Unity 3D - ARkit Zombie Tutorial

by Matthew Hallberg
https://youtu.be/S7kKQZuOdlk

We are going to create a AR Zombie app for iPhone or iPad.

Apple ARkit uses SLAM (Simultaneous Localization and Mapping) and sensor fusion in order to place 3D objects on the ground and other surfaces.

Installation of tools

- Install the newest patch version of Unity 5.6.1p1

http://beta.unity3d.com/download/74c1f4917542/Uni...

- Unity ArKit Plugin

https://oc.unity3d.com/index.php/s/3hfM9T05P9vOpCf
The name of the file is *unity-arkit-plugin.unitypackage*

- Install Xcode 9 beta onMac

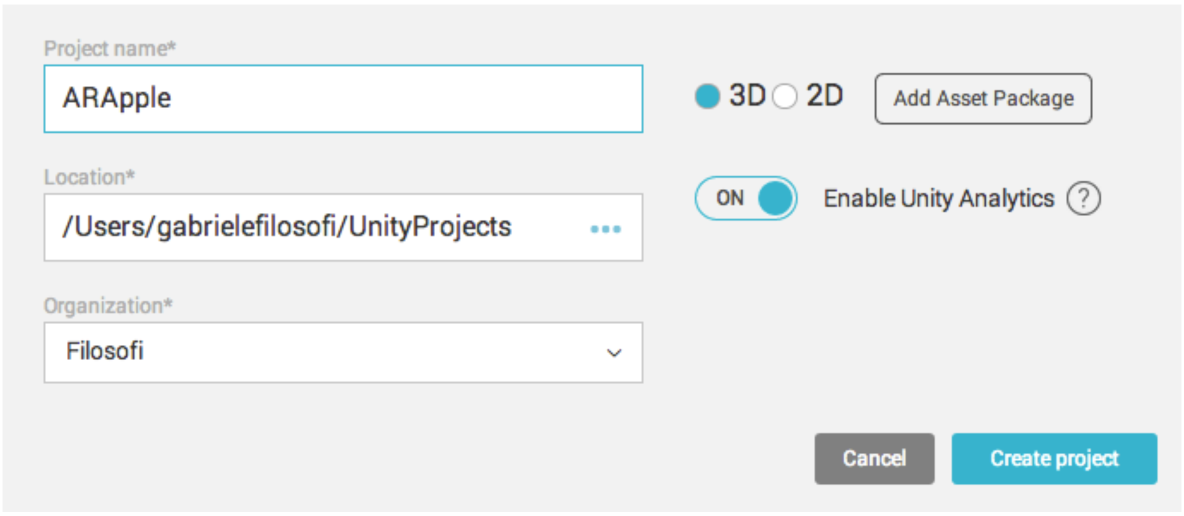
https://developer.apple.com/xcode/

- Install IOS 11 on iPhone:

http://www.redmondpie.com/ios-11-beta-download-fe...
to restore iPhone from backups:
https://beta.apple.com/sp/betaprogram/restore#ios
to unenroll from the Bata Software Program:
https://beta.apple.com/sp/betaprogram/unenroll

Create the project

- Create a new Unity 3D project



Project name*: ARApple

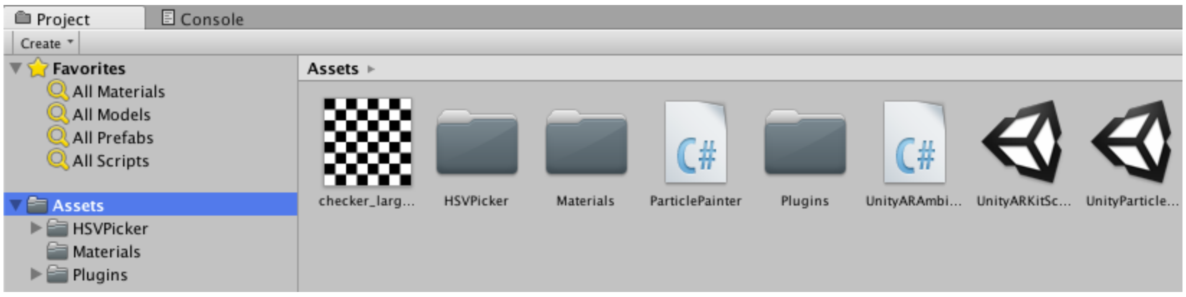
[✓ 3D] 2D

Location*: /Users/gabrielefilosofi/UnityProjects

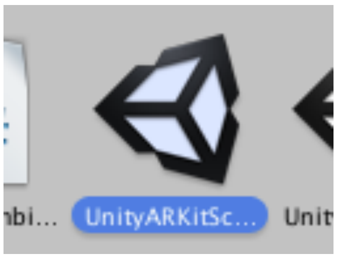
ON Enable Unity Analytics

Organization*: Filosofi

- Under File > Build Settings, select iOS, then click Switch Platform button
- Drag the *unity-arkit-plugin.unitypackage* plugin into the Assets folder, then click Import button

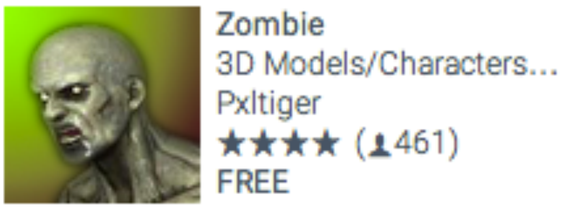


Double click the UnityARKitScene

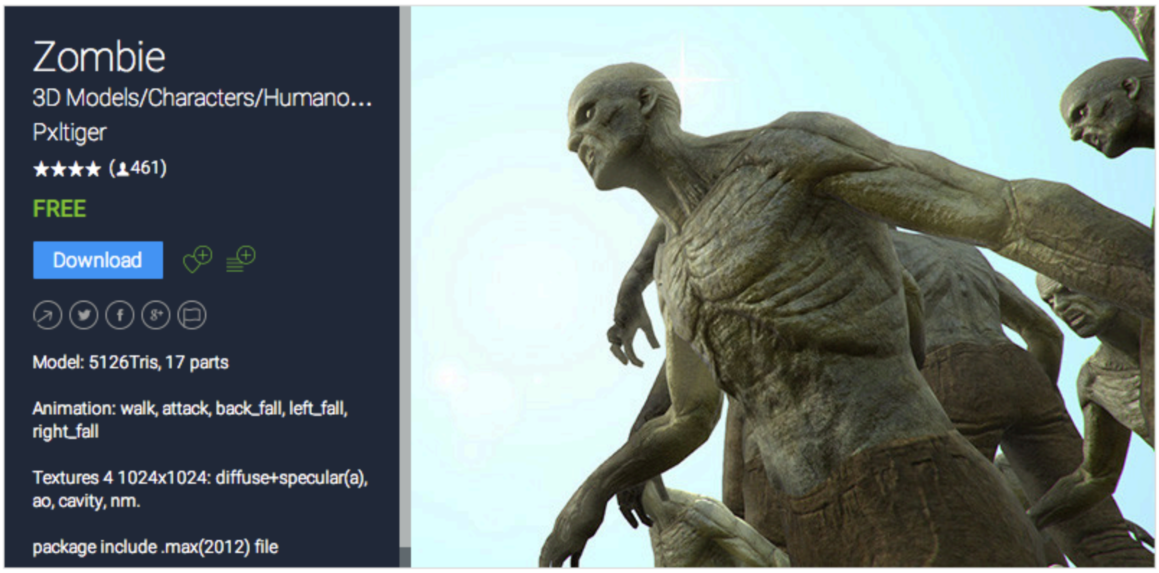


Add the Zombie model

In the Asset Store search and download the Zombie 3D model (free)



