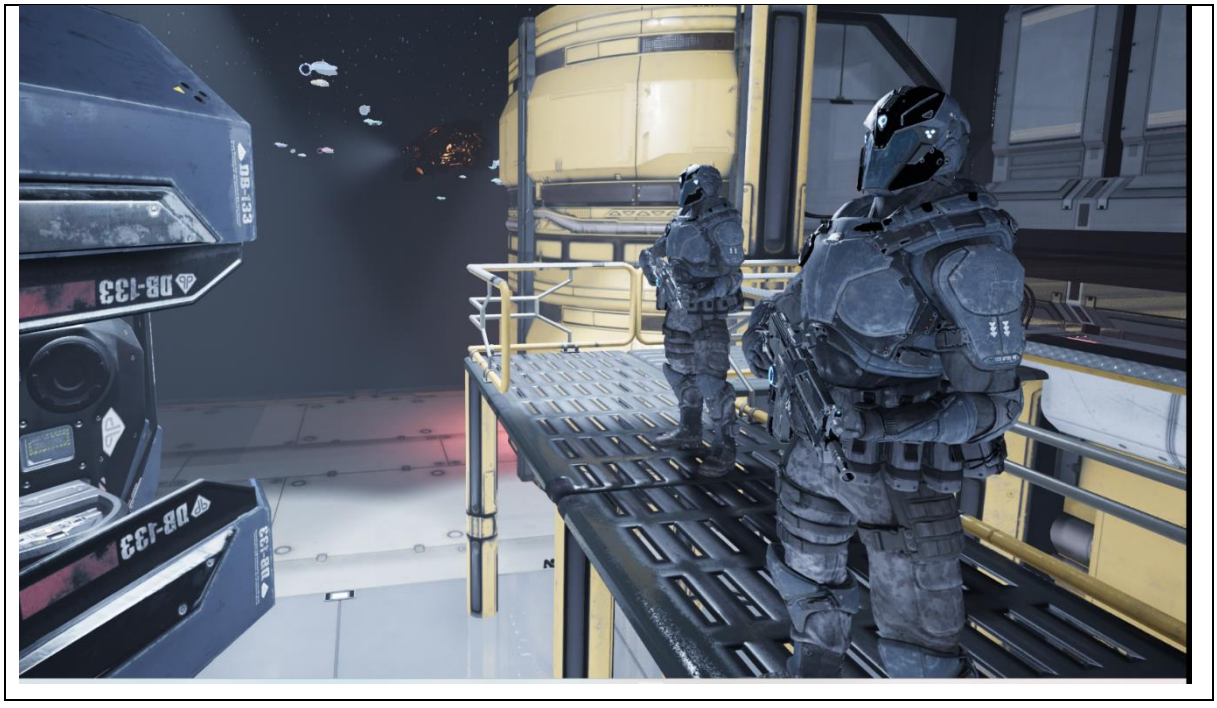
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΠΤΥΞΗ ΒΙΝΤΕΟΠΑΙΧΝΙΔΙΟΥ ΔΡΑΣΗΣ ΠΡΩΤΟΥ ΠΡΟΣΩΠΟΥ ΜΕ UNREAL ENGINE 4 ΚΑΙ C++



Του φοιτητή
Σπυριδόπουλου Κωνσταντίνου
Αρ. Μητρώου: 144189

Επιβλέπων
Ονοματεπώνυμο Δεληγιάννης
Ιγνάτιος
Βαθμίδα Καθηγητής

Ημερομηνία 13/03/2023

Τίτλος Π.Ε. Ανάπτυξη βιντεοπαιχνιδιού δράσης πρώτου προσώπου με Unreal Engine 4 και C++

Κωδικός Π.Ε. 21211

Ονοματεπώνυμο φοιτητή/των Σπυριδόπουλος Κωνσταντίνος

Ονοματεπώνυμο εισηγητή Δεληγιάννης Ιγνάτιος

Ημερομηνία ανάληψης Δ.Ε. 24/03/2021

Ημερομηνία περάτωσης Δ.Ε. 13/03/2023

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Πρόλογος

Η εργασία αυτή αποτέλεσε για εμένα τον συγγραφέα μια πρόκληση καθώς αποσκοπούσα στο να δημιουργήσω ένα μικρό ολοκληρωμένο έργο σε μια μηχανή που απευθύνεται σε ομάδες και προγραμματιστές με αρκετή εμπειρία στην γλώσσα προγραμματισμού C++. Επιπλέον ήταν μια πολύ εποικοδομητική εμπειρία καθώς απορρόφησα αρκετή γνώση για το πως λειτουργούν εσωτερικά τα διάφορα εμπορικά βιντεοπαιχνίδια και κατανόησα σε μεγαλύτερο βαθμό τους κύκλους εκτέλεσης ενός προγράμματος.

Περίληψη

Στα πλαίσια αυτής της πτυχιακής εργασίας ερευνάται η ανάπτυξη ενός βιντεοπαιχνιδιού με τη χρήση της μηχανής Unreal Engine 4. Το έργο μετατρέπεται από μια απλή ιδέα σε τελικό εκτελέσιμο αρχείο. Αναλύεται βήμα προς βήμα η συνήθης μεθοδολογία ανάπτυξης τέτοιου μήκους έργων και παρουσιάζονται οι απαραίτητοι πόροι, σε ανθρώπινο δυναμικό, ειδικότητες και υλικά, για να καταστεί το έργο κατάλληλο για εμπορική χρήση.

Development of an action first person shooter videogame using Unreal Engine 4 and C++

Spiridopoulos Constantinos

Abstract

This thesis constitutes a research on developing an Unreal Engine 4 videogame. The process of an idea becoming a fully-fledged executable is presented step by step, taking into account any human resources, specialties and technical equipment needed to reach the goal of sharing the project as a product to the market.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον καθηγητή μου που μου έδωσε την ευκαιρία να αναλάβω και να παρουσιάσω αυτό το έργο ενώπιον του τμήματος. Επίσης, την οικογένεια και τους φίλους μου που με ενθάρρυναν να συνεχίσω το έργο, όσο μεγάλη και αν ήταν η πρόκληση. Τέλος, θέλω να ευχαριστήσω την Epic Games που αποφάσισε να δώσει την μηχανή Unreal Engine σε ελεύθερη χρήση και μαζί όλους τους δημιουργούς που έδωσαν τα πακέτα υλικών ελεύθερα στο κοινό για χρήση σε προσωπικά και εμπορικά έργα.

Περιεχόμενα

Κατάλογος Σχημάτων και Εικόνων

Κατάλογος Πινάκων

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
Π.Ε.	Πτυχιακή Εργασία
UI	User Interface
RAM	Random Access Memory
PvP	Person versus Person
PvE	Person versus Computer
DLC	Downloadable Content
HDD	Hard Disk Drive
SSD	Solid State Drive
UE	Unreal Engine
NavMesh	Navigation Mesh
UMG	Unreal Motion Graphics
HUD	Heads Up Display
FHD	Full High Definition

Κεφάλαιο 1ο: Το βιντεοπαιχνίδι ως προϊόν

1.1 Εισαγωγή

Τα βιντεοπαιχνίδια έχουν εισέλθει δυναμικά στο σύγχρονο εμπόριο και δεν είναι λίγες οι αναφορές που γίνονται σε αυτά. Σήμερα, εύκολα συναντούμε ποικίλα δημοσιογραφικά και ερευνητικά άρθρα με θέμα τα βιντεοπαιχνίδια και την πορεία τους στο χρονικό διάστημα κατά το οποίο βρίσκονται ενεργά στην αγορά. Παρατηρούμε τις πολλαπλές εμφανίσεις τους σε περιοδικά με αφιερώσεις σε εμπορικές επιτυχίες που σημείωσαν τίτλοι. Επιπλέον, διεξάγονται εκδηλώσεις απονομής τίτλων και βραβείων στα καλύτερα βιντεοπαιχνίδια της χρονιάς με κατατάξεις καθώς και παγκόσμια τουρνουά με χρηματικό έπαθλο. Είναι ευρέως διαδεδομένα κατά τα φαινόμενα και αποτελούν ένα μέσο ψυχαγωγίας και εκμάθησης για άτομα όλων των ηλικιακών ομάδων ενώ ταυτόχρονα αποτελούν σημείο ενδιαφέροντος στον τομέα του εμπορίου, φυσικού ή ψηφιακού και συμβάλλουν στην ανάπτυξη του οικονομικού χαρακτήρα των νέων τεχνολογιών [1].

1.2 Ο ορισμός του βιντεοπαιχνιδιού

Καθώς η παρούσα εργασία αποτελεί ένα βιντεοπαιχνίδι, η προσεγγιστική επεξήγηση του όρου αυτού θα διευκολύνει την πορεία εξέλιξης του από μια ιδέα σε ένα δομημένο έργο. Τι σημαίνει λοιπόν ο όρος βιντεοπαιχνίδι; Θα μπορούσαμε να περιγράψουμε ένα βιντεοπαιχνίδι ως '*μια διαδραστική εφαρμογή την οποία χρησιμοποιούμε με την βοήθεια ενός ψηφιακού παίκτη*'[2] όπου σκοπός της εφαρμογής είναι η ψυχαγωγία του χρήστη. Ακόμα, είναι ένα πρόγραμμα υπολογιστή με σαφώς καθορισμένο ή μη τρόπο συμπεριφοράς, ανάλογα με τις εισόδους του χρήστη και περιλαμβάνει απαραίτητα γραφικό περιβάλλον (Graphics Interface) και διεπαφή χρήστη (User Interface) [2]. Μπορούν να αποδοθούν πολλές διαφορετικές έννοιες στον όρο, χάρη στην πολυμορφικότητα που έχουν αποκτήσει τα βιντεοπαιχνίδια ανά τα χρόνια. Η λέξη ετυμολογικά είναι σύνθετη και περιλαμβάνει τους όρους βίντεο και παιχνίδι.

Βίντεο είναι μια στατικά αποθηκευμένη ακολουθία από στιγμιότυπα εικόνας και ήχου που δίνουν την αίσθηση της κίνησης, αλλιώς της δράσης. Η έκβαση των γεγονότων που θα συμβούν σε ένα βίντεο είναι καθορισμένη με αρχή και τέλος. Ένα βίντεο είναι επίσης ένα εκτελέσιμο αρχείο γιατί χρειάζεται ένα υπολογιστικό σύστημα να το φορτώσει σε μια μη πτητική μνήμη (για παράδειγμα RAM) και να το αποκωδικοποιήσει κατάλληλα για να αναπαραγάγει το περιεχόμενο σε ένα μέσο θέασης, μια οθόνη. Όταν παρακολουθούμε ένα βίντεο, πιο συγκεκριμένα ένα βίντεο εκπαιδευτικού ή ψυχαγωγικού περιεχομένου, είμαστε ενεργοί θεατές αυτού που προβάλλεται στο βίντεο από την άποψη ότι μπορούμε να σκεφτούμε για το τι θα γίνει παρακάτω, να μάθουμε κάτι νέο, να κατανοήσουμε το περιεχόμενο καλύτερα ή να μας κατακλύσουν συγκεκριμένα συναισθήματα ανάλογα με τα γεγονότα που διαδραματίζονται στο βίντεο. Χαρακτηριστικό παράδειγμα εκπαιδευτικού περιεχομένου είναι τα ντοκιμαντέρ, τα βίντεο-μαθήματα και τα διάφορα βίντεο καθοδηγητικού σκοπού. Από αυτά, όπως προδίδει η κατηγορία τους έχουμε την δυνατότητα να απορροφήσουμε γνώση για διάφορες θεματικές ενότητες. Από την άλλη πλευρά, βίντεο ψυχαγωγικού περιεχομένου είναι οι ταινίες και σειρές που χωρίζονται σε διάφορα είδη ανάλογα με το περιεχόμενο τους. Καθώς παρακολουθούμε μια ταινία, αναγνωρίζουμε τους χαρακτήρες, σκεφτόμαστε και αναλύουμε την πλοκή της ιστορίας και σε συγκεκριμένες περιπτώσεις μπορούμε να προβλέψουμε την έκβαση της ιστορίας από τα όσα δεδομένα μας δίνονται. Ακόμα μπορούμε να ταυτιστούμε με κάποιον χαρακτήρα, να ενστερνιστούμε τις ατάκες του, να αποδώσουμε νόημα στις πράξεις του. Η αλληλοεπίδραση του θεατή με ένα βίντεο εξακολουθεί να είναι περιορισμένη. Μπορούμε άλλωστε να το διακρίνουμε στα ανωτέρω παραδείγματα. Βέβαια, με τα κατάλληλα μέσα μπορεί να αποτελέσει μια σπουδαία εμπειρία.

Παιχνίδι είναι μια διαδικασία που απαρτίζεται από κανόνες, έχει σημείο αφετηρίας, σημείο τερματισμού, συνθήκες επιτυχίας/αποτυχίας και συνήθως κάποιο σκοπό. Συχνά τα παιχνίδια περιλαμβάνουν κάποιο είδος βαθμολογίας και κατάταξης που μπορεί να συμμετέχει στις συνθήκες ανάδειξης νικητή. Χωρίζονται εξίσου σε κατηγορίες ανάλογα με το είδος τους, το τρόπο με τον οποίο παίζονται ή το περιεχόμενο τους. Φυσικά, ως επί το πλείστων είναι ψυχαγωγικά. Μπορούν να παίζονται με περισσότερα από δύο άτομα. Συνήθως υπάρχει ανταγωνισμός μεταξύ παικτών ή ομάδων παικτών με σκοπό τη κατάκτηση του στόχου, όπως να κερδίσουν την αντίπαλη ομάδα μαζεύοντας περισσότερους πόντους ή οποιασδήποτε συνθήκης που αποδίδει την νίκη σε ένα παίκτη ή ομάδα. Πάντα, στα πλαίσια που ορίζουν οι κανόνες του παιχνιδιού. Και αυτό, διότι οι κανόνες αποτελούν αναπόσπαστο κομμάτι ενός παιχνιδιού. Αυτοί μπορεί να είναι απλοί ή πολύπλοκοι, σε κάθε περίπτωση όμως διαμορφώνουν το ύφος του παιχνιδιού και το καθιστούν περισσότερο ελκυστικό προς κάποιες ομάδες ατόμων [3]. Τέλος, όπως και στο βίντεο, έτσι και στο παιχνίδι ισχύει η έννοια της θέασης όπου θεατές βλέπουν τους παίκτες να παίζουν ώστε να ψυχαγωγηθούν ή να μάθουν καλύτερα στην πράξη πως παίζεται ένα συγκεκριμένο παιχνίδι. Ομοίως με τη θέαση ενός βίντεο, δημιουργούνται συγκεκριμένα συναισθήματα στους παίκτες και τους θεατές αναλόγως με την έκβαση ενός παιχνιδιού ή τον βαθμό ανταγωνισμού που υπάρχει. Ο κάθε παίκτης ή ομάδα προσπαθεί να κατανοήσει τον αντίπαλο, να αντιδράσει κατάλληλα ή να καταστρώσει μια στρατηγική που θα του δώσει την νίκη. Όταν αναφερόμαστε σε ένα παιχνίδι, συχνά η πρώτη εικόνα που σχηματίζουμε προέρχεται από τον τρόπο με τον οποίο παίζεται το παιχνίδι και αυτό καθορίζει την τοποθέτηση του παιχνιδιού σε ένα συγκεκριμένο είδος. Αναλόγως, το περιεχόμενο ενός βίντεο είναι υπεύθυνο για την κατηγοριοποίηση του.

Οι δύο έννοιες συγκλίνουν σε κάποια σημεία και φαινομενικά δεν απέχουν πολύ μεταξύ τους. Στην πραγματικότητα έχουν μια πολύ μεγάλη διαφορά. Το βίντεο είναι στατικό, προκαθορισμένο από την αρχή έως το τέλος του. Το παιχνίδι έχει προκαθορισμένους κανόνες και ένα ή περισσότερους σκοπούς, συνθήκες επιτυχίας και αποτυχίας, αλλά η έκβαση και το τέλος του καθορίζεται σχεδόν πάντοτε από τις ενέργειες του παίκτη [2]. Κάθε γύρα του ίδιου παιχνιδιού μπορεί να διαφέρει πλήρως από τις υπόλοιπες, παρόλο που βασίζεται σε σταθερούς κανόνες. Υπάρχει η έννοια της πιθανότητας σε κάθε κίνηση και απόφαση οπότε η εξέλιξη του είναι δυναμική και όμοια, το τέλος του.

Η σύνθετη λέξη λοιπόν *βιντεοπαιχνίδι* συνδυάζει στοιχεία στατικού και δυναμικού περιεχομένου, προκαλεί διάφορα συναισθήματα στον θεατή-παίκτη και τον ωθεί στο να σκεφτεί για την πλοκή (εάν υπάρχει), να καταστρώσει στρατηγικές και να αποφασίσει για τις κινήσεις που θα τον οδηγήσουν στην νίκη. Ο ρόλος του παίκτη δεν είναι πάντα σταθερός. Πολλές φορές μετατρέπεται σε θεατή-παρατηρητή, διότι τα βιντεοπαιχνίδια συμπεριλαμβάνουν κινηματογραφικό περιεχόμενο, αποκόμματα από ταινίες και άλλοτε σκηνές φτιαγμένες με CGI (Computer Generated Image), σκηνές όπου τα σκηνικά και οι χαρακτήρες είναι τρισδιάστατα μοντέλα και όλο το περιεχόμενο είναι φτιαγμένο και κινηματογραφημένο ψηφιακά [4]. Άλλες φορές ο παίκτης θεατής αλλάζει από απλός παθητικός δέκτης σε ενεργό θεατή γιατί τα τελευταία χρόνια οι ψηφιακές σκηνές εμπεριέχουν διαδραστικά στοιχεία [2]. Για παράδειγμα, ο παίκτης καλείται να κάνει κάποια ενέργεια που θα αλλάξει τη ροή των γεγονότων μέσα στο βίντεο.

Τι είναι όμως τελικά ένα βιντεοπαιχνίδι; Αρχικά είναι ένα πρόγραμμα. Ένα εκτελέσιμο αρχείο που αναπαράγεται σε μια ηλεκτρονική συσκευή για την οποία κατασκευάστηκε. Έχει προκαθορισμένους κανόνες, εντολές και μεθόδους αναπαραγωγής περιεχομένου και συνδυάζει κείμενο, εικόνα, ήχο και κίνηση [2], [4]. Ένα φυσικό παιχνίδι παίζεται από άτομα που, ανάλογα με το είδος του παιχνιδιού, έχουν στη κατοχή τους αντικείμενα που τους προσδίδεται αξία για τις ανάγκες του παιχνιδιού. Μια μπάλα, μια τράπουλα, πιόνια, πούλια, ζάρια. Αυτά τα αντικείμενα μπορεί να αποτελούν μέσα για την επίτευξη του στόχου του παιχνιδιού και συνεπώς την ανάδειξη ενός νικητή ή μπορεί να είναι απλά βοηθητικά

για την εξέλιξη του. Στα βιντεοπαιχνίδια, το βοηθητικό μέσο του παίκτη είναι το χειριστήριο που ελέγχει το ψηφιακό χαρακτήρα του παίκτη, το avatar του, το οποίο συναντάμε σε διάφορες μορφές. Σαν δύο μοχλούς και κουμπιά τοποθετημένα πάνω σε ένα μηχάνημα με οθόνη όπως τα μηχανήματα Arcade. Ως συσκευή με κουμπιά που συνδέεται ενσύρματα ή ασύρματα σε μια κονσόλα. Ως ψηφιακά κουμπιά επάνω σε μια οθόνη αφής [2]. Εκτός από το μέσο ελέγχου, το βιντεοπαιχνίδι διαθέτει συνήθως ψηφιακά αντικείμενα που βοηθούν στη πρόοδο του παίκτη. Σε βιντεοπαιχνίδια που προσομοιώνουν τα γνωστά επιτραπέζια παιχνίδια, τα ψηφιακά αντικείμενα δεν είναι άλλα παρά αυτά που θα συναντήσουμε και στα πραγματικά επιτραπέζια. Με παρόμοιο τρόπο στα παιχνίδια που προσομοιώνουν αθλήματα θα υπάρχει μια μπάλα, μια ρακέτα ή ένα μπαστούνι. Σε άλλες κατηγορίες παιχνιδιών που ξεφεύγουν από τη πραγματικότητα, τα αντικείμενα μπορεί να είναι κλειδιά, οπλισμός, ρουχισμός, μεταφορικό μέσο, ηλεκτρονικές συσκευές και άλλα πολλά που συναντάμε στον πραγματικό κόσμο. Η συμπεριφορά τους και η αξία τους βέβαια δεν θα είναι ακριβώς η ίδια με την πραγματικότητα καθώς όλα τα αντικείμενα στον ψηφιακό κόσμο λειτουργούν με τους κανόνες του παιχνιδιού και εξυπηρετούν την ψυχαγωγία του παίκτη ή την ανάκαμψη της προόδου του [2], [4].

1.3 Κατηγορίες βιντεοπαιχνιδιών

Τα βιντεοπαιχνίδια χωρίζονται σε δύο μεγάλες κατηγορίες σύμφωνα με το πως προβάλλεται το περιεχόμενο τους στην εικόνα μας. Τα δισδιάστατα, όπου έχουμε δύο άξονες κίνησης X και Y, μήκος και ύψος και τα τρισδιάστατα όπου χρησιμοποιούνται τρεις άξονες κίνησης, X,Y,Z, πλάτος, μήκος και ύψος αντίστοιχα. Στα δισδιάστατα βιντεοπαιχνίδια τα ψηφιακά αντικείμενα στο χώρο προβάλλονται από μια μόνο πλευρά, συνήθως από το πλάι με σημείο αναφοράς τον παίκτη ενώ στα τρισδιάστατα προβάλλονται από όλες τις πλευρές ανάλογα με τη θέση της ψηφιακής κάμερας. Από εκεί και πέρα, ανάλογα με το σκοπό του βιντεοπαιχνιδιού και τον τρόπο με τον οποίο παίζουμε προσπάθησαν διάφοροι αναλυτές να τα κατηγοριοποιήσουν. Στο παρόν, έχουμε πολλές κατηγορίες και υποκατηγορίες καθώς και εξαιρέσεις ή μίξη από κατηγορίες για να περιγράψουμε και να κατηγοριοποιήσουμε τα βιντεοπαιχνίδια [1], [3], [5]. Σπάνια ένα βιντεοπαιχνίδι στις μέρες μας ανήκει καθαρά σε μια κατηγορία ή παίζεται με έναν συγκεκριμένο τρόπο. Ακόμα, οι αρθρογράφοι που μελετούν και παρουσιάζουν διάφορα στοιχεία για τα δημοφιλή βιντεοπαιχνίδια που κυκλοφορούν στην αγορά σκαρφίζονται ορολογίες που περιγράφουν πιο σύντομα μια κατηγορία ή ένα είδος, τρόπο με τον οποίο παίζονται συγκεκριμένα από αυτά. Και αυτό λόγω του περιορισμού λέξεων του κάθε άρθρου ή του περιορισμένου χρόνου που τους δίνεται να μιλήσουν για το κάθε τι καινούργιο που θα δούμε ή που κυκλοφορεί ήδη στην αγορά. Οπότε δεν είναι τίποτα σίγουρο όταν τοποθετούμε ένα βιντεοπαιχνίδι σε κάποια κατηγορία. Η επιλογή μπορεί να είναι προϊόν αναλυτικής μελέτης, οικονομίας χρόνου ή χώρου όπως είδαμε αλλά και θέμα μάρκετινγκ. Πάνω σε αυτό, χαρακτηριστικό παράδειγμα είναι όταν μια κατηγορία επιλέγεται από την κατασκευάστρια εταιρεία γιατί φαίνεται να έχει περισσότερη ζήτηση ή εκτενείς αναλύσεις δείχνουν πως γίνονται πιο εύκολα εμπορικές επιτυχίες τα προϊόντα που ακολουθούν τα στάνταρ της συγκεκριμένης κατηγορίας, πχ τα παιχνίδια ρόλων με ανοικτό κόσμο που έχουν γίνει πολύ δημοφιλή από το 2020 μέχρι και σήμερα [6]. Σχεδόν κάθε καινούργιος τίτλος φαίνεται να προσπαθεί να υιοθετήσει τα στοιχεία ανοικτού κόσμου, ακόμα και αν δεν ταιριάζει στο κλίμα και την αισθητική του παιχνιδιού. Με μικρή ή μεγάλη επιτυχία διότι δεν υπάρχει κάποιος σταθερός κανόνας που να δίνει την επιτυχία του τελικού βιντεοπαιχνιδιού στην αγορά. Και επιτυχία σε πωλήσεις δεν συνεπάγεται μεγάλο βαθμό ικανοποίησης του κοινού.

1.3.1 Παιχνίδια βολών πρώτου προσώπου

Στα πλαίσια της παρούσας εργασίας θα παρουσιαστούν και θα αναλυθούν τα χαρακτηριστικά της κατηγορίας τρισδιάστατων βιντεοπαιχνιδιών «*παιχνίδια βολών πρώτου προσώπου*» ή αλλιώς first person shooter. Είναι σύνθετη κατηγορία και περιλαμβάνει δύο βασικά είδη, τα παιχνίδια βολών και τα παιχνίδια πρώτου προσώπου. Είναι πολύ συνηθισμένο να συναντούμε αυτές τις κατηγορίες μαζί γιατί σερβίρουν στον παίκτη μια εμπειρία δράσης από πρώτο χέρι ή πρόσωπο ακόμα καλύτερα. Όταν οι δύο έννοιες είναι μαζί, το βιντεοπαιχνίδι είναι αποκλειστικά τρισδιάστατο με κίνηση στους τρεις άξονες του χώρου, λόγω της τοποθέτησης της κάμερας και της ανάγκης για κίνηση σε τρισδιάστατο χώρο [6].

Τα παιχνίδια πρώτου προσώπου περιλαμβάνουν όλα τα παιχνίδια με προοπτική κάμερας μέσα από τα μάτια του χαρακτήρα. Βλέπουμε στην οθόνη μας ότι θα έβλεπε ο χαρακτήρας από τον ψηφιακό κόσμο του παιχνιδιού, συνήθως τα χέρια του ή ολόκληρο το σώμα πλην του προσώπου, το οποίο μπορούμε να δούμε σε κάποια αντανάκλαση και αν αλλάξει προοπτική η κάμερα. Θεωρείται πρώτο πρόσωπο γιατί οι υπόλοιποι χαρακτήρες, ηθοποιοί (actors, non playable characters) όπως ονομάζονται στη βιομηχανία βιντεοπαιχνιδιών και οι αντίπαλοι (mobs), απευθύνονται ευθέως στον δικό μας πρωταγωνιστή σαν να μιλούν σε εμάς τους παίκτες. Αυτή η προοπτική αποσκοπεί στο να δώσει την ψευδαίσθηση ότι πρωταγωνιστούμε εμείς οι ίδιοι στο βιντεοπαιχνίδι. Ο πρωταγωνιστής είναι συνήθως κάποιος πολεμιστής που κατέχει γνώσεις για διάφορα όπλα με μεγάλη εμβέλεια. Συχνά κουβαλάει πολλά είδη οπλισμού επάνω του και κάποιου είδους προστατευτικής εξάρτησης. Ο παίκτης καλείται να αντιμετωπίσει τον υπολογιστή (PvE) ή άλλους παίκτες (PvP) σε περιβάλλοντα στρατιωτικού χαρακτήρα. Στρατόπεδα, χαρακώματα, φρούρια, διαστημικές εγκαταστάσεις. Οπουδήποτε θα μπορούσε να διεξαχθεί πόλεμος [6], [7].

Τα παιχνίδια *βολών πρώτου προσώπου* λοιπόν συνδυάζουν τον πόλεμο με μια προοπτική μέσα από τα μάτια του πρωταγωνιστή. Ωθούν τον παίκτη να συμπεριφερθεί όπως ένας αληθινός άνθρωπος σε εμπόλεμη ζώνη. Η οπτική συνδυάζεται με το τρόπο χειρισμού του παίκτη και προσφέρει μια ολοκληρωμένη εμπειρία πολέμου από την ασφάλεια του σπιτιού του καθενός. Σχεδόν σε όλα τα βιντεοπαιχνίδια αυτού του τύπου, βλέπουμε τα χέρια του ψηφιακού μας χαρακτήρα και το όπλο που κρατάει. Σε πιο πρόσφατους τίτλους παρατηρείται μια βελτίωση στο θέμα αισθητικής καθώς μπορούμε πλέον να δούμε όλο το σώμα του χαρακτήρα, τμήματα της εξάρτησης του και τον τρόπο με τον οποίο αλληλοεπιδρά πιο φυσικά με το περιβάλλον γύρω του. Αυτά φυσικά είναι λεπτομέρειες μπροστά στο τι διέπει αυτήν την κατηγορία βιντεοπαιχνιδιών. Τα κύρια χαρακτηριστικά τους είναι η περιήγηση και η αντιμετώπιση εχθρών από ασφαλή κάλυψη καθώς ο παίκτης προσπαθεί να διαχειριστεί σωστά τους διαθέσιμους πόρους του, πυρομαχικά, κουτιά πρώτων βοηθειών και οπλισμό [1], [6], [7]. Σκοπός του παιχνιδιού είναι ο πρωταγωνιστής να φέρει εις πέρας τις αποστολές που του αναθέτονται. Να καταλάβει μια αντίπαλη βάση? Να μπει κρυφά σε ένα στρατόπεδο και να σώσει αιχμαλώτους πολέμου? Να καθαρίσει μια περιοχή ώστε να κερδίσει έδαφος η παράταξη του? Αυτά και πολλά άλλα στοιχεία θα συναντήσει κανείς σε τέτοιου είδους βιντεοπαιχνίδια. Βέβαια υπάρχουν και εξαιρέσεις, κάποια βασίζονται σε αυτό το είδος δράσης αλλά λιγοστεύοντας τα πυρομαχικά ή κάνοντας τον ήρωα πιο ευάλωτο ή τους εχθρούς πιο έμπειρους, το βιντεοπαιχνίδι έχει πιο έντονο το αίσθημα της επιβίωσης και του κινδύνου. Με το να είναι ο χαρακτήρας μας ένας ανώνυμος σε ένα πεδίο μάχης, μεγαλώνει η κλίμακα δράσης του. Πολλές τεχνικές μπορούν να δώσουν άλλη τροπή, άλλο ύφος στο τελικό προϊόν και να το κάνουν να διαφέρει πολύ από τα υπόλοιπα της ίδιας κατηγορίας. Ο τρόπος με τον όποιο αλληλοεπιδρούν οι υπόλοιποι χαρακτήρες με τον πρωταγωνιστή δίνει την αίσθηση ύπαρξης του ως οντότητα.

1.4 Ανάπτυξη βιντεοπαιχνιδιού – Στάδια παραγωγής

Όταν αναφερόμαστε σε ένα βιντεοπαιχνίδι, συχνά έχουμε σαν εικόνα κάποιον χαρακτήρα, ένα τοπίο, μια σκηνή δράσης, μια ατάκα ενός χαρακτήρα, τον τρόπο με τον οποίο παίζεται το βιντεοπαιχνίδι ή ακόμα και την μουσική του αν είναι πολύ χαρακτηριστική. Κοινώς, το τελικό αποτέλεσμα που αποτελεί ένα ολοκληρωμένο προϊόν, έχει κυκλοφορήσει στην αγορά και έφτασε στα χέρια μας σε ψηφιακή ή φυσική μορφή με κάποιον τρόπο έτσι ώστε να βιώσουμε την εμπειρία που προσφέρει. Πριν όμως να έρθει σε αυτή τη μορφή, το βιντεοπαιχνίδι διανύει τέσσερα βασικά στάδια παραγωγής [9]. Αυτό ισχύει τόσο για προϊόντα από μικρές ομάδες ανάπτυξης και ερασιτέχνες, όσο και μεγάλες εταιρείες. Συγκεκριμένα για το προϊόν της παρούσας εργασίας έγινε εφαρμογή των σταδίων αυτών με μερικές αλλαγές ή απλοποιήσεις όπου κρίθηκε απαραίτητο, μιας και αποτελεί προϊόν για ερευνητικούς σκοπούς και μόνο.

1.4.1 Σύλληψη της ιδέας (Concept)

Το πρώτο στάδιο παραγωγής είναι **η σύλληψη της ιδέας** [9]. Μια ιδέα που θα υλοποιηθεί από διάφορα τεχνικά μέσα, εργαλεία, ανθρώπινο δυναμικό και θα καταστεί ολοκληρωμένο προϊόν. Ανάλογα πάντα με τον τρόπο διοίκησης μιας εταιρείας, θα πρέπει να συζητηθεί η ιδέα και να γίνει αποδεκτή από όλα τα μέλη που αποτελούν την ομάδα λήψης αποφάσεων. Να περάσει από κριτική και να βαθμολογηθεί σύμφωνα με το πόσο βιώσιμη είναι στην αγορά, πόσο αποδεκτή θα γίνει στο κοινό, στους καταναλωτές. Να ανατεθεί σαν έργο και να αρχίσει η ανάπτυξη της μέχρι το σημείο που θα υλοποιηθεί πλήρως. Αυτή η ιδέα μπορεί να είναι από κάτι συνηθισμένο όπως ένα βιντεοπαιχνίδι προσομοίωσης αγώνων ταχύτητας μέχρι κάτι πολύ σύνθετο και καινοτόμο. Δεν υπάρχουν όρια στις ιδέες. Σε μια πιο μικρή ομάδα ανάπτυξης συνήθως η ιδέα γεννιέται και ωριμάζει μέσα από τη συζήτηση επί του θέματος από όλα τα μέλη της ομάδας ανεξαρτήτως ειδικότητας ενώ σε σύντομης διάρκειας έργα που έχουν στόχο να αναδείξουν μια λειτουργία η ιδέα αποτελεί ατομική ευθύνη του σχεδιαστή – προγραμματιστή [10].

Η ιδέα δεν στοχεύει πάντοτε στην καινοτομία και την αλλαγή, μπορεί να είναι απόρροια της απόφασης να συνεχιστεί η ιστορία ενός βιντεοπαιχνιδιού ως δεύτερο μέρος (sequel) ή επιπρόσθετο περιεχόμενο για το υπάρχον παιχνίδι (DLC) [10]. Μπορεί να προέρχεται από μελέτη των τάσεων στα είδη βιντεοπαιχνιδιού που προτιμώνται κατά ένα διάστημα στο οποίο προβλέπεται ότι θα κυκλοφορήσει. Μπορεί να είναι καθαρά για επικερδές σκοπό. Συνέπεια αυτού είναι να βλέπουμε πολλά παλαιά παιχνίδια να αναβιώνουν μέσω της ανακατασκευής τους για πιο σύγχρονη τεχνολογία ή να κυκλοφορούν πολλά της ίδιας σειράς.

1.4.2 Προεργασία (Pre-Production)

Αφού πλέον η ιδέα είναι ενεργή και έχει αποφασιστεί η γενική πορεία, αποτελεί πλέον ένα έργο σε εξέλιξη. Αναλαμβάνει δράση το τεχνικό τμήμα, δηλαδή το ανθρώπινο δυναμικό που έχει στην διάθεση του εργαλεία και τεχνική γνώση σε ποικίλα θέματα που αφορούν την ανάπτυξη ενός σχεδόν οποιουδήποτε βιντεοπαιχνιδιού, για να κάνει μια προεργασία. Η προεργασία περιλαμβάνει διαχωρισμό του έργου σε επιμέρους τομείς και προβλήματα και προϋπολογισμό ώστε να φανερωθεί στην ομάδα ανάπτυξης, αν είναι δυνατόν, το ακριβές σύνολο από υλικά που θα χρειαστούν για την κατασκευή του έργου [11]. Όπως ένας ζωγράφος θα φανταστεί την εικόνα που θέλει να ζωγραφίσει και θα βρει τα κατάλληλα πινέλα και χρώματα για να την αποτυπώσει, έτσι και το τεχνικό τμήμα θα οπτικοποιήσει την ιδέα και θα συγγραφεί συλλογικά από όλη την ομάδα ένα έντυπο σχεδίασης. Το έντυπο θα περιλαμβάνει περιγραφικά τους στόχους του βιντεοπαιχνιδιού, τις απαιτήσεις σε σκηνικά και χαρακτήρες, το σενάριο, τις μηχανικές, το στυλ παιχνιδιού, δηλαδή τον ή τους τρόπους με τους οποίους

μπορεί να παίξει ο παίκτης και τέλος το συνολικό αποτέλεσμα που αναμένεται να κατασκευαστεί [10]. Αν το βιντεοπαιχνίδι περιλαμβάνει ιστορία, θα πρέπει να κατασκευαστεί κατά την προεργασία έτσι ώστε ο προγραμματισμός και τα σκηνικά να βασιστούν στην ιστορία. Όρια στο στάδιο προεργασίας θέτουν παράγοντες όπως το είδος του βιντεοπαιχνιδιού που αναπτύσσεται, το αν εμπεριέχει σκηνές ή χαρακτήρες από κάποια ομώνυμη ταινία, αν υπόκειται σε πνευματικά δικαιώματα μιας άλλης εταιρείας ανάπτυξης οπτικοακουστικού υλικού, δηλαδή εταιρεία παραγωγής ταινιών, βίντεο μικρού μήκους, εικόνων ή άλλων βιντεοπαιχνιδιών.

Με δεδομένο το κόστος κατασκευής και την γενικότερη ιστορία του έργου, σειρά έχει η δημιουργία μιας ροής γεγονότων που θα οδηγήσουν τον πρωταγωνιστή να ξετυλίξει την πλοκή. Αυτό μπορεί να γίνει με την βοήθεια σκίτσων και διαφόρων έργων τέχνης (Concept Art) με έμφαση στα τοπία που θα διαδραματίζεται το παιχνίδι [12]. Επίσης με σύντομης διάρκειας κόμικ όπου ο πρωταγωνιστής θα εμφανίζεται σε διάφορες πόζες και θα κάνει κάποιες ενέργειες για να φτάσει τον στόχο του. Μαζί με τα γεγονότα, σχεδιάζονται και οι χαρακτήρες που θα συμβάλλουν με κάποιο τρόπο στην ιστορία, σε πολλές διαφορετικές πόζες, με δοκιμές σε φωτισμό και εφέ, με διάφορες στολές. Είναι πολύ σημαντικό να είναι σαφείς οι εικόνες ως προς το τι προσδοκούν από κάθε χαρακτήρα έτσι ώστε να υλοποιηθεί με μεγαλύτερη ευκολία η ιδέα σε τρισδιάστατο μοντέλο [10].

1.4.3 Παραγωγή (Production)

Έχοντας ολοκληρώσει επιτυχώς την απαραίτητη προεργασία και κατά συνέπεια το δεύτερο στάδιο ανάπτυξης, το προϊόν μεταβαίνει στο στάδιο παραγωγής [9]. Σε αυτό συνεργάζονται διάφορα εξειδικευμένα τμήματα τεχνικού προσωπικού με στόχο την ολοκλήρωση του έργου. Η ανάπτυξη γίνεται σύμφωνα με τις κατευθύνσεις που δίνονται από τον επικεφαλής κάθε τμήματος και ακολουθώντας πάντα πιστά το έντυπο σχεδίασης που συνέταξε η ομάδα ανάλυσης του προϊόντος. Κάθε τμήμα έχει στην διάθεση του εργαλεία που παρέχονται από την εκάστοτε εταιρεία και την απαραίτητη φαντασία αλλά και τεχνική γνώση για να φέρουν εις πέρας το έργο. Είναι πολύ συνηθισμένο να αλλάζουν τα αρχικά σχέδια λόγω μη συμφωνίας ιδεών ή αποτυχίας υλοποίησης. Οι ομάδες ανάπτυξης λοιπόν θα πρέπει να έχουν την απαραίτητη ευελιξία να αλλάξουν το περιεχόμενο ώστε να συμφωνεί με τα νέα δεδομένα και κατευθύνσεις που τους δίνονται. Για παράδειγμα, μπορεί σαν σχέδιο να φαίνεται καταπληκτικός ένας πολύπλοκος χαρακτήρας με διαστημική στολή, κράνος με φωσφορίζον περίβλημα και θώρακα με μικροκυκλώματα, καθώς επίσης και ειδικό εξοπλισμό αναρρίχησης με μαγνητικό πεδίο. Στην πράξη όμως θα είναι εξαιρετικά δύσκολο να υλοποιηθεί ένας χαρακτήρας αυτού του τύπου γιατί ίσως θα υπάρχει περιορισμένος χρόνος στον οποίο είναι αναγκαίο να είναι έτοιμος ο εν λόγω χαρακτήρας [11]. Ίσως δεν υπάρχει αρκετά υψηλής τεχνολογίας μέσο για να κατασκευαστεί το μοντέλο. Η εμφάνιση τεχνικών περιορισμών αναγκάζει τους σχεδιαστές να αλλάξουν λίγο την ιδέα για να ταιριάζει καλύτερα στους διαθέσιμους πόρους.

Στο στάδιο αυτό δημιουργείται όλη η τρισδιάστατη γεωμετρία που θα χαρακτηρίζει αυτό το έργο, περιβάλλον, χαρακτήρες, εφέ, αντικείμενα, διακοσμητικά στοιχεία, αρχιτεκτονική [9], [10]. Σύμφωνα πάντοτε με τις ανάγκες του έργου γίνεται προσεκτική ανάλυση και κατασκευή των διαφόρων τοποθεσιών τις οποίες ο πρωταγωνιστής θα επισκέπτεται. Αν το σενάριο αναφέρεται σε γεγονότα πολέμου μιας συγκεκριμένης περιόδου της ιστορίας, τα μέρη θα πρέπει να ακολουθούν το στυλ εκείνης της περιόδου, τα κτήρια να μοιάζουν με αυτά που συναντούμε σε φωτογραφικά ντοκουμέντα, οι ενδυμασίες να πείθουν, οι διάλογοι των χαρακτήρων να είναι εμπνευσμένοι από την τότε πραγματικότητα. Τα όπλα να ακολουθούν μια συγκεκριμένη τεχνολογία και η τακτική μάχης να μην ξεφεύγει πολύ [9], [10]. Μετά γίνεται προσθήκη όλων αυτών των στοιχείων που καθιστούν το περιβάλλον κατάλληλο για βιντεοπαιχνίδι, όπως για παράδειγμα η τοποθέτηση κουτιών πρώτων

βοηθειών και διάφορων όπλων για να βοηθήσουν τον παίκτη στην πρόοδο του, η απόδοση συγκεκριμένων συνθηκών φωτισμού για να είναι μεν ρεαλιστικό το τοπίο αλλά ο παίκτης να μπορεί να βλέπει με σχετική ευκολία [13].

Στα μοντέλα χαρακτήρων δίνεται πνοή μέσω της κίνησης. Καταγράφονται φυσικές κινήσεις με ειδικό εξοπλισμό (Motion Capture) ή δημιουργούνται μέσω ενός προγράμματος κίνησης (Animation Software) και κατόπιν γίνεται ο προγραμματισμός τους έτσι ώστε να επιλέγεται το σωστό κλιπ κίνησης σε κάθε περίπτωση μιας ενέργειας. Σε πιο σύγχρονα παιχνίδια δίνεται επιπλέον προσοχή στις κινήσεις του προσώπου (Facial Animation) για περισσότερη αληθοφάνεια και καταγράφονται ολόκληρες σκηνές σε στούντιο (Cinematics) όπου ηθοποιοί ενσαρκώνουν τον ρόλο κάποιου ψηφιακού χαρακτήρα ενώ πολλές φορές η διαδικασία παραγωγής δεν διαφέρει από αυτήν για την παραγωγή ενός κινηματογραφικού έργου [13], [14].

Τεχνικοί που ειδικεύονται στην ηχοληψία και την επεξεργασία ήχων αναλαμβάνουν το ακουστικό κομμάτι του έργου, όπου αποδίδουν ήχους σε ότι κινείται και προγραμματίζουν τα γεγονότα που οδηγούν στη σύνθεση ήχων ενώ είναι υπεύθυνοι για την καλή ποιότητα ήχου του τελικού προϊόντος. Τα βήματα, οι πυροβολισμοί, ο αντίλαλος, το θρόισμα φύλλων και ο ήχος μιας καταιγίδας είναι μερικά παραδείγματα από ηχητικά δείγματα που οι ηχολήπτες καλούνται να συγχρονίσουν με τα τρισδιάστατα γραφικά του έργου ώστε να είναι ολοκληρωμένη η εμπειρία του παίκτη και να νιώθει ότι παίζει ένα κινηματογραφικό αριστούργημα [12].

Το έργο περνάει πολλά στάδια παραγωγής κατά την διάρκεια της ανάπτυξης του. Αρχικά δημιουργείται ένα πρότυπο που περιλαμβάνει όλα τα χαρακτηριστικά που θα συμπεριλαμβάνονται σίγουρα στο τελικό αποτέλεσμα. Κατά την κατασκευή προτύπων η γεωμετρία που χρησιμοποιείται είναι περισσότερο κοσμητική και εξυπηρετεί μόνο στην οπτικοποίηση κάποιων λειτουργιών του παιχνιδιού [10]. Περισσότερη βάση δίνεται στον κώδικα έτσι ώστε να λειτουργεί σωστά το κάθε επιμέρους χαρακτηριστικό. Ο πρωταγωνιστής και οι υπόλοιποι χαρακτήρες είναι απλώς αντικείμενα αναφοράς με κάποιο βασικό μοντέλο που μοιάζει με κάτι από την πραγματικότητα. Αν ο πρωταγωνιστής είναι άνθρωπος, τότε και το μοντέλο προτύπου θα είναι μια κούκλα που μοιάζει με άνθρωπο. Έτσι είναι πιο εύκολο στους σχεδιαστές και στους προγραμματιστές να υπολογίσουν το πως θα γίνεται κάποια ενέργεια, από ποια απόσταση να ενεργοποιείται ένας μηχανισμός και άλλα τεχνικά θέματα. Είναι εξαιρετικά σημαντικό αυτό το στάδιο, [9], [10], γιατί από αυτό κρίνεται αν ένα βιντεοπαιχνίδι θα είναι βιώσιμο στην αγορά και αν θα προκαλέσει το ενδιαφέρον του κοινού. Πολλές ιδέες που φτάνουν στη προτυποποίηση απορρίπτονται ενώ σε άλλες, οι κατασκευαστές παρόλες τις επισημάνσεις και τις προειδοποιήσεις αποφασίζουν να ρισκάρουν να τις ολοκληρώσουν, μόνο που αποδεικνύεται ότι έπρεπε τελικά να τις εγκαταλείψουν. Βέβαια υπάρχουν και οι εξαιρέσεις, όπου θεωρητικά παρατημένα έργα με λίγη ακόμη επεξεργασία μετατρέπονται σε κάτι πολύ καινοτόμο που κατακτά μια θέση στο εμπόριο και στην καρδιά των παικτών, αλλά συχνά ο φόβος της αποτυχίας και της σπατάλης πόρων είναι μεγάλος.

Σε επόμενο στάδιο το βιντεοπαιχνίδι αποκτά μορφή που μοιάζει να είναι πιο κοντά στο τελικό στόχο [9], [12], [13], με πολλά από τα πρότυπα μοντέλα να αντικαθίστανται από άλλα, καλύτερης ποιότητας και κάποια περιβάλλοντα να έχουν φυσική διασύνδεση μεταξύ τους ώστε ο παίκτης να μπορεί να εξερευνήσει περισσότερο χώρο καθώς δοκιμάζει τις δυνατότητες του σε αυτόν.

Έχοντας πλέον αποκτήσει μια πιο φιλική μορφή προς τον παίκτη, το παιχνίδι μεταβαίνει σε έκδοση Άλφα όπου πολλά χαρακτηριστικά πέτυχαν και ενσωματώθηκαν ενώ άλλα απορρίφθηκαν και αφαιρέθηκαν. Αρκετό από το υλικό που δημιουργείται κατά την προτυποποίηση δεν φτάνει στην Άλφα έκδοση λόγων τεχνικών προβλημάτων ή θεμάτων αισθητικής, απόδοσης ή επιπρόσθετης πολυπλοκότητας.

