

DEVELOPING VIRTUAL REALITY AND AUGMENTED REALITY PROJECTS WITH UNITY3D

Laura SAVU^{1*}

ABSTRACT

In the current article we present how we can develop games with Unity for Augmented Reality and Virtual Reality. The devices that we have been working with are a HoloLens and the Acer Immersive Headset. Unity is currently the most used game development platform. The scripts are developed in C# programming language. This article shows step-by-step how to create a project in Unity, configure the settings for building a Universal Windows Platform Application that will run on HoloLens, run the project on the device directly from Unity, generate the UWP build and open it in Visual Studio, install the App on the device by running the project on HoloLens using direct device option or remote machine.

KEYWORDS: *Augmented Reality, Virtual Reality, Mixed Reality, HoloLens, Unity, Visual Studio, Physics, 3D, Holograms.*

INTRODUCTION

This paper presents how to develop for Augmented Reality. The device that was used is HoloLens, produced by Microsoft, it is a Windows 10 running device, being itself a PC as there is no need to plug it to a computer, as we do with all the virtual reality headsets. Regarding products and tools, we created the projects in Unity and we generated the build to be opened in Visual Studio, which is Microsoft Integrated Development Environment. This article walks the reader step-by-step through all the phases needed to create, test, deploy and run a Unity project on a HoloLens device.

1. Create a Unity Project

(Requirements: Visual Studio 2017, Unity3D, Windows 10 ,Devices: HoloLens and/or Mixed Reality headset).

In the next section we will use Unity game development platform to create our project that will run on HoloLens.

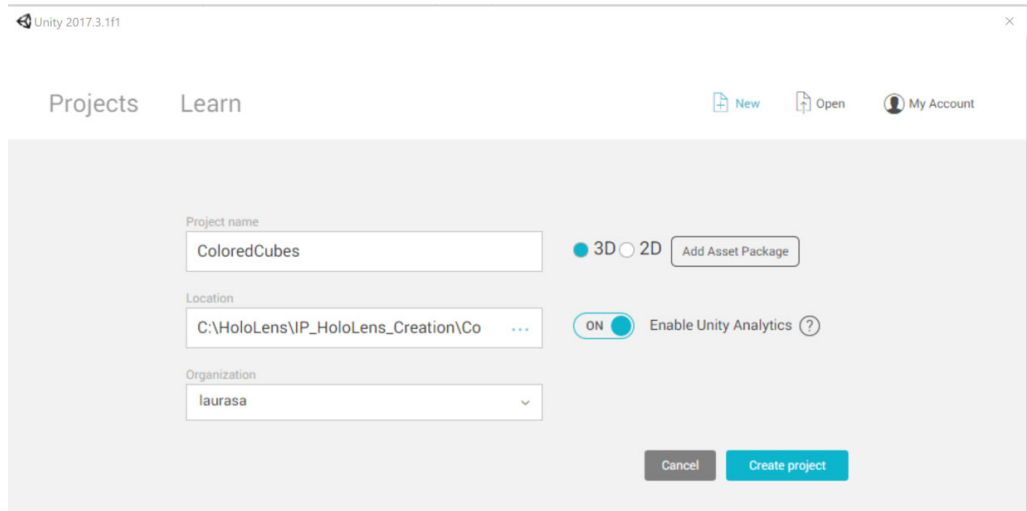
Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Inc.'s Worldwide Developers Conference as an OS X-exclusive game engine. As of 2018, the engine has been extended to support 27 platforms. The engine can be used to create both three-dimensional and two-dimensional games as

^{1*} corresponding author, PhD, Microsoft Bucharest, savu_laura@microsoft.com

well as simulations for its many platforms. Several major versions of Unity have been released since its launch, with the latest stable version being Unity 2018.2.18, released on November 30, 2018.

We are implementing Gaze and Air Tape to action on holograms.

Open Unity and create a new 3D Project.



Now let's download Mixed Reality Toolkit.

The Mixed Reality Toolkit is a collection of scripts and components intended to accelerate development of applications targeting Microsoft HoloLens and Windows Mixed Reality headsets.





Download MRTK: <https://github.com/Microsoft/MixedRealityToolkit-Unity/releases>

2017.2.1.1 Patch Release

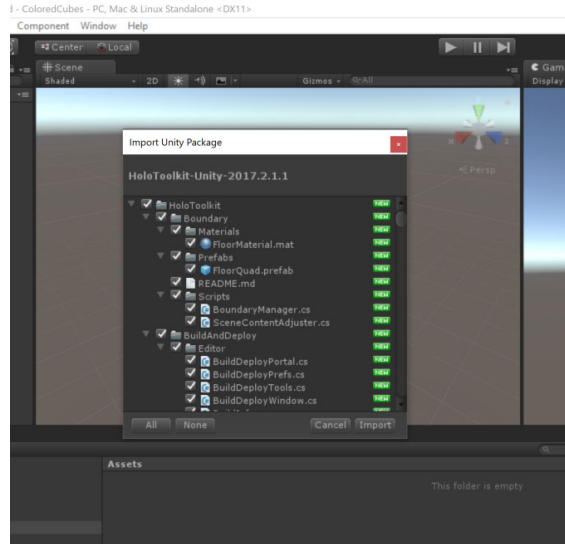
 StephenHodgson released this on Jan 16 · 7 commits to master since this release

