2020

Αντώνης Καδόγλου

[ΑΝΑΠΑΡΑΣΤΑΣΗ ΤΗΣ ΠΑΝΕΠΙΣΤΗΜΙΟΥΠΟΛΗΣ ΚΟΖΑΝΗΣ ΜΕΣΩ UNITY 3D – ΑΝΤΩΝΗΣ ΚΑΔΟΓΛΟΥ]

Η διπλωματική εργασία είναι μια τρισδιάστατη αναπαράσταση της καινούργιας πανεπιστημιούπολης Κοζάνης, με σκοπό ο φοιτητής να έρθει σε μία πρώτη επαφή με το περιβάλλον που πρόκειται να συναντήσει.

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΤΙΤΛΟΣ:

Αναπαράσταση της πανεπιστημιούπολης Κοζάνης μέσω του προγράμματος Unity 3D

Αντώνης Καδόγλου (ΑΕΜ : 687)

Επιβλέπων καθηγητής : Δρ. Αγγελίδης Παναγιώτης

ΚΟΖΑΝΗ

29/6/2020

Σύνοψη

Η διπλωματική εργασία είναι μία τρισδιάστατη αναπαράσταση της καινούργιας πανεπιστημιούπολης Κοζάνης , με σκοπό ο υποψήφιος φοιτητής να έρθει σε μία πρώτη επαφή με το περιβάλλον που πρόκειται να συναντήσει αν επιλέξει τελικά να φοιτήσει στο Πανεπιστήμιο Δυτικής Μακεδονίας.

Επέλεξα να χρησιμοποιήσω την game engine Unity 3D για την υλοποίηση της διπλωματικής , διότι είναι εύκολη στην εκμάθηση , έχει κατανοητό σύστημα εισαγωγής asset και packages, όμορφα γραφικά, φωτισμό και particle system αλλά και άλλες δυνατότητες που μπορεί ο χρήστης να αξιοποιήσει με λίγη εκμάθηση.

Abstract

The dissertation is a three-dimensional representation of the new university in Kozani, in order for the prospective student to come into first contact with the environment he is about to encounter if he finally chooses to study at the University of Western Macedonia.

I chose to use the game engine Unity 3D for the implementation of diplomacy, because it is easy to learn, has an understandable system of introduction of asset and packages, beautiful graphics, lighting and particle system and other features that the user can use with a little learning.

Περιεχόμενα

Κατάλογος Εικόνων

Πνευματικά δικαιώματα

1.Εισαγωγή

Το video game development είναι η διαδικασία δημιουργίας ενός ηλεκτρονικού παιχνιδιού είτε για ηλεκτρονικό υπολογιστή, κονσόλα ή κινητή συσκευή. Η ανάπτυξη του παιχνιδιού γίνεται από έναν game developer, από μια μικρή ομάδα ή από μια μεγάλη επιχείρηση. Η συνήθης πρακτική είναι τα παιχνίδια αυτά να χρηματοδοτούνται από έναν publisher και συνήθως θέλουν κάποια χρόνια για να ολοκληρωθούν. Υπάρχει βέβαια και η κατηγορία των Indie Games(ομάδες πολλών ή ενός ατόμου που δημιουργούν video games χωρίς κάποιον publisher), όπου τα παιχνίδια μπορεί να χρειάζονται λιγότερο καιρό για να δημιουργηθούν αλλά και μικρότερο budget.

Οι μηχανές παιχνιδιών κάνουν την εμφάνισή τους στις αρχές της δεκαετίας του '90 με το ξεκίνημα των 3d γραφικών, όπου απογείωσε την βιομηχανία των ηλεκτρονικών παιχνιδιών. Στα μέσα της σημαντικής αυτής δεκαετίας καθιερώθηκε ο όρος game engine ,με τα παιχνίδια "Doom" και "Quake" να κάνουν την επανάσταση στο game developing. Μετά την επιτυχία των δύο αυτών παιχνιδιών διάφοροι developers άρχισαν και αγόραζαν βασικά κομμάτια και άρχισαν να προσθέτουν δικά τους αντικείμενα στο παιχνίδι όπως όπλα και διάφορα γραφικά στοιχεία. Έτσι ο καθένας άρχιζε να αναπτύσσει το βασικό πρόγραμμα για όποια πλατφόρμα ήθελε και μπορούσε ακόμα να τα πουλήσει σαν ξεχωριστά video games.

Ο σχεδιασμός του παιχνιδιού είναι η τέχνη της εφαρμογής του σχεδιασμού και της αισθητικής για τη δημιουργία ενός παιχνιδιού για ψυχαγωγία ή για εκπαιδευτικούς, γυμναστικούς ή πειραματικούς σκοπούς. Όλο και περισσότερο, στοιχεία και αρχές σχεδιασμού παιχνιδιών εφαρμόζονται και σε άλλες αλληλεπιδράσεις, με τη μορφή gamification.

Παρότι η Unity 3D είναι μηχανή δημιουργίας παιχνιδιών, η διπλωματική εργασία επικεντρώνεται στις σχεδιαστικές δυνατότητες της μηχανής, δηλαδή στο level design. Η προγραμματιστική γλώσσα που χρησιμοποιείτε είναι η C#.

Με την ολοένα αυξανόμενη επεξεργαστική και γραφική δυνατότητα των κονσολών και προϊόντων πληροφορικής, σε συνδυασμό με την αύξηση των προσδοκιών των χρηστών, ο σχεδιασμός των επιπέδων των παιχνιδιών κινήθηκε πέρα από την ιδέα ενός μόνο προγραμματιστή να πρέπει να παράγει ένα εμπορεύσιμο προϊόν σε συγκεκριμένο χρονικό διάστημα. Αυτό πυροδότησε την έναρξη της ανάπτυξης με ομάδες.

1.2 Μηχανές Ανάπτυξης Παιχνιδιών

Μια μηχανή παιχνιδιών, επίσης γνωστή ως αρχιτεκτονική παιχνιδιού, πλαίσιο παιχνιδιού ή πλαίσιο παιχνιδιού, είναι ένα περιβάλλον ανάπτυξης λογισμικού σχεδιασμένο για τους ανθρώπους να κατασκευάζουν βιντεοπαιχνίδια. Οι προγραμματιστές χρησιμοποιούν μηχανές παιχνιδιών για την κατασκευή παιχνιδιών για κονσόλες, κινητές συσκευές και προσωπικούς υπολογιστές. Η βασική λειτουργικότητα που παρέχεται συνήθως από μια μηχανή παιχνιδιών περιλαμβάνει μια μηχανή απόδοσης ("renderer") για 2D ή 3D γραφικά, μια μηχανή φυσικής ή ανίχνευση σύγκρουσης (και απόκριση σύγκρουσης), ήχο, δέσμες ενεργειών, κινούμενα σχέδια, τεχνητή νοημοσύνη, δικτύωση, ροή, μνήμη διαχείριση, συνομιλία, υποστήριξη εντοπισμού, γράφημα σκηνής και μπορεί να περιλαμβάνει υποστήριξη βίντεο για κινηματογραφικά. Οι εκτελεστές συχνά εξοικονομούν τη διαδικασία ανάπτυξης παιχνιδιών επαναχρησιμοποιώντας / προσαρμόζοντας, σε μεγάλο βαθμό, την ίδια μηχανή παιχνιδιών για την παραγωγή διαφορετικών παιχνιδιών ή για να βοηθήσουν στη μεταφορά παιχνιδιών σε πολλές πλατφόρμες.

Η τεχνολογία των βιντεοπαιχνιδιών έχει αναπτυχθεί πάρα πολύ με αποτέλεσμα οι μηχανές ανάπτυξης παιχνιδιών σήμερα αριθμούνται πάνω από 400, αυτό δεν σημαίνει όμως ότι με όλες τις μηχανές σου δίνουν τα ίδια δικαιώματα και ότι έχεις τις ίδιες δυνατότητες. Κάθε μηχανή έχει τα πλεονεκτήματα και τα μειονεκτήματα της. Οι βασικές μηχανές σήμερα που χρησιμοποιούν οι περισσότεροι είναι :

- Unity3D
- UDK - Unreal Engine
- Godot
- Gamemaker

Εικόνα 1 Λογότυπο Unity

1.3 Γιατί Unity ;

Κάθε μηχανή ανάπτυξης έχει τις ευκολίες της και τις δυσκολίες της, έχει το κοινό της και την υποστήριξη ως προς τον Developer. Εγώ κατέληξα στη Unity 3D επειδή έχει έκδοση Free, έχει ένα ForumCommunity υποστήριξης forum.unity3d.com όπου μπορείς να βρεις λύση σε ότι δυσκολία συναντήσεις αλλά και πολλά guides στο Youtube και τέλος έχει ένα asset store όπου σου παρέχει μερικά γραφικά μοντέλα - scripting code δωρεάν.

1.4 Κίνητρο για την Διεξαγωγή της Εργασίας

Το βασικό κίνητρο για την δημιουργία της πτυχιακής, ήταν η αγάπη μου για τα Video games για τα οποία έχω αφιερώσει πολύ χρόνο από παιδί μέχρι σήμερα και ειδικότερα τα ωραία τοπία που πάντα με συνέπαιρναν. Έτσι ήταν απόλυτα λογικό κάποια στιγμή να θελήσω να δημιουργήσω ένα δικό μου παιχνίδι και να ασχοληθώ κυρίως με το γραφικό κομμάτι. Το να προτείνω και να διαλέξω μια πτυχιακή η οποία σχετίζεται και βρίσκεται μέσα στον κλάδο του level design ήταν φυσικό επόμενο. Ήταν μια πρώτης τάξεως ευκαιρία να δω από πρώτο χέρι ποιες είναι οι διαδικασίες ανάπτυξης ενός σχεδιασμού επιπέδου video game και πως θα μπορούσα να ανταπεξέλθω σε αυτές.

1.5 Σκοπός και Στόχοι Εργασίας

Σκοπός της διπλωματικής , πέρα από το τελικό αποτέλεσμα που ήθελα να καταφέρω να φέρω εις πέρας, ήταν να αναπτύξω τις ικανότητες μου σε όλα τα στάδια δημιουργίας ενός level design. Τα στάδια αυτά και οι διαδικασίες είναι πολλά και περίπλοκα. Έπρεπε να αναπτύξω τις ικανότητες μας στο 3d modeling και όλα τα παράγωγα αυτού, το animation και το scripting. Οι στόχοι αυτοί που έθεσα ήταν βασικοί και έπρεπε, μέχρι ένα σημείο, να καταφέρω να ανταπεξέλθω στις απαιτήσεις που θα παρουσιαζόντουσαν. Εξάλλου όλοι αυτοί οι τομείς που ανέφερα, σε μια μεγάλη παραγωγή είναι ξεκάθαρα διακριτοί και ξεχωριστοί και τους αναλαμβάνουν συγκεκριμένες ομάδες με πείρα και πολύ εμπειρία.

2. Unity

Η Unity είναι μια από τις πιο δημοφιλείς μηχανές παιχνιδιών. Στα μέσα του 2013 χρησιμοποιήθηκε από περισσότερους από δύο εκατομμύρια προγραμματιστές παγκοσμίως. Η τελευταία έκδοση της μηχανής είναι η Unity 2019.3 . Κυκλοφόρησε τον Ιανουάριο του 2020. Η Unity έχει τη μεγαλύτερη λίστα υποστηριζόμενων πλατφορμών σε σύγκριση με άλλες μηχανές παιχνιδιών. Διαθέτει επίσης plugin webplayer που επιτρέπει την εκτέλεση του παιχνιδιού στο πρόγραμμα περιήγησης. Η Unity έχει μία από τις χαμηλότερες απαιτήσεις συστήματος για προγραμματιστές σε σύγκριση με τους ανταγωνιστές της.

Σε σύγκριση με άλλες μηχανές, η Unity επιτρέπει τη χρήση πολλών γλωσσών για τον προγραμματισμό παιχνιδιών. Για την ανάπτυξη η Unity υποστηρίζει:

- C#
- JavaScript
- Boo

Όλες οι αναφερόμενες γλώσσες μπορούν να συνδυαστούν σε μια σκηνή και ακόμη και τα μισά script ενός αντικειμένου μπορούν να γραφτούν σε μία γλώσσα ενώ σε άλλο μέρος μπορεί να χρησιμοποιήσει διαφορετική.

2.1 Ιστορική αναδρομή

Η ιδέα της Unity ξεκίνησε το 2002 από ένα Post του Δανού Nicholas Francis όπου ρωτούσε για το ποιός ήθελε να φτιάξει μαζί του ένα Game Engine, λίγες ώρες αργότερα ο Joachim Ante απάντησε στο ερώτημα και έτσι έγινε η αρχή, τελικά οι προγραμματιστές έγιναν τρεις αφού μπήκε στο γκρούπ και ο Devid Helgason. Η εταιρία τελικά ιδρύθηκε το 2004 στη Δανία από τους τρεις αυτούς προγραμματιστές. Η βασική επιτυχία της μηχανής αυτή στηρίζεται στο γεγονός ότι βοηθάει τους ανεξάρτητους Game Developers οι οποίοι δεν είναι σε θέση να δημιουργήσουν την δικιά τους μηχανή για να φτιάξουν το δικό τους παιχνίδι. Το μεγάλο "Μπαμ" έγινε όταν ήρθε στην επιφάνεια το Iphone και το appstore , καθώς η μηχανή ήταν ήδη έτοιμη για την συγκεκριμένη πλατφόρμα. Σύμφωνα με μια έρευνα το Unity χρησιμοποιείται πάνω από το 50% των προγραμματιστών που ασχολούνται με την δημιουργία παιχνιδιών σε iOS και Android.

2.2 Άδεια

Η Unity χρησιμοποιεί δύο βασικούς τύπους αδειών: δωρεάν και Pro. Η δωρεάν έκδοση περιορίζεται από υποστηριζόμενα χαρακτηριστικά. Επίσης, σε περίπτωση εισοδήματος άνω των 100.000 $, η έκδοση Pro πρέπει να αγοραστεί.

2.3 Χαρακτηριστικά

Αν και η Unity έχει πληθώρα χαρακτηριστικών και δυνατοτήτων, τα κυριότερα που εν τέλει με βοήθησαν στην ολοκλήρωση της διπλωματικής ήταν:

- Η Unity διαθέτει πολλές ενσωματωμένες ρυθμίσεις για φωτισμό, ψήσιμο, post processing και shaders

- Μπορούμε να κάνουμε αρκετά όμορφο έδαφος χρησιμοποιώντας το ενσωματωμένο εργαλείο εδάφους, αρκεί να κατεβάσουμε μερικές ωραίες υφές και δέντρα ή γρασίδι.

- Η κλίμακα και η ευελιξία του Asset store, δεν περιορίζεται μόνο σε μοντέλα και περιεχόμενο, αλλά εργαλεία που μπορούμε να χρησιμοποιήσουμε για την διευκόλυνσή μας.

2.3.1 HDRP

Την μεγάλη διαφορά στην τελική εικόνα της αναπαράστασης έκανε η χρήση του HDRP(High Definition Render Pipeline). Το HDRP είναι ένας νέος τρόπος rendering που μας δίνει πόρους για να επιτύχουμε ρεαλιστικά αποτελέσματα στην Unity. Φέρνει ένα νέο σύστημα Lightning που ονομάζεται Physical Light Units (PLU), νέες ρυθμίσεις έντασης που περιέχουν πληροφορίες σχετικά με την ομίχλη, το οπτικό περιβάλλον, τις σκιές, τις αντανακλάσεις κ.λ.π.

Εικόνα 2 Παράδειγμα αποτελέσματος HDRP και HDRI

2.4 Unity Hub

Το Unity Hub είναι μια αυτόνομη εφαρμογή που βελτιστοποιεί τον τρόπο που βρίσκετε, κατεβάζετε και διαχειρίζεστε τα Unity Projects και τις εγκαταστάσεις σας. Επιπλέον, μπορείτε να προσθέσετε μη αυτόματες εκδόσεις του Editor που έχετε ήδη εγκαταστήσει στον υπολογιστή σας στο Hub σας.

Μπορείτε να χρησιμοποιήσετε το Hub για να :

- Διαχειριστείτε τις άδειες λογαριασμού Unity και Editor.
- Δημιουργήστε το έργο σας, συσχετίσετε μια προεπιλεγμένη έκδοση του προγράμματος επεξεργασίας Unity με το έργο και να διαχειριστείτε την εγκατάσταση πολλαπλών εκδόσεων του προγράμματος επεξεργασίας.

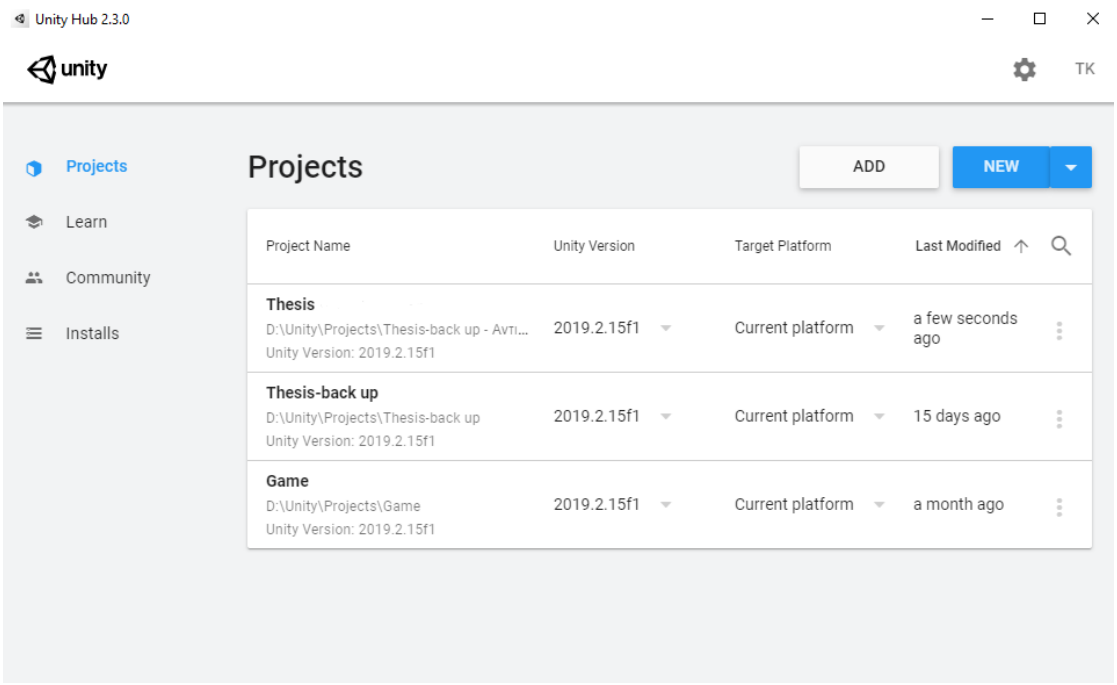
- Εκκινήσετε διαφορετικές εκδόσεις του Unity από το Project View

- Διαχειριστείτε και επιλέξτε Στόχους κατασκευής έργου χωρίς να ξεκινήσετε τον Επεξεργαστή.