1.4.4 Μετά την παραγωγή (Post production)

Το τελευταίο στάδιο είναι ο έλεγχος του προϊόντος. Το εν λόγω προϊόν θεωρείται πλέον ολοκληρωμένο, περιέχει πλοκή, σκηνικά, χαρακτήρες, μηχανικές, μουσική, εφέ και είναι πλήρως λειτουργικό. Βρίσκεται στην έκδοση Άλφα και η ομάδα ελέγχου παίζει και δοκιμάζει κάθε τι διαδραστικό εντός του βιντεοπαιχνιδιού. Κάθε απόφαση, κάθε μηχανική, την ιστορία, το κατά πόσο τα σκηνικά δεν έχουν οπτικά προβλήματα, αν ο ήχος συγχρονίζεται και ταιριάζει σε αυτό που περιγράφει. Αν η ομάδα παρατηρήσει λάθη συντάσσει τα απαραίτητα έγγραφα αναφοράς και τα δίνει στους αρμόδιους του επιτελείου ανάπτυξης ώστε να διορθωθούν. Μετά τις απαραίτητες ενέργειες το διορθωμένο προϊόν θεωρείται ότι έχει περάσει σε Βήτα έκδοση και δίνεται για εκτενή έλεγχο εκ νέου στην ομάδα ελέγχου [9]. Σε περιπτώσεις όπου εντοπιστούν λάθη κατά την δοκιμή της Βήτα έκδοσης αυτά βαρύνονται σύμφωνα με κάποιο βαθμό σημαντικότητας και αποδίδεται συγκεκριμένο περιθώριο για την επιτυχή διόρθωση τους. Επιπλέον, ο έλεγχος δεν γίνεται μόνο στην λειτουργικότητα του βιντεοπαιχνιδιού, αλλά και το κατά πόσο ακολουθεί τους κανόνες που διέπουν την κάθε πλατφόρμα στην οποία πρόκειται να κυκλοφορήσει. Αν εκτελείται σε κατάλληλο ρυθμό πλαισίων, το πόσο πολύπλοκες πράξεις απαιτεί, αν τα μενού είναι κατανοητά και λειτουργικά με βάση το χειρισμό της κάθε πλατφόρμας. Έπειτα, εφόσον πλέον έχουν λυθεί τα όποια ζητήματα πολυπλοκότητας, λογικά λάθη και προβλήματα απεικόνισης, αποστέλλεται το προϊόν για έλεγχο από τους κατασκευαστές της κάθε πλατφόρμας. Στην περίπτωση που είναι να κυκλοφορήσει για υπολογιστές, την ευθύνη της καλής λειτουργίας του φέρει η κατασκευάστρια εταιρεία του προϊόντος [9].

1.5 Ανάπτυξη βιντεοπαιχνιδιού – Ρόλοι στην παραγωγή

Το βιντεοπαιχνίδι διανύει μια πορεία στην ανάπτυξη και τελικά την διανομή του στο κοινό. Καθ’ όλη τη διάρκεια που καταλαμβάνει το κάθε στάδιο ανάπτυξης, διάφοροι ρόλοι εμπλέκονται έτσι ώστε κάποιο τμήμα του βιντεοπαιχνιδιού να ολοκληρωθεί με επιτυχία. Σε μεγάλες εταιρείες αναλαμβάνουν δράση πολλά διαφορετικά τμήματα για την υλοποίηση έστω και ενός στοιχείου του βιντεοπαιχνιδιού [10], [15], ενώ σε μικρότερα στούντιο και σε ερασιτέχνες παραγωγούς παρατηρείται συγκέντρωση ευθυνών και απαιτήσεων σε λίγα άτομα, ενώ κάποια θέματα αγνοούνται εντελώς, εφόσον η ομάδα δεν στοχεύει στην παράδοση ενός κορυφαίου τίτλου στο εμπόριο [9], [14].

1.5.1 Διανομέας (Publisher)

Με όλη την αναφορά στην παραγωγή ενός τίτλου και την επιτυχή τοποθέτηση του στην αγορά, δεν γίνεται να παραλειφθεί ο ρόλος κλειδί ενός βιντεοπαιχνιδιού, δηλαδή ο διανομέας (Publisher) [16]. Ο διανομέας είναι υπεύθυνος για την επιτυχή συσκευασία του παιχνιδιού σε ψηφιακή ή φυσική μορφή και την διανομή του μέσω διαφόρων καταστημάτων. Λαμβάνει αποφάσεις σχετικά με την υποστήριξη ενός έργου παραγωγής βιντεοπαιχνιδιού και τον τρόπο με τον οποίο θα προσεγγίσει το κοινό έτσι ώστε να έχει κάποιο κέρδος. Οπότε οι διανομείς τείνουν να επενδύσουν σε τίτλους που κατά κάποιο ποσοστό δύνανται να γίνουν εμπορικές επιτυχίες σε κάποιο χρονικό διάστημα από την κυκλοφορία τους. Όπως επίσης συχνά απορρίπτουν καλές ιδέες ή ποιοτικά έργα γιατί δεν πληρούν τις προϋποθέσεις για επιτυχία και κέρδος κατά τον διανομέα. Διανομείς είναι οι διάφορες εταιρείες που βάζουν το λογότυπο τους στις εναρκτήριες σκηνές ενός βιντεοπαιχνιδιού, πάνω στο κουτί του δίσκου και στους τίτλους τέλους.

1.5.2 Παραγωγός (Producer)

Ο παραγωγός είναι ο ενδιάμεσος μεταξύ του χρηματοδότη του έργου, δηλαδή του διανομέα και του τεχνικού προσωπικού που θα διεκπεραιώσει το έργο. Είναι υπεύθυνος για την διαχείριση των

διαθέσιμων πόρων που έχουν ανατεθεί στην παραγωγή του βιντεοπαιχνιδιού. Αυτοί είναι το ανθρώπινο δυναμικό, το χρηματικό ποσό που έχει διατεθεί από τον διανομέα, το χρονικό περιθώριο που έχει δοθεί για την υλοποίηση του έργου και τα απαραίτητα εργαλεία που θα χρειαστούν κατά την διάρκεια κατασκευής επιμέρους στοιχείων. Ευθύνη του παραγωγού είναι να συντονίσει την ομάδα σχεδίασης, να δώσει κατευθύνσεις στην τεχνική ομάδα ανάπτυξης και να παρουσιάσει αποτέλεσμα στον διανομέα ο οποίος επιβλέπει ανά τακτά χρονικά διαστήματα για την πρόοδο του έργου και για το αν καλύφθηκαν οι στόχοι που έχουν ανατεθεί σε κάθε ομάδα [17].

1.5.3 Επικεφαλής σχεδιαστής (Lead Designer)

Κάθε έργο έχει απαραίτητα έναν επικεφαλής σχεδιαστή του οποίου η κύρια αρμοδιότητα είναι να κατασκευάσει με θεωρητικό τρόπο ένα παιχνίδι το οποίο θα μπορέσει να προκαλέσει την εντύπωση του κοινού και να είναι ευχάριστη η εμπειρία που θα προσφέρει με το δικό της μοναδικό τρόπο. Είναι ένας από τους πιο δύσκολους ρόλους διότι απαιτεί φαντασία, δημιουργικότητα, ηγετική ικανότητα, δηλαδή να μπορεί να επικοινωνεί με την ομάδα του και να δίνει αποτελεσματικά κατευθύνσεις για το τι κρίνει απαραίτητο να δημιουργηθεί έτσι ώστε να κατασκευαστεί ένα αποτέλεσμα που ο διανομέας ή το κοινό θα μπορεί να κρίνει κατάλληλα. Ο σχεδιαστής είναι υπεύθυνος επίσης για την λήψη σημαντικών αποφάσεων κατά την διάρκεια ανάπτυξης του έργου, όπως το τι θα πρέπει να προστεθεί ή να αφαιρεθεί από το σύνολο και τι θα πρέπει να αλλάξει ώστε να είναι πιο ενδιαφέρον το αποτέλεσμα. Τέλος, όντας σχεδιαστής, θα πρέπει ο ίδιος να μπορεί να φτιάξει κάποιο σχέδιο για το έργο, δηλαδή χάρτες, τοπία, χαρακτήρες, οτιδήποτε είναι αναγκαίο να αποδώσει ο ίδιος ώστε να γίνει κατανοητό από την ομάδα του [18], [19].

1.5.4 Επικεφαλής σχεδιαστής παιχνιδιού (Lead Gameplay Designer)

Ο σχεδιαστής παιχνιδιού είναι αυτός που δίνει τις ιδέες και προγραμματίζει μηχανικές οι οποίες θα λειτουργήσουν σε ένα βιντεοπαιχνίδι. Με τις δικές του κατευθύνσεις καθιερώνεται ένας τρόπος με τον οποίο παίζεται το παιχνίδι και το καθιστά μοναδικό αν είναι αρκετά φιλοσοφημένες οι μηχανικές. Είναι λοιπόν εξίσου σημαντικός ο ρόλος του σχεδιαστή παιχνιδιού, ειδικά για τον παίκτη διότι το πως παίζεται το παιχνίδι είναι αυτό που του αποτυπώνεται πιο έντονα. Μηχανική αποτελεί οτιδήποτε μπορεί να κάνει ένας παίκτης στο βιντεοπαιχνίδι [10], [13], για παράδειγμα η δυνατότητα να πέσει από μια πλατφόρμα όταν φτάνει στην άκρη της, το να μπορεί να ανοίξει έναν μηχανισμό με μοχλό ή διακόπτη, το να πολεμάει ή να συνδιαλέγεται με άλλους χαρακτήρες του βιντεοπαιχνιδιού και πολλές άλλες ψηφιακές δραστηριότητες που ενδέχεται να συναντήσουμε σε ένα βιντεοπαιχνίδι. Η επικοινωνία του με τον παραγωγό είναι συνεχής καθώς προσθήκες και αφαιρέσεις στις μηχανικές γίνονται συχνά κατά την πορεία εξέλιξης του έργου [18], [20]. Μετά από την πρώτη επαφή του κοινού με μια μηχανική μπορεί να εκφέρουν κάποια κριτική η οποία θα επηρεάσει άμεσα το ενδεχόμενο να μείνει ή να διαγραφεί από το τελικό βιντεοπαιχνίδι.

1.5.5 Επικεφαλής Συγγραφέας (Lead Writer)

Το βιντεοπαιχνίδι, ειδικά στην σύγχρονη αγορά δεν είναι απλώς μια επαναλαμβανόμενη διαδικασία που προσφέρει ψυχαγωγία. Αντίθετα, περιβάλλεται από ένα σενάριο, σκηνές δράσης, χαρακτήρες κλειδιά, χρονολογικά γεγονότα, τρόπους αφήγησης, διαλόγους. Θα μπορούσε να πει κανείς ότι δεν απέχει πολύ από μια ταινία. Αφαιρώντας το τομέα παρουσίασης και τα στοιχεία παιχνιδιού, το βιντεοπαιχνίδι θα μπορούσε να μετατραπεί σε ένα αξιόλογο βιβλίο. Ο ρόλος λοιπόν του επικεφαλής της συγγραφικής ομάδας είναι να δημιουργήσει μια ιστορία για το βιντεοπαιχνίδι και να αποτυπώσει σε κείμενο όλη την δράση που θα δει το κοινό. Επιπλέον συντονίζει την ομάδα του και μοιράζεται τις απαραίτητες γνώσεις

και ιδέες που ενδέχεται να έχει με τα υπόλοιπα μέλη έτσι ώστε η τελική ιστορία να είναι δεμένη και να αποτελεί ενιαίο σύνολο. Τέλος βρίσκεται σε επικοινωνία με τον παραγωγό έτσι ώστε να εδραιωθεί η ιστορία και να κοινοποιηθεί στους σχεδιαστές για να φτιάξουν κατάλληλο περιεχόμενο. Η ευθύνη των συγγραφέων είναι μεγάλη γιατί πολύ συχνά η ανάπτυξη του έργου ξεφεύγει και συγκεντρώνονται πολλά στοιχεία που οι συγγραφείς πρέπει να ταιριάξουν μεταξύ τους για να υπάρχει λογική συνοχή. Κάτι που θα παρατηρήσει το κοινό είναι διάφορα ανεξάρτητα κομμάτια του βιντεοπαιχνιδιού όπως χαρακτήρες ή εχθρούς που δεν ταιριάζουν με την συνολική αισθητική ή γεγονότα που δεν αποσκοπούν σε τίποτα [21], [22].

1.5.6 Επικεφαλής Προγραμματιστής (Lead Programmer)

Το βάρος της ανάπτυξης του έργου συγκεντρώνεται όλο στην παραγωγή αποτελεσματικού και πλήρως κατανοητού κώδικα ο οποίος θα εκτελεί και θα παρουσιάζει σωστά όλα τα επιμέρους υλικά του έργου. Είναι ένας από τους πιο σημαντικούς συνδετικούς κρίκους που βοηθάει στην διάκριση μεταξύ ενός ποιοτικού βιντεοπαιχνιδιού και μιας προχειρότητας στην χειρότερη περίπτωση. Επικεφαλής προγραμματιστής μιας ομάδας ανάπτυξης κώδικα θεωρείται το άτομο με τις περισσότερες δεξιότητες στον προγραμματισμό. Είναι το άτομο που μπορεί να διεκπεραιώσει σχεδόν οποιαδήποτε εργασία προγραμματισμού του ανατεθεί, οπότε και επιλέγεται για τα δύσκολα κομμάτια κώδικα έτσι ώστε να οδηγήσει και να εμπνεύσει την ομάδα στην αποστολή της [23].

1.5.7 Τεχνικός Συντονιστής (Technical Director)

Ο τεχνικός συντονιστής κατευθύνει τα διάφορα τμήματα ανάπτυξης και δένει μεταξύ τους προγράμματα, λειτουργίες και διασυνδέσεις με τρόπο έτσι ώστε να μην υπάρχουν αλληλοεπικαλύψεις και αντιδράσεις μεταξύ των εκτελέσιμων. Είναι επίσης υπεύθυνος για την επίδειξη τεχνικών και λειτουργιών μιας συγκεκριμένης τεχνολογίας που κρίνει κατάλληλη να χρησιμοποιηθεί για την ανάπτυξη του έργου. Προτείνει λύσεις τεχνικού χαρακτήρα σε προβλήματα σχεδίασης και οδηγεί τα τμήματα παραγωγής στην έγκαιρη και επιτυχή παράδοση του έργου [24].

1.5.8 Σχεδιαστής επιπέδων (Level Designer)

Ο σχεδιαστής επιπέδων δημιουργεί τον κόσμο στον οποίο θα διαδραματίζονται τα γεγονότα του παιχνιδιού από την σκοπιά των διαθέσιμων δραστηριοτήτων για τον παίκτη. Δημιουργεί τον διαθέσιμο προς εξερεύνηση χάρτη του παιχνιδιού και όλους τους μηχανισμούς που θα απαιτούνται για την πρόοδο στο παιχνίδι. Καταστρώνει σχέδια για τις προκλήσεις που θα συναντήσει ο παίκτης και δένει με δημιουργικό τρόπο τα περιβάλλοντα μεταξύ τους. Ο σχεδιαστής επιπέδων βρίσκεται σε διαρκή επαφή με τον συντονιστή και τον σχεδιαστή του παιχνιδιού καθώς ο κόσμος που φτιάχνει βασίζεται στην υπάρχουσα ιστορία, τις απαιτήσεις του παιχνιδιού και το είδος του [25]. Πχ δεν μπορεί να κατασκευάσει περιβάλλον με πλατφόρμες σε ένα παιχνίδι προσομοίωσης ενός αθλήματος. Αλλά μπορεί να φτιάξει ένα περιβάλλον με γρίφους για ένα παιχνίδι βολών. Αρκεί να συμφωνεί με τις απόψεις και τις κατευθύνσεις του κατασκευαστή.

1.5.9 Σχεδιαστής εμπειρίας χρήστη (User Experience Designer)

Ο σχεδιαστής εμπειρίας είναι υπεύθυνος για την δημιουργία μιας ικανοποιητικής εμπειρίας για τον παίκτη. Προσπαθεί να κατανοήσει τις ανάγκες του εκάστοτε κοινού και να εξηγήσει το τι συμβαίνει στο παιχνίδι με τρόπο κατανοητό προς το κοινό. Οι σχεδιαστές εμπειρίας τείνουν να απλοποιούν τα στοιχεία που βλέπει ο παίκτης κάθε φορά στην οθόνη του έτσι ώστε να μην είναι κουραστικό το περιεχόμενο και προσπαθούν να δώσουν ότι χρειάζεται κάθε φορά για να κρατούν το ενδιαφέρον του

παίκτη υψηλά. Επεμβαίνουν στην διεπαφή του παιχνιδιού, στα εφέ, στο τρόπο με τον οποίο εξηγείται η ιστορία του παιχνιδιού και στον τρόπο με τον οποίο ο παίκτης αλληλοεπιδρά στον τρισδιάστατο χώρο. Συχνά εφευρίσκουν τρόπους για να νιώθει ο παίκτης πιο άνετα καθώς σημειώνει πρόοδο στο παιχνίδι ή τρόπους έτσι ώστε να νιώθει ωραία κάνοντας διάφορα πράγματα [26]. Το επιτυγχάνουν μέσω της εισαγωγής χαρακτηριστικών ήχων όταν ο παίκτης καταφέρει κάτι ή κάνει μια συγκεκριμένη ενέργεια, με σημεία όπου το παιχνίδι αποθηκεύεται αυτόματα (checkpoints) έτσι ώστε να μην χρειάζεται ο παίκτης να επαναλάβει ένα μεγάλο κομμάτι του παιχνιδιού αν αποτύχει παρακάτω και με διάφορα άλλα στοιχεία που δεν ακολουθούν τη λογική αλλά είναι ευχάριστο το αποτέλεσμα που δίνουν.

1.5.10 Προγραμματιστής Γραφικών (Graphics Programmer)

Ο προγραμματιστής γραφικών έχει συνήθως βαθιά γνώση της χρήσης τεχνολογιών γραφικών έτσι ώστε το παιχνίδι από την άποψη των γραφικών να φαίνεται όσο καλύτερο γίνεται, χωρίς να είναι υπερβολικά απαιτητικό σε πόρους. Είναι υπεύθυνος για την σωστή παραμετροποίηση των μοντέλων και των εφέ ώστε να μην δημιουργούν πρόβλημα στην απόδοση του παιχνιδιού σε πραγματικό χρόνο [27]. Ένα παράδειγμα, φτιάχνει κώδικα που απαλείφει γεωμετρία μετά από συγκεκριμένη απόσταση ορατότητας (Visibility Distance) για να χωρέσει περισσότερα στην κάθε παρούσα σκηνή. Επιπλέον φορτώνει και αποφορτώνει στοιχεία στην μνήμη γραφικών με τρόπο ώστε να μην προκαλέσει πρόβλημα η απουσία τους αλλά να μείνει χώρος στη μνήμη για άλλες πράξεις.

1.5.11 Μηχανικός Ήχου (Audio Engineer)

Ο μηχανικός ήχου φροντίζει για το περιεχόμενο του ήχου που ο παίκτης θα ακούει κατά την διάρκεια του παιχνιδιού. Εισάγει ηχητικά εφέ όλων των κατηγοριών στο χώρο και στις δράσεις των χαρακτήρων ώστε να δώσει ζωντάνια στο παιχνίδι. Κάποιες φορές είναι υπεύθυνος για την απουσία ήχου ώστε να δώσει μια πιο δραματική αίσθηση σε μια σκηνή [28]. Σε πιο μικρά στούντιο όπου δεν υπάρχει έκταση στο επιτελείο παραγωγής, ο μηχανικός ήχου αναλαμβάνει και τον κώδικα των ηχητικών εφέ που θα τα μορφοποιεί σε διαφορετικές καταστάσεις, για παράδειγμα όπως ακούγονται μέσα στο νερό ή στον αέρα αν το παιχνίδι δίνει την δυνατότητα για κάτι τέτοιο.

1.6 Ανάπτυξη βιντεοπαιχνιδιού – Πρώτες ύλες

Ένα βιντεοπαιχνίδι, όταν πλέον βρίσκεται στην τελική μορφή του, συνδυάζει επιτυχημένα ψηφιακές και αναλογικές πρώτες ύλες. Αυτές συναντώνται με την μορφή δισδιάστατων – τρισδιάστατων γραφικών, δειγμάτων ήχου και μουσικής, φωνητικά αρχεία, αρχεία περιγραφής κίνησης, πίνακες αντιστοιχίας με μετρήσεις και αποτελέσματα πράξεων, κείμενο, αρχεία εικόνων και αρχεία κώδικα. Η κάθε εταιρεία βιντεοπαιχνιδιών διαθέτει εργαλεία και εξειδικευμένο προσωπικό για την παραγωγή πρώτων υλών αλλά και τη σύνθεση τους στη πορεία της δημιουργίας. Το υλικό που θα συναντήσει κανείς σε ένα βιντεοπαιχνίδι είναι κατά κύριο λόγο οπτικοακουστικής μορφής, όπως φαίνεται και από τα παραδείγματα [8], [13], [14]. Φυσικά, υπάρχει και μεγάλη ποικιλία σε προγράμματα ψηφιοποίησης σχεδόν για οτιδήποτε αφορά αυτή τη μορφή, δηλαδή εικόνα και ήχο. Προγράμματα ηχογράφησης και μίξης ήχου για το ακουστικό υλικό. Προγράμματα παραγωγής και τοποθέτησης γραφικών για το οπτικό και ακουστικό υλικό. Επιπλέον υπάρχουν πολλά προγράμματα για μετάφραση κίνησης από βίντεο σε τρισδιάστατη ψηφιακή μορφή.

1.7 Επίλογος

Όπως προκύπτει από την ανάλυση ενός βιντεοπαιχνιδιού, αυτό αποτελεί ένα πρόγραμμα με κύριο στόχο την ψυχαγωγία του παίκτη - χρήστη. Για την κατασκευή του εμπλέκονται πολλές διαφορετικές

ειδικότητες και προγράμματα προσανατολισμένα σε μία λειτουργία τη φορά. Διαφορετικά επιμέρους στοιχεία συνδυάζονται με κατάλληλες τεχνικές έτσι ώστε να δώσουν πνοή στο βιντεοπαιχνίδι και να αποτελεί διαδραστική εφαρμογή. Στη συνέχεια θα αναλυθεί ένα πολύ βασικό εργαλείο που επιταχύνει σημαντικά την παραγωγή ενός βιντεοπαιχνιδιού. Αυτό είναι μια μηχανή γραφικών, το πλέον πολυμορφικό εργαλείο με το οποίο οι παραγωγοί συνδυάζουν όλες τις πρώτες ύλες και όλες τις ειδικότητες του τεχνικού προσωπικού με τρόπο έτσι ώστε να δημιουργηθεί ένα ενιαίο πρόγραμμα, ένα βιντεοπαιχνίδι.

Κεφάλαιο 2ο: Μηχανή γραφικών Unreal Engine 4

2.1 Εισαγωγή

Για την κάλυψη των περισσότερων απαιτήσεων στον τομέα των βιντεοπαιχνιδιών χρησιμοποιείται μια μηχανή γραφικών. Έχει πολυμορφικό χαρακτήρα και μπορεί να ταιριάξει σε πολλές διαφορετικές ανάγκες της παραγωγής. Είναι επεκτάσιμη και μπορούν σχετικά εύκολα να προστεθούν άλλες λειτουργίες χωρίς να επηρεαστεί η βασική χρήση της μηχανής. Συνήθως είναι κλειστού κώδικα και απαιτεί κάποια άδεια εμπορικής ή ακαδημαϊκής χρήσης, ενώ οι άδειες μπορεί να είναι με ενοίκιο ή μόνιμης κατοχής. Κάποιες μηχανές γραφικών είναι πιο πολύπλοκες από άλλες και απαιτούν πιστοποίηση γνώσης χειρισμού, άλλες απευθύνονται σε λιγότερο επαγγελματικό κοινό [9], [13]. Αντικειμενικά, οι μηχανές γραφικών είναι ένα απαραίτητο εργαλείο για το δέσιμο όλων των στοιχείων που αποτελούν ένα πολύπλοκο έργο και τη συσκευασία τους σε ένα ενιαίο σύνολο το οποίο μπορεί να αναπαραχθεί σε κάποια συσκευή στόχο.

Ανάμεσα στις πλέον διαδεδομένες μηχανές γραφικών, που διανέμονται ελεύθερα για κατασκευή έργων, είναι το Unreal Engine, έκδοσης 4 και πρόσφατα 5. Με αυτήν τη μηχανή αναπτύχθηκε η παρούσα εργασία, συγκεκριμένα με την έκδοση 4.25 και θα αναλυθεί εκτενώς στα επόμενα κεφάλαια αυτής της ενότητας, ξεκινώντας από τα βασικά. Το πρώτο νούμερο στην έκδοση σηματοδοτεί ένα οικοσύστημα λειτουργιών και διαδικασιών που συμβάλουν στην παραγωγή περιεχομένου που διαφέρει αρκετά από τα υπόλοιπα, ενώ τα νούμερα μετά την τελεία σηματοδοτούν αλλαγές στην λειτουργία της βασικής έκδοσης και διορθώσεις. Είναι ένα λογισμικό ανοικτού κώδικα με κλειστή άδεια χρήσης που ανήκει στην Epic Games και ευέλικτες παραλλαγές της άδειας είναι διαθέσιμες για διάφορες περιπτώσεις. Λειτουργεί σε υπολογιστές με λειτουργικό σύστημα Windows, Mac και Linux. Οι τεχνικές προδιαγραφές για την χρήση της μηχανής είναι αρκετά υψηλές αλλά υποστηρίζονται από μια ευρεία γκάμα υπολογιστικών συστημάτων, ακόμα και από συστήματα με προηγούμενης γενιάς εξαρτήματα. Ειδικά για την ανάπτυξη κώδικα, συνεργάζεται πολύ καλά με το εργαλείο Visual Studio έκδοσης 2015, 2017, 2019 και άνω ενώ επιπρόσθετα διατίθεται από την Epic Games ένα ειδικό σύστημα φόρτωσης και μεταγλώττισης κώδικα κατάλληλο για το Unreal Engine, τα Unreal Engine VS Tools. Ο κώδικας της μηχανής παρέχεται για ανάγνωση και χρήση σε διαδικασίες αποσφαλμάτωσης και κατανόησης των εσωτερικών συναρτήσεων. Διαθέτει τεκμηρίωση και οδηγούς χρήσης στον κεντρικό ιστότοπο της εταιρείας καθώς και βιντεομαθήματα, ελεύθερα ή επί πληρωμή [29], [30].

2.2 Δυνατότητες και εφαρμογές

Η μηχανή αυτή είναι κατάλληλη για την κατασκευή βιντεοπαιχνιδιών, ταινιών και τρέιλερ για διάφορες κυκλοφορίες, εφαρμογών επαυξημένης πραγματικότητας (VR), διαδραστικές εφαρμογές εκπαιδευτικού χαρακτήρα, εφαρμογές ιστοσελίδων με τρισδιάστατα γραφικά και γενικά οτιδήποτε μπορεί να προβληθεί. Παρατηρείται χρήση της ακόμα και για αυτοματοποιημένη κατασκευή κώδικα για ποικίλα έργα ανάπτυξης εφαρμογών. Το μόνο που δεν μπορεί να κυκλοφορήσει μέσω αυτής της μηχανής είναι ο πυρήνας της μηχανής σαν νέα μηχανή ή πρόγραμμα χωρίς να μοιράζεται την ίδια άδεια χρήσης με το Unreal Engine. Όπως επίσης δεν μπορεί να χρησιμοποιηθεί για οποιαδήποτε προσομοίωση στρατιωτικού χαρακτήρα [30], [31]. Χρησιμοποιεί κατά κύριο λόγο C++ για ανάπτυξη κώδικα ενώ μπορούν να προστεθούν διάφορα επεκτάσιμα (plugins) γραμμένα σε διάφορες γλώσσες, αρκεί να μεταγλωττιστούν σύμφωνα με το πηγαίο κώδικα του Unreal Engine. Διαθέτει επίσης έναν τρόπο ανάπτυξης κώδικα με οπτικό τρόπο, τα λεγόμενα Blueprints. Με τη βοήθεια έτοιμων μπλοκ εντολών και συνδέσεων διατεταγμένων σε κανονικό σύστημα αξόνων, είναι δυνατό να παραχθούν πολύ γρήγορα

λειτουργικές μονάδες όπως διάφορες μηχανικές και απλές συναρτήσεις. Είναι η καλύτερη μέθοδος ανάπτυξη πρωτοτύπων, αλλά για το τελικό αποτέλεσμα προτείνεται ο συνδυασμός των Blueprint με κώδικα C++, με δεδομένο τις αδυναμίες και τα προτερήματα του κάθε τρόπου.

Απευθύνεται τόσο σε ερασιτέχνες και ομάδες μικρού μεγέθους όσο και σε επαγγελματίες και μεγάλες ομάδες ή τμήματα εταιρειών [29]. Υπάρχει βοηθητικό υλικό σε μορφή γραπτού κειμένου και βιντεομαθήματα για τα περισσότερα πεδία εφαρμογής της μηχανής. Επίσης η Epic Games δίνει πρόσβαση σε κάποια βασικά μοντέλα, υλικά και εφέ για την γρήγορη κατασκευή προτύπων και τις δοκιμές που απαιτούνται πριν την υλοποίηση κάποιας λειτουργίας. Εκτός από τα ελεύθερης πρόσβασης υλικά, διαθέτει ηλεκτρονικό κατάστημα, το Unreal Marketplace όπου ένας κατασκευαστής μπορεί να βρει ποικίλα έργα από πολλές διαφορετικές θεματολογίες. Το ηλεκτρονικό κατάστημα ανοίγει στον κατασκευαστή ενός έργου νέους ορίζοντες και του δίνει την δυνατότητα να προσαρτήσει γραφικά, ήχους, υφές, φίλτρα και κώδικα στο δικό του έργο και να τα συνθέσει με τρόπο ώστε να αποτελούν ένα ενιαίο σύνολο [32]. Τέλος υπάρχουν έτοιμα έργα τα οποία μπορεί να προσαρτήσει στις δημιουργίες του ο κάθε χρήστης ελεύθερα ή ακόμα και να μάθει τον τρόπο κατασκευής κάποιων ιδιαίτερων λειτουργιών. Αυτά συμπεριλαμβάνουν τοπία φτιαγμένα με φωτογραμμετρία, αρχιτεκτονικά σχέδια, μικρού μήκους βιντεοπαιχνίδια που αναδεικνύουν διάφορους τρόπους αντιμετώπισης σχεδιαστικών προβλημάτων. Είναι αρκετό το υλικό σε ελεύθερη πρόσβαση για να ξεκινήσει κανείς να δημιουργεί, αλλά χρειάζεται επαρκείς δοκιμές για να μπορεί να θεωρείται κάτι ότι έχει φτάσει σε έκδοση Άλφα [29], [30], [33]. Ενώ διαθέτει μηχανισμούς για περισσότερη ευκολία στην ανάπτυξη ενός έργου, σχεδόν ότι λειτουργία θα κληθεί να φτιάξει ο προγραμματιστής ή ο καλλιτέχνης, θα πρέπει να το υλοποιήσει από το μηδέν ή να το αγοράσει μέσω των διαθέσιμων πόρων του ηλεκτρονικού καταστήματος της Epic. Δεν εξηγούνται θέματα που αφορούν την συσχέτιση της δομής της C++ με την δομή του κώδικα του Unreal Engine στα μαθήματα και πολλές φορές δεν υπάρχει επαρκής τεκμηρίωση του κώδικα στα έτοιμα έργα. Η κατασκευή λοιπόν ενός πολύπλοκου έργου όπως ένα βιντεοπαιχνίδι προϋποθέτει ο χρήστης να έχει επαρκή γνώση της γλώσσας προγραμματισμού και της ορολογίας που χρησιμοποιείται συχνά κατά την διάρκεια της ανάπτυξης [34]. Παρόλα αυτά, είναι εφικτό να δημιουργηθούν καταπληκτικά έργα σε λίγες μόνο εβδομάδες χάρη στο ελεύθερα προσβάσιμο υλικό, ακόμα και χωρίς καθόλου εμπειρία και προϋπολογισμό.

2.3 Τεχνικές προδιαγραφές

Η μηχανή για σύστημα Windows υποστηρίζεται από την έκδοση Windows 7 έως και την 11 με προτεινόμενη την 10. Από πλευράς επεξεργαστικής ισχύος, απαιτεί ένα σύστημα με τετραπύρηνο Intel ή AMD επεξεργαστή με ισχύ τουλάχιστον 2.5 Gh και πάνω. Από άποψης γραφικών, χρειάζεται ένα κύκλωμα γραφικών με έκδοση DirectX 11 ή 12, αναλόγως με το ποιας έκδοσης πράξεις γραφικών θα χρειαστεί ο κατασκευαστής. Τέλος από άποψης αποθηκευτικού χώρου, χρειάζεται αρκετό χώρο για την εγκατάσταση της μηχανής αλλά και να υπάρχει αρκετός διαθέσιμος για επιπλέον υλικό [35]. Μια τυπική εγκατάσταση της μηχανής με υποστήριξη έργων μόνο για κονσόλες και υπολογιστές καταλαμβάνει χώρο της τάξης των 30 Gb, ενώ με αρκετά επιπρόσθετα η ανάγκη σε αποθηκευτικό χώρο ανεβαίνει γρήγορα. Ειδικά στα έργα που περιλαμβάνουν τρισδιάστατο υλικό όπως αρχιτεκτονική, χαρακτήρες και διακοσμητικά ή ηχητικά εφέ. Επίσης, για έργα που αναπτύσσονται κυρίως σε C++, τα ειδικά σύμβολα αποσφαλμάτωσης, που είναι αναγκαία για την ορθό προγραμματισμό με τη μηχανή, καταλαμβάνουν περίπου 28Gb. Με αρκετά πειράματα που έγιναν κατά την διάρκεια υλοποίησης της εργασίας καθώς και από τις προτάσεις της εταιρείας, απαιτείται ένας αρκετά γρήγορος SSD δίσκος για την ανάγνωση και αρχειοθέτηση των αρχείων της μηχανής σε αποδεκτό χρονικό διάστημα.

2.4 Λειτουργικές λεπτομέρειες

Όταν ανοίγει το έργο για πρώτη φορά στην μηχανή Unreal Engine, θα πρέπει να μεταγλωττιστούν όλα τα αρχεία κώδικα που δημιουργήθηκαν αυτόματα και όλα τα γραφικά να περαστούν σε προσωρινό αποθηκευτικό χώρο στον δίσκο που λειτουργεί σαν κρυφή μνήμη. Έτσι, σε επόμενες εκκινήσεις δεν θα καθυστερεί η φόρτωση του έργου. Κάθε φορά που προστίθεται ένα νέο αρχείο κώδικα θα πρέπει να μεταγλωττιστεί μια φορά έτσι ώστε να δημιουργηθεί ένα παράγωγο του στη λίστα των φορτωμένων προγραμμάτων της μηχανής. Απαραίτητη προϋπόθεση για καινούργια αρχεία είναι να μην έχουν κανένα συντακτικό λάθος και καμία ασάφεια που εμποδίζει τη μεταγλώττιση τους. Εάν δεν ισχύει αυτό η μηχανή δεν μπορεί να αρχικοποιηθεί και να ανοίξει σε επόμενες κλήσεις έως ότου γίνει σωστά η μεταγλώττιση.

2.4.1 Απόδοση μηχανής

Ο χρόνος που χρειάζεται για να μεταγλωττιστεί το έργο είναι ανάλογος της ισχύος του επεξεργαστή του υπολογιστή και της ταχύτητας ανάγνωσης του σκληρού δίσκου. Η μηχανή χρησιμοποιεί workers για παραλληλισμό αλλά δεν αρκεί να έχει πολλούς πυρήνες ο επεξεργαστής. Πρέπει να είναι χρονισμένος αρκετά υψηλά και να εκτελεί μαζικά μεγάλο αριθμό από εντολές για να είναι αποδεκτός ο αρχικός χρόνος αναμονής. Επίσης, η αρχιτεκτονική του να είναι σχετικά καινούργια. Η διαθέσιμη μνήμη RAM δεν επηρεάζει την ταχύτητα φόρτωσης ούτε την απόδοση ιδιαίτερα, γιατί αρχικά δεν χρησιμοποιείται σχεδόν καθόλου όσο ο δίσκος. Αρκεί να είναι μεγαλύτερης ή ίσης χωρητικότητας με 8Gb. Με δοκιμές που έγιναν σε επεξεργαστή Intel i5 8ης γενιάς με 2.8 Ghz ταχύτητα εκτέλεσης πράξεων και έναν δίσκο Solid State με ρυθμό ανάγνωσης 7Gb/s, απαιτούνταν περίπου δέκα λεπτά για να ολοκληρωθεί η μεταγλώττιση του κώδικα και των υλικών. Το ίδιο περιεχόμενο στον ίδιο επεξεργαστή αλλά με σκληρό δίσκο HDD με ταχύτητα ανάγνωσης 730Mb/s, η μηχανή απαιτούσε 30 – 40 λεπτά όπου σε αυτό το διάστημα το υπολογιστικό σύστημα δεν μπορούσε να διαχειριστεί άλλα αιτήματα και εργασίες. Τα υλικά (Materials και Shaders) φάνηκαν να είναι τα πιο χρονοβόρα στην μεταγλώττιση.

Να σημειωθεί ότι μια συγκεκριμένη διαδικασία ακολουθείται κάθε φορά που δημιουργείται ένα νέο υλικό, τροποποιείται ένα υπάρχον ή τροποποιείται ένα αρχείο κώδικα. Πιο συγκεκριμένα, η μηχανή θα πρέπει να κάνει εκ νέου μεταγλώττιση και επαναφόρτωση, οπότε ανάλογα με τον όγκο δεδομένων που προστέθηκαν θα απαιτείται ανάλογος χρόνος. Για αρχεία κώδικα που δεν τροποποιούνται οι ταυτότητες των συναρτήσεων και οι καθολικές μεταβλητές, χρειάζεται από 30 δευτερόλεπτα έως ένα λεπτό για μεταγλώττιση. Αν αλλάξουν και τα παραπάνω, χρειάζεται γύρω στα 1-2 λεπτά. Αν προστεθεί ένα νέο γραφικό ή τροποποιηθεί ένα υπάρχον, ανάλογα με το πόσες φορές γίνεται αναφορά σε αυτό στον τρισδιάστατο χώρο και το πόσο πολύπλοκο είναι το νέο αποτέλεσμα, απαιτούνται από μερικά δευτερόλεπτα έως και ώρες.

Η Epic, με το εργαλείο Unreal Engine Tools επιτρέπει τη μεταγλώττιση των αρχείων εκτός μηχανής και είναι απαραίτητο για έργα που αναπτύσσονται σε C++. Με κατάλληλες ρυθμίσεις στο αρχείο BuildConfiguration.xml που παρέχεται μαζί με το εργαλείο είναι δυνατόν να δοθούν ορίσματα στον μεταγλωττιστή και να επιταχυνθεί κατά αρκετές μονάδες η επεξεργασία και κατασκευή των εκτελέσιμων [36]. Συγκεκριμένα, για αλλαγές στο σώμα των πηγαίων (cpp) απαιτούνται περίπου 15 δευτερόλεπτα και για αλλαγές στις κεφαλίδες των αρχείων περίπου 1 λεπτό. Για κατασκευή ολόκληρου του έργου (Build Solution, Rebuild) απαιτούνται 3 λεπτά.

2.4.2 Ροή πληροφορίας

Η μηχανή Unreal Engine λειτουργεί με έντονο το στοιχείο της αντικειμενοστρέφειας. Η κάθε λειτουργία είναι προϊόν κλάσης, ενώ τα αντικείμενα, υλικά και εφέ είναι συνήθως προϊόντα κληρονομικότητας από βασικές κλάσεις. Έτσι υπάρχει μια ιεραρχία και η υλοποίηση θεωρείται απόλυτα δομημένη. Επιπλέον περιλαμβάνει συλλογές αντικείμενων και απλές δομές δεδομένων όταν πρόκειται να περιγράψει κάτι που έχει μόνο μεταβλητές. Τα διαδραστικά αντικείμενα είναι προϊόντα κληρονομικότητας και αποτελούν συλλογές αντικειμένων γιατί πολλές από τις παραμέτρους τους είναι άλλα αντικείμενα με ανεξάρτητες κάθε φορά λειτουργίες αλλά έχοντας πάντα ως σημείο αναφοράς το βασικό αντικείμενο. Οι ήχοι και τα κλιπ βίντεο υποστηρίζονται εντός της μηχανής σαν δυαδικές ακολουθίες ενώ οι κινήσεις αποθηκεύονται σαν κείμενο με μεταβολές στο χώρο συναρτήσει του χρόνου.

2.4.3 Χειρισμός μηχανής

Η μηχανή διαθέτει τρεις καταστάσεις λειτουργίας, τις Editor, Play και Cinematic. Κατά το τρέξιμο της μηχανής σαν Editor δίνεται πρόσβαση σε όλα τα πεδία κώδικα που είναι φανερά στα Blueprint κάθε στοιχείου και η δυνατότητα να φαίνονται όλα τα αντικείμενα στον χώρο, ακόμα και τα κρυφά που χρησιμοποιούνται μόνο σαν σηματοδότες (markers). Ο χρήστης χειρίζεται την κάμερα του Editor, μπορεί να πλοηγηθεί ελεύθερα στους χάρτες του έργου και να αλλάξει οτιδήποτε εντός του έργου. Στην κατάσταση Play υπάρχουν δύο τρόποι εκτέλεσης για το εκάστοτε έργο, η λειτουργία εκτέλεσης εντός μηχανής (Play in Editor, PIE) και η λειτουργία προσομοίωσης (Simulate in Editor, SIE). Στην πρώτη περίπτωση, στην λειτουργία εκτέλεσης, ο κατασκευαστής κάνει χρήση του χαρακτήρα που έχει δηλωθεί σαν παίκτης (Player Pawn) και χρησιμοποιεί την κάμερα που διαθέτει ο χαρακτήρας. Αν δεν υπάρχει τέτοιος χαρακτήρας, τότε χρησιμοποιείται μια προκαθορισμένη κάμερα που μπορεί να κινηθεί στον χώρο ελεύθερα. Υπάρχει επίσης μια σχετική ρύθμιση που οριοθετεί την αρχική θέση του χαρακτήρα παίκτη. Η μηχανή σε κατάσταση Play ανταποκρίνεται μόνο σε εισόδους για τις οποίες έχουν γραφεί πληροφορίες και συναρτήσεις όπως επίσης και στην τίλντα για πρόσβαση σε εντολές κονσόλας. Στην δεύτερη περίπτωση, δηλαδή στην προσομοίωση, χρησιμοποιεί την κάμερα του Editor παραθύρου για να κινείται στον χώρο αλλά η σκηνή εκτελεί οτιδήποτε θα έτρεχε σε πραγματικό χρόνο, δηλαδή χρονομετρητές, εφέ, κινήσεις και δυνάμεις φυσικής. Έτσι μπορεί ο χρήστης να έχει μια εικόνα του τελικού αποτελέσματος πραγματικού χρόνου άμεσα καθώς προσθέτει υλικά στο χώρο.

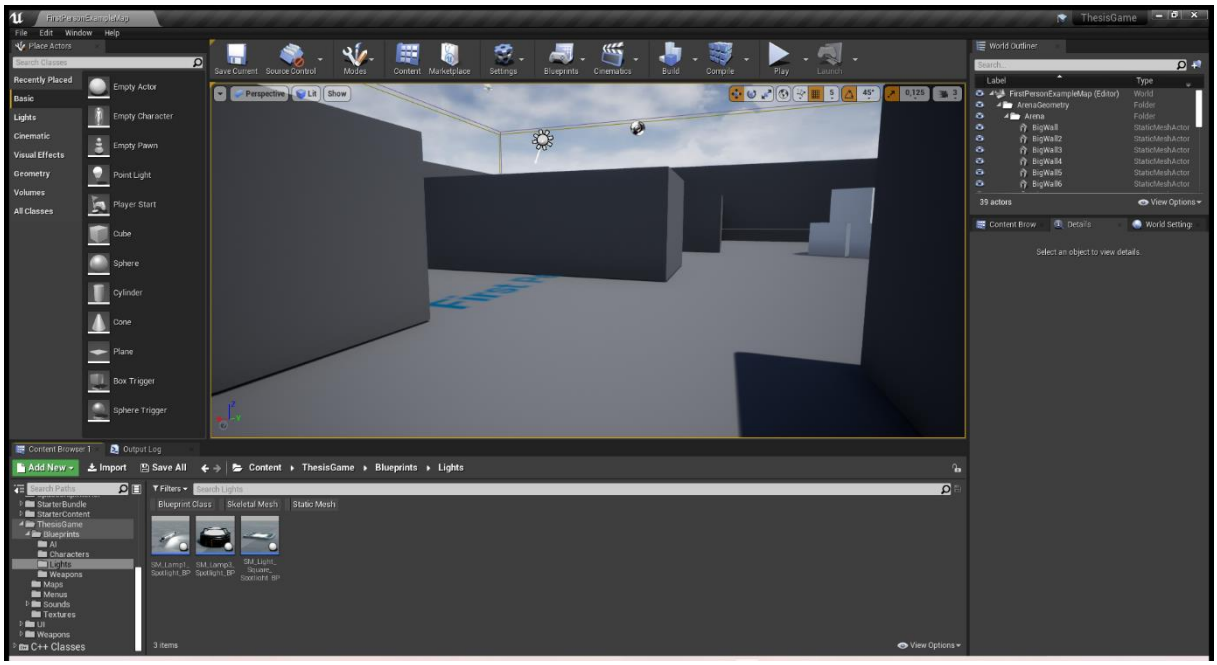
Έξι τρόποι εκτέλεσης (Play) είναι διαθέσιμοι από την μηχανή και είναι οι ακόλουθοι:

- Εκτέλεση στο επιλεγμένο παράθυρο προβολής (Selected Viewport)
- Εκτέλεση με ρυθμίσεις για κινητές συσκευές (Mobile Preview)
- Εκτέλεση σε ανεξάρτητο παράθυρο (New Editor Window)
- Εκτέλεση με σύστημα εικονικής πραγματικότητας (VR Preview)
- Εκτέλεση σαν ανεξάρτητο εκτελέσιμο αρχείο (Standalone Game)
- Προσομοίωση (Simulate)

Η προκαθορισμένη προβολή του χώρου είναι σε μορφή Editor, δηλαδή με όλα τα στοιχεία στο χώρο φανερά αλλά ο χρήστης μπορεί να μεταβεί σε μια προσομοίωση προβολής τύπου Game και Cinematic οποτεδήποτε καθώς χειρίζεται την μηχανή σε κατάσταση Editor. Βέβαια, η μηχανή χρησιμοποιεί υποχρεωτικά αυτές τις προβολές όταν εκτελείται μια εφαρμογή που είναι πακεταρισμένη σε τελικό εκτελέσιμο ενώ όταν γίνονται δοκιμές που εκτελούνται σαν ανεξάρτητο εκτελέσιμο, χρησιμοποιείται αναλόγως με τις ρυθμίσεις η λειτουργία Game και Cinematic. Στην Cinematic λειτουργία εντός του Editor, η εφαρμογή προσομοιώνει οτιδήποτε είναι να παρουσιαστεί σαν σειρά γεγονότων το οποίο προβάλλεται μέσα από στατικές ή κινούμενες κάμερες, με σειρά εναλλαγής που έχει ορίσει ο σχεδιαστής του κινηματογραφικού υλικού [37].

2.5 Περιβάλλον

Το έργο φορτώνεται στην μηχανή και παρατηρούμε ότι το περιβάλλον της μηχανής είναι χωρισμένο σε πέντε βασικές ενότητες σύμφωνα με την λειτουργικότητα τους.

