ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΦΡΑΓΚΟΥΛΗΣ ΔΗΜΗΤΡΗΣ

Σχεδιασμός και Υλοποίηση διαδραστικού παιχνιδιού Η/Υ με τη χρήση της μηχανής γραφικών Unity

Σύνταξη από : ΦΡΑΓΚΟΥΛΗ ΔΗΜΗΤΡΙΟ ΑΜ : 1608

Επιβλέπων: Φούρναρης Απόστολος

Αντίρριο, 2017

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία με θέμα τον «Σχεδιασμός και Υλοποίηση δια δραστικού παιχνιδιού Η/Υ με τη χρήση της μηχανής γραφικών Unity», του τμήματος ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ, του τεχνολογικού εκπαιδευτικού ιδρύματος Δυτικής Ελλάδας.

Στο σημείο αυτό αισθανόμαστε την ανάγκη να εκφράσουμε τις ειλικρινείς και

θερμές ευχαριστίες μας σε όσους συνέβαλαν στην ολοκλήρωση αυτής της

προσπάθειας :

Και πρώτα απ' όλα, στον επιβλέπον καθηγητή μου Φούρναρη Αποστόλη για τη

συνεχή καθοδήγηση, την αμέριστη υποστήριξη, τις ουσιώδεις συμβουλές, καθώς και

την αδιάκοπη συμπαράσταση και ενθάρρυνση που μου παρείχε σε όλο αυτό το

διάστημα.

Περιεχόμενα

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

Περίληψη

Η παρούσα πτυχιακή εργασία ασχολείται με την ανάπτυξη παιχνιδιού με την βοήθεια της μηχανής Unity3D. Η βιομηχανία των παιχνιδιών με τα Game Engine έχει δώσει την δυνατότητα στους νέους προγραμματιστές να τις χρησιμοποιούν και να εργάζονται εύκολα ένα μεγάλο μέρος από αυτούς. Επομένως , η παρακάτω πτυχιακή ασχολείται με την μελέτη και τη δημιουργία ενός βίντεο παιχνιδιού από το μηδέν με την καθοδήγηση της Unity - (Tutorials) για το παιχνίδι το οποίο ονομάζεται Survival Shooter. Πιο συγκεκριμένα μελετήσαμε σταδιακά ένα ένα βήμα και ολοκληρώσαμε το παιχνίδι προσθέτοντας επιπλέον πληροφορίες για τον χρήστη την ώρα που παίζει. Με λίγα λογία, το παιχνίδι μας λαμβάνει μέρος σε ένα δωμάτιο οπού διάφοροι εφιάλτες με μορφές όπως αρκουδάκια και λαγουδάκια προσπαθούν να πιάσουν και να σκοτώσουν τον ηρώα μας που φορώντας ένα νυχτικό και κρατώντας ένα μικροσκοπικό καλασνικοφ προσπαθεί να τους αποφύγει και να τους σκοτώσει. Μέσα στο δωμάτιο του ηρώα μας βρίσκονται πάρα πολλά καθημερινά αντικείμενα όπως για παράδειγμα μια βιβλιοθήκη μικρά αυτοκινητάκια , ένα ρολόι μια σβούρα και πολλά ακόμα. Χρησιμοποιώντας αυτά τα αντικείμενα μπορείς να κρυφτείς πίσω από αυτά (από πίσω τους) και να αποφύγεις τους εχθρούς που σε κυνηγάνε συνεχώς για να σε ακουμπήσουν και να σου αφαιρέσουν την ζωή. Το παιχνίδι παρέχει φιλικό περιβάλλον χρήσης, ευκολία χρήσης, φυσική ρεαλιστική κίνηση, συνδυασμό ποικιλίας τύπων γραφικών, σχέδιο κίνηση (animation) και ηχητικά εφέ διαφόρων τύπων ήχου.

Abstract

This Bachelor Thesis discusses the development of a computer game with the assistance of the Unity3D engine. This cross-platform game engine gives the opportunity to young programmers to use it and most of them can easily work on it. Moreover, this Bachelor Thesis discusses the development and deployment of a video game from scratch, the Survival Shooter, using the cross-platform Unity-(Tutorials). In particular, we considered gradually all the necessary steps and we completed the game by adding more information for the user as long as he is playing. In a few words, the game takes place in a room where different nightmares in the form of teddies and bunnies etc., try to capture and kill our hero, who in a nightie and with a tiny Kalashnikov in his hands tries to outrun and kill them. In the room of our hero, we can find many ordinary things such as a bookcase, car models, a clock, a spinning top and much more. We can use these things and hide behind them in order to outrun the enemies who are continuously chasing the hero trying to touch him and cost him a life till the “game over”. The game is friendly and easy to use. It, also, possesses natural, true-to-life movements, combines a wide variety of graphics, animation and different types of sound effects.

1. Εισαγωγή

Βιντεοπαιχνίδι και μηχανή γραφικών unity

Η παρούσα πτυχιακή εργασία πραγματοποιήθηκε στην Πάτρα κατά το ακαδημαϊκό έτος 2016 - 2017 και δημιουργήθηκε στο πλαίσιο της ανάπτυξης 3D εφαρμογών ή παιχνιδιών εκπαιδευτικού χαρακτήρα. Το όλο Project αναπτύσσεται μέσω της πλατφόρμας ανάπτυξης 3D παιχνιδιών Unity 3D. Κατά συνέπεια το κύριο μέρος της παρούσας πτυχιακής εργασίας αποτελείται από την πρακτική ανάπτυξη της εφαρμογής του παιχνιδιού και όχι τόσο από βιβλιογραφικά στοιχεία.

Η ανάπτυξη ψηφιακών εφαρμογών, καθώς και ηλεκτρονικών παιχνιδιών αποτελεί ένα αρκετά μεγάλο και σημαντικό κεφάλαιο στον τομέα της πληροφορικής και του προγραμματισμού. Πλέον έχει γίνει μία βιομηχανία από μόνο του καθώς μεγάλες εταιρίες ασχολούνται αποκλειστικά με αυτό, διαθέτοντας πολυμελείς ομάδες οι οποίες εργάζονται για μήνες πάνω σε κάποιο συγκεκριμένο Project.

Εικόνα 1 από internet ¨Ο κόσμος των βιντεοπαιχνιδιών¨

Η χρήση της μηχανής γραφικών unity σου δίνει την δυνατότητα να δημιουργήσεις το δικό σου βιντεοπαιχνίδι χρησιμοποιώντας τις δικές σου σκέψεις και ιδέες. Μια εφαρμογή για να υλοποιηθεί χρειάζεται να χρησιμοποιήσουμε περισσότερα από ένα εργαλεία για να έχουμε σωστό και ικανοποιητικό αποτέλεσμα. Μερικά από αυτά είναι Photoshop - Google Sketchup - Blender - Audacity κ.α , γι' αυτό και τα μεγάλα παιχνίδια δημιουργούνται από εταιρείες όπου οι προγραμματιστές ποικίλουν σε αριθμό και γνώσεις ώστε ο καθένας να είναι υπεύθυνος για ένα συγκεκριμένο κομμάτι του παιχνιδιού όπως τμήμα γραφικών, animation, sound, programming, video editing κλπ.

2. Unity

Το Unity είναι ένα ολοκληρωμένο εργαλείο για δημιουργία 3D και 2D βιντεοπαιχνιδιών ή άλλου δια δραστικού περιεχομένου όπως αρχιτεκτονικές μοντελοποίησης ή 3D animation πραγματικού χρόνου. Η φιλοσοφία της είναι ότι ένα ολοκληρωμένο γραφικό περιβάλλον πρέπει να είναι το κύριο μέσο ανάπτυξης των βιντεοπαιχνιδιών και ελαχιστοποιεί τη χρήση προγραμματισμού μόνο στη συμπεριφορά των αντικειμένων του κόσμου του παιχνιδιού. Η μηχανή τρέχει σε Windows και Mac OS και τα παιχνίδια που δημιουργεί είναι για Windows, Mac OS, Nintendo Wii, iPad και iPhone ενώ αναμένεται και υποστήριξη για Android. Τα παιχνίδια της μπορούν ακόμα να παίξουν σε browser με το Unity web player plug-in, που υποστηρίζουν τα Windows και Mac OS ενώ πλέον έχει προστεθεί υποστήριξη για τα Microsoft Xbox360 Xbox One και Sony Playstation 3 και 4.

Εικόνα 2 από Unity (logo)

Αρχικά, η Unity κυκλοφόρησε για λειτουργικά συστήματα “Mac” παράλληλα με την εμφάνισή της στο “Worldwide Developers Conference” με την “Apple” ως διοργανώτρια και μετέπειτα στόχευσε την επέκτασή της σε παραπάνω από δεκαπέντε πλατφόρμες. Φυσικά, αξίζει να σημειωθεί η δυνατότητα που προσφέρει η συγκεκριμένη μηχανή στον εκάστοτε “developer” να δημοσιεύει το παιχνίδι του σε οποιουδήποτε είδους πλατφόρμα επιθυμεί ο ίδιος. Εμείς καταλήξαμε στο Unity3d επειδή έχει έκδοση Free, έχει ένα Forum Community υποστήριξης forum.unity3d.com όπου μπορείς να βρεις λύση σε ό,τι δυσκολία συναντήσεις, και τέλος έχει ένα Asset- Store το οποίο σου παρέχει μερικά γραφικά μοντέλα - scripting code δωρεάν.

3. Βιντεοπαιχνίδι Δράσης

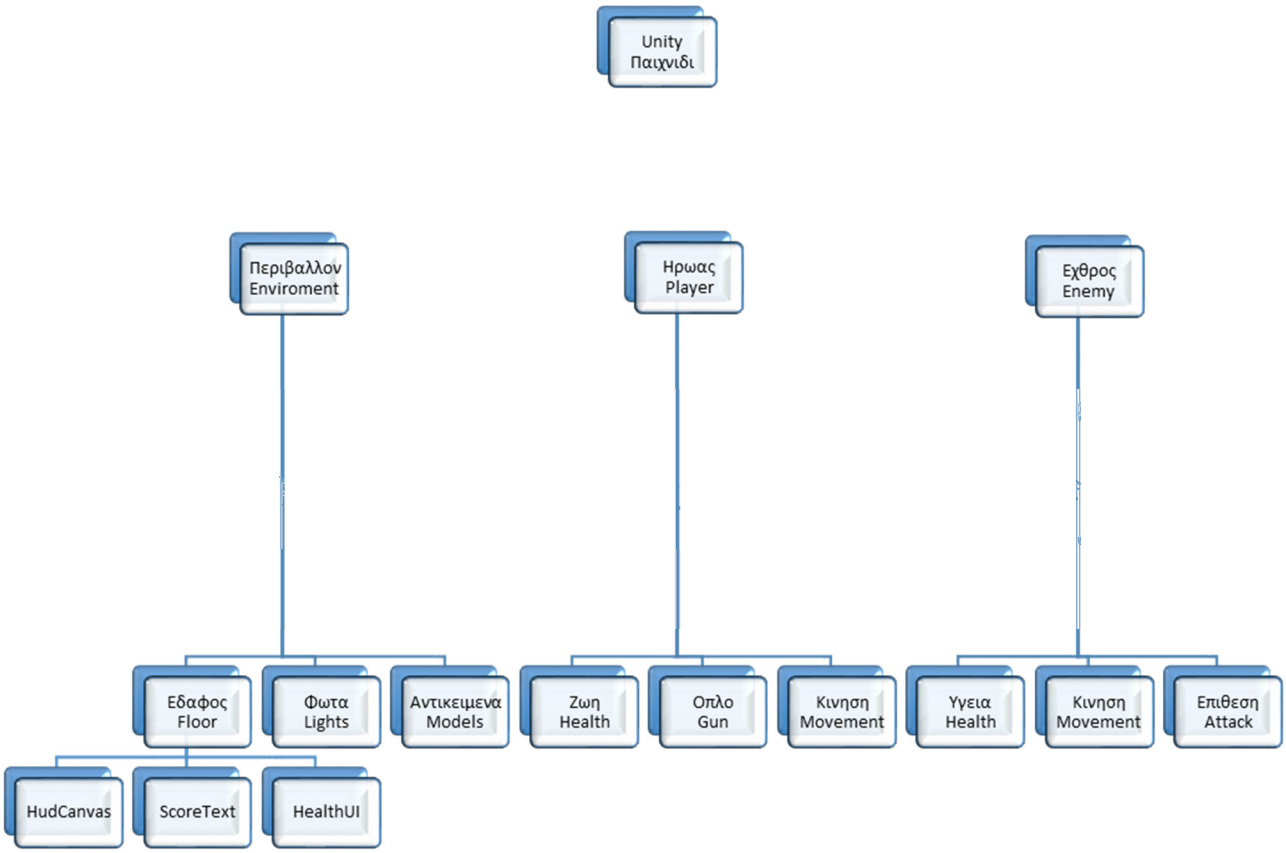
Ένα παιχνίδι δράσης αποτελείται από αποστολές. Θέτονται αποστολές που πρέπει να επιτελεστούν, και έτσι αποκτούνται χρήματα ή ξεκλειδώνονται άλλες αποστολές ή διάφορα αντικείμενα που είναι βασικά για τον πρωταγωνιστή. Ακόμη, στα περισσότερα παιχνίδια υπάρχουν όπλα που χρειάζονται στις επόμενες αποστολές. Τέλος, στα αριστερά η στα δεξιά της οθόνης είτε του υπολογιστή είτε της τηλεόρασης, υπάρχει μια μπάρα ζωής που δείχνει σε τι κατάσταση βρίσκεται ο παίχτης.

Όπως και στο δικό μας παιχνίδι δράσης το οποίο είναι κατασκευασμένο στην πλατφόρμα της Unity με την βοήθεια των Tutorials, υπάρχει ένας βασικός χαρακτήρας, ο ήρωας του παιχνιδιού, και τρεις διαφορετικοί αντίπαλοι. Το σενάριο του παιχνιδιού εξελίσσεται στο δωμάτιο του ήρωα όπου τα γραφικά του είναι σχεδιασμένα στο Photoshop και του επιτίθονται τρεις εχθροί από διαφορετικά σημεία του δωματίου. Σκοπός του παιχνιδιού είναι να σκοτώσεις όσους περισσότερους εχθρούς μπορέσεις.

