JOBS OF THE FUTURE

Augmented Reality with Unity

DIGIJEUNES

Jelgavas Valsts ģimnāzija

CITIZENS IN POWER

MEDUSA DIGITAL

VECTOR REALISM

AMAIS

01

WHAT IS AUGMENTED REALITY

In this tutorial we are going to walk you through making a smartphone application that will use the camera and produce augmented reality ballons to shoot at.
Augmented reality, in this case, means using a camera that processes images in real time whilst adding virtual elements to the picture, making the images "Augmented" when we look at the screen.

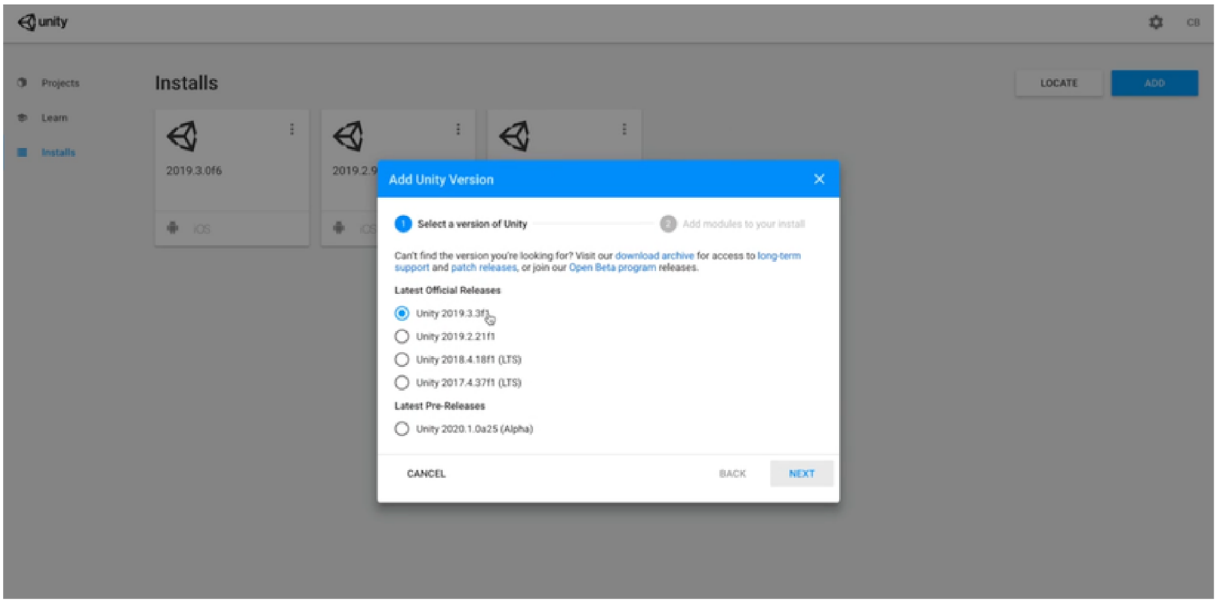
WHAT IS NEEDED FOR THIS TUTORIAL

To go through this tutorial you will need basically two things, a computer and a phone that is compatible with AR Camera and obviously some decent internet connexion.
Compatible phones : **https://developers.google.com/ar/devices**

On your computer you'll need to install Unity Hub and create a personal account.
https://store.unity.com/download
Once Unity hub is installed, open it and activate it. Then go into the Installs section and click on Add. Select the latest version and click next.
Now select Android support (you can as well select the IOS support if you want) and if it's proposed to you, add also the VScode editor as it will be needed, too.
Agree to the terms and conditions and allow installation to complete.

Download also the assets for the project here:
https://drive.google.com/file/d/105EO8Adgi0N1EuPRHQpF1OfsPN4Eewaj/view?usp=sharing