Assets

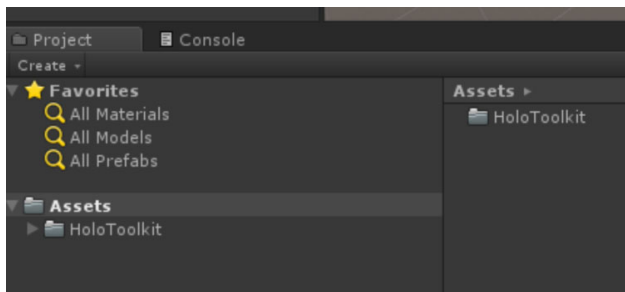
 [HoloToolkit-Unity-2017.2.1.1.unitypackage](#)

 MixedRealityToolkit-Unity-2017.2.1.1	1/26/2018 12:17 P...	File folder	
 HoloToolkit-Unity-2017.2.1.1.unitypackage	1/26/2018 12:15 P...	Unity package file	13,850 KB
 HoloToolkit-Unity-Examples-2017.2.1.1.unitypackage	1/26/2018 12:16 P...	Unity package file	31,665 KB
 MixedRealityToolkit-Unity-2017.2.1.1.zip	2/18/2018 8:19 PM	WinRAR ZIP archive	321,139 KB

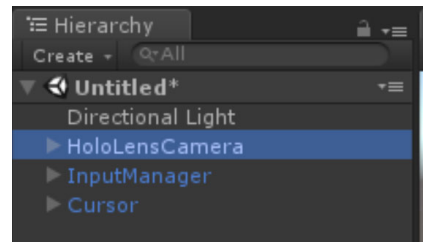
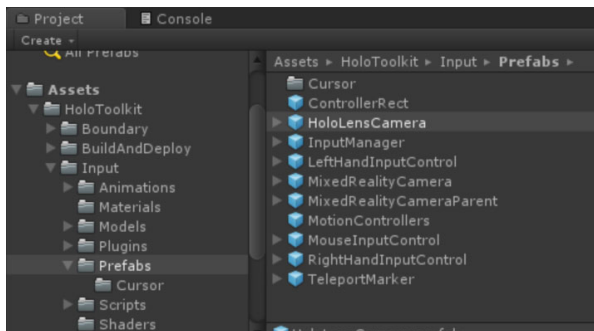
Drag and drop HoloToolkit in the Project Panel. You can see all the resources in the package. Click on Import button.



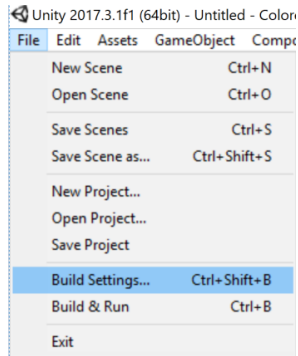
You have the HoloToolkit in your Project:



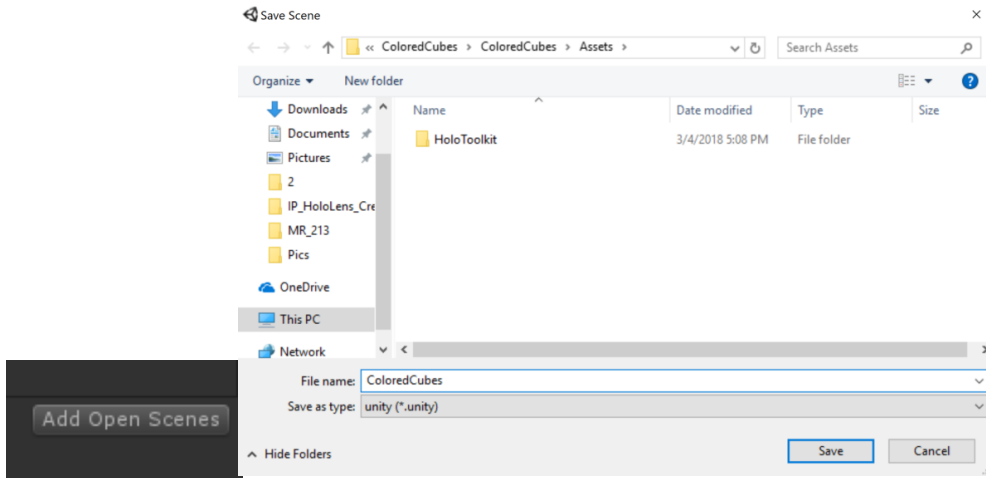
Delete Main Camera from your Scene. Drag and drop the HoloLens Camera from HoloToolkit\Input\Prefabs. Drag and drop the InputManager prefabs as well. Also drag and drop the Cursor prefab.