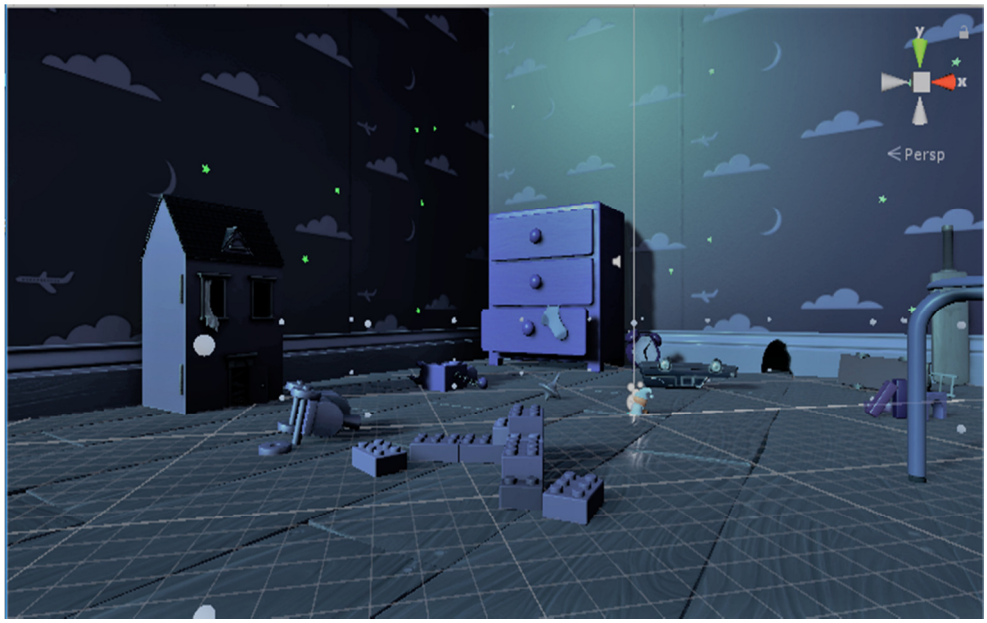
Με την ευκαιρία αυτή πήραμε από την βιβλιοθήκη της unity το παιχνίδι Survival Shooter και το αναβαθμίσαμε βάζοντας κάποια επιπλέον στοιχεία όπως έναν Timer που μετρά τον χρόνο κατά τον οποίο ο ήρωας μένει ζωντανός, και ένα EmeniesKilled που μετράει τους εχθρούς που σκοτώνεις .

Εικόνα 3 Survival Shooter (Unity)

4. Γραφική παράσταση υλοποίησης παιχνιδιού στην Unity.

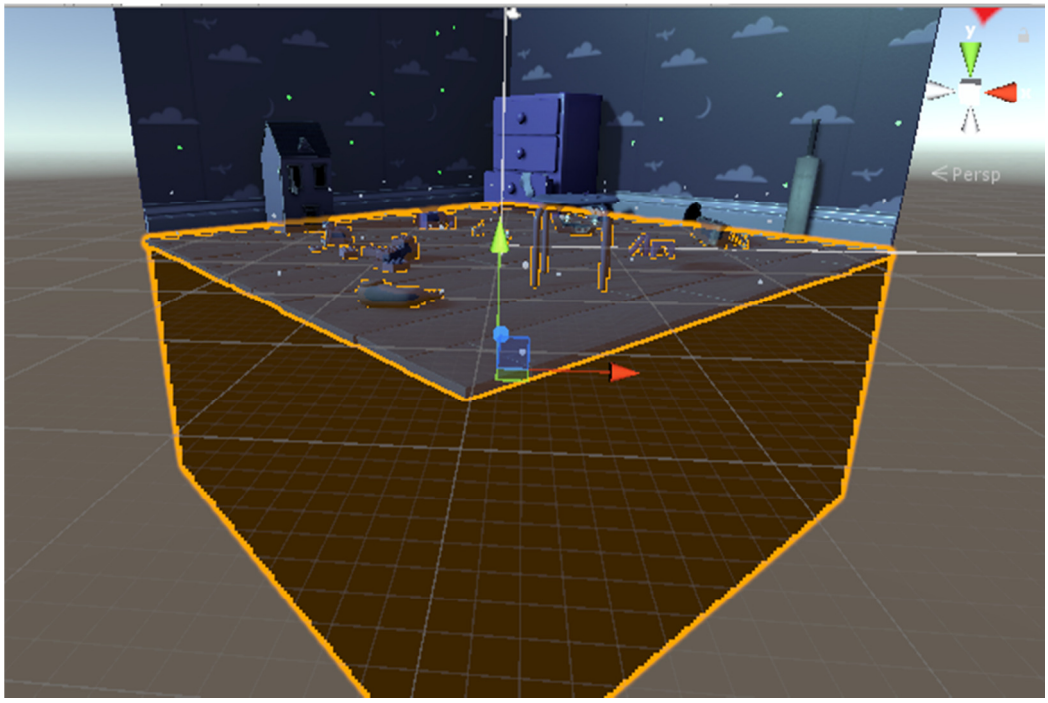


- Περιβάλλον
  (Enviroment) : Είναι το βασικό χαρακτηριστικό του παιχνιδιού το οποίο περιέχει όλα τα αντικείμενα του δωμάτιου. Πιο αναλυτικά το δωμάτιο είναι γεμάτο με διάφορα μικρά και μεγάλα αντικείμενα έτσι ώστε να μπορεί ο ήρωας μας να κρύβεται σ'αυτά από τους εχθρούς.



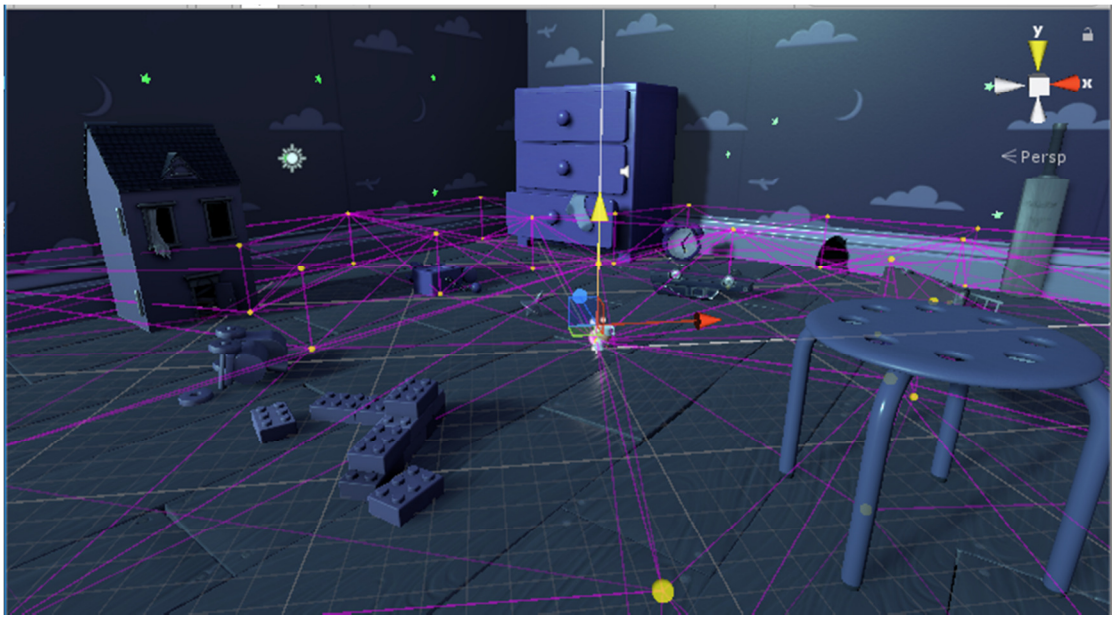
Εικόνα 4 Περιβάλλον παιχνιδιού Unity

- Έδαφος
  (Floor) : Χρειαζόμαστε το έδαφος για να σταθούν όλα τα αντικείμενα και βεβαίως οι χαρακτήρες πάνω στο χώρο. Έχουμε βάλει στο έδαφος μας και ένα component έτσι ώστε το πάτωμα μας να είναι συμπαγές και να μην πέφτουν τα αντικείμενα και οι χαρακτήρες μας στο κενό.



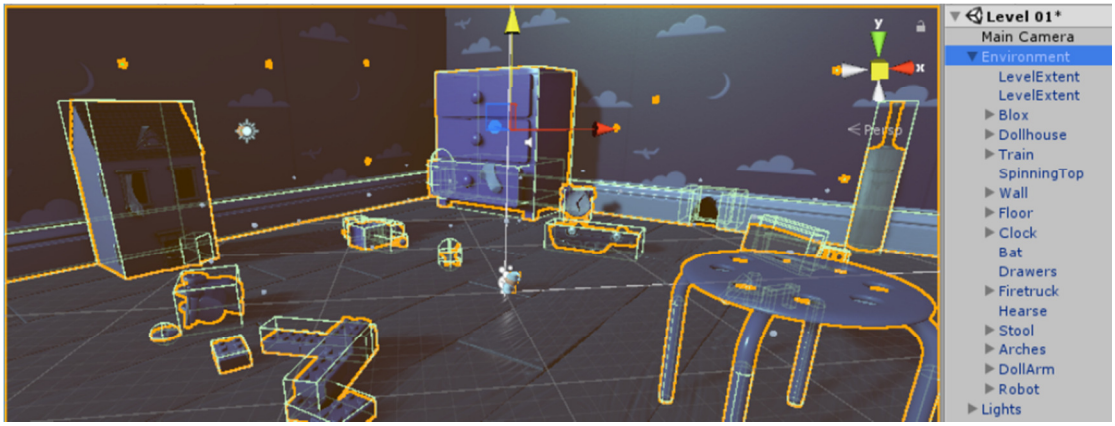
Εικόνα 5 Έδαφος παιχνιδιού Unity

- Φώτα
  (Lights) : Τα φώτα είναι ένα πολύ σημαντικό σημείο γιατί χωρίς αυτά δεν θα μπορούσαμε να δούμε για να παίξουμε ενώ βάζοντας φωτισμό στον χώρο έχουμε ένα πολύ διαφορετικό αποτέλεσμα, με σκιές και ωραίες λεπτομέρειες.



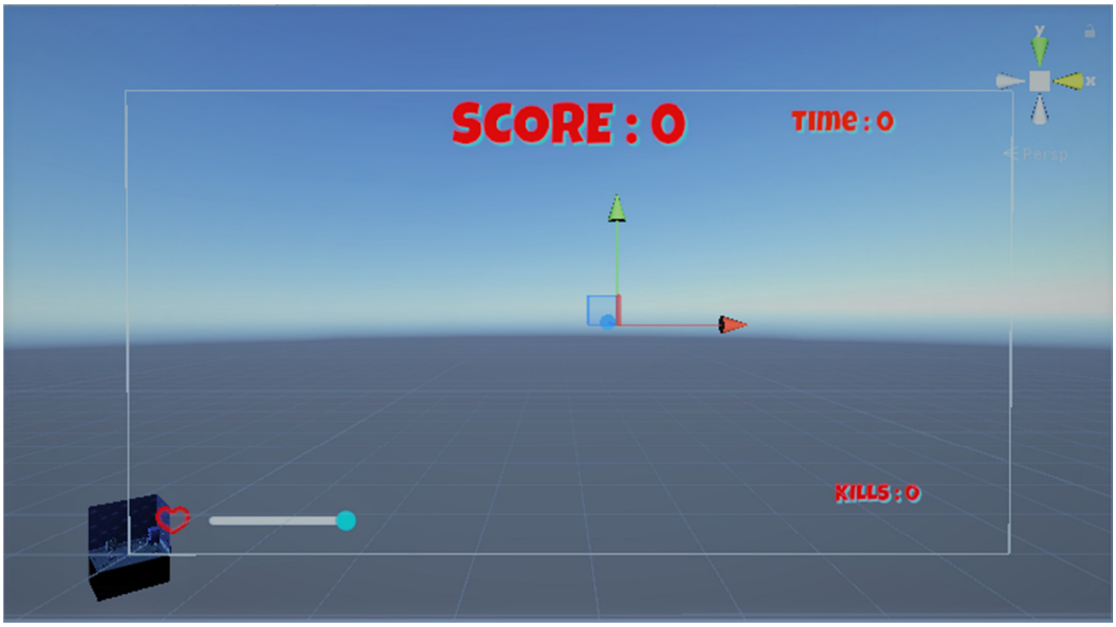
Εικόνα 6 Φώτα στο περιβάλλον Unity

- Αντικείμενα
  (Models) : Υπάρχουν δυο κατηγορίες αντικειμένων , πρώτη είναι οι χαρακτήρες που αποτελούνται από εχθρούς ζόμπι ελέφαντες ζόμπι αρκουδάκια και ζόμπι λαγουδάκια και τον ηρώα μας τον player. Στην δεύτερη κατηγορία ανήκουν τα αντικείμενα του χώρου όπως ένα σκαμπό, μια σβούρα, ένα αυτοκίνητο, ένα κουκλόσπιτο και πάρα πολλά άλλα.



Εικόνα 7 Αντικείμενα περιβάλλοντος Unity

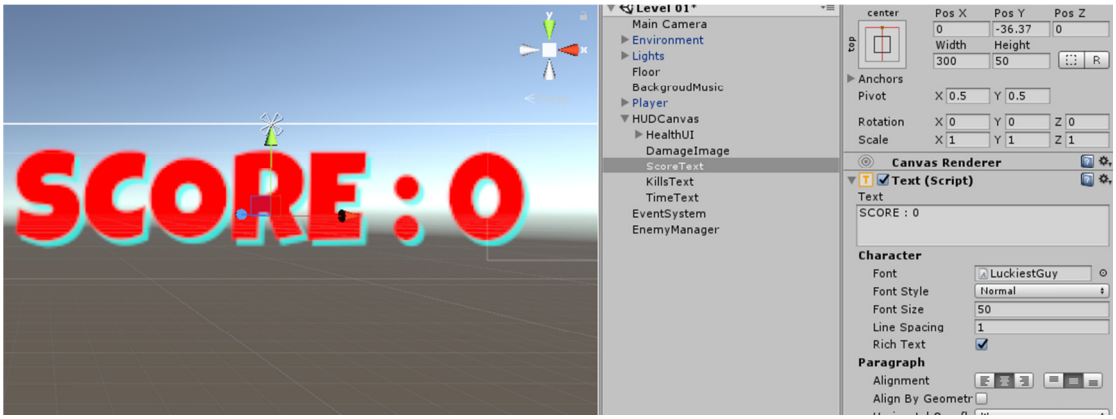
- Καμβάς
  (Canvas) : ο καμβάς είναι το σημαντικότερο γραφικό κομμάτι του παιχνιδιού γιατί πάνω σε αυτόν γράφουμε το τι ακριβώς θέλουμε να φαίνεται στην οθόνη μας την ώρα που παίζουμε. Για παράδειγμα όταν σκοτώνουμε έναν εχθρό μετράμε το score για να μπορέσουμε να δούμε το score στην οθόνη μας πρέπει να το γράψουμε στον καμβά μας όπως και όλα τα υπόλοιπα στοιχεία, τον timer μας ή ακόμα και τη ζωή του ηρώα μας.



