



Πανεπιστήμιο Πελοποννήσου
Τμήμα Ψηφιακών Συστημάτων

Σχολή Οικονομίας και Τεχνολογίας

Τμήμα Ψηφιακών Συστημάτων

(Τ.Ε.Ι ΠΕΛΟΠΟΝΝΗΣΟΥ)

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Δημιουργία δισδιάστατου παιχνιδιού σε μηχανή Unity»

ΓΙΩΡΓΟΣ ΜΑΝΤΑΣΑΣ

ΑΜ:2017062

Επιβλέπων : Παναγιώτης Κόκκινος

ΣΠΑΡΤΗ

2022

Περίληψη

Η παρούσα πτυχιακή εργασία ασχολείται με την σχεδίαση, ανάπτυξη και υλοποίηση 2D παιχνιδιού με χρήση της μηχανής γραφικών Unity καθώς και της γλώσσας προγραμματισμού C#. Στο πρώτο κεφάλαιο γίνεται η ιστορική αναδρομή στον τομέα των βιντεοπαιχνιδιών καθώς και η εξέλιξη τους ανά τα χρόνια. Στην συνέχεια βρίσκουμε την παρουσίαση του παιχνιδιού που δημιουργήθηκε για τις ανάγκες της παρούσας πτυχιακής εργασίας ενώ στο κεφάλαιο δύο γίνεται μια επισκόπηση των προγραμμάτων που βοήθησαν στην υλοποίηση του. Παρακάτω γίνεται η ανάλυση του γραφικού περιβάλλοντος της Unity engine όπου είναι το κύριο λογισμικό που χρησιμοποιήθηκε για την ανάπτυξη και υλοποίηση του παιχνιδιού. Στο τελευταίο κεφάλαιο εμβαθύνουμε στην δημιουργία του παιχνιδιού σε τομείς όπως τα objects, φωτισμοί, ήχοι, τα animations, το σύστημα διεπαφής με τον χρήστη UI, κομμάτια κώδικα και διάφορους μηχανισμούς σε γλώσσα προγραμματισμού C# που χρησιμοποιούνται συνεχώς από το παιχνίδι.

Copyright ©-All rights reserved Γεώργιος Μαντασάς, 2022.

Με την επιφύλαξη παντός δικαιώματος. Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα.

Το περιεχόμενο αυτής της εργασίας δεν απηχεί απαραίτητα τις απόψεις του Τμήματος, του Επιβλέποντα, ή της επιτροπής που την ενέκρινε.

Υπεύθυνη Δήλωση

Βεβαιώνω ότι είμαι συγγραφέας αυτής της πτυχιακής εργασίας, και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης, βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Τμήματος Ψηφιακών Συστημάτων του Πανεπιστημίου Πελοποννήσου.



ΓΕΩΡΓΙΟΣ ΜΑΝΤΑΣΑΣ

Περιεχόμενα

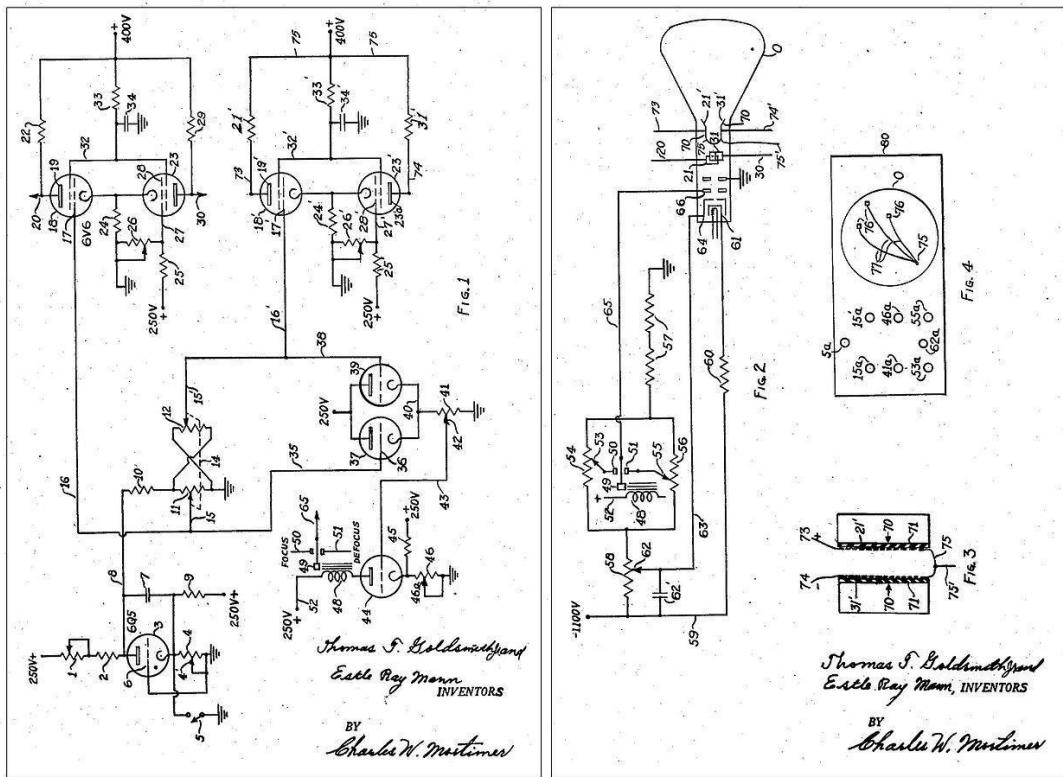
ΚΕΦΑΛΑΙΟ 1 ΕΙΣΑΓΩΓΗ	5
1.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ	5
EIKONA 1.1.1: Cathode Ray Tube Amusement Device	5
EIKONA 1.1.2: Tennis for Two	6
EIKONA 1.1.3: Tennis for Two controller	7
1.2 ΤΑ ΒΙΝΤΕΟΠΑΙΧΝΙΔΙΑ ΣΗΜΕΡΑ	8
EIKONA 1.2.1: The Game Awards	8
EIKONA 1.2.2: Night Shift	9
EIKONA 1.2.3: Virtual reality Firefighter training simulator	10
1.3 ΠΑΡΟΥΣΙΑΣΗ ΠΑΙΧΝΙΔΙΟΥ	11
EIKONA 1.3.1: Trapped Inside Scene	11
EIKONA 1.3.2: Trapped Inside Scene	12
ΚΕΦΑΛΑΙΟ 2 ΠΡΟΓΡΑΜΜΑΤΑ	13
2.1 ΑΝΑΦΟΡΑ ΣΤΑ ΠΡΟΓΡΑΜΜΑΤΑ	13
2.2 ADOBE PHOTOSHOP	14
EIKONA 2.2.1: Photoshop Workspace	14
2.3 ADOBE PREMIERE PRO	15
EIKONA 2.3.1: Premiere pro Workspace	16
2.4 AUDACITY	16
EIKONA 2.4.1: Audacity Workspace	17
2.5 VISUAL STUDIO	18
EIKONA 2.5.1: Visual Studio Workspace	18
2.6 NOTEPAD++	19
EIKONA 2.6.1: Notepad++ Workspace	19
2.7 TRELLO	20
EIKONA 2.7.1: Trello Workspace	20
2.8 GOOGLE DRIVE	21
EIKONA 2.8.1: Google Drive Workspace	21
2.9 UNITY	22
EIKONA 2.9.1: Unity Select Projects	22
EIKONA 2.9.2: Unity New Project	23
EIKONA 2.9.3: Unity Scene-Game	24
EIKONA 2.9.4: Unity Project-Console	24
EIKONA 2.9.5: Unity Assets-Inspector	25
EIKONA 2.9.6: Unity Toolbar	26

ΚΕΦΑΛΑΙΟ 3 ΥΛΟΠΟΙΗΣΗ	27
3.1 ΑΝΤΙΚΕΙΜΕΝΑ (OBJECTS)	27
ΕΙΚΟΝΑ 3.1.1: Unity Asset Store	27
ΕΙΚΟΝΑ 3.1.2: Unity Layers	28
ΕΙΚΟΝΑ 3.1.3: Unity Character Inspector	29
3.2 ΦΩΤΙΣΜΟΣ	30
ΕΙΚΟΝΑ 3.2.1: Unity Lights Off	30
ΕΙΚΟΝΑ 3.2.2: Unity Lights On	31
ΕΙΚΟΝΑ 3.2.3: Unity Light Inspector	32
3.3 ΗΧΟΣ	33
ΕΙΚΟΝΑ 3.3.1: Unity Audiosource	34
ΕΙΚΟΝΑ 3.3.2: Unity Audio Mixer	35
3.4 ΚΙΝΗΣΗ (ANIMATIONS)	40
ΕΙΚΟΝΑ 3.4.1: Animation	40
ΕΙΚΟΝΑ 3.4.2: Unity Animation Timeline	41
ΕΙΚΟΝΑ 3.4.3: Unity Animator	42
3.5 ΔΙΕΠΑΦΗ ΜΕ ΤΟΝ ΧΡΗΣΤΗ. (UI)	45
ΕΙΚΟΝΑ 3.5.1: Unity UI Tools	45
ΕΙΚΟΝΑ 3.5.2: Unity UI Settings	46
ΕΙΚΟΝΑ 3.5.3: Unity UI Text	47
ΕΙΚΟΝΑ 3.5.4: Unity UI On Click	48
3.6 C# SCRIPTS ΚΑΙ ΜΗΧΑΝΙΣΜΟΙ	52
ΒΙΒΛΙΟΓΡΑΦΙΑ	60
ΠΗΓΕΣ	60
ASSETS ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ	62

ΚΕΦΑΛΑΙΟ 1 ΕΙΣΑΓΩΓΗ

1.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Τα βιντεοπαιχνίδια δεν είναι πρόσφατη ανακάλυψη ήδη από το 1947 γινόντουσαν προσπάθειες δημιουργίας βιντεοπαιχνιδιών. Συγκεκριμένα ο Thomas T. Goldsmith Jr. και Estle Ray Mann δημιούργησαν το Cathode Ray Tube Amusement Device που ήταν το πρώτο βιντεοπαιχνίδιο κατασκευασμένο σε έναν καθοδικό σωλήνα. Το παιχνίδι προσομοίαζε έναν πύραυλο εμπνευσμένο από οθόνες ραντάρ του Δευτέρου Παγκοσμίου Πολέμου και χρησιμοποιούσε αναλογικά κυκλώματα για τον έλεγχο της κουκίδας που αντιπροσωπεύει τον πύραυλο στην οθόνη.



EIKONA 1.1.1: Cathode Ray Tube Amusement Device

ΠΗΓΗ: https://en.wikipedia.org/wiki/Cathode-ray_tube_amusement_device

Στην συνέχεια ακολούθησαν και άλλες προσπάθειες δημιουργίας βιντεοπαιχνιδιών. Το έτος 1947-1958 ο γνωστός βρετανός μαθηματικός Alan Turing μαζί με τον συναδέλφου του Dietrich Prinz δημιούργησαν το πρώτο ηλεκτρονικό σκάκι αλλά δυστυχώς οι υπολογιστές της εποχής δεν ήταν αρκετά ισχυροί για να “τρέξουν” πλήρως το παιχνίδι. Το 1958 ο William Higinbotham έφτιαξε το ιστορικό Tennis for Two, ένα βιντεοπαιχνίδι εξομοιωτή τένις που χρησιμοποιεί έναν παλμογράφο για να εμφανίσει την μπάλα του τένις μαζί με το γήπεδο.



EIKONA 1.1.2: Tennis for Two
ΠΗΓΗ: https://en.wikipedia.org/wiki/Tennis_for_Two

Καθώς και δύο ελεγκτές αλουμινίου έναν για κάθε παίκτη για τον χειρισμό του παιχνιδιού.



EIKONA 1.1.3: Tennis for Two controller
ΠΗΓΗ: https://en.wikipedia.org/wiki/Tennis_for_Two

Περνούσαν τα χρόνια και κυκλοφορίες όπως το Pac-man της NAMCO το 1980 και το Tetris του Alexey Pajitnov το 1984 τράβηξαν το ενδιαφέρον του κοινού και έφεραν τα βιντεοπαιχνίδια από το παρασκήνιο στο επίκεντρο του ενδιαφέροντος.

1.2 ΤΑ ΒΙΝΤΕΟΠΑΙΧΝΙΔΙΑ ΣΗΜΕΡΑ

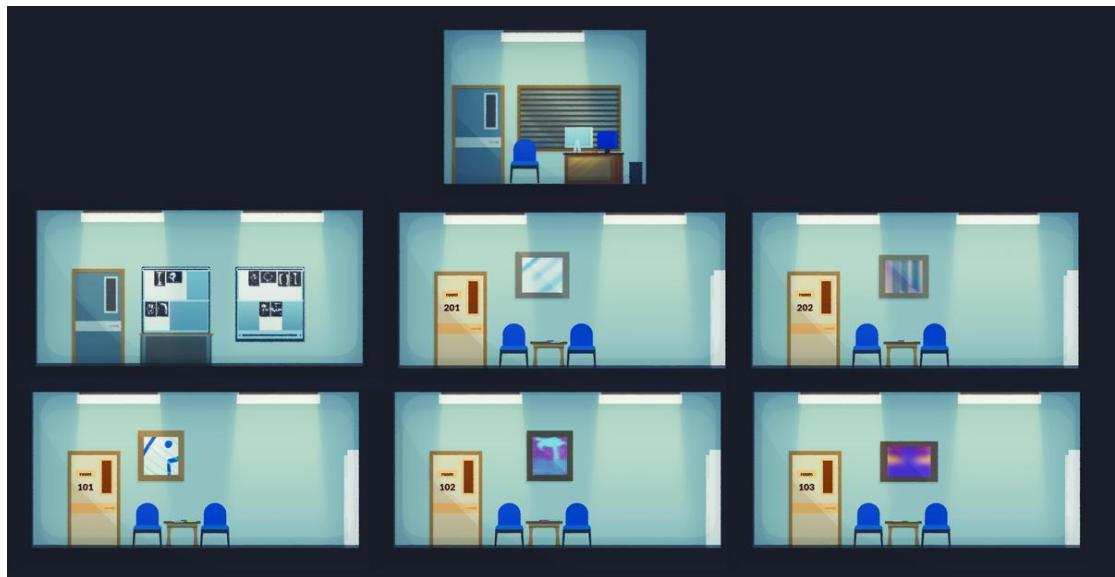
Πλέων υπάρχουν εταιρείες κολοσσοί που ασχολούνται αποκλειστικά με την ανάπτυξη μεγάλων τίτλων AAA (triple-A) αλλά και μικρότερα studios μικρότερης παραγωγής που ονομάζονται independent video game (indie games). Εκτός από το κομμάτι της ψυχαγωγίας, διοργανώνονται εκδηλώσεις για την απονομή βραβείων από ειδικές επιτροπές σε τομείς όπως η σχεδίαση, τα γραφικά, η ιστορία, οι ήχοι, η σκηνοθεσία και άλλα. Με παρόμοιο τρόπο όπως αυτόν της απονομής βραβείων OSCAR του κινηματογράφου.



ΕΙΚΟΝΑ 1.2.1: The Game Awards

ΠΗΓΗ: https://en.wikipedia.org/wiki/The_Game_Awards_2019

Τα βιντεοπαιχνίδια με τον καιρό εισέβαλαν και στον ιατρικό τομέα. Το EndeavourRX είναι ένα παιχνίδι που εγκρίθηκε από την Υπηρεσία Τροφίμων και Φαρμάκων των ΗΠΑ (USA Food and Drug Administration-FDA) ως ένα συνταγογραφούμενο φάρμακο για παιδιά ηλικίας οκτώ έως δώδεκα ετών που έχουν διαταραχή ελλειμματικής προσοχής και υπερκινητικότητας (ADHD). Η εταιρεία εκπαιδευτικών βιντεοπαιχνιδιών Schell Games δημιούργησε το Night Shift, ένα ιατρικό βιντεοπαιχνίδι που στοχεύει στην καλύτερη κατανόηση των γιατρών στην ακριβή διάγνωση τραύματος σε ασθενείς, προκειμένου να τους παρέχουν την κατάλληλη θεραπεία.