- Εκτελέστε ταυτόχρονα δύο εκδόσεις του Unity. Σημειώστε ότι για να αποφύγετε τοπικές διενέξεις και άλλα περίεργα σενάρια, θα πρέπει να ανοίγετε ένα Έργο μόνο σε μία παρουσία του προγράμματος επεξεργασίας κάθε φορά.

- Προσθέστε στοιχεία σε υπάρχουσες εγκαταστάσεις του Editor. Όταν κάνετε λήψη μιας έκδοσης του Editor μέσω του Unity Hub, μπορείτε να βρείτε και να προσθέσετε επιπλέον στοιχεία (όπως συγκεκριμένη υποστήριξη πλατφόρμας, Visual Studio, έγγραφα εκτός σύνδεσης και τυπικά στοιχεία) είτε κατά την αρχική εγκατάσταση είτε σε μεταγενέστερη ημερομηνία.

- Χρησιμοποιήσετε πρότυπα έργου για να ξεκινήσετε τη διαδικασία δημιουργίας κοινών τύπων έργου. Τα Πρότυπα έργου της Unity παρέχουν προεπιλεγμένες τιμές για κοινές ρυθμίσεις όταν δημιουργείτε ένα νέο Έργο. Project Templates προκαθορισμένες ομάδες ρυθμίσεων για έναν τύπο παιχνιδιού στόχου ή επίπεδο οπτικής πιστότητας

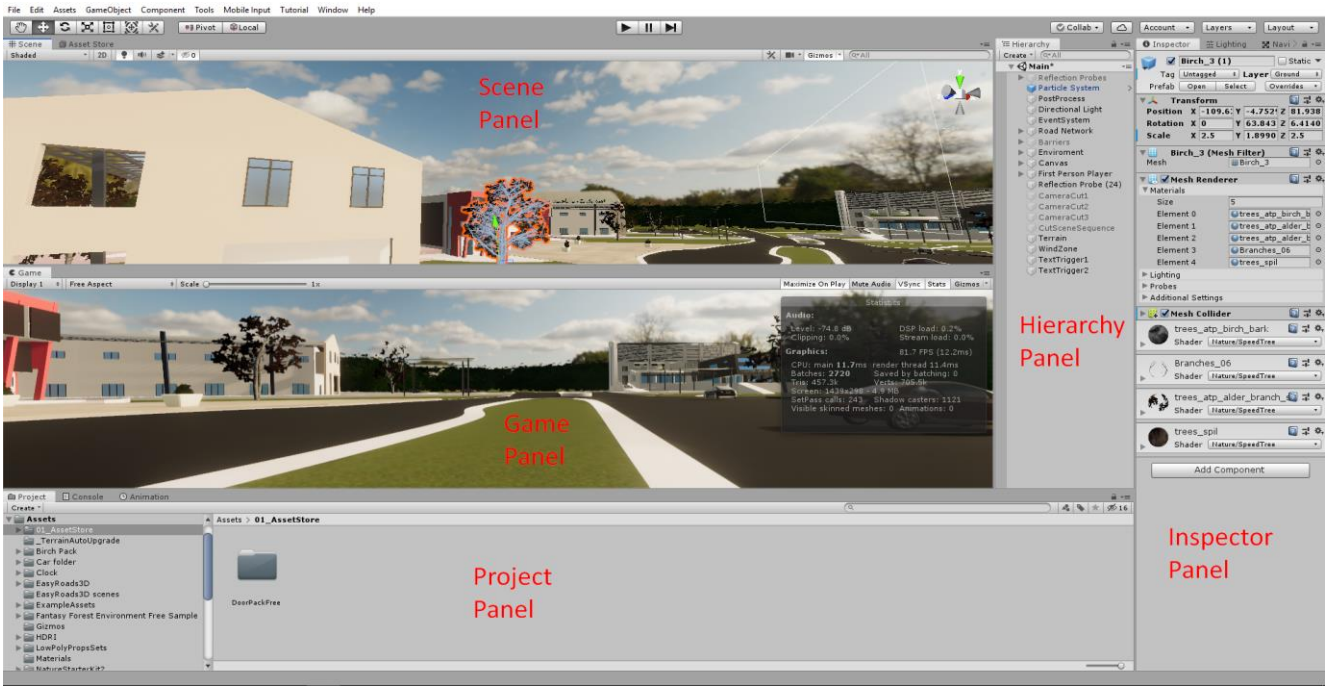


Εικόνα 3 Περιβάλλον Unity Hub

2.5 Unity Editor

Ο Editor του Unity αποτελείτε από διάφορα Panel τα οποία κάνουν το Project πιο εύκολα στη διαχείριση του. Ο χρήστης έχει την δυνατότητα να δημιουργήσει το δικό του Interface καθώς έχει την δυνατότητα να μεταφέρει και να επιλέξει να φαίνεται ότι αυτός επιθυμεί. Τα σημαντικότερα από αυτά και που θα αναλύσουμε στη συνέχεια είναι :

- Scene View
- Game View
- Hierarchy
- Project
- Inspector



Εικόνα 4 Περιβάλλον Unity Editor

2.5.1 Scene Panel

Η “σκηνή” είναι ο χώρος κατασκευής του παιχνιδιού όπου ο χρήστης μπορεί να πλοηγηθεί και να επεξεργαστεί τον χώρο αυτό μέσο των πλήκτρων Q, W, E, R. Αυτό το παράθυρο είναι το πιο σημαντικό μέρος της Unity , επειδή είναι το παράθυρο σχεδίασης.

Πλήκτρο Q(Navigation Tool): 

Μέσω του πλήκτρου αυτού που αντιπροσωπεύει το χεράκι στην οθόνη μας πλοηγούμαστε στον χώρο. Συγκεκριμένα, κρατώντας πατημένο το αριστερό κλικ του ποντικιού μετακινούμε την κάμερα αριστερά, δεξιά, πάνω και κάτω. Πιέζοντας ταυτόχρονα και το πλήκτρο alt μπορούμε να κάνουμε περιστροφή γύρω από το αντικείμενο που έχουμε εστιάσει. Πιέζοντας αυτή την φορά το δεξί κλικ του ποντικιού περιστρέφουμε την κάμερα προς την επιλεγμένη διεύθυνση που του δίνουμε(αριστερά, δεξιά, πάνω, κάτω)ενώ πατώντας ταυτόχρονα και το alt κάνουμε ζουμ.

Πλήκτρο W(Translate Tool): 

Χρησιμοποιείται όταν έχουμε εστιάσει σε ένα αντικείμενο που έχουμε τοποθετήσει και μας επιτρέπει να το μετακινήσουμε προς την διεύθυνση που δείχνουν τα βελάκια πάνω στο αντικείμενο σύμφωνα με τους άξονες x, y, z.

Πλήκτρο E(Rotation Tool): 

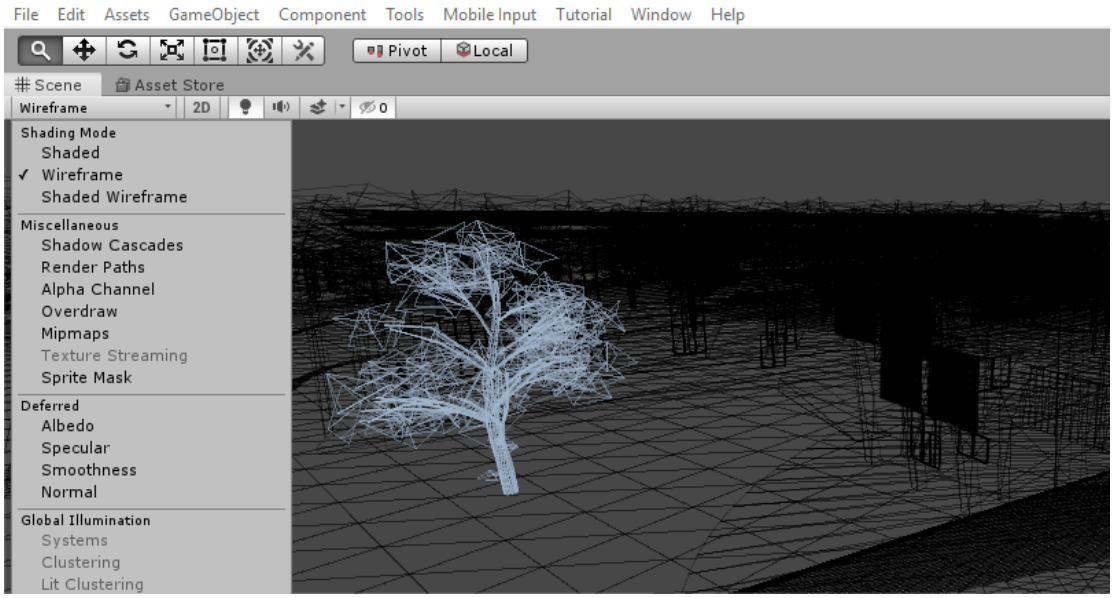
Μας επιτρέπει να περιστρέψουμε ένα αντικείμενο επιλέγοντας κάθε φορά έναν άξονα περιστροφής.

Πλήκτρο R(Scale Tool): 

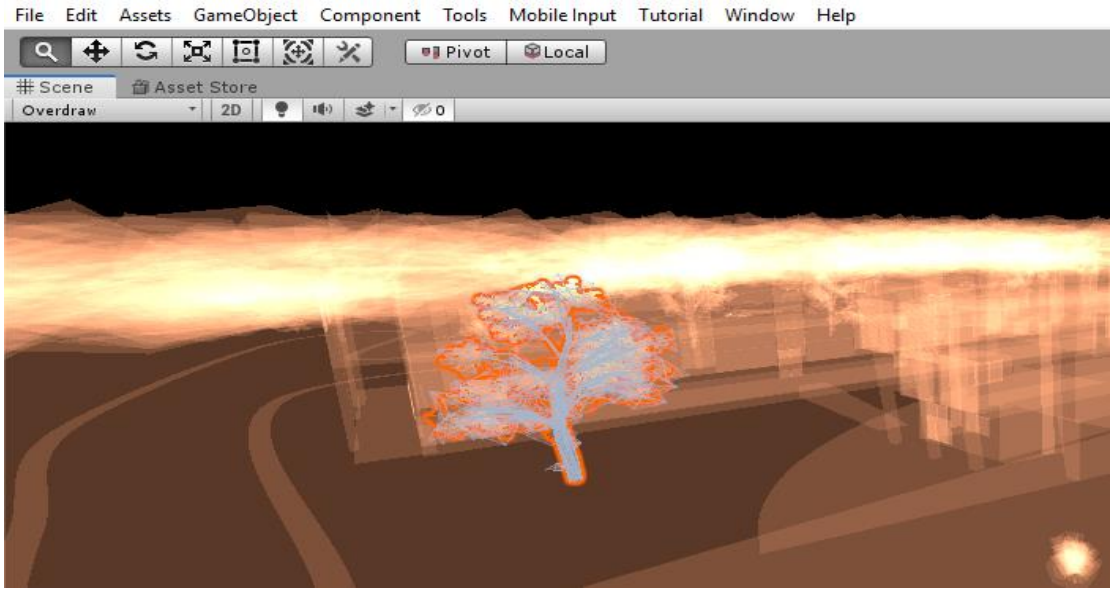
Με το πλήκτρο R μπορούμε να ρυθμίσουμε τις διαστάσεις για το αντικείμενό μας σε κάθε άξονα ξεχωριστά αλλά και να μεγεθύνουμε ή σμικρύνουμε αναλογικά το αντικείμενο πατώντας στο κέντρο του.

Rendering Options:

Ακριβώς κάτω από το παράθυρο Scene βλέπουμε την μέθοδο προβολής(Textured) των αντικειμένων όπου πρόκειται για ένα drop down menu που μας έχει επιπλέον επιλογές(Wireframe, Shaded Wireframe, Render Paths, Overdraw κτλ) και επιλέγουμε την επιθυμητή λειτουργία.



Εικόνα 5 Wireframe Render



Εικόνα 6 Overdraw Render

2.5.2 Game Panel

Το Game Panel αναπαριστά το παιχνίδι στην τελική μορφή του, αυτό γίνεται με την χρήση του κουμπιού "Play" που βρίσκεται στην γραμμή εργαλείων μαζί με τα κουμπιά "Pause" και "Next Frame" του Unity. Συγκεκριμένα :

Play : 

Με το κουμπί αυτό μπορείς να δοκιμάσεις-τρέξεις το παιχνίδι και να πάρεις μια ιδέα για την τελική μορφή του.

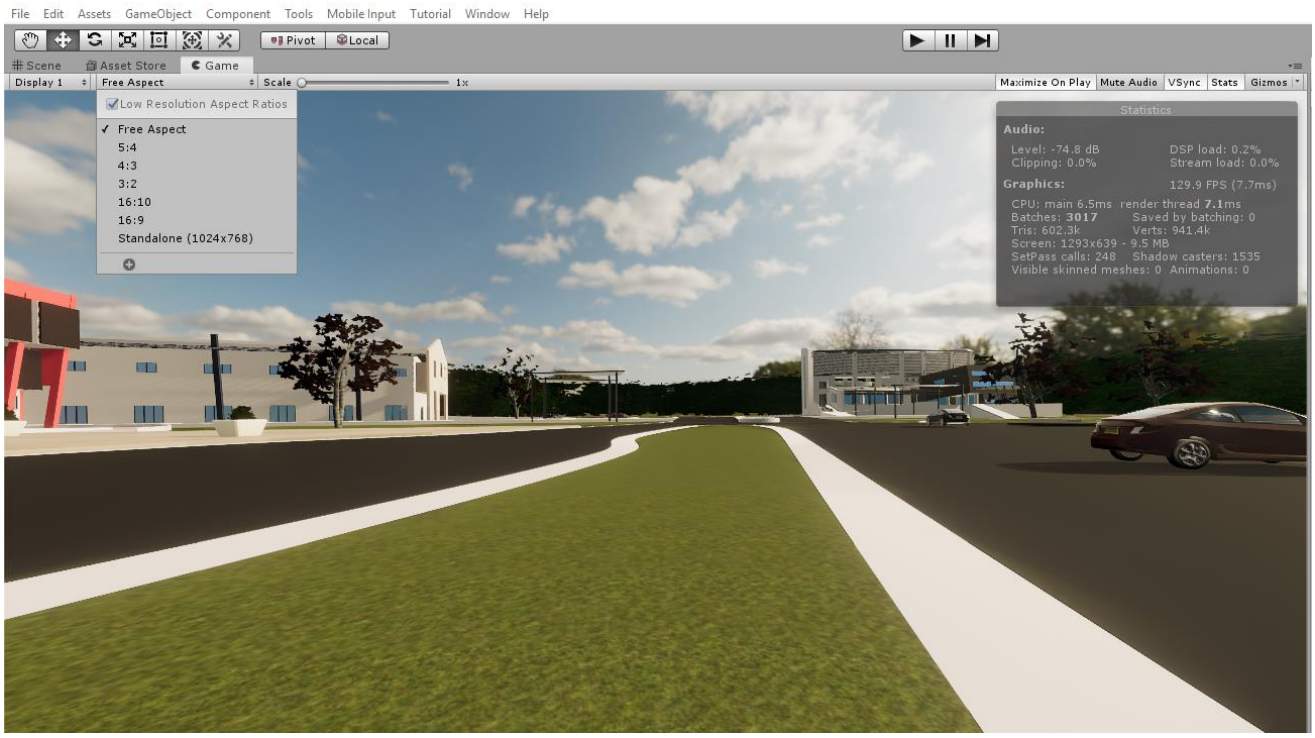
Pause : 

Με το κουμπί αυτό παγώνουμε το παιχνίδι σε ένα συγκεκριμένο Frame ώστε να μπορέσουμε να διακρίνουμε μια λεπτομέρεια, καθώς και να επεξεργαστούμε τιμές που επηρεάζουν την λειτουργία του (ταχύτητα χαρακτήρα-βαρύτητα-φωτισμός κ.α)

Next Frame : 

Με το κουμπί αυτό μπορούμε να δούμε με ακρίβεια τα κάθε Frame του παιχνιδιού και να τα επεξεργαστούμε.

Στην γραμμή εργαλείων του game view βλέπουμε αρχικά από αριστερά ένα drop down menu το οποίο μας επιτρέπει να κάνουμε προεπισκόπηση του παιχνιδιού στην επιθυμητή ανάλυση που του ορίζουμε. Στη συνέχεια προς τα δεξιά βλέπουμε το παραθυράκι "maximize on play" με το οποίο όταν το έχουμε επιλεγμένο και πατήσουμε "Play" το παιχνίδι μας παίζει σε πλήρη οθόνη ακόμα υπάρχει και η επιλογή "Stats" όπου σου δείχνει στατιστικά γραφικών.



