

ΔΗΔΑΚΤΟΡΑΣ ΕΥΚΛΕΙΔΗΣ ΚΕΡΑΜΟΠΟΥΛΟΣ

Εργασία Επαυξημένης Πραγματικότητας

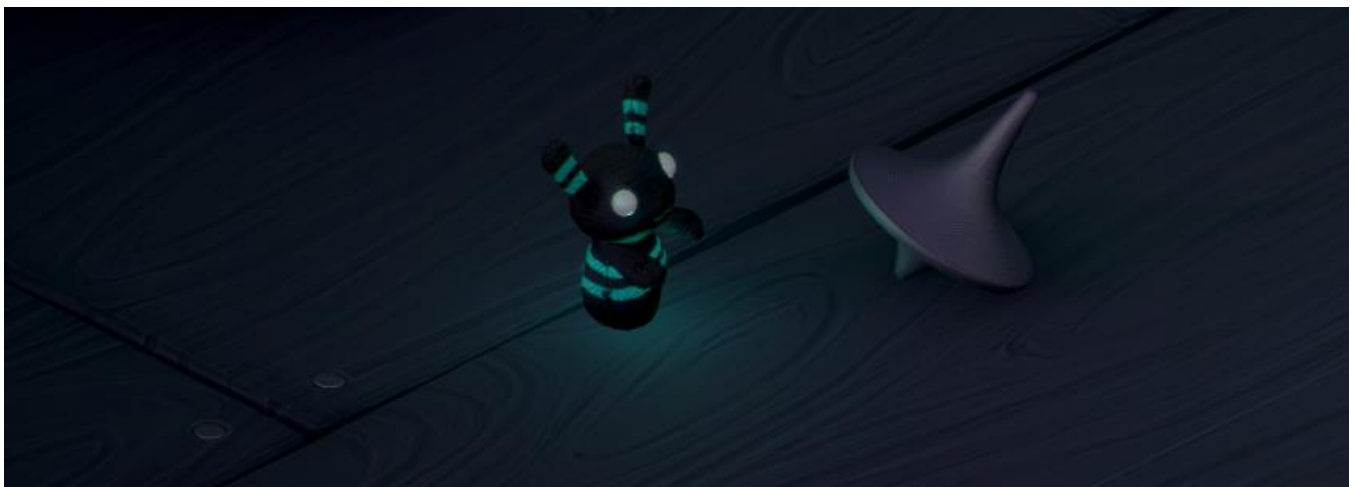
2η Άσκηση Βραβείο

Στεφάνια-Μαρία Μακρυγιαννάκη

ΑΜ: 164703

Πίνακας περιεχομένων

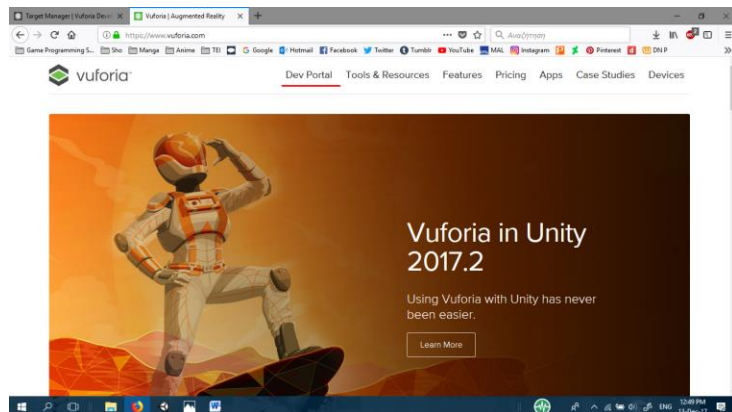
Πρώτο Μέρος	2
Vuuforia	2
License Manager:.....	2
Target Manager:	2
Unity	3
Δεύτερο Μέρος	6
Joystick	6
Ανάλυση του κώδικα:	8
Animation	9
Android.....	10
Βιβλιογραφία:.....	11