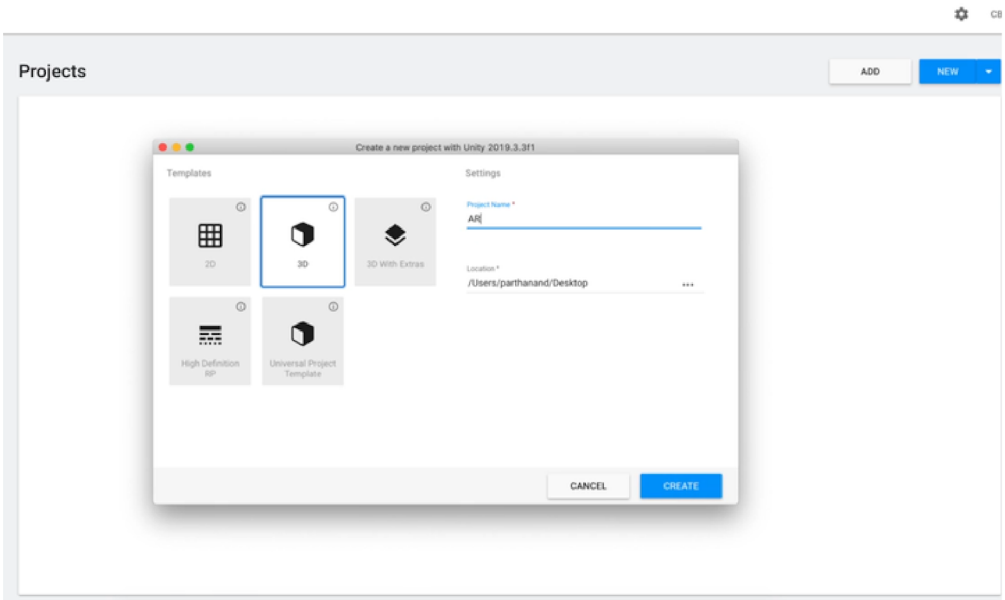
You'll have to download a few more extension but we're ready to start now!



02

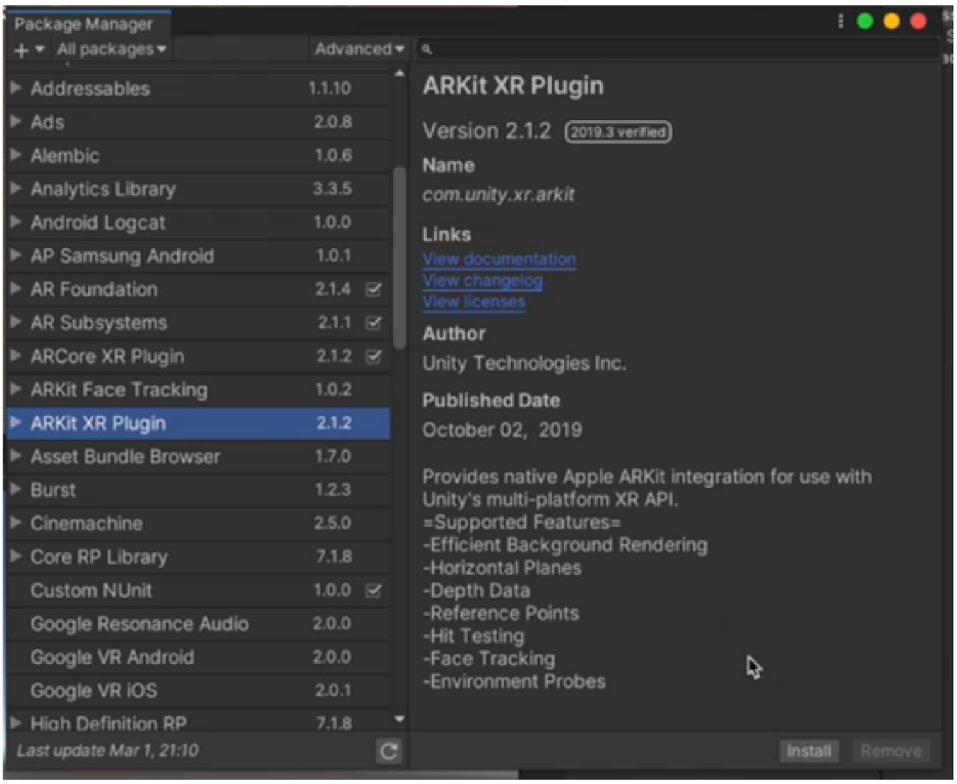
START A NEW PROJECT

Open Unity Hub in the project section, click on New. Select 3D as template, name your project and click on Create.