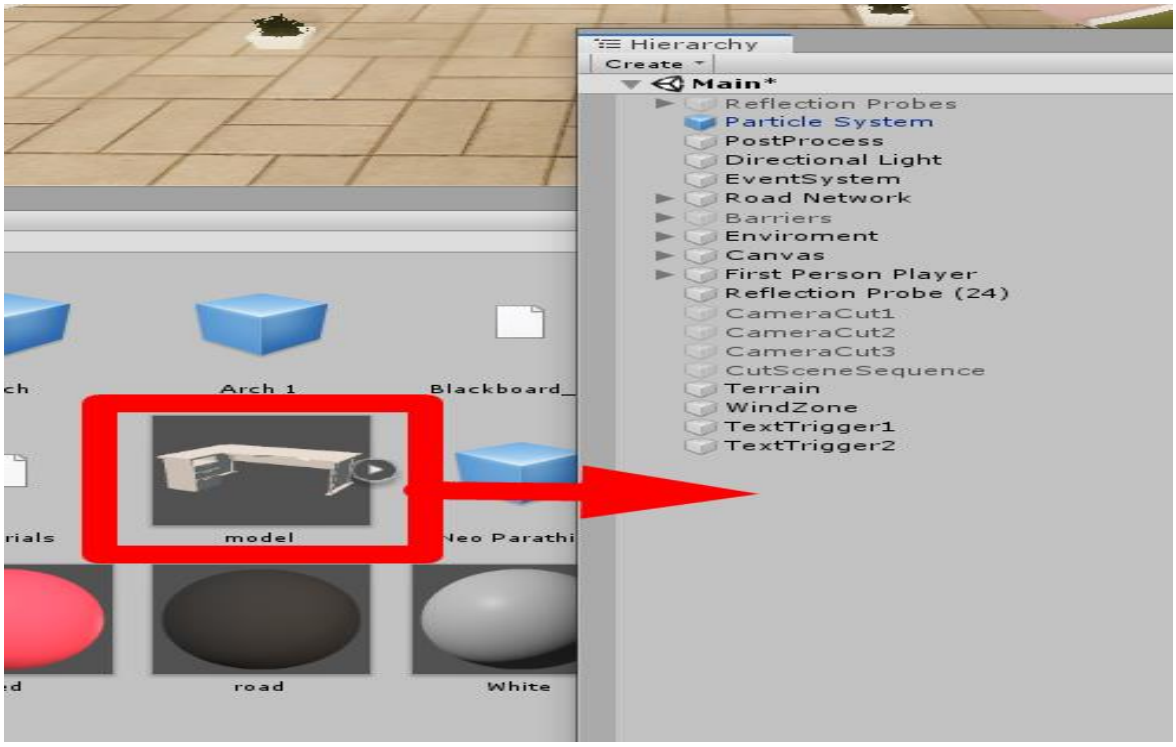
Εικόνα 7 Game View

2.5.3 Hierarchy Panel

Το Hierarchy View χρησιμοποιείται για την αποθήκευση των συγκεκριμένων αντικειμένων στη σκηνή του παιχνιδιού, όπως κάμερα, υφή 2D, υφή 3D, φως, κουτί, σφαίρες, κύβοι, μοντέλα, κάτοψη, έδαφος. Αφού δημιουργηθεί οποιοδήποτε νέο έργο παιχνιδιού, είναι προεπιλογή η δημιουργία σκηνής παιχνιδιού και η προσθήκη της κύριας κάμερας στην Ιεραρχία Προβολή της σκηνής. Η προβολή Ιεραρχίας δείχνει τα αντικείμενα του παιχνιδιού στην τρέχουσα ενεργοποιημένη σκηνή. Αντικείμενα παιχνιδιού που δημιουργούνται δυναμικά και αφαιρούνται από την ενεργοποιημένη σκηνή θα εμφανιστούν εδώ όταν δημιουργούνται και ενεργοποιούνται στη σκηνή.

2.5.4 Project Panel

Το παράθυρο Project από την άλλη περιέχει όλα τα Asset του παιχνιδιού με απλά λόγια οτιδήποτε θα χρειαστούμε στο παιχνίδι είναι εκεί (textures - scripts - prefabs κ.α).

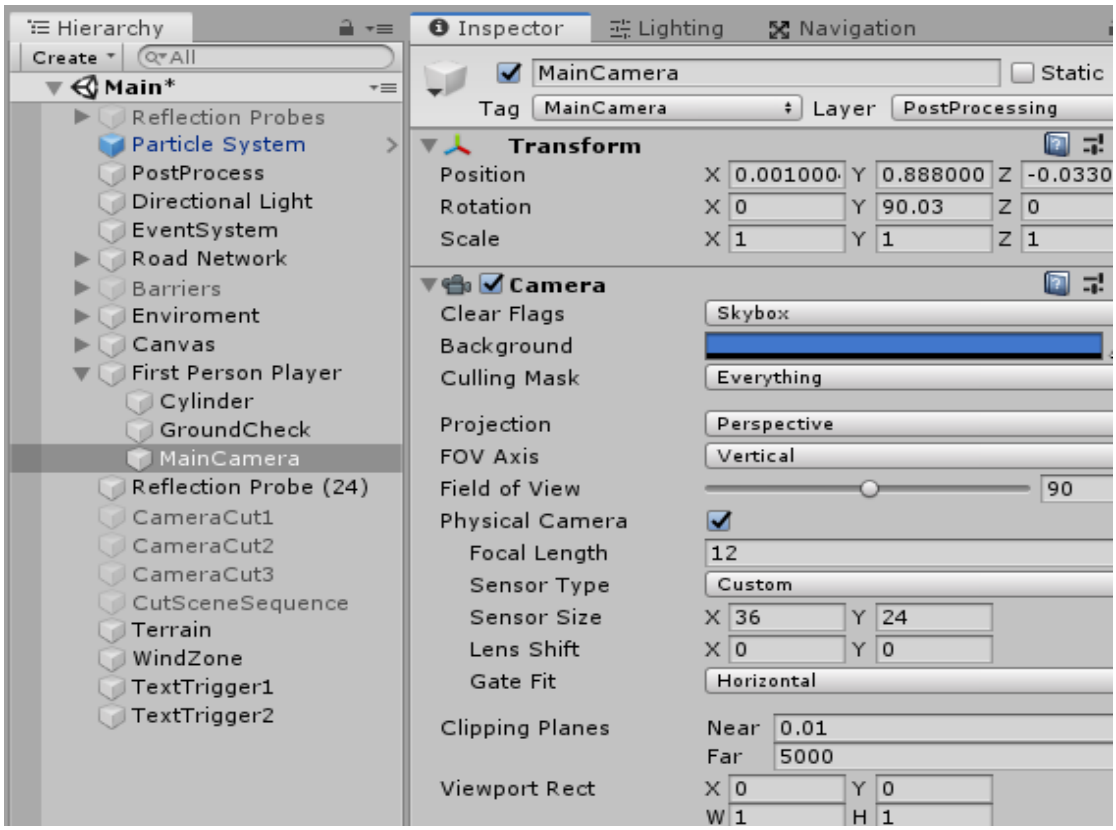


Εικόνα 8 Εισαγωγή Prefab από το Project στο Hierarchy

Το Project View περιλαμβάνει όχι μόνο τα assets που θα χρησιμοποιηθούν για το τρέχον έργο, αλλά και όλες τις σκηνές ή τα επίπεδα για το ολοκληρωμένο παιχνίδι ή την εφαρμογή. Είναι ο χώρος αποθήκευσης που αποθηκεύει τα assets που θα μπορούσαν να χρησιμοποιηθούν για το έργο. Τα assets μπορούν να διαγραφούν από τον σκληρό δίσκο σύμφωνα με την κατάργησή τους από αυτήν την τοποθεσία. Ωστόσο, πρέπει να είμαστε προσεκτικοί για να αφαιρέσουμε τα στοιχεία από τον κατάλογο του Explorer, γιατί θα προκαλούσε καταστροφή της σκηνής που χρησιμοποιεί αυτά τα στοιχεία.

2.5.5 Inspector Panel

Η προβολή Inspector χρησιμοποιείται για την πρόσβαση σε διάφορες ιδιότητες και στοιχεία για τα αντικείμενα του παιχνιδιού που ο χρήστης έχει δημιουργήσει και επιλέξει είτε στο Hierarchy Panel ή στο Project Panel. Εδώ μπορούμε να βρούμε πληροφορίες σχετικά με αντικείμενα . Για παράδειγμα, κάνοντας κλικ σε ένα αντικείμενο που βρίσκεται στην Scene view αυτήν τη στιγμή, την Κύρια κάμερα(Main Camera) παρακολουθήστε την προβολή Επιθεωρητής, η οποία φαίνεται στο παρακάτω σχήμα

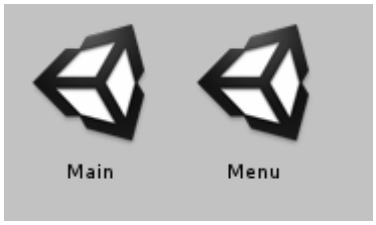


Εικόνα 9 ο Inspector με επιλεγμένη τη κύρια κάμερα

Το Inspector View είναι πιο περίπλοκο σχετικά. Μπορεί να θεωρηθεί ως ο χώρος που αποθηκεύει πληροφορίες για τα αντικείμενα του παιχνιδιού, τα στοιχεία του παιχνιδιού και τη διαμόρφωση του παιχνιδιού. Είτε επιλέξουμε ένα αντικείμενο στο Hierarchy panel είτε στο Project Panel, θα ανοίξει το Inspector View.

2.5.6 Unity Scene

Μία εφαρμογή ή ένα παιχνίδι στη Unity είναι στην πραγματικότητα ένα σύνολο σκηνών. Κάθε σκηνή μπορεί να περιέχει τα δικά της αντικείμενα με ειδικές ρυθμίσεις. Οι σκηνές χρησιμοποιούνται για να χωρίσουν το παιχνίδι σε πολλά κομμάτια και να τα χειριστούν ξεχωριστά. Συνήθως, μία εφαρμογή είναι ένα σύνολο σκηνών που ταξινομούνται σε ειδικές ακολουθίες που σχεδιάστηκαν από τον χρήστη. Οι σκηνές μπορούν να χρησιμοποιηθούν για τη δημιουργία ενός κύριου μενού, μεμονωμένων επιπέδων και οτιδήποτε άλλο. Σκεφτείτε κάθε μοναδικό αρχείο σκηνής ως ένα μοναδικό επίπεδο.

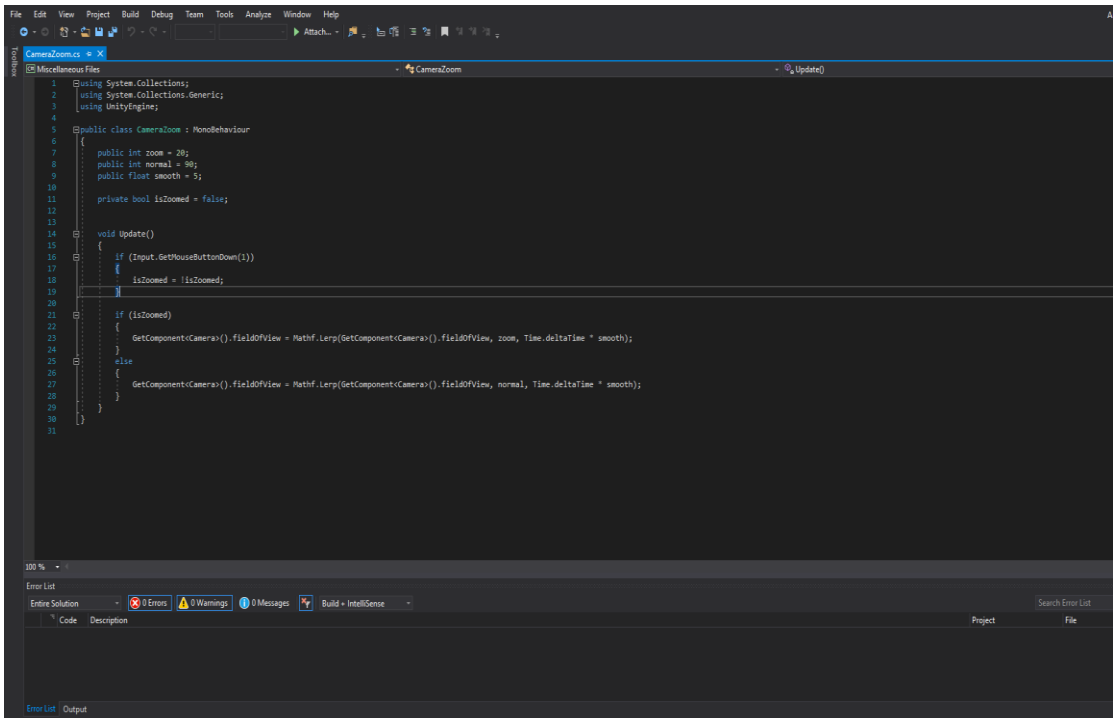


Εικόνα 10 Οι σκηνές της εφαρμογής

2.6 Προγραμματισμός

Αν και η Unity μας δίνει πολλές δυνατότητες επεξεργασίας και παραμετροποίησης χωρίς να χρειαστεί να γράψουμε ούτε μία σειρά κώδικα , για να επιτύχουμε το επιθυμητό αποτέλεσμα η χρήση προγραμματισμού κρίνεται απαραίτητη. Εγκαθιστώντας την Unity μας δίνεται η επιλογή να εγκαταστήσουμε και τον compiler Microsoft Visual Studio αλλά μπορούμε να επεξεργαστούμε με όποιον compiler θέλουμε τα script στην Unity. Η γλώσσα προγραμματισμού που χρησιμοποιεί η Unity είναι η C# όμως μπορούν να χρησιμοποιηθούν και άλλες γλώσσες

προγραμματισμού , όπως BOO και JavasScript, που χρησιμοποιούν το .NET framework, ωστόσο για να λειτουργήσουν στο περιβάλλον της Unity χρειάζονται plug-ins οπότε η ενδεδειγμένη γλώσσα είναι η C#. Για τους παραπάνω λόγους σε αυτό το project χρησιμοποίησα C# και Microsoft Visual Studio.



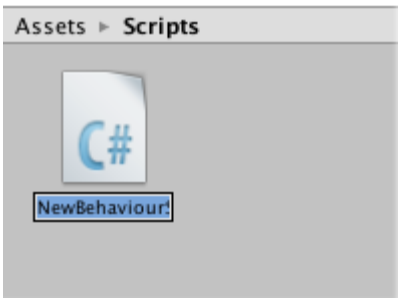
Εικόνα 11 Περιβάλλον Microsoft Visual Studio

2.6.1 Scripting

Στον προγραμματισμό υπολογιστών, ένα script(Εικόνα 10) είναι ένα πρόγραμμα ή μια ακολουθία οδηγιών που ερμηνεύονται ή εκτελούνται από άλλο πρόγραμμα και όχι από τον επεξεργαστή του υπολογιστή (όπως είναι το μεταγλωττισμένο πρόγραμμα). Αποτελούνται από τέσσερις βασικούς τύπους συστατικών: μεταβλητές, συναρτήσεις, εξισώσεις και σχόλια. Οι μεταβλητές διατηρούν τιμές που μπορούν να είναι οτιδήποτε από αριθμούς έως κείμενο. Οι συναρτήσεις πραγματοποιούν , γενικά με μεταβλητές και εξισώσεις. Τα σχόλια

αγνοούνται όταν εκτελείται ο κώδικας, επιτρέποντας στον χρήστη να κάνει σημειώσεις σχετικά με το τι είναι ο κώδικας ή τι πρέπει να κάνει ή ακόμη και να απενεργοποιήσει προσωρινά τον κώδικα.

Σε γενικές γραμμές, οι γλώσσες για Script είναι ευκολότερες και γρηγορότερες στην κωδικοποίηση από τις πιο δομημένες και μεταγλωττισμένες γλώσσες, όπως C και C ++. Ωστόσο, ένα script διαρκεί περισσότερο από ένα μεταγλωττισμένο πρόγραμμα, δεδομένου ότι κάθε εντολή αντιμετωπίζεται πρώτα από άλλο πρόγραμμα (που απαιτεί πρόσθετες οδηγίες) και όχι απευθείας από τον βασικό επεξεργαστή εντολών.



Εικόνα 12 C# script στο Project Panel

2.6.2 Scripting στη Unity

Το σύστημα scripting βασίζεται στο Mono, μια έκδοση ανοιχτού κώδικα του .NET. Όπως το .NET, το mono υποστηρίζει πολλές γλώσσες προγραμματισμού, αλλά το Unity υποστηρίζει μόνο C #, Boo και JavaScript. Κάθε γλώσσα έχει τα πλεονεκτήματα και τα μειονεκτήματά της. Η C # έχει το πλεονέκτημα ότι έχει πολλά εκπαιδευτικά βιβλία και διαδικτυακά σεμινάρια, έχει τα πιο κοντινά χαρακτηριστικά σε μια κύρια γλώσσα στο .NET και το Mono, και έτσι χρησιμοποιείται σε πολλά εμπορικά και ανοιχτού-κώδικα λογισμικά. Αλλά όπως προαναφέρθηκα, η C # είναι η πιο σωστή επιλογή και γιαυτό σε αυτό το project, η γλώσσα προγραμματισμού είναι η C#.

Το script υποδεικνύει στα GameObjects πώς να συμπεριφερόνται. Είναι τα script,τα στοιχεία που συνδέονται με τα GameObjects και πώς αλληλεπιδρούν μεταξύ τους, αυτά που δημιουργούν το πρόγραμμα μας. Το scripting στη Unity είναι διαφορετικό από τον καθαρό προγραμματισμό. Στον καθαρό προγραμματισμό, π.χ. σε μία εφαρμογή, πρέπει ο κώδικας να τρέχει την εφαρμογή, όμως στην Unity δεν χρειάζεται να δημιουργήσετε τον κώδικα που εκτελεί την εφαρμογή, επειδή η Unity το κάνει για εμάς. Αντ 'αυτού, τα scripts μας εστιάζονται στους μηχανισμούς της εφαρμογής.