Πρώτο Μέρος

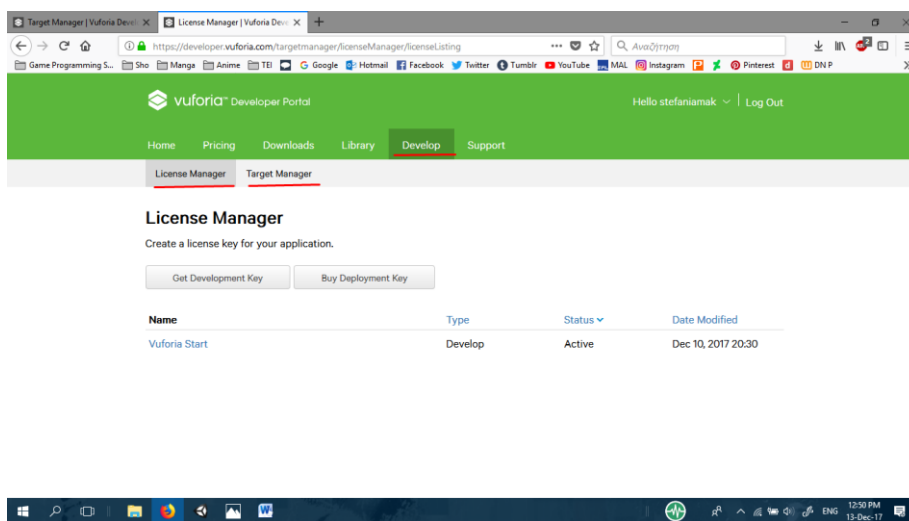
Vuforia

Η διαδικασία ξεκίνησε από την επίσκεψη του site της Vuforia. Μέσω της επιλογής “Dev Portal” μεταφέρθηκα στο κομμάτι της σελίδας για την προσθήκη εικόνων targets, δηλαδή τις εικόνες που θέλουμε να αναγνωρίζει η κάμερά μας για να λειτουργήσει το πρόγραμμα επαυξημένης πραγματικότητας (AR). Στη συνέχεια περιγράφω αναλυτικά τα βήματα που χρειάστηκα να ανεβάσω στη σελίδα αυτές τις φωτογραφίες.



License Manager:

Σε αυτό το κομμάτι φτιάχνω ένα κλειδί, το οποίο θα χρειαστεί να προσθέσω μέσα στο Unity για να τρέξει η Vuforia. Η δημιουργία γίνεται μέσω δύο κλικ, ένα του “Get Development Key”, του Confirm αφού γράψω ένα όνομα τις επιλογής μου, και γίνει η αποδοχής των όρων που δίνονται.



Target Manager:

Εδώ γίνεται η δημιουργία βάσεων για το ανέβασμα φωτογραφιών target. Με το πάτημα του “Add Database” προσθέτω τη βάση, και στη συνέχεια με το πάτημα της δημιουργημένης πια βάσης προσθέτω όσες εικόνες επιθυμώ.

License Manager
Target Manager

Target Manager

Use the Target Manager to create and manage databases and targets.

Add Database

Database	Type	Targets	Date Modified
Ace	Device	1	Dec 11, 2017 19:53
Creepy	Device	6	Dec 11, 2017 23:00
LastOne	Device	6	Dec 12, 2017 23:19
target	Device	1	Dec 10, 2017 20:39

Για παράδειγμα επιλέγω μία ήδη δημιουργημένη βάση, την “LastOne”. Τα αποτελέσματα φαίνονται στην παρακάτω εικόνα. Τα αστεράκια αντιπροσωπεύουν το πόσο καλή και αναγνωρίσιμη είναι η εικόνα με βάση το Vuuforia, όσο πιο πολλά αστεράκια έχει, τόσο πιο εύκολα και αποδοτικά θα αναγνωρίσει η κάμερά την εικόνα. Κατεβάζω αυτά τα targets μέσω του “Download Database (All)”.







LastOne
[Edit Name](#)

Type: Device

Targets (6)

Add Target

Download Database (All)

<input type="checkbox"/>	Target Name	Type	Rating	Status	Date Modified
<input type="checkbox"/>	 A	Single Image	★★★★★	Active	Dec 12, 2017 23:19
<input type="checkbox"/>	 12	Single Image	★★★★★	Active	Dec 12, 2017 23:17
<input type="checkbox"/>	 room-wallpapers-grunge-1	Single Image	★★★★★	Active	Dec 12, 2017 23:15
<input type="checkbox"/>	 10	Single Image	★★★★★	Active	Dec 12, 2017 22:59
<input type="checkbox"/>	 9	Single Image	★★★★★	Active	Dec 12, 2017 22:57
<input type="checkbox"/>	 8	Single Image	★★★★★	Active	Dec 12, 2017 22:55

Last updated: Today 01:13 PM [Refresh](#)

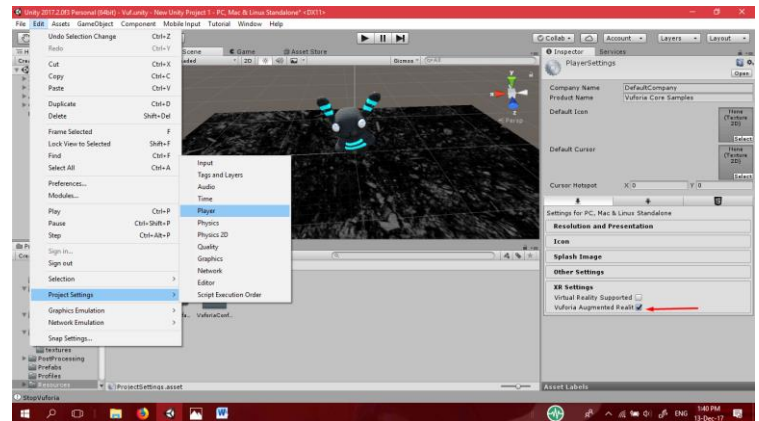
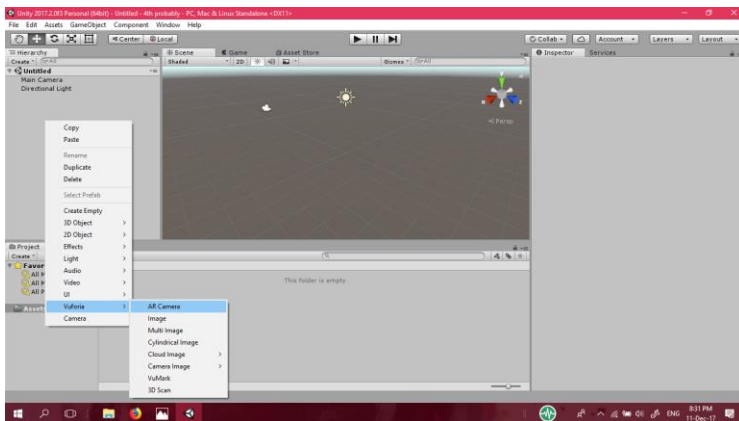
Unity