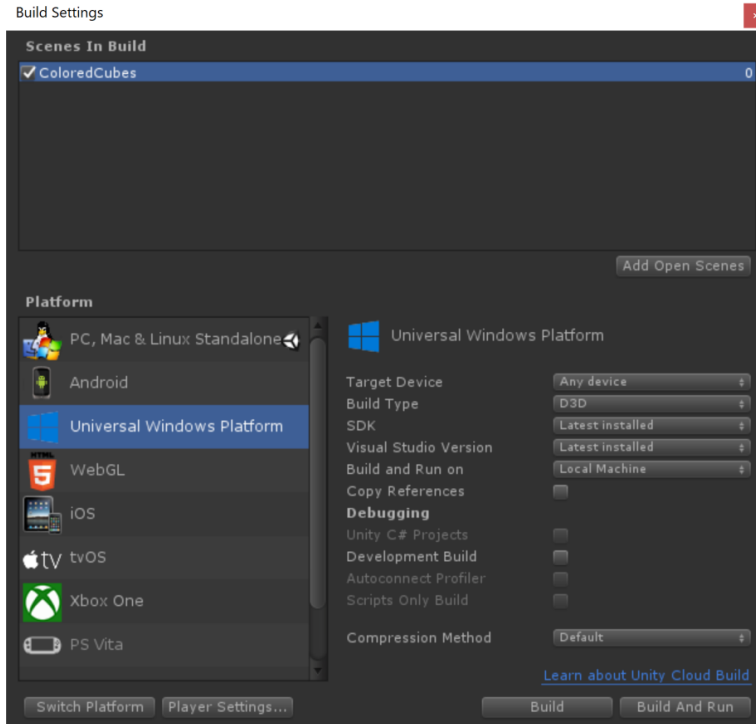
2. Enable Virtual Reality Support



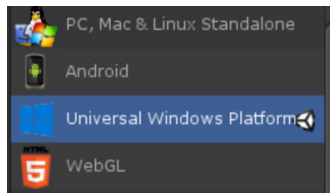
Click on Button “Add Open Scenes”



Select the Platform to UWP



Click on Button Switch Platform. The Unity icon will be on the selected platform.



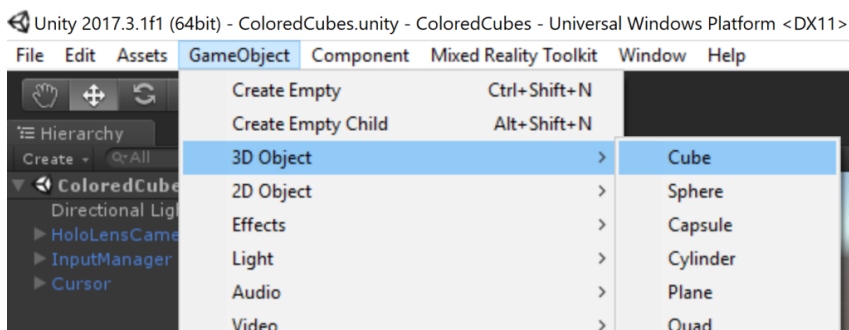
Click on button “Player Settings”

Now click on Player Settings button and you will have in Inspector the settings for the Player. Click on Windows Store green icon, check the option Virtual Reality Support. If there is nothing selected, click on the little plus sign and add Windows Holographic.



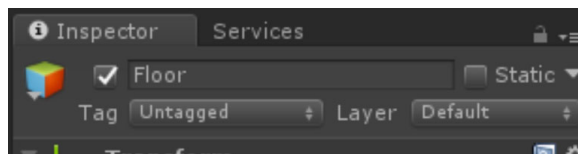
Close Build Settings.

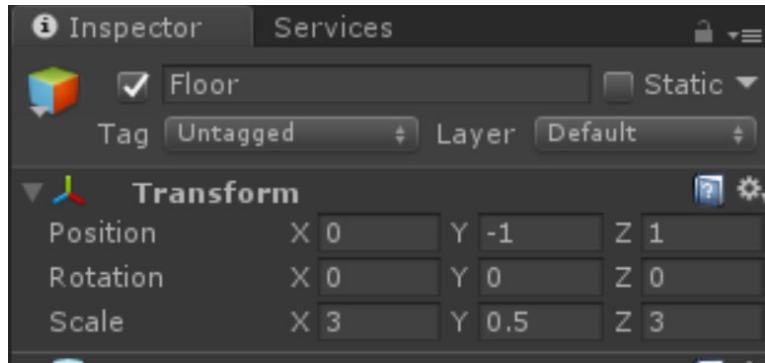
Under Hierarchy, click on Create and chose 3D Object => Cube.