Εικόνα 8 Καμβάς στατιστικών Unity

- Σκορ

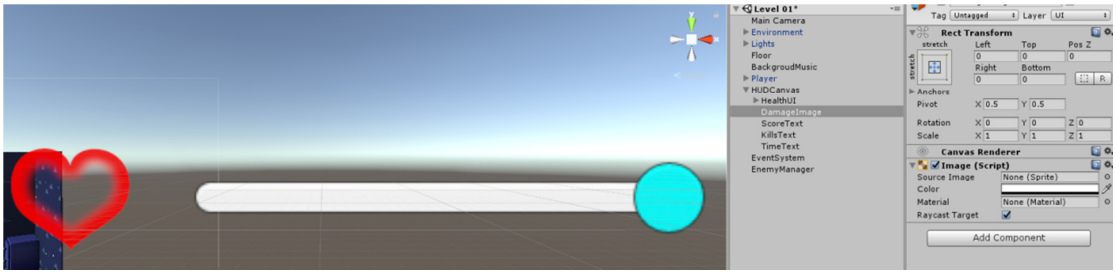
  (ScoreText) : Είναι το γραφικό που φαίνεται στον καμβά και μας δείχνει το συνολικό σκορ από τους εχθρούς που έχουμε σκοτώσει. Για παράδειγμα όταν σκοτώνουμε έναν ζόμπι ελέφαντα παίρνουμε 30 πόντους και όταν ένα ζόμπι αρκουδάκι 10 πόντους. Έτσι με το ScoreText βλέπουμε Live στην οθόνη το συνολικό σκορ μας.



Εικόνα 9 Σκορ και ιδιότητες Unity

- Ζωή

  (Health) : Όπως είναι λογικό δεν θα μπορούσαμε να ξεκινήσουμε να παίζουμε χωρίς να έχουμε δώσει ζωή στον ηρώα μας αλλά και στους εχθρούς μας. Έτσι δώσαμε ένα ποσοστό 100% στον ηρώα μας όταν ξεκινάει το παιχνίδι και κάθε φορά που μας ακουμπάει ένας εχθρός χάνουμε ένα 10% . Αντίστοιχα όταν κάνουμε επίθεση ο εχθρός με κάθε μας βολή χάνει 20% από την αρχική του ζωή που είναι επίσης 100%.

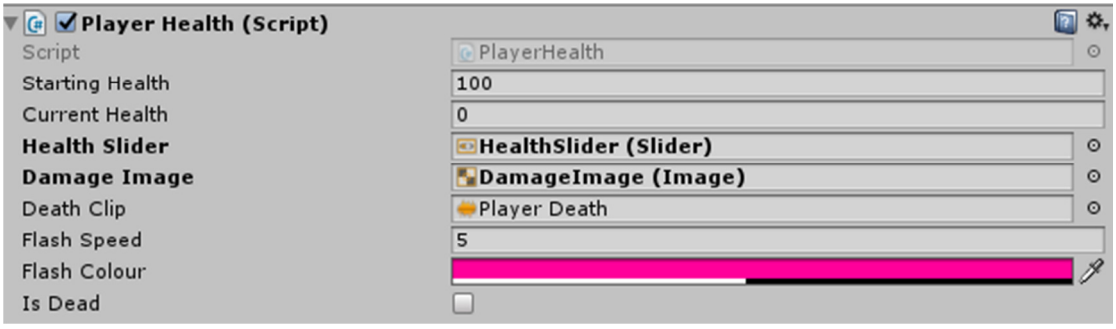


Εικόνα 10 Εικόνα καρδιάς και μπάρα ζωής Unity

- Ήρωας
  (Player) : Ο βασικός μας χαρακτήρας είναι φτιαγμένος από τους ειδικούς της unity στο Photoshop και μας δίνουν ένα καταπληκτικό αποτέλεσμα. Αυτό που εμείς κάναμε είναι να δώσουμε σώμα στον ηρώα μας να του ενσωματώσουμε κάποια scripts έτσι ώστε να έχει ζωή και να μπορεί να επιτεθεί στους εχθρούς μας. Επίσης για να μπορεί ο ήρωας μας να κάνει damage στους εχθρούς μας του έχουμε δώσει ένα όπλο και με το αντίστοιχο script μπορεί να ρίξει και να κάνει ζημιά στην ζωή του εχθρού.

Εικόνα 11 Βασικός χαρακτήρας παιχνιδιού Unity

- Ζωή
  (Health Player) : Για την ζωή του ηρώα μας χρησιμοποιήσαμε το έτοιμο script της unity με κάποιες μικρές αλλαγές που κάναμε. Το συγκεκριμένο script συνδέεται και με το script που έχει ο εχθρός μας έτσι ώστε να μας κατεβάζει την ζωή κάθε φορά που μας ακουμπάει.



Εικόνα 12 Ιδιότητες ζωής Unity

- Όπλο

  (Gun) : Το όπλο το οποίο κρατάει ο ήρωας μας και με το οποίο επιτίθεται στους εχθρούς μας έχει επάνω ένα script το οποίο του επιτρέπει όταν η βολή φύγει από το όπλο και ακουμπήσει τον εχθρό να κάνει ένα damage στην ζωή του 20%. Έχουμε επίσης την επιλογή να διαλέξουμε την ακτίνα που βγάζει από το όπλο μας σε ότι χρώμα και σχήμα θέλουμε εμείς.

Εικόνα 13 Όπλο Unity

- Εχθρός

  (Enemy) : Όσο σημαντικός είναι ο ήρωας στο παιχνίδι μας άλλο τόσο είναι και ο εχθρός τον οποίο κυνηγάμε να σκοτώσουμε. Έχουμε τρεις διαφορετικούς εχθρούς με σχεδόν τα ίδια χαρακτηριστικά όπως η ζωή και η κίνηση τους, το μόνο που αλλάζει είναι το score κάθε εχθρού.

Εικόνα 14 Βασικος εχθρος (zombunny)

- Ζωή
  (Health Enemy) : Για την ζωή του εχθρού μας χρησιμοποιήσαμε το έτοιμο script της unity με κάποιες μικρές αλλαγές που κάναμε. Το συγκεκριμένο script συνδέεται και με το script που έχει ο ήρωας μας έτσι ώστε να του κατεβάζει το ποσοστό της ζωής του κάθε φορά που τον πετυχαίνει με το όπλο του.

```
public int startingHealth = 100;

public int currentHealth;

public float sinkSpeed = 2.5f;

public int scoreValue = 10;

public AudioClip deathClip;
```

- Κίνηση
  (Movement) : Η κίνηση του εχθρού μας είναι μια πιο περιπλοκή διαδικασία γιατί δεν τον κινούμε με τα πλήκτρα όπως κάνουμε με τον ηρώα μας. Ο εχθρός μας για να καταφέρει να μας κατεβάσει την ζωή θα πρέπει να μας ακουμπήσει , αυτό σημαίνει ότι πρέπει να μας βρει μέσα στο χώρο. Αυτό το καταφέραμε βάζοντας ένα Navigation Mesh Agent δηλαδή με αυτό μπορεί να βρίσκει πού είναι ο ήρωας μας μέσα στο χώρο και ο εχθρός να μας ακολουθεί με βάση τις κινήσεις του ήρωας. Δηλαδή όπου κινείται ο ήρωας μέσα στον χώρο ο εχθρός μας ακολουθεί πάντα χωρίς να μας χάσει ποτέ, συνεπώς με αυτόν τον τρόπο έχουμε δώσει κίνηση στον εχθρό χωρίς να χρειάζεται να τον κινούμε .

```
Transform player;

PlayerHealth playerHealth;

EnemyHealth enemyHealth;

UnityEngine.AI.NavMeshAgent nav;

void Update ()

    {

        if(enemyHealth.currentHealth > 0 &&
playerHealth.currentHealth > 0)

        {

            nav.SetDestination (player.position);

        }

        else

        {

            nav.enabled = false;

        }
```

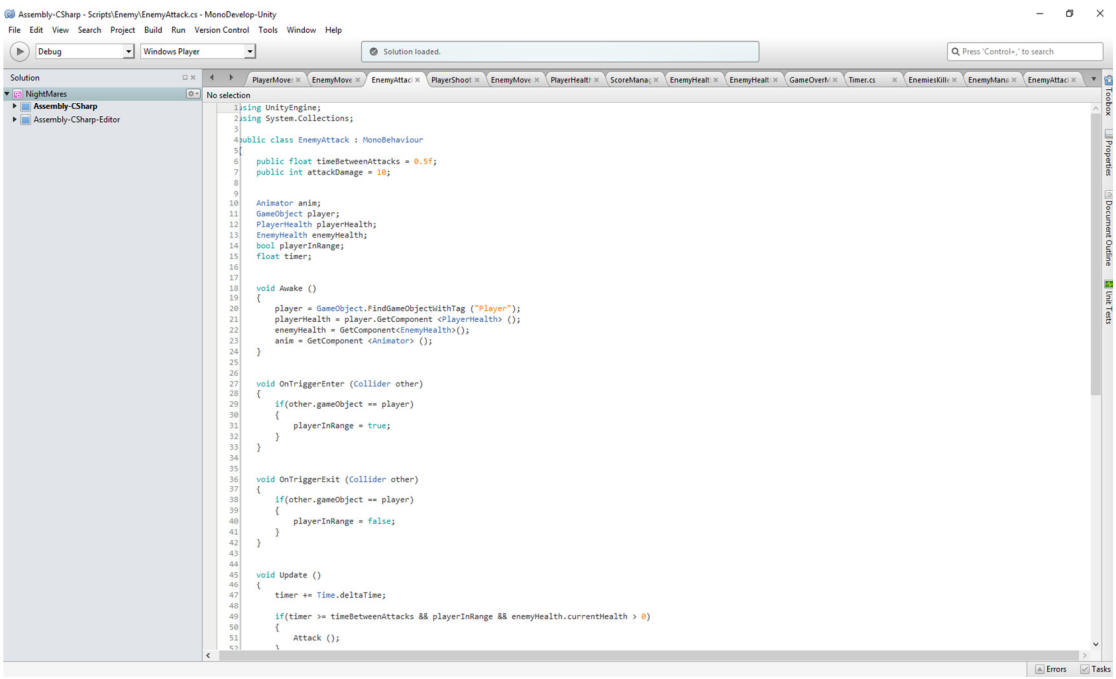
- Επίθεση
  (Attack) : Για να επιτεθεί ο εχθρός και να καταφέρει να κάνει damage στον ηρώα του παιχνιδιού θα πρέπει να τον ακουμπήσει. Όταν το καταφέρει ενεργοποιεί ένα script το οποίο είναι συνδεδεμένο με την ζωή του ήρωα και του κατεβάζει το ποσοστό κατά 10% με κάθε ακούμπημα.

```
Animator anim;

GameObject player;

PlayerHealth playerHealth;

EnemyHealth enemyHealth;

bool playerInRange;

float timer;

void Awake ()
```

```
void Attack ()
    {
  timer = 0f;

  if(playerHealth.currentHealth > 0)
      {
       playerHealth.TakeDamage (attackDamage);
       }
```

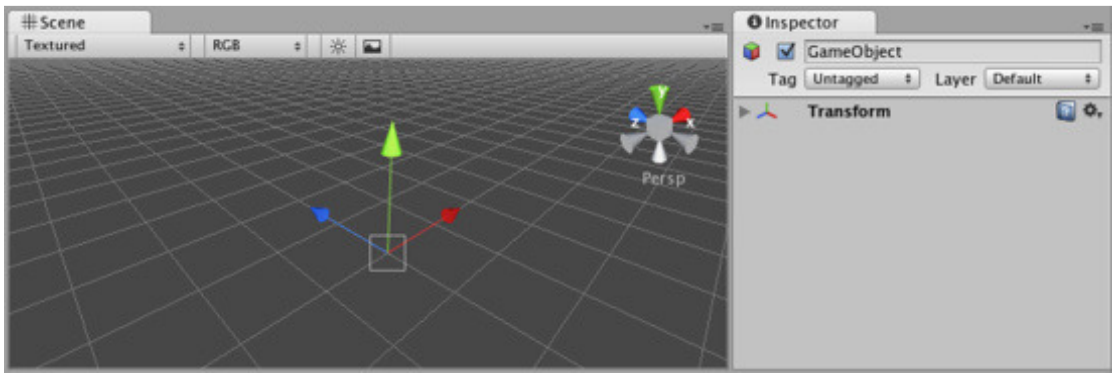
5. Ανάπτυξη 3D παιχνιδιού με χρήση της πλατφόρμας Unity