Download and import it



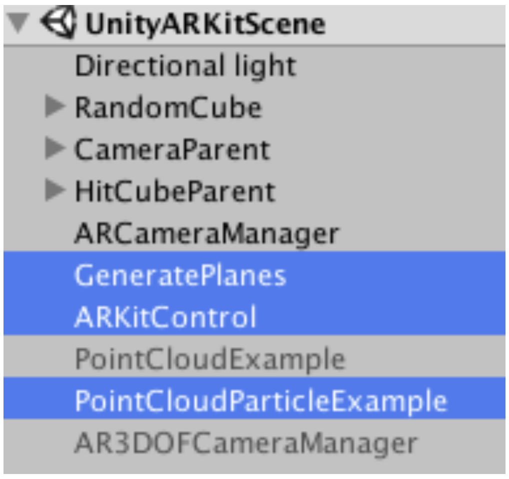
Model: 5126Tris, 17 parts

Animation: walk, attack, back_fall, left_fall, right_fall

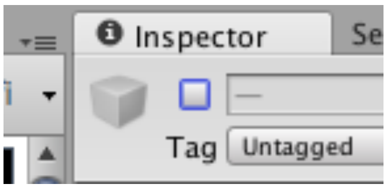
Textures 4 1024x1024: diffuse+specular(a), ao, cavity, nm.

package include .max(2012) file

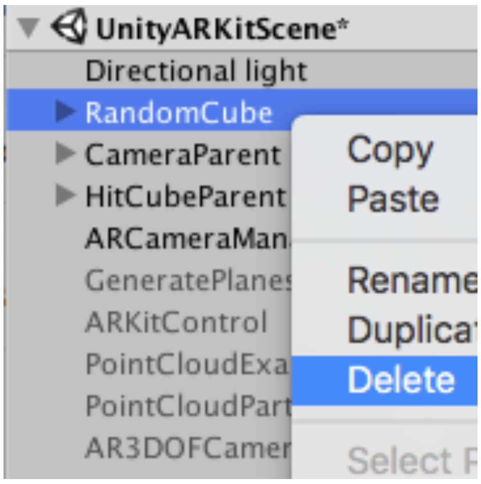
In the Hierarchy window select the following items,



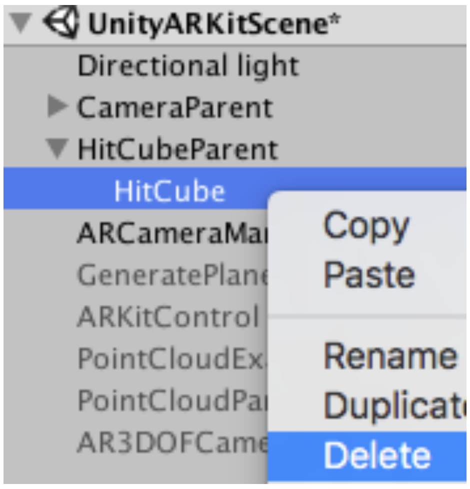
got to Inspector window and disable them



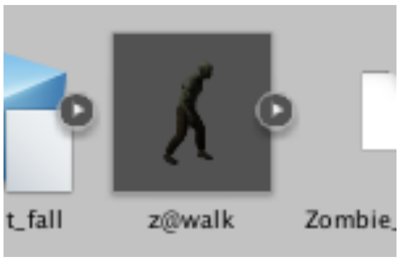
- Now delete the RandomCube game object



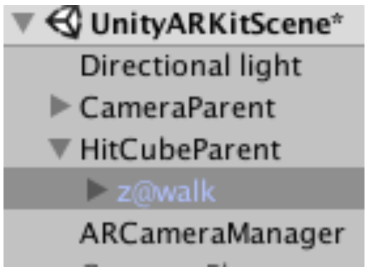
and delete the HitCube child of HitCubeParent



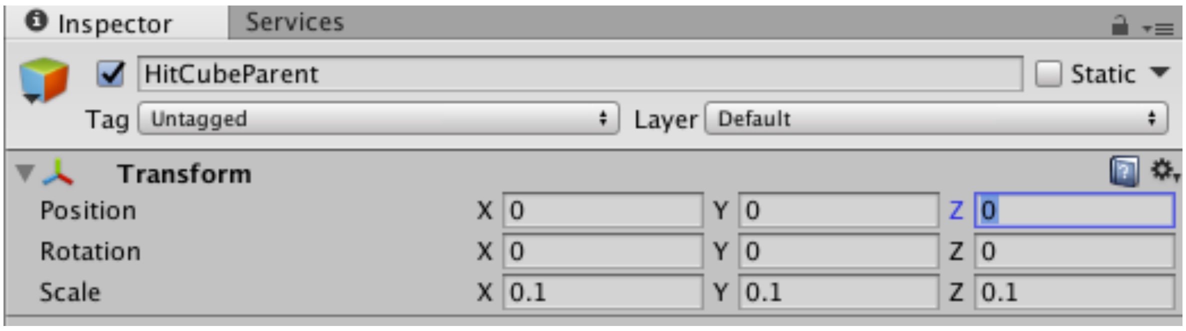
- From the zombie Model folder, drag the walk model called z@walk



on top of the HitCubeParent game object, in order to make it a child



- In the HitCubeParent inspector, ensure z position is zeroed



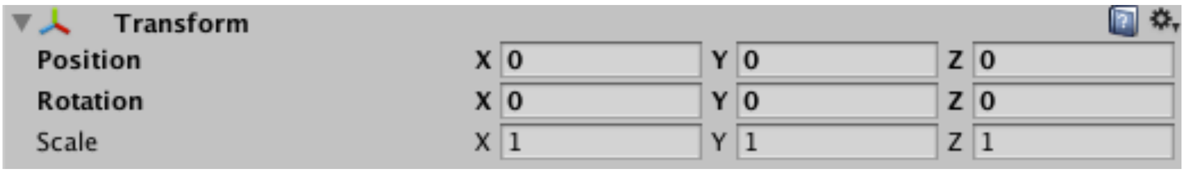
HitCubeParent

Tag Untagged   Layer Default

Transform

Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	0.1	Y	0.1	Z	0.1

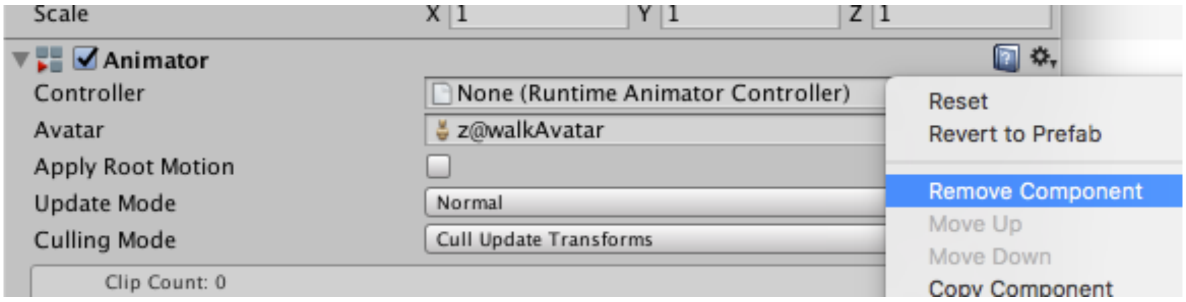
- Select zombie object and set it's x, y, and z scale to 1