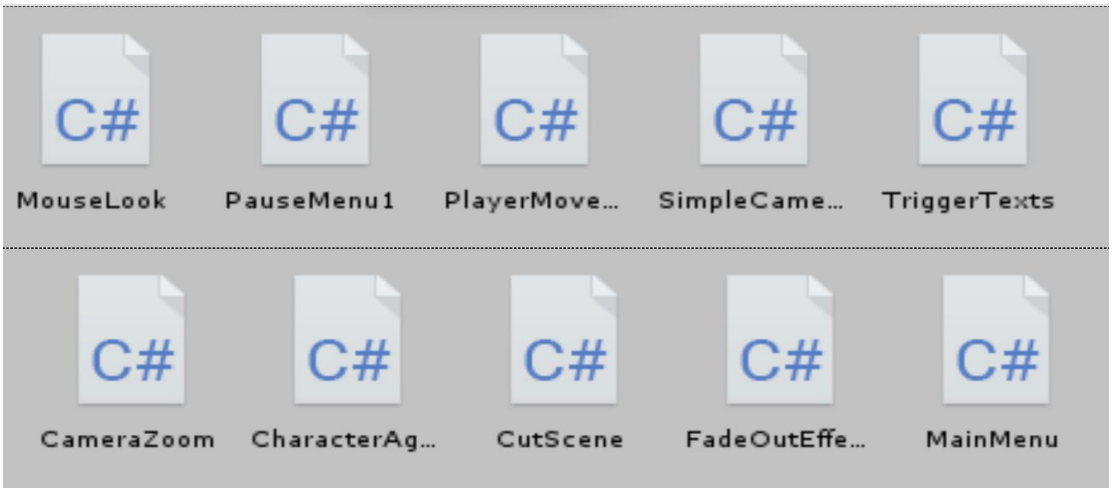
Η Unity τρέχει σε έναν μεγάλο βρόχο. Διαβάζει όλα τα δεδομένα που βρίσκονται σε μια σκηνή παιχνιδιού. Για παράδειγμα, διαβάζει το φωτισμό, τις σκιές, τα αντικείμενα, ποιες είναι οι συμπεριφορές τους και επεξεργάζεται όλες αυτές τις πληροφορίες για εμάς.

Όταν ο χρήστης δημιουργεί ένα νέο script C#, θα δημιουργηθεί ένα νέο αρχείο script στο φάκελο Assets>, όμως ο φάκελος δημιουργίας μπορεί να είναι όποιος θέλει ο χρήστης. Εάν ο χρήστης κάνει διπλό κλικ σε αυτό το αρχείο και στη συνέχεια θα ανοίξει το νέο σενάριο με το MonoDevelop ως νέο αρχείο προγραμματισμού C #. Το MonoDevelop προσθέτει αυτόματα τα απαραίτητα αρχεία κεφαλής και τάξεις

```
using UnityEngine;
using System.Collections;

public class MainPlayer : MonoBehaviour {

    // Use this for initialization
    void Start () {

    }

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

    }
}
```

Εικόνα 13 Αρχικό περιεχόμενο ενός C# script στη Unity

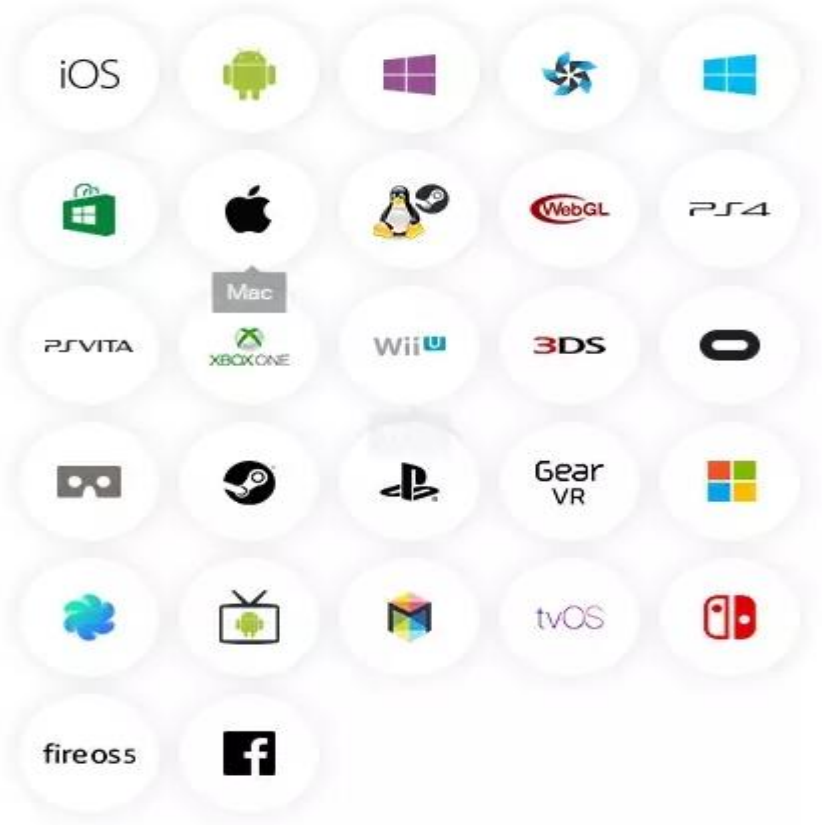
Η συνάρτηση Start() εκτελείτε την πρώτη φορά που θα κληθεί το συγκεκριμένο script. Η συνάρτηση Update είναι μία από τις πιο σημαντικές functions σε οποιαδήποτε μηχανή παιχνιδιών. Αυτή η συνάρτηση , που είναι μία από τις λειτουργίες του συστήματος, ελέγχεται τουλάχιστον κάθε καρέ κατά τη διάρκεια του χρόνου εκτέλεσης για να ανιχνευθεί εάν πρέπει να εκτελεστεί κάτι. Για παράδειγμα, μπορεί να εφαρμόσει τα κινούμενα σχέδια όταν πληρούται μια συγκεκριμένη συνθήκη με έλεγχο.



Εικόνα 14 Μερικά από τα scripts του προγράμματος

2.7 Publishing(Δημοσίευση)

Ένα από τα βασικά κριτήρια επιτυχίας του Unity είναι η δυνατότητα που δίνει στον Developer να κάνει build το παιχνίδι του σε οποιαδήποτε πλατφόρμα. Τελευταία μάλιστα το Unity 5 υποστηρίζει μέχρι και 21 διαφορετικές πλατφόρμες.

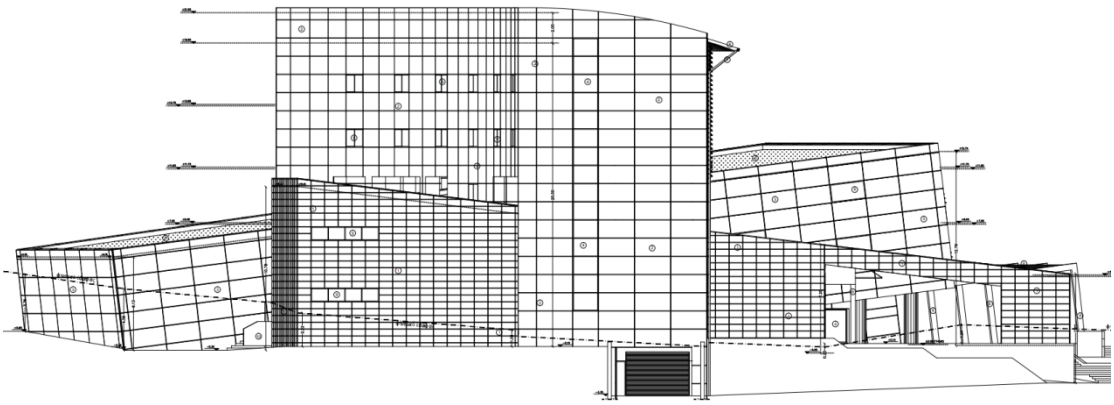


Εικόνα 15 Οι πλατφόρμες που υποστηρίζει η Unity

Ακόμα από την επιλογή Project Settings μπορούμε να επηρεάσουμε τα γραφικά που θέλουμε αλλά και τις διαστάσεις που θα τρέχει αν μιλάμε για web player.Σημαντικό θετικό για τις εφαρμογές web είναι οτι το Unity έχει δικό του Web Player με πάνω από 60 εκατομμύρια εγκατεστημένους. Τέλος να αναφέρουμε οτι η εφαρμογή μας θα υλοποιηθεί σε PC Standalone - Windows αλλά και σε Web player.

3. ΜΕΛΕΤΗ ΤΩΝ ΚΤΗΡΙΩΝ

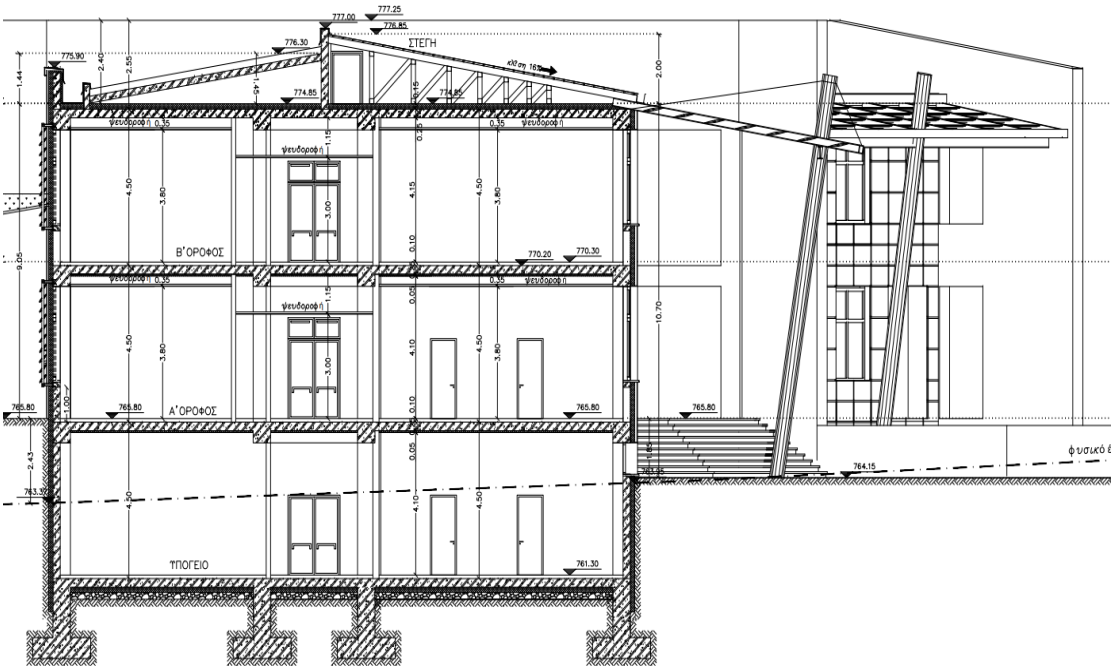
Σε αυτό το στάδιο της ανάπτυξης η μελέτη του χώρου είναι πολύ σημαντική, αυτή θα έχει ως στόχο την διαρρύθμιση του χώρου στο οποίο θα αναπαριστά η εφαρμογή αυτή. Σε πρώτο στάδιο θα πραγματοποιηθεί μια λεπτομερής μελέτη των σχεδίων της καινούργιας πανεπιστημιούπολης Κοζάνη. Όλη η καταγραφή αφού σημειωθούν όλα όσα χρειάζονται, θα αποτελέσει το πρώτο βήμα για την δημιουργία των κτηρίων στη Unity. Όσο καλύτερης ποιότητας είναι, τόσο πιο σωστή και ακριβές θα είναι η απόδοση των χώρων. Όλα τα δεδομένα θα είναι τα προσχέδια για την αναπαράσταση των χώρων που θα διαθέτει το Πανεπιστήμιο Δυτικής Μακεδονίας.



Εικόνα 16 Όψη κτηρίου Διοίκησης

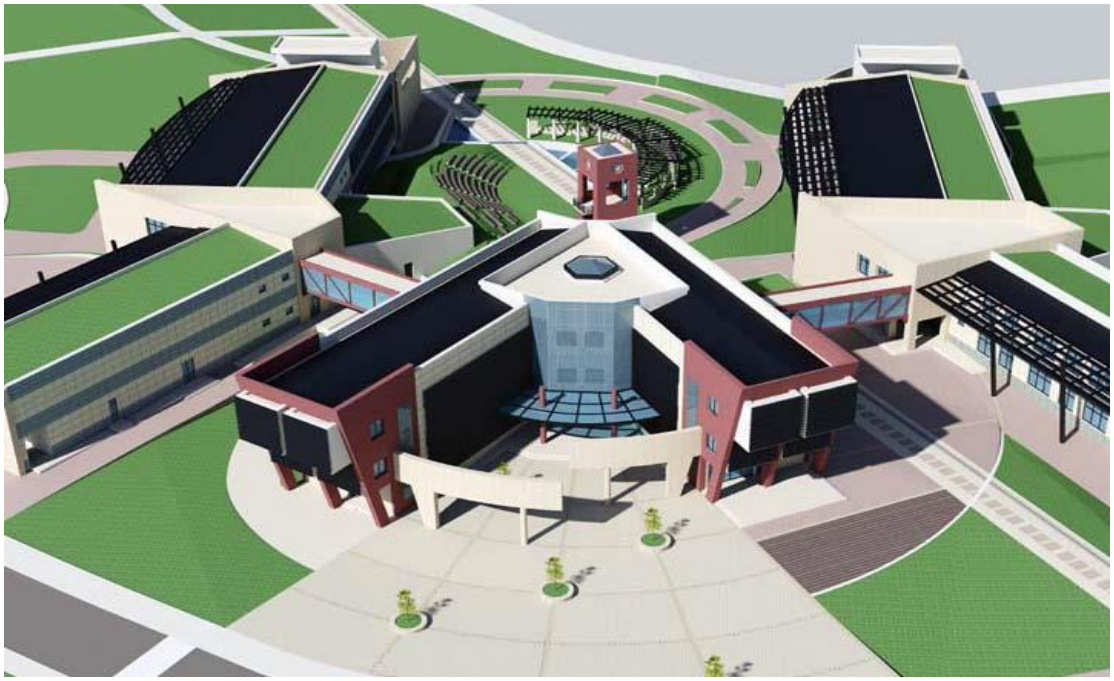
Η μοντελοποίηση τρισδιάστατων μοντέλων είναι μια διαδικασία που αποτελείται από αρκετά στάδια. Ο σχεδιασμός είναι βασισμένος σε τρεις άξονες (X, Y, Z) και η δημιουργία μοντέλων βασίζεται και στους τρεις. Είναι μια διαδικασία η οποία προαπαιτεί σωστή μελέτη πριν αν γίνει η μοντελοποίηση. Η δημιουργία του περιβάλλοντος καθώς επίσης και κάποια πιο απλά μοντέλα έχουν δημιουργηθεί μέσα από το

λογισμικό Unity3D, το οποίο λογισμικό είναι κατάλληλο για δημιουργία περιβάλλοντα, για τρισδιάστατα παιχνίδια και εφαρμογές.



Εικόνα 17 Τομή ενός τμήματος κτηρίου Φοιτητών

Επίσης μεγάλο ρόλο στο αισθητικό κομμάτι παίζει και η εμφάνιση των κτηρίων , διότι έχει μεγάλο ρόλο στην πρώτη εντύπωση.



Εικόνα 18 Μακέτα Κτηρίου Φοιτητών

Η υλοποίηση όλων τον μοντέλων για την εφαρμογή έγινε με την πολυγωνική μοντελοποίηση. Για το λόγο ότι τα κτίρια είχαν αρκετή λεπτομέρεια καθώς επίσης και τα υπόλοιπα μοντέλα. Με την πολυγωνική μοντελοποίηση, τα αποτελέσματα ήταν πολύ πιο καλά και πιο κοντά στην πραγματική εικόνα. Το περιβάλλον της εφαρμογής αποτελείται από τους χώρους αλλά και τα υπόλοιπα μοντέλα, τα οποία βρίσκονται μέσα στο χώρο. Η εφαρμογή αποτελείται από ένα επίπεδο καθώς επίσης και από τις σκηνές με το μενού και τις οδηγίες. Η μοντελοποίηση για τα μοντέλα έγινε μέσα από το λογισμικό Maya κυρίως. Το οποίο επιτρέπει να γίνει μια λεπτομερώς τρισδιάστατη μοντελοποίηση καθώς επίσης και τη εφαρμογή υφών – textures πάνω στα μοντέλα. Το πρώτο στάδιο ήταν η δημιουργία των μοντέλων και ακολούθως η εφαρμογή textures πάνω στα μοντέλα.

Εικόνα 19 Πραγματική εικόνα της υπό κατασκευής Πανεπιστημιούπολης

4. Κατασκευή Επιπέδου

Ένα επίπεδο εφαρμογής-παιχνιδιού είναι ένα τμήμα ή μέρος ενός παιχνιδιού. Τα περισσότερα παιχνίδια είναι τόσο μεγάλα που χωρίζονται σε επίπεδα, οπότε μόνο ένα μέρος του παιχνιδιού πρέπει να φορτωθεί ταυτόχρονα.

Για να ολοκληρώσει ένα επίπεδο παιχνιδιού, ένας παίκτης πρέπει να επιτύχει συγκεκριμένους στόχους ή να εκτελέσει μια συγκεκριμένη εργασία για να προχωρήσει στο επόμενο επίπεδο. Στα παιχνίδια παζλ, τα επίπεδα μπορεί να είναι παρόμοια αλλά πιο δύσκολα καθώς προχωράτε στο παιχνίδι.