Μόλις ολοκληρώσω με τις παραπάνω ρυθμίσεις, ανοίγω το Unity και φτιάχνω ένα καινούριο project. Σε αυτό το project κάνω δύο πράγματα αρχικά για να βάλω τη Vuuforia μέσα σε αυτό.

- Right click>>Vuuforia>>AR Camera,

Με αυτή την ενέργεια προσθέτουμε όλη την ήδη εγκατεστημένη έκδοση της Vuuforia στο project μου.

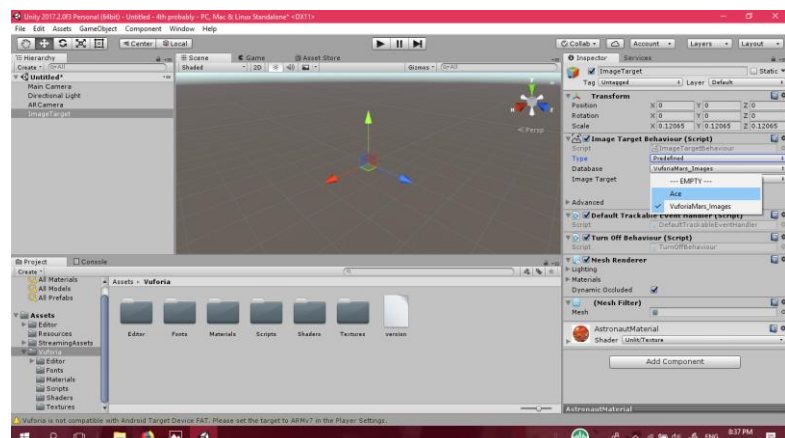
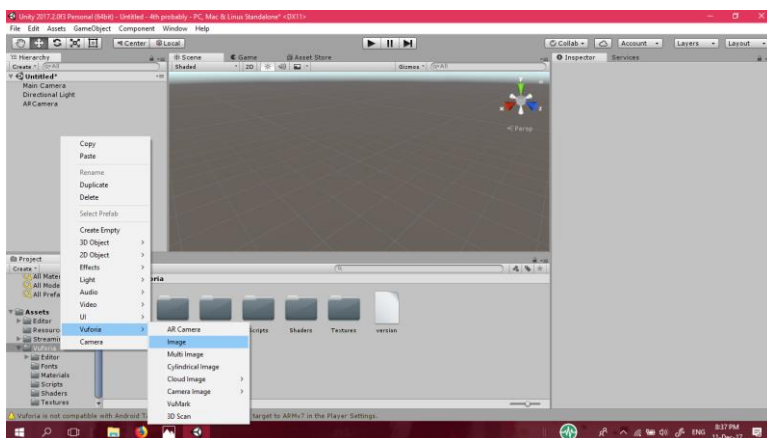
b. Project Settings>>Player, και κάτω δεξιά ενεργοποιώ το “Vuforia Augmented Reality”.

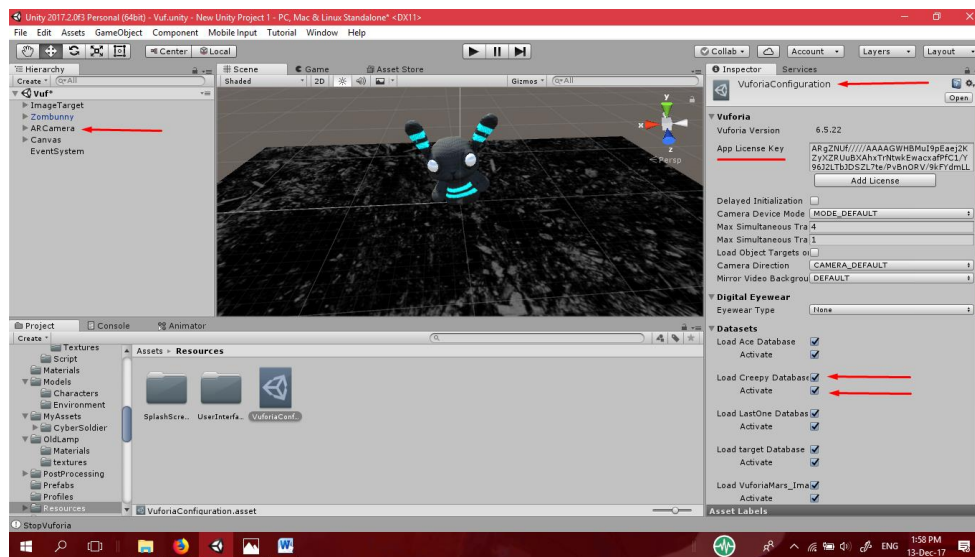


Στη συνέχεια προσθέτω τα target images στο project. Επιλέγω μέσα στο Unity το Assets>>import Package>>Custom Package και στο παράθυρο που εμφανίζεται βρίσκω και επιλέγω το αρχείο που κατέβασα από το Target Manager προηγουμένως στο site της Vuforia.