Transform

Position	X	0	Y	0	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

- In the zombie walk, remove the Animator component by clicking the cog wheel



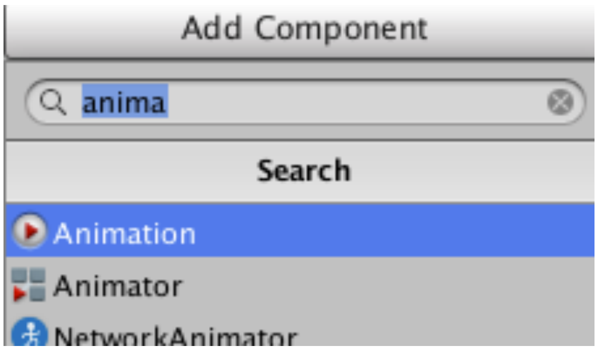
Scale X 1 Y 1 Z 1

Animator

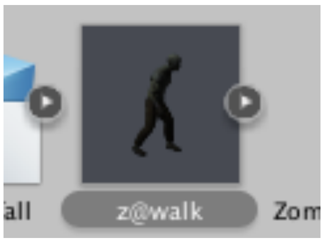
Controller	None (Runtime Animator Controller)
Avatar	z@walkAvatar
Apply Root Motion	☐
Update Mode	Normal
Culling Mode	Cull Update Transforms

Clip Count: 0

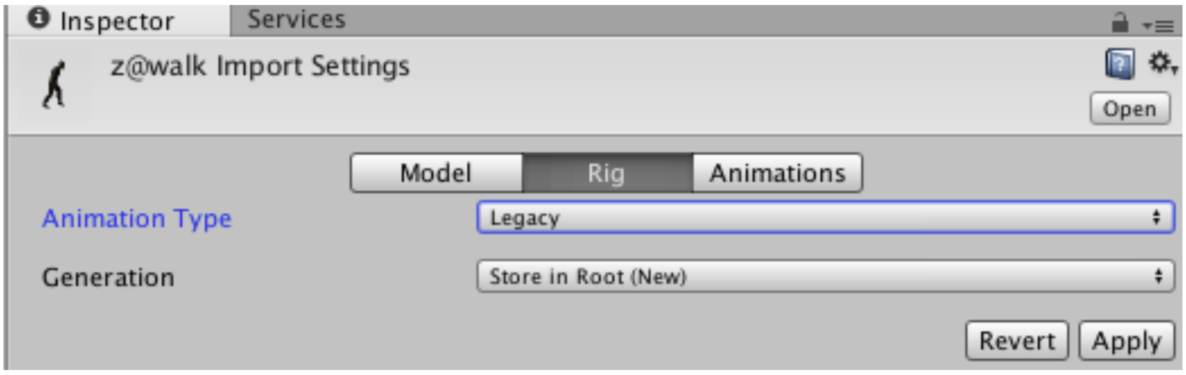
and add an Animation component



- Select the zombie object in the assets folder

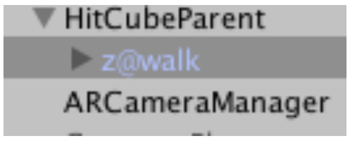


set the "Rig" type tab to Legacy, then press Apply

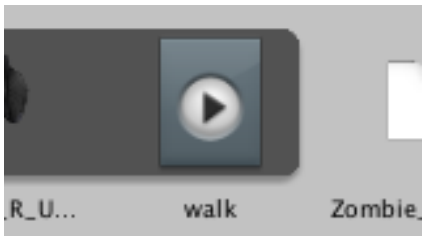


Animation Type: Legacy
Generation: Store in Root (New)

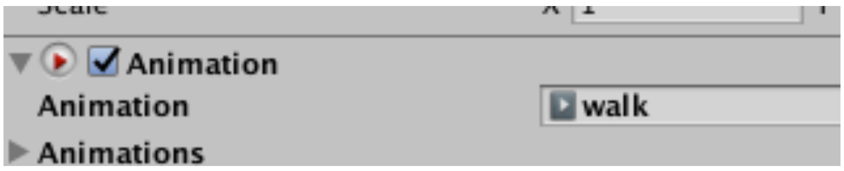
- Select the zombie game object in the hierarchy window



then from the assets folder drag the walk animation of the zombie model

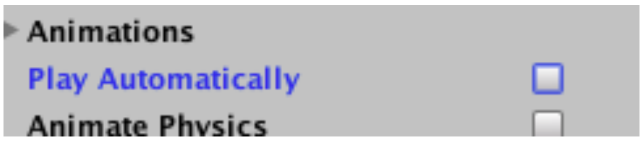


into the Animation's empty slot of the Animation component

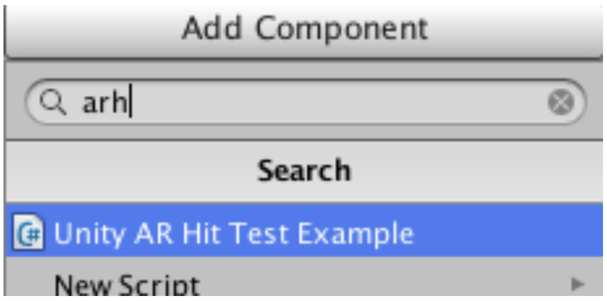


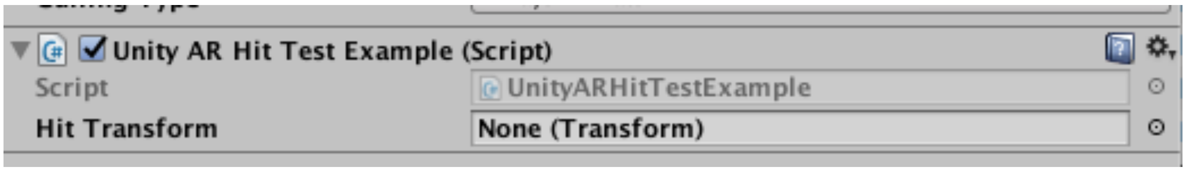
Animation: walk

- Uncheck "Play Automatically"

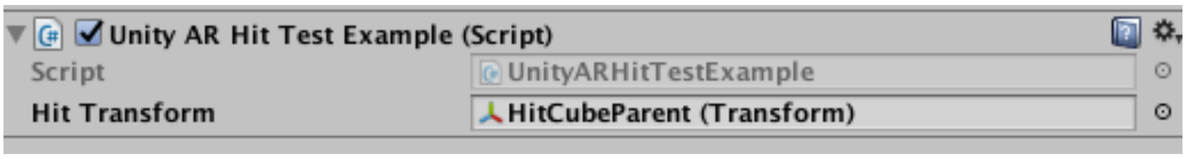

Animations
Play Automatically
Animate Physics

- Select the zombie game object in the hierarchy window, then add a *AR Hit Test Example* C# script component



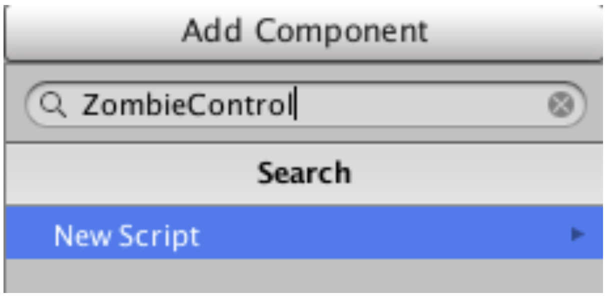

Unity AR Hit Test Example (Script)
Script UnityARHitTestExample
Hit Transform None (Transform)

Drag the HitCubeParent object in the *Hit Transform* empty slot of the component just created to create the reference

