



# **B-INN-000 - Workshop: Video 360°**

## **AR/VR Workshop: Video 360°**

It **V**irtually **R**uns!





# AR/VR Workshop: Video 360°

---

Language: C#

---



**Your repository must contain all of your source files but NO useless files.**

No, you won't find long and useless pages here. Let's begin exercise #1.

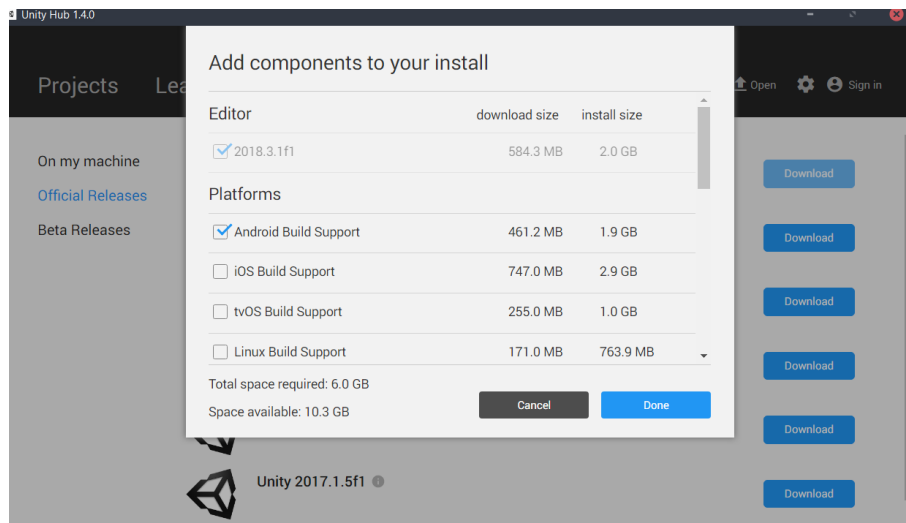


## Exercise 00: Installation

---

Download Unity Hub : <https://store.unity.com/download>

Download Unity 2019.x or after with android package (on unity hub)



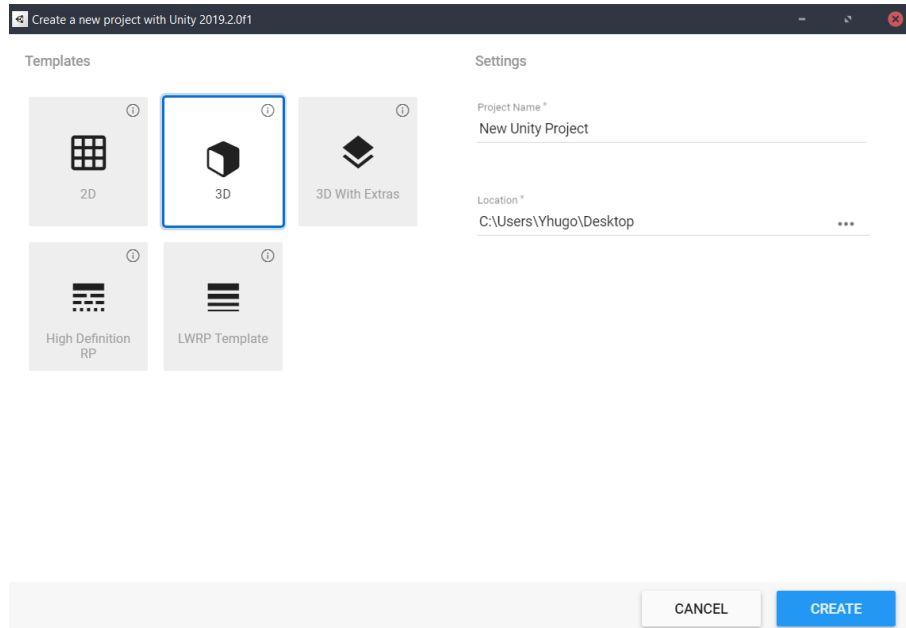
Download android studio if it's not installed