Το μόνο που μένει τώρα που μπήκαν και τα target images, είναι να τα εφαρμόσω. Για αυτό χρειάζονται τρία βήματα.

- Είτε Right Click στην περιοχή που φαίνετε στην κάτω αριστερά εικόνα, είτε GameObject >> Vuforia >> Image. Έτσι δημιουργώ το “ImageTarget”.
- Επιλέγω το “ImageTarget” που μόλις δημιούργησα και στην δεξιά πλευρά στον Component “Image Target Behavior” επιλέγω το όνομα του Database μας και ακριβώς από κάτω το ποιο από τα Targets που έβαλα σε εκείνο το Database θέλω να χρησιμοποιήσω.
- Επιλέγω από το σημείο που βρίσκετε και το ImageTarget το ARCamera, και ύστερα από την δεξιά πλευρά το Vuforia Behavior και “Open Vuforia configuration”. Εκεί κάνω δύο πράγματα. Προσθέτω το App License Key που δημιούργησα αρχικά στο site της Vuforia, και δεύτερον επιλέγω την βάση με τα Targets ακριβώς από κάτω, το Activate option θα εμφανιστεί αφού γίνει check η επιλογή της βάσης. (βλέπετε 3^η εικόνα από κάτω).



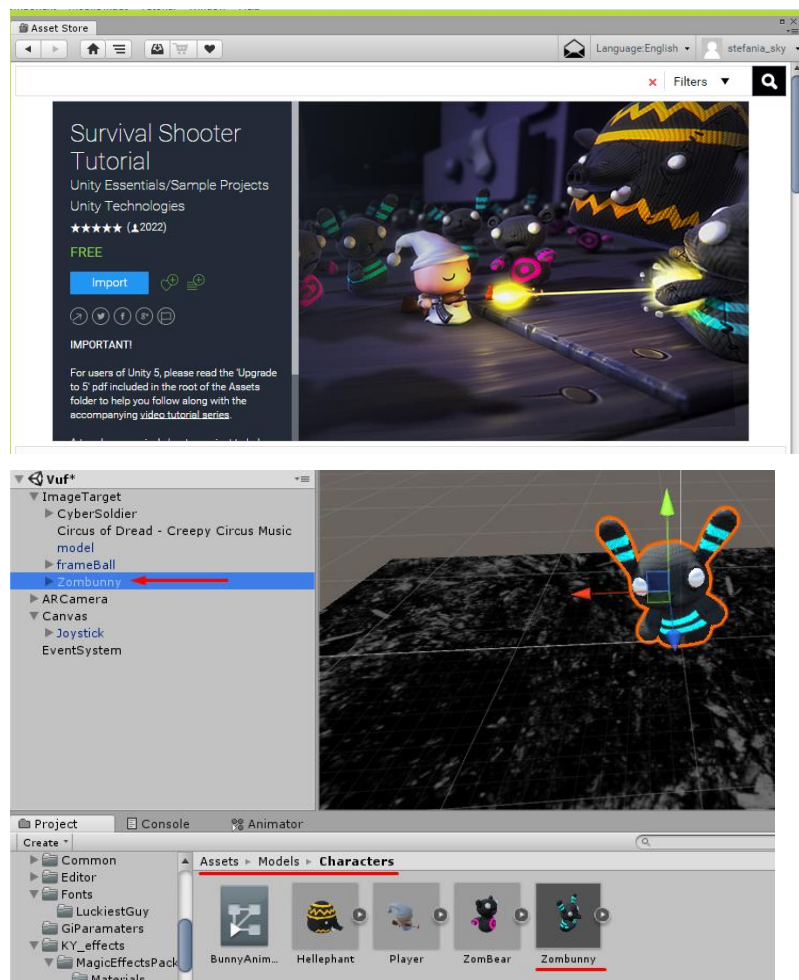


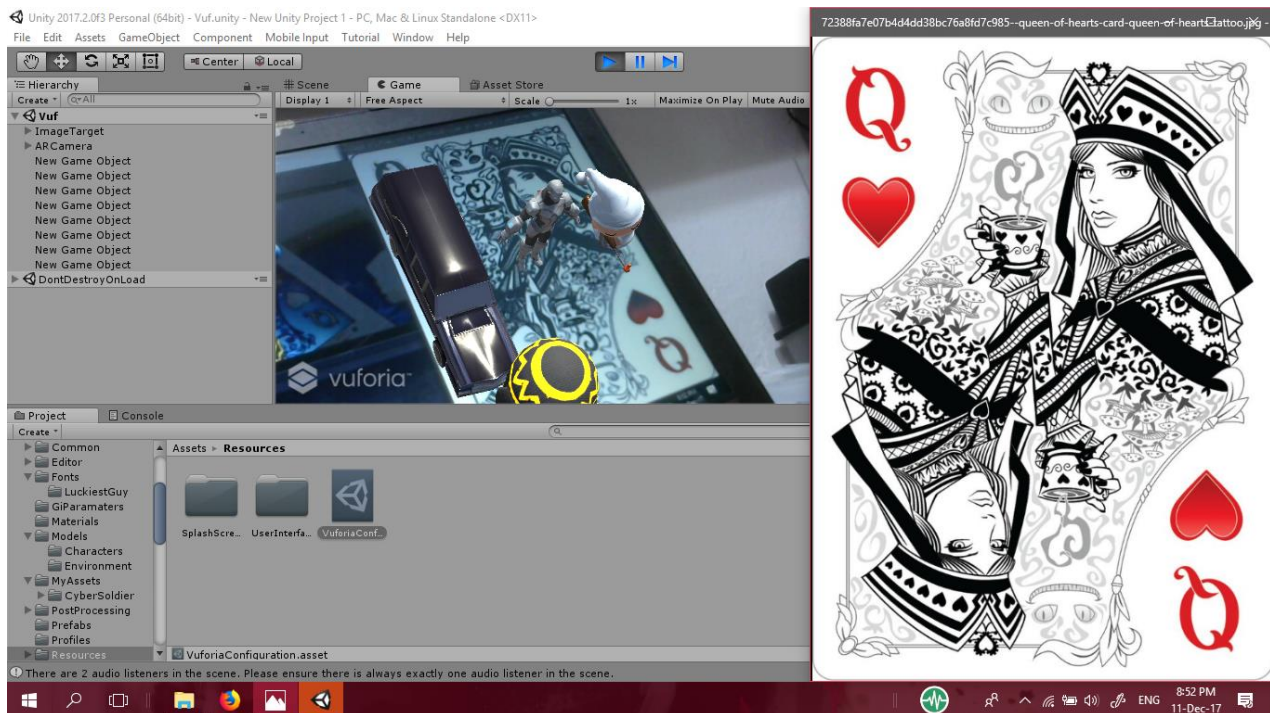
Στη συνέχεια βάζω ένα αντικείμενο πάνω από το Target Image. Αυτό που έκανα, είναι να πάω στο ενσωματωμένο shop της Unity (Asset Store) και να κατεβάσω κάποια από τα Free Models που