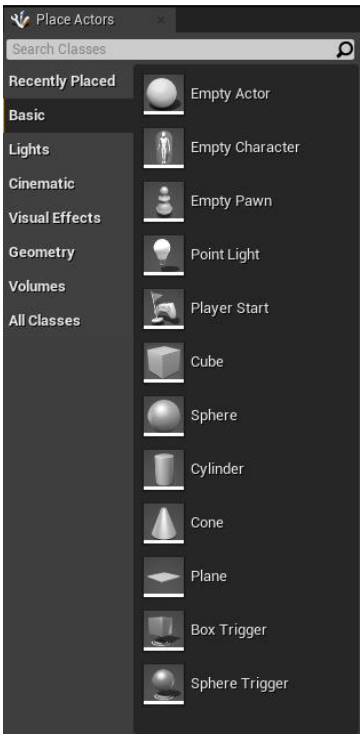


Εικόνα 2.1: Περιβάλλον μηχανής Unreal Engine

Αριστερά βλέπουμε ένα πλαϊνό μενού με αντικείμενα, το λεγόμενο Place Actors. Στο κέντρο βρίσκεται το παράθυρο προβολής τρισδιάστατων (Viewport) και στην άνω δεξιά και αριστερά γωνία υπάρχουν κουμπιά για ρυθμίσεις χωρισμένα σε κατηγορίες. Κάτω από το κέντρο βρίσκεται ένα παράθυρο διαχειριστή αρχείων που έχει πάντοτε ενεργό τον κατάλογο Content, ο οποίος δείχνει εντός του καταλόγου με όνομα το όνομα που έχει ορίσει ο κατασκευαστής για το έργο. Αυτός ο διαχειριστής ονομάζεται Content Browser. Τέλος στα δεξιά έχουμε δύο πλαϊνά πάνελ, το πάνω πάνελ είναι ο περιηγητής του ενεργού τρισδιάστατου χώρου που έχουμε στο Viewport (World Outliner) και το κάτω πάνελ είναι χωρισμένο σε δύο καρτέλες. Στην πρώτη καρτέλα βρίσκεται ο περιηγητής των ιδιοτήτων των αντικειμένων που έχουμε επιλέξει στον τρισδιάστατο χώρο (Details) ενώ στη δεύτερη καρτέλα βρίσκονται οι ρυθμίσεις του ενεργού χάρτη (World Settings) [37].

2.6 Παράθυρο τοποθέτησης αντικειμένων (Place actors)

Πρόκειται για ένα παράθυρο με συντομεύσεις σε συχνά χρησιμοποιούμενα αντικείμενα τύπου Actor, δηλαδή χαρακτήρες, πιόνια, φώτα, οποιοδήποτε αντικείμενο μπορεί να αλληλοεπιδρά με τον παίκτη και τον τρισδιάστατο χώρο [37].



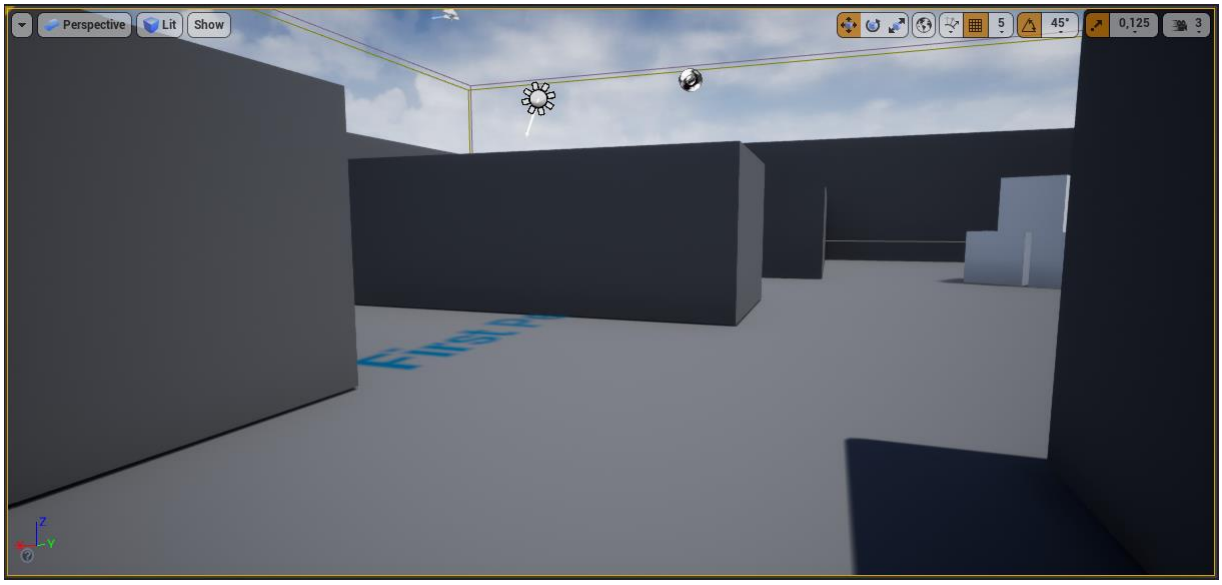
Εικόνα 2.2: Place Actors

Το παράθυρο χωρίζεται σε κατηγορίες με μια κάθετη λίστα επιλογών και διαθέτει μια μπάρα αναζήτησης στη κορυφή. Η λίστα επιλογών περιέχει:

- Προσεχώς τοποθετημένα (Recently placed), δηλαδή αντικείμενα που ο χρήστης τοποθετεί συχνά στο χώρο
- Βασικά αντικείμενα (Basic), αντικείμενα κατασκευασμένα από γεωμετρικά σχήματα και κάποιοι ειδικοί σηματοδότες όπως το αντικείμενο έναρξης του χαρακτήρα (Player Start Actor)
- Φώτα (Lights), όλα τα αντικείμενα που φωτίζουν ή σκιάζουν τον χώρο
- Οπτικά εφέ (Visual Effects), αντικείμενα με συναρτήσεις γραφικών για την παραγωγή κάποιου εφέ, τα πιο συνηθισμένα είναι η φωτιά, ο ηλεκτρισμός και ο καπνός.
- Αντικείμενα γεωμετρίας (Geometry), βασικά μοντέλα γεωμετρίας που βοηθούν στη κατασκευή προτύπων
- Αντικείμενα πυροδότες με βάση τον όγκο τους (Volumes), δηλαδή γεωμετρικά σχήματα τα οποία προκαλούν μια ενέργεια για όσο ένας χαρακτήρας βρίσκεται μέσα σε αυτά
- Όλα τα αντικείμενα (All Classes), όπου φαίνονται όλα τα διαθέσιμα προς τοποθέτηση αντικείμενα

2.7 Παράθυρο προβολής (Viewport)

Το παράθυρο προβολής περιλαμβάνει τον τρισδιάστατο χώρο στον οποίο γίνεται η κατασκευή του περιβάλλοντος ενός έργου καθώς και διάφορες ρυθμίσεις προβολής ή πλοήγησης.



Εικόνα 2.3: Viewport

Αν επιλέξει ο χρήστης κάποιο προσχέδιο έργο κατά την αρχικοποίηση του δικού του έργου, τότε το παράθυρο προβολής θα έχει ενεργό και φορτωμένο το μοναδικό αντικείμενο χώρου, ένα αντικείμενο τύπου Level με κάποιες βασικές ρυθμίσεις φωτισμού και μερικά γεωμετρικά σχήματα που αποτελούν

τον υποτιθέμενο χώρο του βιντεοπαιχνιδιού. Οι καινούργιοι χάρτες δεν έχουν φως και τρισδιάστατη γεωμετρία, πρέπει να προστεθεί. Ο χώρος που δίνεται μαζί με το προσχέδιο μπορεί να επεκταθεί και να χρησιμοποιηθεί για οποιαδήποτε ανάγκη του έργου [37].

Αν δημιουργηθεί και φορτωθεί ένας νέος χάρτης, το μόνο που φαίνεται είναι ένα πλέγμα (grid). Είναι δυνατό να προστεθεί υλικό μέσα στο νέο τρισδιάστατο χώρο είτε επιλέγοντας μια κλάση από το αριστερό μενού κλάσεων είτε ένα αντικείμενο από το κάτω μενού, τον διαχειριστή αρχείων. Όλα τα αντικείμενα μπορούν να τοποθετηθούν στον χώρο με μέθοδο drag ‘n drop. Για την τροποποίηση τους μέσα στον χώρο υπάρχει ένα κανονικό σύστημα αξόνων, το Gizmo. Οι άξονες είναι κουμπιά και κρατώντας τους πατημένους το αντικείμενο μπορεί να μετακινηθεί προς κάθε άξονα. Ενδιάμεσα έχουν ένα τετράγωνο κουμπί που μετακινεί το αντικείμενο κατά δύο άξονες κάθε φορά. Στο κέντρο του Gizmo υπάρχει ένας κύβος που κινεί το αντικείμενο ελεύθερα, προς όλους τους άξονες. Το Gizmo εικονίζεται με βέλη για μετακίνηση, καμπύλες για περιστροφή και τρίγωνο για μεγέθυνση/σμίκρυνση. Με την βοήθεια τριών κουμπιών στην άνω δεξιά γωνία του παραθύρου προβολής, ο χρήστης μπορεί να επιλέξει μετακίνηση του αντικειμένου κατά 1 ή περισσότερες μονάδες. Χρησιμοποιείται για απόλυτο ταίριασμα μεταξύ διαφορετικών επιφανειών, όταν έχουν μοντελοποιηθεί με μια έως και τις τέσσερις πλευρές τους να βρίσκονται σε ζυγές συντεταγμένες. Η τεχνική αυτή για την κατασκευή περιβάλλοντος ονομάζεται Object Snapping. Όλα τα αντικείμενα τοποθετούνται σε συντεταγμένες X,Y,Z στον τρισδιάστατο χώρο και αναφέρονται μόνο σε ένα αρχείο χώρου. Η πληροφορία αυτή όπως και άλλες ιδιότητες του κάθε αντικειμένου φαίνεται στο δεξιό κάτω μενού, το Details. Υπάρχει δυνατότητα ένα αντικείμενο να μετακινηθεί και να περιστραφεί σύμφωνα με το σύστημα αξόνων του χώρου (global), διαφορετικά σύμφωνα με την εκάστοτε περιστροφή του (local) ή σύμφωνα με τις εφαπτόμενες ευθείες που διαπερνούν το κέντρο του αντικειμένου όταν αυτό βρίσκεται σε δεδομένη περιστροφή (normal).

Πολλά αντικείμενα μπορούν να ομαδοποιηθούν έτσι ώστε να επιλέγονται και να μετακινούνται σαν ένα. Αν είναι της ίδιας κλάσης ή μοιράζονται ιδιότητες της ίδιας κλάσης, αυτές μπορούν να τροποποιούνται μαζικά καθώς είναι επιλεγμένη η ομάδα. Υπάρχει επίσης η δυνατότητα πολλαπλής επιλογής αντικειμένων και τροποποίησης τους χωρίς ομάδα. Επιπλέον, ισχύει η διαγραφή, αντιγραφή και αποκοπή ενός ή περισσοτέρων αντικειμένων έτσι ώστε ένα περιβάλλον που στήθηκε σε έναν χώρο να μπορεί να αντιγραφεί σε έναν καινούργιο ή ακόμα και να επαναληφθεί σε άλλο σημείο του χώρου. Μια λειτουργία που είναι αρκετά χρήσιμη κατά την κατασκευή περιβάλλοντος είναι το καθρέφτισμα αντικειμένων, θέτεται αυτόματα το μέγεθος των αντικειμένων στον άξονα X ή Y ίσο με -1, έτσι ώστε όλες οι κορυφές του τρισδιάστατου μοντέλου να βρίσκονται στην αντίθετη πλευρά από την αρχική τους. Με αυτήν την λειτουργία δημιουργούνται συμμετρικοί χώροι πολύ γρήγορα καθώς ο χρήστης χρειάζεται να φτιάξει μόνο τη μία πλευρά και το κέντρο του χώρου.

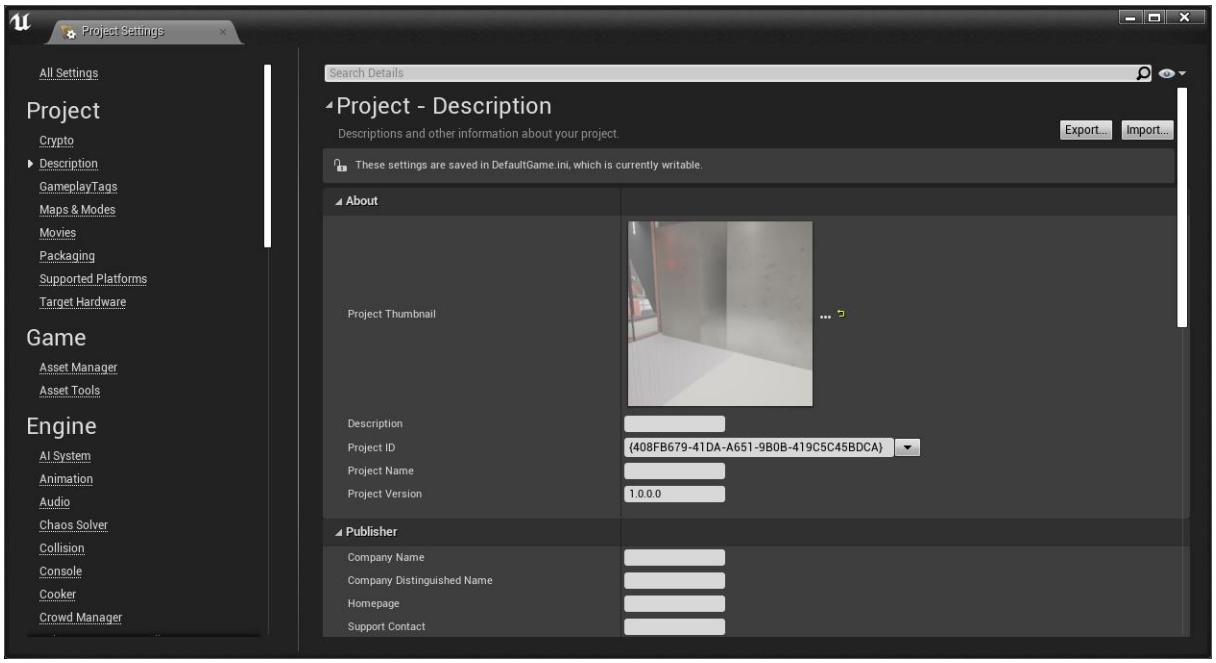
Όλα τα αντικείμενα τοποθετούνται στον χώρο είτε με την επιλογή Add New πάνω από το παράθυρο προβολής είτε από το πλαϊνό αριστερό μενού που δίνει γρήγορη πρόσβαση σε συχνά χρησιμοποιούμενα αντικείμενα και κατόπιν ο χρήστης μπορεί να αλλάξει το μέγεθος και την περιστροφή τους εντός του χώρου, όπως επίσης και το σχήμα τους, κοινώς το αρχείο τρισδιάστατου μοντέλου που έχουν φορτωμένο. Οι αλλαγές ισχύουν για κάθε αντικείμενο ξεχωριστά καθώς αποτελούν ανεξάρτητα χωρία στα οποία μπορεί να αναφερθεί μεμονωμένα (instances) ο χρήστης. Αν αλλάξει το βασικό αντικείμενο, αλλάξουν όλες οι αναφορές σε αυτό. Αν θελήσει ο χρήστης να διαγράψει το βασικό αντικείμενο, δίνεται από τη μηχανή η επιλογή να αντικατασταθούν όλες οι αναφορές με κάποιο άλλο βασικό αντικείμενο ή να αφαιρεθούν από τη μνήμη και τον σκληρό δίσκο. Η δεύτερη επιλογή είναι εξαιρετικά ασταθής, ιδιαίτερα για αναφορές σε αντικείμενα που αλλάζουν σε πραγματικό χρόνο και καταλαμβάνουν πολλή προσωρινή μνήμη για να λειτουργήσουν. Τέτοια αντικείμενα είναι διάφορα εφέ και μοντέλα προσομοίωσης επιφάνειας νερού. Κάποιες φορές κατά την διάρκεια υλοποίησης της εργασίας

παρατηρήθηκαν παγώματα της μηχανής μετά από διαγραφή αρχείων κίνησης χαρακτήρων που ήταν συνδεδεμένα με κάποιον χαρακτήρα και είχε αναφορά στον χώρο.

Δύο τρόποι απεικόνισης ως προς την οπτική γωνία είναι διαθέσιμοι σε αυτό το παράθυρο. Η προοπτική (perspective) όπου ο χώρος απεικονίζεται τρισδιάστατος με βάθος όπως το αντιλαμβανόμαστε με την όραση μας και η ορθογραφική προβολή, στην οποία ο χώρος φαίνεται πάντοτε ως προς δύο διαστάσεις. Επιπλέον η μηχανή διαθέτει διάφορους τρόπους απεικόνισης του χώρου ως προς το φωτισμό και τα τρισδιάστατα αντικείμενα. Οι δύο βασικές επιλογές είναι το φωτισμένο ή μη περιβάλλον (Lit, Unlit). Η πρώτη επιλογή λαμβάνει υπ' όψη όλα τα φίλτρα φωτισμού και εφέ και απεικονίζει τον χώρο όπως θα εμφανίζεται σε κανονικές συνθήκες εντός του βιντεοπαιχνιδιού. Η δεύτερη επιλογή δείχνει τον χώρο με ενεργό μόνο το χρώμα του κάθε αντικειμένου. Έπειτα η επιλογή περιγράμματος (wireframe) απεικονίζει τα τρισδιάστατα αντικείμενα ως κορυφές και ακμές. Η επιλογή λεπτομέρειας φωτισμού (Detail Lighting) αφήνει ενεργό μόνο το φίλτρο λεπτομέρειας στα τρισδιάστατα, που είναι συνδυασμός ενός γκρι χρώματος και του γραφικού λεπτομέρειας (normal map). Η επιλογή απομόνωσης φωτισμού (Lighting Only), φωτίζει τα αντικείμενα χωρίς χρώμα (diffuse map) ή λεπτομέρεια (normal map). Τέλος η επιλογή απομόνωσης αντανακλάσεων δείχνει τα αντικείμενα σαν συνδυασμό των επιμέρους αντανακλάσεων τους. Πέρα από τις βασικές επιλογές υπάρχουν και διάφορες επιλογές αποσφαλμάτωσης του χώρου, όπως η προβολή πολυπλοκότητας φωτισμού (Light Complexity), σκιών (Shader Complexity), επικάλυψη φωτός (Stationary Light Overlap) και η πυκνότητα φωτός (Lightmap Density). Επιπλέον διαθέτει προβολή του χρωματικού δείκτη των διαφορετικών στρωμάτων μακρινού χώρου (LOD Coloration) [37].

2.8 Παράθυρο ρυθμίσεων εφαρμογής (Application Settings)

Σε κάθε έργο ο χρήστης ορίζει, μέσω των ρυθμίσεων του έργου (Project Settings) ποιος χώρος θα φορτώνεται πρώτος κατά την εκκίνηση της εφαρμογής. Σε αυτές τις ρυθμίσεις περιλαμβάνονται πάρα πολλά τεχνικά θέματα του έργου και είναι χωρισμένα σε κατηγορίες ενώ δίνεται και η δυνατότητα αναζήτησης με λέξεις κλειδιά. Το παράθυρο εμφανίζεται ακολουθώντας τη διαδρομή Edit➡Project Settings [38].



Εικόνα 2.4: Project settings

2.8.1 Ρυθμίσεις - Project

Αφορά όλες τις ρυθμίσεις που αναφέρονται στο έργο, περιγραφή, λέξεις κλειδιά, βασικές ρυθμίσεις για τον προκαθορισμένο χάρτη που φορτώνεται, την κατάσταση του παιχνιδιού και τον προκαθορισμένο χαρακτήρα που χειρίζεται ο παίκτης, ρυθμίσεις για τα κινηματογραφημένα κλιπ που μπορεί να περιλαμβάνει (cut-scenes), σε ποιες πλατφόρμες υποστηρίζεται η εφαρμογή και ποιες προδιαγραφές πρέπει να έχει η πλατφόρμα για να εκτελεστεί σωστά. Οι πιο σημαντικές ρυθμίσεις είναι αυτές της περιγραφής και των χαρτών.

Η περιγραφή έχει να κάνει με ότι αναφέρεται στο έργο, τον τίτλο, τα σχόλια, το μοναδικό αναγνωριστικό, την μικρογραφία και την έκδοση του. Επιπλέον εδώ σημειώνεται ποιος είναι ο δημιουργός, τι πνευματικά δικαιώματα συνοδεύουν την εφαρμογή, ποιος είναι ο τίτλος που φαίνεται στο εκτελέσιμο αρχείο και διάφορες ρυθμίσεις του εκτελέσιμου, όπως η δυνατότητα να ελαχιστοποιηθεί το παράθυρο της εφαρμογής.

Οι χάρτες έχουν ρυθμίσεις για το βασικό φωτισμό όλου του βιντεοπαιχνιδιού, προκαθορισμένες κλάσεις καθολικές στο εκτελέσιμο όπως το GameMode, πρόγραμμα που αποθηκεύει καθολικές μεταβλητές του παιχνιδιού, πχ ένα σκορ ή τις ομάδες-παρατάξεις των παικτών. Το Default Pawn που καθορίζει τη προεπιλεγμένη κλάση του χαρακτήρα που χειρίζεται ο παίκτης, το Player Controller το οποίο είναι υπεύθυνο για τον υπολογισμό της κατεύθυνσης και κίνησης του χαρακτήρα στον τρισδιάστατο χώρο, το Game State και Player State, αποθηκεύει μεταβλητές κατάστασης όπως το αν ένας παίκτης θεωρείται νεκρός για αυτή τη παρτίδα ή αν έχουν παιχτεί δύο γύροι για μία παρτίδα. Τέλος το HUD class, Heads Up Display, που καθορίζει την προεπιλεγμένη κλάση για χειρισμό των μενού που βλέπει ο παίκτης στην οθόνη του. Αυτά μπορεί να είναι μια πυξίδα, μια μικρογραφία χάρτη, η ζωή και τα πράγματα του χαρακτήρα κα.

2.8.2 Ρυθμίσεις - Game

Περιλαμβάνει ρυθμίσεις για το πως θα φορτώνονται τα διάφορα αρχεία από το συγκεκριμένο έργο, σε ποιους καταλόγους να γίνει αναζήτηση για αρχεία και το ποια κλάση θα χρησιμοποιηθεί για τα πρωτεύοντα τύπου αρχείων. Συνίσταται να χρησιμοποιούνται τα προκαθορισμένα εκτός πολύ ειδικών περιπτώσεων.

2.8.3 Ρυθμίσεις - Engine

Εδώ ρυθμίζεται η συμπεριφορά της μηχανής ανάλογα με το πρόβλημα που επιλύει. Χωρίζεται σε πολλές υποκατηγορίες που έχουν να κάνουν πχ με τεχνητή νοημοσύνη, συμπεριφορά του τερματικού της μηχανής, το πως γίνεται η είσοδος από ένα φυσικό μέσο, παράμετροι δικτύου, προσομοίωσης φυσικής, ρυθμίσεις για τη διεπαφή χρήστη κα. Επίσης συνίσταται η χρήση των προκαθορισμένων, αλλά κατά την κατασκευή μεθόδων χειρισμού του παιχνιδιού θα γίνεται πολύ συχνά τροποποίηση των παραμέτρων εισόδου. Επίσης, κατά την κατασκευή τεχνητής νοημοσύνης θα πρέπει να αλλάξουν αρκετές μεταβλητές για να τελειοποιηθεί η συμπεριφορά των χαρακτήρων που χειρίζεται ο υπολογιστής.

2.8.4 Ρυθμίσεις - Editor

Αφορά επιλογές του παραθύρου της εφαρμογής της μηχανής και έχει να κάνει με το πως εμφανίζονται ή λειτουργούν κάποια συστατικά της μηχανής, όπως επίσης και πόσο απλοποιούνται τα αντικείμενα που εισάγονται στη μηχανή.

2.8.5 Ρυθμίσεις - Platform

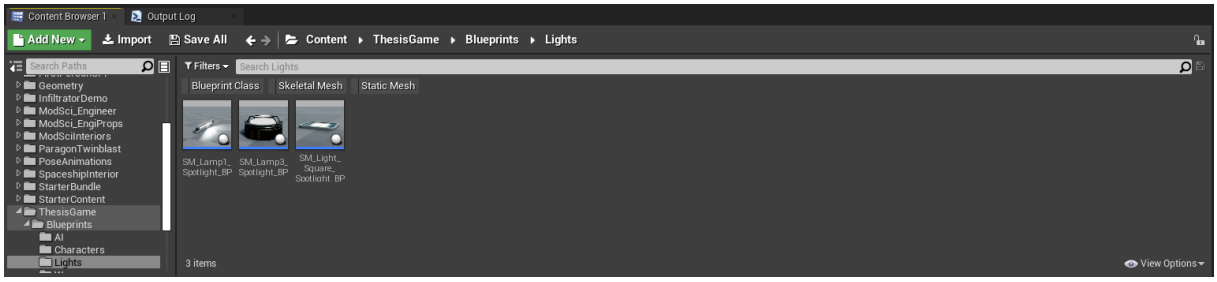
Εδώ γίνεται η παραμετροποίηση ανά πλατφόρμα. Περιέχονται πολύ ειδικές τεχνικές λεπτομέρειες της εφαρμογής. Ανάλογα με το τι περιέχει το τελικό βιντεοπαιχνίδι, επιλέγονται κάποιες εκδόσεις της διεπαφής γραφικών, όπως αν απαιτείται διεπαφή DirectX έκδοσης 11 και 12, αν η εφαρμογή περιέχει συναρτήσεις του Shader Model 5, αν απαιτεί προσομοίωση DirectX σε κινητά. Επίσης, για Windows, Linux και Mac υπάρχουν διάφορες ρυθμίσεις για την εφαρμογή, όπως τι εικονίδιο θα έχει, ποια θα είναι η κατώτατη έκδοση λειτουργικού, τι εικόνα, βίντεο ή κείμενο θα προβάλλει κατά την εκκίνηση, τι επεκτάσεις ήχου και τι μέγεθος μνήμης θα έχει η μονάδα ήχου της εφαρμογής.

2.8.6 Ρυθμίσεις - Plugins

Εδώ βρίσκονται οι ρυθμίσεις για διάφορα επεκτάσιμα του Unreal Engine τα οποία δύναται να φορτωθούν στην εφαρμογή όπως το Niagara, βοηθητικό για κατασκευή πολύπλοκων σωμάτων με εφέ και σωματίδια πάνω σε σκελετό (Particle Systems).

2.9 Παράθυρο περιηγητή αρχείων (Content Browser)

Το παράθυρο περιηγητή αρχείων είναι ένα ενσωματωμένο σύστημα διαχείρισης των αρχείων που χρησιμοποιούνται από την μηχανή.



Εικόνα 2.5: Content Browser

Η προσθήκη οποιουδήποτε αντικειμένου στην μηχανή γίνεται με το πράσινο κουμπί Add New που βρίσκεται στον περιηγητή αρχείων του έργου, δηλαδή τον Content Browser. Ο χρήστης επιλέγει την κατηγορία του αντικειμένου που χρειάζεται, έπειτα του δίνει όνομα και κάποιες ρυθμίσεις αν το κρίνει απαραίτητο. Συνήθως όμως οι προκαθορισμένες ρυθμίσεις αρκούν. Όταν δημιουργείται το νέο αντικείμενο γίνεται ενεργό στο παράθυρο αυτό και στο όνομα του περιλαμβάνεται ένας αστερίσκος που σηματοδοτεί ότι το αντικείμενο αυτό είναι φορτωμένο αλλά δεν έχει αποθηκευτεί στον σκληρό δίσκο. Ότι ισχύει σε έναν περιηγητή των Windows ισχύει και σε αυτό το παράθυρο όσον αφορά τις λειτουργείες που εκτελούνται μέσω της διεπαφής της μηχανής. Ο χρήστης μπορεί να δημιουργήσει νέους καταλόγους σε ιεραρχία και να μετακινήσει αρχεία μεταξύ αυτών. Μπορεί επίσης να διαγράψει και να τροποποιήσει οποιοδήποτε αρχείο. Το μόνο που διαφοροποιείται είναι κάποιες συντομεύσεις, για παράδειγμα το Ctrl + Shift + N δεν δημιουργεί νέο κατάλογο στο περιηγητή αλλά προσπαθεί να δημιουργήσει νέο αρχείο έργου στη μηχανή. Αυτό συμβαίνει γιατί ο περιηγητής δεν είναι ανεξάρτητος από τη μηχανή αλλά ένα εργαλείο της μηχανής.

Το μενού περιηγητή αρχείων διαθέτει επίσης την δυνατότητα να εισάγει φίλτρα αναζήτησης στα αρχεία για να διευκολυνθεί ο χρήστης όταν εισάγει περιεχόμενο από έναν κατάλογο με πάρα πολλά αρχεία ή όταν τα αντικείμενα που χρειάζεται βρίσκονται σε διάφορους υποφακέλους. Τα φίλτρα βρίσκονται σαν drop down μενού στα δεξιά της μπάρας αναζήτησης του περιηγητή και περιλαμβάνουν πολλές επιλογές

για κάθε κατηγορία αρχείου. Τσεκάροντας μια κατηγορία εμφανίζεται πάνω από το πλαίσιο του περιηγητή και επιλέγοντας την φωτίζεται και αναζητά αντικείμενα μόνο αυτής της κατηγορίας.

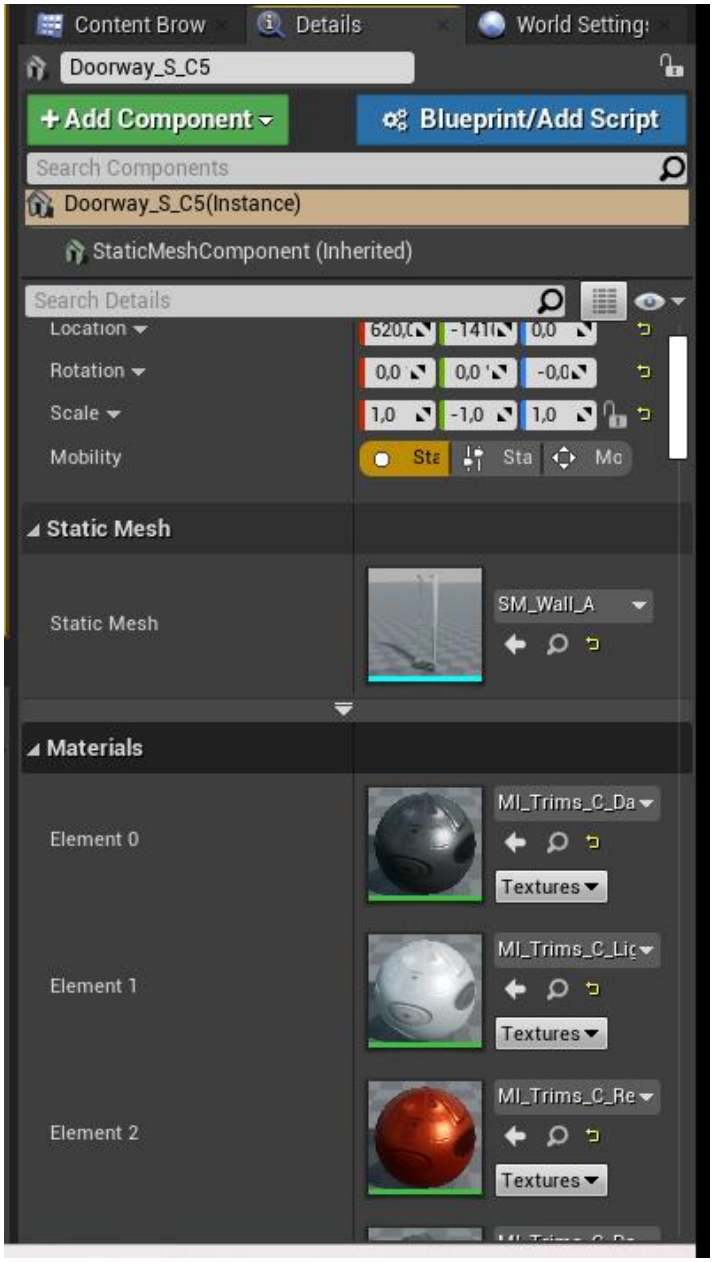
Όταν ο χρήστης επιθυμεί να μετακινήσει αντικείμενα πηγής σε άλλο κατάλογο, να τα διαγράψει ή να τα αντικαταστήσει με νεότερες εκδόσεις τους, η χρήση του περιηγητή δίνει το πιο επιθυμητό αποτέλεσμα καθώς η μηχανή ενημερώνεται σωστά για τις αλλαγές που γίνονται στον περιηγητή. Αυτό δεν συμβαίνει όμως όταν γίνονται εξωτερικές αλλαγές στο σύστημα αρχείων του έργου, όπως αλλαγές με το χέρι στον περιηγητή των windows.

Όλα τα αρχεία εντός της μηχανής αφού χρησιμοποιηθούν αποκτούν μια αναφορά και προστίθενται σε έναν πίνακα αναφορών που χρησιμοποιούνται ή χρησιμοποιούν αυτό το αντικείμενο. Όταν γίνεται μια τροποποίηση στο αρχικό αρχείο, αν αλλάξει θέση στο δίσκο ή όνομα, τότε γίνεται ενημέρωση στον πίνακα αναφορών του. Αν η αλλαγή είναι από εξωτερικό παράγοντα τότε δεν θα πυροδοτηθεί η ενημέρωση και ο πίνακας αυτός θα μείνει ελλιπής καθώς δεν μπορεί να βρει ένα ή περισσότερα αρχεία όπως αυτά αναγράφονται. Σε πολύπλοκες δομές όπου το αλλαγμένο αρχείο είναι άρρηκτα δεμένο με πολλά άλλα και είναι ενεργό στον τρισδιάστατο χώρο σε κάποιο χάρτη, υπάρχει περίπτωση να προκληθεί σφάλμα στη μηχανή και να πρέπει να φορτωθεί προηγούμενη έκδοση του έργου [37].

2.10 Παράθυρο περιηγητή χώρου (World Outliner)

Το παράθυρο περιηγητή χώρου περιλαμβάνει μια λίστα με όλα τα αντικείμενα που έχουν τοποθετηθεί σε ένα χάρτη και δείχνει πάντοτε στον ενεργό χάρτη που έχουμε φορτωμένο στο παράθυρο προβολής. Για κάθε αντικείμενο φαίνεται η ορατότητα του στον χώρο, το όνομα του και ο τύπος του. Κατά την επιλογή ενός αντικειμένου στο παράθυρο προβολής γίνεται επιλογή του ίδιου αντικειμένου στον περιηγητή χώρου. Για την καλύτερη οργάνωση των αντικειμένων ενός χώρου μπορούν να δημιουργηθούν φάκελοι οι οποίοι περιλαμβάνουν αντικείμενα του χώρου. Το όνομα του φακέλου δεν επηρεάζει τίποτα στον χώρο αλλά η ορατότητα του φακέλου κρύβει ή εμφανίζει τα αντικείμενα στο παράθυρο προβολής. Είναι πολύ χρήσιμο ειδικά όταν γίνεται επεξεργασία μεγάλων χώρων με πάρα πολλά διακοσμητικά και φώτα. Γίνεται επίσης ταξινόμηση κατά όνομα ή τύπο με αλφαβητική σειρά. Διαθέτει επιπλέον μια μπάρα αναζήτησης για πιο εύκολη εύρεση ενός συγκεκριμένου αντικειμένου αντί να το αναζητούμε στον τρισδιάστατο χώρο. Το παράθυρο αυτό λειτουργεί συνδυαστικά με το παράθυρο λεπτομερειών αντικειμένου, το οποίο δείχνει για κάθε αντικείμενο που έχουμε επιλεγμένο, όλες τις φανερές στη μηχανή, ιδιότητες [37].

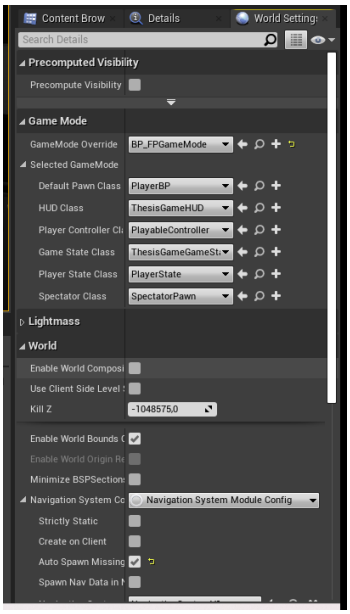
2.11 Παράθυρο λεπτομερειών αντικειμένου (Details)



Εικόνα 2.6: Details

Σε αυτό το παράθυρο γίνεται παραμετροποίηση του κάθε αντικειμένου στον τρισδιάστατο χώρο και όλες οι ρυθμίσεις αναφέρονται στις αρχικές τιμές που δίνονται στον constructor αυτού του αντικειμένου. Για να είναι ορατές σε αυτό το παράθυρο οι μεταβλητές του αντικειμένου, πρέπει να έχουν την ιδιότητα VisibleAnywhere, VisibleDefaultsOnly, EditAnywhere ή EditDefaultsOnly. Ο προσδιορισμός Visible είναι ασύμβατος με τον Edit καθώς όσες μεταβλητές φέρουν τον προσδιορισμό Visible έχουν ορατές τιμές αλλά απαγορεύεται η μετατροπή τους ενώ όσα φέρουν τον Edit έχουν τιμές ορατές και μεταβαλλόμενες από τη μηχανή, ακόμα και αν πρόκειται για σταθερές. Πριν τις μεταβλητές που περιλαμβάνει το αντικείμενο, έχει ορατές και επεξεργάσιμες τις συντεταγμένες του στον τρισδιάστατο χώρο και το μέγεθος και την περιστροφή του. Οπότε η τοποθέτηση στον τρισδιάστατο χώρο γίνεται και μέσω απ' ευθείας αλλαγών στις συντεταγμένες. Για κάθε αναφορά του αντικειμένου, δηλαδή κάθε αντίγραφο του στο χώρο (instance) μπορεί να αλλάξει το υλικό της υφής του (material), οι πληροφορίες συμπεριφοράς του σε δυνάμεις φυσικής (physics), οι πληροφορίες σύγκρουσης (collision), η δυνατότητα του αντικειμένου να παράγει σκιά, καθώς επίσης αν θα είναι κρυφό ή φανερό εντός του βιντεοπαιχνιδιού. (In-Game visibility) [37].

2.12 Παράθυρο ρυθμίσεων χώρου (World Settings)



Αυτό το παράθυρο περιλαμβάνει ρυθμίσεις που αφορούν τον ενεργό χάρτη και είναι χωρισμένες σε κατηγορίες. Υπάρχει επίσης μπάρα αναζήτησης όπως σε όλα τα παράθυρα ρυθμίσεων. Από τις πιο σημαντικές είναι το σημείο στον άξονα Z όπου θεωρείται το τέλος του χάρτη και ενεργοποιεί την συνάρτηση θανάτου σε όποιο αντικείμενο ξεπεράσει αυτό το όριο (Kill Z). Δύο ρυθμίσεις αλλάζουν τον τρόπο με τον οποίο είτε η κάμερα του παίκτη είτε ο κόσμος ο ίδιος απαλείφει γεωμετρία μετά από μια σταθερή απόσταση. Η ρύθμιση GameModeOverride επιτρέπει την επιλογή ενός διαφορετικού τύπου παιχνιδιού όταν ο παίκτης βρίσκεται εντός αυτού του χάρτη. Αυτό σημαίνει πως θα έχει μια άλλη αποστολή, άλλους μετρητές ή θα ισχύουν διαφορετικοί κανόνες. Τέλος διαθέτει βασικές ρυθμίσεις ήχου που αφορούν το βαθμό στον οποίο παράγεται αντίλαλος και βασικούς εξισορροπητές του ήχου για τη μέρα και τη νύχτα εντός του παιχνιδιού [37].

Εικόνα 2.7: World Settings

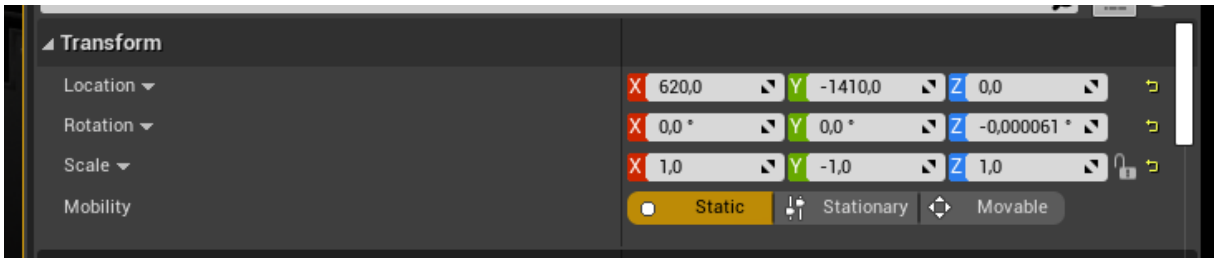
2.13 Αντικείμενα του Unreal Engine (Components)

Η μηχανή απαρτίζεται από αντικείμενα, παράγωγα διαφόρων κλάσεων με συγκεκριμένες ή αόριστες λειτουργίες. Πολλά από αυτά είναι επαναχρησιμοποιήσιμα. Συνήθως έχουν χαρακτηριστικά και ιδιότητες από μια βασική κλάση και μπορούν να επεκταθούν αρκετά αλλά ισχύουν αρκετοί περιορισμοί. Το κάθε αντικείμενο είναι ένα Component στην τεχνική ορολογία του Unreal Engine και κληρονομεί παραμέτρους από τη κλάση γονέα, δηλαδή τη κλάση Actor. Οι πιο συχνά χρησιμοποιούμενες ιδιότητες των αντικειμένων είναι οι ρυθμίσεις της κινητικότητα τους και η συμπεριφορά τους σε πράξεις πρόσκρουσης και φυσικής. Έπειτα χωρίζονται σε μεγάλες κατηγορίες σύμφωνα με την λειτουργικότητα τους ως εξής:

- Στατικά, ακίνητα, όλα τα αντικείμενα που αποτελούν το συμπαγή χώρο του βιντεοπαιχνιδιού
- Φώτα – πηγές φωτός, όλα τα αντικείμενα που μπορούν να αλλάξουν τον φωτισμό των πίξελ σε δεδομένη κλίμακα.
- Πυροδοτητές, όλα τα γεωμετρικά σχήματα που πυροδοτούν γεγονότα όταν ισχύουν συγκεκριμένες συνθήκες που ορίζει ο προγραμματιστής.
- Ειδικά εφέ, δηλαδή τα αντικείμενα ατμόσφαιρας, καιρικών φαινομένων και φυσικών στοιχείων
- Χαρακτήρες, όλες οι έμβιες οντότητες εντός του βιντεοπαιχνιδιού συμπεριλαμβανομένου και του πρωταγωνιστή.
- Αξεσουάρ, ρούχα και οπλισμός, ειδικός τύπος χαρακτήρα που εξαρτάται από τις ενέργειες ενός χαρακτήρα που τα φοράει.
- Οχήματα, επίσης ειδικός τύπος χαρακτήρα τον οποίο μπορεί να χειριστεί ένας χαρακτήρας.

2.13.1 Ιδιότητες αντικειμένων – Κινητικότητα

Υπάρχει μια διαφοροποίηση στα αντικείμενα σύμφωνα με την κινητικότητα τους. Διακρίνουμε τα στατικά (static), τα κινούμενα στατικά (movable static) και τα κινούμενα (movable).



Εικόνα 2.8: Κινητικότητα αντικειμένων

Στα εντελώς στατικά αντικείμενα η θέση τους παραμένει σταθερή καθ' όλη τη διάρκεια εκτέλεσης της εφαρμογής του βιντεοπαιχνιδιού, από την φόρτωση τους στη μνήμη μέχρι την αποθήκευση της κατάστασης τους παιχνιδιού στον δίσκο και το κλείσιμο της εφαρμογής. Μπορούν να τοποθετηθούν με οποιοδήποτε τρόπο στον χώρο αλλά μόνο στο παράθυρο του Editor. Δεν καταλαμβάνουν ιδιαίτερο ποσοστό στην προσωρινή μνήμη καθώς δεν αλλάζει τίποτε επάνω τους. Τέτοια αντικείμενα είναι όλα τα μοντέλα αρχιτεκτονικής, περιβάλλοντος, μακρινού κόσμου (LOD, Land of Distance). Ομοίως τα κινούμενα στατικά, είναι στατικά αντικείμενα στα οποία μπορούν να επενεργήσουν δυνάμεις φυσικής χάρη σε μια κάψουλα που τα περιβάλλει (Physics Asset). Είναι σταθερά και ακίνητα στον χώρο μέχρι να συγκρουστεί κάποιο άλλο αντικείμενο ή χαρακτήρας με αυτά, τότε ενεργοποιείται η κίνηση και μετατοπίζονται κατά ίση δύναμη με την δύναμη πρόσκρουσης μείον την απόσβεση σε χρόνο Delta Seconds, δηλαδή τον χρόνο που πέρασε από την πρόσκρουση. Μετά από λίγο σταματούν και απενεργοποιείται η κίνηση τους, οπότε μεταβαίνουν σε στατική κατάσταση. Σε αυτήν την κατηγορία εντάσσονται και τα εφέ με δυναμικό φωτισμό ή αλλαγή ιδιοτήτων σε πραγματικό χρόνο, είναι στατική η θέση τους αλλά όχι οι ιδιότητες τους. Καταλαμβάνουν ένα σταθερού μήκους χωρίο στη μνήμη για τις δυνάμεις φυσικής και την επιπλέον κινητικότητα τους. Τέλος τα εντελώς κινούμενα αντικείμενα είναι αυτά που είναι δυνατόν να μεταβάλλουν τη θέση τους και τις ιδιότητες τους σε πραγματικό χρόνο χωρίς περιορισμό. Επενεργούν σε αυτά δυνάμεις φυσικής και μπορούν να αλλάξουν τις πληροφορίες φυσικής τους οποτεδήποτε, όπως τη μάζα τους ή το σχήμα της κάψουλας τους. Μπορούν επίσης να δημιουργούνται και να καταστρέφονται δυναμικά κατά την εκτέλεση της εφαρμογής. Τέτοια αντικείμενα είναι όλοι οι χαρακτήρες, οτιδήποτε ζωντανό υπάρχει σε ένα ψηφιακό περιβάλλον. Καταλαμβάνουν μεταβλητού μήκους χωρία στη μνήμη, δυναμικά μεταβαλλόμενα ανάλογα με την εκάστοτε κατάσταση τους. Τα κινούμενα αντικείμενα θεωρούνται ως τα πιο ακριβά σε απαιτήσεις πόρων στοιχεία που αποτελούν έναν ψηφιακό κόσμο και είναι η κύρια αιτία εμφάνισης προβλημάτων απόδοσης της εφαρμογής [39]. Είναι στο χέρι του χρήστη – σχεδιαστή το πως θα χρησιμοποιήσει την κάθε κατηγορία αντικειμένων. Μπορεί να αποφασίσει ότι κάποιοι χαρακτήρες είναι στατικοί οπότε δεν χρειάζονται πλήρη κίνηση στον χώρο, όπως οι θεατές σε ένα βιντεοπαιχνίδι προσομοίωσης αθλημάτων. Ένα άλλο παράδειγμα είναι η απόφαση κάποια αντικείμενα αρχιτεκτονικής να αποτελούν χαρακτήρες! Αυτό γιατί έτσι το αρχιτεκτονικό μοντέλο μπορεί να χτίζεται ή να καταστρέφεται δυναμικά ανάλογα με το τι κάνει ο παίκτης και να διαγράφεται από την μνήμη αφού ο παίκτης περάσει από εκείνη τη τοποθεσία. Ένα μοντέλο οχήματος μπορεί να είναι στατικό αν είναι παρκαρισμένο στον χώρο σαν διακοσμητικό και δεν αλληλοεπιδρούμε με αυτό, αλλά μπορεί να είναι και χαρακτήρας αν ο παίκτης μπορεί να το οδηγήσει.

2.13.2 Ιδιότητες αντικειμένων – Συγκρούσεις

Τα αντικείμενα είναι συνδεδεμένα με μια γεωμετρία πρόσκρουσης (Collider) που τους προσδίδει φυσικά όρια. Όλοι οι τύποι αντικειμένων έχουν την δυνατότητα να δεχτούν οποιοδήποτε τύπο Collider, αρκεί αυτός να δίνει το επιθυμητό αποτέλεσμα. Τα γεγονότα πρόσκρουσης λαμβάνονται υπ' όψη από

δύο αντικείμενα που έχουν σωστή γεωμετρία πρόσκρουσης (Collision Mesh) και ταυτόχρονα ενεργοποιημένη την επιλογή να αποτρέπουν την επικάλυψη (overlap). Αν ένα αντικείμενο δεν εμποδίζει την επικάλυψη από ένα άλλο, τότε το δεύτερο μπορεί να αγνοήσει την ύπαρξη ορίων και να διαπεράσει το πρώτο. Ακόμα και με πλήρη περιγραφή των ιδιοτήτων πρόσκρουσης, αν αυτή συμβεί σε χρόνο μικρότερο από ένα πλαίσιο δεν γίνεται αντιληπτή από τη μηχανή και αγνοούνται πλήρως οι κανόνες φυσικής. Το ίδιο συμβαίνει όταν επενεργούν πολλές ανεξάρτητες δυνάμεις σε πολλά αντικείμενα εντός του ψηφιακού χώρου, η εφαρμογή αγνοεί νέες κλήσεις στον χειριστή φυσικής λόγων μη επαρκούς μνήμης και τα αντικείμενα δεν συμπεριφέρονται σωστά [40], [41].