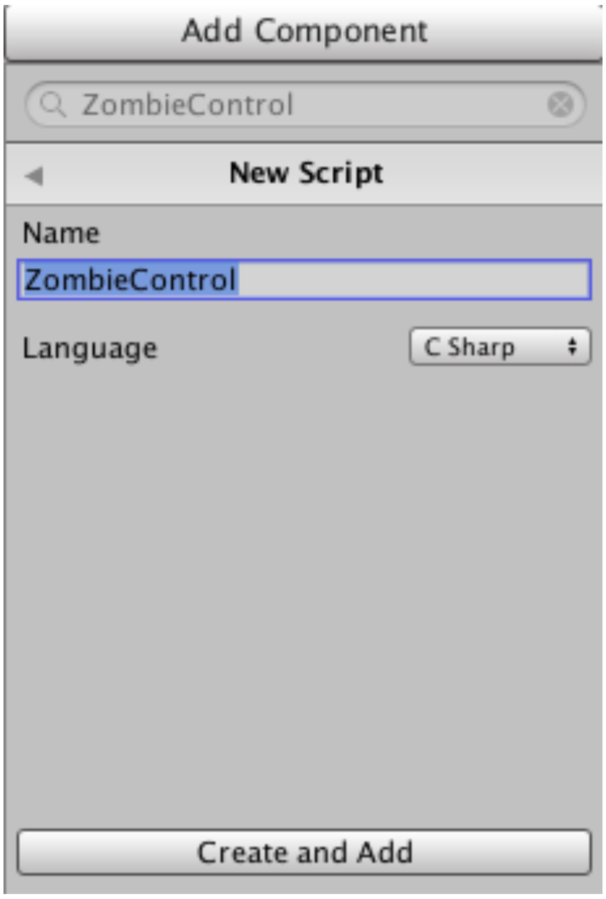

Unity AR Hit Test Example (Script)
Script UnityARHitTestExample
Hit Transform HitCubeParent (Transform)

Add functionality using C# scripts

- Add a new C# script component named "ZombieControl"

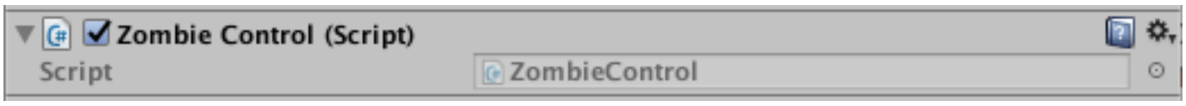


then press *Create and Add* button



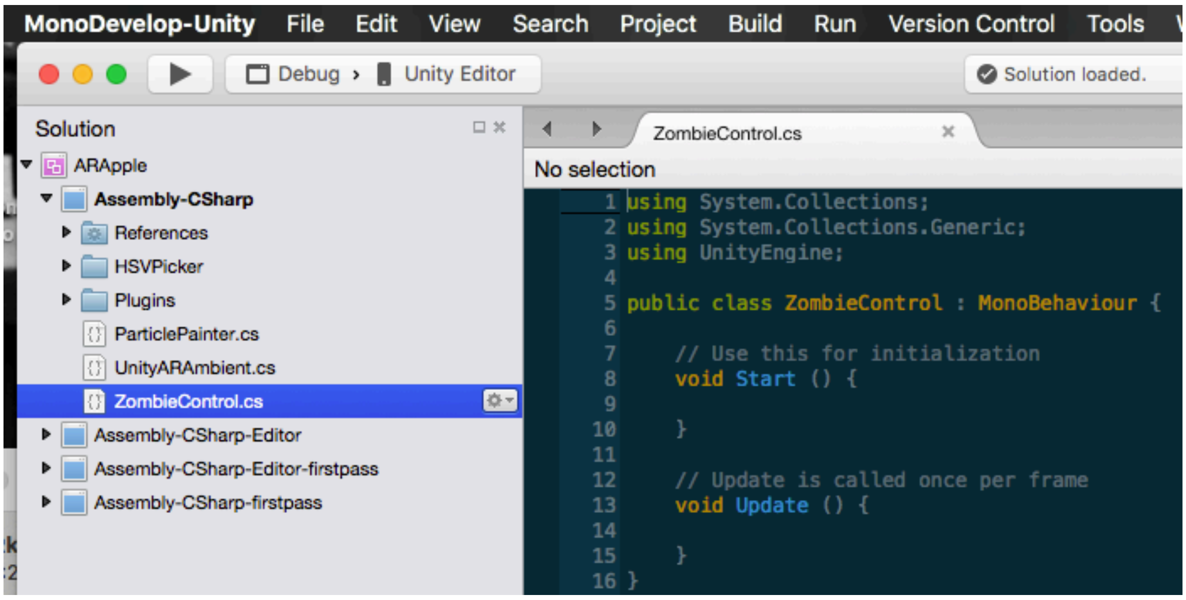
Name: ZombieControl

Language: C Sharp



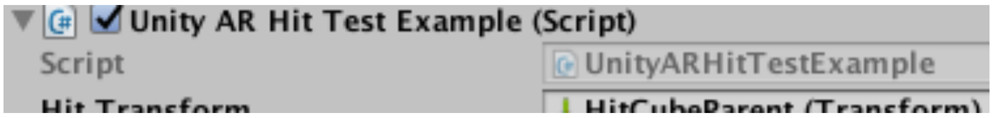
Script: ZombieControl

- Double click the *ZombieControl* script placeholder to open up the script into MonoDelvelop Unity editor



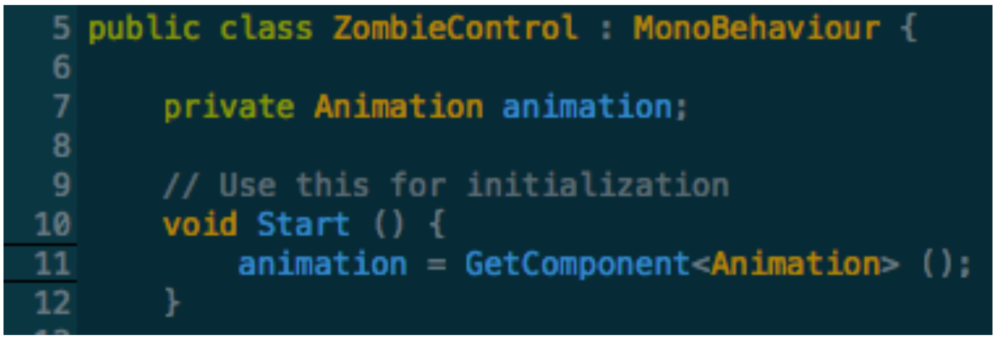
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ZombieControl : MonoBehaviour {

	// Use this for initialization
	void Start () {

	}

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

	}
}
```

Do the same for the *UnityARHitTestExample* script (by double click)



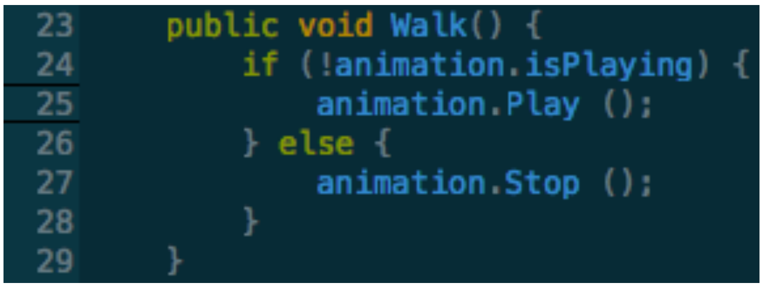
```
Unity AR Hit Test Example (Script)
Script            UnityARHitTestExample
Hit Transform     HitCubeParent (Transform)
```

- Now we have to add some functionality for the zombie. In the ZombieControl class let’s add a reference to the Animation component



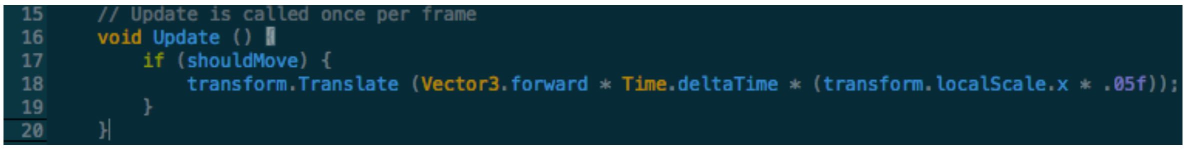
```csharp
public class ZombieControl : MonoBehaviour {

    private Animation animation;