Η δημιουργία ενός 3D παιχνιδιού με χρήση του Unity αποτελείται από δύο κύριους άξονες: Πρώτον το κομμάτι του γραφικού σχεδιασμού όπου το Unity προσφέρει μια πληθώρα εργαλείων, με τα οποία ο δημιουργός του παιχνιδιού μπορεί να κατασκευάσει τα αντικείμενα που επιθυμεί να περιέχονται στο παιχνίδι και να τα τοποθετήσει στο σημείο της σκηνής που επιθυμεί. Δεύτερον, το κομμάτι του κώδικα που αποτελείται από διάφορα scripts, δηλαδή αρχεία κώδικα τα οποία σχετίζονται με κάποια συγκεκριμένα αντικείμενα. Τα scripts προσδίδουν κάποια καθορισμένα χαρακτηριστικά σε αυτά τα αντικείμενα και ορίζουν τη συμπεριφορά τους σε συγκεκριμένες ενέργειες του χρήστη, όπως το πάτημά ενός πλήκτρου. Αρχικά, το παιχνίδι αποτελείται από μία σκηνή. Όμως, ο δημιουργός του παιχνιδιού, μπορεί να κατασκευάσει πολλές διαφορετικές σκηνές, με διαφορετικά ή και με τα ίδια αντικείμενα. Οι σκηνές αυτές θα εναλλάσσονται μεταξύ τους και συνήθως αυτή η εναλλαγή πραγματοποιείται με κάποιο προγραμματισμένο event, όπως trigger. Το Unity είναι σχεδιασμένο να δέχεται αρχεία κώδικα σε C# ή Javascript. Στην Pro version παρέχεται και η δυνατότητα χρήσης C++ κώδικα. Το Unity είναι μία πλατφόρμα που βασίζεται σε συγγενικές στην C++ γλώσσες. Μπορείτε να γράψετε κώδικα σε C#, JavaScript ή , λιγότερο συχνά, σε Boo. Ο δικός σας κώδικας (και όχι το Unity Engine Code) τρέχει σε Mono ή Microsoft. Για να επεξεργαστείτε τον κώδικά σας στο Unity ανοίγετε το προεπιλεγμένο πρόγραμμα επεξεργασίας cross-platform MonoDevelop ή εάν προτιμάτε μπορείτε να ρυθμίσετε το Visual Studio ως συντάκτης. Όμως εάν χρησιμοποιήσετε το Visual Studio, θα πρέπει να χρησιμοποιήσετε το plug-in UnityVS για τον εντοπισμό σφαλμάτων. Από τη άλλη, στο MonoDevelop ο έλεγχος σφαλμάτων (debug) γίνεται αυτόματα καθώς έχει προεγκατεστημένο ένα plug-in που ανοίγει μία σύνδεση με το πρόγραμμα ελέγχου σφαλμάτων του Unity



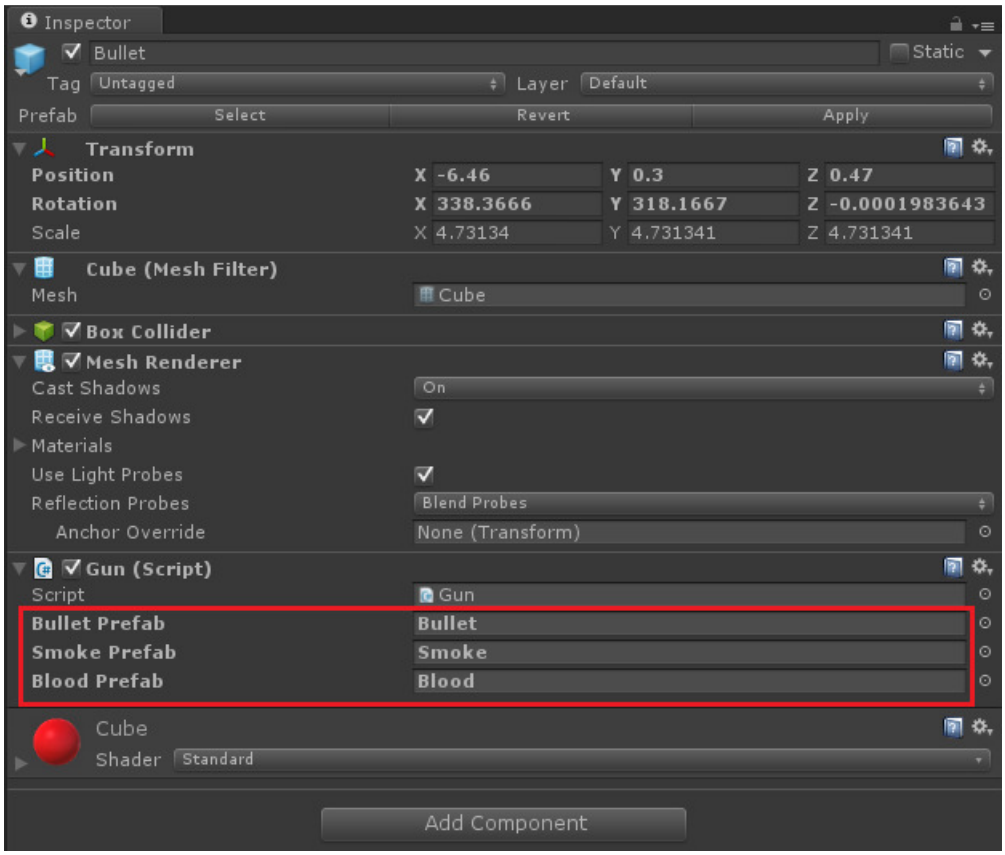
Εικόνα 15 Παράδειγμα script σε C#

Οτιδήποτε τρέχει στο παιχνίδι σας υπάρχει σε μια σκηνή. Όταν συσκευάζετε το παιχνίδι σας για μια πλατφόρμα, το παιχνίδι που προκύπτει είναι μια συλλογή από μία ή περισσότερες σκηνές, συν τον κώδικα που έχετε. Μπορείτε να έχετε όσες σκηνές θέλετε σε ένα Project. Μια σκηνή μπορεί να θεωρηθεί ως ένα επίπεδο σε ένα παιχνίδι, αν και μπορείτε να έχετε πολλαπλά επίπεδα σε μία σκηνή απλά μετακινώντας τον παίκτη / κάμερα σε διαφορετικά σημεία στη σκηνή. Όταν κάνετε λήψη πακέτων τρίτων ή ακόμα δείγματα παιχνιδιού από το Asset Store, θα πρέπει να αναζητήσετε τα αρχεία σκηνής (scene files) στο έργο σας και να τα ανοίξετε. Ένα αρχείο σκηνή είναι ένα μοναδικό αρχείο που περιέχει όλα τα είδη των αρχείων που χρησιμοποιούνται στο έργο για την τρέχουσα σκηνή. Είναι σημαντικό να αποθηκεύετε συχνά μια σκηνή πατώντας Ctrl + S κατά τη διάρκεια της ανάπτυξης, ακριβώς όπως και με κάθε άλλο εργαλείο. Συνήθως, το Unity ανοίγει την τελευταία σκηνή με την οποία έχετε εργαστεί, αν και μερικές φορές όταν ανοίγετε ένα Project με το Unity, αυτό δημιουργεί αυτόματα μία νέα άδεια σκηνή, και θα πρέπει να πάτε να βρείτε τη σκηνή την οποία επιθυμείτε στον εξερευνητή του project σας. Αυτό μπορεί να είναι αρκετά ενοχλητικό για τους νέους χρήστες, αλλά είναι σημαντικό να το θυμόμαστε, αν τύχει να ανοίξετε το τελευταίο έργο σας και αναρωτηθείτε πού πήγε το Project σας! Θα βρείτε την δουλειά σας σε ένα αρχείο σκηνή (scene file) που έχετε αποθηκεύσει στο project σας. Σε μια σκηνή, δεν μπορείτε να δείτε τίποτα χωρίς ένα αντικείμενο κάμερας και δεν μπορείτε να ακούσετε τίποτα χωρίς ένα Audio Listener εφαρμοσμένο πάνω σε κάποιο GameObject . Σημειώνεται, ωστόσο, ότι σε κάθε νέα σκηνή, το Unity δημιουργεί πάντα μια κάμερα που διαθέτει Audio Listener πάνω της. Σχεδόν τα πάντα στη σκηνή σας είναι ένα GameObject. Είναι η βασική κλάση για όλα τα αντικείμενα σε μία σκηνή του Unity. Όλα τα αντικείμενα απορρέουν από ένα GameObject. Ένα GameObject είναι αρκετά απλό, δεδομένου ότι εντάσσεται στο παράθυρο Inspector. Μπορείτε να δείτε στην εικόνα 2.5 ότι ένα κενό GameObject προστέθηκε στη σκηνή. Τα GameObjects by default δεν έχουν οπτικές ιδιότητες, εκτός από το widget που σας δείχνει το Unity όταν επισημαίνετε το αντικείμενο. Σε αυτό το σημείο, είναι απλά ένα άδειο αντικείμενο.



Εικόνα 16 Ένα κενό Game Object

Ένα GameObject έχει ένα όνομα, μια ετικέτα (Tag), ένα Layer και το Transform (ίσως η πιο σημαντική ιδιότητα του συνόλου). Το Transform είναι στην ουσία η θέση, η περιστροφή και η κλίμακα του κάθε GameObject. Το Unity χρησιμοποιεί ως συντεταγμένες το X (οριζόντια), Y (κάθετη) και Z συντεταγμένη (το βάθος, δηλαδή, με φορά σαν μπαίνει ή βγαίνει από την οθόνη). Στην ανάπτυξη παιχνιδιών, είναι αρκετά κοινό να χρησιμοποιούνται οι άξονες τριών διαστάσεων. Τα Transform.Position και Transform.Scale είναι και τα δύο αντικείμενα Vector3. Ένα Vector3 είναι απλά ένα διάνυσμα τριών διαστάσεων. Με άλλα λόγια , δεν είναι τίποτα περισσότερο από τρία σημεία - X , Y και Z. Μέσα από αυτές τις τρεις απλές τιμές, μπορείτε να ορίσετε την τοποθεσία ενός αντικειμένου, ακόμα και να μετακινήσετε ένα αντικείμενο προς την κατεύθυνση ενός άξονα. Το να γίνει λειτουργικό ένα GameObject, γίνεται μέσω της πρόσθεσης διαφόρων στοιχείων (Components). Οτιδήποτε προστεθεί είναι ένα Component και εμφανίζονται όλα στο παράθυρο Inspector. Υπάρχουν MeshRender και SpriteRender Components, Components για τον ήχο και την λειτουργικότητα της κάμερας, Components που σχετίζονται με την Φυσική (Colliders και Rigidbodies) και πολλά ακόμα. Για να εκχωρήσετε κάποιον κώδικα σε ένα GameObject χρησιμοποιείτε το script Component. Τα Components είναι αυτά που ζωντανεύουν ένα GameObject και του προσδίδουν οποιαδήποτε λειτουργικότητα.



Εικόνα 17 Ένα GameObject που του έχουν προστεθεί Components

Το Unity έρχεται με μία ενσωματωμένη μηχανή φυσικής που επιτρέπει να οριστούν οι φυσικές ιδιότητες των αντικειμένων και αφήνει τις λεπτομέρειες της προσομοίωσης στον χρήστη. Γενικά αντί να προσπαθήσει κάποιος να “εφαρμόσει” την δική του φυσική, είναι απλούστερο και καλύτερο να χρησιμοποιήσει την μηχανή φυσικής του Unity όσο περισσότερο μπορεί. Συχνά, κατά την δημιουργία ενός παιχνιδιού, θα θέλατε μια σύγκρουση δύο αντικειμένων να οδηγήσει σε κάποια αλλαγή κατάστασης στο κώδικα. Για να επιτευχθεί κάτι τέτοιο χρησιμοποιείται η μέθοδος ανίχνευσης σύγκρουσης (Collision Detection Method). Η ανίχνευση αυτών των συγκρούσεων απαιτεί μια δόση δουλειάς για να επιτευχθεί στο Unity. Αρχικά ένα τουλάχιστον από τα αντικείμενα στην σύγκρουση χρειάζεται να έχει ένα non-kinematic Rigidbody, ενώ και τα δύο αντικείμενα πρέπει να έχουν σωστούς Colliders αρχικοποιημένα σωστά. Η συνολική ταχύτητα και των δύο πρέπει να είναι αρκετά μικρη ώστε να υπάρχει εντοπισμός της σύγκρουσης και να μην περνάει το ένα μέσα από το άλλο χωρίς να εντοπίζεται η σύγκρουση. Αφού ελεγχθούν όλα αυτά, υπάρχει μία ειδική μέθοδος για ανίχνευση της σύγκρουσης μέσω ενός script που επισυνάπτεται στο αντικείμενο με το οποίο θα θέλατε να ελέγξετε την σύγκρουση. Τέτοιες συναρτήσεις εντοπισμού φαίνονται στην εικόνα 2.8. Σημειώνεται οτι γίνεται αυτή η εκτενής αναφορά στα Collisions, καθώς αποτελούν σημαντικό κομμάτι του παρόντος Project.

Όσον αφορά τον ήχο, το Unity υποστηρίζει 2D και 3D ήχους. Οι 3D ήχοι αλλάζουν ένταση ανάλογα με την απόσταση, και αλλάζουν ανάλογα με την κίνηση τους σε σχέση με την κάμερα. Οι 2D ήχοι από την άλλη είναι πιο κατάλληλοι για χρήση ως μμουσική παρασκηνίου καθώς κρατούν μια ομοιόμορφη ένταση. Για ήχους τώρα που παράγονται από γεγονότα χρησιμοποιούνται οι 3D ήχοι. Υπάρχουν ακόμα πάρα πολλά στοιχεία στην ανάπτυξη ενός παιχνιδιού μέσω Unity, όμως εδώ έγινε μία μικρή καταγραφή κάποιων βασικών συστατικών έτσι ώστε να υπάρχει μία ιδέα του περί τίνος πρόκειται.

6. Περιγραφή Project

Η ιδέα πίσω από το συγκεκριμένο Project είναι η δημιουργία μιας εξελιγμένης μορφής του παιχνιδιού Survival Shooter δηλαδή ένα extended όπου ο χρήστης θα μπορεί την ώρα που θα παίζει το παιχνίδι να βλέπει πολλές περισσότερες πληροφορίες και στατιστικά για τους εχθρούς που σκοτώνει καθώς και για τον χρόνο εκτέλεσης.

Έχουμε δημιουργήσει το παιχνίδι Survival Shooter από την αρχή με την σωστή και λεπτομερή παρακολούθηση Tutorial μέσω της Unity και έχουμε διαφοροποιήσει σχεδόν όλες τις παραμέτρους βάζοντας δικά μας χρώματα , διαστάσεις και γραμματοσειρές έτσι ώστε να το κάνουμε λίγο πιο ξεχωριστό.