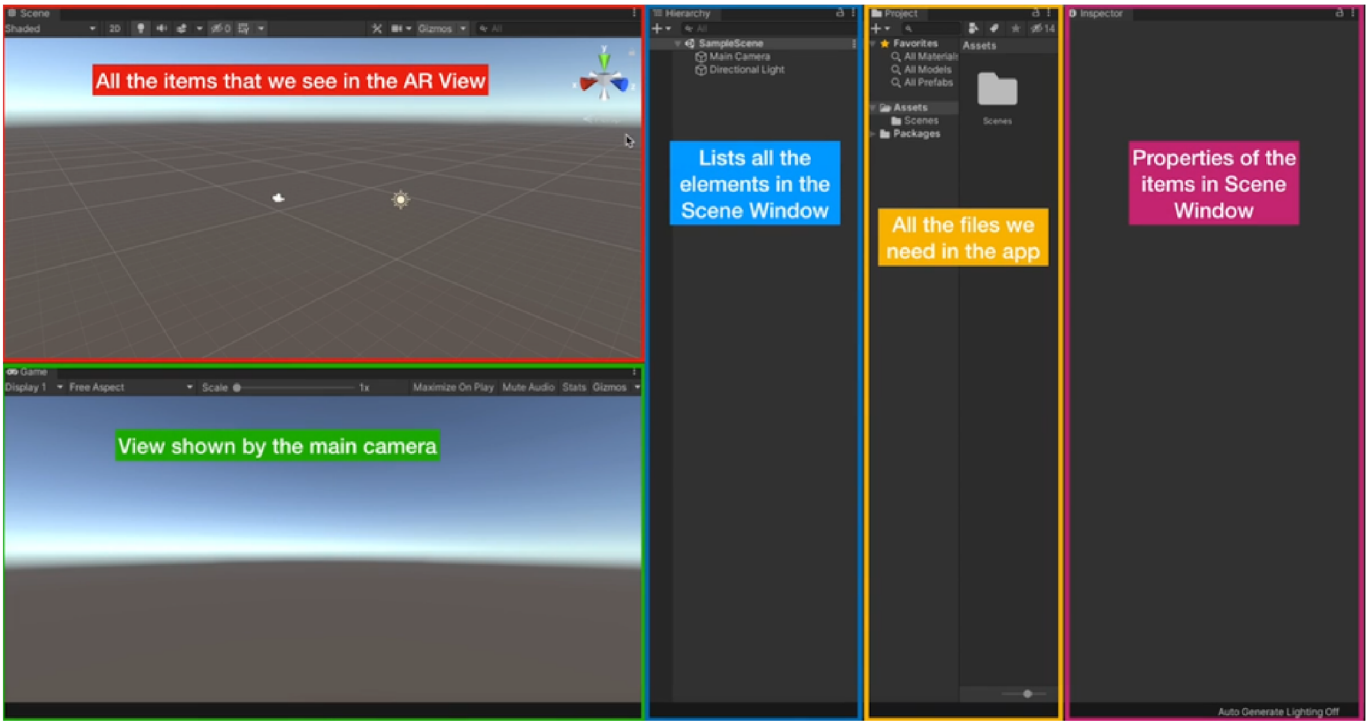
Unity will launch. To get started we need to install some extensions to handle augmented reality.
The first one will be **AR Foundation**.

To do so, click on **windows**, then **package manager**. Search for AR Foundation and click Install.
Then you'll need **ARCore XR Plugin for Android** and/or **ARKit XR Plugin for IOS**.
Search for them and install them.



03

UNITY LAYOUT



For the sake of clarity, if you don't have the same layout as shown here, go to **window** => **Layouts** and select **2 by 3**.

You can see five different section. **Scene**, **Game**, **Hierarchy**, **Project** and **Inspector**. We will refer to this section with these names during the tutorial.

Your Unity engine is now ready for development.