EIKONA 1.2.2: Night Shift

ΠΗΓΗ:

<https://www.ertnews.gr/eidiseis/epistimi/ena-iatriko-vinteopechnidi-voitha-tous-giatrous-sti-diagnosi-travmatos/>

Την τελευταία δεκαετία έχουν έρθει στο επίκεντρο της προσοχής τα παιχνίδια εικονικής πραγματικότητας Virtual reality (VR) και ακούγεται πως είναι το μέλλον της ψυχαγωγίας. Αυτά τα παιχνίδια απαιτούν έναν ειδικό εξοπλισμό που στις απλές περιπτώσεις αποτελείται από ένα ζευγάρι γυαλιά VR ενώ σε άλλες περιπτώσεις υπάρχουν δύο ειδικά χειριστήρια για την αλληλεπίδραση με τον κόσμο της εικονικής πραγματικότητας. Σε αυτά τα παιχνίδια συνήθως διαδραματίζεται ένα σενάριο προσομοίωσης μιας συνθήκης και έχουν την δυνατότητα να σε βάλουν πιο βαθιά στον ρόλο μιας ενδεχόμενης κατάστασης. Αυτή η κατάσταση μπορεί να είναι μια δουλειά, ένα άθλημα ή ένας στρατιώτης του δευτέρου παγκοσμίου πολέμου. Φυσικά αυτή η τεχνολογία δεν σταματάει εκεί καθώς χρησιμοποιείται για εκμάθηση ή εξοικείωση μιας κατάστασης. Χρησιμοποιείται επίσης από γιατρούς για την εκμάθηση κάποιου χειρουργείου ακόμα και από πυροσβέστες για να προσομοιάσουν ένα ενδεχόμενο μιας μεγάλης πυρκαγιάς.



EIKONA 1.2.3: Virtual reality Firefighter training simulator
ΠΗΓΗ: <https://www.virtuallytheretraining.com/training-systems/>

1.3 ΠΑΡΟΥΣΙΑΣΗ ΠΑΙΧΝΙΔΙΟΥ

Στα πλαίσια της παρούσας πτυχιακής εργασίας δημιουργήθηκε ένα δισδιάστατο platformer παιχνίδι που διαδραματίζεται στο κοντινό παρελθόν και δανείζεται στοιχεία από την Ελληνική μυθολογία.



ΕΙΚΟΝΑ 1.3.1: Trapped Inside Scene

Το παιχνίδι παίζεται από έναν παίκτη. Ο χειρισμός του είναι απλός και γνώριμος καθώς χρησιμοποιεί τα πλήκτρα 'A' και 'D' για την περιήγηση του χαρακτήρα στον χώρο, το 'space' για άλμα και τέλος 'W' για να διεπαφή με το περιβάλλον. Ο σκοπός του παίκτη είναι να βγάλει εις πέρας τον στόχο του πρωταγωνιστή που είναι να ανακαλύψει μια μυστηριώδης εξαφάνιση και εν τέλη να ανακαλύψει τι συμβαίνει στον ίδιο του τον εαυτό. Το παιχνίδι αποτελείται από μικρούς γρίφους και χωρίζεται σε τέσσερα κεφάλαια όπου ο παίκτης έχει την επιλογή από το κύριο μενού να επιλέξει ποιο κεφάλαιο θέλει να παίξει ή μπορεί να το ξεκινήσει από την αρχή και να κυλήσει ομαλά η ιστορία μέχρι το τέλος. Υπάρχουν δύο πιθανά σενάρια που ο παίκτης μπορεί να τελειώσει το παιχνίδι, το κάθε ένα από αυτά αποκαλύπτει περισσότερα στοιχεία για την ιστορία του. Όσο εξελίσσεται το παιχνίδι ο χαρακτήρας συναντάει

πλάσματα - θεούς της Ελληνικής μυθολογίας και ξεκλειδώνει 'τρόπαια' που μέσα από το μενού της παύσης φαίνονται πληροφορίες και δεδομένα σχετικά με το εκάστοτε πλάσμα - θεό που έχει ξεκλειδωθεί το τρόπαιο του. Στον παρακάτω σύνδεσμο βρίσκεται το αρχεία για την λήψη του παιχνιδιού.



EIKONA 1.3.2: Trapped Inside Scene

Λήψη του παιχνιδιού: <https://drive.google.com/drive/folders/1rNm-MP87clWsDzvqvTpcVrpKAJ4lCfiB?usp=sharing>

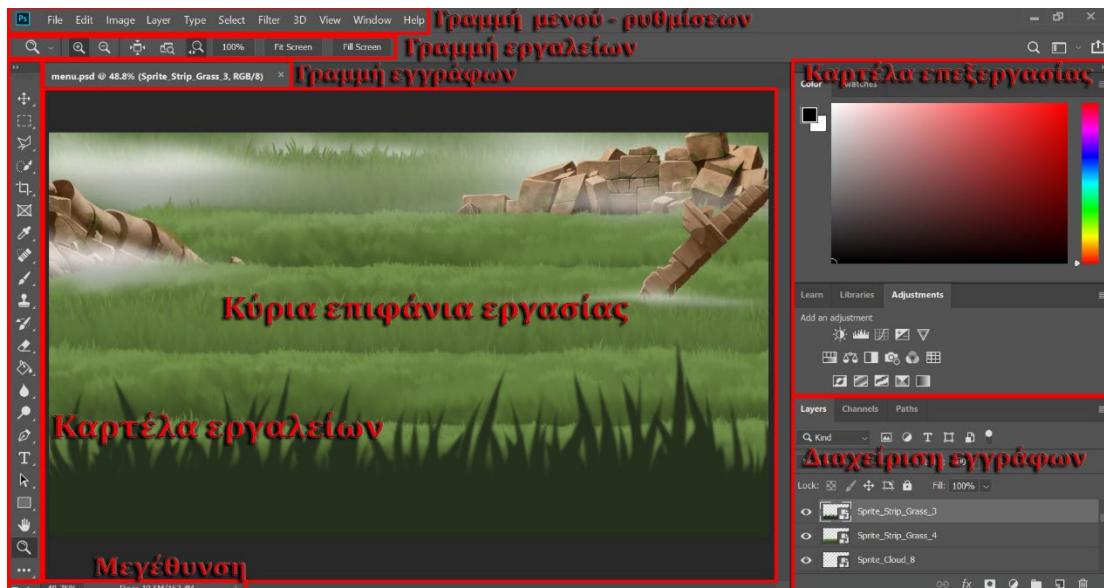
ΚΕΦΑΛΑΙΟ 2 ΠΡΟΓΡΑΜΜΑΤΑ

2.1 ΑΝΑΦΟΡΑ ΣΤΑ ΠΡΟΓΡΑΜΜΑΤΑ

Προκειμένου να σχεδιαστεί και να υλοποιηθεί το παραπάνω παιχνίδι χρησιμοποιήθηκαν τα προγράμματα που αναφέρονται ονομαστικά παρακάτω. Για την δημιουργία και επεξεργασία των γραφικών, για τις εικόνες και τα βίντεο χρησιμοποιήθηκαν τα Adobe Photoshop και Adobe premiere pro που απαιτούν μία χρέωση για την χρήση τους. Σχετικά με την δημιουργία και την επεξεργασία ήχου χρησιμοποιήθηκε το Audacity που παρέχεται δωρεάν, για την συγγραφή κώδικα C# τα δωρεάν προγράμματα Visual studio και Notepad++, για την καλύτερη οργάνωση και διαχείριση το Trello και Google drive, συγκεκριμένα χρησιμοποιήθηκαν οι δωρεάν εκδώσεις τους. Τέλος με την μηχανή γραφικών Unity έγινε η σύνδεση όλων των παραπάνω και η τελική υλοποίηση του παιχνιδιού. Στην Unity engine χρησιμοποιήθηκε το πλάνο Personal που δεν απαιτεί κάποια χρέωση και όπως προδίδει και η ονομασία του είναι κατάλληλο για προσωπική χρήση.

2.2 ADOBE PHOTOSHOP

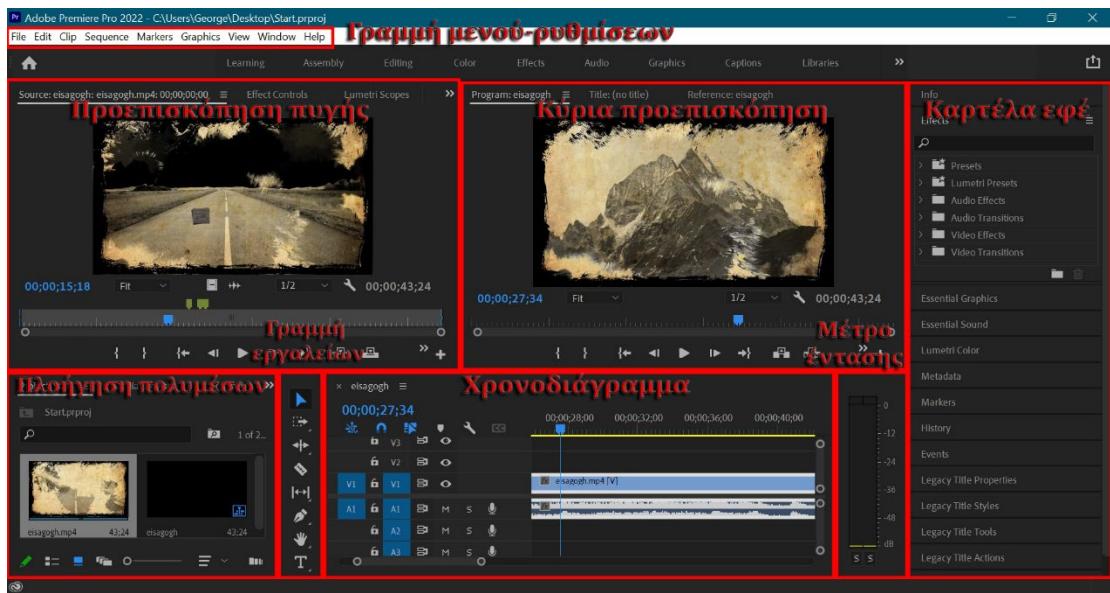
Το Photoshop της Adobe system κατασκευάστηκε αρχικά το 1988 από δύο αδέρφια, τον Thomas Knoll και John Knoll με την αρχική ονομασία του να είναι ImagePro που αργότερα άλλαξε σε Photoshop. Αυτή την στιγμή είναι ένα από τα κορυφαία προγράμματα επεξεργασίας γραφικών. Το περιβάλλον εργασίας του είναι σχετικά απλό αποτελείται από την βασική επιφάνεια εργασίας στο κέντρο την οθόνης όπου εκεί γίνονται οι βασικές επεξεργασίες των γραφικών, στα αριστερά υπάρχει η καρτέλα εργαλείων που προσφέρει στους χρήστες την δυνατότητα να χρησιμοποιήσουν εργαλεία όπως εισαγωγή γραμμάτων, κόψιμο εικόνας, εργαλεία κλωνοποίησης γραφικών και άλλα. Στα δεξιά υπάρχει η καρτέλα επεξεργασίας και διαχείρισης εγγράφων, από εκεί οι χρήστες μπορούν να αλλάξουν τα χρώματα των γραφικών, να προσθέσουν σκιές και γενικότερα να αλλάξουν την εμφάνιση μιας εικόνας. Πάνω βρίσκεται η γραμμή εργαλείων, η γραμμή των μενού-ρυθμίσεων και εγγράφων όπου εκεί παρέχονται περισσότερα εργαλεία επεξεργασίας αλλά και ρυθμίσεις για το ίδιο το πρόγραμμα. Τέλος στη κάτω περιοχή εμφανίζεται η μεγέθυνση που έχει υποστεί η κύρια επιφάνεια του προγράμματος σε μορφή ποσοστού επί τοις εκατό.



EIKONA 2.2.1: Photoshop Workspace

2.3 ADOBE PREMIERE PRO

Το Premiere pro είναι ένα από τα κορυφαία προγράμματα επεξεργασίας βίντεο. Κατασκευάστηκε από την Adobe το 2007 και ο προκάτοχος του είναι το Premiere που είχε κυκλοφορήσει το 1991. Η αρχική οθόνη του προγράμματος αποτελείται από την κύρια προεπισκόπηση όπου οι χρήστες μπορούν να δουν την προεπισκόπηση του τελικού βίντεο που προκύπτει από την επεξεργασία, τη προεπισκόπηση πηγής όπου εκεί εμφανίζεται κάποιο αρχείο που χρειάζεται να προστεθούν στο αρχείο επεξεργασίας. Στην καρτέλα προεπισκόπηση πηγής μπορούν να προστεθεί και άλλες λειτουργίες όπως η δημιουργία κινούμενης εικόνας animation ή η επεξεργασία των εφέ. Κάτω αριστερά βρίσκεται η καρτέλα πλοήγησης πολυμέσων. Σε αυτό το σημείο οι χρήστες μπορούν να εισάγουν τα αρχεία που επιθυμούν να επεξεργαστούν στο πρόγραμμα και στην συνέχεια να τα προσθέσουν στο χρονοδιάγραμμα που βρίσκεται χαμηλά στο κέντρο. Δεξιά από την καρτέλα πλοήγησης υπάρχει η γραμμή εργαλείων που επιτρέπει στους χρήστες να κόψουν, να ενώσουν και να γράψουν πάνω σε κάποιο αρχείο. Στο χρονοδιάγραμμα γίνεται η εισαγωγή των γραφικών και ήχων στο σωστό χρονικό διάστημα που καθορίζουν το τελικό αποτέλεσμα. Ακριβώς δίπλα υπάρχει το μέτρο της έντασης του ήχου σε μορφή κλίμακας Ντεσιμπέλ dB. Στην δεξιά πλευρά της οθόνης υπάρχει η καρτέλα με τα εφέ και κάποια βασικά γραφικά που προσφέρονται από το πρόγραμμα ή που έχουν προστεθεί από τους χρήστες και μπορούν να χρησιμοποιηθούν στο τελικό αποτέλεσμα. Τέλος όπως όλα τα προγράμματα πάνω στην κορυφή υπάρχει η γραμμή μενού και ρυθμίσεων.