Σε παιχνίδια με γραμμική πρόοδο, τα επίπεδα είναι περιοχές ενός μεγαλύτερου κόσμου. Τα παιχνίδια μπορεί επίσης να διαθέτουν διασυνδεδεμένα επίπεδα, που αντιπροσωπεύουν τοποθεσίες. Κάθε επίπεδο έχει συνήθως έναν σχετικό στόχο, ο οποίος μπορεί να είναι

τόσο απλός όσο το περπάτημα από το σημείο Α στο σημείο Β. Όταν ολοκληρωθεί ο στόχος, ο παίκτης συνήθως μεταβαίνει στο επόμενο επίπεδο. Εάν οι παίκτες αποτύχουν, πρέπει συνήθως να δοκιμάσουν ξανά το ίδιο επίπεδο ή ίσως να επιστρέψουν στην αρχή του παιχνιδιού.

Σε παιχνίδια με πολλούς ανθρώπινους παίκτες, το επίπεδο μπορεί απλώς να τελειώσει μόλις επιτευχθεί ένα όριο σε πόντους ή χρόνο. Δεν τακτοποιούν όλα τα παιχνίδια τα επίπεδα σε γραμμική σειρά. Μερικά παιχνίδια επιτρέπουν στον παίκτη να επανεξετάσει τα επίπεδα ή να τα ολοκληρώσει με οποιαδήποτε σειρά, μερικές φορές με έναν κόσμο πέρα από τον οποίο ο παίκτης μπορεί να μεταβεί από το ένα επίπεδο στο άλλο.

Ένα άτομο που δημιουργεί επίπεδα για ένα παιχνίδι είναι σχεδιαστής επιπέδων ή χαρτογράφος. Το τελευταίο χρησιμοποιείται πιο συχνά όταν μιλάμε για fps όπου τα επίπεδα αναφέρονται συνήθως ως χάρτες. Τα προγράμματα υπολογιστών που χρησιμοποιούνται για τη δημιουργία επιπέδων ονομάζονται Level Editors. Μερικές φορές απαιτείται επίσης ένας μεταγλωττιστής για τη μετατροπή της μορφής αρχείου προέλευσης σε μορφή αρχείου που χρησιμοποιείται από το παιχνίδι, ειδικά για fps. Ο σχεδιασμός επιπέδων είναι μια πολύπλοκη τέχνη που απαιτεί προσοχή για την οπτική εμφάνιση, την απόδοση του παιχνιδιού και το παιχνίδι. Η δημιουργία επιπέδων αποτελεί αναπόσπαστο μέρος του παιχνιδιού.

4.1 Διαμόρφωση εδάφους

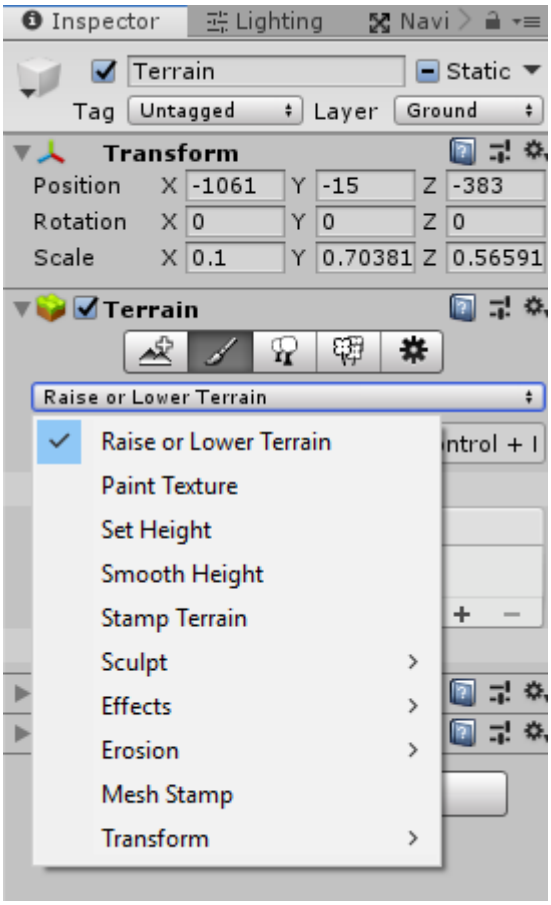
Ένα γενικό χαρακτηριστικό με πολλές μηχανές παιχνιδιών είναι ένας επεξεργαστής εδάφους(terrain). Όχι μόνο οι χρήστες μπορούμε να ζωντανέψουμε την ιδέα μας, αλλά μπορούμε να την αναπτύξουν με χλωρίδα από χόρτο σε θάμνο έως δάσος.

Η Unity παρέχει έναν πλήρη επεξεργαστή εδάφους. Στον επεξεργαστή εδάφους, μπορούν να δημιουργηθούν και να αναπτυχθούν διάφορα αντικείμενα, όπως LOD (επίπεδο λεπτομέρειας), απόσταση ορατότητας

και κινούμενες εικόνες. Αν και αυτό καθιστά εξαιρετικά απλή τη δημιουργία και την ανάπτυξη ενός εδάφους, όπως με οτιδήποτε κάνει πολλά χαρακτηριστικά για αυτά, συνοδεύεται επίσης από περιορισμούς.

Για να αποκτήσετε πιο εμπεριστατωμένες πληροφορίες σχετικά με τη δημιουργία δέντρων, πρέπει να βρείτε το Εγχειρίδιο αναφοράς και, στη συνέχεια, επιλέξτε τον Οδηγό μηχανής εδάφους.

Ένα αντικείμενο εδάφους θα δημιουργηθεί στη σκηνή τόσο στην Ιεραρχία(Hierarchy) όσο και στην προβολή Έργου με τη θέση στο 0, 0, 0 όταν ο χρήστης πατά το κουμπί Δημιουργία εδάφους. Στον Inspector, όταν το αντικείμενο Terrain έχει επιλεγεί , ο χρήστης θα δει τις ιδιότητες και τα εργαλεία που είναι διαθέσιμα για την ανάπτυξή του.



Εικόνα 20 Inpector Panel του Terrain

Τα εργαλεία του Terrain είναι τα κύρια εργαλεία για τη δημιουργία του εδάφους. Για παράδειγμα, ο χρήστης μπορεί να χρησιμοποιήσει το πρώτο κουμπί για να δημιουργήσει βουνά με διαφορετικά ύψη και το τέταρτο κουμπί για να βάψει τα δέντρα, τους θάμνους ή τα λιβάδια. Σε γενικές γραμμές, το πρόγραμμα επεξεργασίας εδάφους είναι μία από τις ισχυρότερες λειτουργίες του Unity 3D.

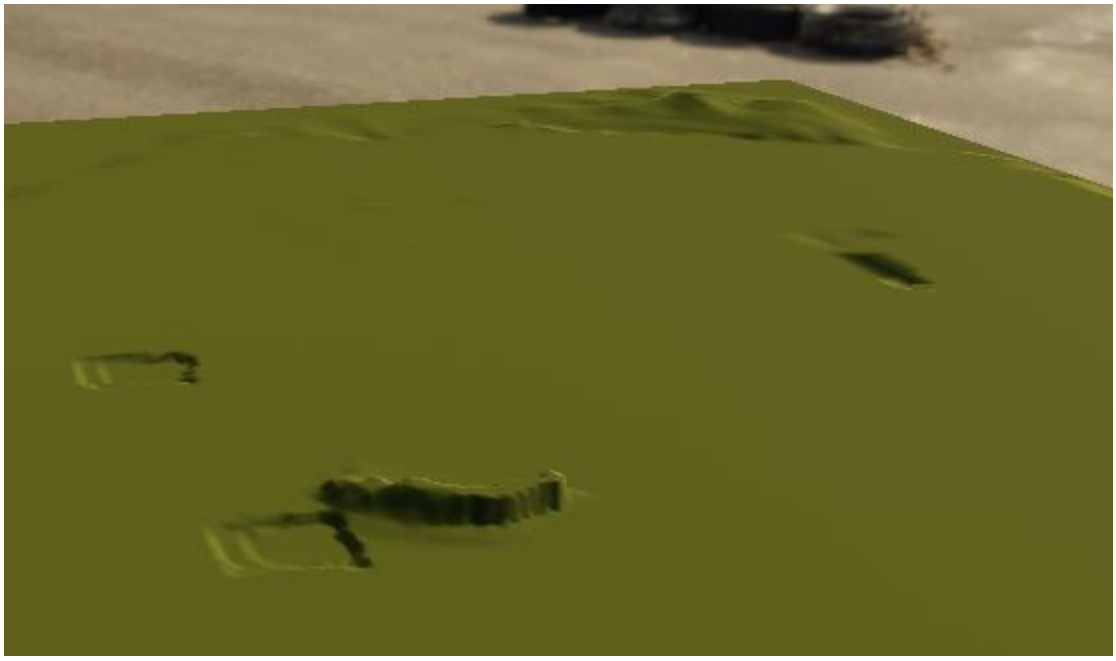
Τα πιο σημαντικά εργαλεία είναι :

- Raise or lower terrain : Μπορούμε να υψώσουμε ή να χαμηλώσουμε το έδαφος, ωστόσο δε μπορούμε να το χαμηλώσουμε πέρα από αρχικό του βάθος λόγο τεχνικών περιορισμών

- Set Height : Υψώνει το έδαφος ακριβώς την τιμή που του βάζουμε

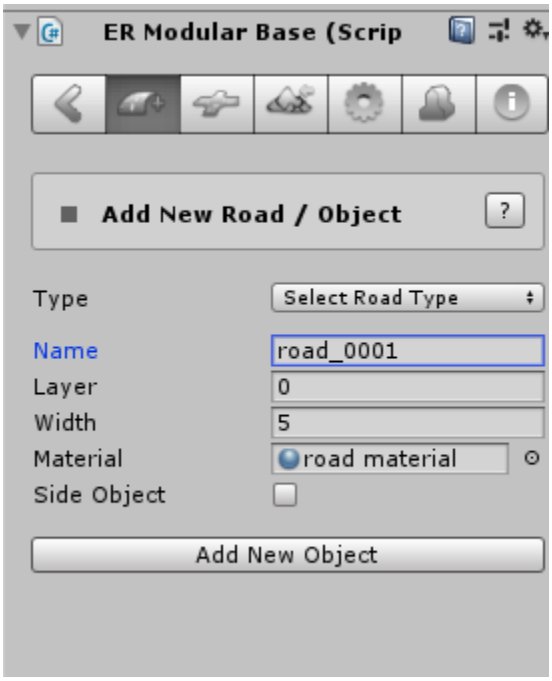
- Smooth Terrain : Κάνει ένα τραχύ έδαφος με πολλές γωνίες να είναι πιο λείο και στρωτό , κάτι που βοηθάει στην αληθοφάνεια του αποτελέσματος

- Paint Texture : Μας επιτρέπει να βάψουμε το έδαφος με το texture που θα επιλέξουμε



Εικόνα 21 Διαμόρφωση Terrain

Στην συνέχεια χρησιμοποίησα το πακέτο EasyRoads3D. Αν και χρησιμοποιείται για την κατασκευή δρόμων, αλλάζοντας το material σε άσπρο , δημιούργησα τα πεζοδρόμια.



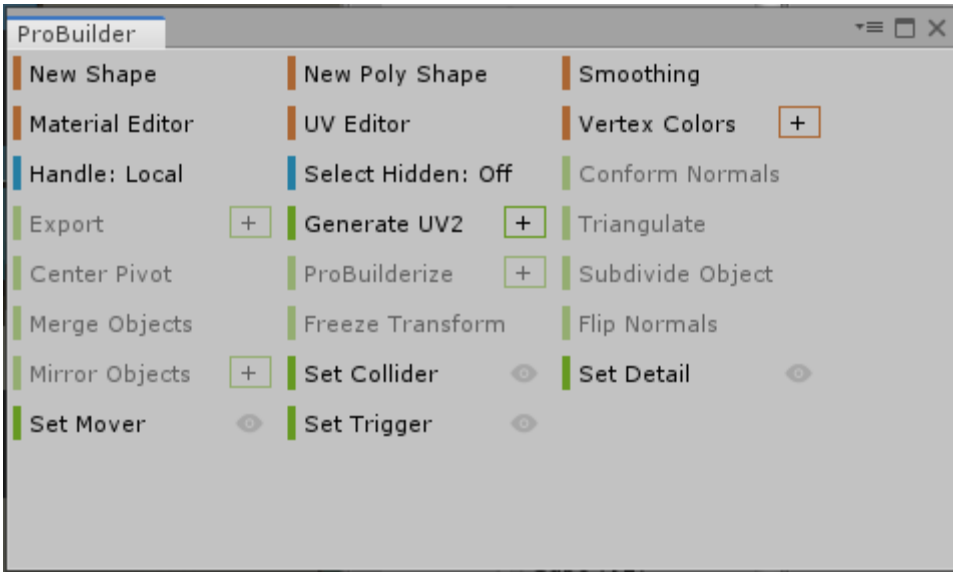
Εικόνα 22 EasyRaods3D interface

Τέλος χρησιμοποίησα την δυνατότητα Paint Texture από τον επεξεργαστή εδάφους για να βάψω του δρόμους μαύρους.

Εικόνα 23 Τελική διαμόρφωση εδάφους.

4.2 Δημιουργία Κτηρίων

Η σωστή απεικόνιση των κτηρίων είναι το άλφα και το ωμέγα σε αυτήν την διπλωματική και ήταν η διαδικασία στην οποία αφιέρωσα τον περισσότερο χρόνο. Αφού πρώτα μελέτησα τα αρχιτεκτονικά σχέδια των κτηρίων έπρεπε να αρχίσω να τα δίνω τρισδιάστατη μορφή. Υπάρχουν πολλά 3D modeling προγράμματα όπως το Maya, το Houdini και το Blender. Η Unity όμως προσφέρει τον δικό της editor για διαμόρφωση 3D αντικειμένων και κτηρίων. Λέγεται ProBuilder και βρίσκεται στο Unity Asset Store. Μετά από ενδελεχή έρευνα και δοκιμή κατέληξα ότι ο ProBuilder είναι το καταλληλότερο λογισμικό για αυτά που έπρεπε να σχεδιάσω.



Εικόνα 24 Μενού Pro Builder

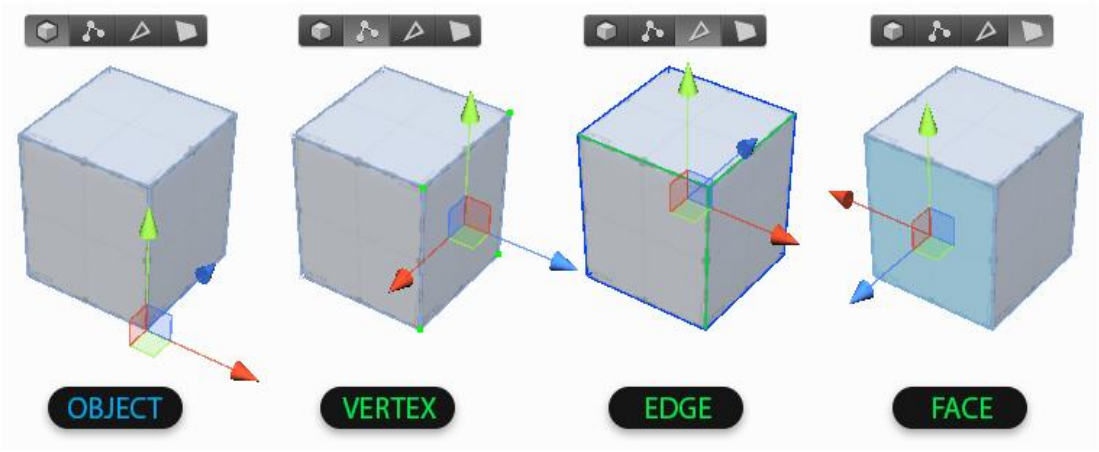
Επίσης υπάρχει ένα μικρό UI μενού στο Scene panel που έχει τις βασικότερες λειτουργίες του ProBuilder οι οποίες είναι :

- Object mode : είναι ο τρόπος συμπεριφοράς της Unity από προεπιλογή – το επιλεγμένο GameObjects μπορούμε να το μετακινήσουμε, περιστρέψουμε και να του αλλάξουμε κλίμακα

- Vertex edit : Επιλέγουμε και επεξεργαζόμαστε κορυφές για λεπτομερή επεξεργασία και λειτουργίες όπως διαχωρισμός κορυφών ή σύνδεση

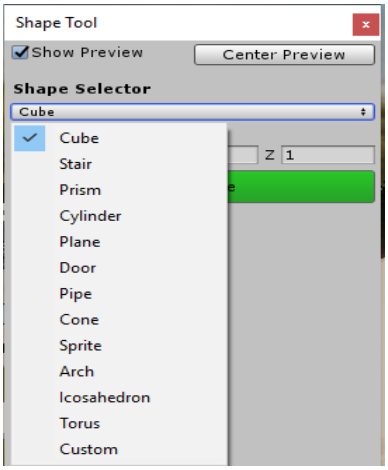
- Edge edit : Επιλέγουμε και επεξεργαζόμαστε τις άκρες των αντικείμενων

- Face edit : Επιλέγουμε και επεξεργαζόμαστε τις πλευρές ενός αντικειμένου, όπως διαγραφή πλευράς ή προέκταση



Εικόνα 25 Βασικές λειτουργίες ProBuilder

Άλλες χρήσιμες δυνατότητες είναι το New Poly Shape, στο οποίο μπορούμε να σχεδιάσουμε ότι σχήμα θέλουμε σε 2D και να το μετατρέψουμε σε 3D, και το UV Editor στο οποίο μπορούμε να αναθέσουμε σε κάθε πλευρά με διαφορετικό material. Ακόμα μπορούμε να περιστρέψουμε ή να μεγενθύνουμε τα materials που χρησιμοποιήσαμε πάνω στο αντικείμενο



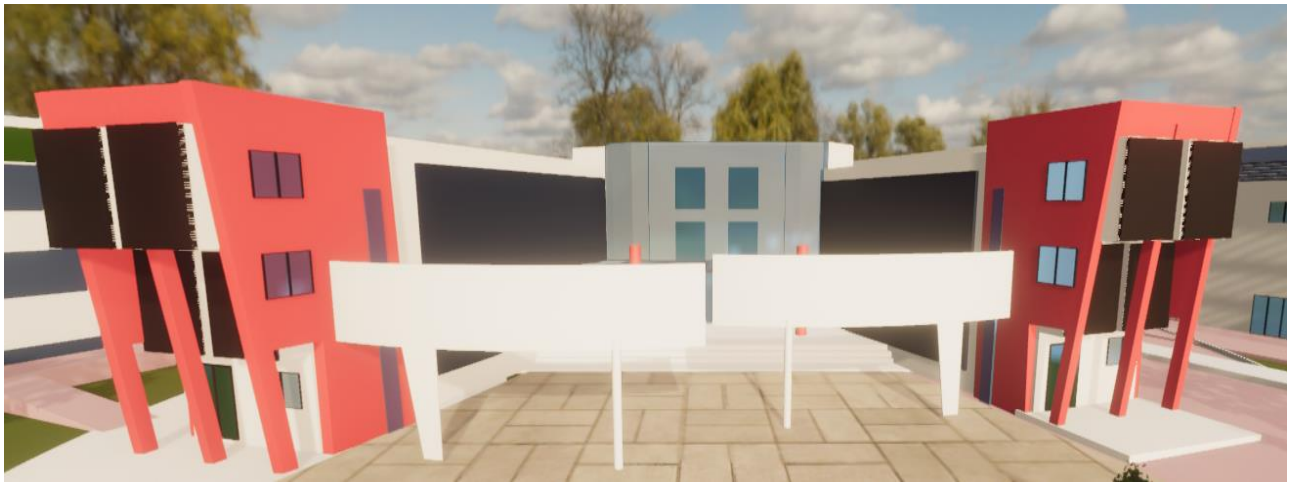
Εικόνα 26 Προσχεδιασμένα σχήματα στο ProBuilder

Το ProBuilder έχει επίσης και κάποια έτοιμα σχήματα όπως κύβος, κώνος και κύλινδρος. Το πιο σημαντικό είναι όμως είναι η αψίδα διότι θέτοντας τον αριθμό των πλευρών μεγάλο , μπορούμε να δημιουργήσουμε καμπυλωτά κτήρια και αντικείμενα, δυνατότητα που δεν δίνει απευθείας το ProBuilder. Ένα τέτοιο κτήριο είναι το κτήριο της διοίκησης .