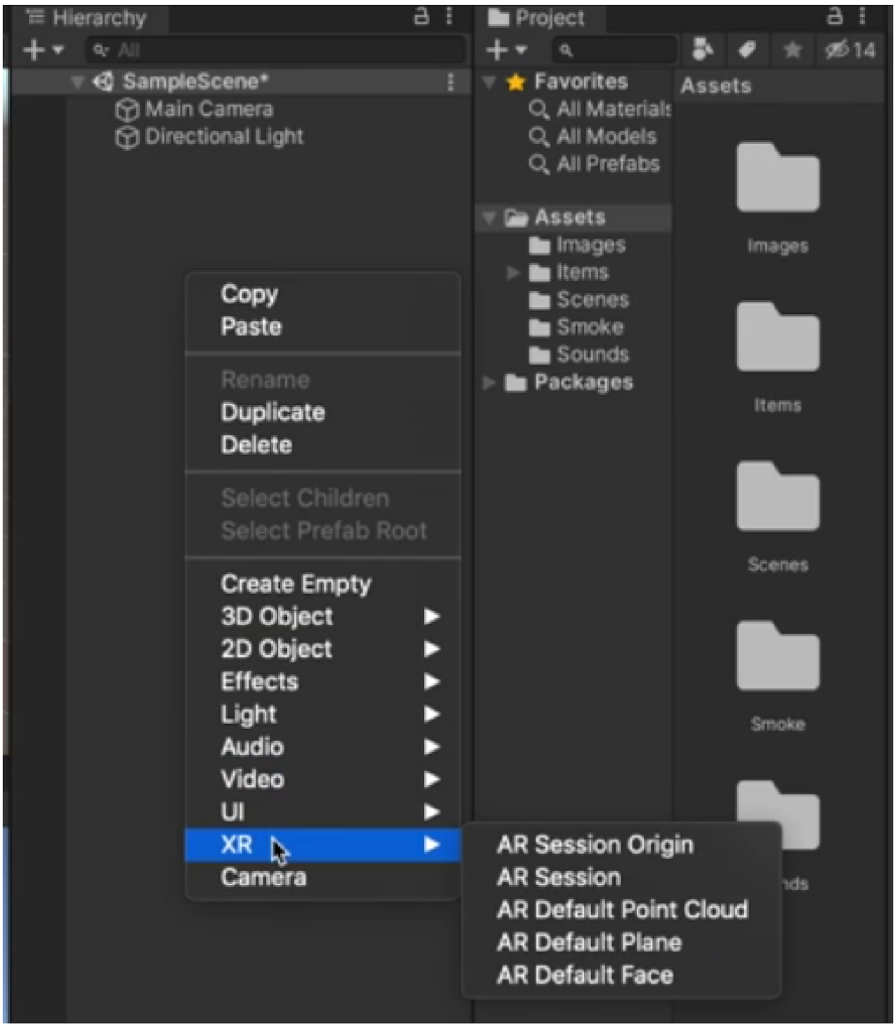
04

APP OVERVIEW

1 - New balloons appears every X seconds
2 - The new balloons start flying
3 - When the Shoot button is pressed, the balloon pops with a sound and smoke effect

We need to import the assets of the project one by one. Open the asset folder and start by sliding the **Balloon.unitypackage** into the **Project** section of Unity.
Do the same for the **Images** folder, **Sounds** folder and the **Smoke package**.

To add the AR camera you'll need to right click on the hierarchy, XR and click on AR Session.
Do the same and add AR Session Origin.
Still in hierarchy, you don't need the Main Camera, right click on it and click Delete.