2.13.3 Κατηγορίες αντικειμένων – Actor

Τα αντικείμενα αυτής της κατηγορίας (Actor objects) είναι όλα αυτά τα αντικείμενα που δέχονται συντεταγμένες μετατόπισης και μπορούν να δημιουργηθούν και να καταστραφούν μέσω κώδικα ακόμα και σε πραγματικό χρόνο. Τα περισσότερα αντικείμενα που θα συναντήσουμε εντός ενός τρισδιάστατου χώρου είναι αντικείμενα με γονέα την κλάση Actor. Βρίσκουν ποικίλες χρήσεις ανάλογα με τον τελικό τύπο του αντικειμένου αλλά όλα μοιράζονται μερικές βασικές ιδιότητες όπως οι συντεταγμένες θέσης και περιστροφής, το μέγεθος τους, οι ιδιότητες σύγκρουσης και ο τρόπος κίνησης τους [42].

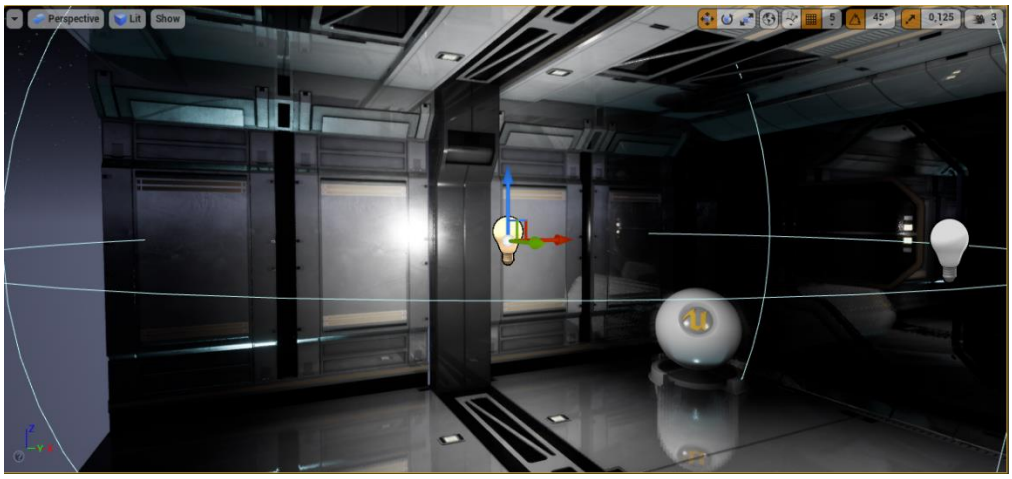
2.13.4 Κατηγορίες αντικειμένων – Στατικά, ακίνητα αντικείμενα

Οποιοδήποτε αντικείμενο είναι συμπαγές και μπορεί να είναι ακίνητο στον χώρο θεωρείται στατικό (static object). Οι χαρακτήρες μπορούν να περπατούν πάνω σε τέτοια αντικείμενα ή να εκτελούν οποιαδήποτε ενέργεια χωρίς να τα επηρεάζουν, αφού αυτά τα αντικείμενα δεν αντιδρούν σε εξωτερικά γεγονότα. Είναι τα πιο ελαφριά σε επεξεργαστική ισχύ αντικείμενα, ειδικά όταν είναι μονίμως ακίνητα (στατική κινητικότητα). Μπορούν να λαμβάνουν πληροφορίες φωτισμού και να παράγουν σκιά αν ενεργοποιηθούν οι κατάλληλες ρυθμίσεις, όπως επίσης και να αλλάξουν ορατότητα σε πραγματικό χρόνο [27]. Συνήθως δεν δίνουν επιθυμητό αποτέλεσμα όταν αλλάζει η ορατότητα τους και για αυτό προτιμάται η αλλαγή να γίνεται όταν δεν τα δείχνει η κάμερα του παίκτη.

2.13.5 Κατηγορίες αντικειμένων – Δυναμικά αντικείμενα

Σε αντίθεση με τα στατικά, τα δυναμικά αντικείμενα είναι όλα εκείνα με τα οποία ο παίκτης μπορεί να αλληλοεπιδράσει με κάποιον τρόπο. Είναι επίσης τα αντικείμενα των οποίων η κατάσταση μεταβάλλεται ανάλογα με συνθήκες. Συνήθως σε αυτά τα αντικείμενα επενεργούν εντολές ύστερα από ένα γεγονός μέσω του μηχανισμού ελέγχου γεγονότων του Unreal. Για παράδειγμα, δυναμικά θεωρούνται τα αντικείμενα της κατηγορίας movable static και movable τα οποία συμμετέχουν σε πράξεις συγκρούσεων και μαθηματικά βαρύτητας. Όλα τα αντικείμενα που μπορούν να καταστραφούν με οπτικό τρόπο, δηλαδή να σπάσουν σε μικρότερα κομμάτια, να λιώσουν κλπ., είναι μέρος της ίδιας κατηγορίας [43].

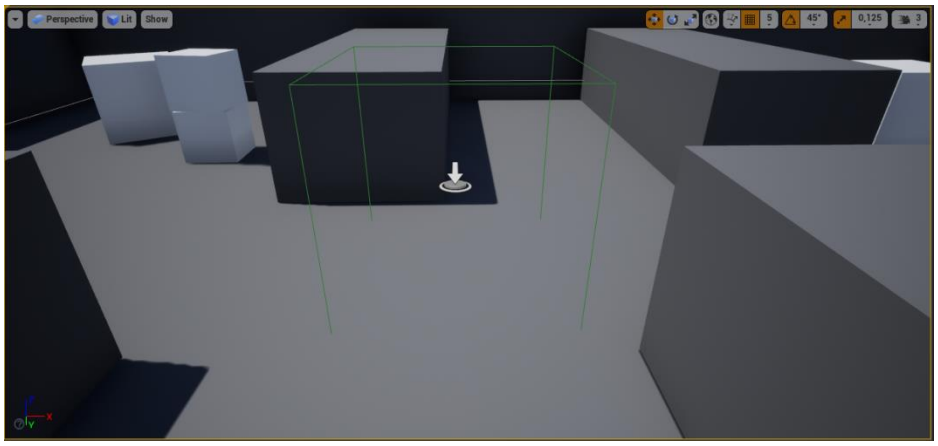
2.13.6 Κατηγορίες αντικειμένων – Φώτα



Εικόνα 2.9: Παράδειγμα από ένα φως

Τα αντικείμενα φωτός στη μηχανή Unreal αποτελούν μια δομή δεδομένων με παραμέτρους όπως χρώμα (Light Color), ένταση (Brightness), εύρος δράσης (Attenuation Radius) και μέγεθος πηγής ανάμεσα σε πολλές άλλες. Χρωματίζουν την επιφάνεια που ορίζει το εύρος τους, σε εκατοστά ανάλογα με μια δεδομένη ένταση που μετριέται σε αφηρημένες μονάδες, σε lumens και σε καντέλες (cds). Τα φώτα δύναται να έχουν συναρτήσεις φωτισμού μέσω υλικού (Material Functions), όπως επίσης τύπους κινητικότητας, σαν τα υπόλοιπα αντικείμενα. Τα στατικά φώτα αφού υπολογιστούν μια φορά δεν αλλάζουν. Οπότε έχουν σταθερή σκιά και φως στον χώρο, με αποτέλεσμα να μπορούν να αποθηκευτούν στο χρωματικό χάρτη του χώρου (Lightmap). Επειδή ακριβώς αποθηκεύονται σε στατικές εικόνες, το χρώμα και οι σκιές τους γίνονται ένα με το στατικό περιβάλλον και δεν μπορούν να φτιάξουν σκιά για κινούμενα αντικείμενα όπως χαρακτήρες. Είναι τα πιο χαμηλού υπολογιστικού κόστους φώτα και χρησιμοποιούνται κυρίως σε εφαρμογές για κινητά. Τα στατικά κινούμενα φώτα μπορούν να μεταβάλλουν την ένταση και το χρώμα τους αλλά δεν μπορούν να αλλάξουν θέση σε πραγματικό χρόνο. Είναι τα πιο υψηλής ποιότητας φώτα της μηχανής, προσφέρουν ποικίλες χρήσεις και πολύ καλό οπτικό αποτέλεσμα. Έχουν την δυνατότητα να φτιάξουν σκιές για κινούμενα αντικείμενα αλλά και στατικές σκιές που δεν υπάρχει λόγος να υπολογίζονται σε κάθε πλαίσιο, μειώνοντας το κόστος υπολογισμού αρκετά. Μόνο τέσσερα διαφορετικά στατικά κινούμενα φώτα επιτρέπεται να φτιάχνουν σκιά για ένα στατικό αντικείμενο γιατί χρησιμοποιούνται τέσσερα σταθερά κανάλια για τον υπολογισμό της τελικής σκιάς του κάθε αντικειμένου. Αν ένα φως με σκιά περισσεύει θα φαίνεται με ένα κόκκινο Χ κατά την αποσφαλμάτωση φωτός και θα αγνοείται από την μηχανή αλλά με απρόβλεπτα αποτελέσματα [44], [45], [46].

2.13.7 Κατηγορίες αντικειμένων – Πυροδοτητές



Εικόνα 2.10: Πυροδοτητής - Trigger

Οι πυροδοτητές (Triggers) είναι βασικά γεωμετρικά σχήματα που αποσκοπούν στο να δώσουν μια πληροφορία σε κάποιο πρόγραμμα κατά τη διέλευση ενός αντικειμένου μέσα από αυτά. Είναι διάφανα εντός της μηχανής ώστε να φαίνεται ο χώρος τον οποίο καλύπτουν και έχουν γεωμετρία σύγκρουσης (Collider Mesh) αλλά από προεπιλογή επιτρέπουν σε οτιδήποτε στατικό ή δυναμικό στο χώρο να τα διαπεράσει. Καθώς ένα ξένο αντικείμενο διαπερνά ένα αντικείμενο Trigger, ο χειριστής του αντικειμένου πυροδοτεί το αντίστοιχο γεγονός πρόσκρουσης. Αν και μπορούν να δημιουργηθούν εξ' ολοκλήρου σε C++, τα αντικείμενα Trigger συναντώνται συνήθως σε μορφή Blueprint έτσι ώστε να μπορεί να οπτικοποιηθεί η τελική τους μορφή άμεσα. Το σχήμα και το μέγεθος τους, το χρώμα τους, οι ιδιότητες τους. Επίσης είναι τα πλέον πολυχρησιμοποιημένα αντικείμενα για διαδικασίες αποσφαλμάτωσης. Κατά την πυροδότηση μπορεί να εκτελεστεί μια πληθώρα από ενέργειες που θα έχει κάποια σημασία για το βιντεοπαιχνίδι. Μπορεί να σηματοδοτεί ότι ο παίκτης πέρασε σε μια τοποθεσία ή ότι ένα αντικείμενο είναι ενεργό όσο ένας χαρακτήρας βρίσκεται στον πυροδοτητή [47], [48].

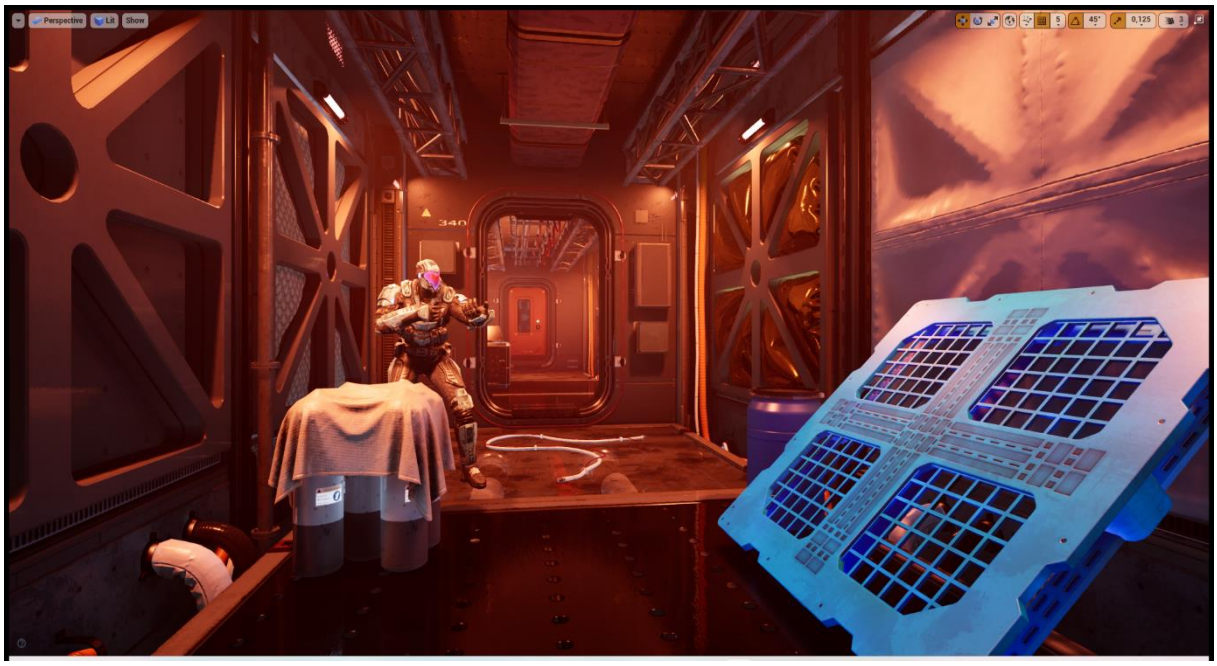
2.13.8 Κατηγορίες αντικειμένων – Ειδικά εφέ

Τα ειδικά εφέ χωρίζονται σε κατηγορίες σύμφωνα με την λειτουργικότητα τους. Αποτελούν άλλοτε δυναμικά αντικείμενα, άλλοτε στατικά και άλλοτε ολόκληρους χαρακτήρες που είναι κυρίως ειδικά εφέ.

Τα εφέ από στοιχεία της φύσης όπως φωτιά, νερό, σκόνη ή βροχή είναι συστήματα σωματιδίων (Particle Systems). Περιλαμβάνουν μια ή περισσότερες συναρτήσεις δημιουργίας και καταστροφής σωματιδίων σε ορισμένο χρόνο, με κίνηση που δίνεται από μια γραφική παράσταση. Τα συστήματα αυτά διαθέτουν επίσης ένα υλικό το οποίο δίνει την υφή και τις επιπλέον ιδιότητες απεικόνισης (Material).

Τα εφέ που έχουν να κάνουν με το οπτικό αποτέλεσμα μετά από μια πρόσκρουση είναι αντικείμενα πρόσκρουσης (Impact effect) και στάμπας (decals). Τα αντικείμενα στάμπας περιλαμβάνουν μια υφή που επικαλύπτει την επιφάνεια στην οποία θα τοποθετηθούν σαν στάμπα ενώ τα Impacts είναι σύνθετα αντικείμενα (blueprint) και συνήθως περιλαμβάνουν ένα εφέ σωματιδίων, έναν ήχο και ένα αντικείμενο στάμπα (decal). Το εφέ χρησιμοποιείται κατά την διάρκεια της πρόσκρουσης, ο ήχος στο τέλος της και η στάμπα εμφανίζεται και παραμένει αφού ολοκληρωθεί ο κύκλος ζωής του εφέ [49].

2.13.9 Κατηγορίες αντικειμένων – Χαρακτήρες



Εικόνα 2.11: Χαρακτήρας

Οι χαρακτήρες είναι σύνθετα αντικείμενα που συναντάμε συχνά σε μορφή ενός ενιαίου blueprint. Περιλαμβάνουν ένα αντικείμενο σκελετού (Skeletal Mesh), μια κάψουλα συγκρούσεων (Collision Bounds) και ένα αντικείμενο κίνησης (Movement Component) [50].

Το αντικείμενο σκελετού περιλαμβάνει διανύσματα στο χώρο με αφετηρία το διάνυσμα (0,0,0) με συγκεκριμένη κατεύθυνση έτσι ώστε να αποτελούν έναν ψηφιακό σκελετό. Τα φορτώνει από ένα αρχείο σκελετού, το οποίο είναι ένα απλό αρχείο κειμένου με τις συντεταγμένες θέσης και κατεύθυνσης του κάθε οστού. Επιπλέον, περιλαμβάνει ένα blueprint κίνησης που περιγράφει ποια αρχεία κίνησης θα παίξει ο χαρακτήρας με αυτόν τον σκελετό. Ο σκελετός είναι συνδεδεμένος με ένα αντικείμενο φυσικής (Physics Object) το οποίο διατηρεί πληροφορίες σύγκρουσης για κάθε ψηφιακό οστό του σκελετού και τα φυσικά όρια μετατόπισης και περιστροφής του κάθε οστού. Έτσι, όταν ένας χαρακτήρας εκτελεί κινήσεις λόγω βαρύτητας (πτώση από ύψος ή σύγκρουση), δεν χρειάζονται αρχεία κινήσεων, αρκεί να λειτουργεί σωστά το αντικείμενο φυσικής του χαρακτήρα και αυτός θα προσομοιώσει την πτώση με φυσικό τρόπο.

Η κάψουλα συγκρούσεων χρησιμοποιείται όταν ο χαρακτήρας εκτελεί κινήσεις από αρχεία καθώς είναι πολύ πιο απλό γεωμετρικό σχήμα και εκτελεί λιγότερες πράξεις συγκρούσεων. Συνήθως περιβάλλει τον χαρακτήρα και δείχνει πότε ο χαρακτήρας συγκρούεται με ένα αντικείμενο και τι να γίνει σε κάθε περίπτωση. Για παράδειγμα, αν η κάψουλα του εφάπτεται με το έδαφος και το έδαφος είναι από συμπαγές υλικό, ο χαρακτήρας θα πατήσει πάνω του. Αν συγκρουστεί με ένα τοίχο ή οποιοδήποτε στερεό αντικείμενο θα σταματήσει να κινείται, δεν θα το διαπεράσει. Αν βρίσκεται στο νερό, θα διαπεράσει την επιφάνεια του νερού αλλά θα επιπλέει σύμφωνα με τις παραμέτρους προσομοίωσης σύγκρουσης με υγρά [51].

Το αντικείμενο κίνησης είναι ένα αντικείμενο κώδικα που μπορεί να προσαρτηθεί σε οποιοδήποτε κινούμενο αντικείμενο. Διαθέτει παραμέτρους κίνησης όπως ταχύτητα, περιστροφή, επιτάχυνση για

χαρακτήρες που κινούνται σε έδαφος, στο νερό και στον αέρα. Έτσι μπορεί εύκολα να προστεθεί ή να αφαιρεθεί ένα χαρακτηριστικό από έναν χαρακτήρα. Για παράδειγμα, να αλλάζει η μέγιστη ταχύτητα ενός χαρακτήρα σύμφωνα με τι οπλισμό κουβαλάει. Να μπορεί να ελιχθεί ή όχι στον αέρα. Ακόμα, να έχει την δυνατότητα να ίπταται, δηλαδή να μην πέφτει όταν φτάσει στο τέλος μιας στερεής πλατφόρμας.

2.13.10 Κατηγορίες αντικειμένων – Αξεσουάρ και ρουχισμός

Οποιοδήποτε αντικείμενο το οποίο μπορεί να φορέσει ένας χαρακτήρας αποτελεί μια ειδική κατηγορία ενός χαρακτήρα. Θεωρείται αντικείμενο σκελετού γιατί περιλαμβάνει έναν σκελετό και κάποιες φορές διαθέτει δικό του αντικείμενο φυσικής. Συνήθως μοιράζεται το ίδιο αντικείμενο σκελετού με τον χαρακτήρα που το φοράει και δεν μπορεί να λειτουργήσει αυτόνομα. Εκτός από τον σκελετό, μοιράζεται και το κώδικα κίνησης του χαρακτήρα έτσι ώστε να εκτελεί όποια κίνηση κάνει ο χαρακτήρας και να φαίνεται σαν να το κρατάει ή να το φοράει. Τέτοια αντικείμενα είναι ο οπλισμός ενός χαρακτήρα, τα διάφορα αξεσουάρ προστασίας όπως ένα κράνος, κάποιο σακίδιο ή πουγκί, μια ασπίδα. Ακόμη και καθημερινά αντικείμενα όπως ένα στυλό ή ένα βιβλίο το οποίο ο χαρακτήρας ξεφυλλίζει.

Αυτά τα αντικείμενα προσαρτώνται σε ειδικές υποδοχές στον σκελετό του χαρακτήρα που ονομάζονται bone sockets [50]. Είναι σημεία επάνω στον σκελετό με συντεταγμένες θέσης και όνομα το οποίο χρησιμοποιείται οπουδήποτε γίνεται αναφορά στην υποδοχή. Η προσάρτηση αντικειμένου σε ανύπαρκτη υποδοχή μπορεί να προκαλέσει κατάρρευση της μηχανής ή του παιχνιδιού, αν το λάθος έχει φτάσει στο στάδιο παραγωγής. Το ίδιο ισχύει και στην περίπτωση που γίνεται προσάρτηση ανύπαρκτου αντικειμένου σε υπαρκτή υποδοχή. Ο όρος ανύπαρκτος εδώ αναφέρεται σε αντικείμενο που δεν έχει δημιουργηθεί καθώς και σε αντικείμενο που δεν έχει προλάβει να φορτωθεί στη μνήμη. Οπότε η προσάρτηση πρέπει να γίνεται πάντοτε στο κατάλληλο σημείο της ροής εκτέλεσης του κώδικα και να ελέγχεται αν το αντικείμενο και η υποδοχή είναι φορτωμένα.

2.13.11 Κατηγορίες αντικειμένων – Οχήματα

Τα οχήματα είναι σύνθετα αντικείμενα (blueprint) και αποτελούνται από διάφορα μηχανικά ή οργανικά μέρη. Αυτό που τα διαφοροποιεί από τους χαρακτήρες είναι ότι μπορούν να τα χειριστούν άλλοι χαρακτήρες μέσω μιας σειράς ενεργειών που ενώνει το όχημα με τον χαρακτήρα - οδηγό και κινούνται σαν μία οντότητα για όσο ο χαρακτήρας θεωρείται ότι χρησιμοποιεί το όχημα.

Το όχημα, ειδικά όταν πρόκειται για αυτοκινούμενο, αποτελείται από διάφορα μέρη που συνεργάζονται μεταξύ τους και δίνουν την αίσθηση του ολοκληρωμένου αντικειμένου. Αυτά τα μέρη είναι αξεσουάρ του και προσαρτώνται σε υποδοχές ή απευθείας στα κόκκαλα του βασικού σκελετού του μοντέλου. Τέλος ο χαρακτήρας που το χρησιμοποιεί, προσαρτάται στην υποδοχή του οδηγού και το ελέγχει, παρόλο που αποτελεί αξεσουάρ του οχήματος. Οπότε προκύπτει μια ιεραρχία από ανεξάρτητα αντικείμενα τα οποία είναι συνδεδεμένα σε υποδοχές και κόκκαλα με κάποιο κώδικα.

2.14 Μενού

Τα μενού στο Unreal Engine μπορούν να κατασκευαστούν με δύο τρόπους, με ένα αυτόματο εργαλείο που ονομάζεται UMG UI Designer [52] και με μια διεπαφή προγραμματιστή, το Slate UI Framework. Το εργαλείο UMG είναι εύχρηστο και απλό στην λειτουργικότητα του καθώς χρησιμοποιεί κυρίως προ-κατασκευασμένα στοιχεία μενού μέσω μιας παλέτας. Όποιος έχει εμπειρία με κατασκευή μενού εντός του Visual Studio ή σε οποιοδήποτε πρόγραμμα που χρησιμοποιείται παλέτα για την διεπαφή χρήστη, τότε δεν θα έχει απολύτως κανένα πρόβλημα να κατασκευάσει μενού στο Unreal Engine. Τα αρχεία μενού είναι επίσης σε μορφή blueprint και δημιουργούνται όπως και οποιοδήποτε άλλο αντικείμενο

εντός της μηχανής. Η διαφορά τους είναι ότι προέρχονται από την κλάση Widget Blueprint της κατηγορίας User Interface και αντί για παράθυρο προβολής στο Blueprint υπάρχει παράθυρο σχεδιαστή. Αυτό έχει λειτουργίες διεπαφής, όπως παλέτα και μενού ιεραρχίας στα αριστερά ενώ στα δεξιά μενού λεπτομερειών του κάθε επιλεγμένου στοιχείου διεπαφής. Στο κέντρο υπάρχει ένα δισδιάστατο καρτεσιανό σύστημα συντεταγμένων με δυνατότητα ζουμ, πάνω στο οποίο τοποθετούνται τα στοιχεία διεπαφής. Ένα ορθογώνιο διακεκομμένο περίγραμμα στο παράθυρο οριοθετεί το μέγεθος της εικόνας του χρήστη εντός του βιντεοπαιχνιδιού. Υποθέτει ότι ο χρήστης παίζει σε ανάλυση Full HD ή αλλιώς 1920 επί 1080 εικονοστοιχεία (1080p). Η διαδικασία κατασκευής μενού είναι παρόμοια με αυτή στο Visual Studio με κάποια στοιχεία που συναντώνται και στη κατασκευή διεπαφής για ιστοσελίδες. Ο σχεδιαστής επιλέγει σχήματα ομαδοποίησης που λέγονται πάνελ (panel), ρυθμίζει τις ιδιότητες τους έτσι ώστε να καταλαμβάνουν κάποιο συγκεκριμένο χωρίο της εικόνας, να έχουν κατάλληλα περιθώρια και συγκεκριμένη στοίχιση και γραμματοσειρά. Έπειτα, προσθέτει μέσα στα πάνελ στοιχεία μενού όπως κουμπιά (buttons), καρτελάκια με περιγραφή (labels), κάθετα και οριζόντια κουτιά για εκ νέου οριοθέτηση στοιχείων ή διάφορα γραφικά όπως μπάρες προόδου (progress bar). Το κάθε στοιχείο περιέχει ρυθμίσεις για στοίχιση, γραμματοσειρά, χρώμα και σχήμα. Κάποια δίνουν την δυνατότητα εισαγωγής ενός γραφικού σαν φόντο ή σαν κείμενο. Τέλος προστίθεται κώδικας σε μορφή blueprint με διάφορους τρόπους και φαίνεται στο παράθυρο γραφήματος το αποτέλεσμα της τελικής συνάρτησης για κάθε αντικείμενο. Στο γράφημα, το κάθε στοιχείο μπορεί να συνδεθεί με έτοιμες συναρτήσεις της μηχανής ή να χρησιμοποιηθούν καινούργιες, γραμμένες και μεταγλωττισμένες σε C++. Μπορεί να γίνει αναφορά σε στοιχεία μενού στο γράφημα μέσω του μοναδικού αναγνωριστικού τους (ID). Επειδή όλα τα στοιχεία κληρονομούν ιδιότητες από τα ανώτερα σε ιεραρχία στοιχεία, αν τροποποιηθεί ένα πάνελ οργάνωσης, τροποποιούνται όλα τα συνδεδεμένα με αυτό στοιχεία αντίστοιχα. Έτσι είναι εύκολο να γίνει εναλλαγή μενού χωρίς να χρειαστεί να φορτωθεί ένα άλλο αρχείο, αρκεί μόνο να κρυφτεί ένα πάνελ οργάνωσης και να εμφανιστεί κάποιο άλλο μέσω της ιδιότητας Visibility. Τα μενού στο UMG υποστηρίζουν επίσης την προβολή πολυμέσων εντός κατάλληλων στοιχείων και την δυνατότητα αλλαγής της εμφάνισης τους μέσω υλικών γραφικών (Materials).

Από την άλλη μεριά η διεπαφή προγραμματιστή Slate UI Framework χρησιμοποιείται για την κατασκευή πιο εξειδικευμένων μενού. Η διεπαφή χρήστη του Unreal Engine είναι κατασκευασμένη με το Slate. Το Slate είναι υλοποιημένο πάνω στην λογική Model-View-Controller (MVC) όπου ένα μοντέλο δεδομένων διαβάζει και επεξεργάζεται δεδομένα, τα συσκευάζει σε συμπαγείς ακολουθίες όπως πίνακες και δυαδικές σειρές και τα στέλνει σε έναν χειριστή για περεταίρω επεξεργασία. Ο χειριστής είναι υπεύθυνος να λάβει αποφάσεις σχετικά με το τι να εμφανίσει και πότε, ανάλογα με τα δεδομένα που πήρε από το μοντέλο. Τέλος ο χειριστής συνδέει τα δεδομένα με την όψη, ένα σύστημα υπεύθυνο μόνο για την εμφάνιση των δεδομένων με κάποιο τρόπο στον χρήστη. Έτσι το Slate έχει έναν χειριστή τον οποίο καλεί ο προγραμματιστής με συναρτήσεις, τον τροφοδοτεί με δεδομένα που προκύπτουν από εγγραφή/ανάγνωση και επεξεργασία και δημιουργεί αντικείμενα όψης (SWidgets και STextBlocks) για την εμφάνιση των δεδομένων. Η πολυπλοκότητα και το κόστος επεξεργασίας ενός μενού σε Slate είναι ανάλογο με τον αριθμό ενεργών Widget στην οθόνη (Live Widgets). Αν αλλάζουν τα δεδομένα πολύ γρήγορα, τότε το μενού καταστρέφεται και ξαναδημιουργείται καθώς είναι πιο εύκολη η διαχείριση του και δεν προκύπτει ποτέ λάθος στην ανάγνωση δεδομένων από κρυφή μνήμη. Τέτοια ενεργά Widget είναι το παράθυρο προβολής (Viewport) της μηχανής και το παράθυρο σχεδίασης ή προβολής ενός Blueprint [52].

2.15 Επίλογος

Η μηχανή γραφικών Unreal Engine φαίνεται κατάλληλη για την ανάπτυξη μεσαίου και μεγάλου μεγέθους έργα ποικίλου χαρακτήρα. Είναι αρκετά απαιτητική σε πόρους και χρειάζεται αρκετό χρόνο για να μάθει κανείς τις λειτουργικές λεπτομέρειες της ώστε να την χρησιμοποιήσει για εμπορικό σκοπό. Ένα μειονέκτημα της είναι ότι απαιτεί ομάδα προγραμματιστών ώστε να δημιουργηθεί ένα έργο σε ικανοποιητικό χρονικό διάστημα. Αντίθετα με την δυσκολία λόγω του όγκου πληροφορίας, η διεπαφή του Unreal είναι αρκετά φιλική προς τον χρήστη και πολλές τον κατευθύνει αποκρύπτοντας πληροφορίες που δεν είναι σχετικές με αυτό στο οποίο εργάζεται.

Κεφάλαιο 3ο: Ανάπτυξη βιντεοπαιχνιδιού με Unreal Engine 4

3.1 Εισαγωγή

Η παρούσα εργασία υλοποιήθηκε γύρω από την ιδέα ενός σύντομης διάρκειας βιντεοπαιχνιδιού βολών πρώτου προσώπου στο οποίο καλούμαστε να φέρουμε εις πέρας μια στρατιωτική αποστολή με θέμα το διάστημα. Σκοπός του έργου αυτού ήταν, πέρα από την απόκτηση γνώσεων μέσω πρακτικής εξάσκησης, η ανάδειξη των δυνατοτήτων της μηχανής Unreal Engine και πως με ελάχιστο υλικό και κόστος ήταν δυνατό να κατασκευαστεί ένα βιντεοπαιχνίδι, ή τουλάχιστον ένα έργο μεγαλύτερης κλίμακας από ένα απλό μάθημα.

Η διαδικασία παραγωγής και ανάπτυξης θα μπορούσε να θεωρηθεί ως μια σειρά μαθημάτων που επέφεραν κάποιο αποτέλεσμα και η πραγματική πρόκληση ήταν ο σωστός συνδυασμός μαθημάτων και γνώσεων ώστε να κατασκευαστεί ένα ενιαίο έργο. Η μεθοδολογία που εφαρμόστηκε σε όλο το έργο έχει βάσεις στα στάδια παραγωγής του βιντεοπαιχνιδιού που αναλύθηκαν στη πρώτη ενότητα, αλλά φυσικά σε πολύ μικρότερη κλίμακα καθώς όλες οι απαραίτητες πρώτες ύλες παρέχονται μέσω της Epic Games και το έργο αποτελεί προϊόν ενός φοιτητή.

3.2 Στάδιο 1ο (Concept)

Επειδή πρόκειται για ένα μικρό έργο χωρίς καθόλου προϋπολογισμό, η ιδέα είχε επιρροή από τα διαθέσιμα υλικά που συμπεριλαμβάνονται στα μαθήματα του Unreal Engine και το ελεύθερα προσβάσιμο υλικό μέσω του Unreal Marketplace [32]. Επικράτησε το στοιχείο της επιστημονικής φαντασίας καθώς υπήρχαν έτοιμα πακέτα κατασκευής τρισδιάστατων χώρων και διακοσμητικά που θα επιτάχυναν πολύ την παραγωγή. Επιπλέον, ο τρόπος προβολής σε πρώτο πρόσωπο είναι πλέον διαδεδομένος σε βιντεοπαιχνίδια στρατιωτικού χαρακτήρα, για τεχνικούς, πρακτικούς και αισθητικούς λόγους. Το αποτέλεσμα ήταν ένα βιντεοπαιχνίδι βολών πρώτου προσώπου στο οποίο οι παίκτες χειρίζονται έναν στρατιώτη με διαστημική στολή και προσπαθούν να σώσουν έναν Αμερικανό πρέσβη από τα χέρια ενός αυτόματου σκάφους με ανδροειδή ρομπότ υψηλής ευφυίας για πλήρωμα. Ο χαρακτήρας καλείται να διεισδύσει στο εσωτερικό του σκάφους και να αναζητήσει τον πρέσβη καθώς επίσης οτιδήποτε μπορεί να αποτελέσει στοιχείο για την αποστολή. Στην πορεία θα αναμετρηθεί με τα ανδροειδή και θα πρέπει να βρει υλικά πρώτων βοηθειών και εξαρτήματα για να εξασφαλίσει την επιβίωση του.

3.3 Στάδιο 2ο (Pre-production)

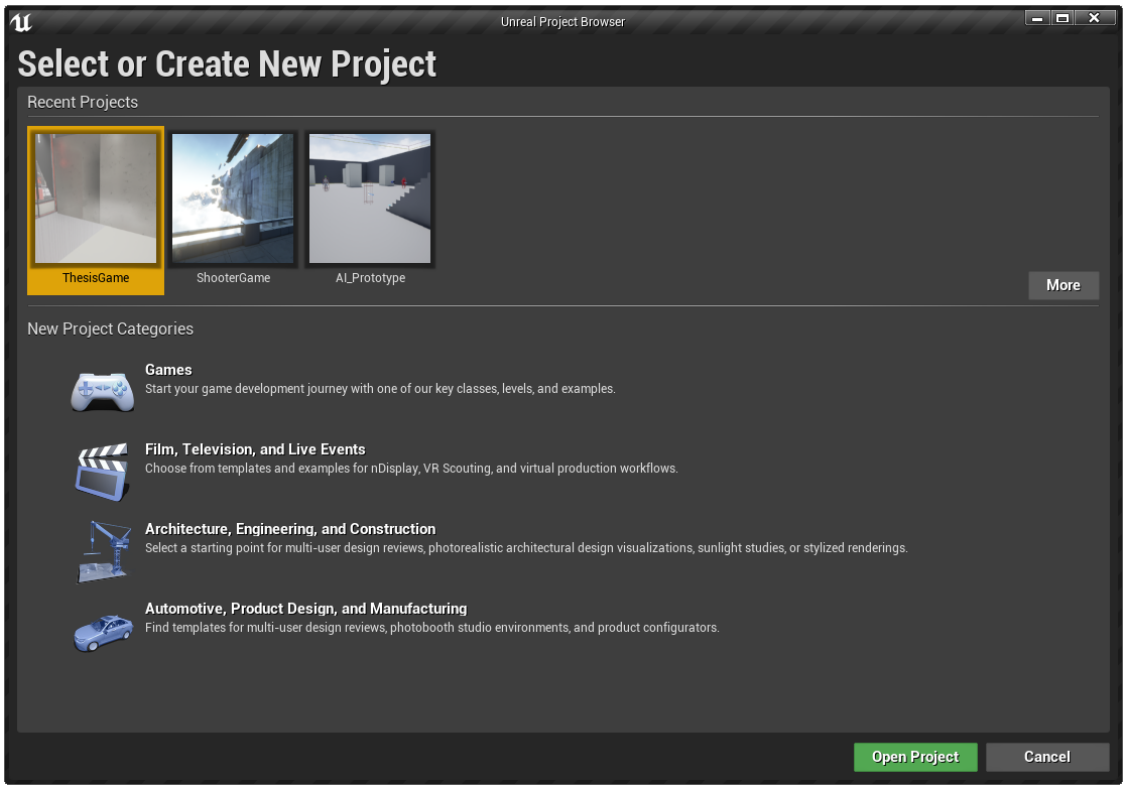
Με την βασική ιδέα έτοιμη, το επόμενο βήμα ήταν να χωριστεί το έργο σε επιμέρους τμήματα και να γίνει μια έρευνα ως προς το τι υλικά θα χρειαστούν [12], [13], [53]. Αυτά παρατίθενται στην παρακάτω λίστα:

- Τρισδιάστατοι χώροι
- Μοντέλα εσωτερικού χώρου με θέμα ένα διαστημόπλοιο (πχ. μεταλλικοί τοίχοι, πατώματα, κουφώματα, παράθυρα και στηρίγματα)
- Διακοσμητικά χώρου (σωλήνες, σχάρες, έπιπλα, αντικείμενα γραφείο, πάνελ, υπολογιστές)
- Φώτα (λάμπες LED, προβολείς, φωτιστικά γραφείων κ.λ.π)
- Εφέ (σπινθήρες, καπνός, λέιζερ)
- Ήχοι (εσωτερικού χώρου που θυμίζει σκάφος)
- Χαρακτήρες
- Μοντέλα χαρακτήρων με διαστημικές στολές
- Σύγχρονος οπλισμός (τουφέκια, πιστόλια κ.λ.π)
- Ήχοι (βηματισμός, πυροβολισμοί, ήχοι κρούσης)
- Κώδικας C++
- Για αλληλοεπίδραση με τον χώρο (συρόμενες πόρτες, σειρήνες συναγερμού)
- Για τους χαρακτήρες (πως θα κρατούν όπλα και θα κινούνται στο χώρο)
- Για τα όπλα (πως θα βγάζουν εφέ, τι βλήματα θα ρίχνουν, τι ζημιά θα κάνουν)
- Για το παιχνίδι (πως θα φορτώνεται η αποστολή, πως θα μεταφέρεται σε άλλο χώρο ο παίκτης κ.α)
- Τεχνητή νοημοσύνη (συμπεριφορά εχθρών και συμπαικτών)

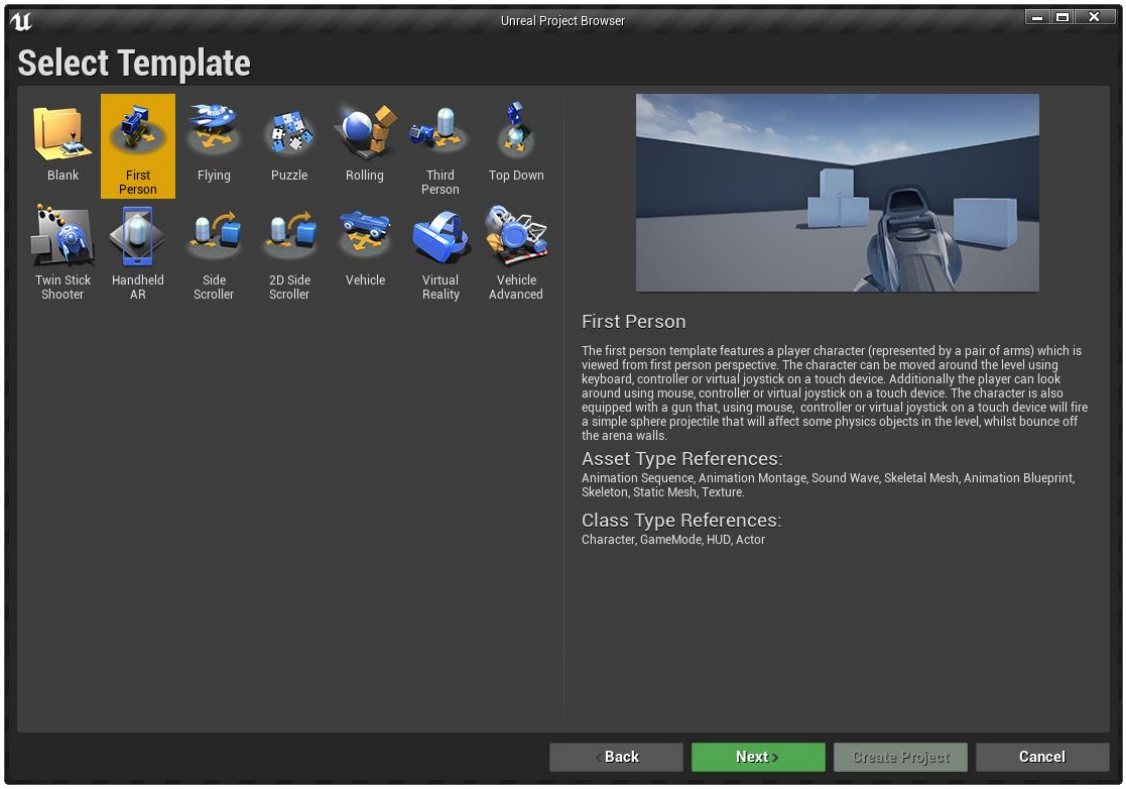
Για να έχει νόημα αυτή η λίστα, δημιουργήθηκε ένας δοκιμαστικός χάρτης στον οποίο έγινε πρακτικά αναπαράσταση των μηχανικών του βιντεοπαιχνιδιού και χρησιμοποιήθηκαν τα διάφορα υλικά της λίστας. Ό,τι πείραμα πέτυχε, ενσωματώθηκε στο τελικό έργο.

3.4 Δημιουργία έργου

Η ανάπτυξη του έργου ξεκίνησε με την επιλογή νέου έργου στην αρχική οθόνη του Unreal Engine και επιλέχθηκε το πρότυπο First person.

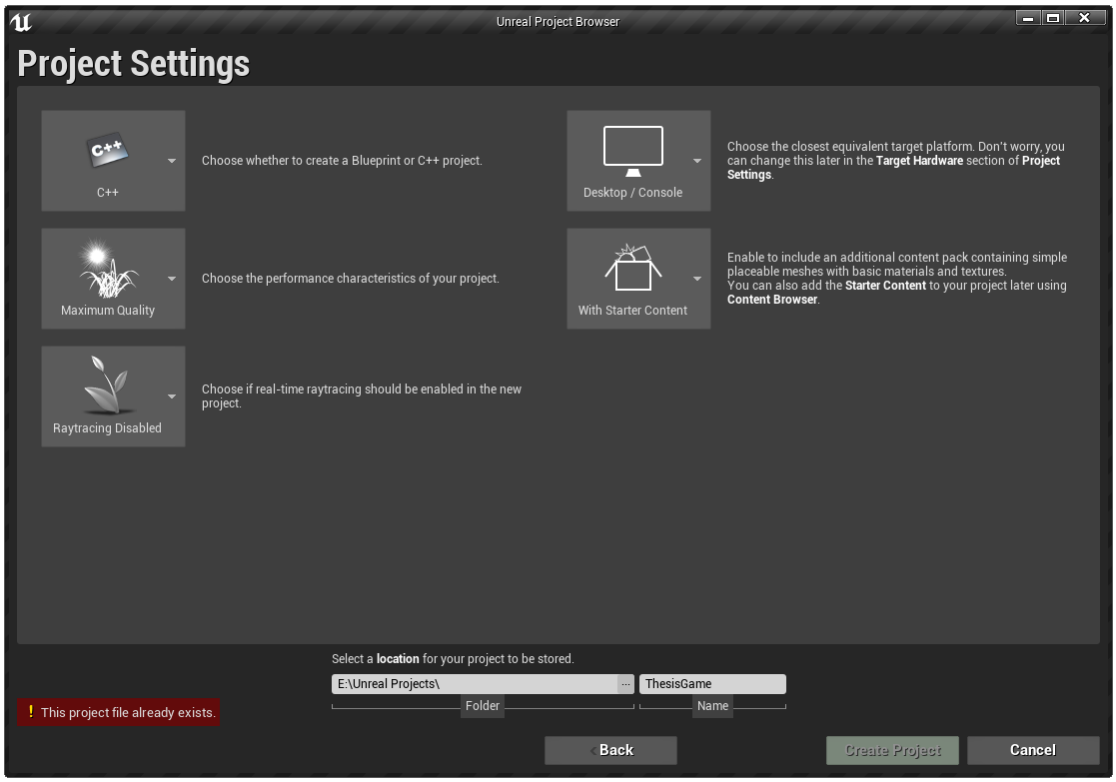


Εικόνα 3.1: Δημιουργία έργου



Εικόνα 3.2: Επιλογή προτύπου

Κατόπιν, στις ρυθμίσεις του έργου επιλέχθηκε ο τύπος έργου C++ για να μεταγλωττιστεί το πρότυπο First Person σε δυαδικό εκτελέσιμο και να δημιουργηθεί αυτόματα ένα αρχείο Visual Studio για επεξεργασία του κώδικα του έργου. Για ποιότητα επιλέχθηκε η μέγιστη (Maximum Quality) καθώς αυτή η ρύθμιση ταιριάζει περισσότερο σε εφαρμογές υπολογιστή. Ύστερα επιλέχθηκε να τρέχει σε υπολογιστή και κονσόλα (Desktop / Console) και να περιλαμβάνει βοηθητικό υλικό (Starter Content). Τέλος, δόθηκε η διαδρομή και το όνομα του έργου (Thesis Game). Αν το έργο υπάρχει ήδη, θα βγάλει μια κόκκινη ένδειξη όπως φαίνεται στην τρίτη εικόνα. Με το πράσινο κουμπί Create Project το έργο δημιουργείται και αποθηκεύεται στην διαδρομή που όρισε ο χρήστης. Την επόμενη φορά μπορεί να φορτωθεί από την λίστα προσωρινών έργων όπως στην πρώτη εικόνα που φαίνεται το ThesisGame ενεργό.



Εικόνα 3.3: Αρχικές ρυθμίσεις έργου

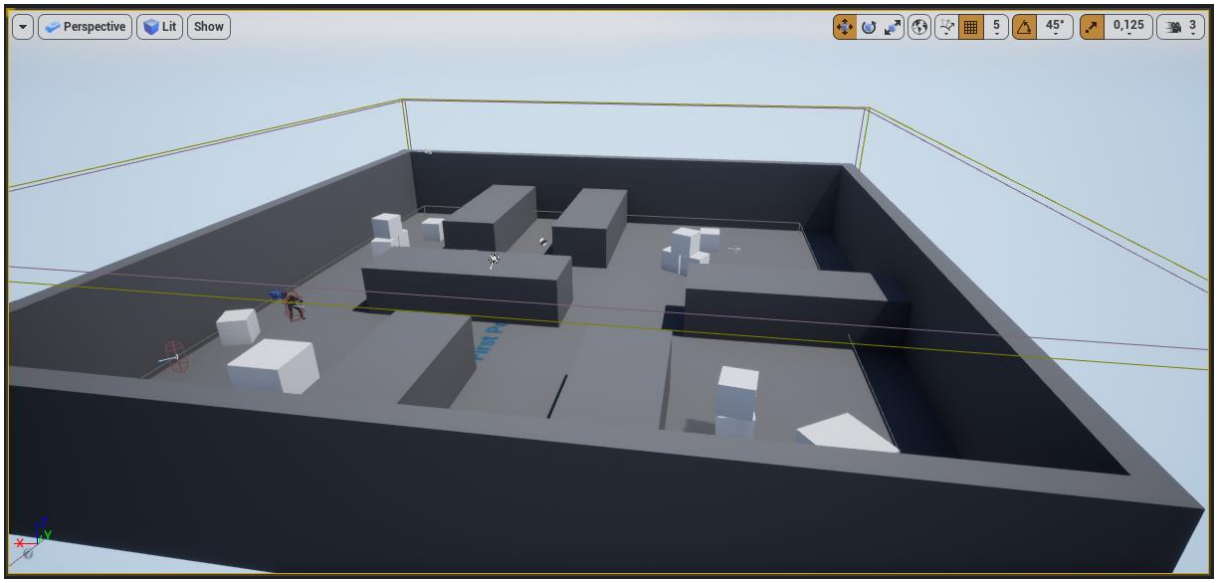
Τη πρώτη φορά που δημιουργείται ένα έργο C++, χρειάζεται κάποιο χρονικό διάστημα για να μεταγλωττιστεί σε δυαδικά εκτελέσιμα αρχεία ο κώδικας του έργου και να δημιουργηθεί το Visual Studio project με παραμέτρους από τη μηχανή. Αντίστοιχα για έργα Blueprint μεταγλωττίζονται τα προκαθορισμένα Blueprint σε εκτελέσιμα. Ο χρόνος εξαρτάται από την ισχύ της επεξεργαστικής μονάδας του χρήστη και κυμαίνεται από μερικά λεπτά έως και μισή ώρα για πολύ αργά μηχανήματα. Έπειτα το έργο είναι έτοιμο για ανάπτυξη.

