MergeCubeSDK 2017.2 Upgrade Guide

This guide aims to helps users migrate their AR projects from Unity 2017.1 and older to Unity 2017.2.

The newest Unity has introduced Vuforia as a built-in plugin with new settings which must be set up in order to use. This conflicts with our old SDK which used Vuforia as a standalone component. However, with the new MergeCubeSDK 1.1.4, we have made several tools to help this upgrade process go smoothly.

To get started, let's open the previous 2017.1 or older Unity project. We are going to change a few things ahead of time because when we upgrade, we are going to face a few compiler errors that will stop the new SDK from properly initializing before it can grant access to the helper tools.

Once it is open, let's bring in the new MergeCubeSDK version 1.1.4. Then once it comes in, we are going to navigate to the new Merge menu in the menubar. Then let's select the MergeCubeSDK Update Helper from the list. This tool that opens has two components that we will be using. The first is the MergeCubeSDK update helper. This button will move some of the Merge Cube components to a different folder structure that is a little cleaner and easier to manage when working with multiple 3rd party plugins. We'll click that and wait a few moments to complete.

Then, we also have an experimental Vuforia Cleaner button. This button is optional. If clicked, it will remove all of the old references to Vuforia in the project files. That way, when brought into the 2017.2 Unity, it will not conflict with the new way Vuforia is being handled. If clicked, there will be various errors in the console about Vuforia being missing. This is fine, as we will be bringing Vuforia back in the next few steps.

If you do not wish to use this button, you may refer to Vuforia's Migration Guide located here: https://library.vuforia.com/articles/Solution/migrate-vuforia-62-to-65.html

Once the helper tools have been run, we can now close the project and begin the migration to the newest version of Unity.

After everything has reimported, we will need to import the new Vuforia package. In Unity 2017.2, this can be accessed via creating a new Vuforia ARCamera by right clicking in the project view and selecting Vuforia -> ARCamera.

Once clicked, it will activate a prompt asking to import Vuforia. Accept that and let it import the new components. Next, the old ARCamera in your scene will need to be replaced with the new ARCamera. These aren't prefabs in the project anymore, and will need to be modified by hand. Additionally, the hierarchy of the ARCamera seems to have changed from an empty gameobject

called ARCamera with a camera object as a child, to simply just a camera itself named ARCamera. Keep this in mind while re-attaching any components you may need, and make sure the new ARCamera has matching transforms to the old camera.

Next, the Vuforia Configuration file was likely reset during this transition, so head over to your Resources folder and click on the Vuforia Configuration file to make sure your license key and related settings have all been changed properly. You may also notice that there are 3 additional datasets that have been added to the datasets section. These are Vuforia's new default ImageTarget collections, and they should be deactivated. (This should leave us with no active datasets and that is OK, our SDK will handle this behind the scenes)

Next, Vuforia itself needs to be "Activated" in the player settings. Open your Player Settings by going to Edit -> Project Settings -> Player Settings, and then navigating to the XR Settings at the bottom. Then enable the Vuforia Augmented Reality option. Do note, that there have been mixed reports of this setting getting disabled when switching build platforms. If you are experiencing errors upon building out, please check that this setting is enabled. It is possible that it has been disabled somehow.

Now let us also make sure that the auto-graphics api has been disabled and that the only api in use is OpenGLES 2.0. This is the only supported api for the moment. Any others should be removed from the list.

And finally, Vuforia seems to have stopped automatically populating the Camera permission dialog for iOS. So if your build target is set to iOS, navigate to the Other Settings section in the Player Settings and ensure that the Camera permission has a short description on why your app is using the camera. Typically this is something as simple as: "Used for image recognition." This dialog is just to ensure that the end user understands why this app needs to use various features of the phone. If this setting is not set, the build will fail to deploy.

With all of that done, your project should now be ready to use in the newest version of Unity! There are quite a few steps, but we hope that this guide has been able to help. If you have any questions or concerns please feel free to contact us at:
developer@mergevr.com

We'd love to hear how we can better assist you and we look forward to the wonderful things you'll create!