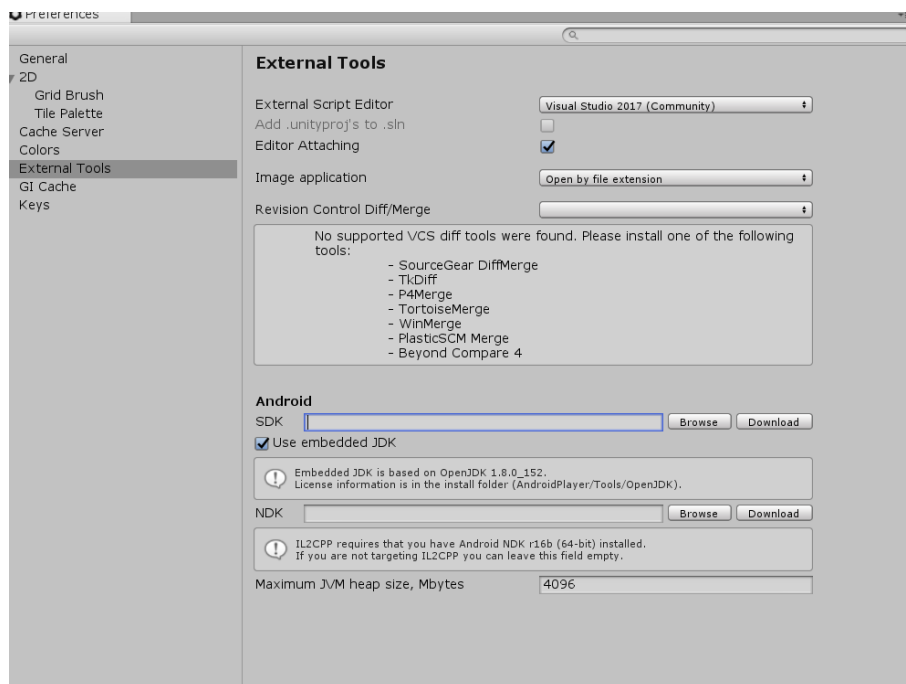
**Watch out where you install Android SDK**



## Create a unity project in unity hub with a "3D" template



Link Android sdk with unity: Go to edit->preference->external tool and set android sdk path



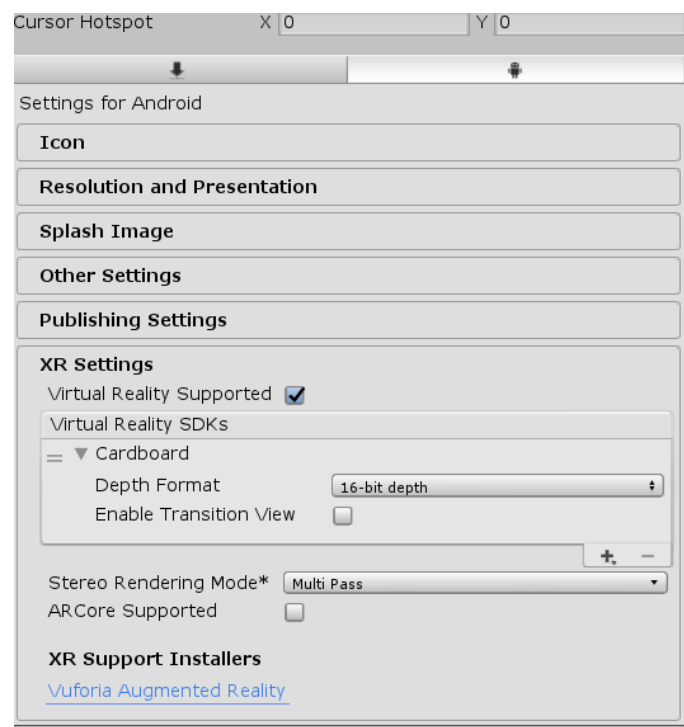


Set up the project setting for build an android project Switch the build setting to Android

Set color to 32 bits

Set the player setting to :

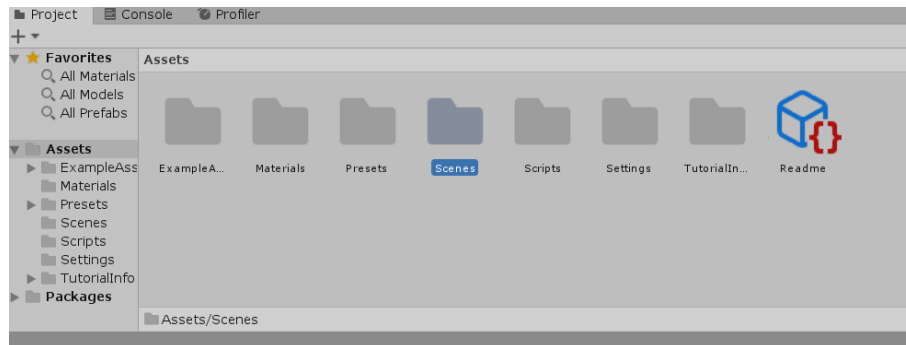
- Other setting :
  - Package name to com.poc.ARVRWorkshop
  - Minimum API level to your phone
- Xr setting :
  - Activate virtual reality
  - Add card board to VR SDK