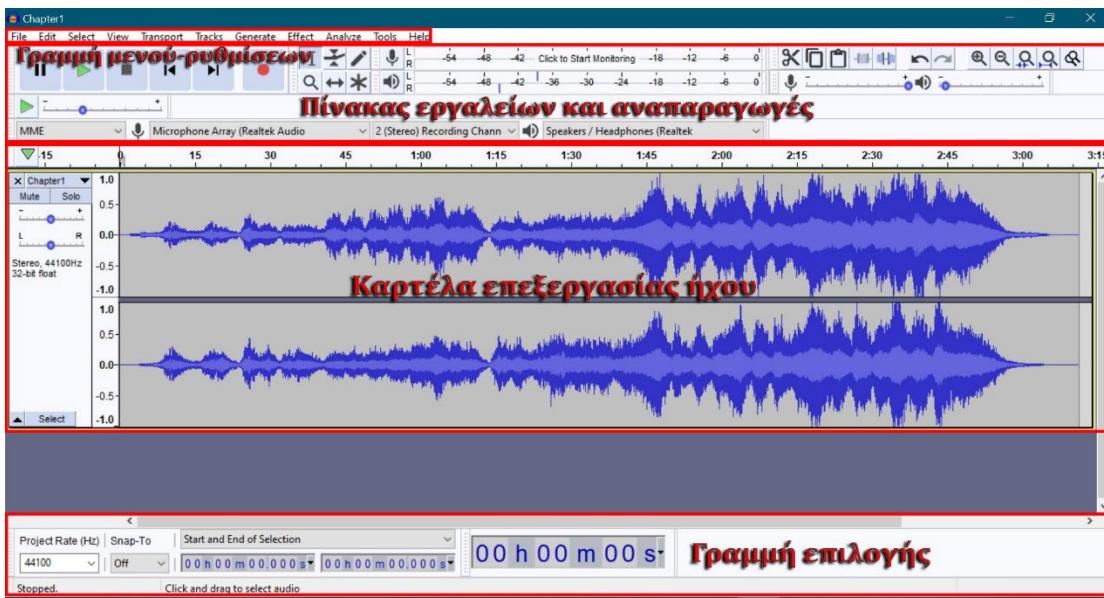


EIKONA 2.3.1: Premiere pro Workspace

2.4 AUDACITY

Το Audacity είναι ένα πρόγραμμα ανοικτού κώδικα εγγραφής και επεξεργασίας ήχου που κατασκεύασε ο πλέον εργαζόμενος της Google Dominic Mazzoni όπου ακόμα παραμένει ο κύριος συντηρητής του προγράμματος. Το γραφικό του περιβάλλον είναι απλό και εύχρηστο. αποτελείται από την καρτέλα επεξεργασίας ήχου που βρίσκεται στο κέντρο της οθόνης. Εκεί οι χρήστες μπορούν να δουν και να αναπαράγουν τον ήχο που θέλουν σε μορφή κύματος. Ψηλά βρίσκεται ο πίνακας εργαλείων και αναπαραγωγής. Από εκεί οι χρήστες με την βοήθεια των εργαλείων που προσφέρονται από το πρόγραμμα μπορούν

να επεξεργαστούν τον ήχο καθώς και να επιλέξουν την μέθοδο εξαγωγής του. Στην κάτω περιοχή της οθόνης βρίσκεται η γραμμή επιλογής όπου εκεί εμφανίζεται ανά πάσα στιγμή ο ακριβής χρονισμός του κομματιού αναπαραγωγής. Στην κορυφή του προγράμματος υπάρχει η γραμμή μενού και ρυθμίσεων όπου εκεί εκτός από τις ρυθμίσεις του προγράμματος υπάρχουν και διάφορα εφέ-φίλτρα που μπορούν εύκολα να προσαρμοστούν στον επιλεγμένο ήχο.



EIKONA 2.4.1: Audacity Workspace

2.5 VISUAL STUDIO

Για την συγγραφή του κώδικα χρησιμοποιήθηκε το visual studio. Είναι μια εφαρμογή επεξεργασίας κώδικα της Microsoft που υποστηρίζει έξυπνη συμπλήρωση (IntelliSense) καθώς και την αναδιαμόρφωση κώδικα. Έχει ενσωματωμένο εντοπισμό σφαλμάτων που λειτουργεί τόσο σε πηγαίο κώδικα όσο και σε γλώσσα μηχανής. Επίσης επιτρέπει τους χρήστες να εισάγουν κάποια πρόσθετα (plug-ins) που προσφέρουν ένα μεγάλο εύρος ευελιξίας και προσαρμογής των δυνατοτήτων του προγράμματος ανάλογα με τις ανάγκες των χρηστών. Ο σχεδιασμός του είναι πολύ απλός καθώς δεν διαφέρει και πολύ από αντίστοιχα προγράμματα αυτής της κατηγορίας. Στο κέντρο της οθόνης υπάρχει η καρτέλα γραφής, επεξεργασίας κώδικα και από πάνω της βλέπουμε το όνομα του αρχείου που επεξεργαζόμαστε. Στην κορυφή του προγράμματος οι χρήστες μπορούν να αλλάξουν τις ρυθμίσεις του προγράμματος και του κώδικα από την γραμμή μενού και ρυθμίσεων.

The screenshot shows the Visual Studio interface with a C# script file open. The file name is 'medusaattak.cs'. The code implements a series of IEnumerator methods for attacking, including 'dilaybreak()', 'dilayspace()', and 'dilay()'. It uses Unity components like Player, CircleCollider2D, Rigidbody2D, and BoxCollider2D. A red box highlights the word 'dilay()' in the code. The status bar at the bottom right shows 'Ln: 170 Ch: 60 SPC CRLF'. A watermark 'Συγγραφή κώδικα' (Code Writing) is overlaid in the center of the code area.

```
yield return new WaitUntil(() => player.GetComponent<Jump>().IsGrounded() == true);

player.GetComponent<CircleCollider2D>().enabled = false;
player.transform.GetComponent<Rigidbody2D>().bodyType = RigidbodyType2D.Static;

}

public IEnumerator dilaybreak()
{
    yield return new WaitForSeconds(0.6f);
    msg.SetActive(true);
}

public IEnumerator dilayspace()
{
    yield return new WaitForSeconds(0.8f);
    times++;
}

public IEnumerator dilay()
{
    yield return new WaitForSeconds(1.0f);
    msg.SetActive(false);
    fire.SetActive(true);
    player.GetComponent<Move>().enabled = true;
    player.GetComponent<Jump>().enabled = true;
    attak.GetComponent<BoxCollider2D>().enabled = true;

    player.transform.GetComponent<Rigidbody2D>().bodyType = RigidbodyType2D.Dynamic;
    player.GetComponent<CircleCollider2D>().enabled = true;
}

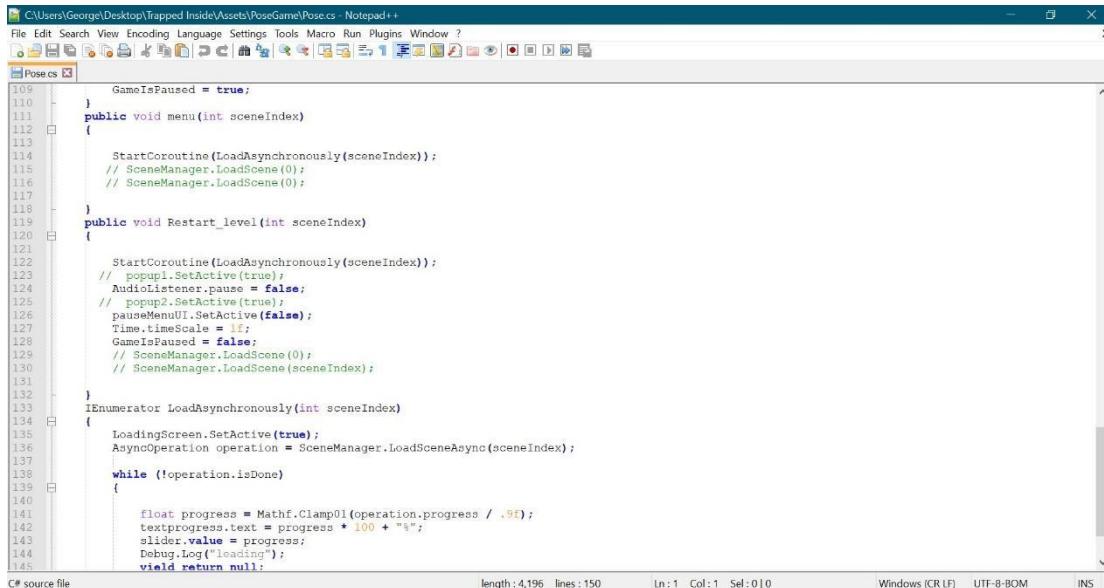
}

public IEnumerator space()
{
```

EIKONA 2.5.1: Visual Studio Workspace

2.6 NOTEPAD++

Ένα εναλλακτικό πρόγραμμα για την συγγραφή του κώδικα που χρησιμοποιήθηκε είναι το Notepad++. Είναι ένα πρόγραμμα επεξεργασίας κειμένου-κώδικα και αναπτύχθηκε το 2003 από τον Don Ho. Η πρώτη φορά που χρησιμοποιήθηκε ήταν για το JEXT που ήταν ένα πρόγραμμα επεξεργασίας κειμένου βασισμένο σε Java. Έχει την δομή ενός κλασικού προγράμματος επεξεργασίας κειμένου αλλά παρόλα αυτά παρέχει εξειδικευμένες λειτουργίες όπως η αλλαγή κωδικοποίησης του κειμένου, η αναγνώριση και εκτέλεση πολλών γλώσσεων προγραμματισμού, η μετατροπή χαρακτήρων ASCII σε δεκαδικό σύστημα γραφής και άλλα.



The screenshot shows the Notepad++ interface with a C# source file named "Pose.cs" open. The code in the file is as follows:

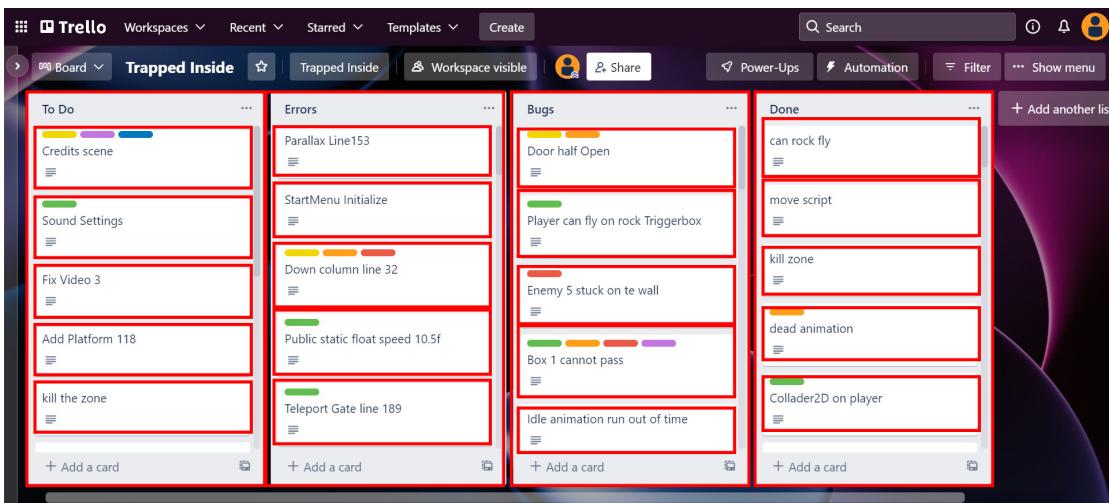
```
109     GameIsPaused = true;
110 }
111 public void menu(int sceneIndex)
{
112     StartCoroutine(LoadAsynchronously(sceneIndex));
113     // SceneManager.LoadScene(0);
114     // SceneManager.LoadScene(0);
115 }
116
117 }
118 public void Restart_level(int sceneIndex)
{
119     StartCoroutine(LoadAsynchronously(sceneIndex));
120     // popup1.SetActive(true);
121     // AudioListener.pause = false;
122     // popup2.SetActive(true);
123     pauseMenuUI.SetActive(false);
124     Time.timeScale = 1f;
125     GameIsPaused = false;
126     // SceneManager.LoadScene(0);
127     // SceneManager.LoadScene(sceneIndex);
128 }
129
130 IEnumerator LoadAsynchronously(int sceneIndex)
{
131     LoadingScreen.SetActive(true);
132     AsyncOperation operation = SceneManager.LoadSceneAsync(sceneIndex);
133
134     while (!operation.isDone)
135     {
136         float progress = Mathf.Clamp01(operation.progress / .9f);
137         textprogress.text = progress * 100 + "%";
138         slider.value = progress;
139         Debug.Log("loading");
140         yield return null;
141     }
142 }
143
144 }
145 }
```

The status bar at the bottom indicates the file is a "C# source file" with a length of 4,196 lines, and the current line is 150. The encoding is Windows (CR LF) and the file is saved in UTF-8 BOM.

EIKONA 2.6.1: Notepad++ Workspace

2.7 TRELLO

Για την καλύτερη οργάνωσή του παιχνιδιού χρησιμοποιήθηκε το Trello. Αναπτύχθηκε από την Trello Enterprise το 2011 και σκοπός του είναι η καλύτερη διαχείριση ενός πλάνου μιας εταιρίας ή κάποιο project. Το πρόγραμμα είναι δικτυακό και η χρήση του είναι πολύ απλή. Στην αρχική οθόνη οι χρήστες μπορούν να δημιουργήσουν θεματικές καρτέλες που μέσα τους μπορούν να εισαχθούν κάποια γεγονότα που είναι σχετικά με την θεματική της εκάστοτε καρτέλας. Στην παρακάτω εικόνα έχουν δημιουργηθεί τέσσερις θεματικές καρτέλες και κάθε μια από αυτές περιέχει καθήκοντα που πρέπει να γίνουν ή που έχουν ολοκληρωθεί.

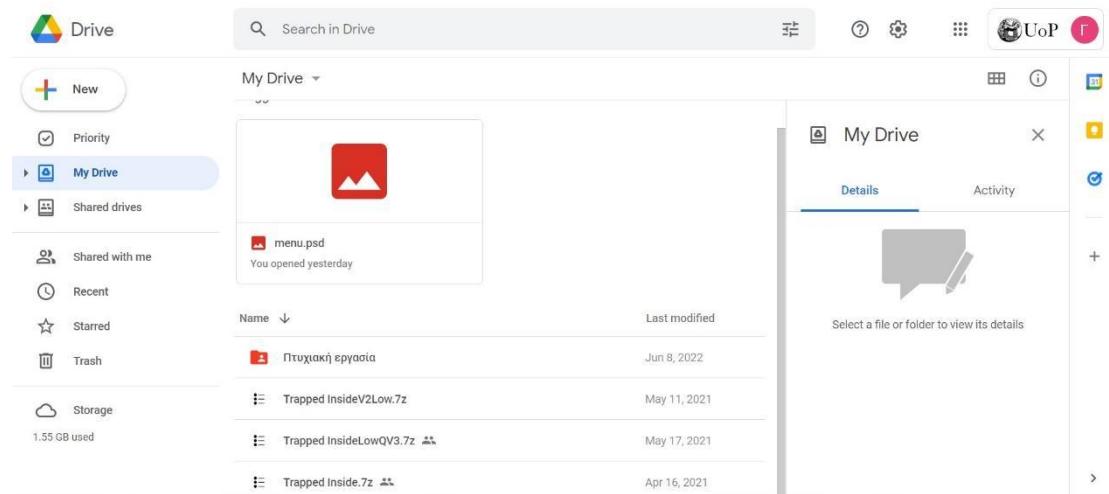


EIKONA 2.7.1: Trello Workspace

Όταν ένα από τα καθήκοντα ολοκληρωθεί π.χ. το Parallax line153 από την καρτέλα με τα Errors μπορούμε το μεταφέρουμε στην καρτέλα με τα ολοκληρωμένα (Done). Με αυτόν τον τρόπο γνωρίζουμε πως το πρόβλημα στο script με όνομα Parallax και στην σειρά 153 έχει διορθωθεί χωρίς να χρειάζεται να σαρώνουμε συνεχώς των κώδικα.