Επίσης έχουμε φτιάξει από την αρχή δικά μας scripts σε γλώσσα C# για το χρόνο (timer) έτσι ώστε να γνωρίζει ο κάθε παίχτης πόση ώρα παίζει από την στιγμή που ξεκινάει έως την στιγμή που θα πεθάνει. Ένα ακόμα στατιστικό που έχουμε προσθέσει είναι τα Kills που κάνει ο ήρωας μας κατά την διάρκεια του παιχνιδιού, δηλαδή πόσους εχθρούς έχει σκοτώσει συνολικά.

6.1 Δημιουργία Περιβάλλοντος
Environment Setup

Εισαγωγή στο περιβάλλον του Unity

• Project (new. open, save)

• folder structure- ορίζει και θυμάται που βρίσκονται όλα τα αντικείμενα και τις συνδέσεις μεταξύ τους. (explorer)

– όταν ανοίγουμε ένα νέο project δημιουργείται αυτόματα μια δομή φακέλων και αρχείων.

Δεν πρέπει να αλλάζουμε την θέση των υλικών μας στον υπολογιστή.

• Assets – Import Asset

• Assets / Library (δημιουργεί ένα νέο αρχείο μέσα στο Assets folder)

• Οποιαδήποτε αλλαγή πρέπει να γίνεται μέσα στο UNITYProject folder (καθρέφτης του project folder στον explorer)

Αν πχ. αλλάξουμε το όνομα ενός αντικειμένου μέσα στο project (rename) κάνει update αυτόματα την library και όλες τις συνδέσεις. Αν κάνουμε το ίδιο μέσα στο assets folder το unity θα κάνει update αντίστοιχα στο περιβάλλον του unity.

• assets - scripts prefabs materials etc.

• create or import new assets-

• create new material (νέο αρχείο στον σκληρό μας δίσκο – όλα τα assets και οι scenes πρέπει να τροποποιούνται με την χρήση του project panel).

Project panel – Resize με την μπάρα από κάτω, κλικάροντας σε κάθε φάκελο βλέπουμε τα περιεχόμενα του.

Δημιουργία σκηνής
(Create New Scene)

Scene / Game view

Create New Scene Inspector

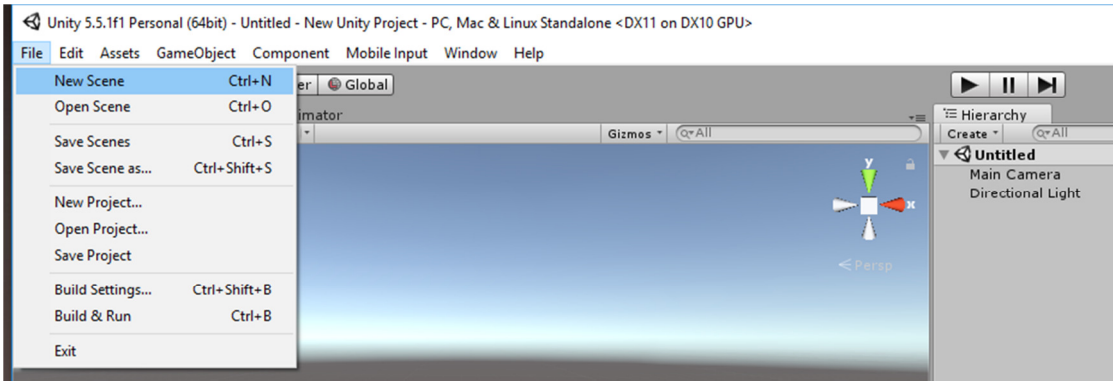
- Hierarchy – περιέχει όλα τα Game Object (& prefabs) στην current Scene

– Σχέσεις ιεραρχίας – ParenÉng (drag & drop) Scene

– περίπου όπως ένα Level- ή καλύτερα αυτόνομο μέρος του παιχνιδιού (φορτώνει τα αντικείμενα εκ νέου/ χωρισμός του παιχνιδιού σε σκηνές ακόμα και αν δεν έχει διαφορετικά επίπεδα)

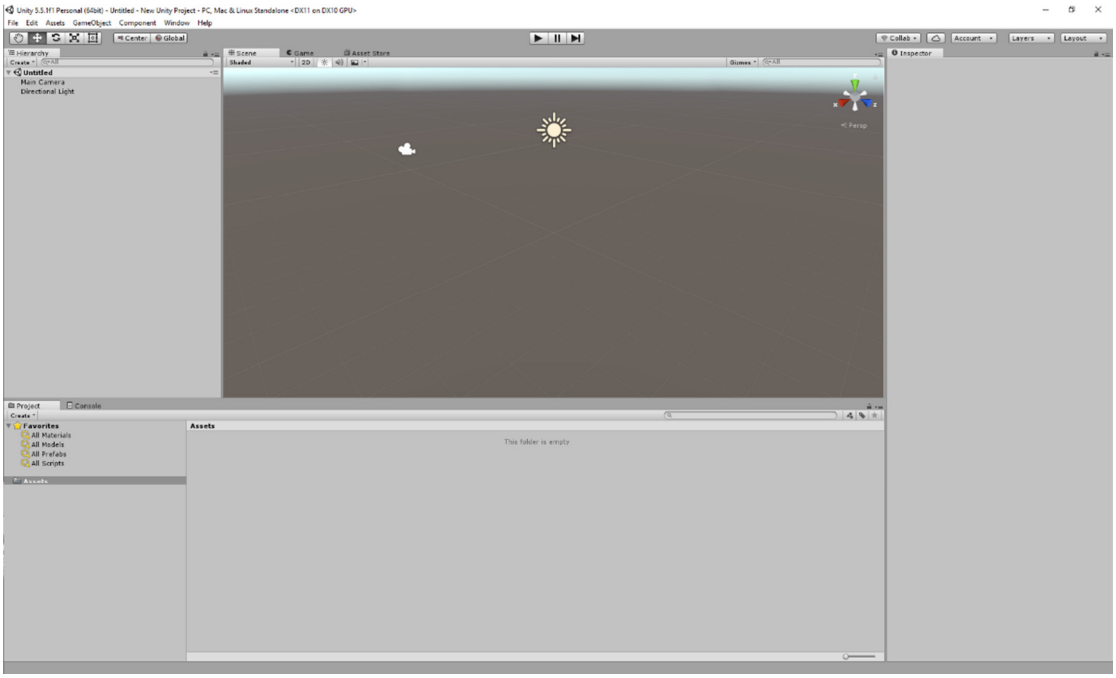
assets - scripts prefabs materials etc.

create or import new assets- create new material (νέο αρχείο στον σκληρό μας δίσκο – όλα τα assets και οι scenes πρέπει να τροποποιούνται με την χρήση του project panel).



Εικόνα 18 Παράδειγμα Νέας Σκηνής

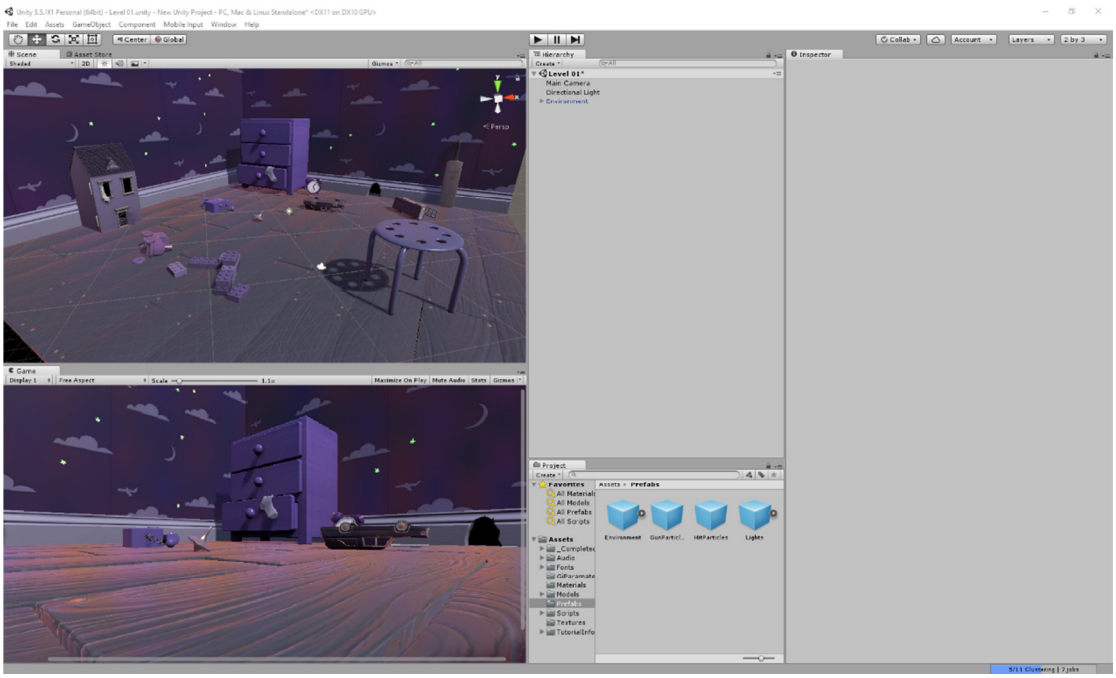
Η πρώτη σκηνή που θα συναντήσουμε φαίνεται στην παρακάτω εικόνα οπού θα δούμε πως ξεκινάμε από την αρχή.



Εικόνα 19 Νεα σκηνή Unity

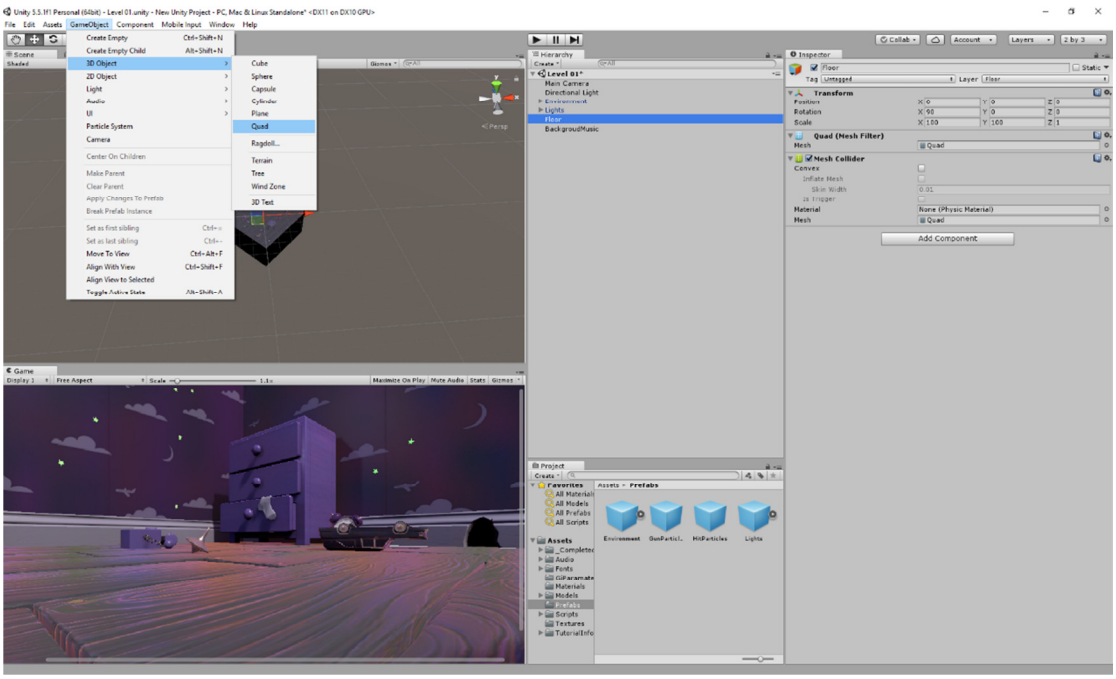
Στην συνέχεια κάνουμε import το παιχνίδι Survival Shooter στο οποίο θα δουλέψουμε και ακολουθούμε τα παρακάτω βήματα:

Βήμα Πρώτο από τον φάκελο Assets διαλέγουμε το Prefabs και στην συνέχεια παίρνουμε το Prefab Environment και το ρίχνουμε μέσα στην Hierarchy περιοχή έχοντας το παρακάτω αποτέλεσμα.



Εικόνα 20 Εισαγωγή περιβάλλοντος παιχνιδιού.

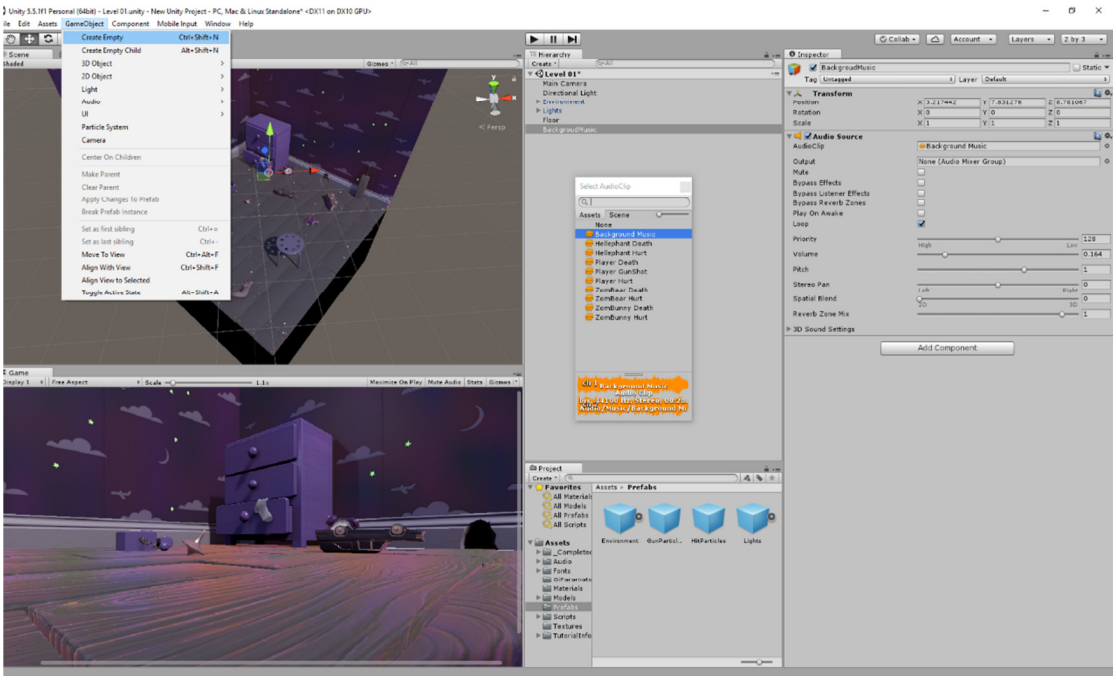
Αφού έχουμε καταφέρει και έχουμε δημιουργήσει το περιβάλλον στο οποίο θα δουλέψουμε πάμε να βάλουμε ένα πάτωμα έτσι ώστε οι χαρακτήρες μας να μπορούν να στέκονται πάνω σε αυτό. Για να δημιουργήσουμε ένα πάτωμα πάμε στο Game Object → 3D Object → Quad και στην συνέχεια μετονομάζουμε το Quad σε Floor και δίνουμε τις κατάλληλες διαστάσεις οι οποίες πρέπει να είναι λίγο μεγαλύτερες από το περιβάλλον μας. Στην παρακάτω φωτογραφία θα δούμε ακριβώς την διαδικασία.