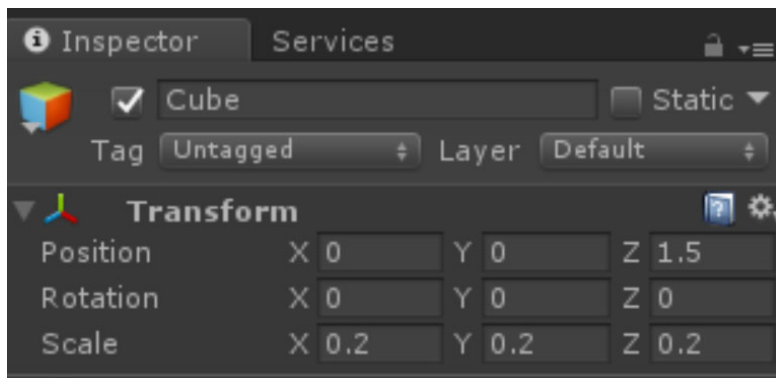
Select the Cube GO in Hierarchy. You will see in Inspector all the properties of the Cube.

Rename it to Floor and set the following values for the Transform properties:



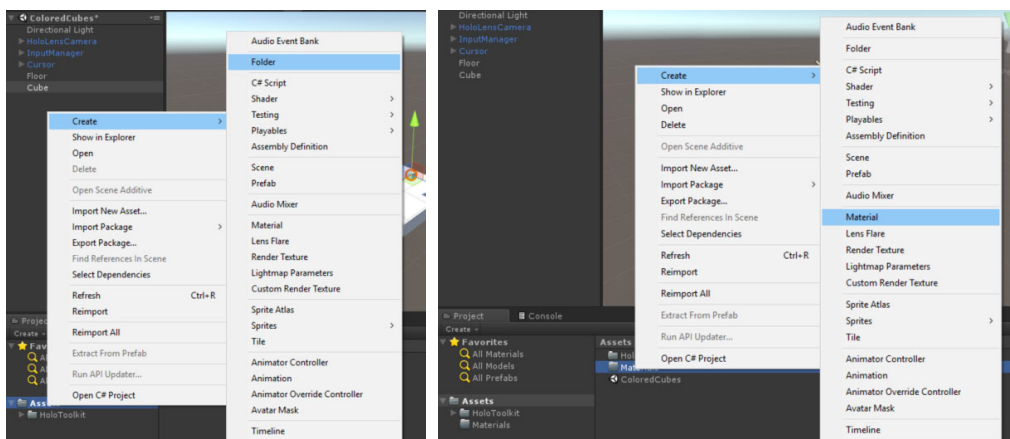


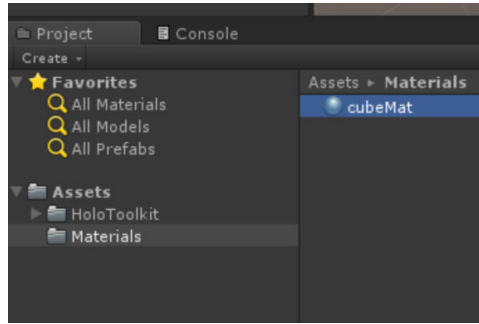
Add one more 3D GO, Cube and set the following values for Transform properties.



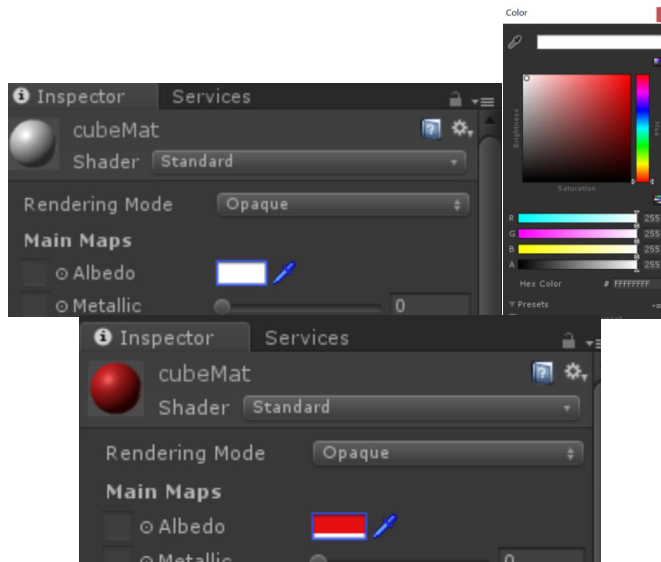
Create a material for the cube.

In Project panel, click Create => Folder and name it “Materials”, for example. Right-click on the folder. Right-click on the folder and Create => Material. Name it cubMat.