2.8 GOOGLE DRIVE

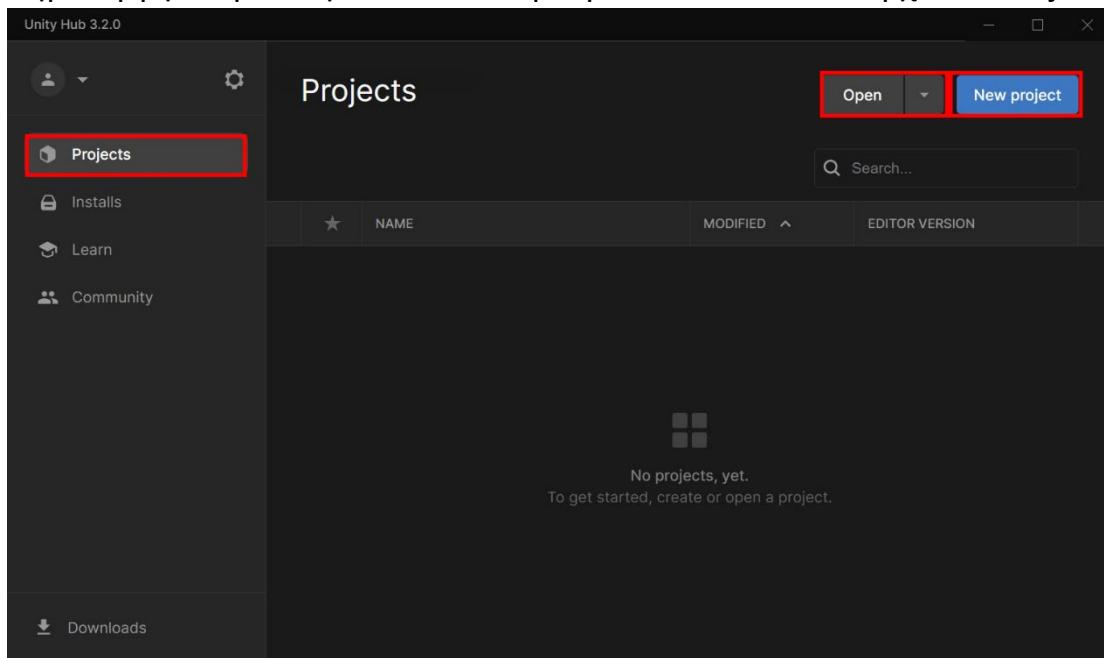
To Google drive είναι μια αρκετά γνωστή υπηρεσία σύννεφο-cloud αποθήκευσης και συγχρονισμού αρχείων που κυκλοφόρησε από την Google το έτος 2012. Οι χρήστες μπορούν να βάζουν τα αρχεία του στο Google drive με την ασφάλεια της Google και να έχουν πρόσβαση σε αυτά οποιαδήποτε στιγμή και από οποιαδήποτε συσκευή θελήσουν, το μόνο που χρειάζεται είναι να γίνει μια απλή σύνδεση στην διαδικτυακή πλατφόρμα. Οι χρήστες μπορούν επίσης να δώσουν πρόσβαση στα αρχεία που επιθυμούν σε τρίτα άτομα και να επιλέξουν αν αυτοί οι χρήστες θα έχουν το δικαίωμα να επεξεργαστούν τα αρχεία ή απλά να έχουν μια περιορισμένη πρόσβαση μόνο ανάγνωσης.



EIKONA 2.8.1: Google Drive Workspace

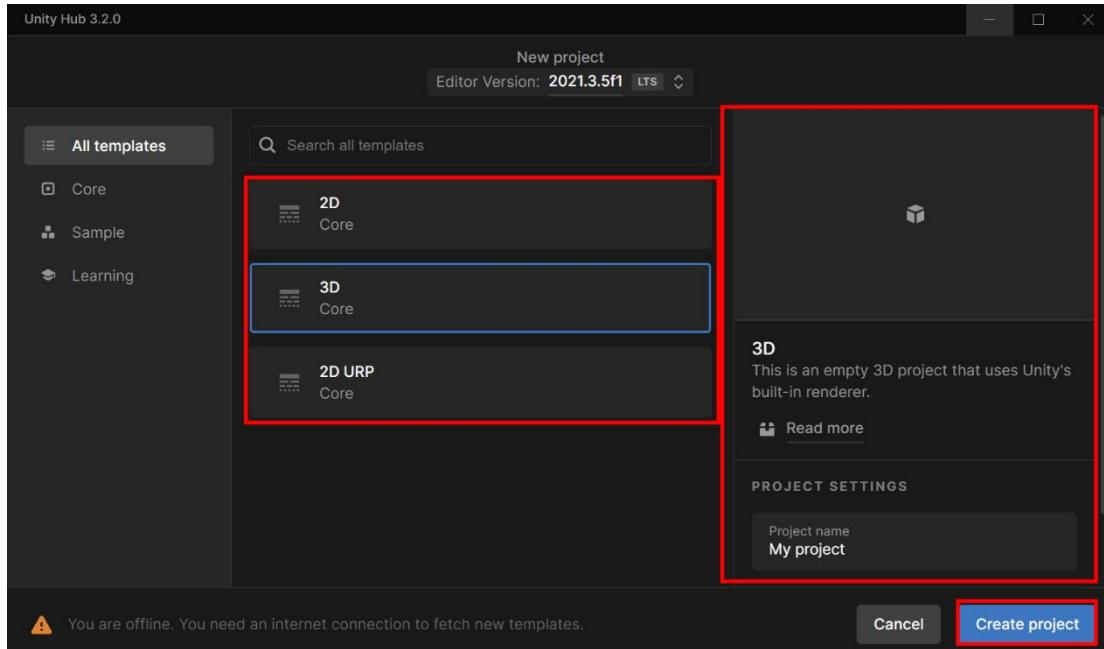
2.9 UNITY

Το τελευταίο και κυριότερο πρόγραμμα που χρησιμοποιήθηκε ήταν μηχανή γραφικών Unity η οποία αναπτύχθηκε από την Unity Technologies το έτος 2005. Από τότε πολλές εταιρείες την χρησιμοποιούν για την κατασκευή παιχνιδιών σε τηλέφωνα, υπολογιστές και κονσόλες. Γνωστά παιχνίδια όπως το Doom, Call of duty mobile, Among us, Beat saber έχουν αναπτυχθεί μέσω αυτής της μηχανής. Το περιβάλλον του προγράμματος φαντάζει περίπλοκο και δυσνόητο αλλά η πραγματικότητα διαφέρει. Μόλις εγκατασταθεί η Unity από την επίσημη ιστοσελίδα, στην καρτέλα Projects μας εμφανίζονται οι επιλογές του να δημιουργήσουμε ή να εισάγουμε κάποιο υπάρχον Project.



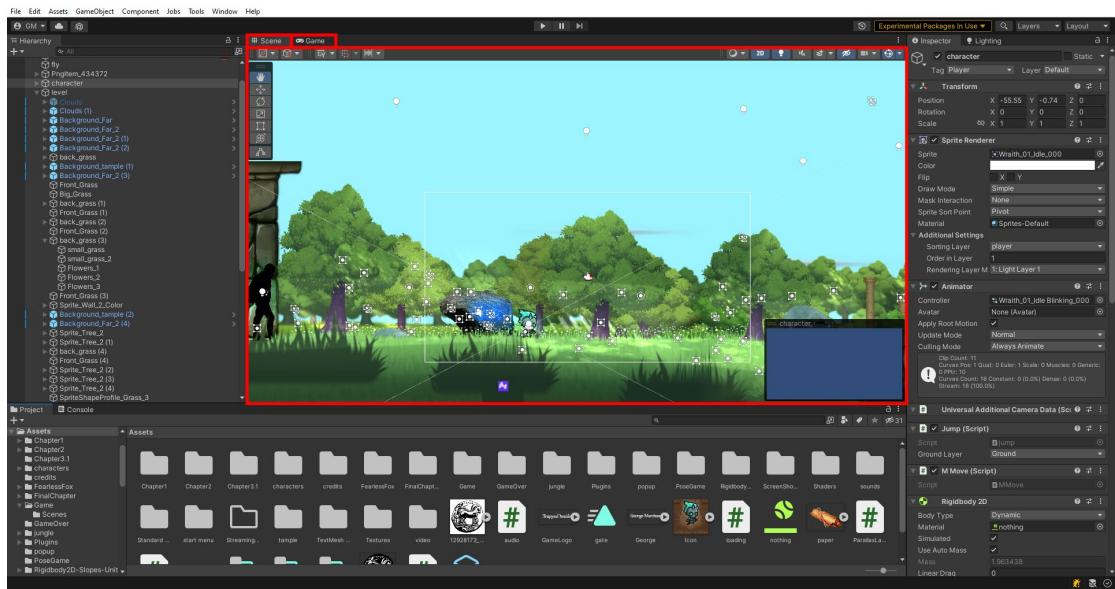
EIKONA 2.9.1: Unity Select Projects

Αν επιλέξουμε την δημιουργία ενός νέου Project τότε θα πρέπει να διαλέξουμε το στυλ και το ύφος του παιχνδιού που σκοπεύουμε να δημιουργήσουμε, να δώσουμε ένα όνομα και να προχωρήσουμε στην ανάπτυξη του.



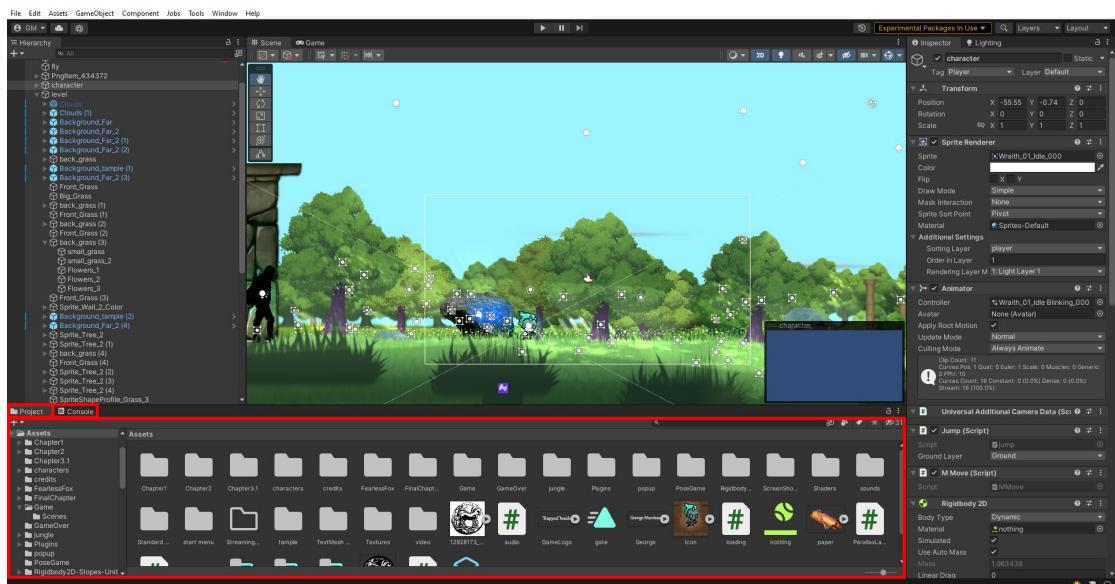
EIKONA 2.9.2: Unity New Project

Εμείς θα επιλέξουμε να ανοίξουμε ένα υπάρχον 2D Project και συγκεκριμένα το παιχνίδι που παρουσιάστηκε στο κεφάλαιο 1.3. Το περιβάλλον που θα εργαστούμε αποτελείται από το σκηνικό, εκεί στήνουμε το σκηνικό δηλαδή όλη την σχεδίαση και το σετ του παιχνιδιού, στην διπλανή καρτέλα βρίσκεται η προεπισκόπηση όπου μας δείχνει το πως θα είναι το τελικό αποτέλεσμα όσων έχουμε σχεδιάσει.



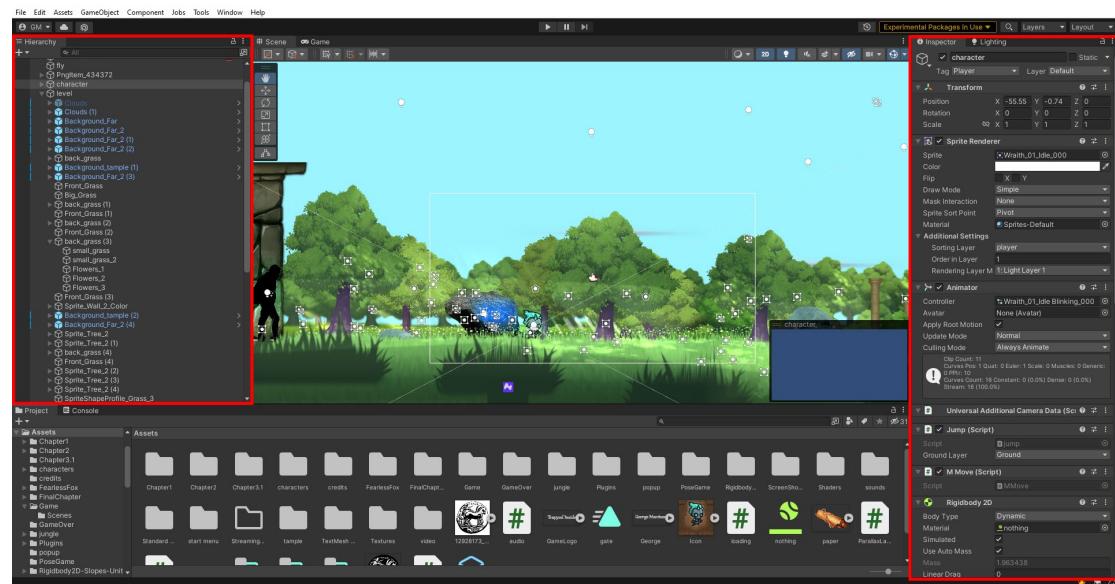
EIKONA 2.9.3: Unity Scene-Game

Χαμηλά στην καρτέλα project υπάρχουν τα αρχεία που χρησιμοποιούμε ή που σκοπεύουμε να χρησιμοποιήσουμε. Αυτά μπορεί να είναι αντικείμενα, animations, scripts, τα γραφικά ενός αντικειμένου, τους ήχους ακόμα και τους κανόνες της φυσικής που θα βασιστεί το παιχνίδι. Επίσης εκεί βρίσκεται και η κονσόλα όπου μας εμφανίζει τυχόν σφάλματα που έχουν προκύψει είτε σε κάποιο αρχείο κώδικα είτε σε κάποιο αντικείμενο ή ρύθμιση.



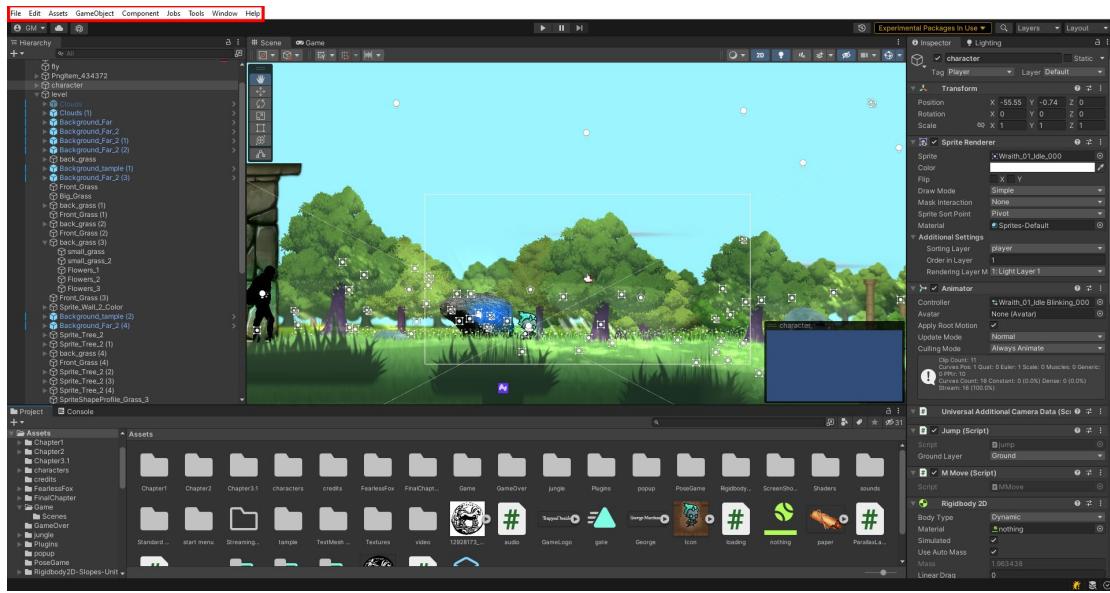
EIKONA 2.9.4: Unity Project-Console

Αριστερά βρίσκονται τα αντικείμενα, τα εφέ και οι φωτισμοί που έχουν εισαχθεί και χρησιμοποιούνται στο σκηνικό. Όταν επιλεχθεί ένα αντικείμενα από το σκηνικό, στα δεξιά εμφανίζονται οι ιδιότητες του όπου μπορούμε να πειράξουμε και να επεξεργαστούμε. Όπως βλέπουμε στην παρακάτω εικόνα έχει επιλεχθεί ο πρωταγωνιστής του παιχνιδιού. Στις ιδιότητες του υπάρχει το Sprite Renderer που με αυτό ορίζουμε την όψη του αντικειμένου. Το Animator που προσθέτουμε στο αντικείμενο μια κίνηση ένα animation, τα script με όνομα Universal additional Camera Data, MMove και jump που μέσα τους περιέχεται ο κώδικας για την μετακίνηση του χαρακτήρα, το άλμα και την σύνδεση του με την κάμερα όπου μέσο αυτής βλέπει ο παίκτης το παιχνίδι. Τέλος υπάρχει το Rigidbody όπου από εκεί ορίζουμε την τριβή, την βαρύτητα και το βάρος του αντικειμένου ως προς το έδαφος.