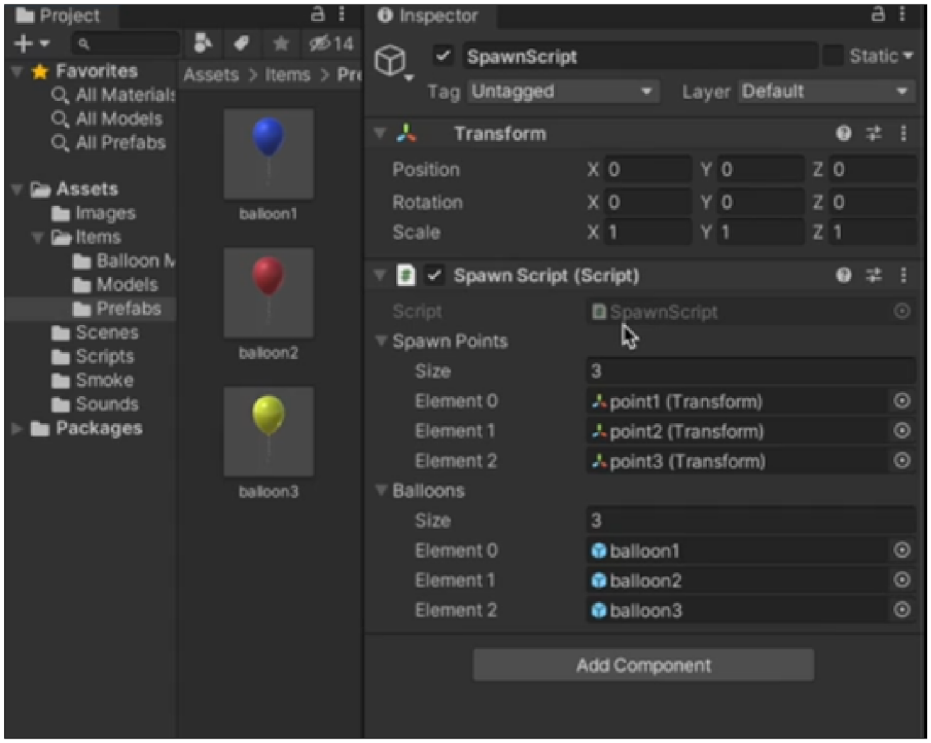
05

MAKING BALLOONS SPAWN

Inside the assets folder in the Project Section, right click and **create a new folder called Scripts**. Open it and right click inside to **create a C# script**, name it **SpawnScript**.
Double click on it, it'll open VS Code editor.

We need to add two new arrays at line 8.
public Transform[] spawnPoints;
public GameObject[] balloons;

Save it and go back to Unity. Inside hierarchy, right click and choose **Create Empty**. Rename it **SpawnScript** and keep it selected.
Drag the SpawnScript code into the inspector section to add component.
You'll now find the two new arrays that you created just before because they are public.



Change the size of both array to 3.
To create the spawn points of the balloons, right click in hierarchy and create empty and name it point1.
Do the same two more times with names point2 and point3.

Inside your hierarchy, click on SpawnScript and drag your points one by one into your array.

06

Because all the three points are at the same position we need to make them appear at different places.
Click on each point and modify the Z axis by 1.5 for each point, then change also the X axis for point2 at 0.5 and point3 at -0.5.

The spawning points are ready but we need to add them the models that will be used, in this case we need to add the balloons.
Open Items folder in your assets and go to Prefabs, you should see your balloons.
Select SpawnScript in hierarchy and slide the balloons here.

SPAWN CODE

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

public class SpawnScript : MonoBehaviour
{

    public Transform[] spawnPoints;
    public GameObject[] balloons;

    // Start is called before the first frame update
    void Start()
    {
        StartCoroutine(StartSpawning());

    }


    IEnumerator StartSpawning() {

        yield return new WaitForSeconds(4);

        for (int i = 0; i < 3; i++) {

            Instantiate(balloons[i], spawnPoints[i].position, Quaternion.identity);
        }

        StartCoroutine(StartSpawning());
    }
}
```

Update the code of SpawnScript with this content.

It spawns the three balloons at the points we prepared and will wait four seconds before spawning new balloons.

Save your script, go back to Unity and click on the play button to see the result.

07

MAKE THE BALLOONS FLY

We need a new script for this, so create a new one named BalloonScript and type the following code:

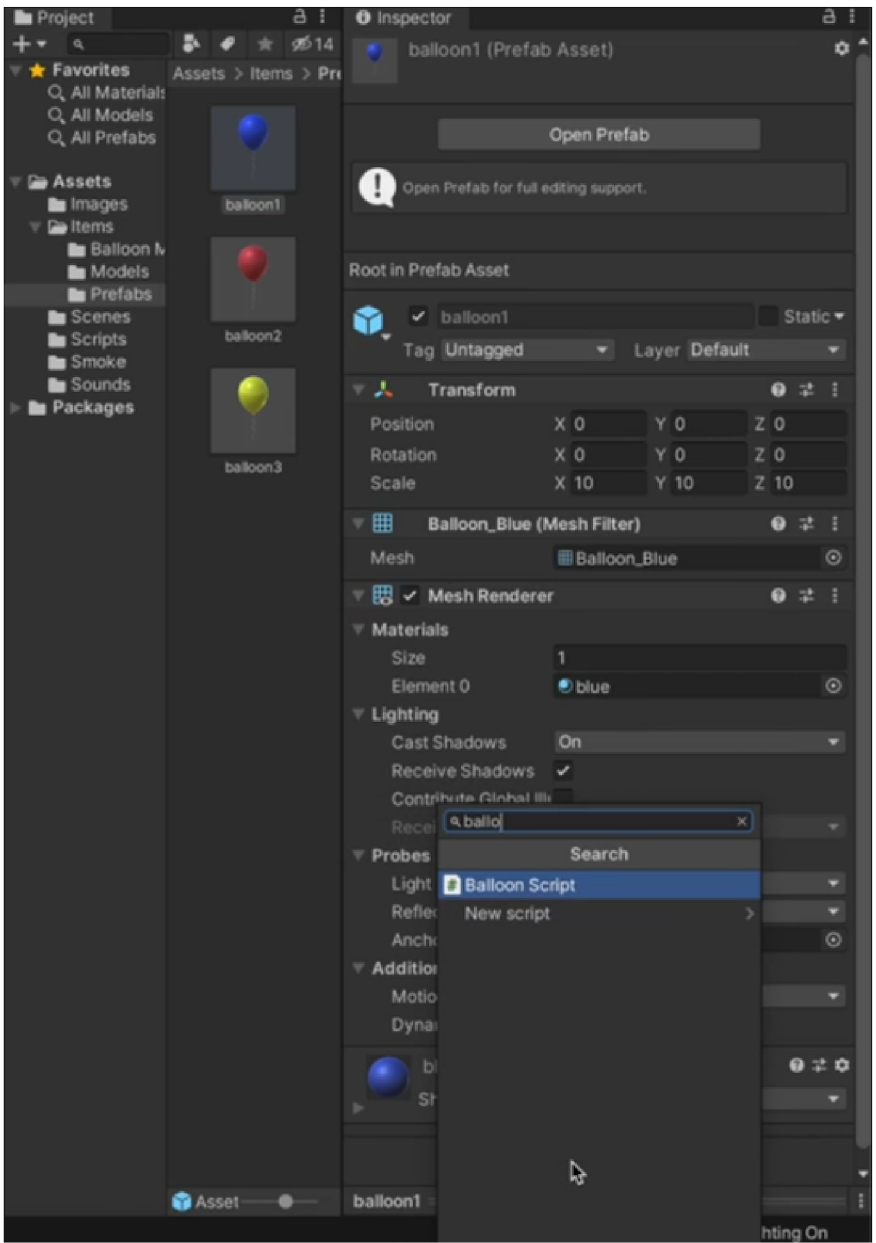
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BalloonScript : MonoBehaviour
{


    // Update is called once per frame
    void Update()
    {
        transform.Translate(Vector3.up * Time.deltaTime * 0.2f);
    }
}