3.5 Δημιουργία τρισδιάστατου χώρου

Όλα τα βιντεοπαιχνίδια, ανεξαρτήτως χαρακτήρα και εργαλείων κατασκευής, απαρτίζονται από έναν τρισδιάστατο χώρο τον οποίο βλέπει μια ψηφιακή κάμερα και προβάλλει το περιεχόμενο των

εικονοστοιχείων στην οθόνη. Οπότε κύριο μέλημα για να ξεκινήσει ένα έργο είναι να δημιουργηθεί ένας τρισδιάστατος χώρος που θα ικανοποιεί τις προγραμματιστικές ανάγκες για δοκιμές.

Για τη δημιουργία χώρου χρειάζεται ένα αρχείο χάρτη. Δηλαδή βάσει της ορολογίας του Unreal χρειάζεται ένα αρχείο συλλογής αντικειμένων δύο ή τριών διαστάσεων με συντεταγμένες στο χώρο. Το αρχείο χάρτη δημιουργείται με δεξί κλικ στον περιηγητή αρχείων και επιλογής του Level. Γίνεται επίσης με το πράσινο κουμπί Add New και κατόπιν επιλογής του Level.



Εικόνα 3.4: TestMap

Ο πρώτος χάρτης ήταν ο TestMap, στον οποίο όπως φαίνεται στην εικόνα τοποθετήθηκαν βασικά γεωμετρικά σχήματα. Ενα τεράστιο τετράγωνο για πάτωμα, ορθογώνια παραλληλόγραμμα για να οριοθετηθούν οι διάδρομοι και οι εξωτερικοί τοίχοι. Σκεπή δεν έχει ο χώρος για να είναι πιο εύκολα διαχειρίσιμος εντός της μηχανής. Τα μικρά λευκά κουτιά σηματοδοτούν διακοσμητικά και κάποια διαθέτουν ρυθμίσεις για προσομοίωση φυσικής. Έχοντας αναφερθεί πλήρως στον χειρισμό του περιηγητή χώρου (Viewport), είναι γνωστό ότι ο χάρτης ήταν κενός εξαρχής. Οπότε, πρώτα προστέθηκε ένα φως τύπου Directional, δηλαδή με συγκεκριμένη κατεύθυνση εκπομπής φωτονίων και ένα αντικείμενο blueprint τύπου Sky Sphere, το οποίο φτιάχνει έναν θόλο ουρανού σε μια απόσταση από το κέντρο του χάρτη. Σε συνδυασμό με το φως και ένα blueprint τύπου Atmospheric Fog το οποίο δημιουργεί ατμόσφαιρα στον ουρανό, ο χώρος φωτίστηκε σαν να είναι μια ηλιόλουστη μέρα. Επόμενο βήμα ήταν να προστεθούν τα γεωμετρικά σχήματα και κάποιοι σηματοδότες, ένα για τις αντανακλάσεις (Sphere Reflection Capture), ένα Lightmass Importance Volume για να υπολογίζει το δυναμικό φωτισμό μόνο εντός του δωματίου και ένα Post Process Volume για να υπολογίζει τα εφέ της σκηνής μόνο εντός του δωματίου. Αυτά τα αντικείμενα υπάρχουν σχεδόν σε κάθε χάρτη όπου υπάρχουν πολύπλοκα εφέ φωτισμού και ατμόσφαιρας. Στο δοκιμαστικό χάρτη προστέθηκαν μόνο για να φαίνεται καθαρά ο χώρος και να φωτίζεται από παντού.

Παρόμοια διαδικασία ακολουθήθηκε για την κατασκευή των εσωτερικών του σκάφους, με τη διαφορά ότι έπρεπε να γίνει σωστό ταίριασμα των μοντέλων εσωτερικού χώρου χωρίς να μείνουν τρύπες και δόθηκε ιδιαίτερη προσοχή στον φωτισμό έτσι ώστε να υπάρχει αληθοφάνεια χωρίς να είναι υπερβολικά βαρύ στην επεξεργασία το σύστημα φωτισμού.

3.6 Στάδιο 3ο (Production)

Στο σημείο αυτό, δηλαδή το σημείο όπου ολοκληρώθηκαν οι δοκιμές στον χάρτη TestMap, το έργο μετέβηκε στο στάδιο παραγωγής. Οτιδήποτε λειτούργησε κατά τις προδιαγραφές του, αποθηκεύτηκε σε μορφή Blueprint και κρατήθηκε σε ένα σαφώς οργανωμένο κατάλογο έτσι ώστε να χρησιμοποιηθεί κατά την παραγωγή. Η διαδικασία της παραγωγής με τη σειρά της περιλαμβάνει αρκετά στάδια για να φτάσει το βιντεοπαιχνίδι σε μια ικανοποιητική κατάσταση όπου θα θεωρείται διαθέσιμο προς τον παίκτη (playable) [9]. Σύμφωνα με τις δοκιμές, λειτούργησαν τα εξής:

- Ένας χαρακτήρας τον οποίο μπορούμε να ελέγχουμε εμείς αλλά και ο υπολογιστής
- Όπλα με διαφορετικές ιδιότητες που μπορεί να αλλάζει ο χαρακτήρας
- Ένα μονοπάτι το οποίο να ακολουθεί σωστά ο υπολογιστής για να μετακινηθεί στον χώρο
- Ένα σύστημα νοημοσύνης που εκτελεί περιπολία, ερευνά τον χώρο αν ακούσει οτιδήποτε απειλητικό και επιτίθεται με συγκεκριμένη τακτική αν δει εχθρό
- Ένα κεντρικό μενού που μπορεί να φορτώνει ένα χάρτη
- Ένα δευτερεύον μενού που είναι πάντα ενεργό στην οθόνη (Heads Up Display, HUD)
- Μια διαδικασία αποθήκευσης και φόρτωσης της κατάστασης του παιχνιδιού (SaveGame)
- Αυτόματες πόρτες με ειδική γεωμετρία για να ανοίγουν αν βρίσκεται κάποιος χαρακτήρας κοντά

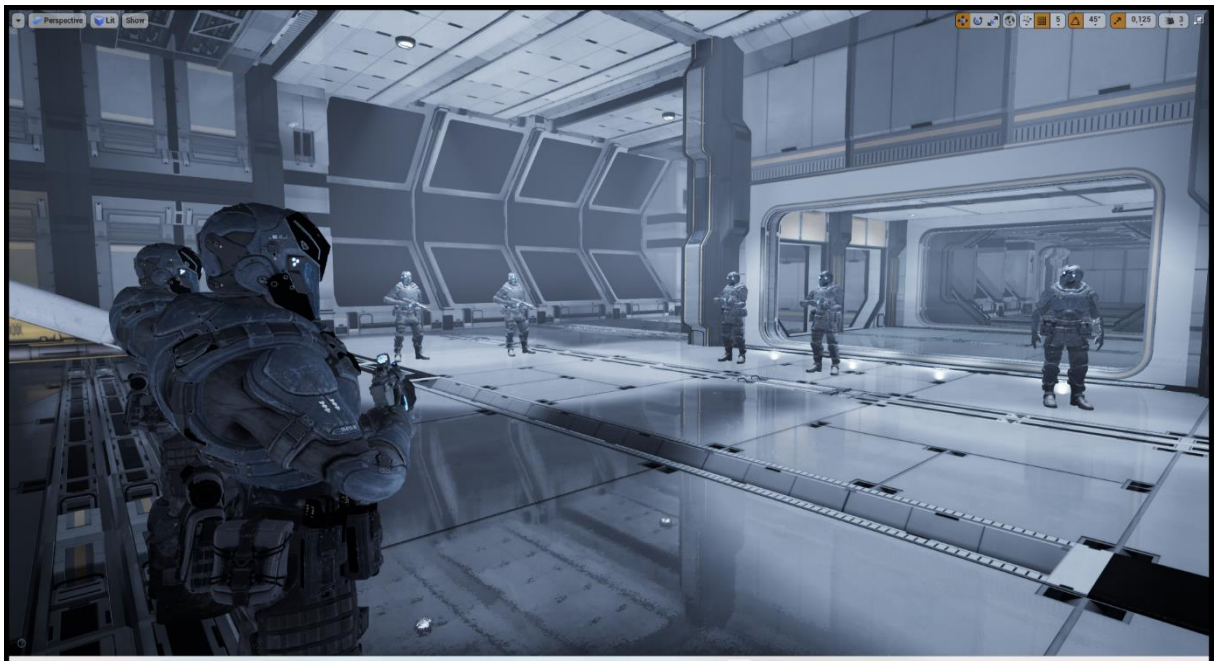
3.7 Εμπλουτισμός χώρου

Μετά την ένωση των σχετικών μοντέλων γεωμετρίας και τοποθέτησης των διακοσμητικών, σειρά έχει η προσθήκη διαφόρων αντικειμένων με έμφαση στην αισθητική και τη λειτουργικότητα. Τα εσωτερικά του διαστημόπλοιου έπρεπε να δίνουν την εντύπωση ότι το σκάφος είναι ενεργό, το οποίο έγινε με τη προσθήκη κατάλληλου φωτισμού και καπνού στους στενούς διαδρόμους. Επειδή οι διάδρομοι αυτοί προορίζονται για τους μηχανικούς, είναι λογικό να βρει κανείς εργαλεία και καλώδια παρατημένα από επισκευές, οπότε προστέθηκαν τα κατάλληλα διακοσμητικά.

Εικόνα 3.5: Χώρος από το πρώτο χάρτη με διακοσμητικά

3.7.1 Φωτισμός

Ο θάλαμος προσγείωσης ήταν αρκετά μεγάλος, τόσο ώστε να χωράει θεωρητικά ένα μικρό σκάφος ανίχνευσης ή συντήρησης. Για να φωτιστεί με αληθοφάνεια θα έπρεπε να έχει πολλές λάμπες LED όπου η κάθε μια να παράγει φως. Το αποτέλεσμα ήταν το καλύτερο δυνατό οπτικά αλλά πρακτικά υπήρχε μεγάλο πρόβλημα με την επιφάνεια που κάλυπτε η κάθε λάμπα. Συνεπώς, αλλάζοντας όλες τις πηγές φωτός σε στατικά φώτα χωρίς σκιά και προσθέτοντας ένα φως με σκιά και μεγάλο εύρος έλυσε το πρόβλημα επικάλυψης καθώς ο χώρος φωτίζεται και φαίνεται να πηγάζει φως από τις λάμπες. Το τελικό αποτέλεσμα φαίνεται στις παρακάτω εικόνες.



Εικόνα 3.6: Στρατιώτες στον θάλαμο προσγείωσης

Εικόνα 3.7: Δωμάτιο συντήρησης

Εικόνα 3.8: Θάλαμος προσγείωσης και το διαστημόπλοιο του παίκτη

3.7.2 Ήχος

Σε όλα τα βιντεοπαιχνίδια, από τα πρώτα που κυκλοφόρησαν σε μηχανήματα arcade μέχρι τα σύγχρονα, η μουσική και τα ηχητικά εφέ αποτέλεσαν σημαντικό παράγοντα της εμπειρίας του παίκτη. Για αυτό λοιπόν δόθηκε προσοχή σε κάποια σημεία κλειδιά που αφορούν τους ήχους του παιχνιδιού.

Τα αρχεία ήχου εισάγονται στον Editor όπως και τα αρχεία κινήσεων ή γραφικών. Η μηχανή υποστηρίζει αρχεία ήχου με κατάληξη .wav, δηλαδή τα sound waves. Για να αναπαραχθεί ένα αρχείο ήχου χρειάζεται να προστεθεί σε ένα αντικείμενο Sound Cue το οποίο επεξεργάζεται τον ήχο με ένα σύστημα με κόμβους και εντολές, παρόμοιο με τα blueprint του προγραμματισμού. Το αρχείο μετά μπορεί να τροποποιηθεί με την βοήθεια συναρτήσεων οι οποίες καταλήγουν στην έξοδο, στον κόμβο Output.

3.7.3 Ήχος – Βήματα

Για τα βήματα χρησιμοποιήθηκε ο κόμβος Random με εισόδους όλα τα διαφορετικά αρχεία ήχου με βήματα και έξοδο έναν τροποποιητή (**Modulator**). Το αποτέλεσμα είναι κάθε φορά που χρειάζεται το παιχνίδι να παίξει έναν ήχο βηματισμού να επιλέγει κάποιο αρχείο τυχαία και να κανονικοποιεί τη συχνότητα και την ένταση του. Σε ένα Sound Cue μπορεί να προστεθεί μια κλάση που περιγράφει τους ήχους μιας συγκεκριμένης κατηγορίας. Στα βήματα χρησιμοποιήθηκε η SC_Footsteps. Ακόμη, στην κλάση αυτή ορίστηκε ένας μίκτης, ο SMix_Footsteps, της κλάσης SoundMix, ο οποίος είναι υπεύθυνος για την τελική ένταση και τη χροιά των ήχων, εδώ των βημάτων. Η κλάση ήχων υποστηρίζει ρύθμιση της ακρόασης των ήχων, οπότε ορίστηκε ένα αντικείμενο ακρόασης, το Footsteps Attenuation.

3.7.4 Ήχος – Μουσική

Παρόμοια μεθοδολογία χρησιμοποιήθηκε για να περαστούν τα αρχεία μουσικής υπόκρουσης και ατμόσφαιρας στο έργο. Σε κάθε αντικείμενο Sound Cue ορίστηκε ότι ανήκει στην κλάση SC_Music και

αντικείμενο ακρόασης το Ambient όπου περιγράφει ακρόαση εσωτερικού χώρου. Για το έργο χρησιμοποιήθηκαν δύο αρχεία μουσικής, ένα στο κεντρικό μενού που χρησιμοποιείται σαν εισαγωγή στην περιπέτεια και ένα για να τονίσει την αίσθηση της δράσης κατά την διάρκεια της αποστολής.

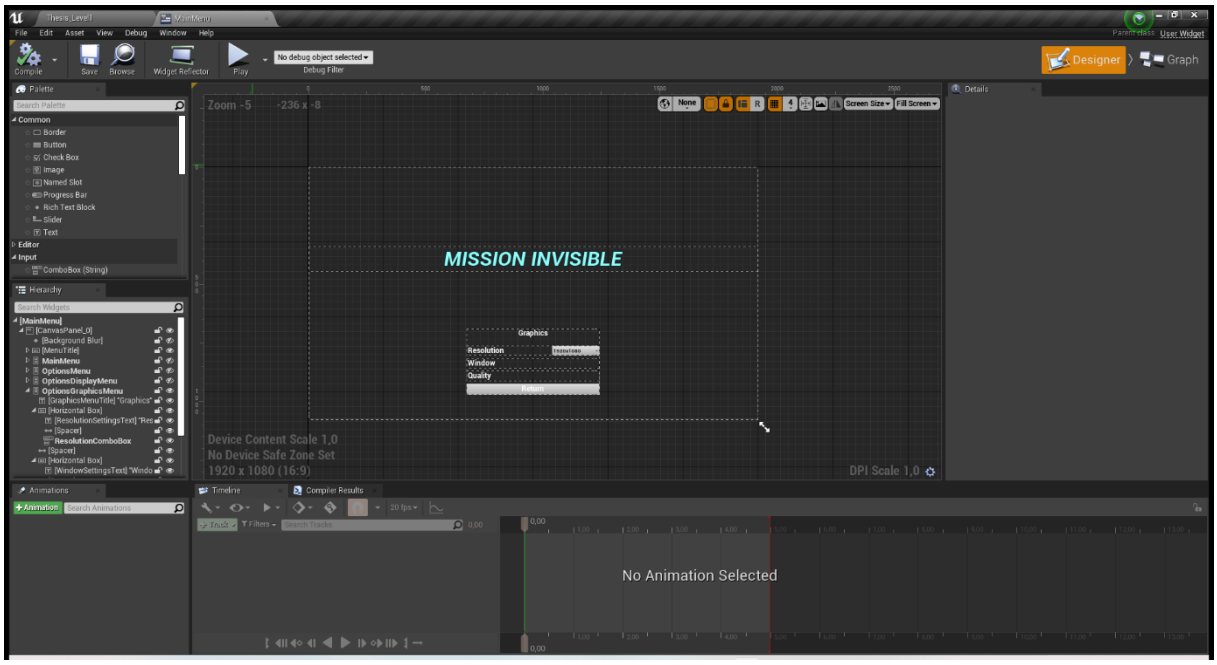
3.7.5 Ήχος – Ηχητικά εφέ

Τα ηχητικά εφέ είναι αναπόσπαστο κομμάτι ενός έργου, είτε αναφερόμαστε σε κινηματογραφικό είτε σε βιντεοπαιχνίδι. Οπότε εισήχθησαν αρχεία ήχου που περιγράφουν κίνηση από διαστημόπλοια και μηχανοκίνητα σκάφη για να ζωντανέψει το έργο κανονικά. Στους διαδρόμους χρησιμοποιήθηκε ένα εφέ εσωτερικής ατμόσφαιρας που θυμίζει πάλι μηχανοκίνητα μέρη και σωλήνες, κάτι που θα άκουγε κανείς σε διαδρόμους συντήρησης ενός σκάφους.

3.8 Δημιουργία μενού

Επόμενο βήμα μετά την κατασκευή του περιβάλλοντος ήταν να υπάρχει ένα κεντρικό μενού μέσω του οποίου να μπορεί να ξεκινήσει το έργο. Το μενού στο Unreal Engine είναι επίσης ένας χάρτης με μια δισδιάστατη εικόνα για αντικείμενο και μια κάμερα με προσανατολισμό προς την εικόνα. Επειδή όμως είναι σε χάρτη, μπορεί να γίνει μενού με τρισδιάστατο φόντο ή ακόμη και με κινούμενη εικόνα, βίντεο ή εικόνα και τρισδιάστατα εφέ. Δεν υπάρχει περιορισμός στο τι μπορεί να περιέχει ένα κεντρικό μενού πέρα από την αισθητική άποψη και το χρονικό περιθώριο στο οποίο είναι να δημιουργηθεί.

Για τις ανάγκες του έργου δημιουργήθηκε ένα μενού με το σύστημα UMG καθώς είναι πιο απλό και εύχρηστο. Με το κουμπί Add New του περιηγητή αρχείων ή με δεξί κλικ στο περιβάλλον του περιηγητή επιλέγεται ένα blueprint τύπου Widget Blueprint. Αυτό συμπεριφέρεται ως blueprint αλλά έχει για γονέα το User Widget αντί για τον Actor και κληρονομεί μεθόδους και μεταβλητές για διεπαφή χρήστη. Έπειτα, πατώντας Edit στο νέο blueprint ανοίγει το μενού επεξεργασίας που έχει παρόμοια παράθυρα με το βασικό μενού της μηχανής Unreal. Αντί για τον περιηγητή χώρου έχει ένα δισδιάστατο πλέγμα όπου ο χρήστης μπορεί να τοποθετήσει στοιχεία διεπαφής (πχ κουμπιά, εικονίδια, κείμενο, αντικείμενα στοίχισης). Καλό είναι να τοποθετηθούν τα αντικείμενα με σειρά έτσι ώστε να είναι πιο εύκολη η τροποποίηση τους και ο ορισμός ιεραρχίας να γίνεται αυτόματα. Ως πρώτο αντικείμενο τοποθετήθηκε το CanvasPanel για να καλύπτει τα όρια της εικόνας. Έπειτα τοποθετήθηκε ένα οριζόντιο κουτί που οριοθετεί τον τίτλο του έργου και εντός αυτού ένα κείμενο. Επειδή τα μενού μπορούν να περιλαμβάνουν στρώματα από περιεχόμενο όταν χρειάζεται, δημιουργήθηκαν τέσσερα υπομενού για τις ρυθμίσεις του παιχνιδιού που αλλάζουν την ιδιότητα ορατότητας όταν ενεργοποιούνται.



Εικόνα 3.9: Περιβάλλον Widget Blueprint

Εικόνα 3.10: Τελικό μενού με τρισδιάστατο φόντο

Το πρώτο υπομενού ενεργοποιείται πατώντας την επιλογή options και περιλαμβάνει τις επιλογές Display, Graphics και Audio. Σε κάθε υπομενού υπάρχει το κουμπί Return που μας επιστρέφει στο κατάλληλο υπομενού ή κεντρικό μενού αντίστοιχα. Κάθε επιλογή είναι προγραμματισμένη με blueprint καθώς δεν απαιτεί ιδιαίτερα πολύπλοκο κώδικα για να λειτουργήσει. Το ιδανικό για ένα μεγάλης κλίμακας βιντεοπαιχνίδι θα ήταν να χρησιμοποιηθεί το σύστημα Slate UI και να γίνει προγραμματισμός της εμφάνισης και της λειτουργικότητας του κάθε μενού με C++. Αλλά για μικρότερα έργα είναι πολύ χρήσιμο το εργαλείο UMG και αν χρησιμοποιηθεί σωστά, είναι δυνατό να κατασκευαστούν πολύπλοκα μενού με ωραία εμφάνιση.

3.8.1 Επιλογές – New Game

Η επιλογή New Game εκκινεί το βιντεοπαιχνίδι από την αρχή με την εντολή **Open Level** με είσοδο τον πρώτο από τους δύο χάρτες και κατόπιν **Change Menu Widget** της κλάσης **ThesisGameGameMode** έτσι ώστε το μενού που φαίνεται στην εικόνα να αλλάξει σε αυτό που είναι φορτωμένο στο **ThesisGame**. Έτσι μπορούμε να βλέπουμε τη ζωή και τα διαθέσιμα πυρομαχικά του χαρακτήρα αφού φορτωθεί ο χάρτης και όχι το κεντρικό μενού. Το **Open Level** είναι μια αυτοματοποιημένη εντολή που αποφορτώνει το τρέχον χάρτη και φορτώνει τον χάρτη της εισόδου. Χωρίς παρέμβαση από κώδικα C++ δεν μπορεί να γίνει παράκαμψη της εντολής έτσι ώστε να προστεθεί μια εικόνα καθώς φορτώνεται ο νέος χάρτης. Αυτό συμβαίνει γιατί η μηχανή παγώνει την εκτέλεση άλλων εντολών μέχρι να ολοκληρώσει την φόρτωση μιας τοποθεσίας.

3.8.2 Επιλογές – Load Game

Η επιλογή Load Game καλεί την συνάρτηση **LoadGame** της κλάσης **ThesisGameInstance** με είσοδο το όνομα της θήκης στην οποία βρίσκεται αποθηκευμένη η τελευταία κατάσταση του παιχνιδιού και τον δείκτη χρήσης της συνεδρίας παιχνιδιού. Η **LoadGame** είναι συνάρτηση περιτύλιγμα(wrapper) στην **LoadGameFromSlot** που ανήκει στις καθολικές συναρτήσεις της μηχανής (**UGameplayStatics**) και ενισχύει την βασική της λειτουργία, εκτός από το να κάνει ανάγνωση από το αρχείο, αρχικοποιεί δείκτες και σημαίες ώστε το παιχνίδι να μεταβεί σε κατάσταση φόρτωσης και να γίνει επαναφορά στο τελευταίο σημείο που βρισκόταν ο παίκτης με ότι είχε διαθέσιμο.

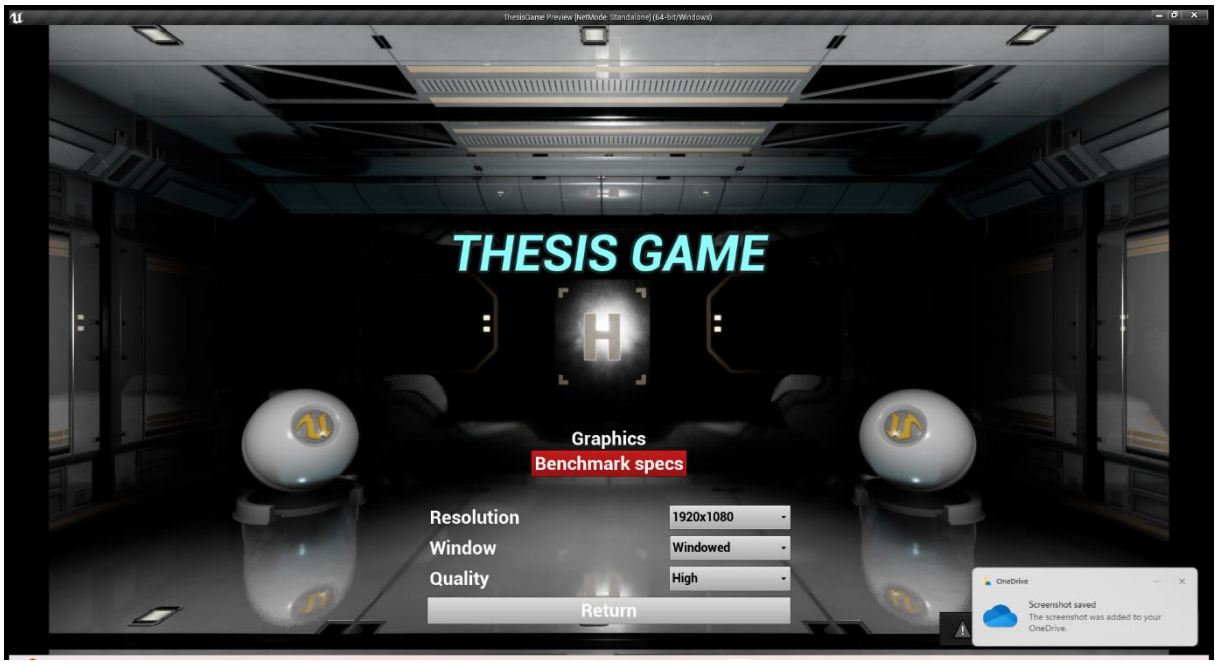
3.8.3 Επιλογές – Options – Display – Brightness

Εικόνα 3.11: Επιλογή φωτεινότητας

Η φωτεινότητα αλλάζει από την επιλογή Brightness που βρίσκεται στην υποκατηγορία Display των ρυθμίσεων. Για να επιτευχθεί καθολική αλλαγή της φωτεινότητας ασχέτως με τον εκάστοτε φωτισμό των χαρτών, χρησιμοποιήθηκε η εντολή κονσόλας “Gamma x “ σε blueprint όπου x είναι μια float τιμή. Η προεπιλεγμένη φωτεινότητα είναι 2.2 μονάδες και μπορεί να κυμανθεί από 0 έως και 10.0 . Στην

μηχανή αναφέρεται ως r.Gamma x αλλά δεν λειτουργεί με αυτήν τη σύνταξη. Το r δηλώνει ότι είναι εντολή κονσόλας όπως π.χ η εντολή r.SetRes.

3.8.4 Επιλογές – Options – Graphics – Benchmark specs



Εικόνα 3.12: Επιλογές γραφικών

Το κόκκινο κουμπί Benchmark specs μετράει την απόδοση του υπολογιστή σε πραγματικό χρόνο σύμφωνα με το τι πρόκειται να φορτώσει η μηχανή και θέτει τις επιλογές γραφικών αυτόματα. Η ποιότητα που θα θέσει ενδέχεται να είναι λίγο διαφορετική από τα προκαθορισμένα καθώς μετράει την δυνατότητα της κάρτας γραφικών να αποδώσει διάφορες εσωτερικές συναρτήσεις γραφικών, όπως το anti-alias και ray tracing.

3.8.5 Επιλογές – Options – Graphics –Resolution

Η επιλογή αυτή αλλάζει την ανάλυση προβολής του βιντεοπαιχνιδιού με drop down πλαίσιο το οποίο έχει προκαθορισμένους συνδυασμούς αναλύσεων. Οι επιλογές είναι σε συνδυασμό X και Y pixels και είναι οι ακόλουθες: 800 x 600, 1280 x 720, 1366 x 768, 1600 x 1400, 1920 x 1080 και 2140 x 1800. Οι συνδυασμοί είναι βασισμένοι σε αναλογία οθόνης 16:9. Κατά την επιλογή μιας ανάλυσης γίνεται μετατροπή των τιμών αναλογίας σε ακεραίους με διαχωρισμό στο x. Δηλαδή το 1920x1080 χωρίζεται σε 1920 για X και 1080 για Y. Έπειτα εισάγονται οι τιμές στην συνάρτηση **Set Screen Resolution** και ακολουθεί κλήση της συνάρτησης **Apply Settings** . Μπορεί να γίνει και με εντολή κονσόλας αλλά προτιμάται οι επιλογές γραφικών να γίνονται μέσω της διεπαφής των **Game User Settings**. Για παράδειγμα με την r.SetRes 1920x1080 θα μπορούσε να τεθεί η ανάλυση FHD στη τρέχουσα συνεδρία. Η εκτέλεση οποιασδήποτε εντολής κονσόλας γίνεται με την εντολή blueprint **Execute Console Command** με είσοδο κάποια συμβολοσειρά με την εντολή κονσόλας.

3.8.6 Επιλογές – Options – Graphics – Window

Η επιλογή αυτή αλλάζει τον τρόπο διαχείρισης του παραθυρικού περιβάλλοντος του παιχνιδιού. Υπάρχουν τρεις επιλογές:

- Windowed Mode, το παιχνίδι παίζει σε παράθυρο μικρότερο ή ίσο της μέγιστης ανάλυσης της οθόνης
- Fullscreen Mode, το παιχνίδι παίζει σε πλήρη οθόνη και αφοσιώνεται η διεργασία στην προβολή του παιχνιδιού. Ο μόνος τρόπος να αλλάξει περιβάλλον ο χρήστης είναι μέσω του συνδυασμού Alt+Tab
- Borderless Fullscreen Mode, το παιχνίδι παίζει σε παράθυρο ίσο της μέγιστης ανάλυσης της οθόνης αλλά δεν απομονώνεται και επιτρέπει την εμφάνιση τρίτων παραθύρων από πάνω του.

Και οι τρεις επιλογές εκτελούν την συνάρτηση **Set Fullscreen Mode** του **Game User Settings**. Έπειτα γίνεται εφαρμογή των νέων ρυθμίσεων με την συνάρτηση **Apply Resolution Settings**. Για να γίνει σωστά η μετάβαση από παραθυρικό περιβάλλον σε πλήρη οθόνη, η μηχανή ελέγχει αν ο χρήστης υποστηρίζει την ανάλυση 1920x1080 και την θέτει στην ρύθμιση ανάλυσης, διαφορετικά θέτει την πρώτη διαθέσιμη που υποστηρίζει.

3.8.7 Επιλογές – Options – Graphics – Quality

Η ποιότητα των γραφικών ρυθμίζεται μέσω μιας συλλογής από αυτόματες διαδικασίες που παρέχει η μηχανή. Το αντικείμενο που προέρχεται από την συνάρτηση blueprint **Get Game UserSettings** χρησιμοποιείται για να καλέσει με την σειρά του την συνάρτηση **Set Overall Scalability Level** με είσοδο τον ακέραιο 0 έως 4 με 0 τη χειρότερη ποιότητα και 4 την ύψιστη. Τέλος η επιβεβαίωση της ρύθμισης γίνεται με την συνάρτηση **Apply Settings** του **Game User Settings**.

3.8.8 Επιλογές – Options – Audio

Οι ρυθμίσεις στον ήχο γίνονται λίγο διαφορετικά. Χρησιμοποιείται η καθολική συνάρτηση blueprint **Set Sound Mix Class Override** με την οποία δίνονται νέες τιμές σε μια κλάση ήχων. Στο παρόν έργο ρυθμίζεται μόνο η ένταση του ήχου. Για να αποθηκευτεί η ρύθμιση ήχου πρέπει να κληθεί η συνάρτηση **Push Sound Mix Modifier**. Σε αυτό το έργο παρέχονται τέσσερις διαφορετικές επιλογές ήχου, Master, Sounds, Footsteps και Music. Ο ήχος κυμαίνεται από 0 έως 1 με προκαθορισμένη τιμή το 0.5.

3.8.9 Επιλογές – Quit Game

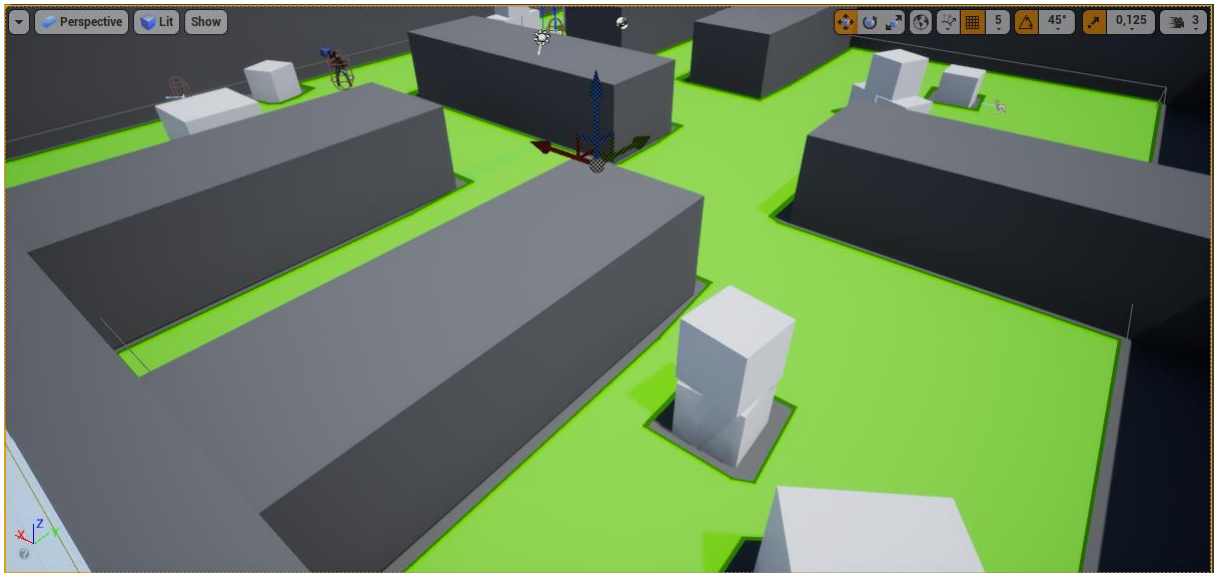
Η επιλογή Quit Game τερματίζει την συνεδρία του βιντεοπαιχνιδιού και κλείνει την εφαρμογή. Γίνεται με αυτόματο τρόπο με την εντολή blueprint του **Quit Game** του System Library με εκτέλεση σε blueprint.

3.9 Ανάλυση ψηφιακού περιβάλλοντος

Μετά την ολοκλήρωση του χώρου από τη σκοπιά του σχεδιαστή, σειρά έχει η ανάλυση του περιβάλλοντος όπως θα το χρησιμοποιεί ο υπολογιστής. Αν ο χώρος περιλαμβάνει πλατφόρμες και διαφορετικού ύψους επίπεδα, θα μπορεί να τα φτάσει ένας χαρακτήρας του υπολογιστή; Αν ο χώρος έχει σε πολύ πυκνή έκταση διακοσμητικά, θα μπορεί να περπατήσει από ανάμεσα τους ο ίδιος υποτιθέμενος χαρακτήρας; Για την επιτυχή απάντηση σε αυτές τις ερωτήσεις θα πρέπει να κατασκευαστεί ένας χάρτης πλοήγησης (navigation mesh) [54] τον οποίο θα χρησιμοποιήσουν οι χαρακτήρες του υπολογιστή (NPCs) για να μετακινηθούν στον χώρο. Η ανάλυση του περιβάλλοντος συνδυάζει γνώση από τον τρόπο λειτουργίας της τεχνητής νοημοσύνης και της κατασκευής χαρτών πλοήγησης. Πολλές φορές επειδή ένας χαρακτήρας του υπολογιστή επιλέγει τον συντομότερο δρόμο, πρέπει να αποδοθεί μεγαλύτερο κόστος στην διαδρομή αυτή καθώς δεν είναι η επιθυμητή. Δηλαδή, μπορεί να φτάσει από ένα δωμάτιο σε ένα άλλο ανοίγοντας μια πόρτα και διαβαίνοντας ένα χωλ ή μπορεί να περάσει πάνω από διακοσμητικά, πάγκους, τραπέζια και τελικά να πηδήξει από ένα παράθυρο

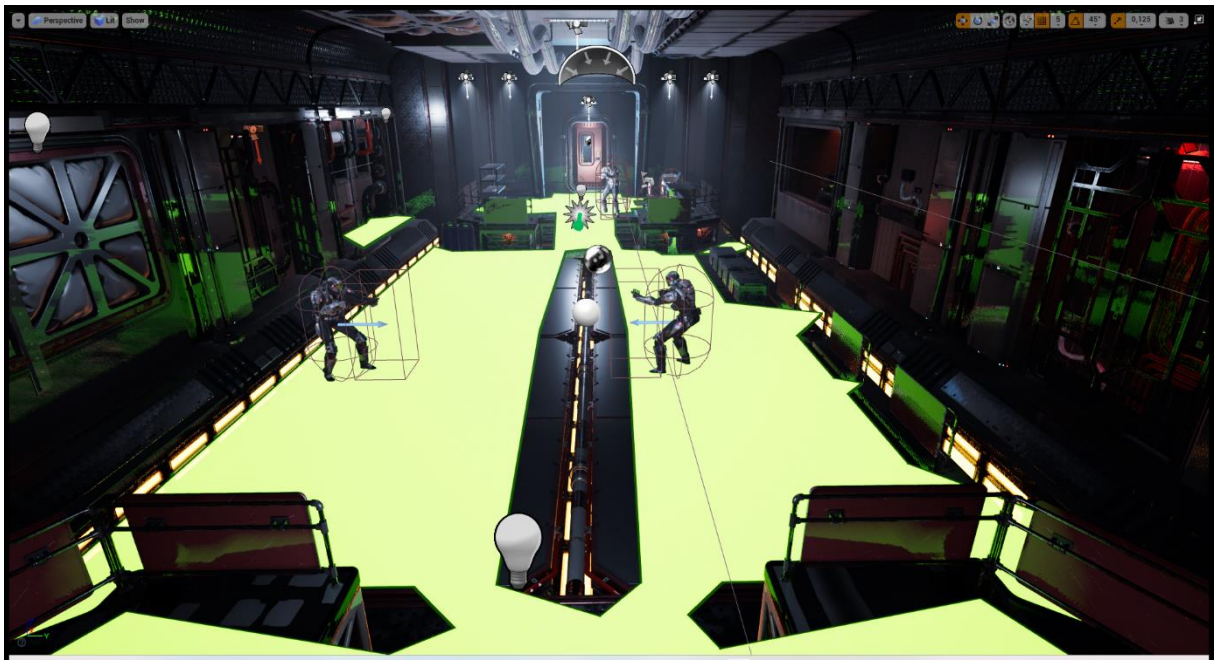
γιατί τυχαίνει να είναι πιο σύντομη η απόσταση. Σε αυτές τις περιπτώσεις τοποθετούνται τεχνητά εμπόδια για να είναι ελεγχόμενες οι επιλογές κίνησης των χαρακτήρων.

Ένας τέτοιο χάρτη πλοήγησης τοποθετήθηκε στο χώρο για κάθε έναν από τους διαθέσιμους χάρτες του έργου σύμφωνα με τις εκάστοτε ανάγκες. Στο δοκιμαστικό χώρο ο χάρτης είναι ένα απλό τετράγωνο χωρίς ιδιαίτερα σημεία τροποποίησης ενώ στους χάρτες που λαμβάνει χώρα το κανονικό παιχνίδι υπάρχουν τεχνητά και φυσικά εμπόδια σημειωμένα για να μην πέφτουν οι αντίπαλοι πάνω σε διακοσμητικά. Η δοκιμή πλοήγησης έγινε με έναν πράκτορα πλοήγησης με ύψος 1,80 m και διάμετρο 55 cm, όσο μέγεθος έχουν και οι κάψουλες των κανονικών χαρακτήρων. Το πράσινο πεδίο σηματοδοτεί τα σημεία όπου ο χάρτης πλοήγησης είναι έγκυρος και μπορούν να μετακινηθούν πάνω του χαρακτήρες. Το πεδίο αυτό εμφανίζεται πατώντας το P στο μενού προβολής. Από προεπιλογή είναι κρυφό και φαίνονται μόνο τα όρια του αντικειμένου.



Εικόνα 3.13 : NavMesh στο δοκιμαστικό χάρτη

Για να συμπεριφέρονται καλύτερα υπό συνθήκες, χρησιμοποιήθηκε το σύστημα Environment Query System (EQS) που στην έκδοση της εργασίας ήταν ακόμα σε δοκιμαστικό στάδιο αλλά λειτουργεί επαρκώς τις περισσότερες περιπτώσεις.



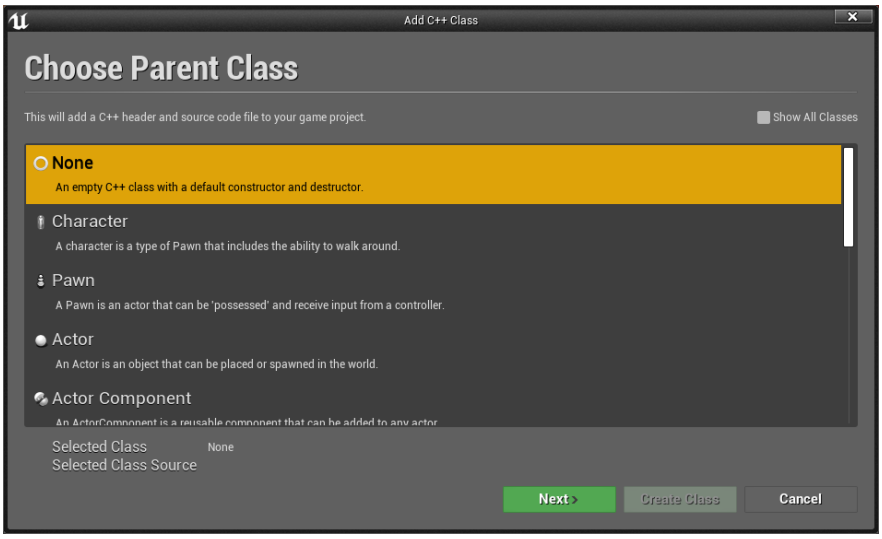
Εικόνα 3.14: Navmesh στο πρώτο χάρτη

3.10 Δημιουργία και ανάπτυξη χαρακτήρα

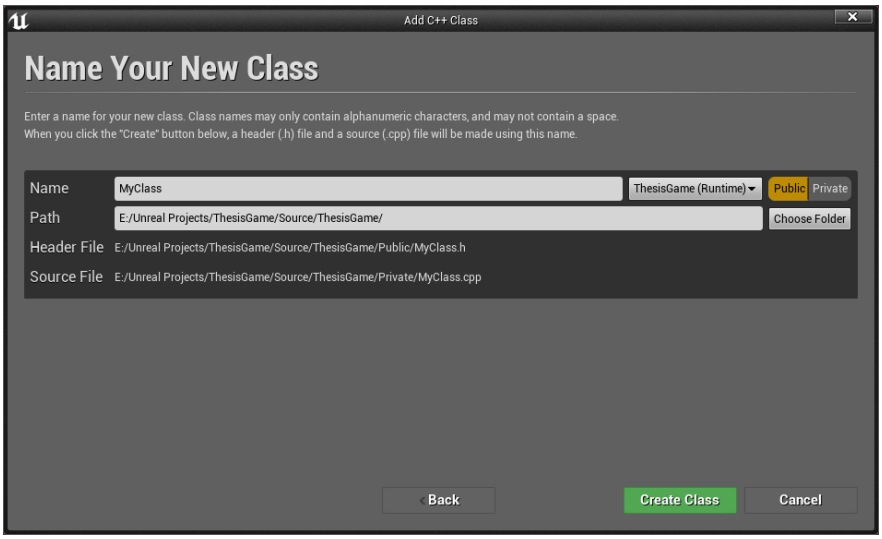
Ένας λειτουργικός χαρακτήρας που μπορεί να κινηθεί μέσα σε τρισδιάστατο χώρο είναι συνήθως παράγωγο αντικείμενο της κλάσης Pawn (γενική κλάση) ή Character (ειδικευμένη κλάση για δίποδα). Σε κάποιες περιπτώσεις μπορούν να είναι και αντικείμενα της κλάσης Actor, της βασικής κλάσης όλων των αντικειμένων που είναι ικανά για αλληλοεπίδραση με το χώρο σε κάποιο βαθμό. Λειτουργικός χαρακτήρας θεωρείται ένας χαρακτήρας με κάψουλα που οριοθετεί τον όγκο του και δεν έχει κανένα σφάλμα σε οποιοδήποτε μπλοκ κώδικα διαθέτει. Στην συγκεκριμένη εργασία χρησιμοποιήθηκε σαν παράδειγμα ο χαρακτήρας πρώτου προσώπου που δίνει έτοιμο η μηχανή κατά την επιλογή προτύπου βολών πρώτου προσώπου καθώς επίσης και ο χαρακτήρας του έργου Shooter Game της Epic. Ο χαρακτήρας θεωρείται πρωταγωνιστής καθώς διαθέτει μια θέση μνήμης για τοποθέτηση κάμερας πάνω του και ένα στοιχείο χειριστή παίκτη, το Player Controller.

3.10.1 Παίκτης – Player Class

Ο παίκτης είναι ένα αντικείμενο παράγωγο της κλάσης Character και γράφηκε σε C++, σε μια νέα κλάση με όνομα Playable Character. Για την δημιουργία μιας νέας C++ κλάσης ακολουθείται μια διαφορετική μέθοδος από τη συνηθισμένη. Πατώντας το πράσινο κουμπί Add New επιλέγουμε την επιλογή New C++ class και εμφανίζεται ένα μενού με επιλογή της γονικής κλάσης. Στη συνέχεια, επιλέγουμε όνομα κλάσης, τοποθεσία και αν θα είναι Public ή Private, δηλαδή αν θα μπορεί να χρησιμοποιηθεί από άλλα συστατικά της μηχανής γραφικών ή μόνο από αναφορές του εαυτού της.



Εικόνα 3.15: Δημιουργία κλάσης C++



Εικόνα 3.16: Ονομασία κλάσης C++

Είναι εύκολη η δημιουργία κλάσεων καθώς η μηχανή συμπληρώνει τα απαραίτητα πεδία στην κεφαλίδα για να τρέξει σωστά η νέα κλάση αλλά δεν υπάρχει δυνατότητα διαγραφής της μέσω της μηχανής. Χειροκίνητα γίνεται με διαγραφή των αρχείων κώδικα, διαγραφή του φακέλου Intermediate στο τρέχον έργο και εκ νέου μεταγλώττιση του έργου. Δεν ενδείκνυται όμως αυτή η μέθοδος καθώς μπορεί να παράγονται σφάλματα κατά την μεταγλώττιση του και να μην μπορεί πλέον να σωθεί καμία αλλαγή.