EIKONA 2.9.5: Unity Assets-Inspector

Τέλος στο πάνω μέρος υπάρχει η γραμμή εργαλείων όπου από εκεί βλέπουμε τις γενικότερες ρυθμίσεις του Project όπως την ποιότητα των γραφικών αλλά και εξειδικευμένες ρυθμίσεις όπως την επεξεργασία αλλαγής σκηνικών. Επίσης από εκεί μπορούμε να κατεβάσουμε κάποια έξτρα εργαλεία (Plug-in) που προσφέρουν περισσότερη ευελιξία στην ανάπτυξη του παιχνιδιού.

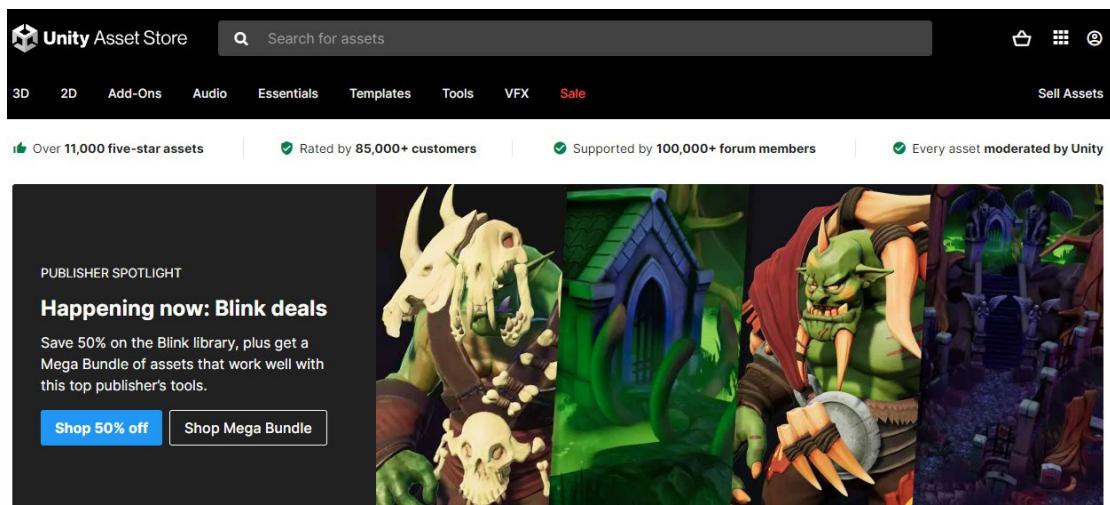


EIKONA 2.9.6: Unity Toolbar

ΚΕΦΑΛΑΙΟ 3 ΥΛΟΠΟΙΗΣΗ

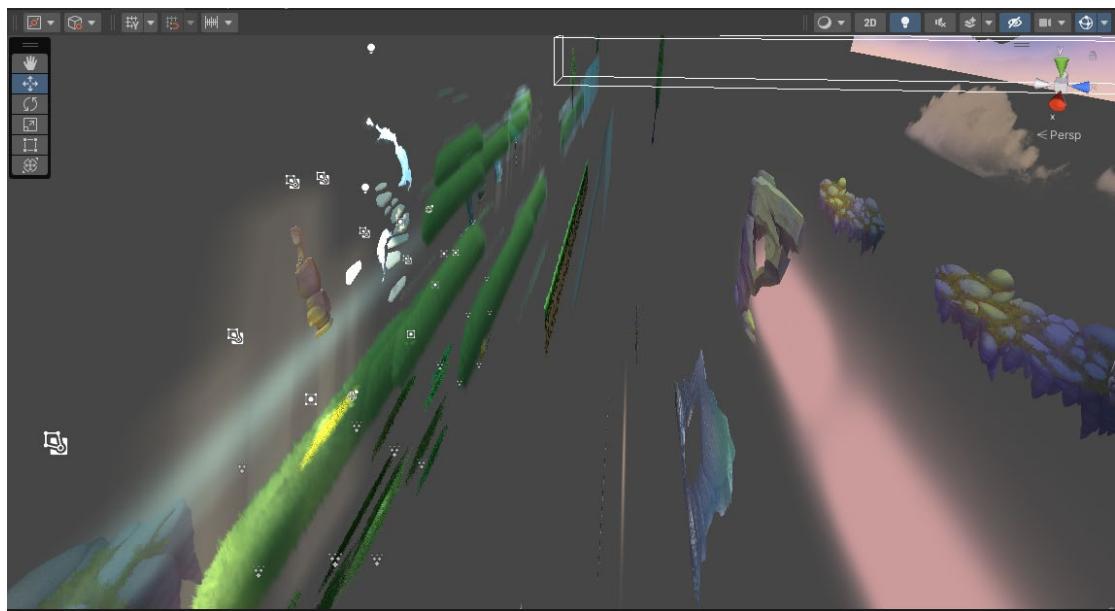
3.1 ΑΝΤΙΚΕΙΜΕΝΑ (OBJECTS)

Αρχικά να σημειώσουμε πως επειδή αναφερόμαστε σε παιχνίδι δύο διαστάσεων θα ασχοληθούμε μόνο με τον άξονα X που είναι το πλάτος και τον άξονα Z που είναι το ύψος καθώς εκεί βασίζεται όλη η φιλοσοφία των 2D γραφικών. Επίσης είναι σημαντικό να αναφέρουμε πως κάποια από τα objects που χρησιμοποιήθηκαν στο παιχνίδι παρέχονται δωρεάν από το ηλεκτρονικό κατάστημα της Unity.



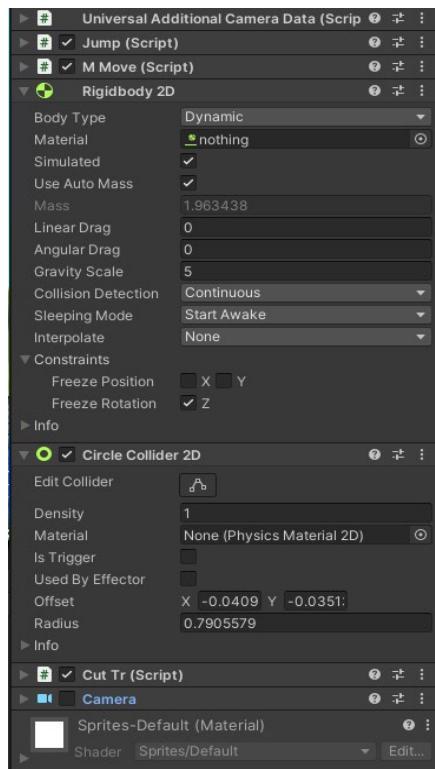
EIKONA 3.1.1: Unity Asset Store

Αφού σχεδιάστηκαν τα αντικείμενα ή αλλιώς objects όπως ονομάζονται στην Unity μέσω του προγράμματος photoshop που αναφέρθηκε στο προηγούμενο κεφάλαιο και εισήχθησαν στο σκηνικό θα χρειαστεί να τους ορίσουμε το στρώμα στο οποίο θα τοποθετηθούν. Το παιχνίδι χωρίζεται σε κάποια στρώματα (Layers) που ορίζονται από τον χρήστη και υπάρχουν για τον σωστό διαχωρισμό των αντικειμένων. Για παράδειγμα ένας βράχος που θέλουμε να φαίνεται πως είναι πίσω από τον παίκτη δεν θα μπεί στο ίδιο στρώμα με αυτό του παίκτη αλλά θα μπεί σε ένα στρώμα που βρίσκεται πριν αυτό του παίκτη.



ΕΙΚΟΝΑ 3.1.2: Unity Layers

Αν το αντικείμενο προορίζεται να μετακινηθεί, όπως για παράδειγμα ο πρωταγωνιστής θα χρειαστεί να προσθέσουμε κάποιες ιδιότητες. Συγκεκριμένα στον πρωταγωνιστή είναι ορισμένο το βάρος του και η τριβή του ως προς το έδαφος με την βοήθεια του Rigidbody 2D, ένα Circle Collider 2D που με αυτό ορίζεται το σώμα του ώστε να μην μπορεί να περάσει μέσα από άλλα αντικείμενα, οι κινήσεις (Animations) που χρησιμοποιεί, η κάμερα που είναι συνδεδεμένος ώστε να όταν κινείται ο χαρακτήρας να τον ακολουθεί και τα αρχεία με τους κώδικες που χρησιμοποιεί.



EIKONA 3.1.3: Unity Character Inspector

3.2 ΦΩΤΙΣΜΟΣ

Μετά την εισαγωγή των αντικειμένων στο σκηνικό θα χρειαστεί να προστεθεί ο κατάλληλος φωτισμός. Χωρίς την εισαγωγή του σωστού φωτισμού το περιβάλλον του παιχνιδιού είναι σκοτεινό και αφιλόξενο για τον παίκτη. Στην παρακάτω εικόνα τα δέντρα και το έδαφος έχουν επεξεργαστεί και έχουν χρωματιστεί καταλλήλως παρόλα αυτά φαίνονται μαύρα.



EIKONA 3.2.1: Unity Lights Off

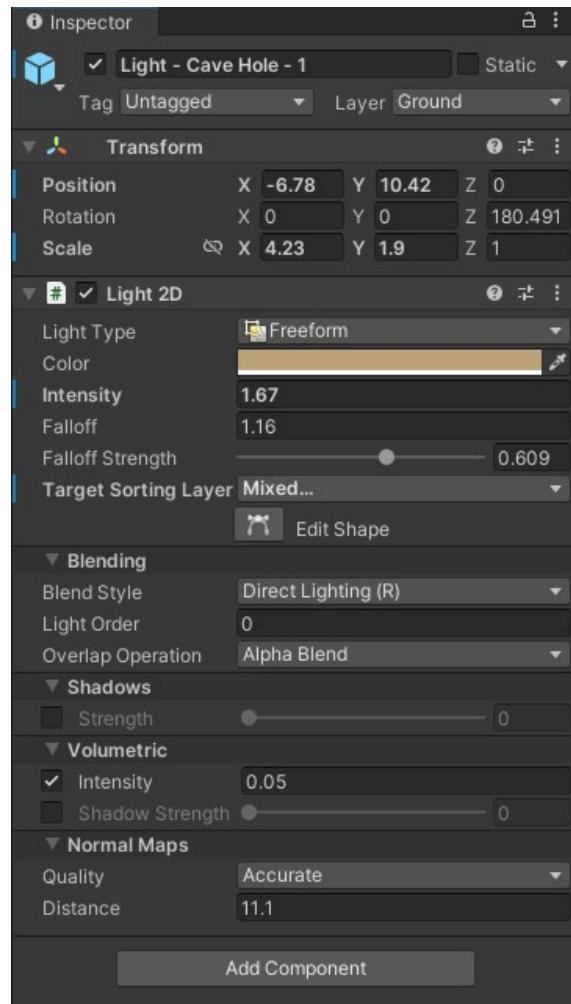
Η Unity προσφέρει εργαλεία εισαγωγής και επεξεργασίας φωτισμών τόσο για παιχνίδια με τρισδιάστατα γραφικά όσο και για δισδιάστατα. Οι δισδιάστατοι φωτισμοί χωρίζονται σε τέσσερις κατηγορίες: Sprite, spot, global και freedom light. Το sprite light είναι μία λάμψη που φωτίζει ένα κομμάτι της επιθυμητής επιφάνειας και συνήθως χρησιμοποιείται για να δημιουργηθεί το εφέ της αντανάκλασης του ήλιου πάνω σε μια επιφάνεια. Το spot light φωτίζει ένα περιορισμένο σημείο μεγάλης ή μικρής εμβέλειας και συνήθως χρησιμοποιείται για την φωταγώγηση μια λάμπας ή ενός προβολέα. Το global light φωτίζει όλο το σκηνικό και όλα τα αντικείμενα που βρίσκονται σε αυτό. Χρησιμοποιείται επίσης για την προσομοίωση του ηλίου σε έναν εξωτερικό χώρο. Τέλος, το freedom light φωτίζει μια περιορισμένη περιοχή, ωστόσο το σχέδιο του φωτός που

εκπέμπεται μπορεί διαμορφωθεί σε διαφορετικές μορφές. Συνήθως το freedom light χρησιμοποιείται για την δημιουργία μιας ακτίνας φωτός ή την προσομοίωσης της διάθλασης του φωτός.



EIKONA 3.2.2: Unity Lights On

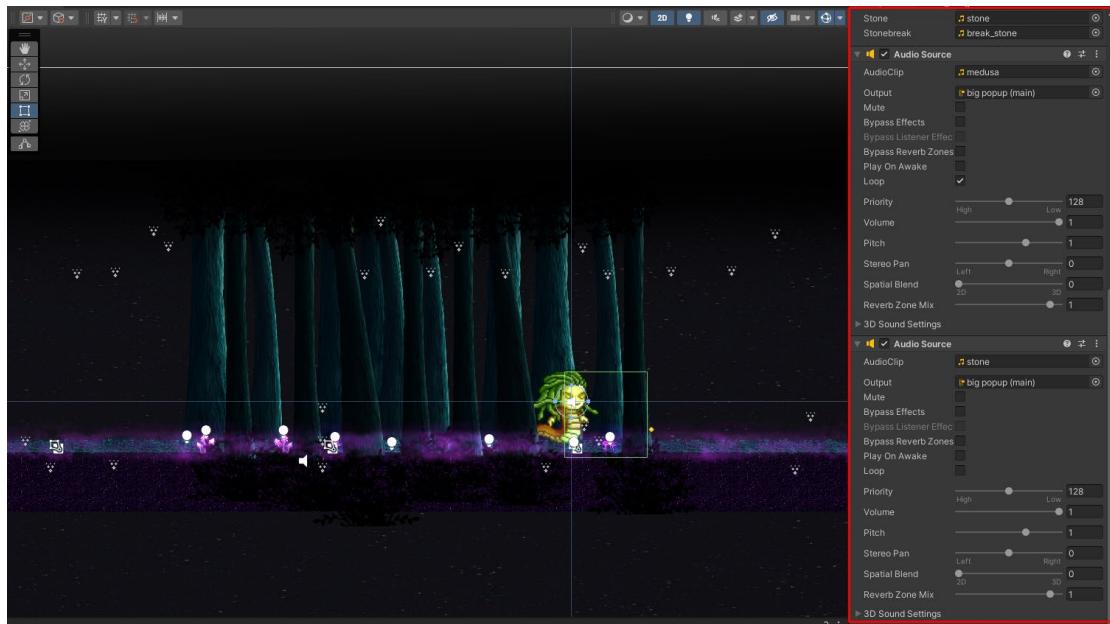
Σε κάθε είδος φωτός που έχει προστεθεί στο σκηνικό του παιχνιδιού, μας δίνεται η δυνατότητα να αυξομειώσουμε την φωτεινότητα, να επιλέξουμε τα αντικείμενα που θα επηρεαστούν από τους φωτισμούς, να ρυθμίσουμε το εύρος και να αλλάξουμε το χρώμα του φωτός.