## Exercise 01: Set Up the scene

Go to the Asset Explorer in the editor and create a new scene in the "Scenes" folder name it "VrVideo"




**When you create something in the project explorer it's : right click  
-> create**



## Exercise 02: Save and Test the App

Save your scene by Control + S on the scene, plug your phone to your computer. Next you'll need to build and run your application by clicking on : File -> Build and Run

 **Your phone need to be in developper mode, with the USB debugging mode activate**

 **If only one eye is rendering verify:**

- **Camera is rendering both**
- **BuildSetting -> Player Setting -> Xr Settings -> Stereo rendering mode = Single Path**

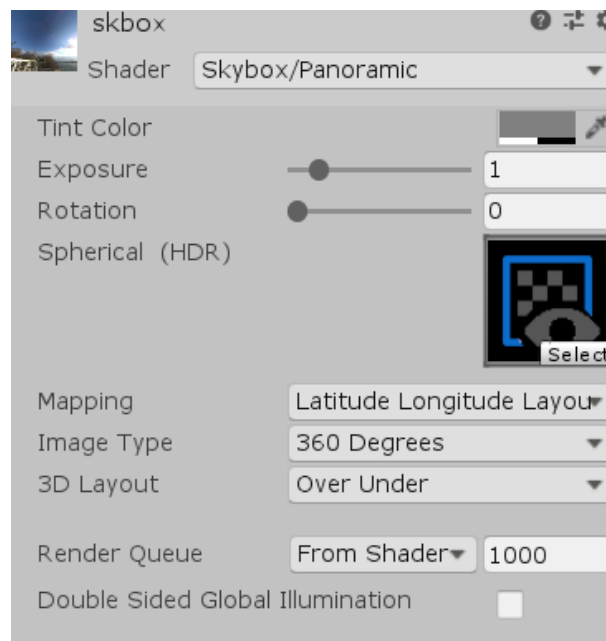
when the build will finish if no error occurred your phone will automatically start the builded application

## Exercise 03: 360 Video

First you need to import the 360 video to your unity project :

- Create a folder "Videos"
- Drag and drop the videos in the folder for the begining you can import only **"3D\_Waterfront.mp4"**

the sky of unity use a skybox and sky box is a type of material so create a material name "360Skybox" with the shader set to : "Skybox/Panoramic" set the 3d layout



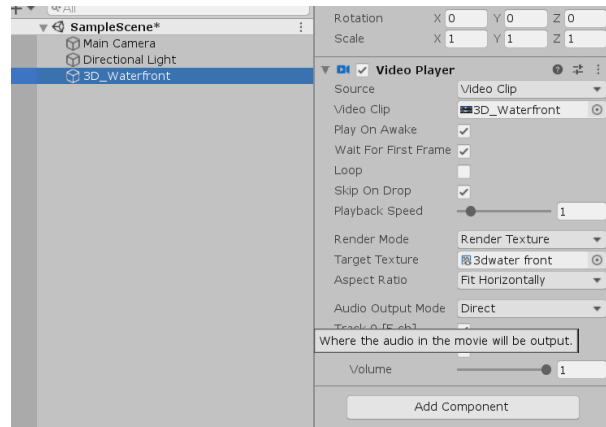


⚠ if 3d layout don't appear you need to activate xr in player setting

then you need a spherical texture and updated at runtime so create a render texture,

set the correct resolution to texture (3840 x 2160 for the waterfront video)

now place the video in the scene



now set the render mode of the video player to render texture and place your render texture in

## Exercise 04: Add UI

we have the video but we need to interact with them, so for interacting with video we will create some UI. we need a Play Pause UI, A next video UI and a UI pointer