    // Use this for initialization
    void Start () {
        animation = GetComponent<Animation> ();
    }
```

Now add a class method to animate and stop the zombie when executed through UI button



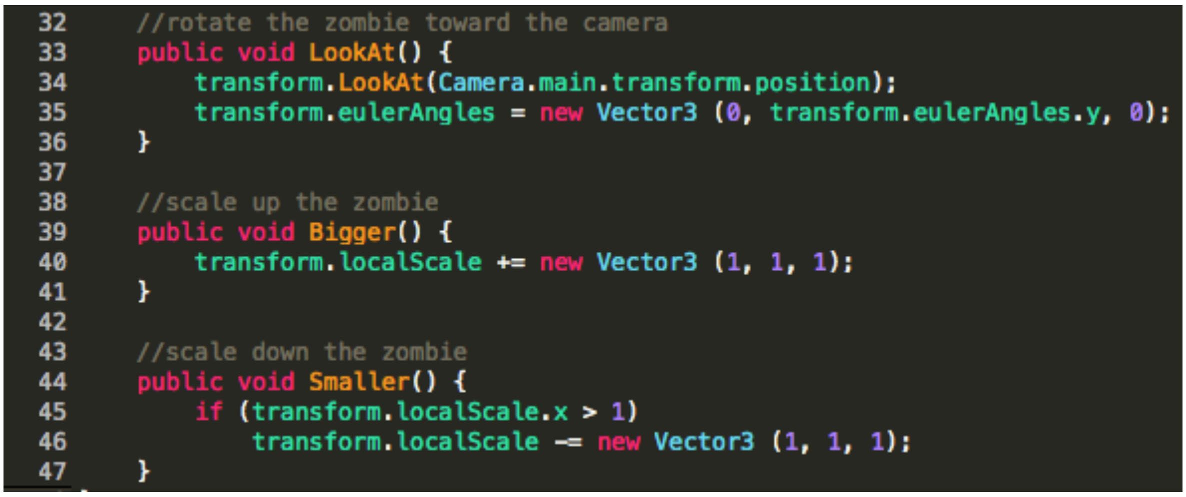
```csharp
    public void Walk() {
        if (!animation.isPlaying) {
            animation.Play ();
        } else {
            animation.Stop ();
        }
    }
```

Let’s add a flag attribute shouldMove, and in the Update function add the following code to move the zombie forward of a space proportional to the zombie scale



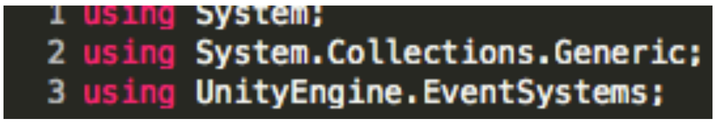
```csharp
    // Update is called once per frame
    void Update () {
        if (shouldMove) {
            transform.Translate (Vector3.forward * Time.deltaTime * (transform.localScale.x * .05f));
        }
    }
```

- Now let’s add methods to make the zombie turn and look at the camera, and to scale the zombie’s size up or down



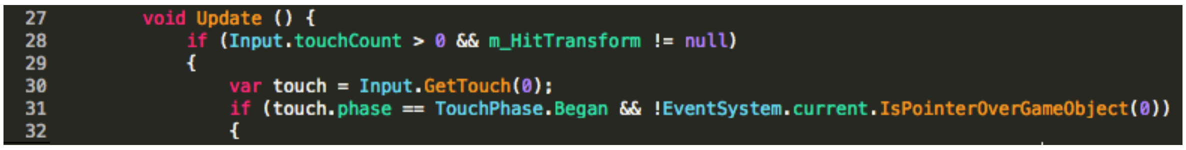
```csharp
    //rotate the zombie toward the camera
    public void LookAt() {
        transform.LookAt(Camera.main.transform.position);
        transform.eulerAngles = new Vector3 (0, transform.eulerAngles.y, 0);
    }

    //scale up the zombie
    public void Bigger() {
        transform.localScale += new Vector3 (1, 1, 1);
    }

    //scale down the zombie
    public void Smaller() {
        if (transform.localScale.x > 1)
            transform.localScale -= new Vector3 (1, 1, 1);
    }
```

- Go to the *UnityARHitTestExample.cs* C# script and fix a problem related to the touch screen input. Everytime you touch the screen the screen point touch is transformed to a world point coordinate and this makes the zombie to move there. We want to prevent this behaviour when we tap any game objects, like the control buttons.
- To do that import a touch library



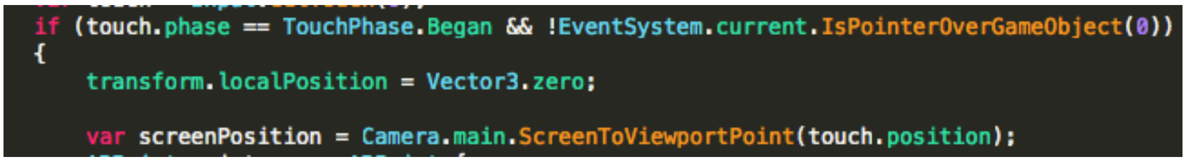
```
using System;
using System.Collections.Generic;
using UnityEngine.EventSystems;
```

and change the Update function this way



```
        void Update () {
            if (Input.touchCount > 0 && m_HitTransform != null)
            {
                var touch = Input.GetTouch(0);
                if (touch.phase == TouchPhase.Began && !EventSystem.current.IsPointerOverGameObject(0))
                {
```

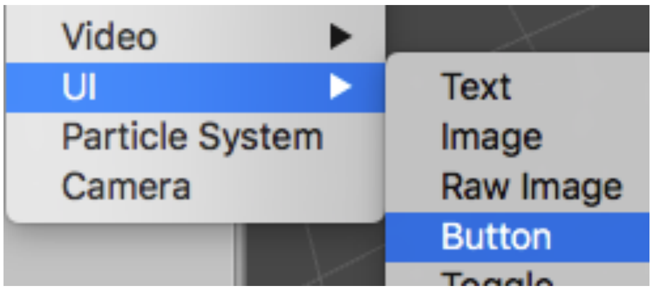
We also want to reset the position of the zombie



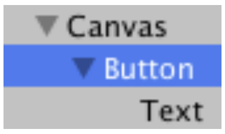
```
if (touch.phase == TouchPhase.Began && !EventSystem.current.IsPointerOverGameObject(0))
{
    transform.localPosition = Vector3.zero;