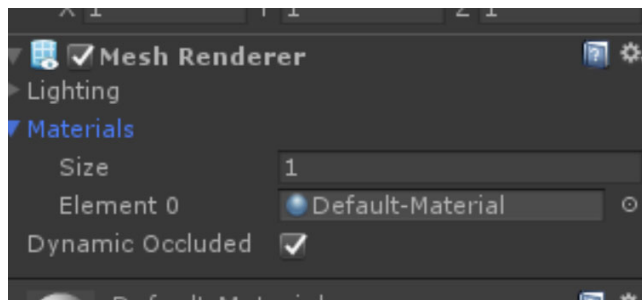




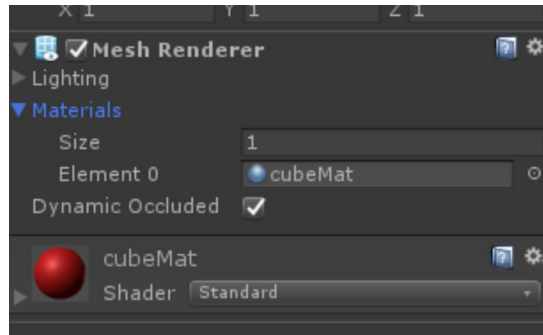
Select cubMat and, in Inspector click on the PickupColor control and chose Red.



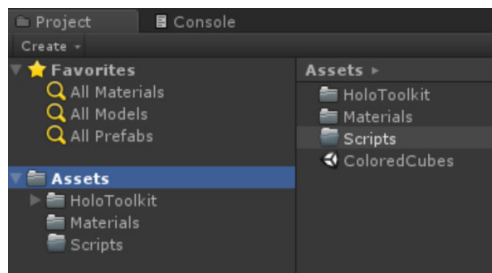
Select the Cube GO in Hierarchy, look at Inspector, in Mesh Renderer section, expand Materials



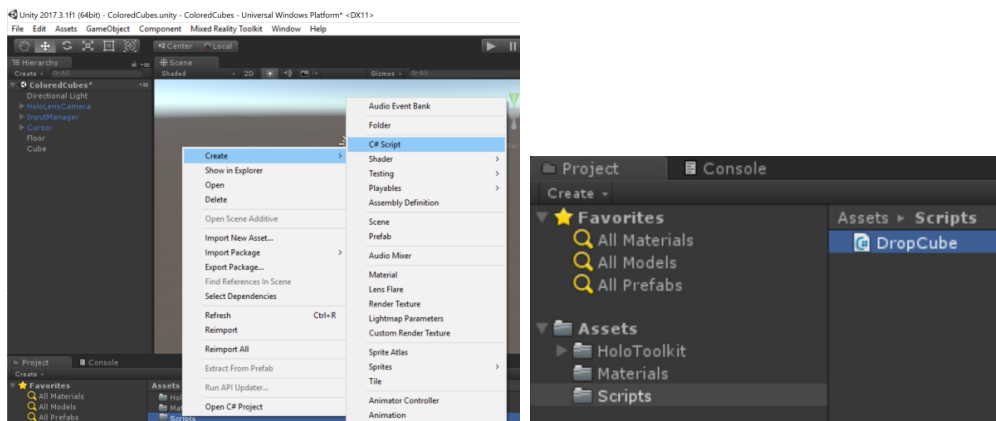
Drag & Drop the cubMat material from Project panel, on Element 0 field in Inspector.



In Project panel, create a new Folder, “Scripts”.



Create a new C# script into the Scripts folder and name it “DropCube”.



Double-click on the file to open it in VS.

Replace the code of your script with the following one:

```
using UnityEngine;
using HoloToolkit.Unity.InputModule;
public class DropCube : MonoBehaviour, IInputClickHandler
{
    // Called by GazeGestureManager when the user performs a Select
    gesture
```

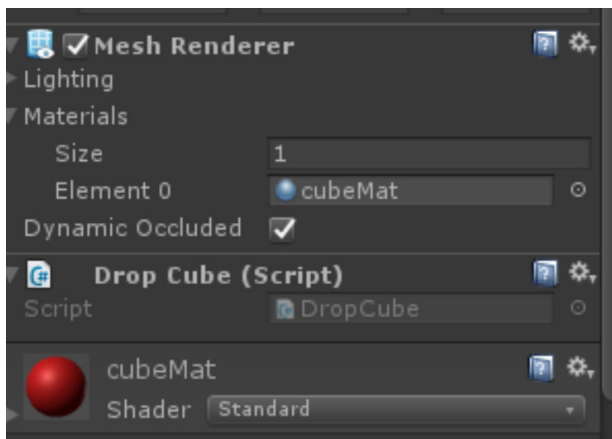
```

public void OnInputClicked(InputClickedEventData eventData)
{
    if (!this.GetComponent<Rigidbody>())
    {
        var rigidbody =
            this.gameObject.AddComponent<Rigidbody>();
        rigidbody.collisionDetectionMode =
            CollisionDetectionMode.Continuous;
    }
}
}

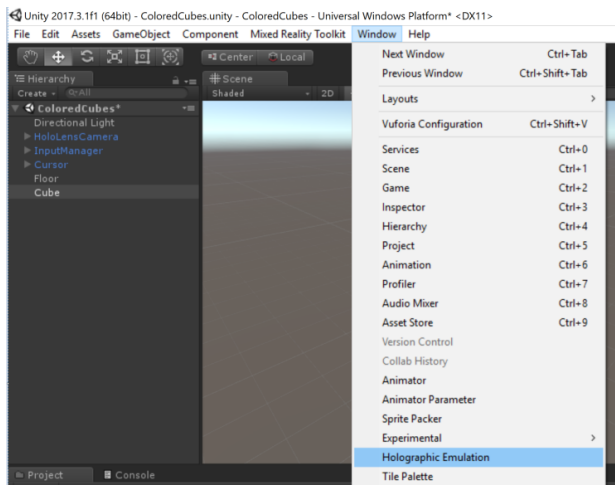
```

Save the file and come back to Unity.