περιέχει. Αυτό που έχετε δει σε κάποιες από τις πάνω εικόνες είναι το τελικό μοντέλο που μου άρεσε περισσότερο από όλα και αποφάσισα να κρατήσω. Το όνομα του αντικειμένου που ανήκει αυτό το λαγουδάκι-κούκλα όπως θα περιέγραφα, ανήκει στο “Survival Shooter Tutorial”.

Στη συνέχεια κάνω Drag and Drop τον χαρακτήρα που θέλουμε από τα αρχεία στο κάτω μέρος του Project –μέσα- στο ImageTarget. Μηδενίζω τις συντεταγμένες του χαρακτήρα από τις ρυθμίσεις του (δηλαδή $x=0$ $y=0$ $z=0$) και ο χαρακτήρας μας θα βρίσκεται στη μέση του Target Image.

Με αυτό, το βασικό πρόγραμμα μπορεί να τρέξει και να εμφανίσει τον χαρακτήρα πάνω στο Target Image.

Ένα πρώτο test που έκανα του προγράμματος με την κάμερα να δείχνει στην οθόνη του υπολογιστή, εμφάνισε το παρά κάτω:



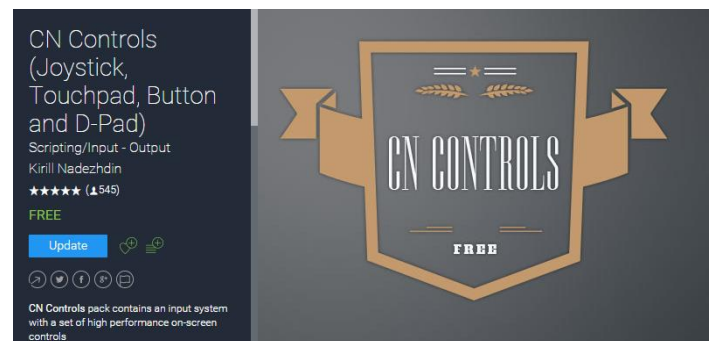


Δεύτερο Μέρος

Στο δεύτερο κομμάτι θα περιγράψω το πώς πραγματοποιήσα την κίνηση του λαγού-χαρακτήρα μέσω ενός Joystick.

Joystick

Για να βάλω το Joystick χρειάστηκε αρχικά να κατεβάσω από το Asset Store του Unity ένα πακέτο κουμπιών που να εμπεριέχει και Joystick. Συγκεκριμένα κατέβασα το “CN Controls (Joystick, Touchpad, Button and D-Pad)” όπως φαίνετε στην δεξιά εικόνα.