```

This will make your balloon move up by very small steps.

You now need to add this script to each model. Go to your assets Items, prefabs and click on the first balloon then in the inspector section click Add component and search your BalloonScript.

Do the same for the two others.

Click on play and watch your balloons fly.



08

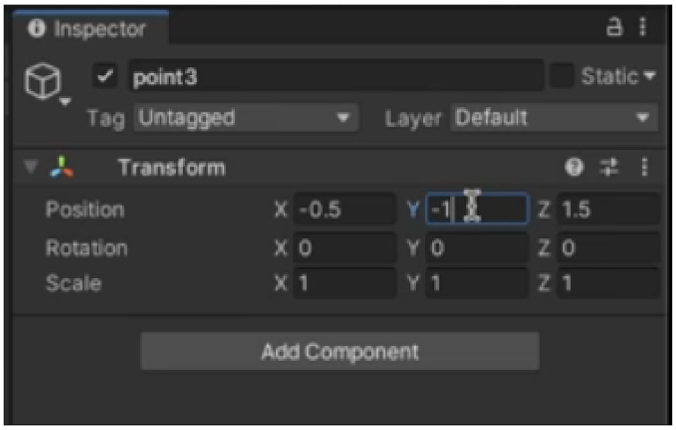
SHOOTING THEM DOWN

We need a final script for all of this to works: The Shooting script.
Add a new script named ShootScript and type it the following code:

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

public class ShootScript : MonoBehaviour
{

    public GameObject arCamera;
    public GameObject smoke;


    public void Shoot() {

        RaycastHit hit;

        if (Physics.Raycast(arCamera.transform.position, arCamera.transform.forward, out hit)) {

            if (hit.transform.name == "balloon1(Clone)" || hit.transform.name == "balloon2(Clone)" || hit.transform.name == "balloon3(Clone)") {

                Destroy(hit.transform.gameObject);