3.10.2 Παίκτης - Δομητής

Στην νέα κλάση παίκτη, κατά την κατασκευή της κλάσης (constructor), πρώτον ελέγχεται αν ο χαρακτήρας είναι έγκυρος και φορτωμένος στην εφαρμογή. Αν παραλειφθεί ο έλεγχος είναι πιθανό να φορτωθεί πιο αργά από την ταχύτητα εκτέλεσης των επόμενων γεγονότων, δίνοντας πρόσβαση σε κομμάτια κώδικα που εξαρτώνται από το αντικείμενο πρωταγωνιστή και εφόσον δείχνουν σε κενή τιμή, ο κώδικας θα αποτύχει και η εφαρμογή θα καταρρεύσει (Crash). Ο δείκτης της κάμερας γεμίζει με ένα στοιχείο κάμερας **UCameraComponent** και ρυθμίζεται κατάλληλα η γωνία του φακού (Field of View), η εστιακή απόσταση (Focal Length) και η θέση στην οποία θα βρίσκεται η κάμερα (Component Translation & Rotation). Έπειτα προσαρτιέται στον κόμβο ρίζας του χαρακτήρα με την συνάρτηση

AttachToComponent. Η συνάρτηση αυτή είναι που προκαλεί το περισσότερο πρόβλημα σε μη ορθά αρχικοποιημένα αντικείμενα, γιατί προσπαθεί να πάρει παραμέτρους φυσικής τοποθεσίας, σε καθολικό ή τοπικό βαθμό και να τις μεταφέρει στο αντικείμενο στόχο έτσι ώστε να ενωθούν και να κινούνται μαζί σαν μια οντότητα, οπότε αν προσπαθήσει να πάρει παραμέτρους από κενούς δείκτες αποτυγχάνει η εφαρμογή. Στην πράξη, προκαλούνται τυχαία σφάλματα κατά την τοποθέτηση αντικειμένων με αυτό το λάθος στο χώρο ή κατά την φόρτωση ενός χώρου και η κονσόλα δείχνει στο προβληματικό αντικείμενο, αλλά σε άλλο σημείο γιατί αγνοείται η κλήση μεθόδων σε κενούς δείκτες. Η ίδια διαδικασία χρησιμοποιείται στην πορεία για να περαστεί μια τρισδιάστατη αναπαράσταση στο αντικείμενο του πρωταγωνιστή. Γίνεται έλεγχος εγκυρότητας στο αντικείμενο, δημιουργείται ένα αντικείμενο σκελετού, ρυθμίζεται ποιος σκελετός θα χρησιμοποιηθεί και το τρισδιάστατο μοντέλο που απεικονίζεται με αυτόν τον σκελετό, η θέση του στο χώρο και ενώνεται με την κάμερα αντί για την ρίζα γιατί με αυτόν τον τρόπο δεν χρειάζεται επιπλέον κώδικας για να κινείται ο χαρακτήρας όταν θέλουμε να κοιτάξουμε στον χώρο με τη κάμερα. Επίσης, η θέση της κάμερας στον Z άξονα καθορίζει και το ύψος των ματιών του χαρακτήρα ή γενικότερα ολόκληρου του χαρακτήρα. Έτσι δεν χρειάζεται ψηλότερο τρισδιάστατο μοντέλο ούτε διαφορετικό σκελετό για να αλλάξει ύψος. Τέλος θέτονται οι αρχικές τιμές για τις παραμέτρους του χαρακτήρα. Ο χαρακτήρας διαθέτει έναν δείκτη δομής δεδομένων που περιγράφει το όπλο που θα κρατήσει (Weapon Struct), έναν πίνακα δεικτών κλάσεων για τα πράγματα του (Inventory), μια σειρά από Boolean μεταβλητές που περιγράφουν ιδιότητες κατάστασης για τον αν τρέχει, είναι στον αέρα, σκοπεύει, γεμίζει το όπλο του και αν είναι ζωντανός ή όχι. Ο γενικός κανόνας που ακολουθήθηκε είναι οι μεταβλητές κατάστασης να μην αλλάζουν σε τυχαία σημεία μέσα στον κώδικα αλλά μόνο μέσω μεθόδων, με εξαίρεση τις προκαθορισμένες τιμές τους που θέτονται στο constructor. Έτσι γίνεται καλύτερος έλεγχος καθώς η συγγραφή κώδικα του χαρακτήρα χωρίζεται σε επιμέρους προβλήματα. Μια βελτίωση πάνω σε αυτό ήταν κάποιες από τις μεταβλητές που έπρεπε να ρυθμιστούν με δοκιμές να είναι εμφανείς στο αντικείμενο blueprint και να μπορούν να πάρουν από εκεί μια προκαθορισμένη τιμή. Αν μια μεταβλητή οριστεί να είναι εμφανής και διαθέσιμη για τροποποίηση στο Blueprint με το όρισμα **UBlueprintEditable** και ταυτόχρονα έχει συνδεθεί με κάποια αλλαγή που γίνεται σε ιδιότητα στον constructor, τότε καθώς αλλάζει η μεταβλητή εντός της μηχανής, χωρίς καν να τεθεί η μηχανή σε κατάσταση προσομοίωσης, ενημερώνεται ο αντικείμενο αυτόματα και το αποτέλεσμα φαίνεται αμέσως στο παράθυρο προβολής. Για παράδειγμα, μια μεταβλητή Integer με όρισμα **UBlueprintEditable** ορίζεται ότι περιέχει την τιμή 10. Στον constructor η μεταβλητή αυτή είναι είσοδος για την αρχική μετατόπιση του αντικειμένου (offset). Οπότε το αντικείμενο εμφανίζεται στον χώρο με offset 10. Αν αυτό αλλάξει εντός της μηχανής, στο παράθυρο προβολής φαίνεται το αντικείμενο με το διαφορετικό offset και έτσι μπορεί να γίνει καλύτερη ρύθμιση σε τιμές που δεν είναι σίγουρος ο σχεδιαστής. Τέτοιες μεταβλητές ήταν η αρχική μετατόπιση και περιστροφή της κάμερας του πρωταγωνιστή γιατί για κάθε μοντέλο χρειάζεται διαφορετικές τιμές.

3.10.3 Παίκτης – Μετά την αρχικοποίηση (μέθοδος PostInitializeComponent)

Σε επόμενο διάστημα αφού ολοκληρωθεί η αρχική επεξεργασία του πρωταγωνιστή έπρεπε να τεθούν αρχικές τιμές στα πράγματα του και στα όπλα που θα κρατάει. Για τον σκοπό αυτό χρησιμοποιείται η μέθοδος **PostInitializeComponents** που καλείται αυτόματα από την μηχανή όταν το αντικείμενο είναι σε κατάσταση φόρτωσης στον χώρο και μόνο εντός της συνεδρίας παιχνιδιού (GamePlay). Είναι η λύση στο πρόβλημα που δημιουργείται όταν πρέπει να αρχικοποιηθούν αντικείμενα τα οποία καταλαμβάνουν ήδη χώρο στη μνήμη αλλά δεν πρέπει να φανούν στην εικόνα του παίκτη πριν ολοκληρωθεί η παραμετροποίηση τους. Ο οπλισμός του χαρακτήρα είναι ένας τέτοιος πίνακας κλάσεων που γεμίζει κατά την φάση φόρτωσης και όχι στον constructor γιατί πρέπει να δημιουργηθεί πρώτα. Γίνεται κλήση στη μέθοδο **SpawnDefaultInventory** στο επόμενο πλαίσιο από το τρέχον με τη βοήθεια της

συνάρτησης **SetTimerForNextTick**. Έπειτα γίνεται αναζήτηση για προκαθορισμένες τιμές που δόθηκαν στις κλάσεις όπλων και αν βρεθεί έστω μια, γεμίζει ένα κελί του πίνακα και καλεί την συνάρτηση φόρτωσης του όπλου που ακολουθείται από την μέθοδο εξόπλισης (**Equip**) του χαρακτήρα με αυτό το όπλο. Μετά το αίτημα για εύρεση των όπλων, δίνονται πόντοι ζωής στον χαρακτήρα ίσοι με ότι έχει οριστεί σαν προεπιλογή με τη συνάρτηση **GetBaseHealth**, ελέγχεται αν ο χαρακτήρας είναι σε πρώτο ή τρίτο πρόσωπο για να εμφανιστεί η κατάλληλη γεωμετρία σώματος και καταχωρείται σαν πηγή αισθήσεων ακοής και όρασης στον πίνακα Perception Stimuli με την γενική συνάρτηση **RegisterForPerceptionStimuli** και ορίσματα τις κλάσεις **UAISense_Sight** και **UAISense_Hearing**. Το **StimuliSource** είναι ένας διαχειριστής αισθήσεων που στέλνει μηνύματα broadcast στη μηχανή για κάποια αίσθηση [55]. Αν ένας χαρακτήρας ρυθμιστεί να ακούει τα μηνύματα αυτά, τότε θα λάβει το γεγονός **OnTargetSenseUpdated**, το οποίο υλοποιήθηκε στο έργο αυτό για τις ανάγκες αντίληψης των αντιπάλων.

3.10.4 Παίκτης - Κίνηση

Ο χαρακτήρας μπορεί να κινείται με τα γεγονότα εισόδου του χρήστη μέσω των μεθόδων **BindAction** και **BindAxis** του στοιχείου Movement Component του χαρακτήρα. Για κάθε μια δραστηριότητα που μπορεί να κάνει ο παίκτης πρέπει να υπάρχει η αντίστοιχη συνάρτηση που δένει την δραστηριότητα με μια ή περισσότερες εισόδους. Υπάρχει ένας πίνακας με εισόδους από διάφορες υποστηριζόμενες συσκευές που μπορεί να χρησιμοποιηθεί για να δέσει τις εισόδους με τις δραστηριότητες του χαρακτήρα.

Στις ρυθμίσεις της μηχανής, στην κατηγορία εισόδων προστέθηκε μια γραμμή πίνακα για κάθε είσοδο και επιλέχθηκε ποιο πλήκτρο σε ποιο μέσο θα εκτελεί την δραστηριότητα. Έπειτα δόθηκε ένα όνομα στη δραστηριότητα. Έτσι η μηχανή κατά την συγκεκριμένη είσοδο του χρήστη δημιουργεί ένα γεγονός με το όνομα που δόθηκε στη δραστηριότητα. Το κάθε αντικείμενο στον χώρο είναι υπεύθυνο για τον χειρισμό του γεγονότος μόνο αν είναι ενεργή η επιλογή να δέχεται εισόδους από τον παίκτη. Έτσι όλα τα αντικείμενα απορρίπτουν οποιοδήποτε γεγονός εισόδου εκτός αν ενεργοποιηθεί η δυνατότητα και είναι πολύ ελαφρύ το πρόγραμμα, σε σχέση με την πολυπλοκότητα των εισόδων που δέχεται σε κάθε πλαίσιο. Στο συγκεκριμένο έργο ορίστηκαν τα εξής:

- ένα κουμπί για άλμα (Jump)
- τέσσερα κουμπιά για κίνηση δεξιά – αριστερά, εμπρός και πίσω στο χώρο (Move Left/Right, Forward/Backward)
- ένα για ακριβής σκόπευση (Iron Sights)
- ένα για βολή (Fire)
- ένα για ανεφοδιασμό του όπλου (Reload)
- ένα για σπριντ (Sprint)
- δύο προσημασμένοι άξονες κίνησης του ποντικιού για χειρισμό της κάμερας και περιστροφή του χαρακτήρα (Look Up/Down, Turn Left/Right)

Η κίνηση του χαρακτήρα και της κάμερας στο συγκεκριμένο έργο έγινε με μίξη των συναρτήσεων **Tick**, **BindAction** και **BindAxis** αναλόγως με τις ανάγκες. Περιγράφηκε με έτοιμες συναρτήσεις που διαθέτει η μηχανή, την **MoveForward**/**MoveRight** για κίνηση και την **AddControllerYawInput**/**AddControllerPitchInput** για περιστροφή, με ορίσματα την κατεύθυνση κίνησης και την ποσότητα που δίνονται από την είσοδο χρήστη. Πιο αναλυτικά, η κατεύθυνση είναι διάνυσμα και δίνεται αν δημιουργηθεί ένα διάνυσμα με τιμή προς κάποιο άξονα ίση με την τιμή της εισόδου από τον παίκτη. Αν θέλουμε κίνηση εμπρός και πίσω, αρκεί να πάρουμε τον πίνακα

περιστροφής του χαρακτήρα με τη μέθοδο **GetControlRotation** της κλάσης Player Controller του χειριστή που ανήκει στον χαρακτήρα και να τον πολλαπλασιάσουμε κατά τον άξονα X με πρόσημο ότι ορίζει η είσοδος. Έτσι ο χαρακτήρας κινείται μπροστά με θετικές τιμές, πίσω με αρνητικές αντίστοιχα με σημείο αναφοράς την κατεύθυνση του χαρακτήρα. Η παραπάνω μέθοδος κίνησης έχει την ίδια λειτουργικότητα με την έτοιμη, με την διαφορά ότι ο προγραμματιστής μπορεί να εισάγει οποιαδήποτε συνάρτηση φιλτραρίσματος θέλει στην κίνηση. Μπορεί να καθορίσει την ταχύτητα κίνησης και την κατεύθυνση πριν αυτές δοθούν προς επεξεργασία στην μηχανή. Αυτός ο τρόπος επιλέχθηκε για τον πρωταγωνιστή έτσι ώστε να υπάρχει ελευθερία για τροποποίηση της κίνησης σύμφωνα με τις ανάγκες που τυχόν να προκύψουν. Και στην πορεία όντως προέκυψαν ανάγκες όπως ο χαρακτήρας να αναπτύσσει ταχύτητα σταδιακά και να επιβραδύνει σταδιακά πριν σταματήσει πλήρως τη κίνηση του. Επίσης χρειάστηκε να προστεθεί μια καθυστέρηση στις εισόδους για να μην μπορεί ο παίκτης να ακυρώσει την εκτέλεση κινήσεων όποτε θελήσει, χωρίς να νιώθει ότι το παιχνίδι δεν αποκρίνεται σωστά.

Η κατεύθυνση του χαρακτήρα αλλάζει όταν αλλάζει η περιστροφή της κάμερας μέσω του ποντικιού. Αυτό έγινε με την βοήθεια των δύο συναρτήσεων μηχανής, **AddControllerYawInput** και **AddControllerPitchInput** που αλλάζει την περιστροφή του χειριστή του χαρακτήρα στους άξονες X και Y αλλά εμπλουτισμένες με δυνατότητα αλλαγής της ευαισθησίας στην περιστροφή και κάποια όρια ως προς το πόσο μπορεί να γυρίσει η κάμερα, ιδιαίτερα στον άξονα Y. Δόθηκε προσοχή έτσι ώστε ο προσανατολισμός του χειριστή να είναι σταθερός και να λειτουργεί το ίδιο άσχετα με την κατεύθυνση του χαρακτήρα. Διαφορετικά όταν ο χαρακτήρας γυρνούσε 180 μοίρες το εμπρός/πίσω στην κίνηση θα ήταν ανεστραμμένο, κάτι που γίνεται πολύ εύκολα αν γίνει απευθείας αλλαγή στην περιστροφή του χαρακτήρα ή του χειριστή χωρίς την παρέμβαση της έτοιμης συνάρτησης. Και στις δύο περιπτώσεις, κίνηση και περιστροφή τα αποτελέσματα της εισόδου, με ή χωρίς επεξεργασία έπρεπε να κανονικοποιηθούν σε τιμές γύρω από το διάστημα [0, 1], διαφορετικά προκαλούσαν υπερβολική μετατόπιση και περιστροφή.

Ο χαρακτήρας από προεπιλογή βαδίζει γρήγορα με τιμή που αρχίζει από 0 και αγγίζει σταδιακά το 0.4. Αν σταματήσει θα επιβραδύνει την μετατόπιση του σταθερά κατά 0.1 έως ότου φτάσει 0, οπότε σε 4 πλαίσια σταματάει πλήρως την κίνηση του. Όταν τρέχει, πρώτα θα βαδίσει έτσι ώστε να πάρει επιτάχυνση και ύστερα θα αυξήσει ταχύτητα, έτσι δεν μπορεί να τρέξει πριν μετατοπιστεί ελάχιστα για να πάρει την απαραίτητη επιτάχυνση. Όταν σταματάει να τρέχει επιβραδύνει πιο αργά γιατί είχε μεγαλύτερη ταχύτητα. Όταν σκοπεύει περπατάει αργά με τιμή 0.15 για να δώσει προτεραιότητα στον στόχο του και όταν σκοπεύει ενώ έτρεχε επιβραδύνει πιο γρήγορα έως ότου φτάσει στο 0.15. Αυτές οι μικρές αλλαγές στην συνάρτηση κίνησης έδωσαν μια φυσικότητα στην αίσθηση χειρισμού του πρωταγωνιστή καθώς δεν άλλαζε πλέον απότομα ταχύτητες σαν μηχανοκίνητο αλλά σαν ζωντανός οργανισμός. Οι σταδιακές αλλαγές στις τιμές έγιναν σαν τμήμα της συνάρτησης **Tick** του πρωταγωνιστή με την βοήθεια της μαθηματικής συνάρτησης **Lerp** με ορίσματα την αρχική τιμή, την τελική τιμή και ένα βήμα. Η συνάρτηση **Lerp** προκαλεί γραμμική μεταβολή σε μια τιμή με σταθερό βήμα. Προτιμήθηκε έναντι άλλων μεθόδων μεταβολής διότι αν υπήρχε απότομη αλλαγή ή κάποια καθυστέρηση στην μεταβολή της ταχύτητας λόγω εξομάλυνσης, θα προκαλούσε την καθυστέρηση αλλαγής ταχύτητας του χαρακτήρα και θα έμοιαζε περισσότερο με αυτοκίνητο παρά με άνθρωπος. Εκτός από την φυσικότητα στην κίνηση, αυτές οι αλλαγές μπορούν να θεωρηθούν ως μια μηχανική στο παιχνίδι γιατί δεν επιτρέπει σε κάποιον χαρακτήρα να σκοπεύει με πλήρη ευστοχία και μικρή διασπορά ενώ τρέχει γρήγορα, γεγονός που αυτομάτως τον καθιστά ευάλωτο σε αντίπαλα πυρά. Αν όμως σκοπεύσει, εκτελέσει μια βολή και κατόπιν τρέξει αρκετά γρήγορα θα έχει αρκετό όφελος στην κίνηση του ενώ εκτέλεσε μια εύστοχη βολή και θα φτάσει σε κάλυψη πιο γρήγορα.

3.10.5 Παίκτης – Γεγονότα ζημιάς και θανάτου

Ο πρωταγωνιστής έχει μια float μεταβλητή που περιγράφει την υγεία του. Ξεκινάει με μία προκαθορισμένη τιμή μέσω του constructor ή μέσω μιας αναφοράς στο χώρο και αν δεχτεί ζημιά από κάποιον παράγοντα μειώνεται. Η ζημιά περιγράφεται σε ένα έτοιμο γεγονός της μηχανής, το γεγονός **TakeDamage** το οποίο καλείται κάθε φορά που θέλουμε να στείλουμε την πληροφορία της ζημιάς σε έναν άλλο χαρακτήρα της κλάσης Actor. Είναι abstract γεγονός και από μόνο του προσφέρει απλά την επικοινωνία μεταξύ χαρακτήρων έτσι ώστε να υπάρχει το ενδεχόμενο για ανάπτυξη ενός συστήματος δράσης-αντίδρασης μεταξύ χαρακτήρων. Αν δεχτεί ζημιά ο χαρακτήρας πρώτα ελέγχεται το ποσοστό της ζημιάς και αν ξεπερνάει την διαθέσιμη υγεία του τότε καλείται η μέθοδος θανάτου του χαρακτήρα για να χειριστεί τις μηχανικές πίσω από την αλλαγή κατάστασης. Αν έχει ακόμα διαθέσιμη υγεία τότε παίζει ένα εφέ στην εικόνα που σηματοδοτεί ότι ο χαρακτήρας δέχτηκε κάποια σφαίρα και κάνει θόρυβο. Ο θόρυβος συμβάλλει στην ολική αντίληψη των υπόλοιπων χαρακτήρων, αν είναι αντίπαλοι και δεν έχουν δει τον παίκτη είναι πιθανό να εξετάσουν τον θόρυβο και να μαζευτούν όλοι γύρω από τον παίκτη, αν είναι σύμμαχοι θα προσπαθήσουν να καλύψουν τον παίκτη που δέχεται πυρά. Καθώς παίζει το εφέ στην κάμερα η μηχανή σπρώχνει τον ήρωα προς τα πίσω με την συνάρτηση **ApplyDamageMomentum** έτσι ώστε ο χαρακτήρας να μην μπορεί να έχει πλήρη έλεγχο της κίνησης του. Αυτό αποτελεί επίσης μηχανική γιατί δίνει βάρος στο γεγονός να δεχτεί βολή ο παίκτης καθώς αν αυτό συμβεί, είναι δύσκολο να ξεφύγει από επερχόμενες βολές.

Όταν ο χαρακτήρας πεθάνει καλείται η συνάρτηση **Die** που ελέγχει το αν ο χαρακτήρας βρίσκεται σε κατάσταση θανάτου ή όχι και έπειτα αποθηκεύει εσωτερικά τον χαρακτήρα που έριξε την τελευταία βολή ενώ ενεργοποιεί το γεγονός **OnDeath**. Το γεγονός **OnDeath** είναι αφηρημένη συνάρτηση της μηχανής και είναι υπεύθυνο για την αλλαγή κατάστασης του χαρακτήρα. Όταν ένας χαρακτήρας πεθαίνει πρέπει να σταματήσουν οι κινήσεις του και να παίξει ένα τελευταίο αρχείο κίνησης που σηματοδοτεί τον θάνατο του. Επιπλέον, ο θάνατος του χαρακτήρα δηλώνεται στο GameMode έτσι ώστε το παιχνίδι να εκτελέσει κατάλληλες ενέργειες σύμφωνα με το αν ανήκει στη κλάση παίκτη ή εχθρού.

Σε πιο σύγχρονα βιντεοπαιχνίδια εκτός από την κατάσταση ακινησίας ενεργοποιείται μια γεωμετρία βαρύτητας που δίνει πιο φυσική κίνηση στα μέλη του σώματος ενός χαρακτήρα. Αυτή η γεωμετρία καλείται Ragdoll Physics Component στο Unreal Engine. Οπότε στον πρωταγωνιστή συμβαίνουν ακριβώς αυτά. Πρώτα ο χαρακτήρας μεταβαίνει σε κατάσταση dying, με τιμή στη σημαία κατάστασης **bIsDying** αληθή, έπειτα σταματούν να παίζουν όλα τα αρχεία κινήσεων του χαρακτήρα και επιπλέον όλα τα αρχεία κινηματογραφημένης κίνησης, τα λεγόμενα Animation Montage. Αν πρόκειται για τον πρωταγωνιστή, μεταβαίνει σε τρίτο πρόσωπο και γίνεται αλλαγή στην γεωμετρία του χαρακτήρα έτσι ώστε να φαίνεται ολόκληρο το σώμα του. Διαγράφεται από την μνήμη ο πίνακας με τα πράγματα του χαρακτήρα και αποκολλείται ο χειριστής κίνησης του χαρακτήρα με την μέθοδο **DetachFromControllerPendingDelete** του χειριστή. Σταματούν να παίζουν οι ήχοι του χαρακτήρα και την θέση τους παίρνει ένας χαρακτηριστικός ήχος που δηλώνει τον θάνατο του. Σε κάποια βιντεοπαιχνίδια είναι αναφωνητά, σε άλλα κάτι πιο γενικό. Εδώ είναι ένας γδούπος.

Ο χαρακτήρας παίζει ένα αρχείο κίνησης θανάτου και η βαρύτητα του μεταβαίνει σε κατάσταση Ragdoll ενώ ταυτόχρονα γίνεται αλλαγή της γεωμετρίας των ορίων του, απενεργοποιείται η λειτουργικότητα της κάψουλας του ενώ ενεργοποιείται η γεωμετρία που είναι συνδεδεμένη με τον σκελετό του χαρακτήρα. Τέλος τίθεται ένας χρονομετρητής, με την βοήθεια της συνάρτησης **SetTimer** του διαχειριστή χρόνου της μηχανής **WorldTimerManager**, που ελέγχει αν η κίνηση θανάτου σχεδόν ολοκληρώθηκε. Μερικά μικροδευτερόλεπτα πριν ολοκληρωθεί η κίνηση ο χαρακτήρας παγώνει και μεταβαίνει ολόκληρος σε κατάσταση Ragdoll με την βοηθητική μέθοδο **SetRagdollPhysics**. Σε αυτήν,

ελέγχεται αν ο χαρακτήρας βρίσκεται σε κατάλληλη κατάσταση για να επενεργήσουν δυνάμεις φυσικής πάνω του και αν έχει έγκυρη γεωμετρία μοντέλου και βαρύτητας. Έπειτα εμποδίζεται η λειτουργία **Tick**, ενεργοποιείται η προσομοίωση φυσικής σε όλα τα γεωμετρικά σχήματα που καλύπτουν τον σκελετό του χαρακτήρα και σταματάει η κίνηση του χαρακτήρα αν δεν έχει γίνει ήδη. Αν εκτελεί ακόμα κινήσεις καθώς βρίσκεται σε κατάσταση βαρυτικής κίνησης, τότε τα κόκκαλα θα δώσουν επιπλέον ορμή στις ήδη υπάρχουσες δυνάμεις και θα προκαλέσουν την αδιάκοπη εκτόξευση του χαρακτήρα από σημείο σε σημείο. Αφού σταματήσει να πέφτει στο έδαφος, μετά από ένα μικρό χρονικό διάστημα απενεργοποιείται και διαγράφεται από την μνήμη για λόγους οικονομίας χώρου. Στην περίπτωση του πρωταγωνιστή, μετά το πέρας του χρονικού διαστήματος αντί να διαγραφεί ο χαρακτήρας το παιχνίδι επιστρέφει στο αρχικό μενού και επαναφέρει οποιαδήποτε αλλαγή έγινε στο κόσμο του βιντεοπαιχνιδιού.

3.10.6 Παίκτης – Οπλισμός

Ο οπλισμός του χαρακτήρα είναι ένας μονοδιάστατος πίνακας κλάσεων που κρατούν αναφορά για τη βάση των αντικειμένων που περιέχονται στον πίνακα. Κατά την κατάσταση φόρτωσης του χαρακτήρα στη μέθοδο **PostInitializeComponent** καλείται η μέθοδος αρχικοποίησης των πραγμάτων, **SpawnDefaultInventory** στην οποία γίνεται γραμμική αναζήτηση σε έναν πίνακα με προκαθορισμένες κλάσεις όπλων επαναληπτικά. Αν το κελί προς εξέταση περιέχει μια κλάση, δημιουργείται ένα νέο αντικείμενο με την προκαθορισμένη συνάρτηση μηχανής **SpawnActor** με τύπο κλάσης **AWeaponMaster** και πληροφορίες κλάσης από το κελί. Το νέο αντικείμενο όπλου στην συνέχεια προστίθεται στα πράγματα του παίκτη στο πρώτο διαθέσιμο κελί και καλείται στο όπλο το γεγονός πρώτης εισαγωγής του στα πράγματα, το **OnEnterInventory** που δίνει τιμές στο όπλο και διαχειρίζεται τις καταστάσεις του. Έτσι προσθέτοντας ή αφαιρώντας κλάσεις από διαφορετικά όπλα, ο πρωταγωνιστής ή ο αντίπαλος μπορεί να έχει στην κατοχή του ένα συγκεκριμένο οπλοστάσιο, χωρίς να χρειάζεται τροποποίηση στον κώδικα. Ο πίνακας προκαθορισμένων κλάσεων είναι ορατός και επεξεργάσιμος στα Blueprint χαρακτήρων που τον περιέχουν. Τέλος στη συνάρτηση **SpawnDefaultInventory** αν τελικά ο χαρακτήρας έχει στην κατοχή του έστω ένα όπλο, επιλέγεται το πρώτο και ενώνεται στον χαρακτήρα με την συνάρτηση Equip.

Όταν επιλέγει ένα νέο όπλο ο χαρακτήρας με την συνάρτηση **Equip**, ελέγχει αν προηγουμένως κρατούσε κάποιο έτσι ώστε να καλέσει στο όπλο το γεγονός αποεπιλογής, το **OnUnEquip**. Στη συνέχεια καλεί στο νέο όπλο το γεγονός **OnEquip** για να το κρατήσει. Έτσι ποτέ δεν υπάρχει περίπτωση να κρατήσει δύο όπλα ταυτόχρονα ή να ξεμείνουν ιδιότητες από το προηγούμενο όπλο.

3.10.7 Παίκτης – Εναλλαγή πρώτου/τρίτου προσώπου

Το ποια γεωμετρία θα εμφανιστεί το καθορίζει η βοηθητική συνάρτηση **IsFirstPerson** που ελέγχει αν ο χαρακτήρας είναι ζωντανός και ελέγχεται από έναν χειριστή παίκτη. Οπότε το Boolean αποτέλεσμα της πράξης χρησιμοποιείται σαν όρισμα για τη μέθοδο **SetOwnerNoSee** για τα δύο σώματα του παίκτη. Αυτή η μέθοδος αποτρέπει τον ίδιο τον χαρακτήρα από το να βλέπει κάποια γεωμετρία επάνω του ενώ ακόμα είναι φορτωμένη και ενεργή στο χώρο του βιντεοπαιχνιδιού. Με αυτήν την δυνατότητα δεν χρειάζεται να σπαταληθεί χρόνος για να δημιουργηθεί ένα νέο αντικείμενο γεωμετρίας ή να υπολογιστούν οι συναρτήσεις γραφικών του, απλά αλλάζει η ορατότητα του αντικειμένου. Χρειάζεται βέβαια προσοχή ώστε να μην χρησιμοποιούνται υπερβολικά πολλά κρυμμένα αντικείμενα στον χαρακτήρα γιατί αυξάνουν τις απαιτήσεις σε μνήμη παρόλο που ο παίκτης στο τελικό αποτέλεσμα δεν βλέπει τίποτα. Για παράδειγμα, τα όπλα του χαρακτήρα είναι προσαρτημένα πάνω του κατά την δημιουργία του και κρύβονται. Αν ο χαρακτήρας διαθέτει 7 διαφορετικά όπλα με πολύπλοκη γεωμετρία

και εφέ, θα είναι όλα κρυμμένα αλλά φορτωμένα στην μνήμη. Ο παίκτης θα αντιλαμβάνεται ότι κρατάει ένα όπλο την φορά και το αλλάζει με κάποιο άλλο σύμφωνα με τις ανάγκες του ενώ δεν μπορεί να κατανοήσει γιατί το παιχνίδι έχει μειωμένη απόδοση από την στιγμή που ο ίδιος πήρε και το 7ο όπλο στο σακίδιο του.

3.10.8 Παίκτης – Σκόπευση (Iron Sights)

Όταν ο παίκτης θέλει να σκοπεύσει με περισσότερη ακρίβεια καλεί το ζεύγος γεγονότων **OnIronSightsStart** και **OnIronSightsStop**. Στο γεγονός έναρξης της σκόπευσης καλείται η μέθοδος **StartIronSights** στο όπλο που χρησιμοποιεί ο χαρακτήρας και μειώνει τη διασπορά των πυρών, δίνοντας την εντύπωση ότι η σκόπευση είναι μεγαλύτερης ακρίβειας. Αυτό γίνεται με τη βοήθεια μιας μεταβλητής που παίζει τον ρόλο του τροποποιητή (modifier) και μειώνει ή αυξάνει το τελικό σκορ στην διασπορά. Επιπλέον παίζει μια κίνηση που φέρνει τη διόπτρα του όπλου πιο κοντά στο υποτιθέμενο οπτικό πεδίο του χαρακτήρα. Όμως όταν ο παίκτης σκοπεύει δεν αρκεί μόνο η κίνηση του χαρακτήρα για να δώσει την αίσθηση ότι συγκεντρώνεται στον στόχο. Η αλλαγή στην εστίαση του φακού της κάμερας συμπληρώνει την εμπειρία και επιπλέον βοηθάει έτσι ώστε ο παίκτης να βλέπει καλύτερα μέσα από την διόπτρα αντί για το ψηφιακό στόχαστρο της διεπαφής (HUD reticle). Επειδή είναι προτιμότερο η αλλαγή στην εστίαση να γίνεται σταδιακά, από οπτικής και αισθητικής άποψης, το κομμάτι κώδικα για αυτή τη λειτουργία έπρεπε να γραφεί στην μέθοδο **Tick** του χαρακτήρα έτσι ώστε να γίνεται αλλαγή στις παραμέτρους της κάμερας σταθερά αυξανόμενα σε κάθε πλαίσιο. Οπότε στην μέθοδο **Tick** προστέθηκε ένας έλεγχος για το αν χρησιμοποιείται η διόπτρα και έπειτα με την μαθηματική συνάρτηση **FInterpTo** (Float Interpolate To) με ορίσματα την αρχική - τελική τιμή εστίασης και το βήμα τίθεται σε κάθε πλαίσιο νέα τιμή στην μεταβλητή που αποθηκεύει την εστίαση της κάμερας και τελικά αυτή εισάγεται στην παράμετρο εστίασης. Η συνάρτηση **FInterpTo** δημιουργεί μια καμπύλη με απότομη μεταβολή στην αρχή και ομαλό τερματισμό ενώ λειτουργεί με float τιμές. Επειδή σε κάθε δράση του χαρακτήρα χρειάζεται αλλαγή στην εστίαση, υπάρχει ένα μπλοκ κώδικα με έναν έλεγχο για όλες τις καταστάσεις του χαρακτήρα, οπότε αν δεν τρέχει, δεν σκοπεύει και δεν έχει κανονική ταχύτητα, η ταχύτητα μαζί με την εστίαση επαναφέρονται σε μια σταθερή προκαθορισμένη τιμή.

3.10.9 Παίκτης – Θάνατος και αναγέννηση (OnDeath)

Αν ο παίκτης δεν έχει άλλους πόντους υγείας και μεταβεί σε κατάσταση θανάτου, περνάει λίγος χρόνος έως ότου κληθεί ο αποδομητής της κλάσης **Playable**, δηλαδή η συνάρτηση **OnDestroyed**. Στο ενδιάμεσο αυτό διάστημα η κλάση Playable ενημερώνει το **ThesisGameGameMode** ότι πέθανε ο κεντρικός παίκτης με **broadcast** της εντολής (**Delegate**) **OnPlayerDied** και όρισμα τον εαυτό του (**this**). Πριν προλάβει να κληθεί ο αποδομητής το παιχνίδι εκκινεί νέα συνεδρία και δημιουργεί έναν νέο παίκτη στην τελευταία γνωστή θέση πριν το θάνατο. Αν έκανε κάποια αποθήκευση της προόδου του, τότε θα μεταφερθεί εκεί, αλλιώς στο σημείο αφετηρίας. Εάν κληθεί ο αποδομητής και δεν κάνει κάτι το παιχνίδι, ο παίκτης εξαφανίζεται από τον χώρο και η ζωή του αδειάζει, καθώς επίσης και τα πυρομαχικά του δείχνουν 0/0. Επειδή αυτό δεν είναι και η καλύτερη εμπειρία για τον χρήστη, η μηχανή φροντίζει να φορτώσει έναν νέο παίκτη καθώς καθαρίζει από την μνήμη τον παλιό.

3.10.10 Οπλισμός – Weapon Class

Τα όπλα ανήκουν στην κλάση **AWeaponMaster** και αντιμετωπίζονται σαν χαρακτήρες από την μηχανή γιατί έχουν δικό τους σκελετό και κινήσεις. Παρόλα αυτά δεν τους δίνεται κάποιο σετ συμπεριφοράς όπως στους χαρακτήρες, αντίθετα χρησιμοποιούν τη συμπεριφορά του χαρακτήρα που κρατάει το όπλο. Έτσι είναι διαχειρίσιμα από τον εκάστοτε χαρακτήρα που τα κρατάει ενώ ταυτόχρονα διατηρούν τα

μοναδικά χαρακτηριστικά τους. Η βασική κλάση όπλων περιλαμβάνει έναν απαριθμητή (enumerator) που συμβολίζει τις καταστάσεις του όπλου με κείμενο, τον **EWeaponStates**. Επιπλέον μια δομή δεδομένων για τις ιδιότητες-χαρακτηριστικά του όπλου, την **FWeaponProperties** και μια δομή δεδομένων για τις πληροφορίες βολής, **FHitProperties**. Οι καταστάσεις στις οποίες μπορεί να βρεθεί το όπλο είναι η αδρανής (**Idle**), η ενεργή (**Firing**), του ανεφοδιασμού (**Reloading**) και της χρήσης (**Equipping**). Το όπλο σαν χαρακτηριστικά έχει μέγεθος γεμιστήρα, μέγιστο αριθμό από γεμιστήρες που μπορεί να κρατάει ο χαρακτήρας, αρχικό αριθμό γεμιστήρων, χρόνο μεταξύ συνεχόμενων βολών, προεπιλεγμένη διάρκεια κινήσεων αν δεν βρεθεί κάποιο αρχείο κίνησης, εύρος διασποράς και βαθμό ευστοχίας, ποσοστό ζημιάς που προκαλεί και αν έχει δυνατότητα να σκοπεύσει από το κλείστρο ο χαρακτήρας με αυτό το όπλο. Όταν ένα όπλο δημιουργείται, πρώτα δημιουργούνται δύο στοιχεία γεωμετρίας, ένα για το πρώτο πρόσωπο και ένα για το τρίτο και ρυθμίζονται οι παράμετροι ορίων. Το μόνο ενεργό στρώμα σύγκρουσης (Collision Layer) παραμένει το Visibility γιατί αποτελεί συμπαγές αντικείμενο αλλά αν χτυπηθεί από βολή αγνοείται, δεν προκαλεί ζημιά στον χαρακτήρα. Έπειτα προσαρτιέται στον χαρακτήρα στο κόμβο ένωσης που διαθέτει ο χαρακτήρας στον σκελετό του. Αυτός ο κόμβος λέγεται socket στην ορολογία του Unreal Engine και περιγράφει ένα θεωρητικό οστό με καθορισμένο διάνυσμα θέσης και γονέα κάποιο άλλο πραγματικό οστό του σκελετού του χαρακτήρα. Φέρει όνομα σε μορφή TEXT έτσι ώστε να μπορεί να γίνει αναφορά σε αυτό σε κώδικα και Blueprints. Οι μεταβλητές κατάστασης γίνονται ψευδείς και τίθεται σαν κατάσταση η αδρανής.

Αφού δημιουργηθεί το αντικείμενο, στην μέθοδο **PostInitializeComponent** ελέγχεται αν ο χαρακτήρας έχει γεμιστήρες ή το όπλο περιλαμβάνει από προεπιλογή και προσθέτει τα πυρομαχικά στο όπλο ενώ ενημερώνεται ο αριθμός διαθέσιμων γεμιστήρων. Επίσης το όπλο αποεπιλέγεται από τον χαρακτήρα, αποκολλείται από τον κόμβο ένωσης με την συνάρτηση **DetachMeshFromPawn**. Τέλος κρύβεται η γεωμετρία. Αν ο χαρακτήρας επιλέξει αυτό το όπλο, θα πυροδοτηθεί το γεγονός OnEquip το οποίο με την σειρά του θα εμφανίσει το όπλο σωστά στα χέρια του χαρακτήρα με την αντίστοιχη μέθοδο AttachMeshToPawn. Η μέθοδος **DetachMeshFromPawn** ελέγχει αν είναι έγκυρη η γεωμετρία πρώτου και τρίτου προσώπου και έπειτα χρησιμοποιεί την συνάρτηση μηχανής **DetachFromComponent** για να αποκολλήσει το όπλο από οποιοδήποτε parent έχει αλλά διατηρεί κίνηση σύμφωνα με αυτό με το όρισμα **KeepRelativeTransform**. Η μέθοδος **AttachMeshToPawn** είναι λίγο πιο σύνθετη, πρώτα αφαιρεί την ένωση του όπλου με τον χαρακτήρα και ύστερα αν ο χαρακτήρας είναι ο παίκτης ενώνει το όπλο και στους δύο σκελετούς του χαρακτήρα, δηλαδή του πρώτου και τρίτου προσώπου. Αν όμως είναι χαρακτήρας του υπολογιστή τότε κάνει ένωση μόνο στον σκελετό τρίτου προσώπου αφού δεν χρειάζονται άλλη οπτική οι υπόλοιποι χαρακτήρες. Και στις δύο περιπτώσεις γίνεται χρήση της μεθόδου μηχανής **AttachToComponent** με όρισμα τον αντίστοιχο σκελετό με σχετική μετατόπιση (Relative Transform) και σημείο ένωσης το κόμβο ένωσης που έχουμε ορίσει, το socket στο χέρι του χαρακτήρα.

Όταν πυροδοτείται το γεγονός **OnEquip** γίνεται ένωση του καινούργιου όπλου που πυροδότησε το γεγονός ενώ δίνεται σαν παράμετρος το παλιό όπλο σε περίπτωση που κρατούσε ο χαρακτήρας ήδη ένα. Αν δοθεί το παλιό όπλο σαν όρισμα τότε η συνάρτηση ζητάει από τον χαρακτήρα στον οποίο ανήκει το όπλο (Owner) να παίξει μια κίνηση για να αλλάξει όπλο με την μέθοδο **PlayAnimMontage**, χαρακτηριστική μέθοδος των αντικειμένων της κλάσης Pawn, με την οποία επιστρέφει την διάρκεια της κίνησης. Ο κώδικας μπαίνει σε κατάσταση αναμονής για χρήση (pending equip). Αν υπάρχει διάρκεια τότε ενεργοποιείται ένας χρονομετρητής από το πρώτο πλαίσιο που θα παίξει η κίνηση μέχρι το πέρας της, όπου θα πυροδοτηθεί η μέθοδος λήξης του χρονομετρητή, υπεύθυνη για να θέσει το όπλο σε νέα κατάσταση και να ζητήσει από τον ιδιοκτήτη του να το γεμίσει αν δεν έχει σφαίρες. Αν δεν υπάρχει διάρκεια στη κίνηση ή καθόλου κίνηση τότε απλά πυροδοτείται η μέθοδος λήξης. Και στις δύο περιπτώσεις ο κώδικας βγαίνει από την κατάσταση αναμονής και θεωρεί ότι το όπλο βρίσκεται στα

χέρια του χαρακτήρα και μπορεί να το χρησιμοποιήσει κανονικά. Όταν αλλάζει κατάσταση το όπλο (όχι ο κώδικας) περνάει από κάποιους ελέγχους, αν ο κώδικας βρίσκεται σε κατάσταση αναμονής για χρήση με την σημαία Pending Equip τότε το όπλο μπαίνει σε κατάσταση αναμονής, αν βρίσκεται σε κατάσταση αναμονής για γέμισμα με την σημαία Pending Reload τότε και το όπλο μπαίνει σε κατάσταση γεμίσματος. Αυτή η αλλαγή κατάστασης στον κώδικα και στο όπλο είναι απαραίτητη έτσι ώστε καθώς ο χαρακτήρας εκτελεί μια λειτουργία να μην μπορεί να μεταβεί σε άλλη και να διακόψει αυτό που κάνει. Επίσης αποτρέπει το κλέψιμο σε κινήσεις (animation cheating) όπου για παράδειγμα ένας παίκτης γεμίζει το όπλο με μισή κίνηση και στην υπόλοιπη μισή πυροβολεί. Αυτό δίνει άδικο προβάδισμα στον παίκτη ο οποίος εκμεταλλεύεται μια λειτουργία που σε καμία περίπτωση δεν ήταν κατασκευασμένη για να λειτουργεί κατ' αυτόν τον τρόπο. Κάθε φορά που εκτελείται μια λειτουργία στο όπλο ελέγχεται η κατάσταση του και είτε επιτρέπει την περεταίρω εκτέλεση είτε απορρίπτεται.

Με το γεγονός **OnUnEquip** γίνεται το ανάποδο από το **Equip**, το όπλο αποκολλείται από τον χαρακτήρα, σταματάει τις ενέργειες του και ότι κίνηση ζήτησε από τον ιδιοκτήτη του και καθαρίζει όλους τους χρονομετρητές. Κατόπιν κάνει έλεγχο κατάστασης και τίθεται σε αδράνεια.

Με το ζεύγος γεγονότων **OnEnter-OnLeaveInventory** το όπλο δίνει ιδιοκτησία στον χαρακτήρα που κάλεσε το γεγονός ή την αφαιρεί.

Όταν ο χαρακτήρας ζητάει από το όπλο να πυροβολήσει, πυροδοτείται το γεγονός **StartFire** του όπλου το οποίο αλλάζει κατάσταση μόνο αν δεν πυροβολεί ήδη. Κατόπιν ελέγχεται αν το όπλο μπορεί να πυροβολήσει, δηλαδή αν ο χαρακτήρας που το ζήτησε είναι ζωντανός, αν το όπλο βρίσκεται σε αδρανή κατάσταση ή πυροβολεί ήδη, έχει διαθέσιμες σφαίρες στον γεμιστήρα και επιπλέον δεν βρίσκεται σε κατάσταση αναμονής για γέμισμα. Αν ισχύουν οι συνθήκες αυτές το όπλο μεταβαίνει σε κατάσταση βολής (firing) και πυροδοτείται το γεγονός **OnBurstStarted** όπου ανάλογα αν απαιτείται τίθεται χρονομετρητής για να προσομοιώσει την καθυστέρηση μεταξύ βολών. Το κάθε όπλο έχει τις δικές του παραμέτρους, αν για παράδειγμα είναι αυτόματο έχει πολύ μικρή καθυστέρηση. Αν το όπλο έχει ήδη πυροβολήσει μια φορά αποθηκεύεται η χρονική στιγμή της τελευταίας βολής και δίνεται στον χρονομετρητή, διαφορετικά η βολή εκτελείται κανονικά. Και στις δύο περιπτώσεις, με ή χωρίς χρονομετρητή το όπλο εκτελεί την βολή με την συνάρτηση Handle Firing. Αυτή είναι μια συνάρτηση χειριστή κατά την οποία γίνεται έλεγχος, συγκεκριμένα την χρονική στιγμή που εκτελείται η βολή, για το αν το όπλο έχει σφαίρες έτσι ώστε να προσομοιώσει το εφέ των πυρών του όπλου και να υπολογίσει την βολή. Αν το όπλο δεν έχει σφαίρες και υπάρχουν διαθέσιμοι γεμιστήρες τότε καλεί την μέθοδο **StartReload** όπου ζητάει από τον χαρακτήρα να γεμίσει το όπλο. Αν δεν υπάρχουν ούτε γεμιστήρες τότε παίζει ένα ηχητικό από άδειο όπλο και ολοκληρώνει τον κύκλο βολής με την μέθοδο **OnBurstFinished**. Η συγκεκριμένη ακολουθία μεθόδων και αποφάσεων αποτελεί ένα σύστημα και μπορεί να αναπαρασταθεί με διάγραμμα ροής. Όταν το όπλο μεταβαίνει σε κατάσταση βολής και καλείται το ζεύγος μεθόδων **SimulateWeaponFire** και **FireWeapon**. Η πρώτη προσομοιώνει τα εφέ ενώ η δεύτερη προσομοιώνει την κρούση από την εύστοχη βολή.

Για το εφέ πυροβολισμού, ελέγχεται αν υπάρχει διαθέσιμη θήκη στο σκελετό του όπλου και κατόπιν δημιουργείται ένα αντικείμενο γεννήτριας εφέ (**Emitter**) με τη βοήθεια της μεθόδου μηχανής **SpawnEmitterAttached**, με είσοδο την θέση της κάννης του όπλου, το αρχείο εφέ και το σκελετό στον οποίο θα γίνει ένωση, εδώ το σκελετό του χαρακτήρα. Αν ο χαρακτήρας δεν ελέγχεται τοπικά αλλά απομακρυσμένα, γίνεται η ίδια διαδικασία αλλά το εφέ ενώνεται με το σκελετό του όπλου αντί για τον χαρακτήρα. Ταυτόχρονα το όπλο ζητάει από τον χαρακτήρα να παίξει μια κίνηση πυροβολισμού και ενεργοποιεί την σημαία κατάστασης κίνησης πυροβολισμού. Επειδή γίνονται αλλεπάλληλες κλήσεις σε αυτήν τη μέθοδο, ειδικά από αυτόματα όπλα, η κίνηση πυροβολισμού εκτελείται μόνο όταν η σημαία

κατάστασης κίνησης είναι κλειστή έτσι ώστε να μην εμποδίζεται η κίνηση από τυχόν επικαλύψεις κλήσεων. Τέλος παίζει ένα αρχείο ήχου πυροβολισμού και ένα εφέ δόνησης στην κάμερα και στο χειριστήριο του παίκτη, αν βρεθεί χειριστήριο συνδεδεμένο με δυνατότητα δόνησης. Το αρχείο ήχου προσομοιώνει επίσης τον θόρυβο σαν μια μεταβλητή έτσι ώστε οι αντίπαλοι να αντιληφθούν ότι έπεσε κάπου στον χώρο πυροβολισμός και να εξετάσουν την περιοχή.