Στη συνέχεια δημιούργησα έναν καμβά (Canvas) και τοποθέτησα το Joystick μέσα σε αυτόν. Τον καμβά τον δημιούργησα ακολουθώντας το μονοπάτι “GameObject >> UI >> Canvas”, και μετά από αυτό κάτω στον κατάλογο project κάνω drag and drop μέσα στον καμβά το Joystick από το Assets >> Standard Assets >> CNControlls >> Prefabs.

Οι μόνες αλλαγές που έκανα στο joystick είναι να αλλάξω την θέση του στην οθόνη, και να του αφαιρέσω την δυνατότητα να μετακινεί την βάση του, να εξαφανίζεται όταν ξεκινάει η εφαρμογή και να

εμφανίζεται όπου πατάει το δάχτυλο. Όλα τα άλλα είναι προγραμματισμένα από το άτομο που κατέβασα το πακέτο.

Το joystick έχει εξ αρχής ένα script για την λειτουργία του (διαφορετικό από αυτό που αναφέρθηκε στην προηγούμενη παράγραφο) το οποίο χρησιμοποιώ για να συνδέσω την κίνηση τον χαρακτήρα με το joystick. Επειδή έχω 1 χαρακτήρα που θέλω να ελέγχω, μετακινώ το script από το joystick και το κάνω μέσω drag and drop πάνω στο “Zombunny”, που είναι το όνομα που έχει ο χαρακτήρας. Από εκεί ακολούθησε το παρακάτω coding μέσα στο script.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using CnControls;
5
6 public class InputController : MonoBehaviour {
7
8     // Variables for the speed and turning speed of the rabbit
9     public float speed = 5f;
10    public float turnSpeed = 5f;
11
12    Animator anim;
13
14    // Use this for initialization
15    void Start () {
16        anim = GetComponent <Animator>();
17    }
18
19    // Update is called once per frame
20    void Update () {
21
22        Vector2 direction2d = JoystickDirection ();
23        Vector3 direction3d = new Vector3 (direction2d.x, 0, direction2d.y);
24
25        // Movement and Rotation
26        if (direction2d != Vector2.zero) {
27            transform.position += direction3d * Time.deltaTime * speed;
28
29            Quaternion targetRotation = Quaternion.LookRotation (direction3d);
30            transform.rotation = Quaternion.Lerp (transform.rotation, targetRotation, Time.deltaTime * turnSpeed);
31
32            anim.SetBool ("IsWalking", true);
33        } else
34            anim.SetBool ("IsWalking", false);
35    }
36 }
37
38 // Get the joystick's direction from the joystick input axis
39 Vector2 JoystickDirection () {
40     float verticalValue = CnInputManager.GetAxis ("Vertical");
41     float horizontalValue = CnInputManager.GetAxis ("Horizontal");
42
43     return new Vector2 (horizontalValue, verticalValue);
44 }
45 }
46
```


Ανάλυση του κώδικα:

- a. Κλάση `public class InputController : MonoBehaviour`