                Instantiate(smoke, hit.point, Quaternion.LookRotation(hit.normal));
            }
        }
    }
}
```

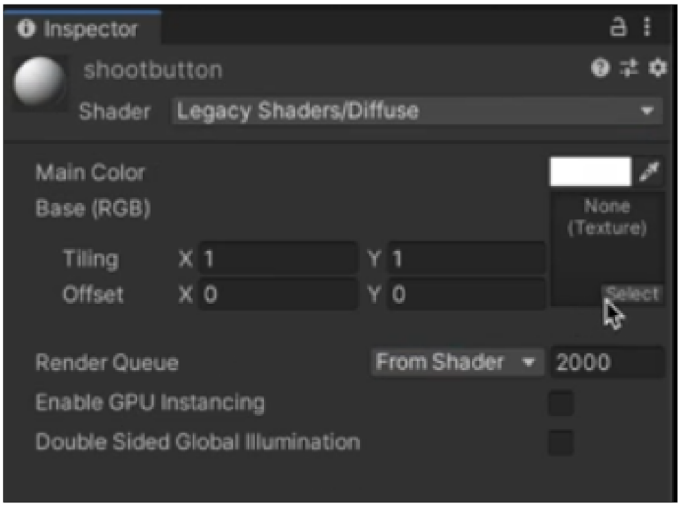
This script checks if when you click Shoot the camera is centered on a balloon and destroys the balloon if that's the case. It'll also bring the smoke effect when they disappear.

We need to make a shoot button to connect with this Shoot() function. Save your script and go back to Unity.
Right click in your hierarchy, UI and select canvas. In the inspector, change UI Scale Mode to Scale with screen size.



Right click on your canvas, UI then button.
Delete the text of the button in the button's dropdown.

Click on the button and change its size by 100 to 100, and its position by 300 and -120.



Go to your assets and open images. Create a new folder named Materials. Inside it create a new Material that you name ShootButton.
In the inspector change the shader by selecting Legacy Shaders then Diffuse.

Click on select and choose the shoot image.
Last click on Shaders again, UI and default.

Now create a new material for the crossbar and do the same but add the cross picture instead.

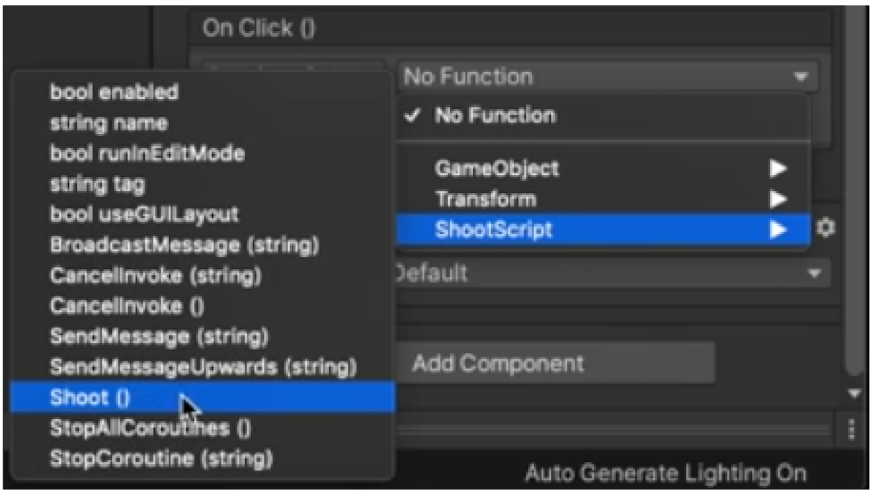
Now that the materials are ready, click on your button, change the Source Image in the inspector by none. Then click on Material and put your shoot button.

To add the cross, right click on canvas, UI, image and change the material by your crossbar.

LINKING THE SCRIPT AND THE BUTTONS

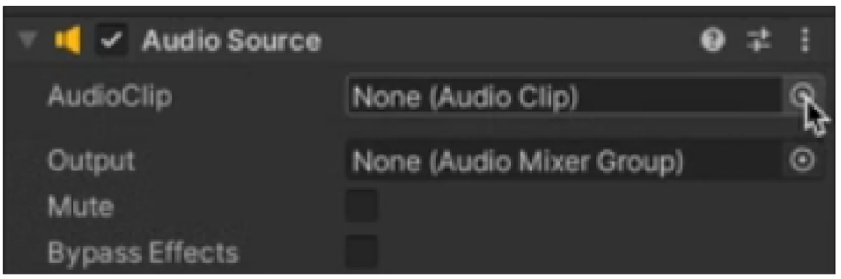
Inside your hierarchy, right click, create empty and name it ShootScript. Find your shoot script and drag it to the inspector like the previous time.
We now need to link the two public arrays.
In hierarchy open AR Session Origin and drag AR Camera into the inspector variable. Do the same for the smoke in your assets.

Now click on your button in the hierarchy and find the OnClick section. Click on add and drag the ShootScript from the hierarchy in it and then change No Function by ShootScript > Shoot()



Everything is linked, we need now just two last things, a box collider and a pop sound.