    var screenPosition = Camera.main.ScreenToViewportPoint(touch.position);
```

Add UI buttons

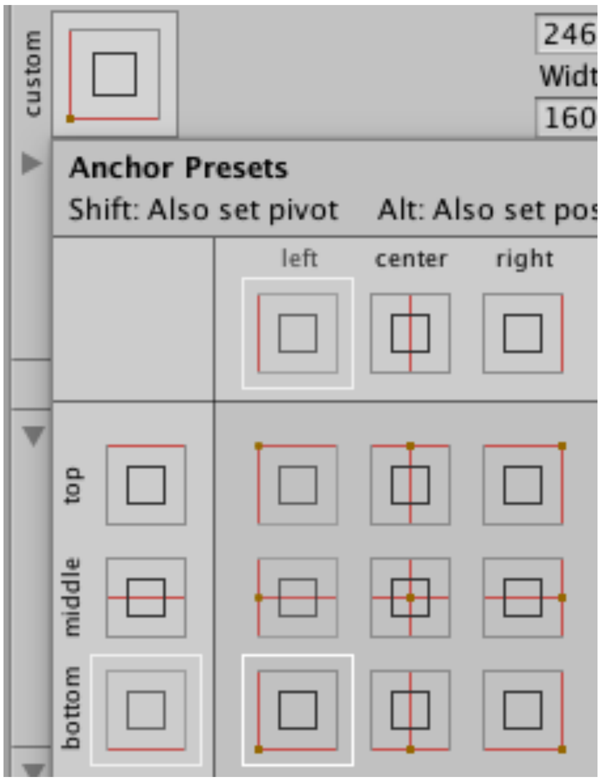
Now go back to the hierarchy window and right click to add the UI button



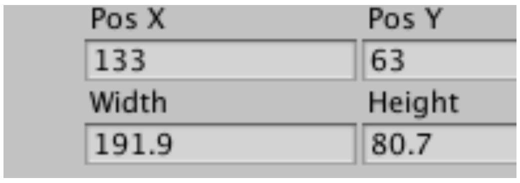
Select the button object



and change the anchor to bottom-left

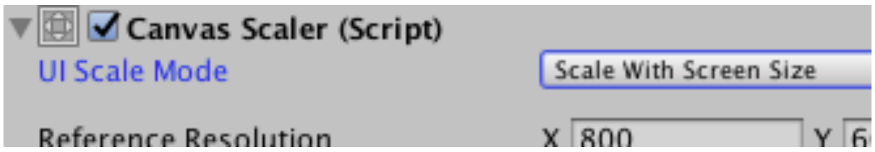


the adjust the button size by swiping over the Width and Height attributes



Pos X	Pos Y
133	63
Width	Height
191.9	80.7

Select the Canvas object and set the scaling attribute che "Scale With Screen Size"

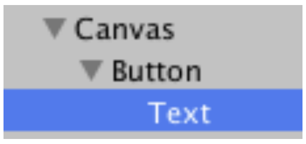


Canvas Scaler (Script)

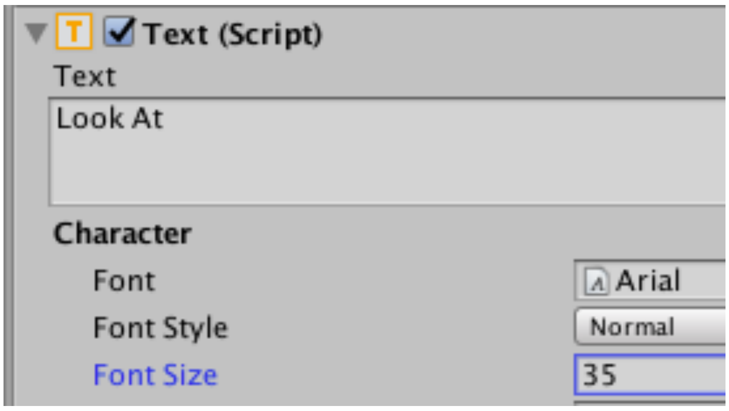
UI Scale Mode: Scale With Screen Size

Reference Resolution X 800 Y 6

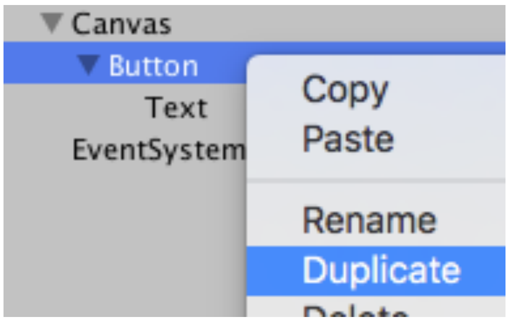
- Select the Text object



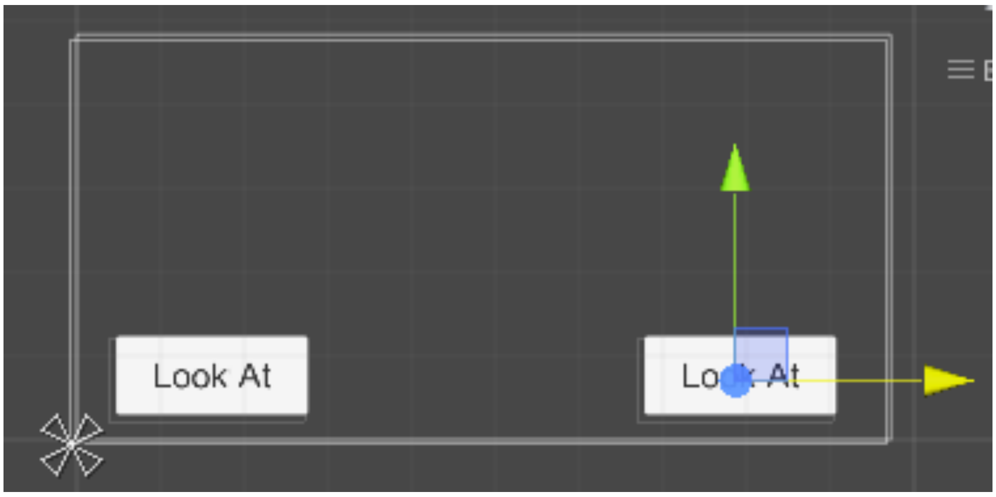
set the text to "Look At" and adjust the Font Size


Text (Script)
Text
Look At
Character
Font: Arial
Font Style: Normal
Font Size: 35

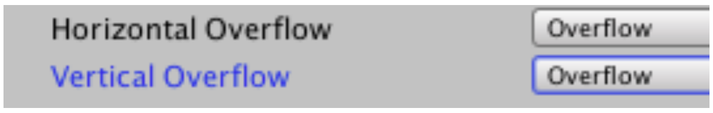
Select the Button, right click and duplicate it



then drag this button in the scene

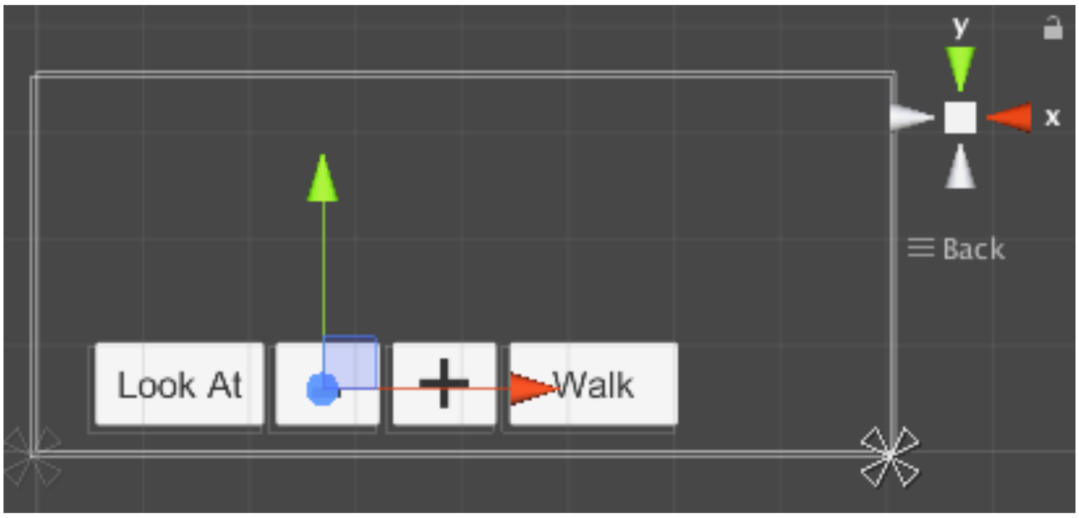


Anchor this button at bottom right corner and set the text to "Walk"
Now duplicate the Walk button to create the "+" button. Adjust the position. For this button set horizontal and vertical overflow attributes to "Overflow"


Horizontal Overflow: Overflow
Vertical Overflow: Overflow

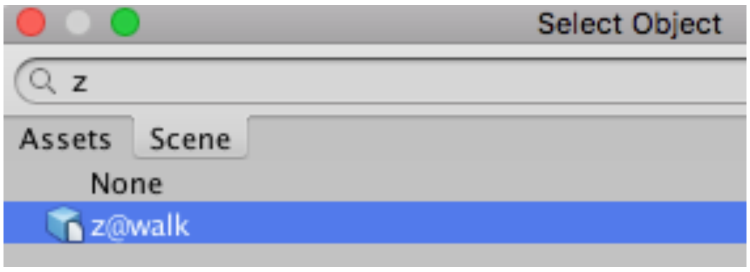
Duplicate the scale size up button to create the "-" button.

- Save the scene with Cmd +S

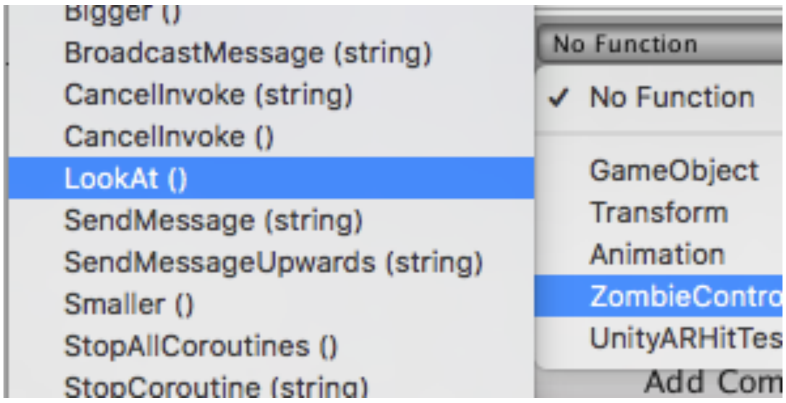


- Now, for each button let's create a link to the corresponding function