Όταν υπολογίζεται η κρούση της βολής στη μέθοδο **FireWeapon**, δημιουργείται μια τυχαία τιμή με την μαθηματική συνάρτηση **MathRand()** του Unreal και εισάγεται σε μια τυχαία ακολουθία αριθμών **FRandomStream**. Έπειτα υπολογίζεται η γωνία θέασης της κάμερας του παίκτη και επιστρέφεται η κατεύθυνση στην οποία βαίνει ως διάνυσμα. Η κατεύθυνση με τη σειρά της βοηθάει στον υπολογισμό της εναρκτήριας θέσης της ακτίνας που θα δημιουργηθεί στην πορεία για να εξετάσει το σημείο πρόσκρουσης. Η θέση αυτή είναι συνάρτηση της κατεύθυνσης της κάμερας και της θέσης στην οποία στέκεται ο παίκτης ενώ όταν πρόκειται για αντίπαλο, χρησιμοποιείται η θέση και κατεύθυνση της κάννης του όπλου του αντιπάλου. Η τελική κατεύθυνση της βολής υπολογίζεται σαν ένα τυχαίο σημείο σε ένα τρισδιάστατο κωνικό εμβαδό που βαίνει στην κατεύθυνση της κάμερας και έχει διάμετρο ίση με το μισό της διασποράς του όπλου. Η βολή εξαρτάται αρκετά από την διασπορά καθώς όσο πιο μικρή διασπορά, τόσο πιο εύστοχη θα είναι η βολή. Το πέρας της ακτίνας βολής θα είναι η εμβέλεια του όπλου και το σημείο κρούσης υπολογίζεται από την πρώτη εφαπτόμενη ενός αντικειμένου στην ακτίνα, υπολογίζεται αυτόματα από την μηχανή μέσω της συνάρτησης **LineTrace** με είσοδο την αρχή και το πέρας της ακτίνας βολής. Ένα εφέ ουράς (trail effect) δημιουργείται κατά μήκος της τελικής ακτίνας βολής και αν υπάρχει σημείο κρούσης, δημιουργείται εφέ κρούσης στο σημείο αυτό και εφάπτεται στην επιφάνεια του σημείου. Έπειτα υπολογίζεται η ζημιά που εφόσον αυτό που χτυπήθηκε από την ακτίνα βολής είναι αντικείμενο της κλάσης Actor ελέγχεται αυτόνομα (δεν αποτελεί διακοσμητικό) τότε καλείται η συνάρτηση μηχανής **TakeDamage** σε αυτόν τον χαρακτήρα. Αυτή η συνάρτηση είναι ένα κλασικό παράδειγμα του πως ένα αντικείμενο στον χώρο του Unreal Engine μπορεί να ζητήσει μια λειτουργία από ένα άλλο αντικείμενο χωρίς να έχει κάποια σχέση κληρονομικότητας με αυτό. Αφού ο χαρακτήρας λάβει το γεγονός **TakeDamage** με παραμέτρους τη δύναμη βολής του όπλου και το όπλο σαν θύτη, υπολογίζει πόση ζωή πρέπει να αφαιρέσει από τον εαυτό του σύμφωνα με τη ζημιά που του προκλήθηκε. Σε αυτήν την μέθοδο μπορεί να υλοποιηθεί οποιαδήποτε αντίσταση σε ζημιά, περιβαλλοντική ή από εχθρό, με αποτέλεσμα ο χαρακτήρας να δέχεται ζημιά κάποιου μεγέθους αλλά να αγνοεί ένα μέρος αυτής. Στο τέλος της μεθόδου **FireWeapon** υπολογίζεται ή νέα διασπορά του όπλου η οποία είναι συνάρτηση της τρέχουσας διασποράς συν το βαθμό εκπυρσοκρότησης του όπλου. Έτσι σε κάθε επόμενη επανάληψη βολής θα υπάρχει μεγαλύτερη αστοχία εκτός αν οι βολές γίνονται πιο αργά και σταθερά. Αυτό αποτελεί μια ακόμη μηχανική καθώς αναγκάζει τον παίκτη να επιλέγει το πως θα επιτεθεί με στρατηγική, αν έχει αυτόματο όπλο και εκτελέσει μόνο μια μακράς διαρκείας ριπή θα χάσει πυρομαχικά άδικα αλλά ίσως κάνει αρκετό θόρυβο για να προσελκύσει αντιπάλους σε εκείνο το σημείο.

Με την μέθοδο **StartReload** το όπλο μεταβαίνει σε κατάσταση αναμονής για γέμισμα και εκτελείται η εξής ακολουθία: Ελέγχεται πρώτα αν το όπλο μπορεί να γεμίσει, δηλαδή αν ο χαρακτήρας που το ζήτησε είναι ζωντανός, αν ο κουμπωμένος γεμιστήρας έχει λιγότερες σφαίρες από το μέγιστο, αν υπάρχουν άλλοι διαθέσιμοι γεμιστήρες και αν το όπλο είναι σε κατάσταση αδράνειας ή βολής. Η επιτυχία σε αυτές τις συνθήκες αλλάζει κατάσταση στο όπλο όπου ελέγχεται αν είναι σε κατάσταση βολής και σταματάει για να μεταβεί σε κατάσταση γεμίσματος. Στη συνέχεια το όπλο ζητάει από τον ιδιοκτήτη του να παίξει μια κίνηση γεμίσματος και αφού επιστραφεί η διάρκεια της κίνησης τίθεται χρονομετρητής για να ολοκληρωθεί ο κύκλος με την μέθοδο **StopReload**, αν δεν υπάρχει διάρκεια μεταβαίνει απευθείας στην μέθοδο αυτήν. Ταυτόχρονα τίθεται ένας ακόμη χρονομετρητής όπου θα εκτελεστεί λίγο πριν

ολοκληρωθεί η κίνηση γεμίσματος και στο πέρας του καλεί την μέθοδο **Reload** όπου γίνεται ο υπολογισμός για να γεμίσει το όπλο. Σε αυτήν την μέθοδο υπολογίζεται η διαφορά μεταξύ του διαθέσιμου χώρου στον γεμιστήρα και του υπόλοιπου των σφαιρών από τους εναπομείναντες γεμιστήρες. Έτσι το όπλο θα γεμίσει όσο χρειάζεται και ποτέ περισσότερο από το μέγιστο χώρο που υπάρχει στον γεμιστήρα. Επίσης θα γεμίσει με όσες σφαίρες έμειναν και όχι πλήρως ανεξάρτητα από το τι έμεινε στον δεύτερο γεμιστήρα. Τέλος σταματάει το γέμισμα και κατά συνέπεια ο κύκλος αυτός με την μέθοδο **StopReload**, όπου τίθεται νέα κατάσταση αδράνειας στο όπλο και αν ακόμα παίζει η κίνηση του χαρακτήρα διακόπτεται.

3.10.11 Εχθρός – Bot Class

Ο χαρακτήρας και το όπλο σαν κλάσεις είναι φτιαγμένα με γνώμονα την ανθρώπινη παρέμβαση για να εκτελεστούν σωστά και με λογική σειρά τα γεγονότα δράσεων του. Οπότε για να δημιουργηθεί ένας χαρακτήρας που χειρίζεται ο υπολογιστής θα έπρεπε να δομηθεί διαφορετικά ο κώδικας του και να αποτελεί άλλη οντότητα. Όμως χάρη στην κληρονομικότητα και στους πολλαπλούς ελέγχους στις μεθόδους, είναι εύκολο να δημιουργηθεί ένας χαρακτήρας Bot που έχει σαν βάση τον πρωταγωνιστή και να χρησιμοποιεί ακόμα και την ίδια φιλοσοφία με αυτόν. Μια διαφορά στον δομητή του είναι ότι δημιουργείται μια κλάση τεχνητής νοημοσύνης και τίθεται σαν η προκαθορισμένη κλάση χειριστή για τα παράγωγα του (παράμετρος AI Controller Class της κλάσης Pawn) αντί για την προκαθορισμένη κλάση Player Controller που χρησιμοποιεί ο πρωταγωνιστής. Επίσης παραμετροποιείται μόνο το τρίτο πρόσωπο για αυτόν τον χαρακτήρα. Αυτό γίνεται με υπερφόρτωση της boolean μεθόδου **IsFirstPerson** της κλάσης **PlayableCharacter** όπου βραχυκυκλώνει τους υπολογισμούς και επιστρέφει πάντοτε false. Μια σημαντική παράμετρος που δεν επηρεάζει τον παίκτη αλλά καθιστά αδύνατο να κινηθούν στο χώρο τα Bot είναι η Boolean **bUseControllerRotationYaw** της κλάσης Pawn. Δίνοντας τιμή true, ο χειριστής κίνησης, εδώ ο **AIController,** που έχουμε ορίσει επιτρέπει στον χαρακτήρα να περιστρέφεται έτσι ώστε να βαίνει ευθεία με σημείο αναφοράς τον εαυτό του αντί για τον χώρο. Σε συνδυασμό με την υπερφόρτωση της μεθόδου **FaceRotation** της κλάσης **Pawn** με την οποία ο χαρακτήρας παίρνει την κατεύθυνση του χειριστή του, το αποτέλεσμα είναι ο χαρακτήρας να μπορεί να περιστραφεί και να προχωρήσει εκεί που θέλει να πάει. Στον παίκτη, αυτή η λειτουργία γίνεται με την βοήθεια του ποντικιού όπου η μηχανή περιμένει από την είσοδο του ποντικιού να δώσει ταχύτητα και κατεύθυνση περιστροφής στον χειριστή του χαρακτήρα, τον **PlayerController**. Σε δοκιμές που έγιναν, αν παραλειφθεί η αλλαγή ταχύτητας περιστροφής, ο χαρακτήρας τεχνητής νοημοσύνης **Bot** χρειάζεται αρκετές επαναλήψεις της διαδρομής για να καταφέρει να στρίψει. Αν έχει καλή ταχύτητα περιστροφής αλλά δεν κατευθύνεται εκεί που δείχνει ο χειριστής του, θα πηγαίνει μπρος πίσω στο ίδιο σημείο αφού κοιτάει πάντα στην ίδια κατεύθυνση. Μπορεί να πηγαίνει δεξιά ή αριστερά με κίνηση προς θετικές ή αρνητικές συντεταγμένες, αλλά θα προχωράει πλάγια κοιτώντας πάντα ευθεία. Σε παιχνίδια βολών αυτό είναι επιθυμητό αποτέλεσμα όταν κάποιος θέλει να είναι προσηλωμένος στον στόχο του αλλά σε περιπολία πρέπει να μπορεί να περιστρέφεται στον χώρο ελεύθερα.

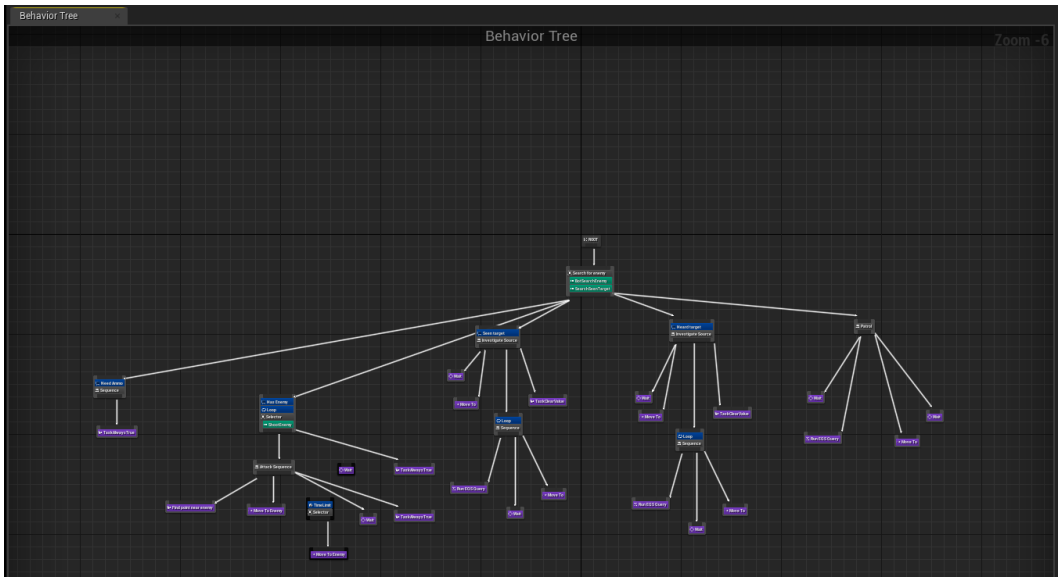
3.11 Τεχνητή νοημοσύνη χαρακτήρων

Όλα τα παράγωγα της κλάσης **Actor** διαθέτουν τη δυνατότητα να λάβουν εντολές χειρισμού από τον υπολογιστή μέσω ενός χειριστή τεχνητής νοημοσύνης, τον **AIController**. Ο χειριστής με την σειρά του χρησιμοποιεί ένα ή περισσότερα δέντρα αποφάσεων, γνωστά στην βιομηχανία παιχνιδιών ως **Behavior Trees**. Έτσι εκτελεί ενέργειες και λαμβάνει εισόδους από το περιβάλλον του με τη μορφή γεγονότων (Event Driven), τις οποίες δίνει πίσω στο δέντρο απόφασης για να αλλάξει συμπεριφορά ανάλογα με συνθήκες [56].

3.11.1 Τεχνητή νοημοσύνη – Δέντρο αποφάσεων (Behavior Tree)

Το δέντρο αποφάσεων με την εκτέλεση να αρχίζει από την ρίζα και να προχωράει προς τα κάτω, από αριστερά προς τα δεξιά, ελέγχει τις ενέργειες ενός χαρακτήρα [57]. Το κάθε κλαδί είναι μια διαδρομή εκτέλεσης ενώ το κάθε φύλλο αποτελεί μια ακολουθία ενεργειών, Tasks, που μπορεί να εκτελέσει ο χαρακτήρας όσο η ροή εκτέλεσης βρίσκεται στο συγκεκριμένο φύλλο. Οι ενέργειες μπορεί να είναι οποιεσδήποτε, από απλή κίνηση σε ένα σημείο μέχρι πολύπλοκη συμπεριφορά υπό συνθήκες. Σε κάθε φύλλο μπορεί να χρησιμοποιηθούν μπλοκ ελέγχου που ονομάζονται Decorator και είναι υπεύθυνα για την απόφαση του αν θα εκτελεστεί η ενέργεια στο φύλλο. Η σύγκριση που γίνεται είναι απλή, ελέγχεται πάντα αν ένα κλειδί μαυροπίνακα έχει έγκυρη τιμή. Ο μαυροπίνακας είναι ένα Blueprint που αποθηκεύει και τροποποιεί μεταβλητές που αφορούν τις αποφάσεις του χαρακτήρα. Για να λειτουργήσει σωστά το δέντρο συμπεριφοράς σε αποφάσεις και συνθήκες, ο μαυροπίνακας είναι απαραίτητος.

Οι περισσότερες δοκιμές πάνω στην συμπεριφορά των χαρακτήρων έγιναν στο δοκιμαστικό χάρτη όπου ήταν εύκολο να αλλάξει η διαρρύθμιση του χώρου για να δοκιμαστούν περισσότερα από ένα σενάρια. Στην αρχή τοποθετήθηκαν δυο παραλληλόγραμμα που χώριζαν το χάρτη σε δύο διαδρόμους. Ο χαρακτήρας αντίπαλος έπρεπε να κινείται και επιπλέον να μπορεί να ξεχωρίσει τις στροφές και να μην προσπαθεί να περάσει από τους τοίχους. Αυτό έγινε με την τοποθέτηση ενός χάρτη πλοήγησης και έπειτα με την προσθήκη ενός σετ ενεργειών στο δέντρο συμπεριφοράς του εχθρού. Το σετ περιλαμβάνει επιλογή ενός σημείου τυχαίου σε εύρος 1000 μονάδες ή εκατοστά καθώς στο Unreal Engine τα εκατοστά για την αναπαράσταση των μεγεθών είναι σε κλίμακα 1:1 με τα πραγματικά μεγέθη. Μετά την επιλογή σημείου, ακολουθεί η συνάρτηση κίνησης (**AIMoveTo**), που δίνεται έτοιμη από τη μηχανή και παίρνει σαν ορίσματα τον τελικό προορισμό, τον χαρακτήρα και την ταχύτητα. Επιπλέον μπορεί να αλλάξει η τεχνική διαίρεσης του χάρτη πλοήγησης για καλύτερη κίνηση στο χώρο. Οι επιλογές είναι **Polygon Based** και **Recast Based**. Ο τελικός προορισμός περιγράφεται με ένα διάνυσμα τριών διαστάσεων το οποίο επιστρέφεται κατά την εύρεση τοποθεσίας και αποθηκεύεται σε μια μεταβλητή στον μαυροπίνακα. Στην πιο απλή μορφή μιας ενέργειας, όσο ενημερώνεται η μεταβλητή προορισμού, τόσο ο χαρακτήρας θα κινείται προς τον προορισμό, λαμβάνοντας πάντοτε υπ' όψη το χάρτη πλοήγησης και δίνεται η αίσθηση της απόφασης, ότι ο χαρακτήρας αποφασίζει να κινηθεί προς το σημείο Β όταν βρίσκεται στο Α.

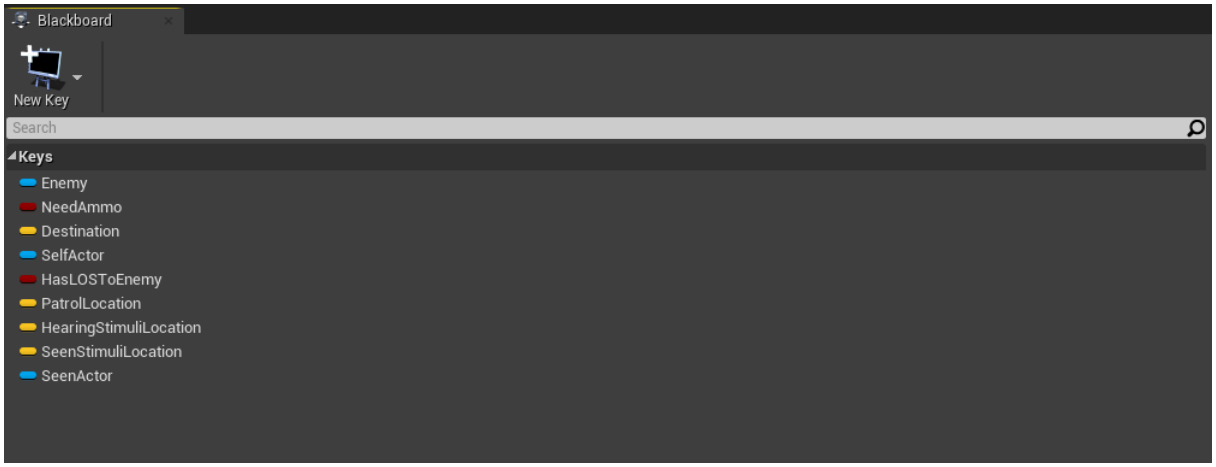


Εικόνα 3.17: Τελικό δένδρο αποφάσεων του εχθρού

3.11.2 Τεχνητή νοημοσύνη – Μαυροπίνακας (Blackboard)

Ο μόνος τρόπος να διαβάσει μεταβλητές το δέντρο αποφάσεων είναι: Να βρει έναν ενεργό μαυροπίνακα (**blackboard**) που είναι δηλωμένος στη μνήμη κατά το τρέξιμο της εφαρμογής και συνδεδεμένος με το δέντρο σαν Active Blackboard Component με την μέθοδο **SetBlackboardComponent**. Έπειτα, με την βοήθεια της μεθόδου **GetBlackboardKeyAs*** να βρει μια μεταβλητή και να την επιστρέψει σαν κάτι συγκεκριμένο, οι επιλογές είναι οποιοσδήποτε τύπος μεταβλητής. Έτσι κάνει casting εντός της μεθόδου και επιστρέφει αυτό που χρειαζόμαστε. Επειδή όμως το casting μπορεί να αποτύχει, χρειάζεται πάντα ένα μπλοκ ελέγχου πριν τη χρήση των μεταβλητών μαυροπίνακα για να δει η μηχανή αν είναι έγκυρες οι τιμές τους για πράξεις. Η απουσία ενός τέτοιου ελέγχου ενδέχεται να προκαλέσει ανεξήγητο σφάλμα στη μηχανή κατά το τρέξιμο της εφαρμογής όταν ο χαρακτήρας θα αποφασίσει να εκτελέσει το αφιλτράριστο κλαδί.

Για την εισαγωγή τιμών στις μεταβλητές μαυροπίνακα ακολουθείται παρόμοια διαδικασία. Χρησιμοποιείται η μέθοδος **SetBlackboardKeyAs*** του ενεργού μαυροπίνακα του δέντρου με επιλογή κάθε τύπο μεταβλητής ανάλογα με τις εκάστοτε ανάγκες του προγραμματιστή. Έτσι αποθηκεύεται η επιλογή τυχαίας τοποθεσίας στο δέντρο και χρησιμοποιείται σαν νέα είσοδος στο επόμενο μπλοκ ενεργειών του δέντρου για να μετακινηθεί ο χαρακτήρας προς το διάνυσμα.



Εικόνα 3.18: Μαυροπίνακας του εχθρού

3.11.3 Τεχνητή νοημοσύνη – Μπλοκ Ελέγχου (Decorators)

Τα μπλοκ ελέγχου **decorators** ενημερώνονται αυτόματα κατά την τροποποίηση των μεταβλητών του μαυροπίνακα, στο επόμενο ορισμένο tick της μηχανής, συνήθως σε κάθε πλαίσιο. Όταν η συνθήκη ενός decorator γίνει αληθής, τότε στέλνει ένα γεγονός στη μηχανή στον χειριστή γεγονότων για να εξετάσει την τρέχουσα κατάσταση, να ακυρώσει ή να τερματίσει την τρέχουσα λειτουργία και να μεταβεί στο νέο φύλλο με ενέργειες. Ενέργειες αναμονής (Wait Task) μπορούν να επιβραδύνουν την αλλαγή κατάστασης όταν χρειάζεται περισσότερος χρόνος να ολοκληρωθεί μια σειρά ενεργειών. Άλλες φορές είναι απαραίτητο να υπάρχουν τέτοιες ενέργειες έτσι ώστε να δίνεται λίγος χρόνος στον παίκτη να αντιδράσει. Οι ενέργειες μπορεί να είναι προκαθορισμένες ή καινούργιες, το ίδιο και τα μπλοκ ελέγχου. Όταν χρειάζεται διαρκής ενημέρωση μεταβλητών σύμφωνα με το περιβάλλον στο οποίο βρίσκεται ο χαρακτήρας, χρησιμοποιούνται υπηρεσίες (**Services**) που τοποθετούνται επίσης στο μπλοκ εκτέλεσης ενός φύλλου.

3.11.4 Τεχνητή νοημοσύνη – Υπηρεσίες (Services)

Οι υπηρεσίες είναι μια επαναληπτική διαδικασία που εκτελείται παράλληλα με τις αποφάσεις σε προκαθορισμένο χρόνο σύμφωνα με την παράμετρο **Interval** και μπορεί να κάνει ελέγχους ή πολύπλοκους υπολογισμούς τους οποίους έπειτα να καταχωρεί σε μεταβλητές του μαυροπίνακα. Σε υπηρεσία τρέχει συνήθως ένα ερώτημα περιβάλλοντος (EQS) καθώς είναι πολύπλοκη διαδικασία και απαιτεί κάποια μικροδευτερόλεπτα. Μπορεί να εκτελεστεί και σαν ενέργεια (**Task**), αλλά ο χαρακτήρας θα περιμένει καθώς θα εξετάζει το περιβάλλον για την καλύτερη επιλογή σημείου στο χώρο και θα δείχνει αρκετά παράξενος, σαν να πάγωσε η νοημοσύνη του.

3.11.5 Νοημοσύνη εχθρού – Enemy Behavior Tree

Η νοημοσύνη που κατασκευάστηκε για τους εχθρούς ήταν βασική. Οι εχθρικοί στρατιώτες μπορούν να κάνουν περιπολία, να εξετάσουν τον χώρο σε περίπτωση που ακούσουν ή δουν κάτι που μοιάζει με τον παίκτη και να επιτεθούν στον παίκτη όταν είναι σε απόσταση βολής ή τον έχουν αντιληφθεί και είναι σίγουροι ότι είναι ο παίκτης. Όλες οι ενέργειες βρίσκονται στο ίδιο δέντρο αποφάσεων και είναι χωρισμένες στην ρίζα, πρώτα σύμφωνα με το αν βρίσκονται σε κατάσταση μάχης και δεύτερον αν έχουν δεχτεί κάποιο εξωτερικό ερέθισμα. Σε περίπτωση που δεν ισχύει καμία συνθήκη, ο εχθρός εκτελεί περιπολία γύρω από το σημείο που τοποθετήθηκε, σε απόσταση 4000 μονάδων ή 4 μέτρων. Η περιπολία περιλαμβάνει έναν decorator που εξετάζει αν υπάρχει ένα έγκυρο σημείο προορισμού και αν υπάρχει, μετακινεί τον χαρακτήρα προς αυτό το σημείο με την ενέργεια **AIMoveTo**, περιμένει μερικά δευτερόλεπτα με την ενέργεια **Wait**, έπειτα διαβάζει ένα νέο σημείο με μια χειροποίητη ενέργεια, την και ολοκληρώνει τον κύκλο εκτέλεσης. Στον επόμενο κύκλο θα μεταφερθεί στο νέο σημείο κ.ο.κ.

Όταν δεχθεί κάποιο ερέθισμα και δεν βρίσκεται ήδη σε μάχη, ο εχθρός να προσπαθήσει να βρει την πηγή του. Για παράδειγμα, αν ακούσει πυροβολισμό ή βήματα, θα πάει κοντά στην περιοχή που άκουσε τον ήχο για να ερευνήσει την πηγή. Η πηγή θεωρείται άγνωστη μέχρι να έχει οπτική επαφή με το αντικείμενο, αλλιώς θα ήξερε εξ αρχής ότι ο παίκτης πυροβόλησε και θα ερχόταν επάνω του σε χρόνο ρεκόρ. Με την βοήθεια ενός decorator, βρίσκει ένα πιθανό σημείο στο εύρος του ερεθίσματος και πηγαίνει να εξετάσει το σημείο. Έπειτα βρίσκει ένα σημείο πίσω του έτσι ώστε να εξετάσει και πιο μακριά από το άκουσμα. Τέλος πηγαίνει σε ένα ακόμη τυχαίο σημείο πιο κοντά στο άκουσμα και μετά επιστρέφει σε κανονική περιπολία. Αν κατά το διάστημα που βρίσκεται σε περιπολία ή έλεγχο δει τον παίκτη, το δέντρο βραχυκυκλώνει όλα τα κλαδιά που δεν έχουν σχέση με την μεταβλητή Boolean **IsInCombat**, που γίνεται αληθής και ο εχθρός εκτελεί συμπεριφορά μάχης.

Η μεταβλητή **IsInCombat** είναι ένα Boolean κλειδί μαυροπίνακα που ενημερώνεται κάθε φορά που ένας εχθρός δέχεται ζημιά από τον παίκτη ή δέχεται ένα ερέθισμα και βρίσκεται σε απόσταση μικρότερη από 2 μέτρα από το ερέθισμα και πηγή αυτού είναι ο παίκτης. Σε αυτήν την κατάσταση ο εχθρός προσπαθεί να κινείται πλάγια και να μειώνει την απόσταση από τον παίκτη έτσι ώστε να είναι δύσκολος στόχος ο ίδιος αλλά και να μην αστοχούν οι βολές του. Με την βοήθεια μιας υπηρεσίας, προσπαθεί κάθε φορά να εντοπίσει την θέση του παίκτη σύμφωνα με το σύστημα ερεθισμάτων του (να ακούσει ή να δει) αλλά είναι υπερευαίσθητος και θεωρεί οποιοδήποτε ερέθισμα, με πηγή τον παίκτη, εχθρικό. Με μια ακόμη υπηρεσία πυροβολεί προς την κατεύθυνση του παίκτη αν και εφόσον τον έχει δει και είναι σε ανεκτή απόσταση βολής. Επειδή όμως η κατεύθυνση του εχθρού γίνεται σταθερή ως προς την κατεύθυνση του παίκτη, ο εχθρός στις αρχικές δοκιμές ήταν υπερβολικά εύστοχος και δεν άφηνε κανένα περιθώριο αποφυγής ή κάλυψης. Έτσι προστέθηκε περισσότερη τεχνητή διασπορά στις βολές του για να αστοχεί τυχαία και να αφήνει μεγαλύτερο περιθώριο αντίδρασης.

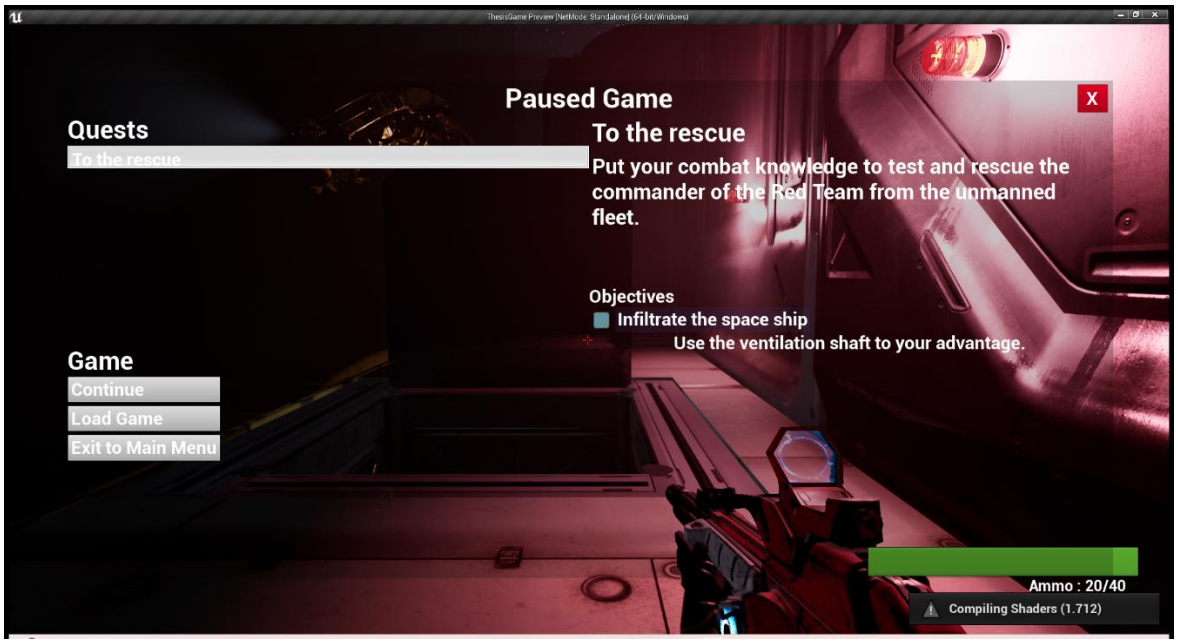
3.12 Αποστολή – Quest Class

Η αποστολές δημιουργούνται από το **ThesisGameGameMode** στη συνάρτηση **BeginPlay,** καθώς φορτώνεται ο κόσμος του παιχνιδιού και κληρονομούν τη συμπεριφορά ενός αντικειμένου (Actor) στον χώρο, αλλά στην πιο γενικευμένη του μορφή. Δηλαδή διαθέτουν μια εικόνα (marker) που είναι εμφανής στον editor και συντεταγμένες στο τρισδιάστατο χώρο. Χρησιμοποιούν ως βάση τη κλάση **AInfo** που είναι η πιο γενικευμένη κλάση των Actor. Αυτή είναι κατάλληλη για αντικείμενα που διατηρούν μόνο πληροφορίες και δείκτες ενώ δεν χρειάζονται τρισδιάστατη αναπαράσταση, υλικά και πληροφορίες σκίασης [58].

Η κλάση της αποστολής περιλαμβάνει όνομα, περιγραφή, τρέχουσα κατάσταση, δείκτη αφετηρίας και λήξης, σημαίες κατάστασης για το αν είναι ενεργή στο μενού και αν έχει ξεκινήσει, μια δομή δεδομένων για τα επιμέρους βήματα που θα προτρέπει τον παίκτη να ακολουθήσει και πίνακες για τη καταγραφή των βημάτων. Επιπλέον εμπεριέχει συναρτήσεις που βοηθούν την χρήση της σε blueprint καθώς είναι παιδί μιας κενής κλάσης χωρίς ιδιαίτερα προκατασκευασμένα στοιχεία.

Η αποστολή μπορεί να αρχίσει (**Start**), να προχωρήσει σε νέο στάδιο (**Update**) και να ολοκληρωθεί (**Complete**). Στον δομητή και στην ενημέρωση της αποστολής χρησιμοποιείται η εσωτερική συνάρτηση **DetermineQuestFlow** που, όπως ορίζει και το όνομα της, αποφασίζει για την ροή της οντότητας, δηλαδή αν θα ξεκινήσει ή θα ολοκληρωθεί. Όταν αναφερόμαστε στην αρχή της αποστολής, τη συνάρτηση **Start**, σημαίνει ότι θα μεταβεί στην κατάσταση εκκίνησης (started), θα παίξει έναν χαρακτηριστικό ήχο που σημαίνει ότι ο παίκτης πήρε μια νέα αποστολή και θα ζητήσει από το **ThesisGameHUD** να εμφανίσει το κατάλληλο μήνυμα στην οθόνη. Όταν ολοκληρώνεται, μια αποστολή μεταβαίνει στη κατάσταση ολοκλήρωσης (completed) και ο δείκτης δείχνει στον ακέραιο που σηματοδοτεί την λήξη. Όμοια με την εκκίνηση, ζητάει από το **ThesisGameHUD** να εμφανίσει ένα μήνυμα, δηλαδή καλεί την συνάρτηση **ShowMSGWidget**.

Όταν ενημερώνεται (**Update**), καλεί την συνάρτηση AddObjectiveToQueue του ThesisGameHUD για κάθε τρέχον βήμα (objective) που είναι ενεργό και ενωμένο με το τρέχον στάδιο της αποστολής. Δηλαδή, προσθέτει σε μια ουρά εμφάνισης κάθε βήμα από τα δύο σύνολα, των ολοκληρωμένων και των προσεχώς διαθέσιμων. Οπότε, ένα προς ένα τα βήματα εμφανίζονται ότι ολοκληρώθηκαν ή άρχισαν Οπότε, ένα προς ένα τα βήματα εμφανίζονται ότι ολοκληρώθηκαν ή άρχισαν, ανάλογα με την τρέχουσα κατάσταση τους.



Εικόνα 3.19: Η αποστολή όπως τη βλέπει ο παίκτης

3.12.1 Στάδια αποστολής

Τα στάδια της αποστολής είναι ακέραιοι αριθμοί που σηματοδοτούν την αφετηρία, τα ενδιάμεσα βήματα και τη λήξη. Η αφετηρία είναι συνήθως το 0 ή όποιο άλλο νούμερο θεωρεί κατάλληλο ο προγραμματιστής ενώ η λήξη ένα νούμερο μεγαλύτερο από την αφετηρία. Κάθε ενδιάμεσο στάδιο μπορεί να είναι ενωμένο με κάποιο βήμα έτσι ώστε όταν η αποστολή ενημερωθεί ότι προχώρησε στο νέο στάδιο να μπορεί να ολοκληρώσει τα προηγούμενα βήματα και να αρχικοποιήσει τα καινούργια, προσθέτοντας τα σε έναν πίνακα τιμών δομών δεδομένων. Όταν ενημερωθεί και μεταβεί σε στάδιο ίσο ή μεγαλύτερο με το τελικό, ολοκληρώνεται.

3.13 Εναύσματα (QuestTrigger)

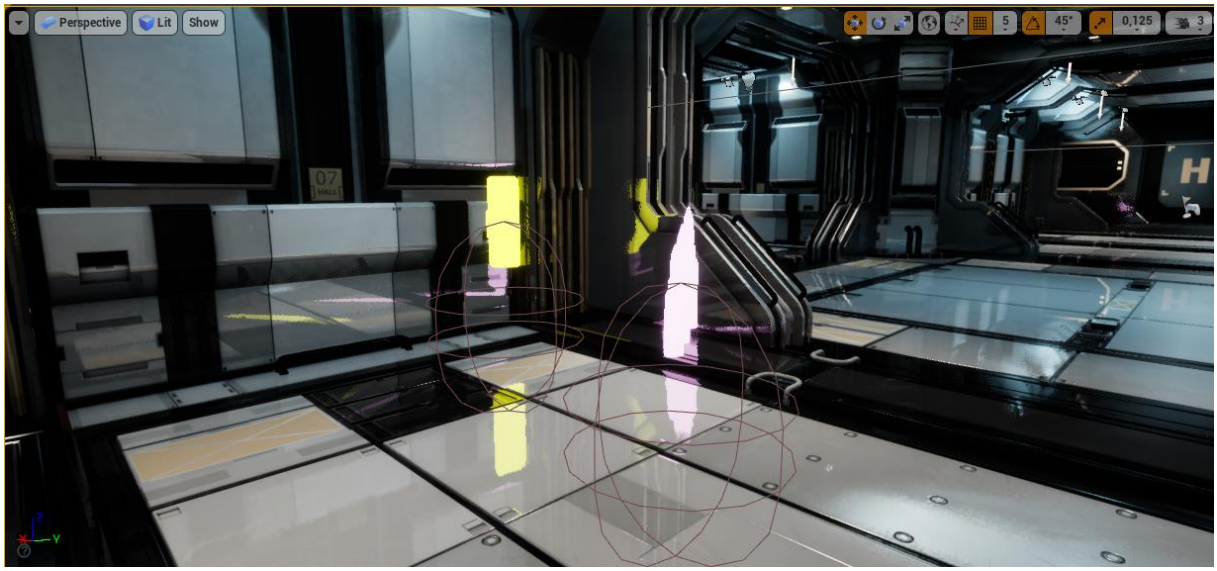
Τα εναύσματα αποστολής είναι αντικείμενα με κάποιο βασικό γεωμετρικό σχήμα (ορθογώνιο παραλληλόγραμμο, σφαίρα ή κάψουλα) χωρίς ορατή αναπαράσταση στον χώρο και υπερφορτώνουν τις συναρτήσεις εισόδου και εξόδου από το τρισδιάστατο περίβλημα που ορίζει το γεωμετρικό σχήμα. Οι βασικές ταυτότητες των συναρτήσεων είναι οι **BeginOverlap** και **EndOverlap** όπου δένονται δυναμικά κατά την παραμετροποίηση του αντικειμένου στον δομητή του, με τη βοήθεια της προκαθορισμένης **AddDynamic**. Η συνάρτηση **AddDynamic** δέχεται ως όρισμα την διεύθυνση μνήμης της συνάρτησης που πρόκειται να δέσει με την ταυτότητα μιας προκαθορισμένης συνάρτησης. Απαραίτητη προϋπόθεση για να μεταγλωττιστεί κώδικας που περιέχει δυναμική ανάθεση είναι οι νέες συναρτήσεις να δέχονται τα ίδια ορίσματα με την πηγαία ταυτότητα και να έχουν το ίδιο τύπο επιστροφής.

Όταν ο παίκτης περνάει μέσα από ένα έναυσμα, δεδομένου ότι αυτό έχει γεωμετρία πρόσκρουσης και έχει οριστεί η τιμή Overlap στον Editor της μηχανής ή στον δομητή της κλάσης, καλείται η **BeginOverlap** και αν υπάρχει έγκυρη τιμή για την αποστολή στόχο, την ενημερώνει σε νέο στάδιο σύμφωνα με το στάδιο στόχο που όρισε ο προγραμματιστής. Αν ενημερωθεί με επιτυχία, θέτει στον εαυτό του χρόνο ζωής ένα δευτερόλεπτο (**SetLifeSpan**) πριν κληθεί ο αποδομητής. Όταν εξέρχεται από τη γεωμετρία πρόσκρουσης ο παίκτης καλείται η αντίστοιχη συνάρτηση **EndOverlap** που έχει αφεθεί κενή για περαιτέρω ανάπτυξη αν απαιτούνταν στη πορεία του έργου.

Για να είναι έγκυρη η αναφορά στην αποστολή στόχο, καλείται η συνάρτηση **RegisterTargetQuest** κατά το γεγονός BeginPlay της μηχανής σε χρόνο 3 κλάσματα του δευτερολέπτου από την εισαγωγή στο γεγονός. Δεν καλείται αμέσως διότι πρέπει να δοθεί λίγος χρόνος στη μηχανή έτσι ώστε να δημιουργήσει τις αποστολές μέσα στο τρέχον χώρο. 3 κλάσματα είναι υπεραρκετά σύμφωνα με όσες δοκιμές έγιναν, περισσότερο από αυτό το χρόνο είναι μεν καλύτερο για να δημιουργηθούν σίγουρα οι σωστές αναφορές αλλά είναι και μεγάλο το χρονικό διάστημα κατά το οποίο ο παίκτης μπορεί να πρέπει να ενεργοποιήσει το έναυσμα και να αστοχήσει ο κώδικας. Εδώ, επειδή είναι σημαντικό να λειτουργούν πάντα σωστά τα εναύσματα καθώς η ροή του παιχνιδιού εξαρτάται από αυτά, καλείται η ίδια συνάρτηση κατά το γεγονός **BeginOverlap** έτσι ώστε ακόμα και αν απέτυχε να αρχικοποιηθεί η αναφορά κατά τη φόρτωση του κόσμου, να γίνει επί τόπου. Το μόνο μείον αυτής της τεχνικής είναι μια καθυστέρηση (μείωση των πλαισίων προβολής, framerate, lag) αν υπάρχουν πολλές αποστολές που τρέχουν ταυτόχρονα και πρέπει να γίνει η αναζήτηση αναφορών.

Η συνάρτηση RegisterTargetQuest ζητάει από τη κατάσταση παιχνιδιού, το ThesisGameState να κάνει αναζήτηση στις τρέχουσες αποστολές για κάποιο αντικείμενο με κλάση ίδια με αυτή που όρισε ο προγραμματιστής στο έναυσμα. Για την σύγκριση κλάσεων χρησιμοποιείται η συνάρτηση GetClass που ισχύει για τα UObjectBase, δηλαδή όλα τα αντικείμενα Unreal.

3.14 Βοηθητικά αντικείμενα (Pickup)



Εικόνα 3.20: Τα Pickup, Ammo και Medpack

Τα pickups είναι αντικείμενα που ο παίκτης μπορεί να πάρει για να τον βοηθήσουν στην πρόοδο του μέσα στο παιχνίδι και αναφέρονται σε πυρομαχικά και κουτιά πρώτων βοηθειών. Έχουν ένα γεωμετρικό περίβλημα που καθορίζει σε ποιο σημείο πρέπει να έρθει σε επαφή με αυτά ο παίκτης έτσι ώστε να ενεργοποιηθεί μια σειρά από διαδικασίες για να μπορεί να πάρει το αντικείμενο. Έχουν ένα προαιρετικό εφέ και έναν ήχο που θα ακουστεί όταν ο παίκτης πάρει με επιτυχία το αντικείμενο. Η βασική κλάση Pickup ορίζει τη γεωμετρία του αντικείμενου, τον ήχο και το εφέ, καθώς επίσης και ένα εξάρτημα κώδικα (component) που ενεργοποιεί το διάβασμα των εισόδων του παίκτη, το PlayerInputComponent. Αφού αυτό αρχικοποιηθεί, ζητάει να δεθεί μια ενέργεια του παίκτη με κάποια συνάρτηση της κλάσης. Εδώ, η ενέργεια **Interact** που αντιστοιχεί στο κουμπί E του πληκτρολογίου δένεται με το γεγονός **OnPickup** με την βοήθεια της **BindInput**. Στα γεγονότα Begin και End Overlap ενεργοποιείται και απενεργοποιείται η πρόσβαση στο χειρισμό του παίκτη έτσι ώστε να μην μπορεί να πάρει ένα

αντικείμενο χωρίς πρώτα να είναι αρκετά κοντά. Αυτό επιτυγχάνεται με την συνάρτηση της βασικής κλάσης Actor, **EnableInput** και **DisableInput** αντίστοιχα. Όταν ενεργοποιηθεί το γεγονός **OnPickup**, καλεί την συνάρτηση **TransferPickup** που είναι υπεύθυνη για τη μεταφορά του αντικειμένου από το κόσμο στα πράγματα του παίκτη. Είναι εικονική συνάρτηση που υλοποιείται από όλες τις υποκλάσεις του Pickup και επιστρέφει λογική τιμή true ή false. Αν επιστρέψει true η συνάρτηση, το γεγονός προχωράει και εκτελεί την καταστροφή του αντικειμένου, αλλιώς επιστρέφει στη μηχανή. Σημασία έχει να μην έχει τεθεί η σημαία **AutoReceiveInput** στην δημιουργία του Pickup γιατί θα λειτουργήσει ανάποδα η είσοδος του παίκτη με αποτέλεσμα να μπορεί να πάρει το αντικείμενο μόνο όταν βρίσκεται εκτός εμβέλειας από αυτό. Πράγμα που δεν μπορεί να γίνει γιατί θα λείπει ο δείκτης μνήμης του παίκτη, οπότε απλά δεν λειτουργεί η οντότητα.

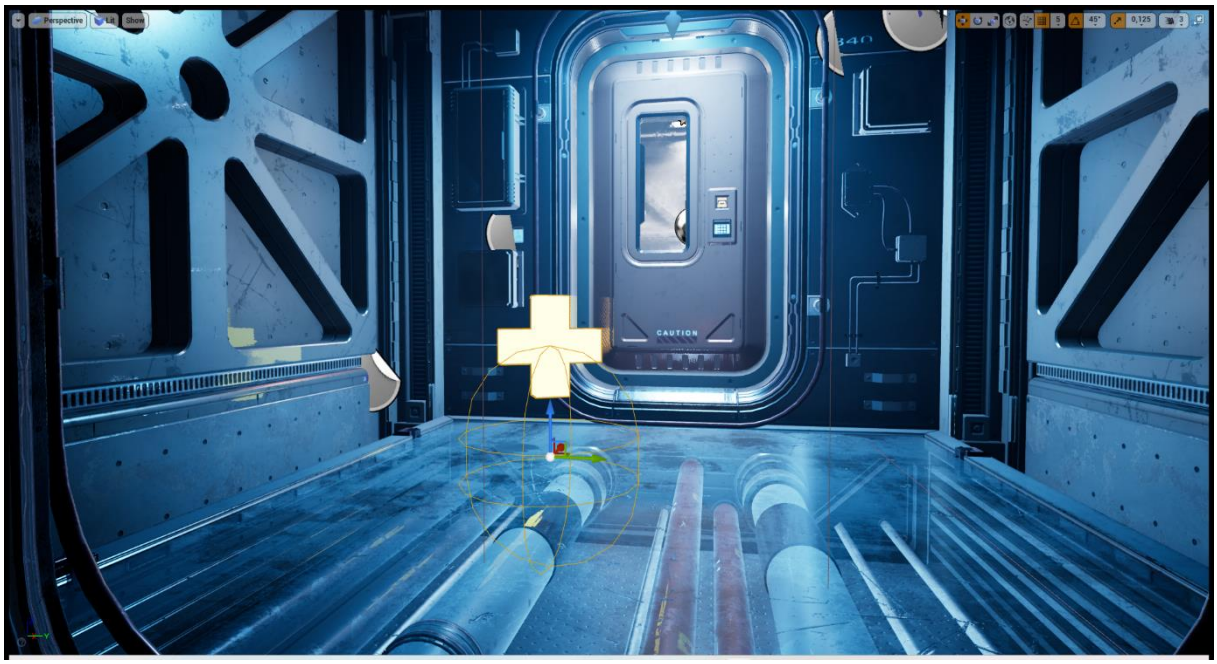
Όταν ο παίκτης είναι στη ζώνη που ενεργοποιείται η επικάλυψη γεωμετρίας, λίγο πριν εμφανιστεί το μήνυμα για αλληλεπίδραση δίνεται στο ThesisGameHUD η τιμή της μεταβλητής Prompt του Pickup. Αυτή ορίζει τι κείμενο θα εμφανιστεί στη θέση του προεπιλεγμένου Activate για το αντικείμενο.