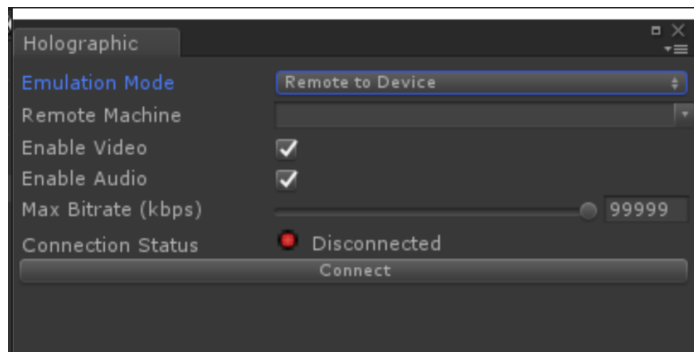
In Hierarchy, select the Cube GO and drag and drop the DropCube script on it.



In Menu, go to Windows => Holographic Emulation.



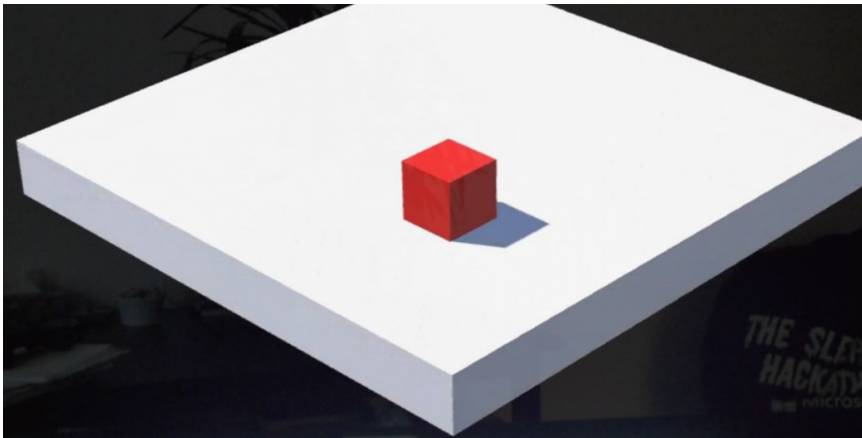
Chose for Emulation Mode, the property Remote to Device. Provide the IP address of the HoloLens and Connect.



On the HoloLens, you need to have the app Holographic Remoting App opened.

In Unity click on Play to run the Project on HoloLens.

Tap on the red cube and the Rigidbody component will be added to it and it will fall on the Plane.



Go Back to Script in Visual Studio and add a public property named createPrefab, and the code to create a new cube every time you Tap on the cube.

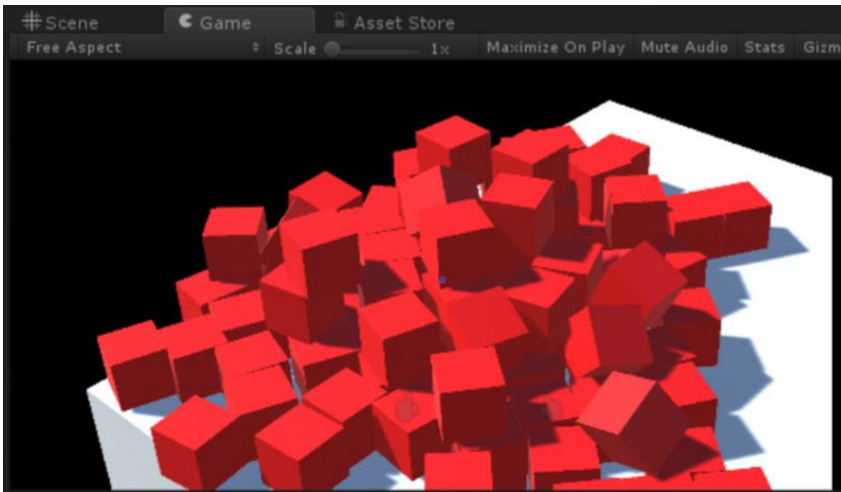
```
public GameObject createPrefab;
GameObject createdCubs = Instantiate(createPrefab,
    this.gameObject.transform.position,
    this.gameObject.transform.rotation) as GameObject;
```