Εικόνα 21 Δημιουργία κενού αντικείμενου

Τέλος για να κλείσουμε το κεφάλαιο αυτό θα πρέπει να δώσουμε στο περιβάλλον μας μια μουσική έτσι ώστε όταν μπαίνουμε να παίξουμε μέσα σε αυτό να ακούμε την κατάλληλη μουσική που θα έχουμε διαλέξει εμείς.

Για να το καταφέρουμε αυτό πρέπει να δημιουργήσουμε ένα κενό αντικείμενο(Game Object→Create Empty) και να το ονομάσουμε BackgroundMusic στο οποίο θα προσθέσουμε το κατάλληλο Component για να έχουμε τα αποτέλεσμα που θέλουμε. Στην παρακάτω φωτογραφία θα δούμε αναλυτικά τη διαδικασία.



Εικόνα 22 Δημιουργία μουσικής παιχνιδιού

Πινακας συντομευσεων-(Project panel SHORTCUTS)

• Tab Shift focus between first column and second column (Two columns)

• Ctrl/Cmd + F Focus search field

• Ctrl/Cmd + ASelect all visible items in list

• Ctrl/Cmd + DDuplicate selected assets

• Delete Delete with dialog • Delete + Shi| Delete without dialog

• Backspace + Cmd Delete without dialogs (OSX)

• Enter Begin rename selected (OSX)

• Cmd + down arrow Open selected assets (OSX)

• Cmd + up arrow Jump to parent folder (OSX, Two columns)

• F2 Begin rename selected (Win)

• Enter Open selected assets (Win)

• Backspace Jump to parent folder (Win, Two columns)

• Right arrow Expand selected item (tree views and search results). If the item is already expanded, this will select its first child item.

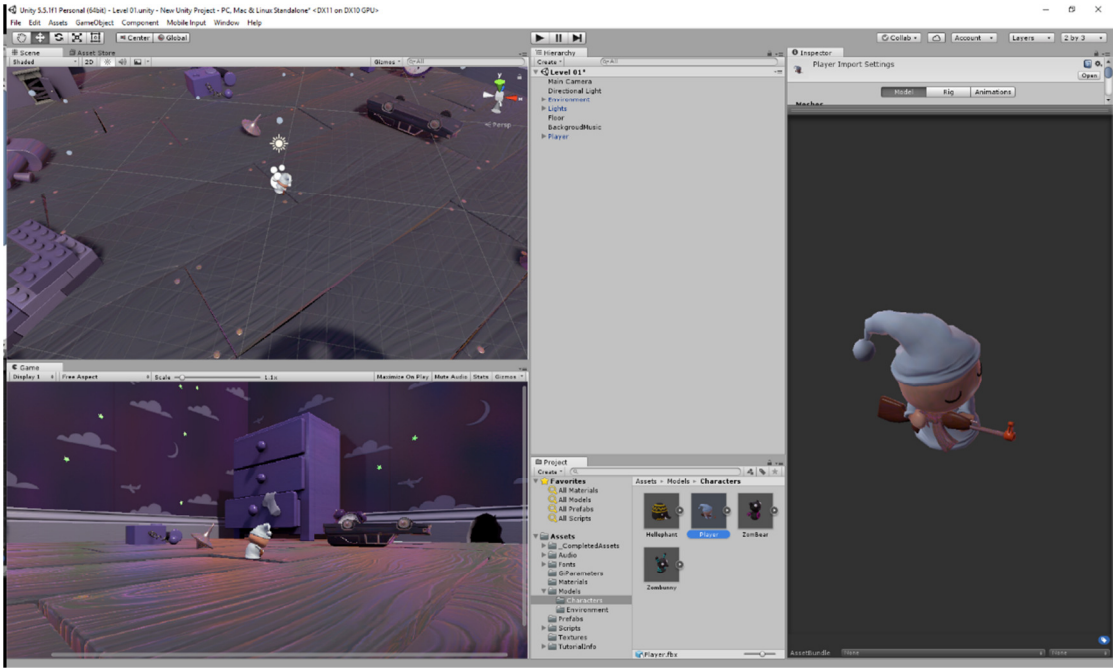
• Le| arrow Collapse selected item (tree views and search results). If the item is already collapsed, this will select its parent item.

• Alt + right arrow Expand item when showing assets as previews

6.2 Δημιουργία Χαρακτήρα Player Character

Το επόμενο και ένα από τα πιο σημαντικά κομμάτια του παιχνιδιού είναι η δημιουργία του ηρώα μας και οι κατάλληλες ρυθμίσεις που θα χρειαστεί έτσι ώστε να μπορέσει να κινείτε και να στοχεύει του αντίπαλους του.

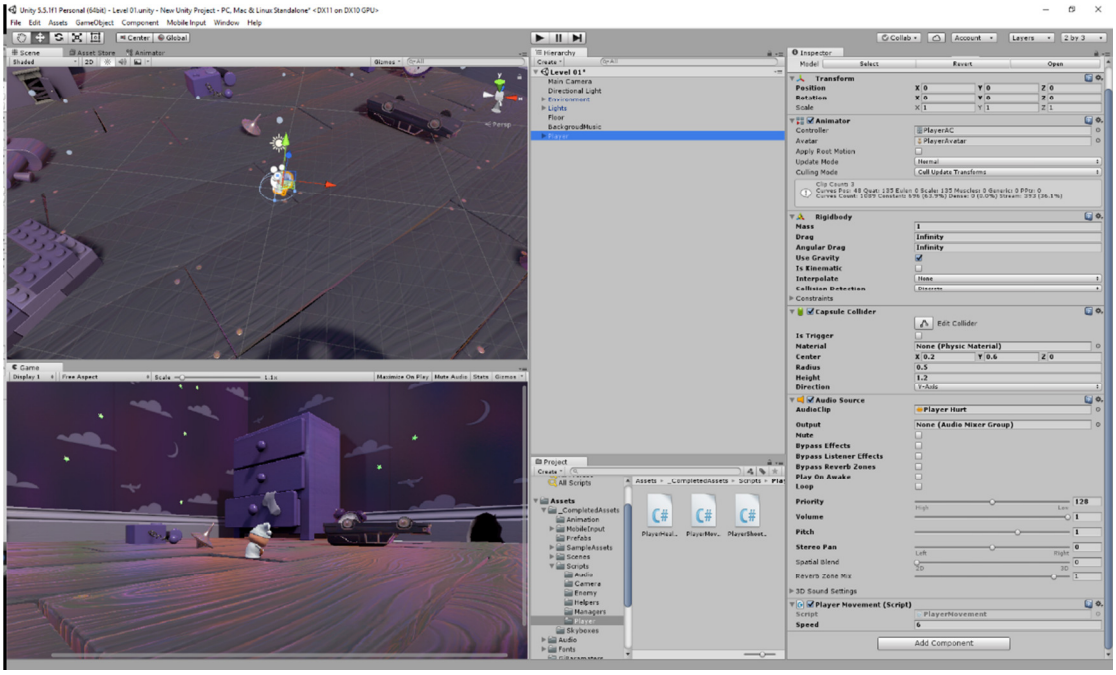
Η διαδικασία είναι απλή και έχει ως εξής , πηγαίνουμε στα assets και διαλέγουμε το φάκελο models μέσα στον οποίο βρίσκονται όλες οι φιγούρες που θα χρησιμοποιήσουμε . Διαλέγουμε την φιγούρα Player και την ρίχνουμε μέσα στην Hierarchy περιοχή μας έχοντας το αποτέλεσμα που θα δούμε στην παρακάτω φωτογραφία.



Εικόνα 23 Προσθήκη βασικού χαρακτήρα

Αφού έχουμε καταφέρει να δημιουργήσουμε τον χαρακτήρα πρέπει να του δώσουμε κίνηση και εδώ θα αναφερθούμε περιληπτικά στο Animation κομμάτι του ηρώα μας όπου του δίνουμε κίνηση στον οριζόντιο και κάθετο άξονα καθώς και την επιλογή να πεθάνει όταν οι εχθροί τον καταστρέψουν.

Για να καταφέρουμε ο ήρωας μας να μπορεί να έχει σώμα και να χτυπηθεί από οποιοδήποτε εχθρό θα πρέπει να του προσθέσουμε κάποια components όπως ένα Rigid body έτσι ώστε ο χαρακτήρας μας να έχει σώμα , ένα Capsule Collider έτσι ώστε ο ήρωας μας να έχει φυσικά χαρακτηριστικά όπως ύψος και άξονες και τέλος ένα Audio Source όπως δώσαμε προηγούμενος στο περιβάλλον μας . Τέλος θα δώσουμε κίνηση στον παίχτη μας χρησιμοποιώντας ένα Script με κώδικα γραμμένο σε C# με βασικές εντολές για την εμπρός και πίσω κίνηση του παίχτη μας.



Εικόνα 24 Ιδιότητες Χαρακτήρα

Μόλις τελειώσαμε τον βασικό χαρακτήρα του παιχνιδιού μας και μπορούμε να ελέγξουμε αν όλα είναι σωστά. Επιλέγοντας το Play θα δούμε ότι όντως ο ήρωας μας κινείτε στον οριζόντιο και κάθετο άξονα με τα πλήκτρα W και S για μπροστά και πίσω και A και D για δεξιά και αριστερά.

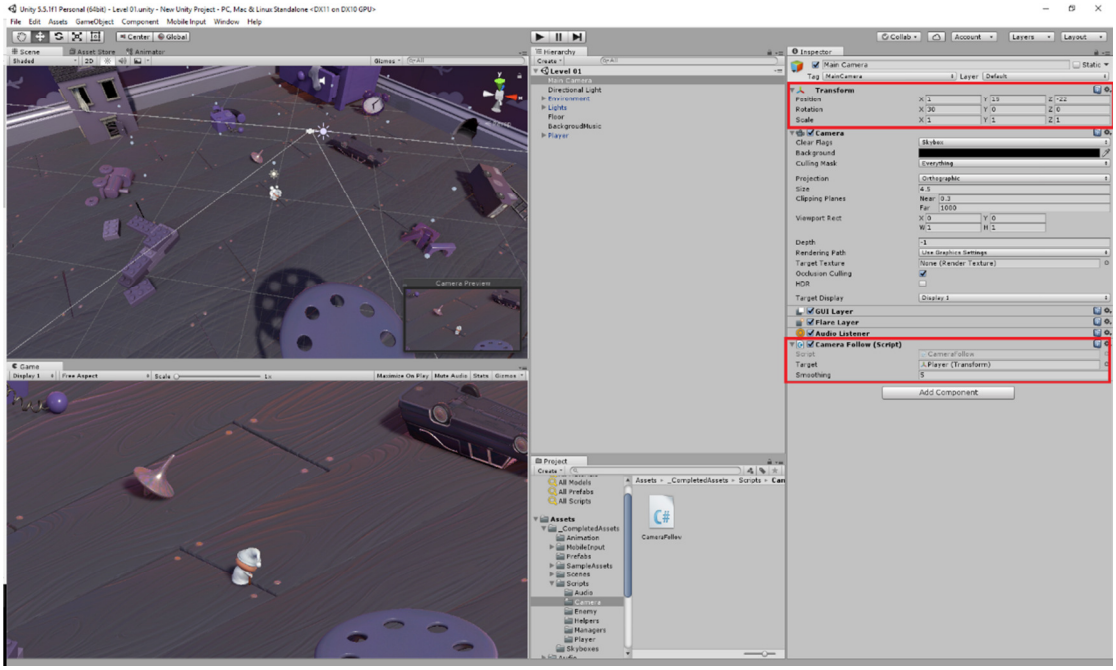
Script PlayerMovement

```
public float speed = 6f; // Η ταχύτητα που θα μετακινήσει ο παίκτης.

Vector3 movement;                // Ο φορέας για την αποθήκευση της κατεύθυνσης της κίνησης του παίκτη.

Animator anim;              // Αναφορά στο στοιχείο animator.

Rigidbody playerRigidbody; // Αναφορά στο άκαμπτο σώμα του παίκτη.

int floorMask;             // Μια μάσκα στρώματος έτσι ώστε μια ακτίνα να μπορεί να μεταδίδεται απλά σε αντικείμενα παιχνιδιού στο στρώμα δαπέδου.

float camRayLength = 100f;  // Το μήκος της ακτίνας από την κάμερα στη σκηνή.

void FixedUpdate ()

{

float h = CrossPlatformInputManager.GetAxisRaw("Horizontal");

float v = CrossPlatformInputManager.GetAxisRaw("Vertical");

Move (h, v);

Turning ();

Animating (h, v);

}

void Move (float h, float v)

{

movement.Set (h, 0f, v);


movement = movement.normalized * speed * Time.deltaTime;


playerRigidbody.MovePosition (transform.position + movement);

}
```

6.3 Ρυθμίσεις Κάμερας
Camera Setup

Ένα ακόμα σημαντικό κομμάτι του παιχνιδιού μας είναι να συνδέσουμε την main camera μας με τον χαρακτήρα έτσι ώστε να τον ακολουθεί την ώρα που θα τον κινούμε μέσα στον χώρο. Αρχικά, ορίζουμε τις σωστές συντεταγμένες στο Transform δηλαδή τον άξονα x y και z στο position αλλά και τον άξονα x στο Rotation .