EIKONA 3.2.3: Unity Light Inspector

3.3 ΗΧΟΣ

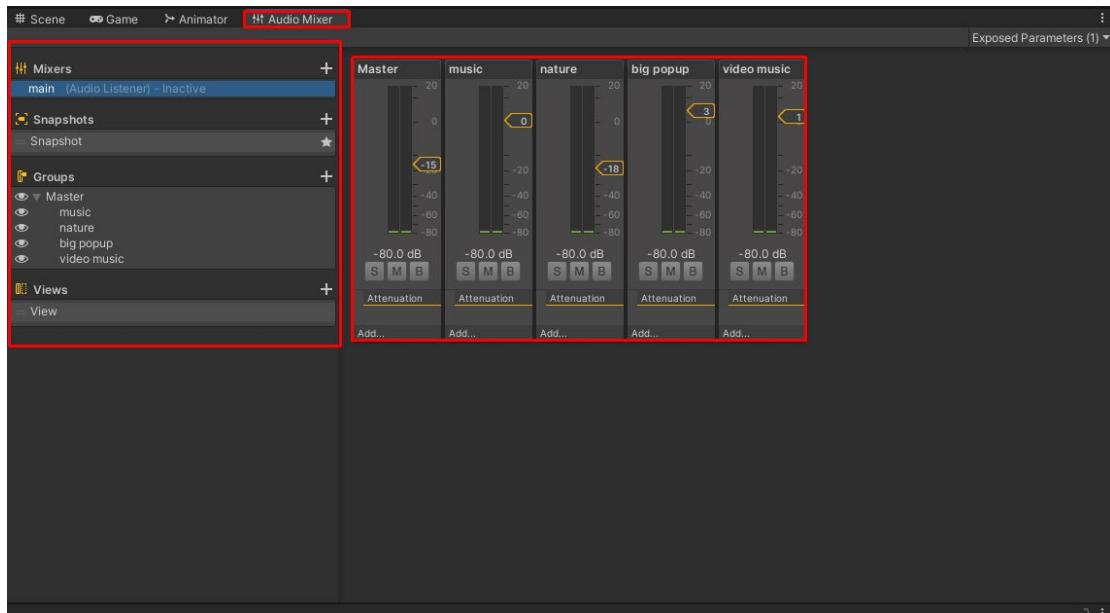
Οι ήχοι προσφέρουν μια ευχάριστη νότα και κάνουν τον κόσμο των παιχνιδιών να φαντάζει περισσότερο ζωντανός, διασκεδαστικός και προσιτός στους παίκτες. Για αυτόν τον λόγο, με την βοήθεια του Audacity, που αναφέρθηκε σε προηγούμενο κεφάλαιο, και της Unity, που παρέχει αρκετούς μηχανισμούς για την διαχείριση και αξιοποίηση των ήχων προστέθηκε μουσική και ηχητικά εφέ στο παιχνίδι της παρούσας πτυχιακής. Αφού γίνει η επεξεργασία των ήχων μέσω του Audacity και αφού εισαχθούν μέσα στο project χρειάζεται να προσθέσουμε τον εκάστοτε ήχο μέσα στο σκηνικό και να προγραμματίσουμε το σενάριο που θα ενεργοποιηθεί, τον χρόνο που θα διαρκέσει, την ένταση αλλά και το πόσες φορές θα επαναληφθεί η αναπαραγωγή του ήχου. Στην παρακάτω εικόνα βλέπουμε την μέδουσα να είναι επιλεγμένη και στο δεξί μέρος υπάρχουν δύο Audiosource. Το Audiosource είναι ένα εργαλείο που προσφέρει η Unity με σκοπό την εισαγωγή ενός ήχου σε ένα αντικείμενο και την διαχείριση του. Συγκεκριμένα μας δίνει την επιλογή να συνδέσουμε τον ήχο με ένα κανάλι εξόδου ώστε να υπάρξει η αναπαραγωγή του, να βάλουμε τον ήχο στο αθόρυβο, την προσθήκη φίλτρου παράκαμψης για την μαζική ενεργοποίηση ή απενεργοποίηση των ήχων, την δημιουργία αντίλαλου, η επιλογή του να ενεργοποιείται ο ήχος κατά την έναρξη του παιχνιδιού και η δυνατότητα της συνεχόμενης αναπαραγωγής του εκάστοτε ήχου. Επίσης υπάρχει η επιλογή διαχείρισης της προτεραιότητας του ήχου. Για παράδειγμα μπορούμε να βάλουμε υψηλή προτεραιότητα σε έναν συγκεκριμένο ήχο ώστε να ακούγεται περισσότερο από τους υπόλοιπους ήχους του παιχνιδιού. Η αλλαγή έντασης ήχου, η αλλαγή της ταχύτητας του, η ρύθμιση του στερεοφωνικού, η διαχείριση του τρισδιάστατου ήχου και η ποσότητα του αντίλαλου του ήχου.



EIKONA 3.3.1: Unity Audiosource

Όλες οι παραπάνω λειτουργίες μπορούν να αλλάξουν μέσω κώδικα ανάλογα με τα σενάρια που έχουν δημιουργηθεί. Κάποιες βασικές εντολές είναι η AudioSource audioSourceStone; που δηλώνεται η μεταβλητή με όνομα audioSourceStone ως AudioSource, με την εντολή AudioClip stone; ορίζεται πως στην μεταβλητή stone θα αποθηκευτεί ένα αρχείο ήχου. Με την εντολή audioSourceStone = GetComponent<

αφου επιλεχθεί εμφανίζεται το γραφικό του περιβάλλον. Όπως δείχνει παρακάτω η εικόνα, στο αριστερό μέρος βρίσκονται οι ομαδοποιημένες ενότητες. Στο παράδειγμα μας αποτελούνται από τα Mixers που μέσα του περιέχει το Master για την μίξη των ήχων. Το Snapshots που μέσα έχει καταγεγραμμένες ρυθμίσεις όλων των παραμέτρων των ήχων, τα Groups που έχουν όλα τα κανάλια εξόδων όπως το Master, music, nature, big popup και video music. Τέλος βρίσκουμε τα Views που έχουν προφίλ με διαφορετικά πακέτα ρυθμίσεων.



EIKONA 3.3.2: Unity Audio Mixer

Το αρχείο κώδικα C# που ακολουθεί είναι από το τελευταίο chapter του παιχνιδιού, συγκεκριμένα χρησιμοποιείται από της πλατφόρμες στο έδαφος και περιέχει κομμάτια κώδικα που αναφέρθηκαν παραπάνω.

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

public class button : MonoBehaviour
{
    private Animator anim;
    private int is_on_butt = 0;
    public GameObject light_butt;
    private bool box = false;
    private bool player = false;
    public AudioClip push;
    public AudioClip unpush;
    AudioSource audioSource;
    public float volume;
    private bool pressed;
    void Start()
    {
        audioSource = GetComponent<
```

```

if ((Player.gameObject.tag == "box") || (Player.gameObject.tag ==
"Player"))

{
    if (Player.gameObject.tag == "box")

    {
        if (!audioSource.isPlaying)
            audioSource.PlayOneShot(push, volume);

        box = true;
    }

    if (Player.gameObject.tag == "Player")
    {
        player = true;
    }

    if ((box) && (player))

    {
        anim.SetBool("is_on", true);
        light_butt.SetActive(true);
    }

    else

    {
        anim.SetBool("is_on", true);
        is_on_butt++;
        light_butt.SetActive(true);
    }

    pressed = true;
}

```

```

}

void Update()
{
    if (!box && pressed)
    {
        if (player)
        {
            if (!audioSource.isPlaying)
            {
                audioSource.PlayOneShot(push, volume);
                pressed = false;
            }
        }
    }
}

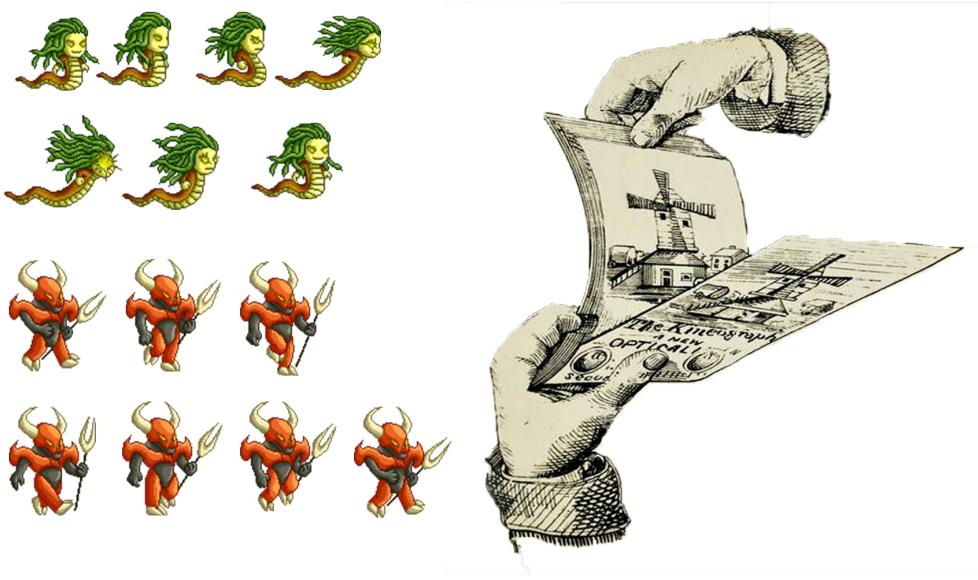
void OnTriggerEnter2D(Collider2D Player)
{
    if ((Player.gameObject.tag == "box") || (Player.gameObject.tag ==
"Player"))
    {
        if (Player.gameObject.tag == "box")
        {
            box = false;
            pressed = true;
        }
        if (Player.gameObject.tag == "Player")

```

```
{  
    player = false;  
    if (!box)  
        audioSource.PlayOneShot(unpush, volume);  
    }  
    if ((!box) && (!player))  
    {  
        is_on_butt--;  
        anim.SetBool("is_on", false);  
        light_butt.SetActive(false);  
    }  
    else  
    {  
        anim.SetBool("is_on", true);  
        light_butt.SetActive(true);  
    }  
}  
}
```

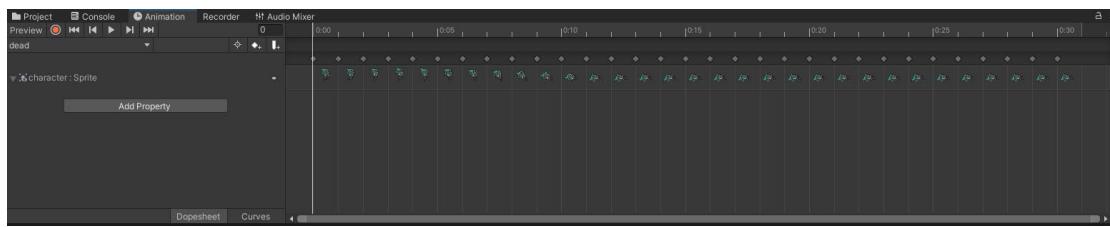
3.4 ΚΙΝΗΣΗ (ANIMATIONS)

Η κίνηση ορίζεται ως η μεταβολή της θέσης ενός αντικειμένου ή σώματος ως προς ένα σημείο αναφοράς και για την δημιουργία της αρκεί απλά κάποιος να κινήσει το χέρι του. Αυτά όσον αφορά τον φυσικό κόσμο. Σε μια οθόνη έχουμε να κάνουμε με απλές στατικές εικόνες, ο μοναδικός τρόπος για να δημιουργηθεί μία κίνηση είναι να γίνει μια γρήγορη εναλλαγή πολλών διαφορετικών εικόνων όπως ακριβώς γίνεται στα flip books.



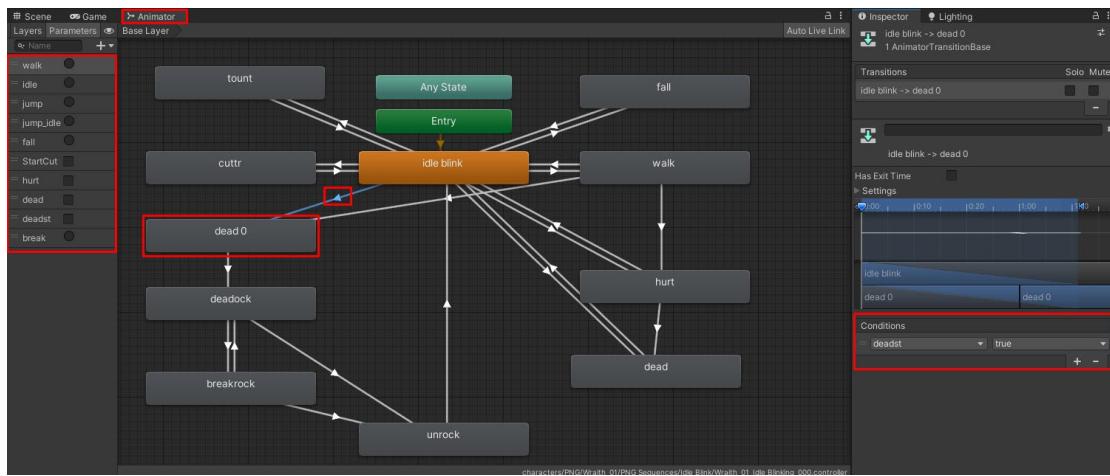
EIKONA 3.4.1: Animation

Για την δημιουργία της κίνησης κάποιου αντικειμένου στην unity όπως για παράδειγμα του χαρακτήρα χρησιμοποιήθηκε το εργαλείο Animation που παρέχεται από το ίδιο το πρόγραμμα. Αφού δημιουργηθούν οι εικόνες που θα χρησιμοποιηθούν για την ψευδαίσθηση της κίνησης, στην ενότητα Windows της γραμμής εργαλείων βρίσκεται η καρτέλα Animation. Εκεί εισήχθησαν οι εικόνες στα σωστά καρέ του χρόνου ώστε να φαίνεται η κίνηση του χαρακτήρα ομαλή.



EIKONA 3.4.2: Unity Animation Timeline

Η Unity προσφέρει και άλλο τρόπο δημιουργίας κίνησης. Οι χρήστες έχουν την δυνατότητα μέσω του Animation να κάνουν εγγραφή την κίνηση ενός αντικειμένου από το σκηνικό και να το αναπαράγουν συνέχεια. Αφού ολοκληρώθηκε η διαδικασία χρειάζεται με κάποιον τρόπο να ενωθούν όλα τα animations και να προγραμματιστούν. Αυτό γίνεται στην καρτέλα Animator. Το κάθε σύνολο από τις κινήσεις που έχουμε δημιουργήσει απεικονίζεται ως ορθογώνια. Στην παρακάτω εικόνα βλέπουμε πως υπάρχει μια σύνδεση μεταξύ όλων των animations του χαρακτήρα. Αυτό συμβαίνει για να μπορούν να συνδεθούν με ορισμένες μεταβλητές ώστε να μπορέσουμε να δημιουργήσουμε ένα σενάριο για το πότε θα ενεργοποιηθεί η εκάστοτε κίνηση. Όπως μπορούμε να διακρίνουμε στην εικόνα είναι επιλεγμένη η σύνδεση που ενώνει το Idle blink όπου μέσα περιέχει την κίνηση που ανοιγοκλείνει ο χαρακτήρας τα μάτια του με το dead 0 που είναι η κίνηση που εκτελείται όταν ο παίκτης χάνει. Στο αριστερό μέρος υπάρχουν οι μεταβλητές που χρησιμοποιούνται για την εκτέλεση της συνθήκης, ενώ στα δεξιά βλέπουμε πως προκειμένου να εκτελεστεί η συνθήκη που έχουμε επιλέξει πως θα πρέπει η μεταβλητή deadst να είναι πάντα true.



ΕΙΚΟΝΑ 3.4.3: Unity Animator

Ο κώδικας σε C# για την δημιουργία ενός σεναρίου που θα ενεργοποιήσει την συνθήκη που είδαμε παραπάνω είναι απλός. Αρχικά πρέπει να οριστεί το αντικείμενο και το animation του αντικειμένου που θα χρησιμοποιήσουμε, αυτό γίνεται με την εντολή GameObject Player; που στην προκειμένη περίπτωση είναι ο χαρακτήρας και Animator Player_Animation; για το animation του. Με την εντολή Player_Animation = Player.GetComponent<Animator>(); συνδέουμε το animation του χαρακτήρα με τον ίδιο τον χαρακτήρα. Ποιό αναλυτικά του λέμε να αποθηκεύσει όλο το Animator του χαρακτήρα δηλαδή όλο το σχεδιάγραμμα της παραπάνω εικόνας και να το αποθηκεύσει μέσα στο Player_Animation. Τέλος με την εντολή Player_Animation.SetBool("deadstst", true); του λέμε να βρει την μεταβλητή με όνομα deadst μεσα απο όλο το σχεδιάγραμμα (Animator) και να την κάνει true και έτσι θα εκτελεστεί η σύνδεση που είναι επιλεγμένη στην εικόνα. Παρακάτω ακολουθεί ένα κομμάτι κώδικα από το παραχνίδι που είναι υπεύθυνος για την κίνηση του χαρακτήρα στον άξονα X με την χρήση των κουμπιών A και D που βασίζεται πάνω σε αυτην την λογική.