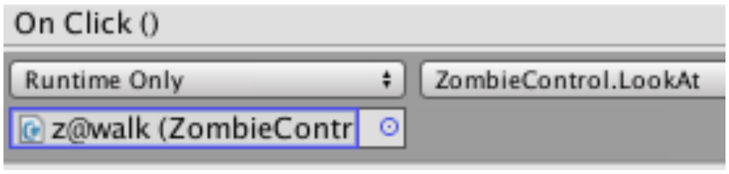
Select the "Look At" button, the in the Inspector window "On Click()" press the plus button, click the small circle and search the zombie object z@walk from the assets



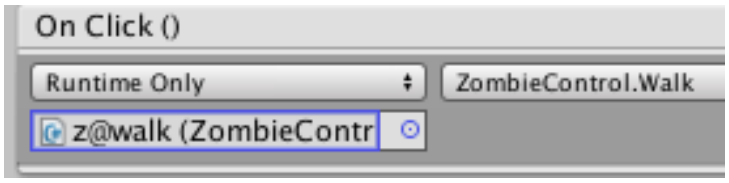
In the function selector, set ZombieControl -> LookAt()



The final configuration looks like this



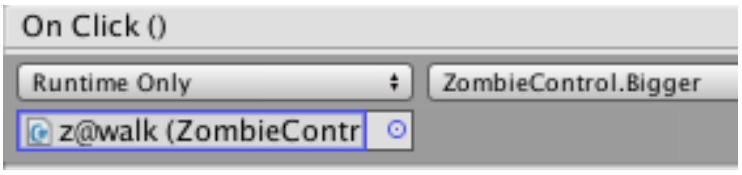
- Repeat the same for all buttons.



On Click ()

Runtime Only — ZombieControl.Walk

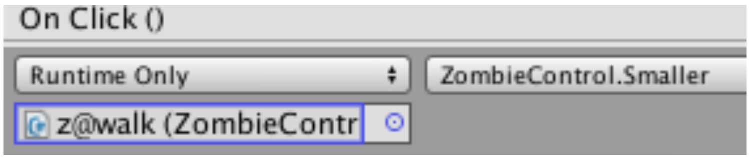
z@walk (ZombieContr



On Click ()

Runtime Only — ZombieControl.Bigger

z@walk (ZombieContr



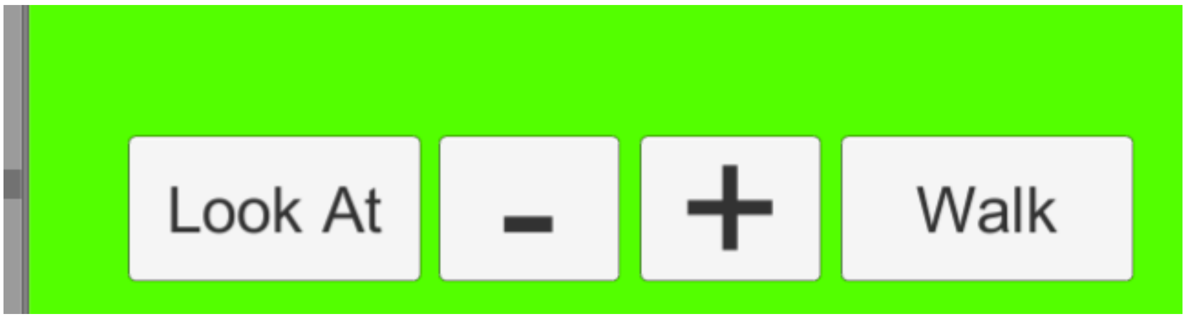
On Click ()

Runtime Only — ZombieControl.Smaller

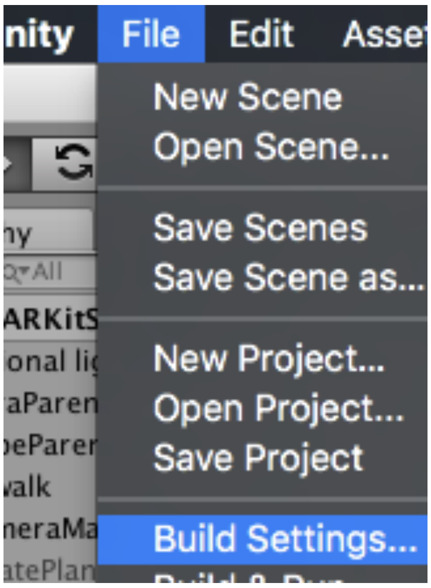
z@walk (ZombieContr

Let’s run the app.
Switch the bottom pane from Project to Console
Click the Play button

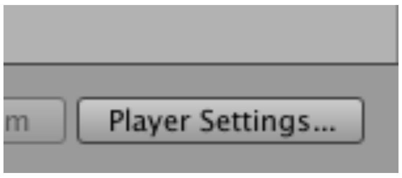
You should see something like this (this is the Game window)



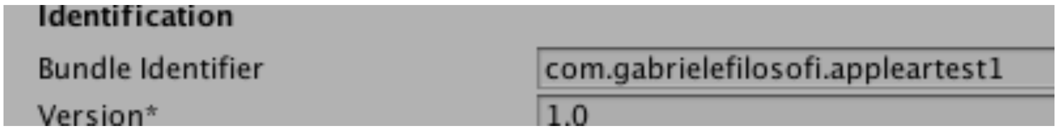
Change Build Settings
File > Build Settings



then click *Player Settings* button



Set the Bundle Identifier

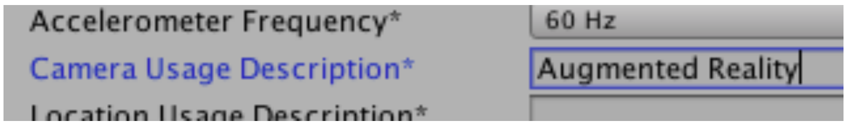


Identification

Bundle Identifier com.gabrielefilosofi.appleartest1

Version* 1.0

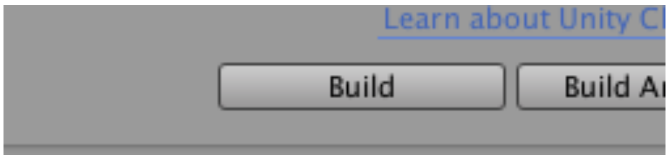
and put a string in the Camera Usage Description, for example



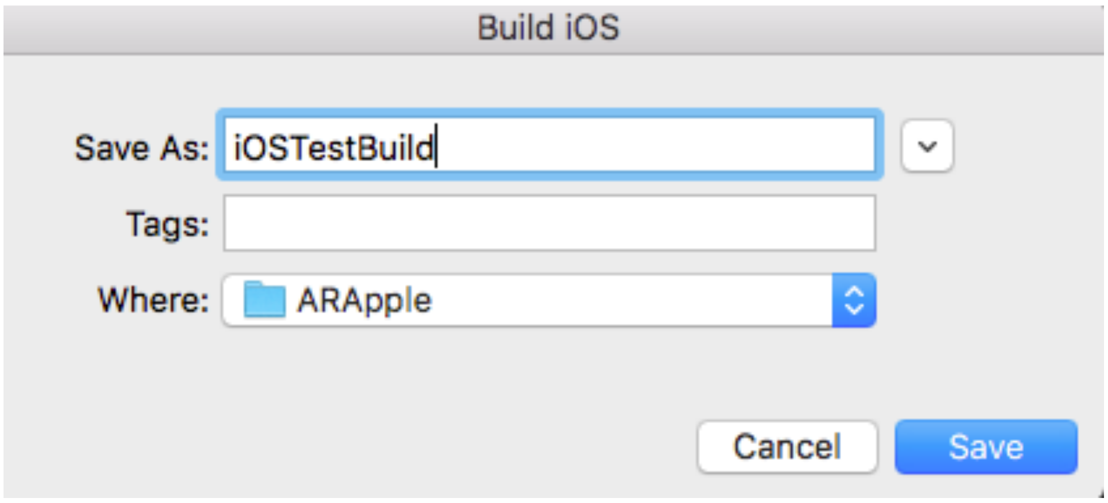
Accelerometer Frequency* 60 Hz

Camera Usage Description* Augmented Reality

Let’s press the Build button



and set a name



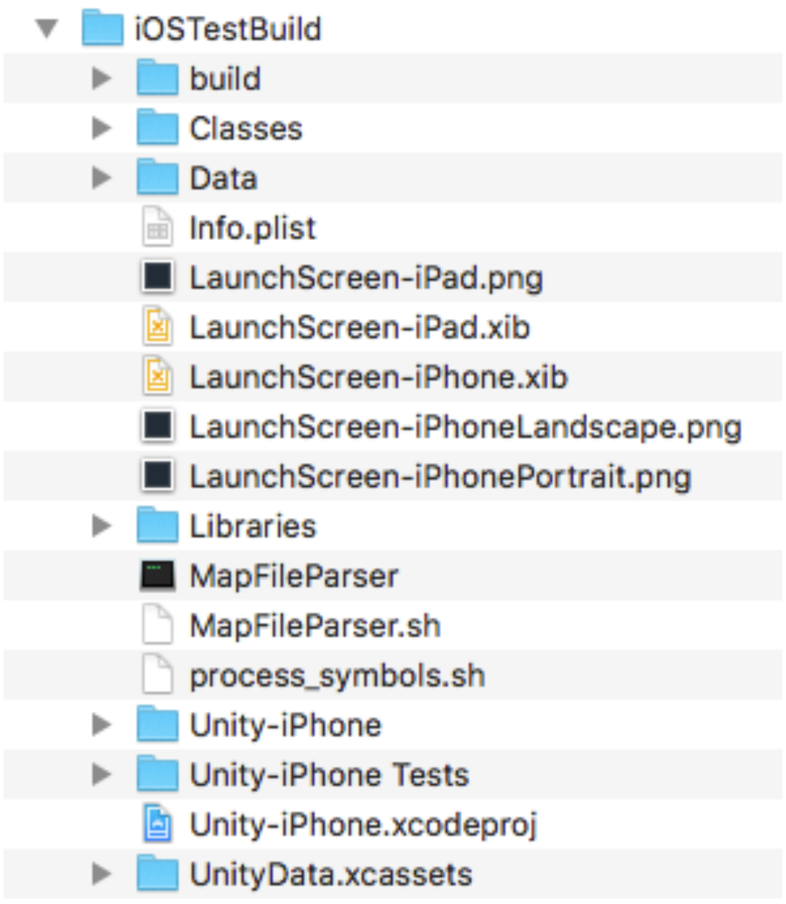
Build iOS

Save As: iOSTestBuild

Tags:

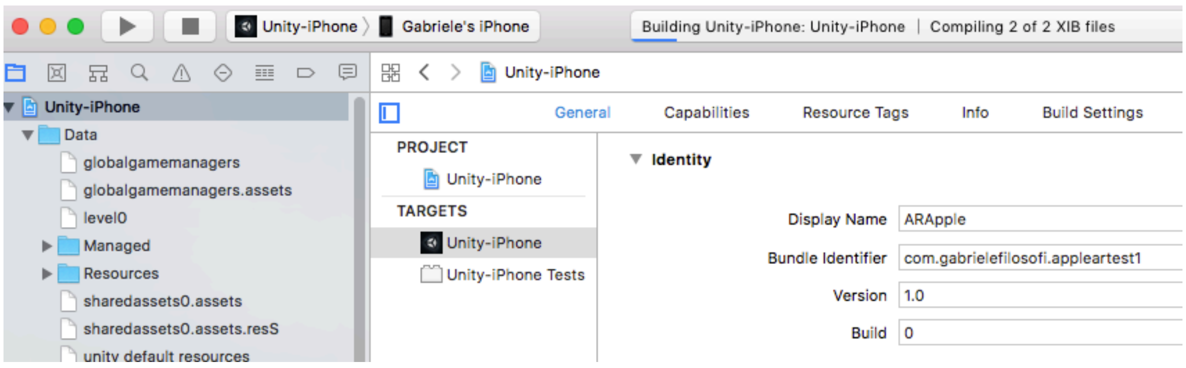
Where: ARApple

A folder has been created with a lot of stuff inside