Εικόνα 27 Βασικό μέρος κτηρίου διοίκησης

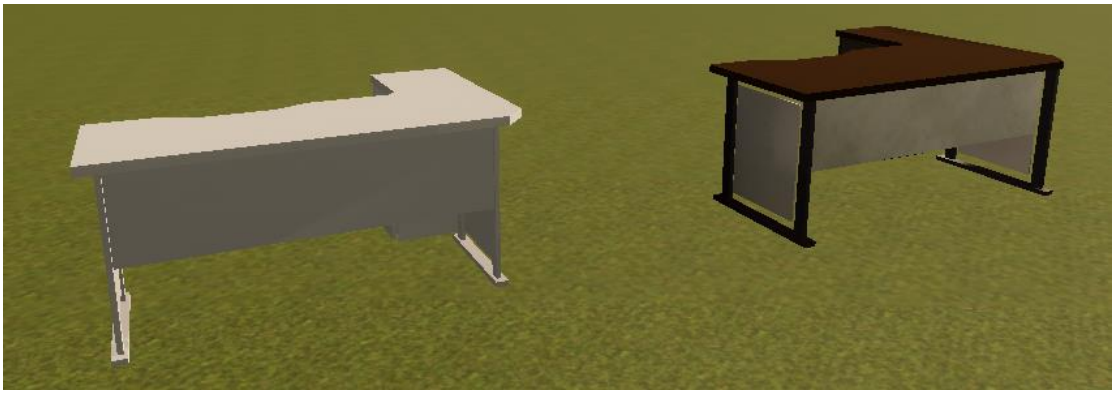
4.2.1 Εφαρμογή texture στα κτήρια

Ο επόμενο στάδιο μετά την δημιουργία των μοντέλων είναι η τοποθέτηση textures πάνω στα μοντέλα. Τα textures είναι πολύ σημαντικό στάδιο, εφόσο επηρεάζουν την όψη των μοντέλων. Ένα πολύ καλό texture μπορεί να βελτιώσει σε πολύ μεγάλο βαθμό ένα μέτριο μοντέλο. Επίσης έχουν την ιδιότητα να δίνουν την ψευδαίσθηση στον θεατή, για ανάγλυφες επιφάνειες, για βάθος και για λεπτομέρεια στα μοντέλα, τα οποία ίσως και να μήν υπαρχουν πάνω στα μοντέλα. Στην ανάπτυξη της εφαρμογής, η σωστή εφαρμογή των textures είναι πολύ σημαντική. Μπορεί να δώσει περισσότερο ρεαλιστικότητα στα μοντέλα αλλά μπορεί και να φέρει αρνητικό αποτέλεσμα αν δεν εφαρμοστεί σωστά. Μερικές από τις αρχικές φωτογραφίες των κτηρίων έδωσαν πλήρες πληροφορίες για τα textures.



Εικόνα 28 Κεντρικό Κτήριο Εκπαίδευσης

Εφαρμοστήκαν μάλιστα και πάνω στα τρισδιάστατα μοντέλα (Εικόνα 25) .



Εικόνα 29 Ένα αντικείμενο χωρίς και με textures

Εικόνα 30 Εξωτερική Διακόσμηση

4.3 Φωτισμός Επιπέδου

Η ρύθμιση της σωστής διάθεσης για την εφαρμογή-παιχνίδι καθορίζεται από τις επιλογές φωτισμού που κάνουμε. Ο φωτισμός έχει έναν από τους πιο σημαντικούς ρόλους στην εφαρμογή. Πολλές από τις ίδιες αρχές φωτισμού που ισχύουν για ταινίες και έργα μπορούν να χρησιμοποιηθούν για παιχνίδια. Ωστόσο, η φύση των παιχνιδιών καθιστά τον τρόπο παρουσίασης και εφαρμογής των αρχών δραστικά διαφορετικό.

Τα παιχνίδια είναι διαδραστικά και ο φωτισμός μπορεί συχνά να αλλάξει με βάση τις ενέργειες του χαρακτήρα. Για παράδειγμα, ο παίκτης μπορεί να έχει τη δυνατότητα να πυροβολήσει ένα φως και να αλλάξει εντελώς ενστικτωδώς τον κόσμο του παιχνιδιού. Ο φωτισμός σε μια σκηνή μπορεί επίσης να αλλάξει απλά με βάση την τρέχουσα θέση και την ορατότητα του παίκτη στο παιχνίδι. Εάν ο παίκτης μετακινηθεί σε διαφορετικό σημείο, θα βλέπει τις αντανακλάσεις και τις ακτίνες φωτός πολύ διαφορετικά.

Σε μια ταινία, ή οτιδήποτε προκαταβολικά, αυτές οι προκλήσεις δεν είναι πραγματικά παρούσες. Το φως καθορίζεται από τη γωνία της κάμερας και έχει σχεδιαστεί έτσι ώστε να φαίνεται τέλεια σε αυτήν την περίπτωση. Ένας θεατής που παρακολουθεί μια ταινία δεν έχει τη δυνατότητα να μετακινήσει την κάμερα για να πάρει διαφορετική γωνία.

Τα φώτα στα παιχνίδια συνήθως δημιουργούνται από τα πολλά διαφορετικά ενσωματωμένα ελαφριά εργαλεία που βρίσκονται στη μηχανή παιχνιδιών. Ένα από αυτά τα εργαλεία είναι η δυνατότητα “ψησίματος” φώτων(bake light) στα αντικείμενα για εξοικονόμηση χρόνου απόδοσης. Αυτό δίνει την ψευδαίσθηση ότι το φως εκπέμπει σε πραγματικό χρόνο, αλλά στην πραγματικότητα είναι ακριβώς το φως που έχει “ψηθεί” πάνω στην επιφάνεια των αντικειμένων.

Μια λάμπα που κρέμεται σε τοίχο μπορεί να χρησιμοποιηθεί ως παράδειγμα. Το φως που εκπέμπεται μπορεί να προσομοιωθεί από έναν ελαφρύ χάρτη και στη συνέχεια το πραγματικό φως μπορεί να αφαιρεθεί. Αυτή η τεχνική είναι μια εξαιρετική εξοικονόμηση χρόνου

και επιτρέπει την προσθήκη περισσότερων λεπτομερειών στο παιχνίδι, αλλά σημαίνει επίσης ότι τα φώτα που ψήνονται δεν θα έχουν καμία επίδραση στην κίνηση αντικειμένων.

Επομένως, είναι σημαντικό να γνωρίζουμε ποια φώτα να ψήνουμε και ποια φώτα πρέπει να έχουμε στη μηχανή παιχνιδιών, δηλαδή να εκπέμπουν φως σε πραγματικό χρόνο.

Καθώς οι μηχανές παιχνιδιών εξελίσσονται και το υλικό που παράγεται βελτιώνετε, το ίδιο συμβαίνει και με τον τρόπο προσομοίωσης του φωτός στο παιχνίδι. Στο παρελθόν, ο φωτισμός παραβλέφθηκε ή απλά δημιουργήθηκε χωρίς ιδιαίτερη προσοχή στη λεπτομέρεια, επειδή το υλικό και οι μηχανές δεν ήταν στο ίδιο επίπεδο. Καθώς το υλικό εξελίσσεται, το ίδιο κάνουν και οι προσδοκίες του παίκτη για το τι αναμένει να δει ενώ παίζει ένα παιχνίδι.

Ο κακός φωτισμός μπορεί να καταστρέψει τελείως μια διαφορετικά πολύ καλή εμπειρία. Ανεξάρτητα από το πόσο εκπληκτικά είναι τα μοντέλα ή τα textures, εάν ο φωτισμός δεν είναι σωστός, το αποτέλεσμα δεν θα είναι το αναμενόμενο.

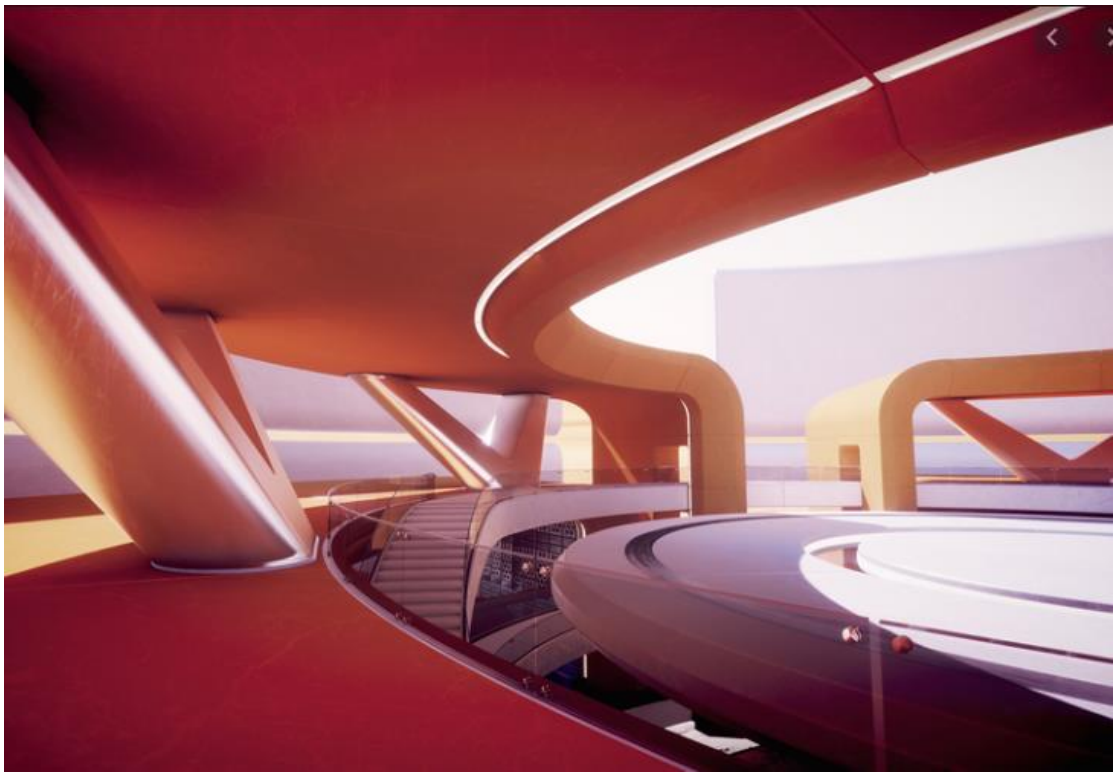
Αν κοιτάξουμε μια ταινία, ξέρουμε πόσο ζωτικός είναι ο φωτισμός. Ο φωτισμός μιας σκηνής δεν τοποθετείται ποτέ χωρίς λόγο, επειδή ο φωτισμός παίζει κρίσιμο ρόλο στη διάθεση και το συναίσθημα που παρουσιάζονται στο κοινό.

Καθώς τα παιχνίδια προσπαθούν να βυθίσουν τους παίκτες σε μια πιο κινηματογραφική εμπειρία, ο φωτισμός χρειάζεται εξίσου φροντίδα και σκέψη όπως σε μια ταινία.

Το φως μπορεί να κάνει ένα φαινομενικά αβλαβές περιβάλλον να φαίνεται απειλητικό. Μπορεί να κάνει ένα μέρος να αισθάνεται ζεστό και φιλόξενο ή τρομακτικό και κρύο.

4.3.1 Global Illumination

Στο Project γίνεται χρήση της τεχνολογίας GI(Global Illumination). Είναι ένα σύστημα που διαμορφώνει τον τρόπο με τον οποίο το φως αναπηδά από τις επιφάνειες σε άλλες επιφάνειες (έμμεσο φως) αντί να περιορίζεται μόνο στο φως που χτυπά μια επιφάνεια απευθείας από μια πηγή φωτός (άμεσο φως). Η μοντελοποίηση του έμμεσου φωτισμού επιτρέπει εφέ που κάνουν τον εικονικό κόσμο να φαίνεται πιο ρεαλιστικό και συνδεδεμένο, καθώς τα αντικείμενα επηρεάζουν την εμφάνιση του άλλου. Ένα κλασικό παράδειγμα είναι η «αιμορραγία χρώματος» όπου, για παράδειγμα, το φως του ήλιου που χτυπά έναν κόκκινο καναπέ θα προκαλέσει την ανάκαμψη του κόκκινου φωτός στον τοίχο πίσω από αυτόν. Ένα άλλο είναι όταν το φως του ήλιου χτυπά το πάτωμα στο άνοιγμα μιας σπηλιάς και αναπηδά γύρω από το εσωτερικό, έτσι ώστε τα εσωτερικά μέρη του σπηλαίου να φωτίζονται επίσης.



Εικόνα 31 Παράδειγμα Global Illumination

4.3.2 Ουρανός

Η πιο γενική μέθοδος για τη δημιουργία ενός ουρανού με υφές σύννεφου και καιρού είναι η εισαγωγή ενός skybox. Στην πραγματικότητα πρόκειται για έναν εσωτερικό κύβο τοποθετημένο πάνω από την κύρια κάμερα με μία φαινομενική εικόνα του ουρανού.

Η Unity παρέχει διάφορα skybox. Για να τα χρησιμοποιήσουμε, πρέπει να εισάγουμε το πακέτο skybox κάνοντας δεξί κλικ στο στοιχείο Asset> Import Package> Skyboxes. Για να εφαρμόσουμε ένα skybox, επιλέγουμε την κύρια κάμερας στην ιεραρχία και βρίσκουμε τη θέση του στοιχείου της κάμερας στο Inspector , στη συνέχεια ορίζουμε την τιμή του Clear Flags στο Skybox.

Στο συγκεκριμένο Project δεν χρησιμοποίησα κάποιο Skybox από τα Assets αλλά έκανα import μία φωτογραφία που χρησιμοποιεί τεχνολογία HDRI. Το HDRI είναι μια πανοραμική φωτογραφία, η οποία καλύπτει όλες τις γωνίες από ένα σημείο και περιέχει μια μεγάλη ποσότητα δεδομένων (συνήθως 32 bit ανά pixel ανά κανάλι), τα οποία μπορούν να χρησιμοποιηθούν για τον φωτισμό της σκηνής .

Εικόνα 32 HDRI Ουρανός

4.3.3 Post-Processing

Το post-processing είναι η διαδικασία εφαρμογής φίλτρων και εφέ σε μια σκηνή ενός παιχνιδιού ή μιας εφαρμογής! Σε αυτήν τη διαδικασία, η Unity προτού αποδώσει την εικόνα στην οθόνη θα προκύψει μια άλλη διαδικασία απόδοσης εικόνας σχετικά με την αρχική εικόνα, στην οποία θα εφαρμοστούν πολλά εφέ και φίλτρα απευθείας σε αυτήν.

Δεδομένου ότι είναι μια γραφική επιλογή, αυτή η διαδικασία χρησιμοποιείται για να συμβάλει στον οπτικό τομέα του παιχνιδιού, επιτρέποντας στον χρήστη να κάνει την εικόνα του παιχνιδιού όσο πιο κοντά γίνεται στην πραγματικότητα.