Your .cs file will look like this:

```
using UnityEngine;
using HoloToolkit.Unity.InputModule;
public class DropCube : MonoBehaviour, IInputClickHandler
{
```

```
// Called by GazeGestureManager when the user performs a
Select gesture
public GameObject createPrefab;
public void OnInputClicked(InputClickedEventData
    eventData)
{
    if (!this.GetComponent<Rigidbody>())
    {
        var rigidbody =
            this.gameObject.AddComponent<Rigidbody>();
        rigidbody.collisionDetectionMode =
            CollisionDetectionMode.Continuous;
    }
    GameObject createdCubs = Instantiate(createPrefab,
        this.gameObject.transform.position,
        this.gameObject.transform.rotation) as GameObject;
}
}
```

3. Run the project on HoloLens



Let's add some color to the cubes.

Add a public array of colors and assign it a random color from the defined ones.

```
public Color[] cubColorArray;
```

Your .cs file will look like this:

```
using UnityEngine;
using HoloToolkit.Unity.InputModule;
public class DropCube : MonoBehaviour, IInputClickHandler
{
    // Called by GazeGestureManager when the user performs a
    Select gesture
```

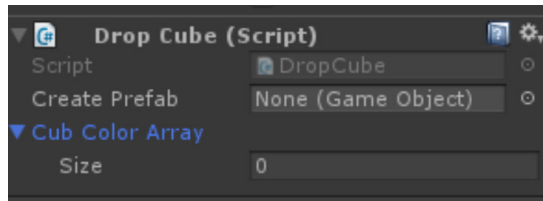
```

public GameObject createPrefab;
public Color[] cubColorArray;
public void OnInputClicked(InputClickedEventData eventData)
{
    if (!this.GetComponent<Rigidbody>())
    {
        var rigidbody =
            this.gameObject.AddComponent<Rigidbody>();
        rigidbody.collisionDetectionMode =
            CollisionDetectionMode.Continuous;
    }
    GameObject createdCubs = Instantiate(createPrefab,
        this.gameObject.transform.position,
        this.gameObject.transform.rotation) as GameObject;
    int randomInt = Random.Range(0, cubColorArray.Length);
    createdCubs.GetComponent<Renderer>().material.color =
        cubColorArray[randomInt];
}
}

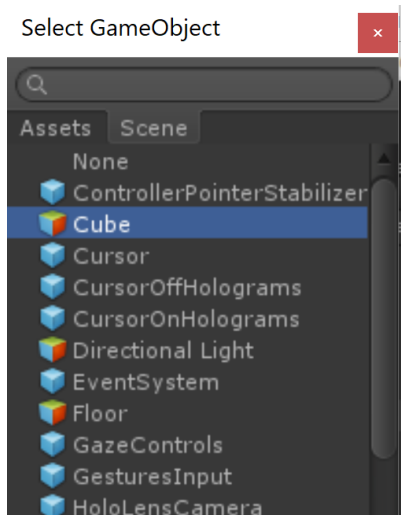
```

Save the file and come back to Unity.

Select the Cube and see in Inspector the public properties declared in the Script.

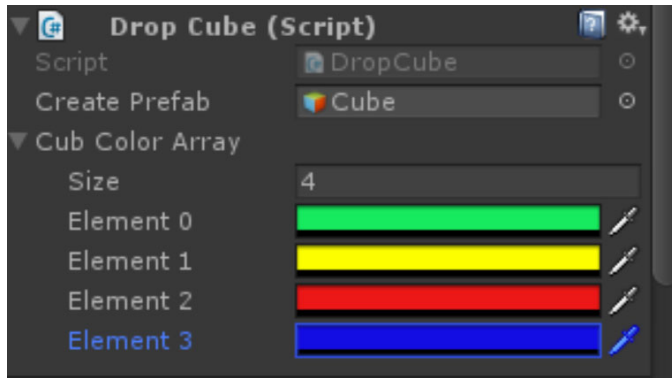


Click on the little circle which is on the right of the Create prefab field.