```
▼ iOSTestBuild
    ▶ build
    ▶ Classes
    ▶ Data
      Info.plist
      LaunchScreen-iPad.png
      LaunchScreen-iPad.xib
      LaunchScreen-iPhone.xib
      LaunchScreen-iPhoneLandscape.png
      LaunchScreen-iPhonePortrait.png
    ▶ Libraries
      MapFileParser
      MapFileParser.sh
      process_symbols.sh
    ▶ Unity-iPhone
    ▶ Unity-iPhone Tests
      Unity-iPhone.xcodeproj
    ▶ UnityData.xcassets
```

- Launch Xcode 9 beta, open the Unity-iPhone.xcodeproj and run it over your iPhone (iOS 11 required)



Unity-iPhone 〉 Gabriele's iPhone

Building Unity-iPhone: Unity-iPhone | Compiling 2 of 2 XIB files

Unity-iPhone

- Data
  - globalgamemanagers
  - globalgamemanagers.assets
  - level0
  - Managed
  - Resources
  - sharedassets0.assets
  - sharedassets0.assets.resS
  - unity default resources

General | Capabilities | Resource Tags | Info | Build Settings

PROJECT
- Unity-iPhone

TARGETS
- Unity-iPhone
- Unity-iPhone Tests

Identity

Display Name	ARApple
Bundle Identifier	com.gabrielefilosofi.appleartest1
Version	1.0
Build	0