Πως το χρησιμοποιώ :

- Δεξί κλικ σε ένα φάκελο στο "Project" για να δημιουργήσουμε το Post-Processing προφίλ

Εικόνα 33 Post Process προφίλ στο Project Panel

- Επιλέγουμε την Camera της σκηνής και την τοποθετούμε σε
- ένα ξεχωριστό layer που δημιουργήθηκε μόνο για το Post-Processing

- Προσθέτουμε το component " Post-processing Layer " στο "αντικείμενο" της κάμερας και μετά επιλέγουμε το Layer που δημιουργήσαμε για το Post-Processing

- Δημιουργούμε ένα empty object στη σκηνή και προσθέτουμε το component "Post-Process Volume" , σέρνουμε το αντικείμενο στο Layer (Post-Process) και, στη συνέχεια, σέρνουμε το προφίλ Post-Process που δημιουργήθηκε στην ιδιότητα "Profile" από το Post-process Volume

- Για να ρυθμίσουμε τα εφέ κάνουμε κλικ στο "Add Effects" και επιλέγουμε αυτό που σας ενδιαφέρει

Από όλες τις επιλογές που έχουμε, αυτά που έκανα την μεγαλύτερη διαφορά στη τελική εικόνα το προγράμματος είναι το Color Grading, το Bloom και το Depth of Field:

- Color Grading : Καθορίζει τα χρώματα της τελικής εικόνας της σκηνής μας εφαρμόζοντας ένα είδος φίλτρου οθόνης

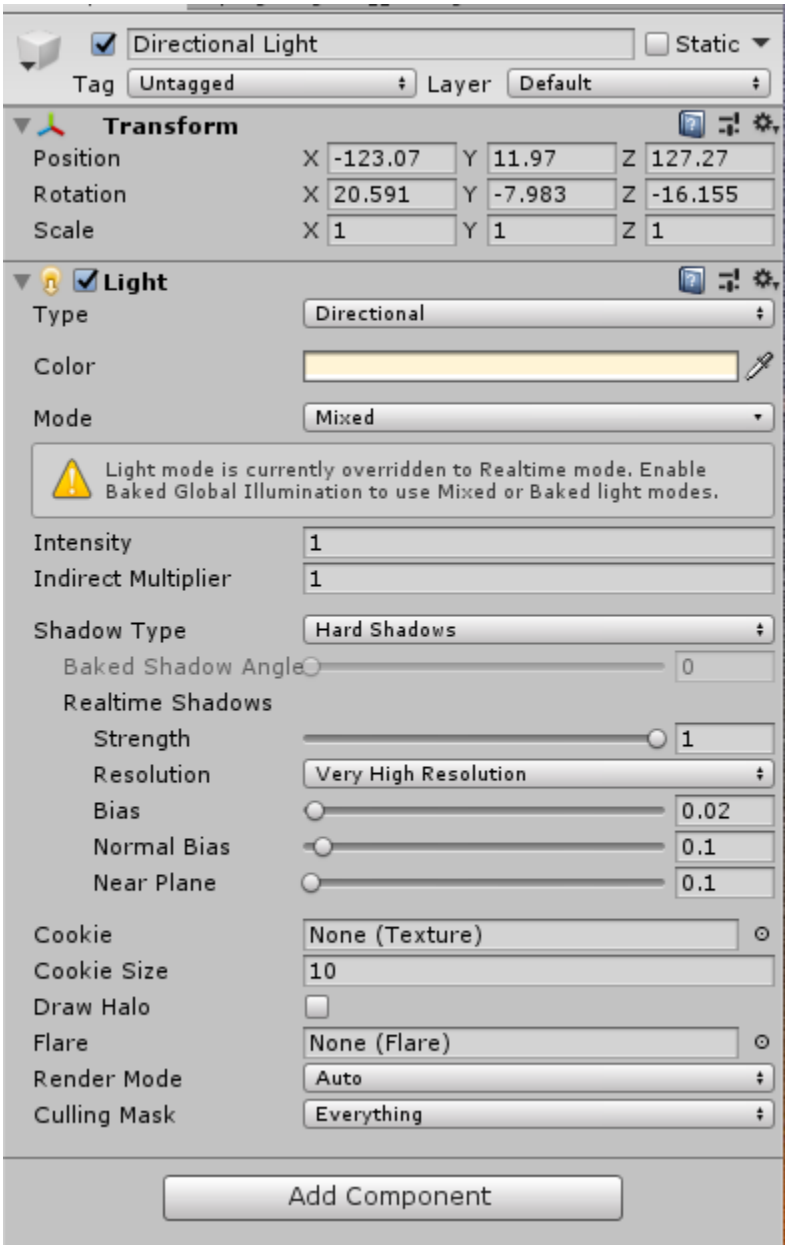
- Bloom : Κάνει τα φωτεινότερα σημεία του φωτός ελαφρώς «θολά», δίνοντας μία πολύ ωραία αίσθηση στον παίχτη

- Depth of Field : Προσομοιώνει τις ιδιότητες μιας πραγματικής κάμερας θολώνοντας αντικείμενα σε απόσταση από την κάμερα

Εικόνα 34 Πριν και μετά την εφαρμογή των παραπάνω εφέ

4.3.4 Ήλιος

Η βασική πηγή φωτός του προγράμματος είναι το Directional Light component το οποίο δημιουργείται αυτόματα με την δημιουργία του Project. Τα κυριότερα χαρακτηριστικά του είναι το Intensity, δηλαδή πόσο έντονη θα είναι η φωτεινότητα στη σκηνή αλλά και το Shadow Type, το οποίο καθορίζει την ποιότητα και την δύναμη των σκιών των αντικειμένων στη σκηνή από το φως του ήλιου.



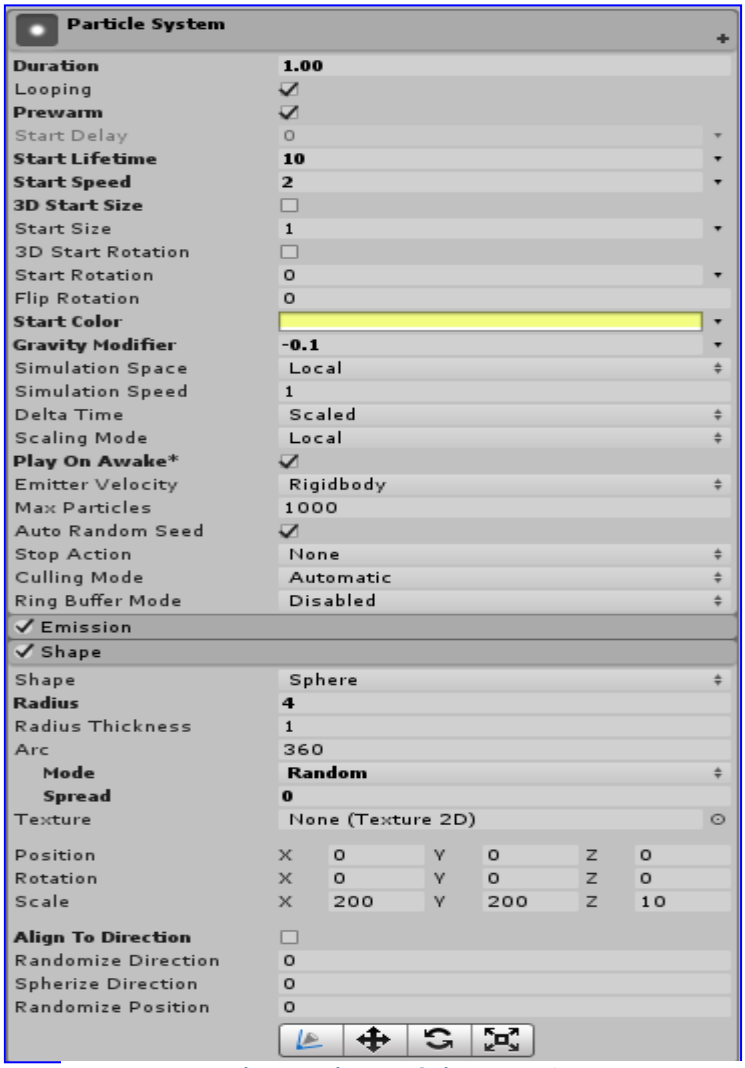
Εικόνα 35 Inspector Panel του Directional Light

4.4 Particle System

Η Unity είναι σε θέση να δημιουργεί σωματίδια με την πάροδο του χρόνου χρησιμοποιώντας ένα σημείο ως αναφορά, αυτό λέγεται Particle System ή σύστημα σωματιδίων. Ενώ μπορεί να χρησιμοποιηθεί σε διαφορετικές καταστάσεις, σε αυτό το παιχνίδι χρησιμοποιείται για την προσομοίωση μικρών φωτών στη σκηνή.

Τα μικρά αυτά φώτα είναι πολλά σωματίδια που δημιουργούνται τυχαία μέσα σε μία νοητή σφαίρα, υψώνονται στον ουρανό και μετά

από ένα συγκεκριμένο χρονικό διάστημα καταστρέφονται . Υπάρχουν πολλές παράμετροι που ρυθμίζουν όλες τις λεπτομέρειες σχετικά με τη συμπεριφορά τους, αλλά οι πιο σημαντικές είναι οι παρακάτω.



Εικόνα 36 Κύριες ρυθμίσεις Particle System

5. Λειτουργικό Μέρος

Το λειτουργικό μέρος του προγράμματος είναι ένα άλλο πολύ σημαντικό κομμάτι του project. Σε προηγούμενο κεφάλαιο, σημειώθηκε ότι τα scripts είναι το κλειδί που κάνει τα αντικείμενα του παιχνιδιού να λειτουργούν. Σε αυτή την ενότητα θα δούμε πώς εφαρμόστηκαν όλα τα scrits πάνω στα αντικείμενα(game objects).

Όλα τα scripts είναι γραμμένα στην γλώσσα προγραμματισμού C#. Στη μηχανή παιχνιδιών Unity, όλα τα αντικείμενα παιχνιδιού που έφτιαξε ο σχεδιαστής πρέπει να εμφανίζονται στο παράθυρο "Hierarchy" (Ιεραρχία) και στη συνέχεια, αυτά τα αντικείμενα τοποθετούνται στο παράθυρο "Scene". Σε αυτήν την ενότητα, θα δούμε τις θεμελιώδεις λειτουργίες του προγράμματος.

Υπάρχουν δύο είδη τρόπων δημιουργίας αντικειμένων παιχνιδιού(game oject) :

- **Δημιουργία**: Μπορούμε να κάνουμε κλικ στο κουμπί "Create" στο παράθυρο "Ιεραρχία" για να δημιουργήσουμε νέα αντικείμενα παιχνιδιού, όπως "Cube", "Sphere", "Κάμερα" κ.λπ. Αυτά τα αντικείμενα παιχνιδιού πρέπει να σχεδιαστούν και να προσαρμοστούν από τους σχεδιαστές στα τελικά αντικείμενα που επιθυμούν.

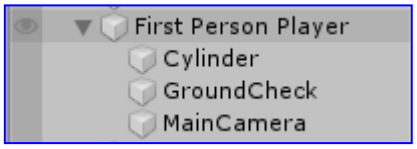
- **Prefabs** : Όταν βρεθούμε σε μία κατάσταση όπου πρέπει να χρησιμοποιήσουμε το ίδιο αντικείμενο παιχνιδιού αρκετές φορές, είναι αρκετά ενοχλητικό να δημιουργούμε το ίδιο αντικείμενο

επανειλημμένα. Η μηχανή παιχνιδιών Unity υποστηρίζει έναν τρόπο επίλυσης αυτής της κατάστασης, και αυτό είναι τα «Prefabs». Οι σχεδιαστές μπορούν να δημιουργήσουν αντικείμενα παιχνιδιού με τα πρόθεμα στο παράθυρο «Έργο». Όταν θέλουμε να χρησιμοποιήσουμε τα αντικείμενα, απλά σέρνουμε το prefab από το παράθυρο Project στο παράθυρο «Ιεραρχία».

5.1 Κίνηση Χαρακτήρα

Ο έλεγχος του χαρακτήρα είναι μια πολύ σημαντική πτυχή κάθε παιχνιδιού, καθώς αυτό δίνει στον παίκτη την αίσθηση ότι βρίσκεται πραγματικά στην πανεπιστημιούπολη. Υπάρχουν διαφορετικοί τρόποι εφαρμογής του ελέγχου χαρακτήρα και υπήρξαν πολλές αλλαγές σε όλο το παιχνίδι.

Αρχικά δημιούργησα ένα άδειο αντικείμενο(Εικόνα 34) που το ονόμασα First Person Player στο οποίο έβαλα μέσα το φυσικό κομμάτι του χαρακτήρα, την κάμερα και το Ground Check. Ο χαρακτήρας είναι ένας κύλινδρος, οπότε ο καλύτερος τρόπος για να το μετακινήσουμε είναι να χρησιμοποιήσουμε τα πιο τυπικά πλήκτρα κίνησης χαρακτήρων (W, A, S και D) τα οποία η Unity έχει ήδη ρυθμισμένα εξ αρχής. Κάθε πλήκτρο προσθέτει μια δύναμη στον χαρακτήρα και τον οδηγεί σε μια κίνηση προς αυτήν την κατεύθυνση.

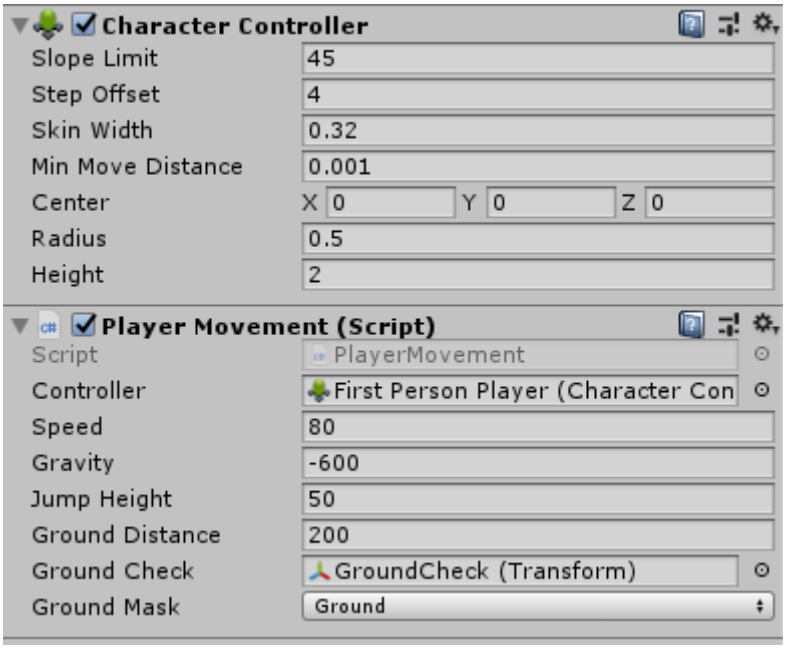


Εικόνα 37 Empty Object Χαρακτήρα

Για να επιτευχθεί αυτή η κίνηση έχω αναθέσει ένα script και ένα Character Controller component στο First Person Player αντικείμενο. Το Character Controller μας επιτρέπει να προσομοιώσουμε εύκολα την κίνηση του χαρακτήρα η οποία περιορίζεται από συγκρούσεις π.χ. να μη μπορεί να ανέβει ψηλούς τοίχους.

Με το script μπορούμε να διαλέξουμε την ταχύτητα του παίχτη, το ύψος που μπορεί να πηδάει αλλά και την βαρύτητα την οποία δέχεται όταν είναι στον αέρα.

Με το ground check ελέγχουμε αν ο χαρακτήρα είναι στο έδαφος, στην ουσία με αυτή την τακτική μπορούμε να απαγορέψουμε στον παίχτη να πηδήξει για όσο είναι στον αέρα(Εικόνα 35).



Εικόνα 38 Ρυθμίσεις Character Controller και PlayerMovement

```
public CharacterController controller;

public float speed = 12f;
public float gravity = -9.81f;
public float jumpHeight = 3f;

public float groundDistance = 0.4f;
public Transform groundCheck;
public LayerMask groundMask;

Vector3 velocity;
bool isGrounded;

// Update is called once per frame
void Update()
{
    isGrounded = Physics.CheckSphere(groundCheck.position, groundDistance, groundMask);

    if (isGrounded && velocity.y < 0)
    {
        velocity.y = -2f;
    }

    float x = Input.GetAxis("Horizontal");
    float z = Input.GetAxis("Vertical");

    Vector3 move = transform.right * x + transform.forward * z;
    controller.Move(move *speed * Time.deltaTime);

    if (Input.GetButtonDown("Jump") && isGrounded)
    {
        velocity.y = Mathf.Sqrt(jumpHeight * -2f * gravity);
    }

    velocity.y += gravity * Time.deltaTime;

    controller.Move(velocity * Time.deltaTime);
```

Εικόνα 39 Player Movement Script

5.2 Μοντέλο Χαρακτήρα

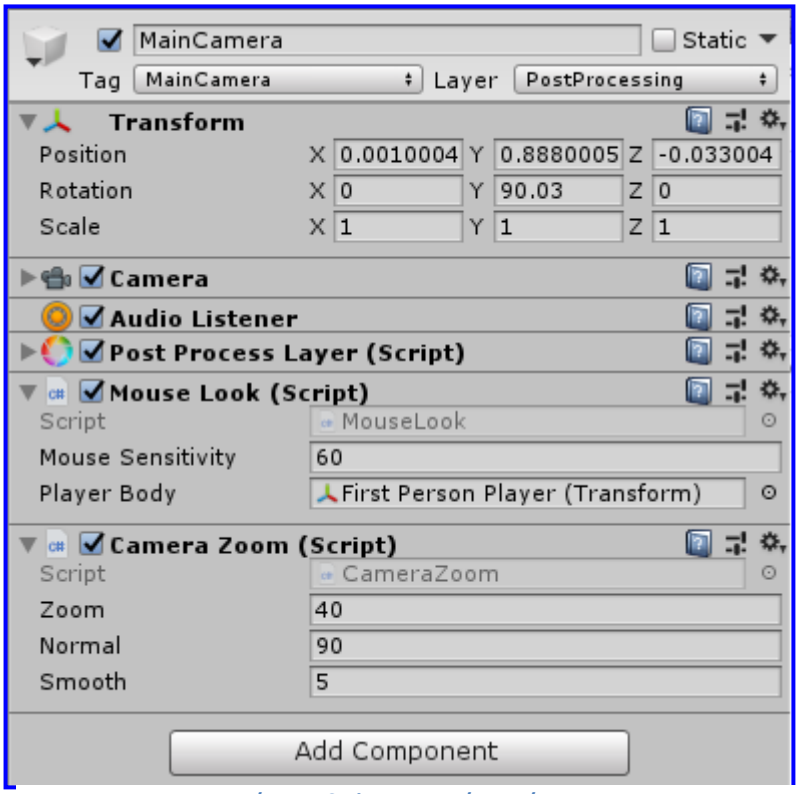
Ένα άλλο πολύ σημαντικό κομμάτι στην ωραία εικόνα ενός τέτοιου προγράμματος αναπαράστασης είναι η μορφή του παίχτη που θα ελέγχει ο χρήστης. Στην προκειμένη περίπτωση χρησιμοποίησα ένα δωρεάν 3D μοντέλο από το Unity Store με την ονομασία Anime Character : Souta (Free). Περιείχε μαζί και ένα controller script, στο οποίο έγιναν αλλαγές για να ταιριάζει στο πρόγραμμα, αλλά και animator το οποίο περιέχει τις γραφικές κινήσεις του χαρακτήρα μας.