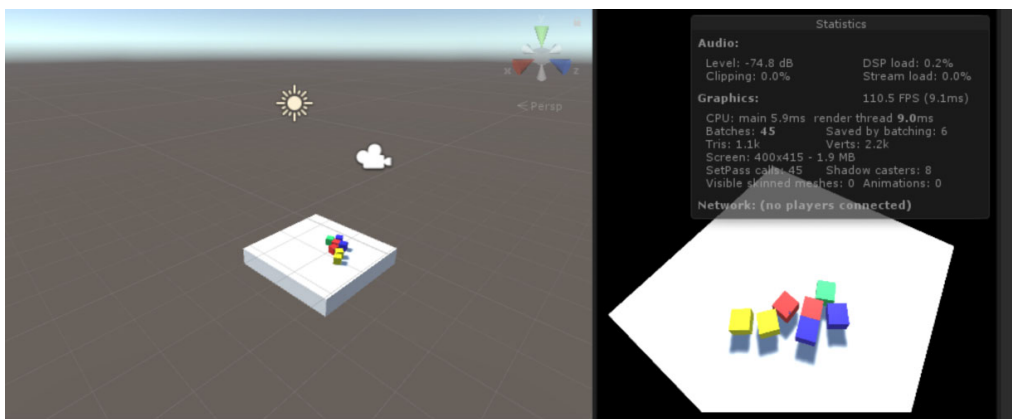
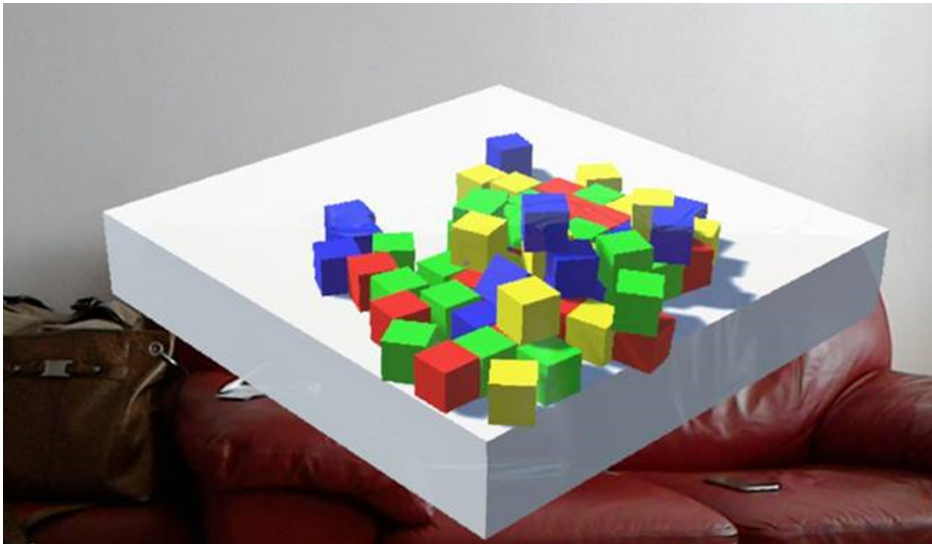


Select the Cube prefab.

Set the size of the array to 4, for example, and provide some colors



Test the project on the HoloLens:

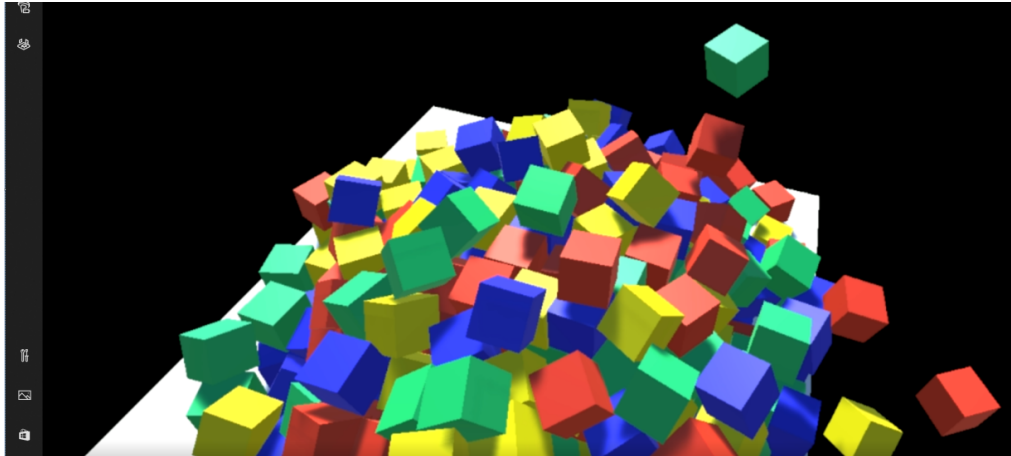


4. The project was successfully deployed and run in Virtual Reality environment.

Because the project is a Universal Windows platform Application, it can be deployed on any device running Windows 10, so it can be installed on a PC, as a Virtual Reality Application and run using an immersive headset from one of the Microsoft partners: Acer, HP, Asus, Dell, Lenovo, Samsung. For our tests, we used an Acer immersive headset device. We had the VR headset and the motion controllers.



We used the same package that was generated for HoloLens, to run it in Mixed Reality Portal. This is the result:



CONCLUSIONS

This article demonstrates how to create an Application for Augmented Reality, for HoloLens device, using Unity. It goes through all the steps needed to create the project, test it, generated the build for the specific platform and install it on the device.

BIBLIOGRAPHY

- [1] Mixed Reality Toolkit: <https://github.com/Microsoft/MixedRealityToolkit-Unity>
- [2] Design Labs: https://github.com/Microsoft/MixedRealityDesignLabs_Unity
- [3] Microsoft Academy: <https://docs.microsoft.com/en-us/windows/mixed-reality/academy>
- [4] Unity tutorials: <https://unity3d.com/learn/tutorials>