```
public float speed = 5f;           : μεταβλητή για την ταχύτητα μετακίνησης
public float turnSpeed = 5f;       : μεταβλητή για την ταχύτητα περιστροφής
```

- b. Συνάρτηση `void Start ()`

Χρησιμοποιείτε για την αρχικοποίηση μεταβλητών της κλάσης.

- `anim = GetComponent <Animator>();` : παίρνω το component για το animation

- c. Συνάρτηση `void Update ()`

Αυτή η συνάρτηση καλείτε κάθε frame. Την χρησιμοποιώ για να αλλάζω τη θέση και τη περιστροφή του παίκτη.

- `Vector2 direction2d = JoystickDirection ();`

Σε αυτή τη μεταβλητή αποθηκεύεται η δισδιάστατη κατεύθυνση του joystick.

- `Vector3 direction3d = new Vector3 (direction2d.x, 0, direction2d.y);`

Χρησιμοποιείται για να “μετατρέψουμε” το 2D διάνυσμα του joystick σε “3D”, διότι χρειάζεται ο χαρακτήρας να μετακινείται στους άξονες x και z και όχι x και y.

- `if (direction2d != Vector2.zero)`

Αν οι συντεταγμένες του joystick είναι διαφορετικές του μηδέν, δηλαδή ο χρήστης μετακινεί το joystick, τότε να γίνεται η κίνηση.

Η κίνηση πραγματοποιείται λαμβάνοντας τη θέση του χαρακτήρα μέσω του `transform.position`, στο οποίο γίνεται συνεχής πρόσθεση του γινομένου του διανύσματος που έχουμε πάρει από το joystick, του χρόνου μεταξύ των frames της εφαρμογής και της ταχύτητας που θέσαμε πιο πάνω.

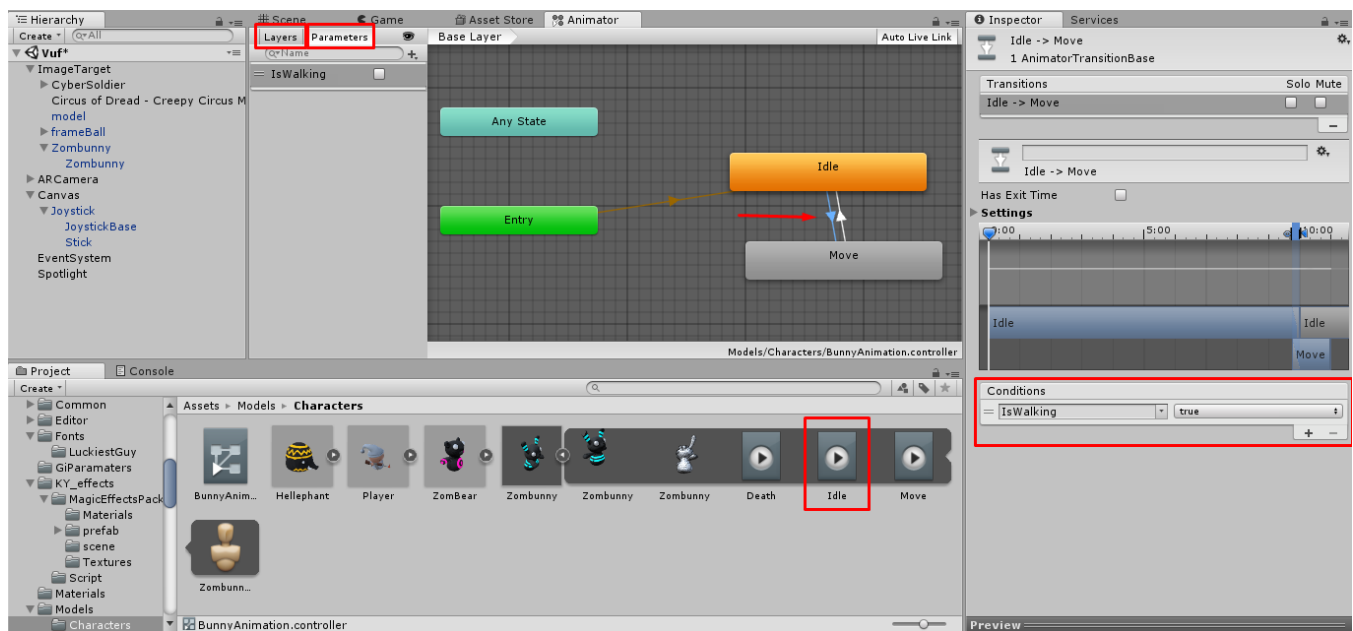
Έπειτα υπολογίζεται η τελική περιστροφή του χαρακτήρα μέσω της συνάρτησης `LookRotation` της Unity, δίνοντας σαν όρισμα το τριπλό διάνυσμα κατεύθυνσης (`direction3d`) του joystick. Η τελική αυτή περιστροφή αποθηκεύεται στην μεταβλητή `targetRotation`. Έπειτα γίνεται ανανέωση της περιστροφής του χαρακτήρα μέσω της συνάρτησης `Lerp` της Unity, η οποία κάνει παρεμβολή από μία αρχική τιμή (τρέχουσα περιστροφή του χαρακτήρα) σε μία τελική (`targetRotation`) που θέλουμε να φτάσουμε ανάλογα με την ταχύτητα στροφής του χαρακτήρα (`turnSpeed`).

Τέλος, θέτω το κατάλληλο animation στον χαρακτήρα ανάμεσα στο ακίνητο (idle) και περπάτημα (walking). Στο Animator component του χαρακτήρα έχω δηλώσει μια Boolean μεταβλητή IsWalking την οποία τη θέτω true ή false ανάλογα αν όντως ο χαρακτήρας μετακινείται ή όχι.

d. Συνάρτηση **Vector2** JoystickDirection ()

Ανάγνωση των συντεταγμένων του joystick, δηλαδή της θέσης x και y αντίστοιχα. Ύστερα επιστρέφει έναν vector με τη κατεύθυνση του joystick.

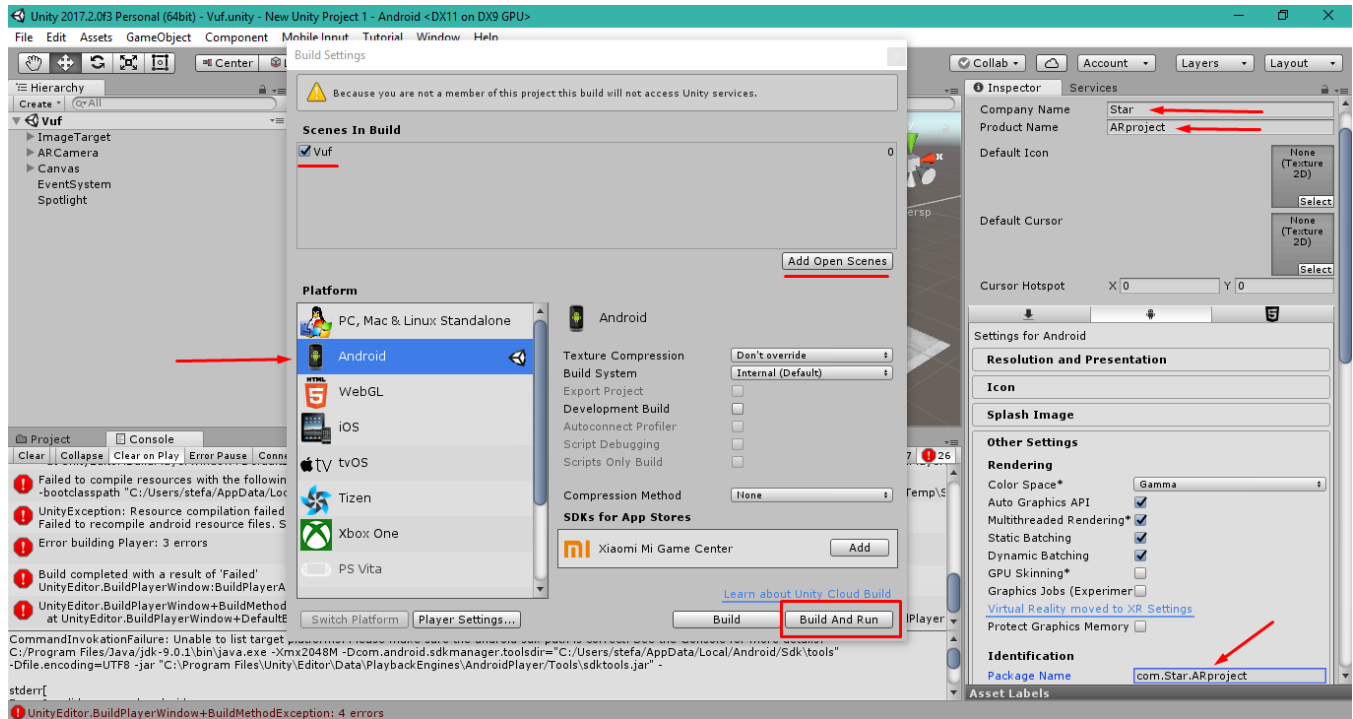
Animation



- Φτιάχνω έναν Animator, πατώντας Assets >> Create >> Animator Controller.
- Δημιουργώ μια στρώση πατώντας την επιλογή "Layers" και του εικονιδίου «+».