Εικόνα 40 3D Μοντέλο Χαρακτήρα

5.3 Κάμερα

Η κάμερα στη Unity δημιουργείται ως αντικείμενο παιχνιδιού στην προβολή Ιεραρχίας και δείχνει αυτόματα στο παράθυρο Scene(σκηνή). Σε ένα παιχνίδι η κάμερα παρέχει την ορατή περιοχή στην οθόνη, πράγμα που σημαίνει ότι η κάμερα παρέχει το ύψος και το πλάτος της προβολής, επίσης το βάθος μπορεί να ρυθμιστεί. Ολόκληρος ο ορατός χώρος από μια κάμερα ονομάζεται όγκος προβολής. Εάν ένα αντικείμενο που δημιουργείται στη σκηνή δεν τοποθετείται μέσα στον όγκο προβολής, δεν μπορεί να εμφανιστεί στην οθόνη.

Στο πρόγραμμα αυτό η κύρια κάμερα υπάρχει μέσα στο αντικείμενο First Person Player(Εικόνα 33). Έχει πάνω της δύο scripts, το Mouse Look και το Camera Zoom, όπως επίσης και το Post Process Layer script το οποίο αναφέραμε σε προηγούμενη ενότητα. Το Mouse Look script δίνει στον παίχτη την δυνατότητα να κοιτάει ελεύθερα όπου θέλει χωρίς να είναι ‘’κλειδωμένη’’ στον παίχτη. Το Camera Zoom script δίνει την δυνατότητα στον παίχτη με το δεξί κλικ του ποντικιού να κάνει zoom η κάμερα στο σημείο όπου κοιτάει.

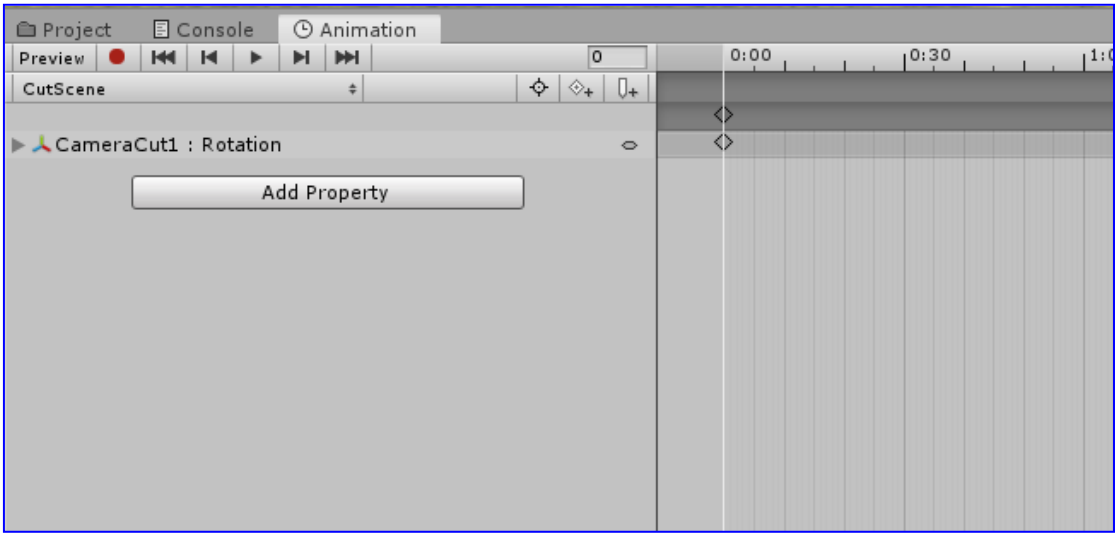


Εικόνα 41 Scripts στην κύρια κάμερα

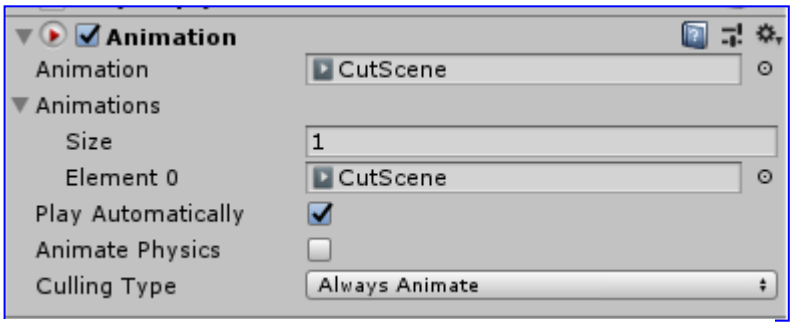
5.3.1 CutScene

Στην αρχή του προγράμματος έχω βάλει να αναπαράγεται ένα cut scene σε πραγματικό χρόνο το οποίο αποτελείται από τρις κάμερες και καταλήγει στην κύρια κάμερα.

Αυτές οι τρις κάμερες έχουν προγραμματιστεί να περιστρέφονται ορισμένες μοίρες σε συγκεκριμένες κατευθύνσεις. Όταν η πρώτη κάμερα τελειώσει την κίνηση της ενεργοποιείται η δεύτερη κάμερα και ούτω καθεξής. Αυτό επιτυγχάνεται με τη βοήθεια ενός script και της τεχνολογίας Animation(Εικόνα 37) της Unity, το οποίο δημιουργεί ένα Animation clip που πρέπει να αναθέσουμε στο Animation Component της ανάλογης κάμερας(Εικόνα 38).



Εικόνα 42 Animation Interface



Εικόνα 43 Animation Component με Animation Clip

Όπως βλέπουμε στην παρακάτω Εικόνα 39 έχουμε δημιουργήσει τέσσερα αντικείμενα για κάθε κάμερα που υπάρχει στο πρόγραμμα. Η κλάση Coroutine χρησιμοποιείτε όταν θέλουμε να κάνουμε παράλληλες ενέργειες ή σχεδόν παράλληλες όπως στην προκειμένη περίπτωση με τις κινήσεις των καμερών.

Δηλαδή μία Coroutine είναι σαν μια λειτουργία που έχει τη δυνατότητα να διακόψει την εκτέλεση και να επιστρέψει τον έλεγχο στη Unity, αλλά στη συνέχεια να συνεχίσει από εκεί που σταμάτησε.

Το IEnumerator είναι αυτό που επιστρέφει η Coroutine για αυτό και είναι αναγκαστικό να χρησιμοποιηθεί. Μέσα στο IEnumerator βλέπουμε ότι καθορίζεται ο χρόνος που περιμένει η Unity, με την εντολή 'return yield new WaitForSeconds()', ώστε να συνεχίσει τον κώδικα και να αλλάξει κάμερα. Η αλλαγή καμερών επιτυγχάνεται θέτοντας την καινούρια κάμερα True και την παλιά False.

```
void Start()
{
    StartCoroutine(TheSequence());

}

IEnumerator TheSequence()
{
    yield return new WaitForSeconds(6);
    cam2.SetActive(true);
    cam1.SetActive(false);
    yield return new WaitForSeconds(6);
    cam3.SetActive(true);
    cam2.SetActive(false);
    yield return new WaitForSeconds(6);
    cam4.SetActive(true);
    cam3.SetActive(false);
    /*yield return new WaitForSeconds(5.5f);
    cam5.SetActive(true);
    cam4.SetActive(false);*/
}
```

Εικόνα 44 Cut Scene script

5.3.2 Ελεύθερη και μπροστινή κάμερα

Έχει προστεθεί η δυνατότητα ο χρήστης να μπορεί να ελέγχει μία ελεύθερη κάμερα ώστε να μπορεί να δει την πανεπιστημιούπολη από όποια οπτική γωνία θέλει και όχι μόνο από την οπτική ενός ανθρώπου. Για αυτό το χαρακτηριστικό χρησιμοποίησα τη δωρεάν κάμερα Free Fly Camera από το Unity Store και μπορεί να ενεργοποιηθεί με το πλήκτρο E(εικόνα 40).

Επίσης ο χρήστης έχει τη δυνατότητα με το πλήκτρο R να ενεργοποιήσει την μπροστινή κάμερα του χαρακτήρα που ελέγχει.

Εικόνα 45 Ελεύθερη κάμερα την ώρα που εκτελείτε το πρόγραμμα

5.4 Trigger Boxes

Trigger boxes είναι αόρατα τετράγωνα τα οποία όταν ακουμπήσει ο παίχτης συμβαίνει μία ενέργεια ή μία σειρά ενεργειών που έχουμε προγραμματίσει.

Σε αυτό το πρόγραμμα έχω προγραμματίσει όταν ο παίχτης φτάνει κοντά σε κάποια κτήρια να εμφανίζονται στην οθόνη τα ονόματά τους. Για παράδειγμα όταν ο παίχτης φτάνει κοντά στο κτήριο διοίκησης, εμφανίζεται ένα κείμενο πάνω στη οθόνη που αναγράφει '' ΚΤΗΡΙΟ ΔΙΟΙΚΗΣΗΣ ''.

Για την επίτευξη αυτού χρησιμοποιήθηκε ένα Script αλλά και το Canvas System της Unity. Το Canvas είναι η περιοχή στην οποία πρέπει να βρίσκονται όλα τα στοιχεία διεπαφής χρήστη. Στην ουσία το Canvas είναι ένα αντικείμενο παιχνιδιού με ένα στοιχείο Canvas. Στο script όπως βλέπουμε στην παρακάτω εικόνα(Εικόνα 40) υπάρχει ένα gameobject αντικείμενο στο οποίο αναθέτουμε το κείμενο μας, που στη ουσία είναι ένα αντικείμενο και αυτό για την Unity. Μόλις ο παίχτης ακουμπήσει το TriggerBox εμφανίζεται στην οθόνη το μήνυμα με την εντολή uiObject.SetActive(true) και μετά από 5 δευτερόλεπτα καταστρέφεται με την εντολή Destroy(uiObject,5).

```
public class TriggerTexts : MonoBehaviour
{
    private const int Seconds = 5;
    public GameObject uiObject;

    void Start()
    {
        uiObject.SetActive(false);
    }

    private void OnTriggerEnter(Collider player)
    {
        if (player.gameObject.tag == "Player")
        {
            uiObject.SetActive(true);
            Destroy(uiObject, 5);
            Destroy(gameObject, 5);

        }
```

Εικόνα 46 Trigger Text script

5.4 Μενού

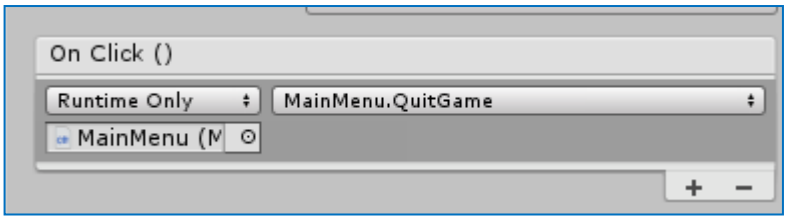
Υπάρχουν τρία διαφορετικά μενού στο πρόγραμμα. Το ένα είναι το αρχικό μενού και αποτελεί μία σκηνή μόνο του, και το άλλο είναι το μενού παύσης που υπάρχει στην κύρια σκηνή.

Το αρχικό μενού αποτελείτε από τέσσερα αντικείμενα Button(κουμπί) και ένα script(Εικόνα 41). Το κουμπί έναρξης, με το οποίο δίνεται η εντολή να ‘’φορτωθεί’’ η κύρια, το κουμπί Οδηγίες το οποίο σου εμφανίζει τα κουμπιά του προγράμματος στο νέο μενού , το κουμπί Πίσω το οποίο εμφανίζει το προηγούμενο μενού και το κουμπί Έξοδος το οποίο κλείνει το πρόγραμμα.

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

public class MainMenu : MonoBehaviour
{
    // Start is called before the first frame update
    public void PlayGame()
    {
        SceneManager.LoadScene(1);
    }

    public void QuitGame()
    {
        Debug.Log("Quit");
        Application.Quit();

    }
}
```

Εικόνα 47 Script μενού αρχικής σκηνής

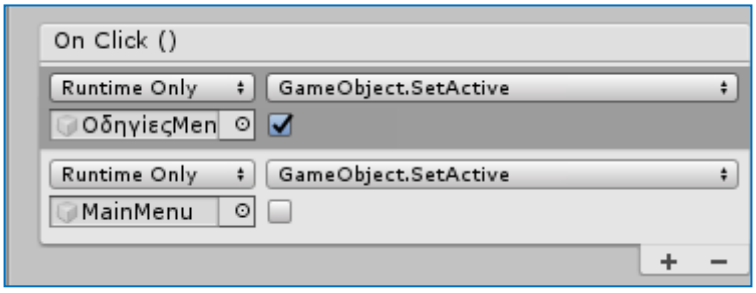
Ο τρόπος λειτουργίας είναι πολύ απλός. Αναθέτουμε το script σε ένα άδειο αντικείμενο μέσα στο Canva και μέσα σε αυτό το άδειο αντικείμενο έχουμε όλα τα αντικείμενα κειμένου και τα κουμπιά τους. Διαλέγουμε ένα κουμπί και στο Inspector διαλέγουμε την ενέργεια που θέλουμε να κάνει το κουμπί αυτό.



Εικόνα 48 Ρυθμίσεις κουμπιού Έξοδος στο Inspector

Για παράδειγμα στο κουμπί ‘’Έξοδος’’ βλέπουμε ότι έχουμε αναθέσει το MainMenu αντικείμενο στην On Click λειτουργία του. Επειδή στο αντικείμενο MainMenu έχουμε αναθέσει το script μας, μπορούμε να πάρουμε όποια λειτουργία έχουμε γράψει στο Script. Στην προκειμένη περίπτωση διαλέγουμε να εκτελεί την λειτουργία QuitGame, η οποία τερματίζει το πρόγραμμά μας. Το ίδιο συμβαίνει και με το κουμπί έναρξη, το οποίο φορτώνει την κύρια σκηνή.

Το κουμπί Οδηγίες λειτουργεί λίγο διαφορετικά. Με την πίεση του κουμπιού δεν καλούμε κάποια λειτουργία από το script, αλλά θέτουμε ανενεργό το κύριο μενού για να εξαφανιστούν τα κείμενά του από την οθόνη και θέτουμε ενεργό το μενού οδηγιών(Εικόνα 43). Το πίσω κουμπί κάνει την αντίθετη λειτουργία.



Εικόνα 49 Ρυθμίσεις κουμπιού ‘’Οδηγίες’’

Το μενού στη κύρια σκηνή αποτελείται από τρία κουμπιά. Το κουμπί ‘’Συνέχεια’’ που συνεχίζει το πρόγραμμα και κλείνει το μενού, το κουμπί ‘’Μενού’’ που σε πηγαίνει στο αρχικό μενού και το Κουμπί ‘’Έξοδος’’ που κλείνει το πρόγραμμα.

6. Συμπεράσματα

Η Unity είναι ένα ολοκληρωμένο πακέτο όπου δίνει την δυνατότητα στον Developer να φτιάξει το δικό του όραμα σε 2D ή 3D με σχετικά απλό τρόπο.

Η Unity είναι ένα από τα καλύτερα Game Engine όσον αφορά την δημιουργία παιχνιδιών σε Android και iOS. Με την τελευταία έκδοση της και την μεγάλη αλλαγή και την έμφαση που δίνει στο οπτικό τομέα αναμένεται να κυριαρχήσει κυρίως σε υπολογιστές αλλά και σε game consoles γιατί πολύ απλά είναι πιο απλό και εύχρηστο για έναν αρχάριο αλλά και έναν απαιτητικό Developer.

Το βασικό χαρακτηριστικό που κάνει την Unity πιο προσιτή είναι οι βασικοί οδηγοί, τα πολλά tutorials που υπάρχουν στο διαδίκτυο, αλλά και το Community Forum όπου εύκολα και γρήγορα μπορούν να σε βοηθήσουν.

Τέλος, η δωρεάν έκδοση αλλά και οι πολλές πλατφόρμες που υποστηρίζει κάνει την μηχανή νούμερο ένα στις προτιμήσεις των μικρών εταιριών ανάπτυξης σε μια βιομηχανία παιχνιδιών που συνεχώς αυξάνεται.

Βιβλιογραφεία

1. https://unity.com/
2. https://forum.unity.com/
3. https://learn.unity.com/tutorials
4. https://en.wikipedia.org/wiki/Unity_Technologies
5. https://web.wpi.edu/Pubs/E-project/Available/E-project-030614-143124/unrestricted/Haas_IQP_Final.pdf
6. https://whatis.techtarget.com/definition/script