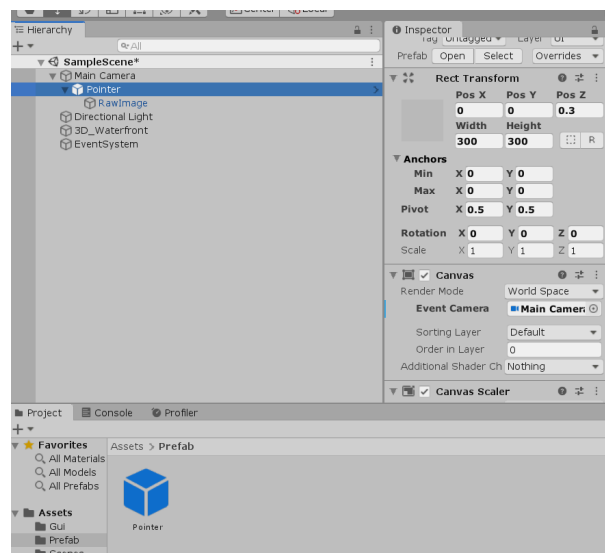
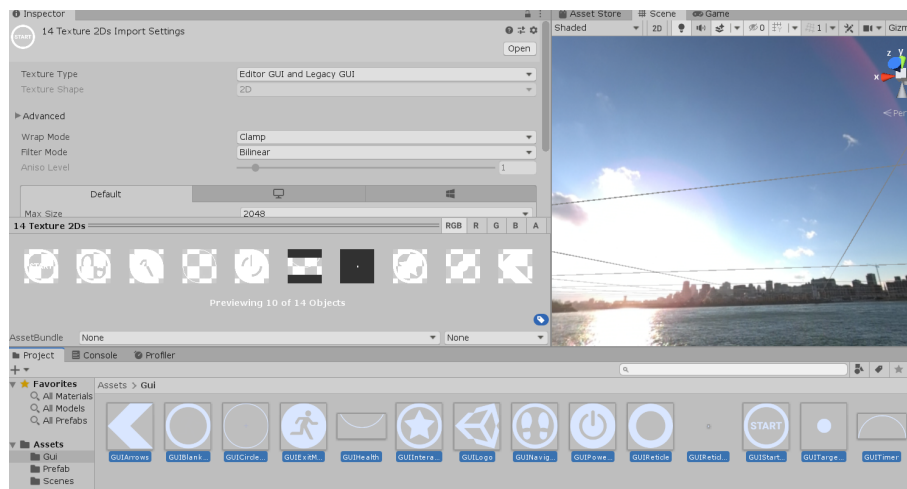
let's start with the pointer UI create a "UI-> Raw Image" in hearrchy rename the canvas pointer and set the Canva's Render mode to "World Space" and the camera to your main camera.

Now Import the "GUI" folder to Asset's root. Got the gui folder select all the Guis and set the texture type to "Editor GUI and Legacy GUI" in the inspector





So now set the row image of point to your chosen GUI IMAGE and modify the color to a more visible color.



Now save the Pointer Game object as prefab in a Prefabs folder at the assets root and put the game object as son of the main camera. Create two "ui->canvas" name "PlayPause" and "Next" and a component's canvas with the same parameter as Pointer's canvas component

- Set the two ui to layer ui
- Put a the next gui Image on Next
- save them to prefab folder
- add box collider (with the correct size)



Let's create some script We want to create a script who set play Image to PlayPause if the video is playing and pause when...  
This script will be name play "PlayDetector"

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.Video;

public class PlayDetector : MonoBehaviour
{
    public Texture playTexture;
    public Texture pauseTexture;
    public VideoPlayer player;
    public RawImage rawImage;

    // Update is called once per frame
    void Update()
    {
    }
}
```

now create an other script this one will detect if you are looking on a UI or not This script will be name "IsOnUI"

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class IsOnUI : MonoBehaviour
{
    GameObject mainCamera;

    void Update()
    {
    }

    void OnUI(RaycastHit hit)
    {
        Debug.Log("you are loocking at : " + hit.transform.name);
    }
}
```