Go to your balloons in your assets and in the inspector add component, Box Collider.

For the pop sound, go to your smoke asset, add component, audio source. Then click audio clip and add you balloon pop.



10

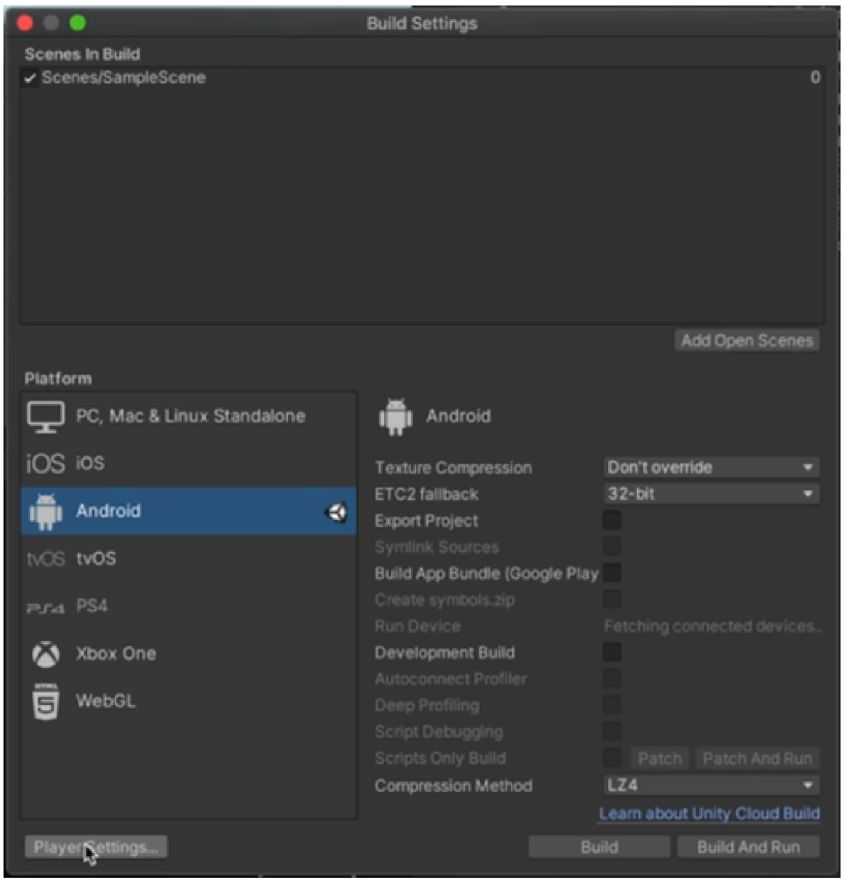
BUILDING THE APP

Click on Files, Build Settings.

In the new windows, click Add Open Scenes. Select Android and click on Switch Platform.
Click on player settings and remove Vulkan API then uncheck Portrait and Multithreading Rendering.

You can now click on Build, name your application and click save.

You'll now have an apk file that you can install on your Android.
The same you can do for IOS and your apps are ready!



JOBS OF THE FUTURE

Augmented reality has already made its way into the world of tech and has several fields of application. You'll find augmented reality in many camera applications such as snapchat, instagram or messenger that will modify your face in real time but also more advanced ones such as the google glass project which consists in a pair of glasses that display information such as directions, weather, data and time. Pokemon Go is yet another successful application of AR in the game industry. In the medical field, AR is a promising technology especialy in the domain of surgery where surgeons will be able to visualise indications regarding the operations to perform on a patient. To conclude, AR is currently being applied in the military field, where soldiers can for example view information regarding the enemy's position, targets or simply data about the battleground in real time via tools similar to the google glasses.
AR is already an emerging field which promises to open many new job opportunities all around the globe.

Jobs of the Future is an international cooperation co-financed by the Erasmus + programme of the European Union. It aims to create synergies between enterprises active in the tech sector, youth organizations and educational institutions. The objective is to empower young people to pursue their own professional and educational goals in the tech field.

Contact

Jobs of the Future
www.jobsofthefuture.eu
info@digijeunes.com

Co-funded by the Erasmus+ Programme of the European Union