Στην συνέχεια παίρνουμε από τα scripts το script με όνομα Camera Follow και το ρίχνουμε μέσα στην Main Camera μας καθώς και τον χαρακτήρα μας (player) για να μπορεί η κάμερα να τον ακολουθεί , θα τα δούμε όλα στην παρακάτω φωτογραφία .



Εικόνα 25 Ρυθμίσεις κάμερας

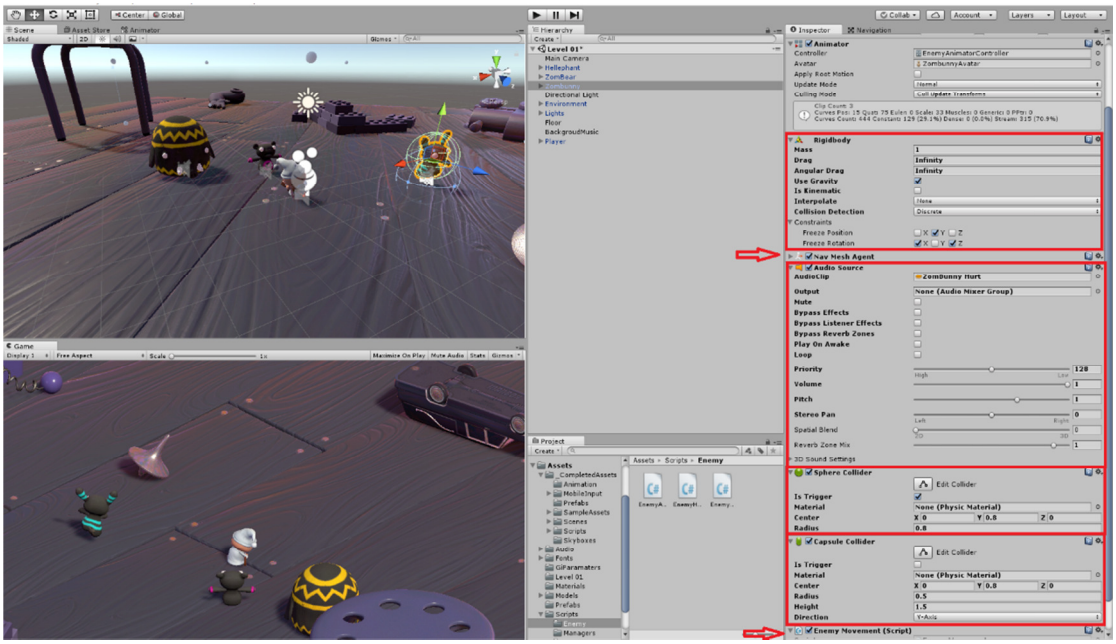
Script CameraFollow

```
public class CameraFollow : MonoBehaviour
{
    public Transform target;
    public float smoothing = 5f;


    Vector3 offset;


    void Start ()
    {
        offset = transform.position - target.position;
    }


    void FixedUpdate ()
    {
        Vector3 targetCamPos = target.position + offset;

        transform.position = Vector3.Lerp
(transform.position, targetCamPos, smoothing *
Time.deltaTime);
```

6.4 Πρώτος Εχθρός
The First Enemy

Στην επόμενη φάση θα πρέπει να δημιουργήσουμε τον πρώτο μας εχθρός. Ο εχθρός μας θα πρέπει να προκαλεί ένα damage στον ηρώα μας καθώς και να τον ακολουθεί με σκοπό να τον εξοντώσει.

Αρχικά θα τοποθετήσουμε τους εχθρούς που έχουμε σχεδιάσει μέσα στο περιβάλλον μας και όπως κάναμε προηγουμένως με τον ηρώα μας έτσι και τώρα θα δώσουμε σώμα στους εχθρούς μας, ήχο και κίνηση. Με την ιδιά διαδικασία θα προσθέσουμε components όπως Rigid body για να δώσουμε βαρύτητα στον εχθρό και να του ορίσουμε τους άξονες που θα μπορεί να πατάει, Capsule Collider για να δώσουμε σώμα και να μπορεί να χτυπηθεί ο εχθρός μας, Audio Source για να δώσουμε ηχητικό clip όταν θα χτυπάμε με σφαίρα τον εχθρό . Τέλος, θα προσθέσουμε ένα navigation system έτσι ώστε να μπορεί ο εχθρός μας να ακολουθεί συνεχώς τον ηρώα μας . Στην παρακάτω φωτογραφία θα δείτε όλα τα Components που περιγράψαμε προηγουμένως .



Εικόνα 26 Ρυθμίσεις Εχθρού

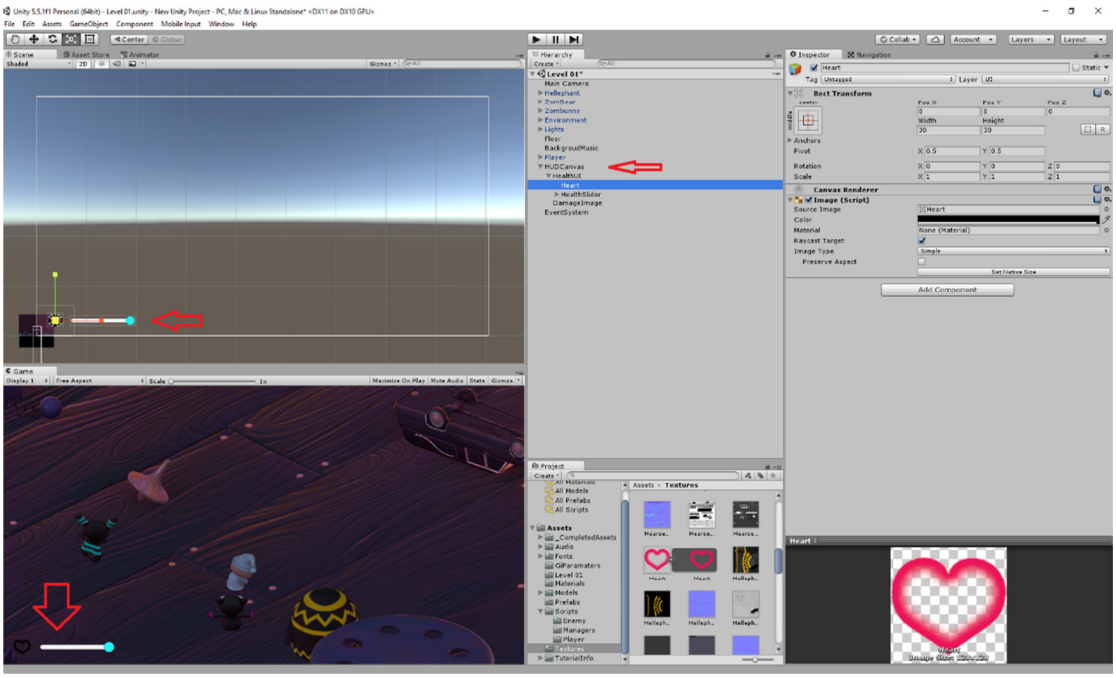
Τέλος θα ακολουθήσουμε την ιδιά διαδικασία για όλους τους εχθρούς ξεχωριστά.

6.5 Σχεδιασμός Πίνακα HUDCanvas

Σε αυτό το κεφάλαιο θα δημιουργήσουμε έναν πίνακα πίσω από το περιβάλλον μας στον οποίο θα προσθέσουμε όλα τα γραφικά στοιχεία όπως την ζωή του ηρώα μας , τον χρόνο παιχνιδιού , τα kills που θα πραγματοποιήσουμε και πολλά αλλά που θα δούμε αργότερα.

Για να φτιάξουμε τον πίνακα μας (canvas) πηγαίνουμε στο Game Object → UI → Canvas .

Στην συνέχεια θα δημιουργήσουμε ένα Empty Child το οποίο θα ονομάσουμε HealthUI και θα περιέχει την ζωή του ηρώα μας καθώς και μια μικρή φωτογραφία της καρδίας του. Επίσης θα δημιουργήσομε ένα ακόμα κενό στοιχείο το οποίο θα ονομάσουμε Health Slider και θα είναι η κινούμενη μπάρα με το ποσοστό της ζωής που μας απομένει .

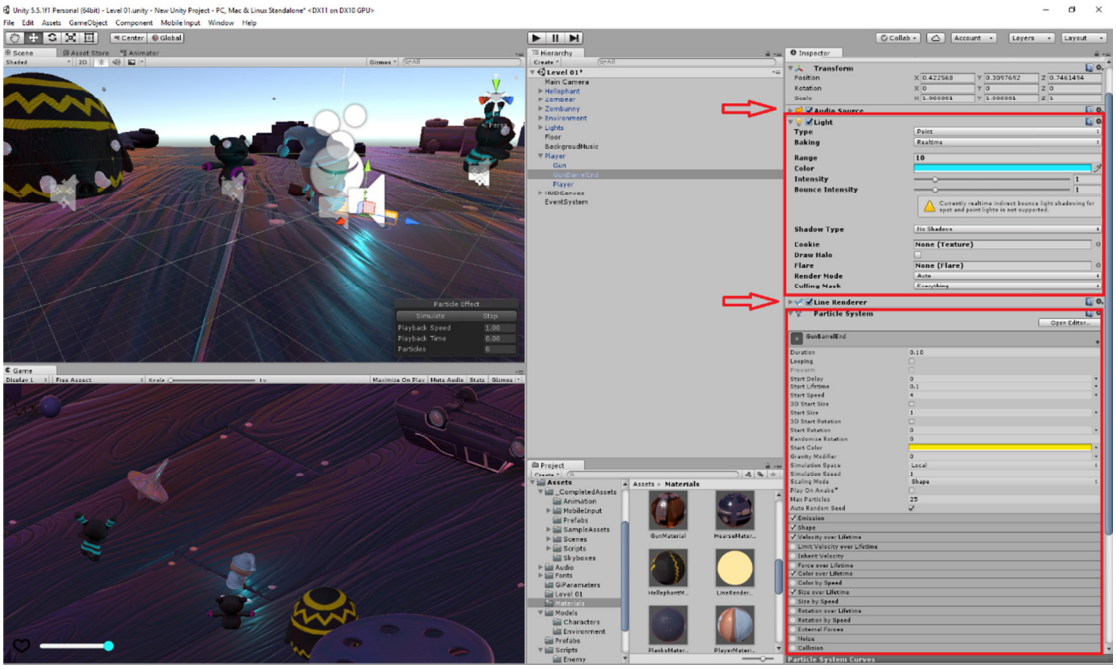


Εικόνα 27 Ρυθμίσεις Καμβά

6.6 Βλάπτοντας τον εχθρό
Harming Enemy

Σε αυτό το κεφάλαιο θα δούμε πως μπορούμε να κάνουμε damage στον εχθρό μας και πως το όπλο μας θα καταφέρει να ρίξει.

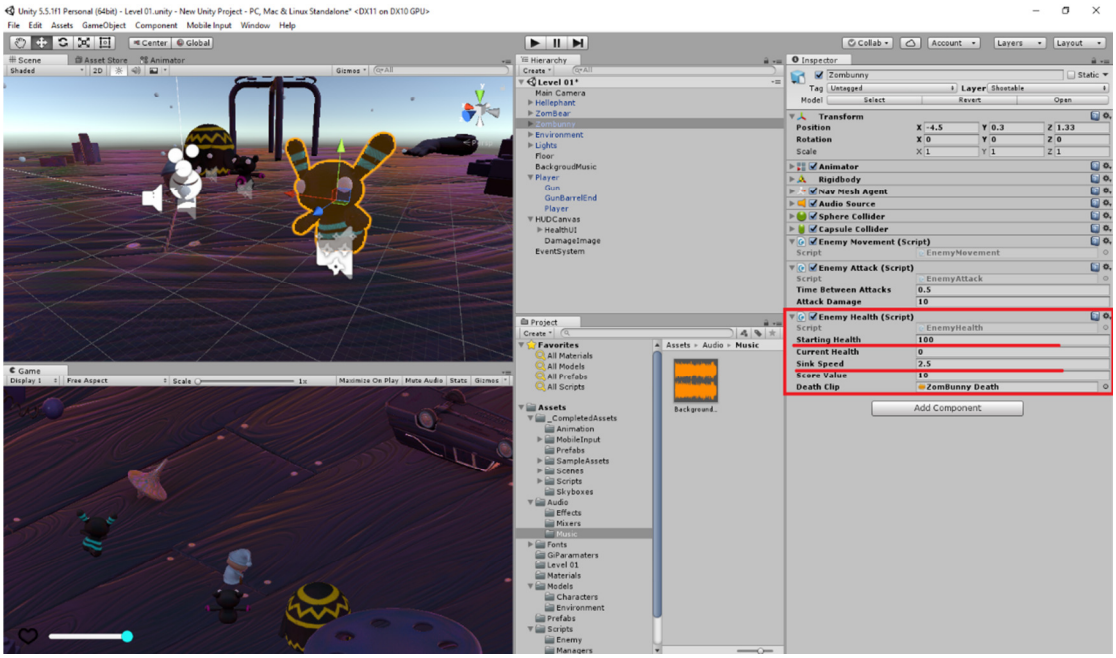
Αρχικά θα πάμε να προσθέσουμε κάποια components στο όπλο μας το οποίο υπάρχει μέσα στον ήρωας μας. Τα components μας είναι ένα Audio Source σαν αυτά που έχουμε ήδη χρησιμοποιήσει στους εχθρούς μας , στον ηρώα και στο περιβάλλον μας , το οποίο θα έχει τον ήχο της σφαίρας κάθε φορά που ρίχνουμε. Ένα ακόμα θα είναι το Light που το χρησιμοποιούμε για να φτιάξουμε την λάμψη της σφαίρας μας και τέλος ένα Line Renderer για να δώσουμε ακτίνα και να δημιουργήσουμε την πορεία της σφαίρας μας καθώς και το χρώμα της ακτίνας.



Εικόνα 28 Ρυθμίσεις Όπλου

Στην συνέχεια του κεφαλαίου μας θα δούμε πως ο εχθρός μας μπορεί να χτυπηθεί και τη script θα χρησιμοποιήσουμε.

Στην παρακάτω εικόνα θα δούμε τις παραμέτρους που έχει η ζωή του εχθρού μας .