```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class Move : MonoBehaviour
{
    Vector3 characterScale;
    float characterScaleX;
    private Animator ani;
    public static float speed = 10;
    private int speed2;
    private float buttonPressedTime;
    void Start()
    {
        characterScale = transform.localScale;
        characterScaleX = characterScale.x;
        ani = GetComponent<Animator>();
        ani.SetBool("StartCut", false);
    }

    void Update()
    {
        transform.Translate(Input.GetAxis("Horizontal") * speed * speed2 *
Time.deltaTime, 0f, 0f);

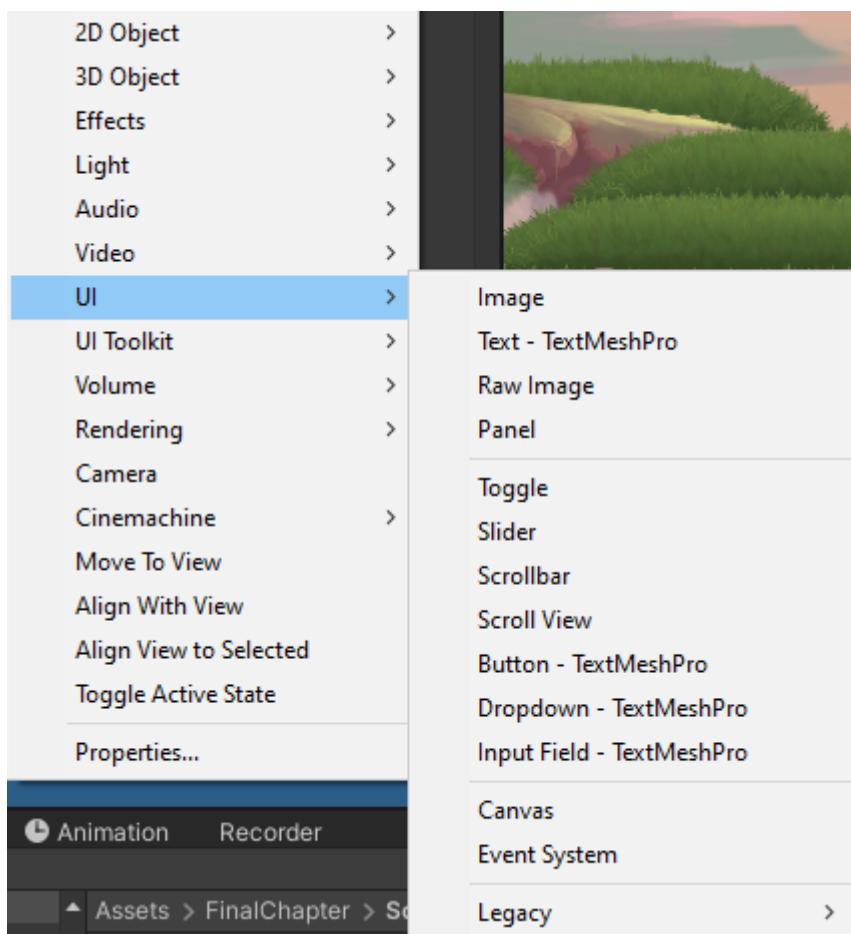
        if (Input.GetKeyUp("a") || Input.GetKeyUp("d") ||
(Input.GetKey("a") && Input.GetKey("d")) || (Input.GetKey("d") &&
Input.GetKey("a")))
    }
}

```

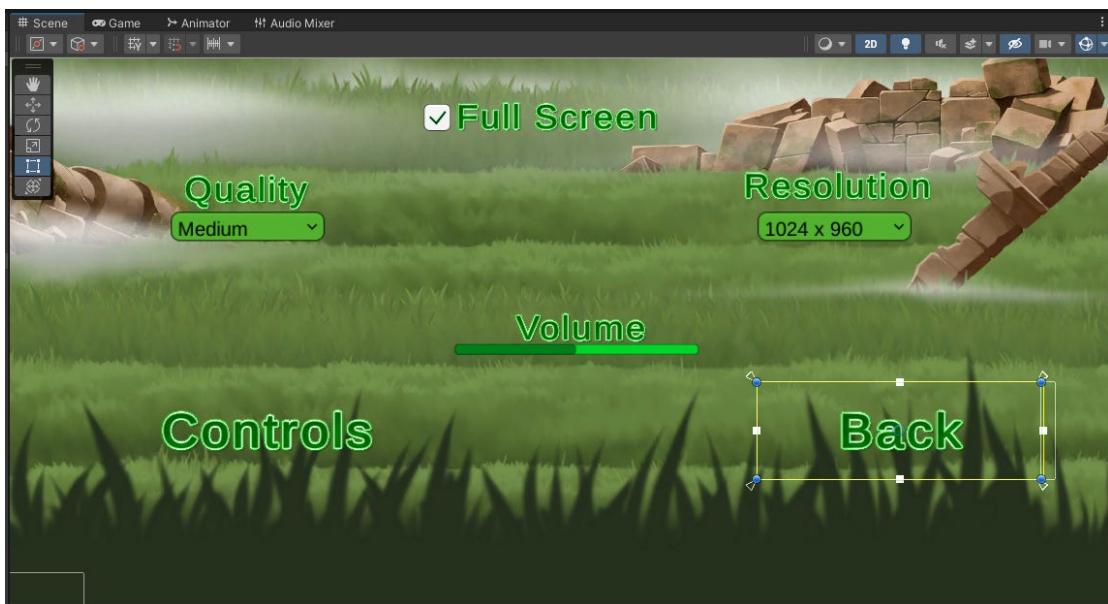
```
{  
    speed2 = 0;  
    ani.SetTrigger("idle");  
    ani.ResetTrigger("walk");  
}  
  
else if (Input.GetKeyDown("a"))  
{  
    speed2 = 1;  
    ani.ResetTrigger("idle");  
    ani.SetTrigger("walk");  
    characterScale.x = -characterScaleX;  
}  
  
else if (Input.GetKeyDown("d"))  
{  
    speed2 = 1;  
    ani.ResetTrigger("idle");  
    ani.SetTrigger("walk");  
    characterScale.x = characterScaleX;  
}  
  
transform.localScale = characterScale;  
}  
  
}
```

3.5 ΔΙΕΠΑΦΗ ΜΕ ΤΟΝ ΧΡΗΣΤΗ. (UI)

Η διεπαφή με τον χρήστη ή αλλιώς user interface(UI) είναι απαραίτητη για ένα πρόγραμμα ή παιχνίδι καθώς είναι ένα τρόπος για να μπορέσει ο χρήστης να διαχειριστεί και να αλληλεπιδράσει με τα στοιχεία που βρίσκονται στην οθόνη του. Αρχικά πρέπει να κατανοήσουμε τον ορισμό του UI συστήματος. Στην μηχανή γραφικών Unity το UI αποτελείται από ένα σύνολο εργαλειών όπως κουμπιά, εισαγωγή κειμένου, κουμπιά επιλογής, μπάρες αυξομείωσης, πτυσσόμενα μενού και ότι άλλο χρειαζόμαστε για την δημιουργία ενός μενού, μια ρύθμισης ακόμα και για την εισαγωγή ενός βίντεο ή φωτογραφίας.



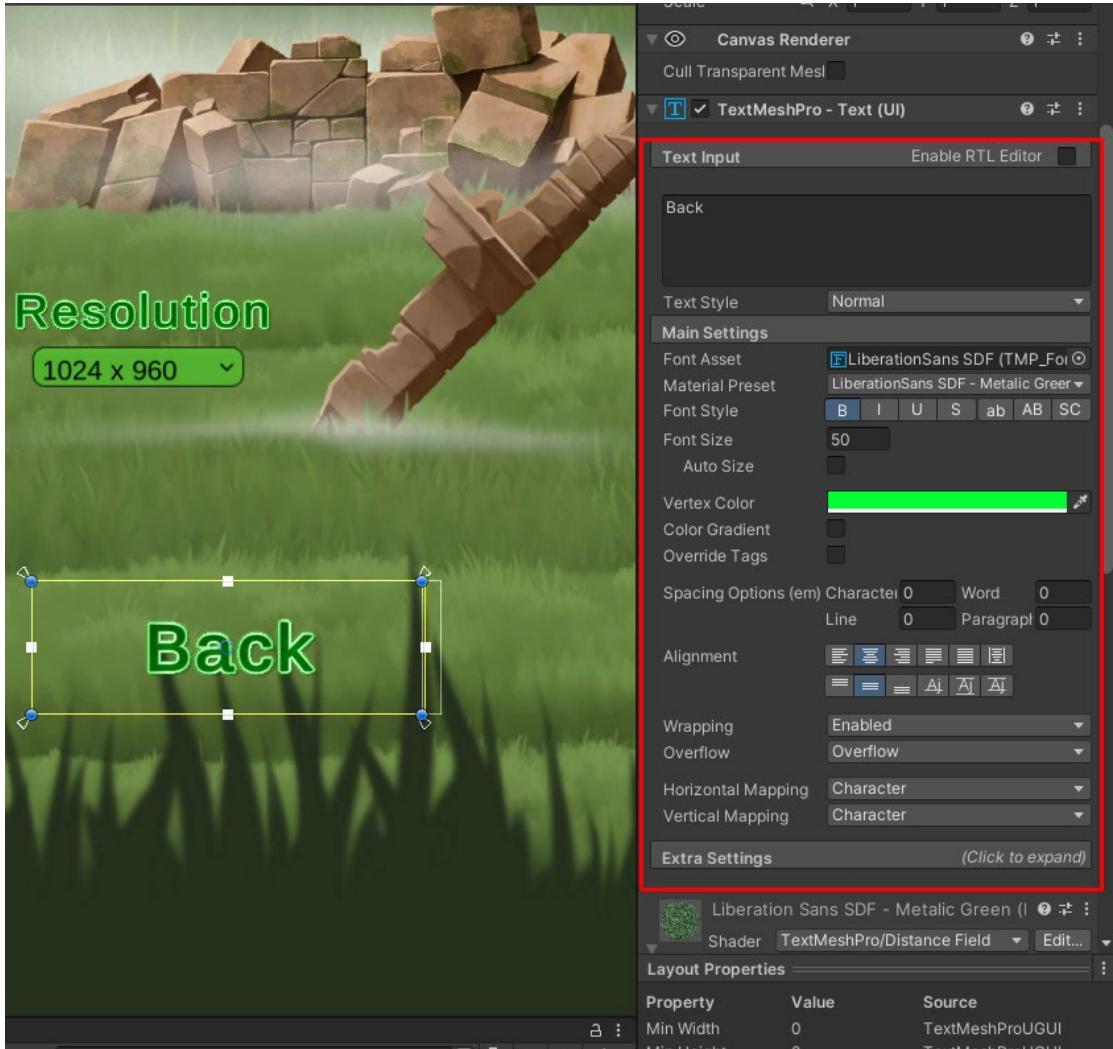
ΕΙΚΟΝΑ 3.5.1: Unity UI Tools



EIKONA 3.5.2: Unity UI Settings

Όλα τα μενού, οι επιλογές, ρυθμίσεις, βίντεο, εικόνες, κείμενα αλλά και τα credits του παρόντος παιχνιδιού έχουν αναπτυχθεί με την βοήθεια των προαναφερθέντων UI εργαλείων. Πίσω από τις επιλογές, τα κουμπιά και τα κείμενα που απαρτίζουν την παραπάνω φωτογραφία υπάρχει ένας πίνακας ή αλλιώς Panel και ένας καμβάς. Για να κατανοήσουμε την ύπαρξη τους θα χρειαστεί να αναφερθούμε στην κατασκευή ενός UI μενού όπως αυτό στην φωτογραφία. Το πρώτο πράγμα που θα χρειαστεί να εισαχθεί στο σκηνικό πριν ακόμα εισαχθούν τα κουμπιά και οι επιλογές είναι να εισαχθεί ένας καμβάς. Ο καμβάς είναι η βάση όλου του μενού που ο χρήστης θέλει να δημιουργήσει. Έχει παρόμοια λειτουργία με τον καμβά που χρησιμοποιούν οι ζωγράφοι ώστε να έχουν μια καθαρή επιφάνεια για να ζωγραφίσουν. Έτσι και στην Unity χρησιμοποιούμε έναν καμβά ώστε να προσθέσουμε όλα τα στοιχεία σε αυτόν. Στην συνέχεια είναι σημαντικό να εισαχθεί ένας πίνακας ή αλλιώς Panel που ορίζει τα όρια και το ακριβές σημείο που θα εισαχθούν τα κουμπιά και οι επιλογές. Μέσα στο Panel μπορεί να προστεθεί μια εικόνα ως φόντο, όπως ακριβώς φαίνεται στην εικόνα 3.5.2 που έχει για φόντο ένα λιβάδι. Τέλος μέσα στο Panel μπορεί να γίνει η εισαγωγή των στοιχείων όπως

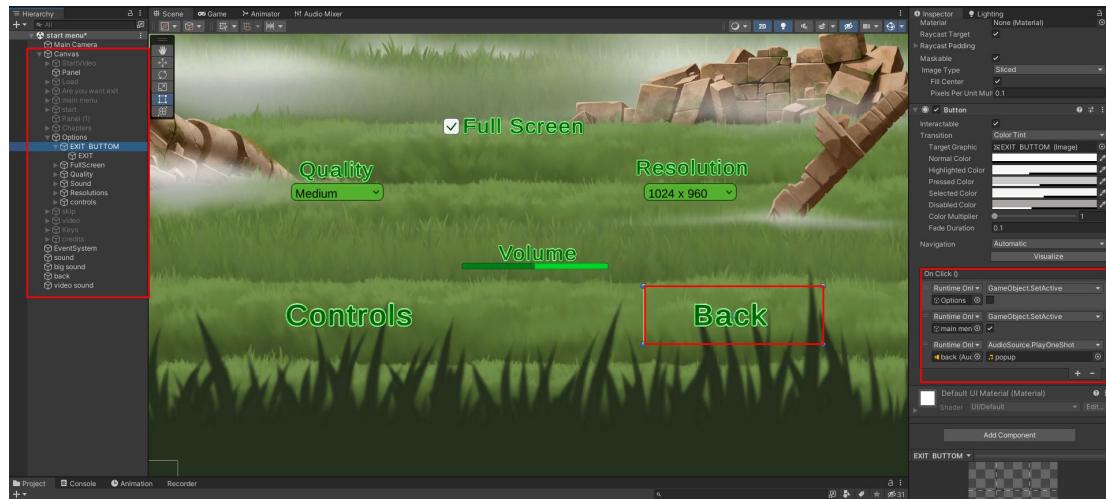
κουμπιά επιλογής, μπάρες αυξομείωσης κτλ.. Ο τρόπος που έχει γίνει η επεξεργασία και ο προγραμματισμός των ρυθμίσεων του παιχνιδιού της παραπάνω εικόνας γίνεται μέσω κώδικα αλλά και από τις ιδιότητες του εκάστοτε κουμπιού. Η ονομασία η μορφή και η σχεδίαση των κουμπιών γίνεται από τις ιδιότητες στα αριστερά και έχει παρόμοιο τρόπο με την διαχείριση κειμένου των Microsoft office.



ΕΙΚΟΝΑ 3.5.3: Unity UI Text

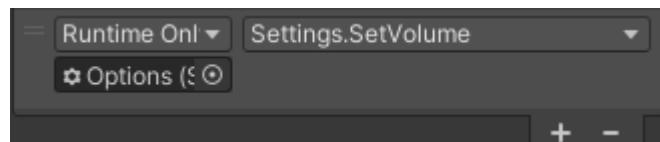
Αφού έγινε η σχεδίαση τους χρειάστηκε να δοθεί στα κουμπιά μια ιδιότητα. Όπως φαίνεται στην παρακάτω εικόνα είναι επιλεγμένο το κουμπί back που έχει προγραμματιστεί να λειτουργεί ως ένας τρόπος ώστε να μπορέσει ο χρήστης να πάει μια σελίδα πίσω. Στο δεξί μέρος υπάρχει η καρτέλα με όνομα On Click(). Σε αυτό το σημείο ορίζονται οι

ιδιότητες στο επιλεγμένο κουμπί. Συγκεκριμένα έχει οριστεί πως αν πατηθεί το κουμπί back τότε κλείσε τον πίνακα με όνομα Options, ενεργοποίησε το main menu και ενεργοποίησε μια φορά τον ήχο με όνομα popup.



EIKONA 3.5.4: Unity UI On Click

Με την ίδια λογική γίνεται η εισαγωγή ενός αρχείου κώδικα. Όπως φαίνεται παρακάτω λέμε το επιλεγμένο κουμπί να χρησιμοποιήσει ένα αρχείο κώδικα με όνομα Settings και συγκεκριμένα την συνάρτηση SetVolume που χρησιμοποιείται από το αντικείμενο με όνομα Options.



EIKONA 3.5.5: Unity UI On Click SetVolume

Το παρακάτω αρχείο κώδικα προγραμματίζει όλα τα κουμπιά στο μενού Options του παιχνιδιού και έχει εισαχθεί με τον ίδιο τρόπο που αναφέρθηκε παραπάνω.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Audio;
using UnityEngine.UI;
public class Settings : MonoBehaviour
{
    public AudioMixer mainMixer;
    public Slider sound;
    public Dropdown resolutionDropDown;
    public Dropdown quality;
    public Toggle fullScreen;
    Resolution[] resolutions;
    private float value;
    private bool result;
    public static int ResolutionDropIndex;
    private int currentResolutionIndex = 0;
    void Start ()
    {
        resolutions = Screen.resolutions;
        resolutionDropDown.ClearOptions();
        List<string> options = new List<string>();
        for (int i = 0; i < resolutions.Length; i++)
        {
            string option = resolutions[i].width + " x " + resolutions[i].height;
            options.Add(option);
        }
        resolutionDropDown.AddOptions(options);
        resolutionDropDown.value = currentResolutionIndex;
        SetResolution();
    }
    void SetResolution()
    {
        Resolution resolution = resolutions[resolutionDropDown.value];
        mainMixer.SetFloat("Sound", sound.value);
        if (fullScreen.checked)
        {
            Screen.SetResolution(resolution.width, resolution.height, true);
        }
        else
        {
            Screen.SetResolution(resolution.width, resolution.height, false);
        }
    }
}
```

```

        if (resolutions[i].width == Screen.currentResolution.width &&
resolutions[i].height == Screen.currentResolution.height)

    {
        currentResolutionIndex = i;

    }

}

resolutionDropDown.AddOptions(options);

ResolutionDropIndex = currentResolutionIndex;

resolutionDropDown.value = ResolutionDropIndex;

resolutionDropDown.RefreshShownValue();

}

public void Initialize()

{

result = mainMixer.GetFloat("volume", out value);

sound.value = value;

if (Screen.fullScreen == true)//screen.fullScreen == true

{

fullScreen.isOn = true;

}

else

{

fullScreen.isOn = false;

}

quality.value = QualitySettings.GetQualityLevel();

}

public void SetVolume (float volume)