- Προσθέτω μια παράμετρο Boolean πατώντας την επιλογή "Parameters", ξανά μέσω του εικονιδίου «+», με το όνομα "IsWalking".
- Κάνω drag and drop το animation "Idle" και "Move" του χαρακτήρα Zombunny από το Assets >> Models >> Characters μέσα στον Animator.
- Δεξί κλικ στο "Idle" πορτοκαλί πλαίσιο που δημιουργήθηκε στον Animator, επιλέγω το "Make transition" και το ενώνω με το Move. Αντίστοιχα κάνω από το "Move" προς το "Idle".
- Σε κάθε μετάβαση (transition) επιλέγω στα "Conditions" την παράμετρο "IsWalking" που έφτιαξα προηγουμένως, και την τοποθετώ ως true κατά την μετάβαση από "Idle" σε "Move" και ως false στη μετάβαση από το "Move" σε "Idle".

Android

Τέλος, δημιουργήθηκε η εφαρμογή για Android.



Πολύ μεγάλη βοήθεια ήταν οι τελευταίοι ιστότοποι στη Βιβλιογραφία. Αρχικά κατέβασα τις προ τελευταίες εκδόσεις από δύο αρχεία, το SDK και το JDK τα οποία τα βρίσκω από το Edit >> Preferences >> External Tools. Στη συνέχεια μέσω του File >> Build Settings εμφανίζεται το παραπάνω παράθυρο. Εκεί γίνονται οι αλλαγές στα επισημασμένα με κόκκινο κομμάτια της φωτογραφίας από πάνω [και μίας ακόμα αλλαγής που δεν φαίνεται στην εικόνα, η διαγραφή της επιλογής του “Auto Graphics API”]. Τέλος συνδέουμε μια συσκευή Android και τρέχουμε το πρόγραμμα πατώντας το “Build And Run”.

Βιβλιογραφία:

- Δημιουργία assets για το Unity:

<https://developer.vuforia.com/>

- Βίντεο για τη δημιουργία της εφαρμογής επαυξημένης πραγματικότητας (AR)

<https://youtu.be/HnjbTytHH6U> (Vuforia Unity Android Tutorial, Your First AR App in 20 minutes, Vergium, 15/8/2016)

<https://www.youtube.com/watch?v=WdfStRynCLw> (Augmented Reality with Unity3D and Vuforia, 2/9/2017)

- Για τα Animations του χαρακτήρα

<https://unity3d.com/learn/tutorials/projects/survival-shooter/player-character?playlist=17144> (Player Character)

- Για την περιστροφή του χαρακτήρα

<https://docs.unity3d.com/ScriptReference/Quaternion.LookRotation.html> (Quaternion.LookRotation)

- Μουσική

<https://www.youtube.com/watch?v=bKFixiZtKVU> (Circus of Dread - Creepy Circus Music)

- Για την δημιουργία της Android εφαρμογής

<https://www.sitepoint.com/how-to-build-an-ar-android-app-with-vuforia-and-unity/> (How to Build an AR Android App with Vuforia and Unity)

<https://answers.unity.com/questions/1323731/unable-to-list-target-platforms-please-make-sure-t.html> (15/3/2017)

<https://answers.unity.com/questions/828689/resource-compilation-failed-failed-to-recompile-an.html> (28/11/2017)