GLAMVR AR WORKSHOP

Author: Dominic Manley (dominic@dominicmanley.com)

Date: 26th August 2016

Below is a write-up of the process I went through in the workshop for anyone who would like to download Unity and try it out for themselves. I altered the process a little to exclude the paid Unity plugin Text Mesh Pro that I made use of on the day (with one gotcha, explained at the bottom). Instead, I've used the free Canvas UI tools that come with Unity.

1. Download Unity from https://unity3d.com and install it. Get the Personal edition which is free. You can use either the PC or the Mac version.
2. Download the Unity Vuforia package from https://developer.vuforia.com/downloads/sdk
3. Launch Unity and create a new Unity project. Choose 3D and disable Unity Analytics.
4. Locate the Vuforia Unity package file you downloaded earlier, double click it and import it into the current project. This should add some files and folders to the Project pane.
5. Select the Main Camera from the Hierarchy pane, right click and delete it. We're going to replace the default camera with the Vuforia one.
6. The Vuforia AR camera is under Vuforia->Prefabs->ARCamera in the Project pane. Drag it into the Hierarchy pane and the new camera will be added to the Scene.

7. Now visit https://developer.vuforia.com/ again and Register for a developer account. You'll need to do that to access the License Manager and Target Manager under the "Develop" tab.
8. When you have registered and are logged into the Vuforia Developer Portal, go to the License Manager and create a license key for your app. Choose "Development" as the project type, give the app a name, choose "Mobile" as the device type.
9. Copy and paste the license key into the "App license key" box you'll see in the Inspector pane in Unity, if you select the ARCamera.
10. Hit the play icon and you the Game window should show you the output from the primary web cam plugged into your computer.
11. Now's a good time to save your scene file. File -> Save Scene.

    Tip: I recommend saving all the scenes, scripts and assets for your project into an _Project folder. If you like you can then put scene files into a Scenes/ sub-folder, scripts into a Scripts/ sub-folder, and so on. It's best to leave the root folder a free-for-all for any plugins and assets you import from the Unity app store. Some tend to depend on their folder structure being maintained so if you try to neaten things up... things can break! So just let them take over the root and keep all your own files specific to this project under _Project (the underscore pushes it to the top of the Project pane list). To create a new folder in the Project pane click the Create button and select Folder.
12. Now go back to the Vuforia Developer Portal and go to the Target Manager page (under Develop). Create a new Database there. Again choose "Device" as the type.
13. Click on the new database link and you'll be taken to a list of targets which will be currently empty. This is where you upload the photos of

the drawings on your badges. If you like, you can experiment with uploading other types of photos to see what works and what doesn't.

14. Click the Add Target button. Choose Single Image (again, you can experiment with the others later), click the Browse... button and locate the image file of your badge. Enter a width of 1 and click the Add button. You can add more targets to your database file by repeating this step.
15. When you've finished adding targets, click the "Download Database (All)" button, select the "Unity Editor" platform and save the Unity package file to your computer.
16. Locate the database Unity package file, double click it and import it your Unity project.
17. Now drag the "ImageTarget" prefab from the Project pane (under the Vuforia->Prefabs folder) into the Hierarchy pane.
18. With the new ImageTarget selected, choose your database from the dropdown menu in the Inspector pane. When you do this you should see the first image target in your database displayed in the scene. You can swap between the different targets in your database using the "Image Target" dropdown.
19. While the ImageTarget is still selected, set the x, y and z position to 0, 0, and 3. Then set the x, y and z scales of the ImageTarget to 1, 1 and 1.
20. If necessary, rotate the image target by 90 degrees on the y axis.
21. Next, we're going to add some text (your name and an interesting fact about yourself) beneath the image target. This text will augment over the ImageTarget when the Vuforia detects it in the camera view.
22. From the main Unity menu select GameObject -> UI -> Text. This will add a Canvas object, with a child Text object, and an EventSystem object to your Hierarchy. The EvenSystem object is needed in a Unity

scene whenever you're dealing with UI objects and capturing events such as mousedowns.

23. Select the Canvas object and change its "Render Mode" to "World Space" in the Inspector. Also change the Dynamic Pixels Per Unit value to 100. This will keep text sharp even when we shrink the size of the Canvas and Text objects right down (which we're about to do).
24. Next, in the Hierarchy pane, drag the Canvas object and its child Text object over and onto the ImageTarget to make them both a child of the ImageTarget. This will make all positions and transforms we apply to the Canvas and Text relative to the ImageTarget.
25. Now make sure the Canvas is selected in the Hierarchy pane and then in the Inspector set the x, y and z positions to 0, 0 and -1.1 (or 1.1, 0, 0 if you rotated your ImageTarget earlier).
26. Then set the Canvas x rotation to 90 (and its z rotation to 90 too if you rotated the ImageTarget) and then its x, y and z scales to 1, 1 and 1.
27. Now select the Text object and add your name and interesting fact to the Text box. Check the Best Fit box, set the alignment to center and middle, the font size low to about 5 and finally, choose a bright colour for the text.

    Warning: if you have lots of ImageTargets in your scene the "Best Fit" option can cause a "all the needed characters do not fit onto a single texture" error in Unity. If that happens you're best de-selecting "Best Fit" and manually setting the font size. You shouldn't encounter this if you're just working with a few ImageTargets.
28. If you want to create more targets for your app to detect right click on the ImageTarget in the Hierarchy pane, copy, then right click again and paste. This will create a copy of your ImageTarget (named "ImageTarget(1)"). You then only need to move the new target to another space in

the scene (e.g. add 2 to its x position), change its ImageTarget in the inspector and tweak the text/sizes of your Canvas/Text.

29. FINALLY, if you've tried pressing play already, you'll notice the app doesn't detect or augment anything yet. It's easy to forget this next step...
30. Select the ARCamera in the Hierarchy pane and in the Inspector check the Load YOUR_DATABASE_NAME box and then the subsequent "Activate" box that appears below it. You'll find these boxes under the "Database Load Behavior (Script)" area in the Inspector.
31. Click the play button and try it out! :)

Now you're familiar with Unity and the the Vuforia SDK, why not try adding a box to the ImageTarget and creating a switch! To avoid having to code anything just yet, you can use the plugin I used in the workshop. It's called "Gameflow". You'll find it in the Unity Asset Store (for about US$20). You'll find the Asset Store pane in Unity (hint: look under "Window" in the main menu). You'll need to register with Unity first to buy and download assets. Then import the Gameflow plugin and get stuck in! Create a cube, place it under the ImageTarget and then use Gameflow's if... else... logic to activate/de-activate your text.

If you get stuck, try reading the Gameflow and Unity documentation. You'll also find information there for how to build your app onto different platforms.

Tip: Android is the easiest mobile platform to get going… you can simply copy the generated .APK files onto your device. Or you could simply build a .EVE Windows app file, or a .APP Mac app file.

Final Note: there is one gotcha I discovered to using Unity's own Canvas UI and Text objects (instead of the Text Mesh Pro plugin)… when a target is detected it will show ALL the Canvas text objects! You may notice this if you experiment with multiple targets. The issue is discussed and solutions offered here: https://developer.vuforia.com/forum/qcar-api/unity-46-ui-canvas-Vuforia.