```

```

{
    if (sound.value <= -40)
    {
        mainMixer.SetFloat("volume", -80);
    }
    else if (sound.value > -40)
    {
        mainMixer.SetFloat("volume", volume);
    }
}

public void SetFullScreen(bool isFullScreen)
{
    Screen.fullScreen = isFullScreen;
}

public void SetQuality(int qualityIndex)
{
    QualitySettings.SetQualityLevel(qualityIndex);
}

public void SetResolution(int resolutionIndex)
{
    Resolution resolution = resolutions[resolutionIndex];
    Screen.SetResolution(resolution.width, resolution.height,
    Screen.fullScreen);

    ResolutionDropIndex = resolutionDropDown.value;
}
}

```

3.6 C# SCRIPTS KAI MΗΧΑΝΙΣΜΟΙ

Σχεδόν κάθε αντικείμενο που έχει εισαχθεί στο σκηνικό του παιχνιδιού συνοδεύεται από ένα αρχείο κώδικα σε γλώσσα C# ή αλλιώς Script ώστε να δημιουργηθούν συνθήκες και να έχουν κάποιον ρόλο στο σύνολο του παιχνιδιού. Το παρακάτω αρχείο κώδικα χρησιμοποιείται από την μέδουσα του Chapter 3 και της επιτρέπει να κινείται χωρίς την παρέμβαση του παίκτη αλλά ταυτόχρονα τα αλλάζει κατεύθυνση και πορεία όταν πλησιάσει σε κάποιο εμπόδιο ή όταν υπάρχει κενό στο έδαφος.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class walk : MonoBehaviour
{
    public float speed;
    public bool mouvingRight = true;
    public Transform grounddetection;
    public float distance;
    private Animator anim;
    private bool isonbox = false;
    void Start()
    {
        anim = GetComponent<Animator>();
        anim.SetBool("walk", true);
    }
    void OnTriggerEnter2D(Collider2D other)
```

```

{
    if (other.gameObject.tag == "block")
    {
        isonbox = true;
    }
}

void OnTriggerExit2D(Collider2D other)
{
    if (other.gameObject.tag == "block")
    {
        isonbox = false;
    }
}

void Update()
{

    if (attack.iswalk)
    {
        transform.Translate(Vector2.right * speed * Time.deltaTime);
        RaycastHit2D groundInfo = Physics2D.Raycast(grounddetection.position, Vector2.down, distance);
        if (!groundInfo.collider || isonbox)
        {
            if (mouvingRight)
            {
                transform.eulerAngles = new Vector3(0, -180, 0);
            }
        }
    }
}

```

```

        mouvingRight = false;

    }

    else

    {

        transform.eulerAngles = new Vector3(0, -0, 0);

        mouvingRight = true;

    }

}

}

```

Το επόμενο αρχείο με κώδικα ανήκει στο Chapter 3 και πυροδοτεί την επίθεση των εχθρών. Όταν ο παίκτης πλησιάσει μέσω του πρωταγωνιστή στους εχθρούς αυτοί του επιτίθενται με την βοήθεια μιας κίνησης με το δόρυ τους και της αναπαραγωγής του αντίστοιχου ήχου. Αφού ο πρωταγωνιστής χτυπηθεί και λόγο της φόρας που έχει το δόρυ, μετατοπίζεται μερικά βήματα προς τα πίσω και εκτελείται το animation του θανάτου του χαρακτήρα.

```

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class attack : MonoBehaviour

{

    public Animator ani;

```

```
private float entpos;  
private float pos;  
public Animator playeranim;  
public static int damage = 0;  
public GameObject Demon;  
public GameObject player;  
public static bool iswalk = true;  
AudioSource audioSource;  
public float volume;  
public AudioClip attacksound;  
public static bool oneTimeAttack = true;  
void Start()  
{  
    ani = Demon.GetComponent<Animator>();  
    playeranim = player.GetComponent<Animator>();  
}  
void OnTriggerEnter2D(Collider2D Player)  
{  
    pos = player.transform.position.x;  
    if (Player.gameObject.tag == "Player")  
    {  
        if (oneTimeAttack)  
        {  
            ani.SetBool("attack", true);  
            playeranim.SetBool("hurt", true);  
            StartCoroutine(attacksd());  
        }  
    }  
}
```

```

        damage++;

        entpos = player.transform.position.x;
        StartCoroutine(ExampleCoroutine());
        iswalk = false;
        oneTimeAttack = false;
    }

}

void OnTriggerExit2D(Collider2D Player)
{
    if (Player.gameObject.tag == "Player")
    {
        ani.SetBool("attack", false);
        playeranim.SetBool("hurt", false);
        iswalk = true;
    }
}

public IEnumerator attacksd()
{
    yield return new WaitForSeconds(0.3f);
    audioSource = GetComponent< AudioSource >();
    audioSource.PlayOneShot(attacksound, volume);
}

public IEnumerator ExampleCoroutine()
{
    yield return new WaitForSeconds(0.4f);
}

```

```

if (Demon.transform.position.x > player.transform.position.x)

{
    player.transform.position = new
Vector3(player.transform.position.x + -3.5f, player.transform.position.y,
player.transform.position.z);

playeranim= GetComponent<Animator>();

}

else

{

    player.transform.position = new
Vector3(player.transform.position.x + 3.5f, player.transform.position.y,
player.transform.position.z);

playeranim= GetComponent<Animator>();

}

}

```

Το επόμενο αρχείο κώδικα έχει δημιουργηθεί ώστε να καλύψει μια ανάγκη που προέκυψε κατά την διάρκεια της ανάπτυξης του παιδιού. Όσο ο χαρακτήρας περιπλανιόταν στο περιβάλλον του παιχνιδιού φαινόταν πως όλα τα αντικείμενα που αποτελούσαν το φόντο πίσω του ήταν μία φωτογραφία. Έδινε την εντύπωση πως ο χαρακτήρας κουνιόταν μπροστά από μία ζωγραφιά και έλειπε η αίσθηση του βάθους και της απόστασης των αντικειμένων. Προκειμένου να λυθεί αυτό το πρόβλημα και να δοθεί στον παίκτη η ψευδαίσθηση του βάθους και της απόστασης αναπτύχθηκε ο παρακάτω κώδικας. Η λογική είναι απλή, αφού δεν μπορούμε να έχουμε βάθος λόγω της δισδιάστατης φύσης του παιχνιδιού μπορούμε να προσποιηθούμε πως υπάρχει. Αυτό που κάνει ο κώδικας είναι πως κάθε φορά που κουνιέται ο χαρακτήρας κινούνται μαζί του όλα τα στρώματα που υπάρχουν στο φόντο αλλά με διαφορετικές ταχύτητες. Μαζί με την μετακίνηση του παίκτη

μετακινείται ο ουρανός με πολύ μικρή ταχύτητα. Ταυτόχρονα μετακινούνται τα βουνά με ελάχιστα μεγαλύτερη ταχύτητα από αυτήν του ουρανού. Με λίγο μεγαλύτερη ταχύτητα μετακινούνται τα δέντρα που βρίσκονται λίγο πιο κοντά στον παίκτη. Αυτό το μοτίβο συνεχίζεται με την διαφορά πως τα αντικείμενα που βρίσκονται δίπλα στον χαρακτήρα όπως το έδαφος που πατάει μένουν τελείως στάσιμα.

```
using UnityEngine;

public class ParallaxLayer : MonoBehaviour
{
    [SerializeField] float multiplier = 0.0f;
    [SerializeField] bool horizontalOnly = true;

    private Transform cameraTransform;

    private Vector3 startCameraPos;
    private Vector3 startPos;

    void Start()
    {
        cameraTransform = Camera.main.transform;
        startCameraPos = cameraTransform.position;
        startPos = transform.position;
    }
}
```

```
private void LateUpdate()
{
    var position = startPos;
    if (horizontalOnly)
        position.x += multiplier * (cameraTransform.position.x - startCameraPos.x);
    else
        position += multiplier * (cameraTransform.position - startCameraPos);

    transform.position = position;
}

}
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

ΠΗΓΕΣ

Για την ανάπτυξη της παρούσας πτυχιακής εργασίας χρησιμοποιήθηκαν οι παρακάτω πηγές.

Κεφάλαιο 1

http://qbrainblog.blogspot.com/2013/12/blog-post_4.html

https://en.wikipedia.org/wiki/Cathode-ray_tube_amusement_device

https://en.wikipedia.org/wiki/Tennis_for_Two

<https://el.wikipedia.org/wiki/Pac-Man>

<https://el.wikipedia.org/wiki/Tetris>

https://en.wikipedia.org/wiki/The_Game_Awards_2019

<https://health4u.gr/psykiki-ygeia/to-proto-video-game-me-iatriki-syntagi/>

<https://www.akiliinteractive.com/news-collection/akili-announces-endeavortm-attention-treatment-is-now-available-for-children-with-attention-deficit-hyperactivity-disorder-adhd-al3pw>

<https://www.ertnews.gr/eidiseis/epistimi/ena-iatriko-vinteopechnidi-voitha-tous-giatrous-sti-diagnosi-travmatos/>

<https://futurism.com/neoscope/medical-video-game-improves-doctors-ability-recognize-trauma>

[https://www.virtuallythereretaining.com/training-systems/#iLightbox\[16867a4d9d945b150e3\]/0](https://www.virtuallythereretaining.com/training-systems/#iLightbox[16867a4d9d945b150e3]/0)

Κεφάλαιο 2

https://en.wikipedia.org/wiki/Adobe_Photoshop

https://en.wikipedia.org/wiki/Adobe_Premiere_Pro

<https://el.wikipedia.org/wiki/Audacity>

https://en.wikipedia.org/wiki/Microsoft_Visual_Studio

<https://en.wikipedia.org/wiki/Notepad%2B%2B>

<https://en.wikipedia.org/wiki/Trello>

https://el.wikipedia.org/wiki/Google_Drive

[https://en.wikipedia.org/wiki/Unity_\(game_engine\)](https://en.wikipedia.org/wiki/Unity_(game_engine))

ASSETS ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ

Για την ολοκλήρωση του παιχνιδιού κάποια γραφικά κατασκευάστηκαν από την αρχή ενώ κάποια άλλα πάρθηκαν από τις παρακάτω ιστοσελίδες και αργότερα επεξεργάστηκαν.

<https://assetstore.unity.com/2d>

<https://www.freepik.com/free-photos-vectors/png>

<https://pngimg.com/>

<https://www.pngwing.com>

<https://pngtree.com>

Οι ήχοι που ακούγονται κατά την διάρκεια του παιχνιδιού προέρχονται από τις παρακάτω σελίδες. Οι περισσότεροι ήχοι έχουν υποστεί επεξεργασία ενώ κάποιοι άλλοι έχουν προκύψει από την μίξη πολλών επεξεργασμένων ήχων.

<https://www.zapsplat.com/>

<https://freesound.org/>

<https://www.youtube.com/>