Εικόνα 29 Προσθήκη Ζωής Εχθρού

Script EnemyHealth

```
public int startingHealth = 100;
  public int currentHealth;
  public float sinkSpeed = 2.5f;
  public int scoreValue = 10;
  public AudioClip deathClip;
```

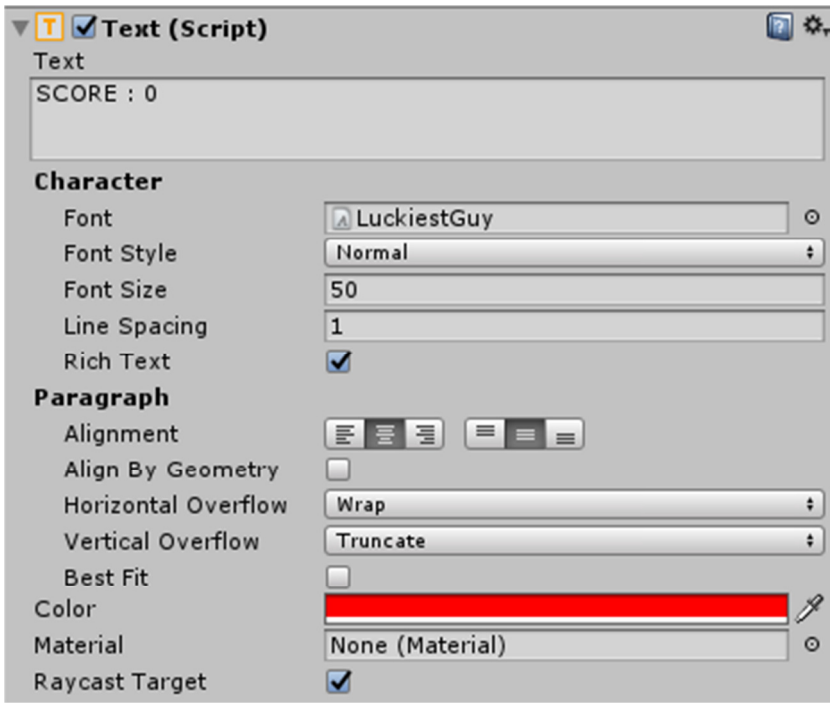
```
   void Death ()
{
 isDead = true;

 capsuleCollider.isTrigger = true;

 anim.SetTrigger ("Dead");

 enemyAudio.clip = deathClip;
 enemyAudio.Play ();
}



public void StartSinking ()
{
 GetComponent    <UnityEngine.AI.NavMeshAgent>    ().enabled    =
false;
 GetComponent <Rigidbody> ().isKinematic = true;
 isSinking = true;
 ScoreManager.score += scoreValue;
 EnemiesKilled.number++;
 Destroy (gameObject, 2f);
 }
}
```

Τέλος με αυτό το script θα δώσουμε στον εχθρό μας τις ρυθμίσεις που θέλουμε , δηλαδή την ζωή με την οποία θέλουμε να ξεκινήσει το παιχνίδι , ανά ποσά δευτερόλεπτα θέλουμε να βγαίνει ο επόμενος εχθρός καθώς και το σκορ το οποίο θα κερδίζει ο ήρωας μας όταν εξοντώνει έναν αντίπαλο.

6.7 Σκορ
(Scoring Points)

Σε αυτό το κεφάλαιο θα σας δείξουμε πώς καταφέραμε να εμφανίσουμε στην οθόνη μας την λέξη Score και με ποιο script μπορεί το score να ανεβαίνει ανάλογα με τον κάθε εχθρό που σκοτώνουμε. Αρχικά, θα χρησιμοποιήσουμε τον καμβά που ήδη έχουμε δημιουργήσει και θα φτιάξουμε μέσα σε αυτό ένα 'παιδί' του με όνομα Score Text.

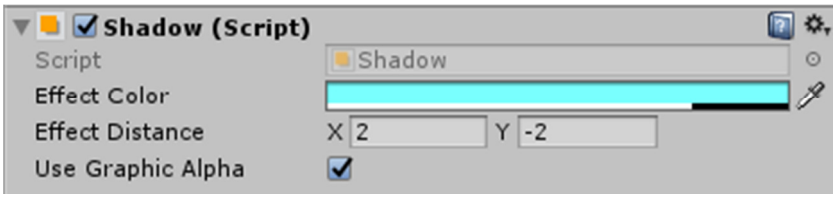
Το Score Text αρχικά μας δίνει την δυνατότητα σε συνδυασμό με τον καμβά να γράψουμε και να εμφανίσουμε στην οθόνη μας ότι θέλουμε. Σε αυτήν την περίπτωση εμείς θέλουμε να εμφανίσουμε το Score.



Εικόνα 30 Πίνακας Σκορ

Στο σημείο του Text γράφουμε το κείμενο που θέλουμε να εμφανίσουμε και στις παρακάτω ρυθμίσεις μπορούμε να μεγαλώσουμε η να μικρύνουμε τα γράμματα και στο Paragraph να φέρουμε το κείμενο μας στην θέση που εμείς επιθυμούμε. Τέλος, διαλέγουμε το χρώμα που θέλουμε να έχει το κείμενο μας και στην δική μας περίπτωση το κόκκινο.

Επίσης θα δούμε ότι μπορούμε να βάλουμε και ένα component το οποίο θα μας δώσει μια ωραία σκιά πίσω από τα γράμματα μας σε οποία απόχρωση εμείς θέλουμε. Στην δική μας περίπτωση χρησιμοποιήσαμε το γαλάζιο έτσι ώστε να κάνει μια ωραία αντίθεση με το κόκκινο των γραμμάτων.



Εικόνα 31 Ρυθμίσεις Σκιών

Τέλος, για να κλείσουμε το κεφάλαιο θα αναφέρουμε τον κώδικα που χρησιμοποιήσαμε.

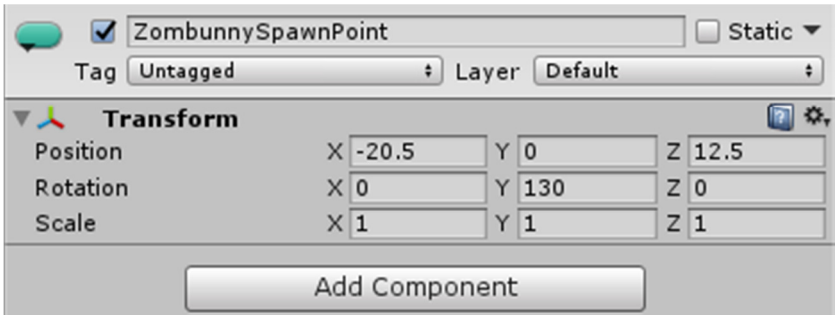
```
    public class ScoreManager : MonoBehaviour
{
    public static int score;


    Text text;


    void Awake ()
    {
        text = GetComponent <Text> ();
        score = 0;
    }


    void Update ()
    {
        text.text = "Score: " + score;
    }
}
```

6.8 Σημείο Αναγέννησης Εχθρών (Spawning Enemies)

Σε αυτό το κεφάλαιο θα αναφερθούμε στα σημεία οπού οι εχθροί μας εμφανίζονται στο δωμάτιο μας και θα ξεκινούν να μας κυνηγούν. Θα δημιουργήσουμε τρία σημεία αφού οι εχθροί μας είναι τρεις στον αριθμό , και θα τα μοιράσουμε στο δωμάτιο. Καθένα από αυτά τα σημεία θα είναι σταθερά και θα έχουν το καθένα συγκεκριμένες συντεταγμένες.

Ας ξεκινήσουμε δημιουργώντας ένα κενό αντικείμενο στον χώρο δηλαδή ένα Game Object κάτι που ήδη έχουμε ξαναφτιάξει στο παρελθόν. Θα κανουμε remane σε αυτό το αντικειμενο και θα του δωσουμε το ονομα του πρωτου μας εχθρου (ZombunnySpawnPoint).

Στο αντικείμενο που έχουμε δημιουργήσει μπορούμε να αλλάξουμε μόνο τις συντεταγμένες όπως θα δείτε στην παρακάτω εικόνα .



Εικόνα 32 Ρυθμίσεις Συντεταγμένων Εχθρού

Επάνω αριστερά θα δούμε το χρώμα που έχουμε διαλέξει για το συγκεκριμένο αντικείμενο και τις συντεταγμένες που χρησιμοποιήσαμε για να στείλουμε το σημείο του πρώτου μας εχθρού στο σημείο που θα δούμε στην παρακάτω εικόνα.

Εικόνα 33 Σημείο Έναρξης Εχθρού

Κλείνοντας το κεφάλαιο πρέπει να αναφέρουμε πως και για τους υπολοίπους εχθρούς θα χρησιμοποιήσουμε την ίδια διαδικασία αλλάζοντας μόνο τα χρώματα.

6.9 Timer – Kills

Στο τελευταίο κεφάλαιο σας παρουσιάζουμε τα δυο δικά μας scripts που χρησιμοποιήσαμε στο παιχνίδι έτσι ώστε να δώσουμε μια δική μας πινελιά και να το διαφοροποιήσουμε από το αρχικό.

Timer Script

```
public class Timer : MonoBehaviour
{

	float startTime;
	Text myText;
	PlayerHealth myPlayer;

	void Awake()
	{
		myPlayer = FindObjectOfType<PlayerHealth> ();
		myText = GetComponent<Text> ();
		startTime = Time.time;
	}
	void Start () {
	}

		void Update () {
		if (!myPlayer.isDead) {
			myText.text = "Time: " + ((int)(Time.time -
startTime)).ToString ();
```

Enemies Killed Script

```
    public class EnemiesKilled : MonoBehaviour
    {
        public static int number;        // The player's score.


        Text text;                      // Reference to the Text component.


        void Awake ()
        {
            // Set up the reference.
            text = GetComponent <Text> ();

            // Reset the score.
            number = 0;
        }


        void Update ()
        {

            // Set the displayed text to be the word "Score" followed by the
score value.
            text.text = "Kills: " + number;
        }
```

ΣΥΜΠΕΡΑΣΜΑΤΑ

Το Unity είναι ένα ολοκληρωμένο πακέτο όπου δίνει την δυνατότητα στον Developer να φτιάξει το δικό του παιχνίδι 2d ή 3d με σχετικά απλό τρόπο.
Το Unity Game Engine είναι στατιστικά το καλύτερο Game Engine όσον αφορά την δημιουργία παιχνιδιών σε Android και iOS. Με την έκδοση 5 και την μεγάλη αλλαγή στο rendering το Unity πιστεύουμε κυριαρχήσει και σε PC & Game Consoles γιατί πολύ απλά είναι πιο απλό και εύχρηστο για έναν αρχάριοι αλλά και έναν απαιτητικό Developer.
Το βασικό χαρακτηριστικό που το κάνει το πιο προσιτό Game Engine είναι οι βασικοί οδηγοί, τα πολλά tutorials που υπάρχουν στο διαδίκτυο, αλλά και το Community Forum όπου εύκολα και γρήγορα μπορούν να σε βοηθήσουν.
Σκοπός της παρούσας πτυχιακής
είναι να δείξει στο κοινό πως μπορεί να εκμεταλλευτεί το περιβάλλον της Unity και με λίγες γνώσεις προγραμματισμού να φτιάξει ένα παιχνίδι από το μηδέν. Η Unity παρέχει οτιδήποτε χρειάζεται κάποιος για την ανάπτυξη του παιχνιδιού του, από οδηγούς στον ιστοχώρο της, μέχρι και απίστευτα χαρακτηριστικά και επιλογές στο πρόγραμμά της. Τέλος, η πολλαπλή επιλογή πλατφορμών για την κυκλοφορία των παιχνιδιών την κάνει πιο προσιτή αφού οι κονσόλες, οι υπολογιστές και γενικά μία συσκευή αναπαραγωγής παιχνιδιών δε λείπει από τις ζωές των ανθρώπων, με τα βιντεοπαιχνίδια να αποτελούν μία από τις πιο «δυνατές» μορφές ψυχαγωγίες στις μέρες μας, όπου η τεχνολογία κιόλας αυξάνεται συνεχώς

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
3rd Person View	Προβολή παιχνιδιού σε τρίτο πρόσωπο
Audio Listener	Ακροατή ήχου
Browser	Πρόγραμμα περιήγησης
Collision	Επαφή
Collision Detection Method σύγκρουσης	Μέθοδος ανίχνευσης
Components	Στοιχεία
Developer	Προγραμματιστής
Debug	Έλεγχος σφαλμάτων
Editor	Συντάκτης προγραμματισμού
Event	Συμβάν
Game Engine	Εργαλείο ανάπτυξης παιχνιδιών
Live Preview	Ζωντανή προεπισκόπηση
Main Menu	Κύριο Μενού
Mute	Σίγαση
Non-Kinematic	Μη κινηματική
Packages	Πακέτα
Project	Έργο
Scene	Σκηνή
Scene Files	Αρχεία σκηνής
Scripts	Αρχείο κώδικας
Tag	Ετικέτα
Trigger Zone	Ζώνη ενεργοποίησης

ΑΝΑΦΟΡΕΣ

[1] Unity 3D –Game Engine https://unity3d.com

[2] MonoDevelop (IDE) http://www.monodevelop.com/

[3] Unity (game engine) https://en.wikipedia.org/wiki/Unity_(game_engine)

[4] Asset Store https://www.assetstore.unity3d.com/

[5] Windows https://www.microsoft.com/en-us/windows

[6] Unity- Answers (Forum) http://answers.unity3d.com

[7] Youtube https://www.youtube.com

[8] http://unity3d.com/learn/documentation - έγγραφα της Unity στον ιστότοπό της.

[9]] https://www.youtube.com/channel/UCG08EqOAXJk_YXPDsAvReSg - το κανάλι της Unity στο Youtube

[10] https://www.youtube.com/channel/UC0Ahv9Y12r2To-syiUoR8Lg - κανάλι στο Youtube περί Unity

[11] https://www.youtube.com/channel/UCmtyQOKKmrMVaKuRXz02jbQ - κανάλι στο Youtube περί Unity

[12] http://pixelnest.io/tutorials/2d-game-unity/ - ιστότοπος περι Unity

[13] https://gamedevelopment.tutsplus.com/ - ιστότοπος περι Unity