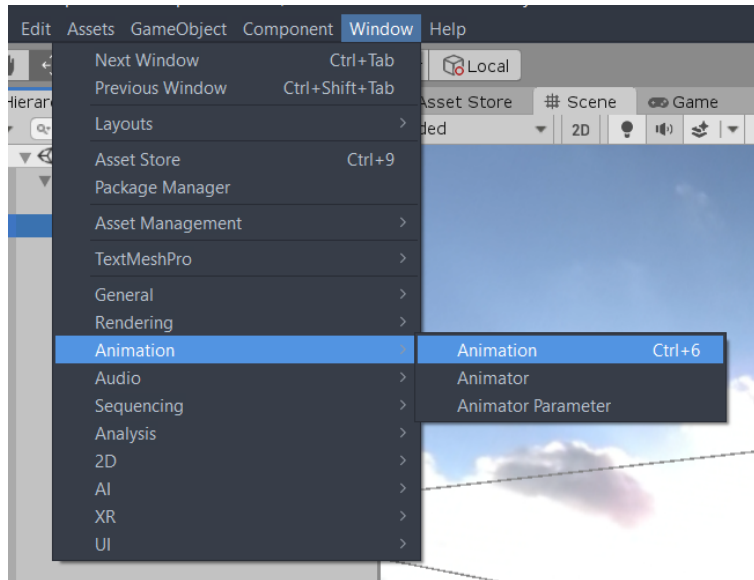


May be use Raycast function with the camera on UI Layer, could be the solution

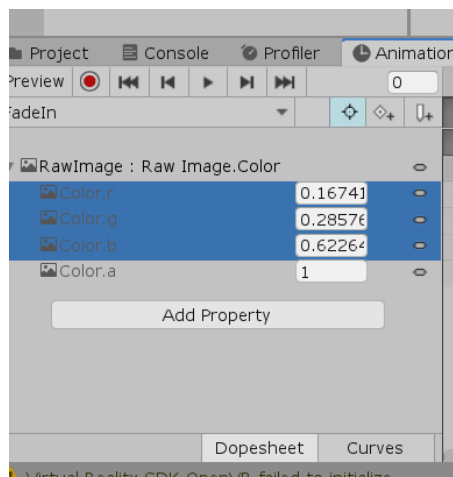


## Exercise 04: Add Animation On UI

now we need to create a transition when your on a ui for this crate animator controller name "Image", put the animator on object with RawImage click on Window-> Animation -> Animation



select the "raw image" a go on the animation window click on create name the animation clip FadeIn add property-> Raw Image -> color delete all the r,g,b parameter



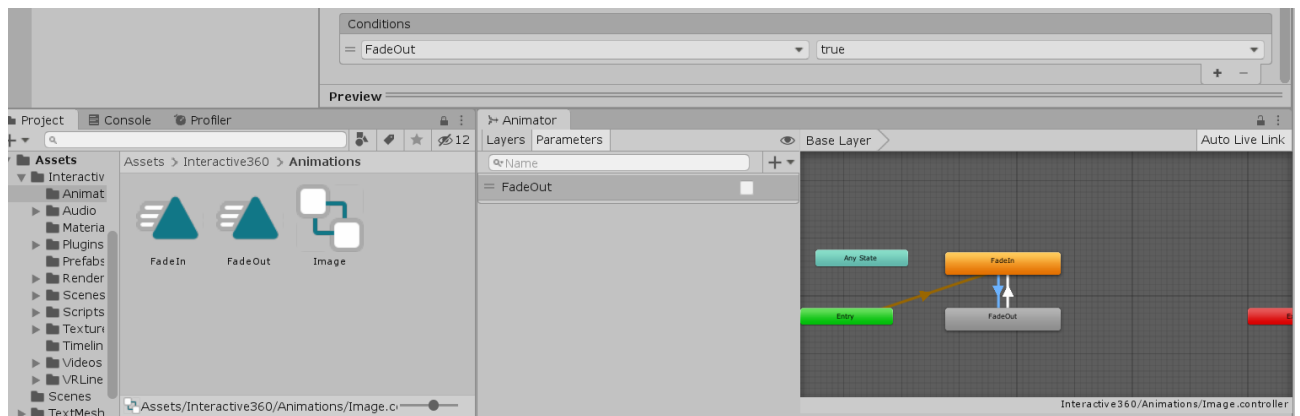


set the alpha to 0 at 0:00 and to 1 at 1:00

⚠ you can navigate to the created key frame with the arrow button

now create an other animation clip by clicking the arrow near FadeIn the name of this new animation clip is : FadeOut it has only alpha parameter and at 0:00 alpha is equal to 1 and 0 at 1:00 go to animator window (dubble click on Image animator in the assets) create a parameter bool named FadeOut by default he is false, add transition between :

- FadeIn and FadeOut this transition as for condition FadeOut parameter is false
- FadeOut and FadeIn this transition as for condition FadeOut parameter is true



now modify the Method "OnUI" in IsOnUI whith this :

```
void OnUI(RaycastHit hit)
{
    var animator = hit.transform.GetChild(0).GetComponent<Animator>();
    var rawImage = hit.transform.GetChild(0).GetComponent<RawImage>();
}
```

now this method need to play fading and tell something on the console if the image has faded out

⚠ add animator on every UI button image PlayPause Next



## Exercise 05: Add UI Action

now you need to add a function who detect the name and it's "PlayPause" he do play pause and if next it change the video