that is added to the build list) and look for that gameobject. If it finds a match, it will "Revert" the prefab instance. This will update each “VRMain” prefab in all scenes:

- Select “VRMain” and click *‘VR Build Settings > Update Selected Prefab In All Scenes’*

- Select “GvrViewerMain” click *‘VR Build Settings > Update Selected Prefab In All Scenes’*

Now all the scenes in the build settings have updated the VRMain prefab, and all scenes will either be configured for either GearVR or GoogleVR. You are ready to build.

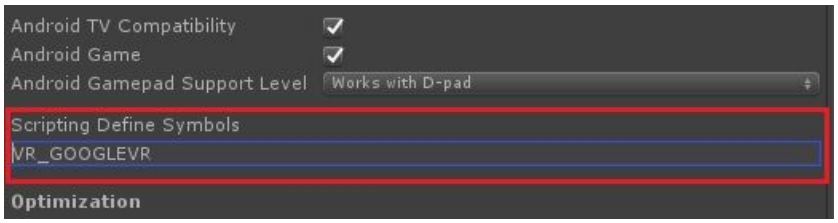
Platform Dependent Compilation

We have included Platform Custom Defines so that it is possible to write code which is solely executed on either GoogleVR or GearVR. More information on custom defines can be found here:
https://docs.unity3d.com/Manual/PlatformDependentCompilation.html

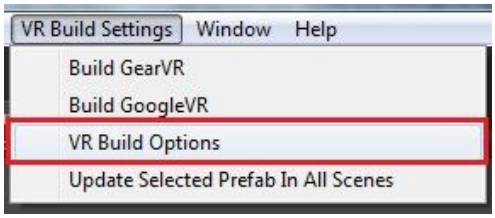
Platform Custom Defines

Property:	**Function:**
VR_GOOGLEVR	#define directive for calling/executing code for GoogleVR platform.
VR_GEARVR	#define directive for calling/executing code for GearVR platform.

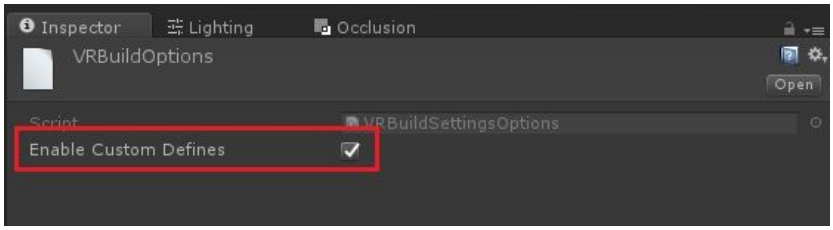
For this to work, 'Scripting Define Symbols' will be modified in Player Settings:



This feature may be disabled, for example if you want to manage define symbols yourself or with a tool like our Custom Define Manager. Click *VR Build Settings* > *VR Build Options*:



This will select the Options scriptableobject in the Unity Inspector. Currently there is 1 option, enable or disable Custom Defines:



The *VRSimpleAutowalk.cs* script contains an example of how to use custom defines:

```
#if VR_GEARVR
        Debug.Log("GearVR Mode Enabled");
#elif VR_GOOGLEVR
        Debug.Log("GoogleVR Mode Enabled!");
#endif
```

3. Demo

The new Demo contains 3 scenes, a main menu, level 1 and level 2. This shows the functionality of a multi scene setup, some UI buttons, and an AutoWalk script. To ensure the demo works properly, all 3 scenes need to be added to the build settings. Build order settings should be MainMenu = 0; Level01 = 1; Level02 = 2.

4. How It Works

This asset works from a single editor script, *VRBuildSettings.cs*. Gear VR works by using a single Unity camera and then setting the “Virtual Reality Supported” build option. Google VR SDK works by rendering with multiple cameras. However, GvrViewer Class has a “Virtual Reality Enabled” mode which disables the multiple cameras and renders with only a single fullscreen camera. This is what we need for Gear VR, except GvrHead script needs to stop tracking rotation because when we selection “Virtual Reality Supported” from the build settings, rotation is tracked automatically. Take a look at the script in */GoogleVRandGearVR/Scripts/Editor/* for more information, the code is well commented.

5. Further Info

For a video tutorials related to this asset, please click here:
Intro & Tutorial: https://www.youtube.com/watch?v=Z4RVRwb8kGA

Join our VR community here for VR tutorials and videos:
https://www.youtube.com/nurfacegames/

For any questions or support, please email:
nurfacegames@gmail.com