3.14.1 Σφαίρες (PickupAmmo)

Εικόνα 3.21: Ammo Pickup

Η υποκλάση **PickupAmmo** υλοποιεί την συνάρτηση **TransferPickup** όπως είναι αναμενόμενο και θέτει νέες μεταβλητές, τη ποσότητα των πυρομαχικών, το τύπο όπλου που θα φορτωθούν τα πυρομαχικά και μια βοηθητική συνάρτηση που ελέγχει αν ο τύπος όπλου ταιριάζει με αυτόν που κρατάει το πιόνι που θέλει να πάρει το αντικείμενο. Όταν ο παίκτης πατήσει το ‘E’ για να πάρει τα πυρομαχικά, η συνάρτηση **TransferPickup** ελέγχει αν το όπλο το οποίο κρατάει ο παίκτης είναι συμβατό με τον τύπο όπλου για τον οποίο είναι αυτά τα πυρομαχικά. Αν ισχύει η συνθήκη καλεί την συνάρτηση γεμίσματος του όπλου, **AddAmmo** πάνω στην κλάση του όπλου του παίκτη έτσι ώστε να γεμίσει με όσα πυρομαχικά χρειάζεται. Η συνάρτηση **AddAmmo** επιστρέφει όσα πυρομαχικά είναι περιττά για διευκόλυνση, έτσι το Pickup αφαιρεί από τα διαθέσιμα όσα χρησιμοποιήθηκαν και δεν καταστρέφει το αντικείμενο πριν ο παίκτης πάρει ολόκληρο το πακέτο. Αυτό το επιτυγχάνει με το να επιστρέφει false κάθε φορά που περισσεύουν πυρομαχικά. Αν επιστρέψει true τότε καταστρέφεται το αντικείμενο και θεωρείται ότι ο παίκτης άδειασε το κουτί με τα πυρομαχικά.

3.14.2 Κουτί πρώτων βοηθειών (Medpack Pickup)

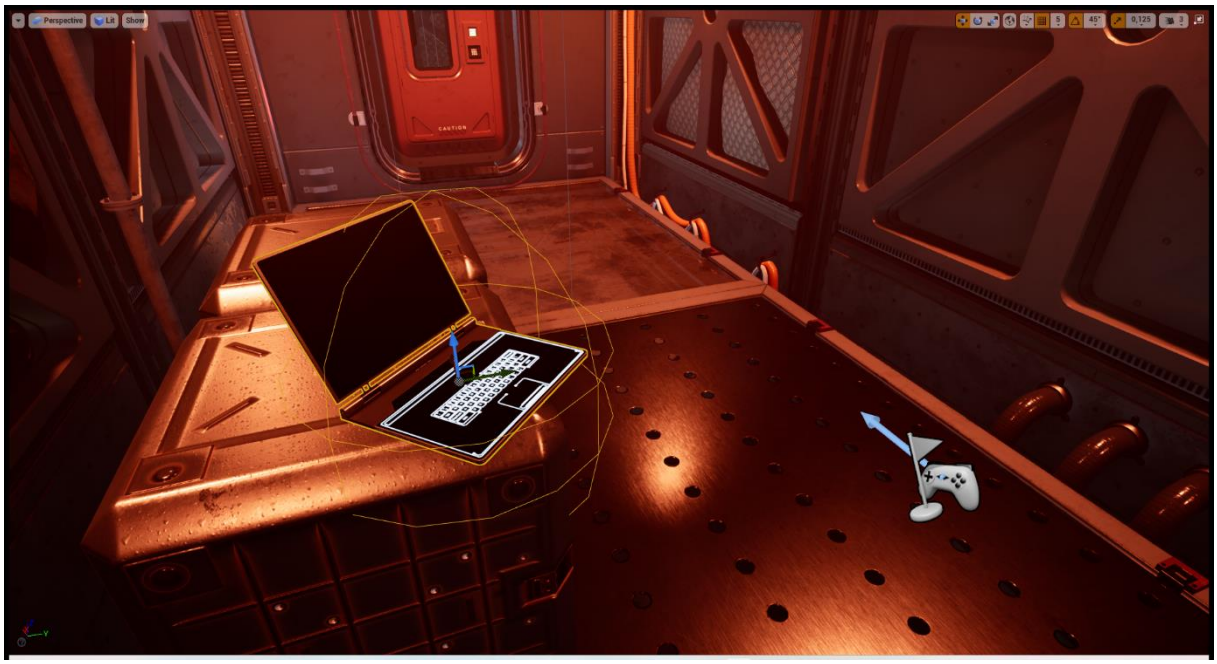


Εικόνα 3.22: Κουτί πρώτων βοηθειών

Το κουτί πρώτων βοηθειών γεμίζει τους πόντους ζωής του παίκτη και τον κρατάει περισσότερο στη ζωή. Υλοποιεί την παραπάνω συνάρτηση, την **TransferPickup** και δίνει στον παίκτη πόντους ζωής όσους έχει διαθέσιμους το αντικείμενο. Εδώ, επειδή είναι κουτί πρώτων βοηθειών θεωρείται ότι το χρησιμοποιεί ολόκληρο για να γιατρευτεί οπότε καταστρέφεται απευθείας μετά την χρήση του επιστρέφοντας πάντα true.

3.14.3 Τερματικό αποθήκευσης του παιχνιδιού (Terminal Pickup)

Το τερματικό αποθήκευσης είναι ένα λάπτοπ στο οποίο ο παίκτης μπορεί να σώσει την πρόοδο του και την κατάσταση του παιχνιδιού.



Εικόνα 3.23: Το τερματικό αποθήκευσης

Για να σώσει το παιχνίδι ο παίκτης πρέπει να πάει κοντά στο τερματικό και να πατήσει κάποιο κουμπί για να το ενεργοποιήσει. Οι ανάγκες του αντικειμένου αυτού ικανοποιούνται από την κλάση Pickup και για αυτό δημιουργήθηκε ως υποκλάση του και υλοποιεί την ίδια συνάρτηση **TransferPickup**. Η διαφορά με τα προηγούμενα αντικείμενα είναι ότι το τερματικό τεχνικά είναι ένα διακοσμητικό στον χώρο και ο παίκτης πρέπει να μπορεί να σώσει την πρόοδο του οποτεδήποτε θελήσει και παραπάνω από μια φορά. Άρα επιστρέφει πάντοτε false στην συνάρτηση παρόλο που τρέχει με επιτυχία ο κώδικας αποθήκευσης.

Το τερματικό διαθέτει μια μεταβλητή για το όνομα της θήκης του αρχείου αποθήκευσης (SaveSlot), τον δείκτη αποθήκευσης (SaveIndex) και το όνομα από το ενωμένο σημείο αναφοράς (LinkedCheckpoin). Για να αποθηκεύσει το παιχνίδι καλεί την συνάρτηση **SaveGame** της τρέχουσας περιόδου λειτουργίας (κλάση **ThesisGameInstance**) με όρισμα τον εαυτό του. Ανακοινώνει δηλαδή τον εαυτό του στην περίοδο λειτουργίας στη μορφή δείκτη έτσι ώστε η μηχανή να αποθηκεύσει για σημείο αφετηρίας αυτό που είναι συνδεδεμένο με το τερματικό. Επιπλέον, γίνεται έλεγχος αν η μηχανή κάνει ήδη αποθήκευση δεδομένων και απορρίπτει αλλεπάλληλα αιτήματα αποθήκευσης σε μικρό χρονικό διάστημα.

3.14.4 Πόρτα για φόρτωση χάρτη (BP_Door_A_Loading, BP_Door_B_Loading)

Εικόνα 3.24: Πόρτα φόρτωσης

Το συγκεκριμένο αντικείμενο δημιουργήθηκε πλήρως σε blueprint υλοποιώντας την βασική κλάση APickup και υπερφορτώνοντας την συνάρτηση TransferPickup. Αυτή η πόρτα δεν έχει κίνηση αλλά μεταφέρει τον παίκτη από χάρτη σε χάρτη κρατώντας την κατάσταση του παιχνιδιού. Επειδή μπορούμε να έχουμε αρκετές πόρτες τέτοιου τύπου διατηρεί μια μεταβλητή με το όνομα του χάρτη που θέλουμε να φορτώσουμε έτσι ώστε να είναι πλήρως παραμετροποιήσιμη στον Editor.

Όπως και στο τερματικό αποθήκευσης, έτσι και εδώ η συνάρτηση επιστρέφει πάντα false ώστε να μην εξαφανιστεί η πόρτα καθώς δεν την «συλλέγουμε» στο οπλοστάσιο μας. Όταν καλείται η συνάρτηση TransferPickup η πόρτα δημιουργεί ένα προσωρινό αρχείο αποθήκευσης (autosave) με την συνάρτηση **SaveGame** του ThesisGameGameMode και έπειτα το φορτώνει έτσι ώστε να δώσει στο GameMode να καταλάβει ότι στην επόμενη φόρτωση χάρτη θα πρέπει να χρησιμοποιήσει το αρχείο. Κατόπιν καλεί την συνάρτηση μηχανής Open Level με το όνομα του επόμενου χάρτη.

3.15 Αποθήκευση παιχνιδιού (ThesisSaveGame)

Η κλάση ThesisSaveGame είναι η οντότητα που αποθηκεύει τα δεδομένα του παιχνιδιού σε ένα αρχείο στον υπολογιστή. Είναι παιδί της βασικής κλάσης της μηχανής USaveGame και περιλαμβάνει επιπλέον μεταβλητές για κάθε τι που χρειάζεται να αποθηκευτεί [59]. Υπάρχουν συναρτήσεις που μπορούν να μετατρέψουν ολόκληρες κλάσεις με τα τρέχοντα δεδομένα της μνήμης σε δυαδικές σειρές και να περαστούν σε αρχείο, οι σειριοποιητές (Serializers), με αποτέλεσμα μια κλάση να χρειάζεται μόνο δύο μεταβλητές, μια δυαδική σειρά εισόδου και εξόδου για να λειτουργήσει, αλλά εμπεριέχει κίνδυνο κενών αναφορών και ανεξήγητων διακοπών λειτουργίας της μηχανής [60]. Για την παρούσα εργασία επιλέχθηκε η πιο χειροκίνητη μέθοδος αποθήκευσης. Οι χρήσιμες τιμές της μνήμης να αποθηκεύονται σε μεταβλητές και να καταγράφονται σε ένα αρχείο. Όταν έρθει η ώρα της φόρτωσης, να δημιουργούνται όλα τα αντικείμενα στον χώρο λαμβάνοντας πληροφορία από τις μεταβλητές της κλάσης αποθήκευσης αντί από τα προεπιλεγμένα της δικής τους κλάσης.

Για να θεωρείται αποθηκευμένο το παιχνίδι έπρεπε να καταγράφει πληροφορίες για τον παίκτη και την τρέχουσα θέση του στον κόσμο, για κάθε αντίπαλο που υπάρχει στον χώρο και για τις αποστολές. Οπότε γίνεται χρήση πολλών δομών δεδομένων καθώς αυτές περιέχουν τιμές και όχι δείκτες, γεγονός που τις

καθιστά κατάλληλες για απευθείας αποθήκευση σε αρχείο. Συγκεκριμένα χρησιμοποιήθηκε μια δομή τύπου **FActorData** για τις πληροφορίες του παίκτη, ένας πίνακας δομών **FActorData** για τους εχθρούς, ένας πίνακας δομών **FQuestData** για τις τρέχουσες ενεργές αποστολές και ένας ακόμα για το σύνολο των μέχρι το σημείο εκείνο αποστολών. Για το τρέχοντα φορτωμένο χάρτη χρησιμοποιήθηκε το όνομα του χάρτη και το τελευταίο σημείο αφετηρίας, το checkpoint.

Το ίδιο το αντικείμενο της κλάσης **ThesisSaveGame** χρησιμοποιείται μόνο για εγγραφή και διάβασμα δεδομένων ενώ η πραγματική λειτουργία της αποθήκευσης γίνεται στην περίοδο λειτουργίας του παιχνιδιού, έτσι ώστε να έχει πρόσβαση σε κάθε πληροφορία αναφορικά με το αρχείο αποθήκευσης.

3.16 Τύπος παιχνιδιού (ThesisGameGameMode)

Ο τύπος παιχνιδιού χειρίζεται πληροφορίες για το τρέχον παιχνίδι και θέτει κανόνες που θα ισχύουν όσο παίζει το συγκεκριμένο παιχνίδι [61]. Χρησιμοποιείται εκτενώς σε βιντεοπαιχνίδια με πολλούς παίκτες όπου χωρίζονται σε ομάδες και παίζουν αγώνες με χρονικό όριο. Στο παρόν έργο που αποτελεί μια περιπέτεια γραμμικής εξέλιξης χρησιμοποιήθηκε σαν αποθήκη για τα διάφορα μενού που χρησιμοποιούνται στο παιχνίδι, σαν γεννήτρια για αποστολές και ως χειριστής για κάποιες διαδικασίες που αφορούν τον παίκτη, δηλαδή χειριστής για το που θα εμφανιστεί ο παίκτης στον κάθε κόσμο και τι θα γίνει αν πεθάνει.

Κατά την φόρτωση ενός χάρτη μετά από αρκετές κλήσεις σε κλάσεις της μηχανής καλείται το **BeginPlay** του **GameMode**, εδώ του **ThesisGameGameMode [**62**]**. Πριν η κλάση δημιουργήσει αντικείμενα δένει το γεγονός **OnPlayerDeath** με μια καινούργια ταυτότητα συνάρτησης, ο τύπος της οποίας υπάρχει δηλωμένος στην κεφαλίδα της κλάσης. Έπειτα ζητάει από το **ThesisGameHUD** να αλλάξει το τρέχον μενού στο εναρκτήριο μενού (από MainMenuWidget να μεταβεί στο StartingWidget) έτσι ώστε μόλις φορτωθεί ο παίκτης στο χώρο να βλέπει έναν σταυρό για σκόπευτρο και τους πόντους ζωής του στη κάτω δεξιά γωνία και όχι το κεντρικό μενού με τις επιλογές για νέο παιχνίδι ή ρυθμίσεις. Επιπλέον δημιουργείται ένας χρονομετρητής που θα εκτελεστεί στο επόμενο πλαίσιο (**SetTimerForNextTick**) και δένεται με την συνάρτηση δημιουργίας αποστολών. Αν το παιχνίδι περιλαμβάνει δεδομένα στον δείκτη μνήμης του ThesisSaveGame θεωρείται ότι προέρχεται από φόρτωση προηγούμενου παιχνιδιού γιατί μόνο τότε γεμίζει ο δείκτης με δεδομένα. Οπότε φορτώνονται οι αποστολές από το αρχείο στον υπολογιστή (**SpawnLoadedQuestList**). Αλλιώς, φορτώνονται καινούργιες αποστολές (**SpawnDefaultQuestList**) σύμφωνα με μία λίστα από προκαθορισμένες κλάσεις αποστολών.

Ταυτόχρονα με την απόφαση για τις αποστολές καλούνται οι συναρτήσεις **StoreDefaultBots** και **StoreLoadedBots** αντίστοιχα για καινούργια/παλαιά συνεδρία. Η λέξη Store χρησιμοποιείται γιατί οι εχθροί (bots) είναι φορτωμένοι στον χάρτη εξ αρχής και το παιχνίδι χρειάζεται απλά να καταχωρήσει την τρέχουσα κατάσταση τους ή να αποφασίσει για αυτήν. Αν οι εχθροί είναι ζωντανοί τότε καταχωρούνται σε μια δομή δεδομένων για τους ζωντανούς, αλλιώς σε μια δομή για τους νεκρούς. Αυτό ήταν απαραίτητο έτσι ώστε στην επαναφόρτωση του παιχνιδιού, να μην «ανασταίνονται» οι εχθροί. Για την επίτευξη αυτού χρησιμοποιείται ένας μοναδικός αριθμός για κάθε εχθρό που υπάρχει σε οποιοδήποτε χάρτη. Έτσι το παιχνίδι καταχωρεί τον κάθε εχθρό και μπορεί να κάνει έλεγχο για να φορτώσει την παλαιά του κατάσταση αν ήταν ζωντανός κατά την αποθηκευμένη συνεδρία ή να τον αφαιρέσει από το χώρο καλώντας τον αποδομητή του (**Destroy**).

Η δημιουργία αποστολών γίνεται με μια επανάληψη για κάθε αντικείμενο στον πίνακα προκαθορισμένων ή στον πίνακα των αποθηκευμένων κλάσεων αποστολής. Για κάθε κλάση καλείται η πρότυπη (template) συνάρτηση **SpawnActor** με τύπο επιστρεφόμενης τιμής **AQuest** και ορίσματα

την κλάση ή υποκλάση της αποστολής και τις παραμέτρους φόρτωσης. Η SpawnActor ανήκει στη κλάση UWorld οπότε για να κληθεί πρέπει να υπάρχει φορτωμένος ένας κόσμος στο παιχνίδι. Στην ιεραρχία εκτέλεσης της μηχανής το GameMode είναι πιο κάτω από το UWorld οπότε είναι απόλυτα ασφαλές να πάρουμε τον φορτωμένο κόσμο με την καθολική συνάρτηση GetWorld εντός του BeginPlay στο GameMode. Οι παράμετροι φόρτωσης είναι μια δομή δεδομένων τύπου FActorSpawnParameters και η τιμή που πρέπει να αλλαχθεί είναι το SpawnHandlingCollisionOverride. Πρέπει να τεθεί ως AlwaysSpawn έτσι ώστε οι αποστολές να δημιουργούνται στον χώρο, στο διάνυσμα (0, 0, 0) για X, Y, Z συντεταγμένες αγνοώντας το αν το γεωμετρικό περίβλημα βρίσκεται μέσα σε άλλη γεωμετρία. Σε διαφορετική περίπτωση, αν οι αποστολές είχαν τρισδιάστατη αναπαράσταση στον χώρο θα χρειαζόταν μια συνάρτηση εύρεσης της καλύτερης θέσης για να τοποθετηθούν στον χώρο χωρίς να επικαλύπτει ή να διαπερνάει οποιαδήποτε άλλη γεωμετρία το περίβλημα τους. Κάτι τέτοιο απαιτείται για τον παίκτη και τους εχθρούς.

Η επιλογή σημείου φόρτωσης του παίκτη γίνεται στην εικονική συνάρτηση ChoosePlayerStart και σχεδόν αντικαθιστά την βασική μεθοδολογία της. Με την καθολική συνάρτηση μηχανής **GetAllActorsOfClass** γίνεται εύρεση όλων των αντικειμένων **APlayerStart**, μιας βασικής κλάσης της μηχανής που χειρίζεται το που θα φορτωθεί ο παίκτης. Το αποτέλεσμα της αποθηκεύεται σε έναν πίνακα δεικτών κλάσης και με μια επανάληψη TActorIterator ελέγχεται αν το καρτελάκι κάθε αντικειμένου έχει όνομα που ανήκει στα προκαθορισμένα που ορίστηκαν στην περίοδο λειτουργίας **ThesisGameInstance**. Τα καρτελάκια είναι ένας τρόπος να γίνει αναζήτηση για αντικείμενα που καλύπτουν προϋποθέσεις εντός του παιχνιδιού και επειδή είναι τύπου FName, μπορεί να γίνει αναζήτηση σε πραγματικό χρόνο με αρκετά μεγάλη ταχύτητα και αμελητέα καθυστέρηση. Εδώ, έχουν οριστεί δύο καρτελάκια στην περίοδο λειτουργίας, το FirstPlayerStartName και το CheckpointPlayerStartName. Ο έλεγχος για το προηγούμενο σημείο γίνεται πρώτος έτσι ώστε να δοθεί προτεραιότητα έναντι του αρχικού σημείου. Αν βρεθεί κάποιο PlayerStart με καρτελάκι που ταιριάζει στα προκαθορισμένα, επιστρέφεται ο δείκτης του, αλλιώς καλείται η βασική συνάρτηση μηχανής. Αν παραλειφθεί η κλήση της βασικής συνάρτησης το Unreal Engine αποτυγχάνει να ολοκληρώσει την εκτέλεση του κύκλου εντολών και σε περίπτωση που το εκτελέσιμο βρίσκεται εκτός του χώρου του Editor ενδέχεται να προκληθεί BSOD λόγω αστοχίας εύρεσης αναφορών στη μνήμη.

Η συνάρτηση επαναφόρτωσης του παιχνιδιού εκτελείται σε 5 δευτερόλεπτα αφότου ανακοινωθεί ο θάνατος του παίκτη, δηλαδή όταν ο παίκτης κάνει broadcast το γεγονός **OnPlayerDeath**. Στο GameMode υπάρχει μια έτοιμη συνάρτηση επαναφόρτωσης του παίκτη (**RestartPlayer**) αλλά το παιχνίδι παραμένει στην ίδια κατάσταση. Είναι ιδανική για άλλους τύπους παιχνιδιού όπου ο θάνατος του παίκτη μετράει αρνητικά σε κάποια βαθμολογία αλλά σε αυτό το έργο ο θάνατος του παίκτη έχει άλλη βαρύτητα, θεωρείται ότι αποτυγχάνει στην αποστολή του και πρέπει να προσπαθήσει ξανά. Οπότε αντί για επαναφόρτωση του παίκτη γίνεται κλήση στην συνάρτηση LoadGame της περιόδου λειτουργίας και γεμίζει ο δείκτης του **ThesisSaveGame** με τα δεδομένα από το αρχείο. Έπειτα καλείται η συνάρτηση μηχανής **OpenLevel** με ορίσματα το τρέχον GameMode, τον τρέχοντα κόσμο στον οποίο πέθανε ο παίκτης και τη σημαία bAbsolute true για να καθαρίσει τον κόσμο από αλλαγές και να φορτώσει τα πάντα εκ νέου.

3.17 Κατάσταση παιχνιδιού (ThesisGameGameState)

Η κατάσταση παιχνιδιού (GameState) είναι υπεύθυνη για τον έλεγχο διαφόρων γεγονότων που γίνονται στο παιχνίδι και για την αποθήκευση δεδομένων σχετικών με το παιχνίδι και το τύπο παιχνιδιού αλλά

όχι τους παίκτες. Στην παρούσα εργασία το ThesisGameGameState είναι ένας χειριστής των αποστολών που τρέχουν στο παιχνίδι. Ο κώδικας χειρισμού αποστολών τοποθετήθηκε εδώ γιατί η κατάσταση του παιχνιδιού διατηρεί όλες τις αλλαγές που γίνονται σε μια συνεδρία παιχνιδιού άσχετα από το που βρίσκεται ο παίκτης, σε ποιόν κόσμο και με ποιο τύπο παιχνιδιού. Ο δομητής της κατάστασης δεν περιέχει τίποτα καθώς δεν κρίθηκε απαραίτητο να αρχικοποιούνται τιμές.

Κάθε φορά που ενεργοποιείται μια αποστολή καλείται η συνάρτηση AddQuest της κατάστασης για να προστεθεί στην λίστα (πίνακας ονομάτων FName) με τις ενεργές αποστολές αλλά και στο σύνολο όλων των αποστολών. Με την συνάρτηση GetCurrentQuests μπορεί να βρεθεί το σύνολο των ενεργών αποστολών που τρέχουν σε μια συνεδρία παιχνιδιού.

Έπειτα αν η αποστολή έχει οριστεί να γίνεται αυτόματα η ενεργή εκχωρείται η τιμή της στον δείκτη της ενεργής αποστολής. Έτσι, όταν το παιχνίδι αναφέρεται από οπουδήποτε στην ενεργή αποστολή θα μπορεί να την βρει εύκολα μέσω της συνάρτησης GetActiveQuest έναντι να κάνει αναζήτηση στους πίνακες αποστολών. Επίσης, μπορεί να υπάρχει μόνο μια ενεργή αποστολή κάθε φορά καθώς η διαχείριση ενός μόνο δείκτη είναι ευκολότερη.

Όταν μια αποστολή ολοκληρώνεται καλείται η συνάρτηση RemoveQuest όπου ελέγχεται αν δεν είναι ολοκληρωμένη έτσι ώστε να κληθεί η συνάρτηση CompleteQuest και έπειτα αφαιρείται από τον πίνακα των ενεργών αποστολών και τοποθετείται στον πίνακα των ολοκληρωμένων. Έτσι με τη συνάρτηση GetCompletedQuests μπορεί να βρεθεί εύκολα το σύνολο των ολοκληρωμένων αποστολών.

Με την συνάρτηση FindQuest μπορεί να γίνει γραμμική αναζήτηση στις τρέχουσες αποστολές και αν η κλάση της αποστολής ανήκει στον πίνακα, επιστρέφεται ένας σταθερός δείκτης της αποστολής.

3.18 Συνεδρία παιχνιδιού (ThesisGameInstance)

Η συνεδρία του παιχνιδιού (GameInstance) είναι υψηλού επιπέδου χειριστής για τη τρέχουσα συνεδρία. Στο έργο αυτό αποφασίστηκε να χρησιμοποιηθεί ως χειριστής του συστήματος αποθήκευσης και φόρτωσης έτσι ώστε να είναι συγκεντρωμένες οι λειτουργίες αυτές σε μια κλάση και πλήρως προσβάσιμες από οποιοδήποτε συστατικό του παιχνιδιού. Η ιδέα ήταν να αποθηκεύονται γραμμικά τα δεδομένα που αφορούν τις αλλαγές που έγιναν στο κόσμο και να φορτώνονται σε έναν δείκτη τύπου **UThesisSaveGame** κατά την επαναφόρτωση του παιχνιδιού. Έπειτα ο κάθε δομητής κάθε κλάσης που έπρεπε να αρχικοποιήσει δεδομένα από το αρχείο, να διαβάζει την τιμή του δείκτη και να βρίσκει τις πληροφορίες που ενδιαφέρουν την δική του κλάση. Έτσι μειώνονται δραστικά οι πιθανότητες να φορτωθεί κάτι λάθος αφού η ευθύνη ανήκει σε κάθε κλάση αντικειμένων ξεχωριστά και η ευθύνη της συνεδρίας είναι μόνο να διαβάσει τα δεδομένα και να τα καταγράψει σωστά.

Η συνάρτηση SaveGame αρχικοποιεί πρώτα το αντικείμενο αποθήκευσης, ονόματι GameData με την καθολική συνάρτηση **CreateSaveGameObject**. Η συνάρτηση αυτή εκχωρεί ένα χωρίο της μνήμης σε μια μεταβλητή που προορίζεται για δυναμικό γέμισμα με δεδομένα σύμφωνα με την κλάση εισόδου, εδώ την **UThesisSaveGame**. Φορτώνει δύο δείκτες, έναν που δείχνει στον παίκτη και έναν στη κατάσταση του παιχνιδιού. Ελέγχει αν υπάρχει δείκτης στο αντικείμενο που την κάλεσε (Caller), συνήθως ένα τερματικό έτσι ώστε να αντλήσει πληροφορία για το ενωμένο με το τερματικό σημείο φόρτωσης του παίκτη (LastCheckpoint). Έπειτα κάνοντας χρήση των βοηθητικών συναρτήσεων της κατάστασης βρίσκει όλες τις τρέχουσες αποστολές αλλά και το σύνολο των αποστολών, τις οποίες έπειτα προσθέτει σε πίνακες δομών τύπου FQuestData, δημιουργώντας μια δομή δεδομένων για κάθε αποστολή. Με την ίδια λογική αποθηκεύει δεδομένα για τον παίκτη και για κάθε φορτωμένο εχθρό σε μια δομή ή πίνακα δομών τύπου FActorData.

Τέλος, αφού έχει γεμίσει το αντικείμενο αποθήκευσης με πληροφορίες αποφασίζει αν θα καλέσει την εσωτερική συνάρτηση αποθήκευσης ασύγχρονα ή σύγχρονα, σύμφωνα με μια σημαία εισόδου, την Boolean bUseAsync. Η ασύγχρονη υλοποίηση απαιτεί να δεθεί μια συνάρτηση αντιπρόσωπος (SaveDone) με τη ταυτότητα αντιπροσώπου FAsyncSaveGameToSlotDelegate και να κληθεί η συνάρτηση AsyncSaveGameToSlot με όρισμα αυτόν τον αντιπρόσωπο. Αυτός θα εκτελεστεί όταν ολοκληρωθεί η εκτέλεση του εσωτερικού κώδικα αποθήκευσης και επιστραφεί true ή false ανάλογα με την επιτυχία ή αποτυχία ανάγνωσης του αρχείου. Η αντιπρόσωπος συνάρτηση πρέπει να έχει ίδιο αριθμό και τύπο εισόδων με την ταυτότητα, όπως αυτή ορίζεται από την μηχανή, δηλαδή μια σταθερά για τη θήκη αποθήκευσης, μια σταθερά για τον δείκτη του χρήστη και μια Boolean για την επιτυχία ή αποτυχία της αποθήκευσης.

Προκειμένου να αποφευχθεί η αλλεπάλληλη κλήση στην ασύγχρονη συνάρτηση δημιουργώντας μια ουρά διεργασιών που προσπαθούν να σώσουν περιεχόμενο στην ίδια μεταβλητή και πιθανόν λογικά λάθη, χρησιμοποιήθηκε μια σημαία Boolean, η PendingSave που γίνεται true πριν ξεκινήσει η ασύγχρονη διαδικασία αποθήκευσης και false στο μπλοκ εντολών της συνάρτησης αντιπροσώπου SaveDone.

Η φόρτωση των δεδομένων από το αρχείο αποθήκευσης στην μεταβλητή GameData γίνεται στην συνάρτηση LoadGame όπου αποφασίζει αν θα κάνει ασύγχρονη ή σύγχρονη φόρτωση δεδομένων ομοίως με την αποθήκευση. Αν γίνει ασύγχρονη φόρτωση ενεργοποιείται η σημαία PendingLoad και καλείται η συνάρτηση AsyncLoadGameFromSlot με συνάρτηση αντιπρόσωπο την LoadDone. Αντίθετα με τον αντιπρόσωπο της αποθήκευσης, στην φόρτωση για είσοδος υπάρχει ένας δείκτης μνήμης που γεμίζει με τα δεδομένα από το αρχείο και αυτός ο δείκτης εκχωρείται στον δείκτη GameData έτσι ώστε τα δεδομένα του παιχνιδιού να είναι προσβάσιμα από όλες τις κλάσεις ακόμα και μετά την κλήση της φόρτωσης. Αλλιώς θα έπρεπε κάθε φορά να διαβάζονται τα δεδομένα από το αρχείο αποθήκευσης, γεγονός που θα πρόσθετε επιπλέον φόρτο IO στον σκληρό δίσκο.

Αφού γεμίσει ο δείκτης GameData καλείται η συνάρτηση LoadFillData όπου αρχικός σκοπός της ήταν να φορτώνει στο χώρο του παιχνιδιού όλα όσα περιέχονται στον δείκτη μνήμης. Όμως τελικά χρησιμοποιήθηκε για να εκχωρεί νέα τιμή στη τελευταία θέση του παίκτη (CheckpointPlayerStartName) και να ανοίγει ο κόσμος που ήταν φορτωμένος κατά την αποθήκευση με true στη σημαία bAbsolute έτσι ώστε να μην κρατήσει καμία αλλαγή και να φορτώσει τα πάντα εκ νέου (προκαλώντας εξαναγκασμένη κλήση στους δομητές των αντικειμένων και κάνοντας τους να διαβάσουν πληροφορίες από το GameData αντί από τα δικά τους προκαθορισμένα.

3.19 HUD (ThesisGameHUD)

Το HUD είναι ένας χειριστής των μενού που εμφανίζονται στην οθόνη του χρήστη, ανήκει στον παίκτη και περιλαμβάνει βασικές συναρτήσεις εμφάνισης κειμένου, δισδιάστατων αντικειμένων και πληροφοριών αποσφαλμάτωσης. Για τον ίδιο ακριβώς σκοπό χρησιμοποιήθηκε σε αυτό το έργο, δηλαδή ως χειριστής όλων των widget που συναντάμε εντός του παιχνιδιού. Συγκεκριμένα το **ThesisGameHUD** διαθέτει έναν απαριθμητή, το **EInteractionTypes** για το κείμενο που θα εμφανιστεί αν ο παίκτης θέλει να πάρει αντικείμενα Pickup, συναρτήσεις ενεργοποίησης/απενεργοποίησης του μενού αλληλεπίδρασης με τα αντικείμενα Pickup, μια συνάρτηση αλλαγής του κεντρικού μενού και μια συλλογή συναρτήσεων για την εμφάνιση διαφόρων μηνυμάτων στην οθόνη του χρήστη.

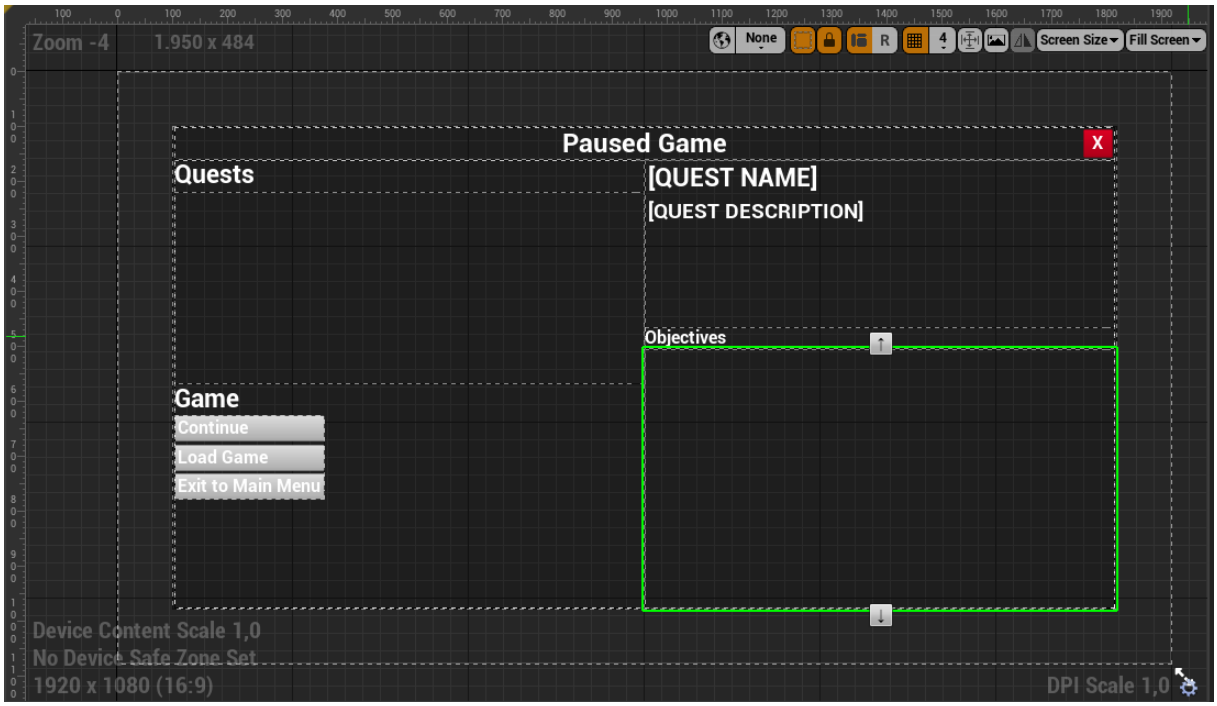
Όταν ο παίκτης φτάνει κοντά σε ένα αντικείμενο και καλείται η συνάρτηση επικάλυψης του αντικειμένου, ζητείται από το **ThesisGameHUD** να αποφασίσει τι τύπου αντικείμενο είναι αυτό που

βλέπει ο παίκτης και να εμφανίσει το κατάλληλο μήνυμα στην οθόνη από τα δεδομένα του απαριθμητή. Για αυτόν τον σκοπό χρησιμοποιήθηκε η καθολική συνάρτηση μηχανής των απαριθμητών **GetDisplayValueAsText** όπου παίρνει την δυαδική τιμή του ενεργού στοιχείου του απαριθμητή και το μεταφράζει σε κείμενο. Για την σωστή μετάφραση χρειάζεται το μεταδεδομένο UMETA DisplayName να έχει οριστεί στο στοιχείο που πρόκειται να μεταφραστεί. Η απόφαση για τον τύπο αντικειμένου γίνεται με ανάθεση στο αντικείμενο pickup και αν όντως βρεθεί ένα τέτοιο αντικείμενο, καλείται η συνάρτηση GetPrompt που επιστρέφει την τιμή του απαριθμητή που αντιστοιχεί σε μια ενέργεια.

Εικόνα 3.25: Διεπαφή που εμφανίζεται στα Pickup, επάνω αριστερά

Στο ίδιο σενάριο εμφάνισης του μενού αλληλεπίδρασης στον παίκτη, ζητείται να γίνει εμφάνιση ή απόκρυψη του μενού ανάλογα με το αν ο παίκτης επικαλύπτει την γεωμετρία πρόσκρουσης του Pickup. Όλα τα μενού που χρησιμοποιήθηκαν στην εργασία είναι κατασκευασμένα με το σύστημα μενού UMG του Unreal Engine και όλα είναι υποκλάσεις της UUserWidget. Η εμφάνιση μενού UMG γενικά γίνεται με την εντολή του HUD **CreateWidget** με είσοδο μια υποκλάση του UUserWidget και τον κόσμο στον οποίο είναι να φορτωθεί το μενού. Έπειτα συνδέεται με τον παίκτη ή τον κόσμο με την εντολή **AddToPlayerScreen** ή **AddToViewport** αντίστοιχα. Για να μην δημιουργηθούν διπλότυπα μενού, είναι καλή πρακτική να υπάρχει ένας δείκτης που γεμίζει με την τιμή ενός μενού και κατά την δημιουργία ενός νέου να καλείται η συνάρτηση **RemoveFromViewport** στον ήδη γεμάτο δείκτη.

Η συνάρτηση **EnableQuestLogWidget** χρησιμοποιήθηκε για να δείχνει αρχικά τις τρέχουσες αποστολές και τελικά έγινε η κύρια συνάρτηση του μενού παύσης. Οπότε εκτός από το να εμφανίσει το μενού αποστολών, παίρνει έναν δείκτη στον χειριστή του παίκτη και καλεί τις συναρτήσεις **SetInputMode**, **SetPause** και **SetCinematicMode** στον χειριστή έτσι ώστε το παιχνίδι να μεταβεί σε κατάσταση παύσης, ο παίκτης να μπορεί να κινήσει μόνο το ποντίκι και να χρησιμοποιεί μόνο τα πλήκτρα που αντιστοιχούν σε ενέργειες μενού [63]. Στην συνάρτηση **SetInputMode** δίνεται ως όρισμα η δομή δεδομένων **FInputModeGameAndUI** που όπως περιγράφει το όνομα δίνει πρόσβαση στις εισόδου του παιχνιδιού και της διεπαφής χρήστη. Το CinematicMode αποτρέπει τον παίκτη από το να εκτελεί ενέργειες που αφορούν το παιχνίδι, αν τεθούν οι κατάλληλες Boolean τιμές.



Εικόνα 3.26: Η διεπαφή Quest Log στον Editor

Στην συλλογή συναρτήσεων για εμφάνιση/απόκρυψη μηνυμάτων ανήκουν η συνάρτηση **ShowMSGWidget** και ο αντιπρόσωπος **MarkWidgetForDeletion**. Κατά την κλήση της **ShowMSGWidget** δημιουργείται ένα **UUserWidget** με κλάση αυτήν που ορίζεται από την είσοδο και ένας χρονομετρητής που θα καλέσει τον αντιπρόσωπο **MarkWidgetForDeletion** σε χρόνο 5 δευτερολέπτων με είσοδο τον δημιουργημένο δείκτη μνήμης που έχει τιμή το νέο **UUserWidget**. Με αυτόν τον τρόπο μπορούμε να έχουμε περισσότερα από ένα μενού να εμφανίζονται διαδοχικά στην οθόνη και να καταστρέφονται με το πέρασμα του χρόνου. Ο αντιπρόσωπος **MarkWidgetForDeletion** ενώνεται κάθε φορά με την ταυτότητα της συνάρτησης **FTimerDelegate** με κλήση της **BindUFunction**. Η συνάρτηση **BindUFunction** είναι κατάλληλη για δέσιμο συναρτήσεων με εισόδους.

Η συνάρτηση **MarkWidgetForDeletion** όταν εκτελεστεί καλεί την **RemoveFromViewport** στον δυναμικά δημιουργημένο δείκτη και έπειτα τον αποδομητή έτσι ώστε να αδειάσει η μνήμη από αυτό το περιεχόμενο.

Εικόνα 3.27: Η διεπαφή QuestLog

3.20 Στάδιο 4ο (Post Production)

Σε αυτό το σημείο πλέον το παιχνίδι θεωρήθηκε πλήρως λειτουργικό εφόσον κατασκευάστηκαν όλες οι απαραίτητες κλάσεις, τα σκηνικά, τα γραφικά και οι μηχανικές του παιχνιδιού. Πέρασε στην άλφα έκδοση όπου έγιναν δοκιμές και σχεδόν αμέσως παρατηρήθηκαν κάποια λάθη για τα οποία υπήρχε η ανάγκη να χρησιμοποιηθούν τα εργαλεία αποσφαλμάτωσης του Visual Studio [9], [64], [65]. Ως έκδοση βήτα θεωρήθηκε η έκδοση κατά την οποία τα παρακάτω λάθη διορθώθηκαν και το παιχνίδι ήταν πλέον εκτελέσιμο από την αρχή ως το τέλος κανονικά.

3.20.1 Λάθος 1

Ο παίκτης ξεκινάει να προχωράει στο χώρο αφότου εμφανίζεται στην εξέδρα απογείωσης και τρέχει μπροστά αρκετά γρήγορα έτσι ώστε να περάσει μέσα από ένα έναυσμα που αρχίζει την πρώτη αποστολή. Όμως δεν υπάρχει αρκετός χρόνος για να αρχικοποιηθεί η αναφορά στην αποστολή με αποτέλεσμα να αποτυγχάνει ο κώδικας και ο παίκτης να συνεχίζει χωρίς αποστολή παρακάτω.

3.20.2 Λύση 1

Έπρεπε να αναπτυχθεί κώδικας που αναγκάζει τη μηχανή να ψάξει για την αναφορά στο αντικείμενο αποστολής αν αστοχήσει την πρώτη φορά. Οπότε, τοποθετήθηκε μια δεύτερη κλήση της συνάρτησης FindQuest κατά την επικάλυψη (Overlap) από τη γεωμετρία του παίκτη ώστε να είναι σίγουρο ότι ακόμα και με μικρή καθυστέρηση, θα υπάρχει πάντοτε ενεργή αναφορά στην αποστολή.

3.20.3 Λάθος 2

Ο παίκτης φτάνει στην αίθουσα με τους τρεις πρώτους αντιπάλους που πρέπει να εξουδετερώσει για να προχωρήσει αλλά επειδή δεν ανακοινώνουν σωστά τον θάνατο τους στο GameMode, δεν προχωρούσε ποτέ η αποστολή και επομένως το παιχνίδι.

3.20.4 Λύση 2

Το αντικείμενο που μετράει αν όλοι οι εχθροί είναι εκτός φτιάχτηκε σε blueprint. Οπότε το γεγονός του θανάτου τους μετατράπηκε σε Blueprintable έτσι ώστε να μπορεί να το αναγνωρίσει μέσω των Blueprint η μηχανή και να επιτρέπει να χρησιμοποιηθεί σε οπτικό προγραμματισμό. Έτσι, μέσω Blueprints ψάχνει η μηχανή για μια αναφορά στο αντικείμενο μετρητή και καλεί την συνάρτηση του για να αυξήσει τον μετρητή.

3.20.5 Λάθος 3

Ο παίκτης μπορεί να πέσει στο κενό από την αρχική εξέδρα αλλά η μηχανή δεν γνωρίζει τι να κάνει και απλά πέφτει στο άπειρο.

3.20.6 Λύση 3

Προστέθηκε Kill Z έναυσμα σε όλο το χάρτη που όταν το διαπεράσει ο παίκτης καλείται η συνάρτηση OnDeath και τον σκοτώνει λόγω πτώσης από μεγάλο υψόμετρο.

3.20.7 Λάθος 4

Οι αντίπαλοι που πέθαναν κατά την προηγούμενη συνεδρία εμφανίζονται ζωντανοί αν γίνει φόρτωση του παιχνιδιού από αποθηκευμένη κατάσταση.

3.20.8 Λύση 4

Χρειάστηκε αλλαγή το γεγονός ανακοίνωσης θανάτου OnPlayerDied. Πλέον γίνεται broadcast και από τους εχθρούς, το ThesisGameGameMode ελέγχει αν ο εν λόγω χαρακτήρας είναι ο παίκτης ή εχθρός και αν βρει τον παίκτη, καλεί την συνάρτηση επαναφόρτωσης του παιχνιδιού. Αν βρει εχθρό τον καταχωρεί στο ThesisGameState σε λίστα με νεκρούς χαρακτήρες έτσι ώστε στην επαναφόρτωση να μην τον ενεργοποιήσει.

3.20.9 Λάθος 5

Ο παίκτης αργεί να ανακοινώσει τον θάνατο του και καταστρέφεται το αντικείμενο του, αφήνοντας το μενού με τη ζωή και τα πυρομαχικά άδεια. Αυτό ήταν περισσότερο λάθος αισθητικής και εμπειρίας χρήστη.

3.20.10 Λύση 5

Η ανακοίνωση του θανάτου του πέρασε στο γεγονός OnDeath αντί για τον Destructor του με χρονομετρητή στα 5 δευτερόλεπτα. Ο παίκτης έχει τεθεί να καταστρέφεται στα 10 δευτερόλεπτα οπότε υπάρχει αρκετός χρόνος για να δημιουργηθεί μια νέα συνεδρία καθώς ο παλιός καταστρέφεται. Το αποτέλεσμα είναι οπτικά βελτιωμένο.

3.20.11 Λάθος 6

Προκαλείται μερική παραμόρφωση του μοντέλου του χαρακτήρα όταν πεθαίνει αρκετά κοντά σε κάγκελα ή πολύπλοκες επιφάνειες με αποτέλεσμα κατά την πτώση του να κολλάνε τα άκρα του στις επιφάνειες.

3.20.12 Λύση 6

Το μοντέλο να μην χρησιμοποιεί αρχείο κίνησης κατά τον θάνατο και να περνάει κατευθείαν σε κατάσταση ragdoll. Αυτό όμως προκάλεσε οπτικά κακό αποτέλεσμα αφού οι χαρακτήρες δεν φαίνονταν να έχουν πληγωθεί και απλώς πέφτουν χωρίς ορμή. Τελικά προτιμήθηκε η αρχική έκδοση.

3.21 Πακετάρισμα

Το πακετάρισμα του παιχνιδιού γίνεται από την επιλογή File – Package Project - *Προτιμώμενη πλατφόρμα. Για αυτό το έργο επιλέχθηκε η πλατφόρμα Windows και χρησιμοποιήθηκαν κυρίως οι βασικές ρυθμίσεις.

Απαραίτητη προϋπόθεση για να δημιουργηθεί το εκτελέσιμο του παιχνιδιού είναι να οριστεί ο προεπιλεγμένος χάρτης που θα φορτώνει η μηχανή όταν αρχίζει το παιχνίδι [66]. Επιλέχθηκε το Thesis_MainMenu και έπειτα όσοι χάρτες ήταν απαραίτητο να περάσουν στο τελικό αρχείο. Συμπεριλήφθηκαν οι τρεις χάρτες του παιχνιδιού, Thesis_Level1, Thesis_Level2 και Thesis_Level3 ενώ αφαιρέθηκαν τα δύο TestMaps.

Το τελικό αρχείο είναι διαθέσιμο ως συμπιεσμένο αρχείο εγκατάστασης και απαιτεί 1.44 Gb χώρο στον δίσκο για να κατέβει ενώ αποσυμπιεσμένα τα δεδομένα του παιχνιδιού καταλαμβάνουν 6.79 Gb.

3.22 Επίλογος

Ανακεφαλαιώνοντας είδαμε την διαδικασία ανάπτυξης ενός μίνι-βιντεοπαιχνιδιού με μια απλή αποστολή από την αρχή της δημιουργίας της μέχρι και το σημείο που αποτέλεσε ένα εκτελέσιμο αρχείο υπολογιστή. Ακολουθώντας τα στάδια παραγωγής και προσέχοντας σε κάθε σημείο τις τεχνικές λεπτομέρειες, ήταν δυνατό να κατασκευαστεί ένα πλήρως λειτουργικό περιβάλλον με διαδραστικά στοιχεία, έναν παίκτη με όπλο, εχθρούς με τεχνητή νοημοσύνη και ένα σύστημα αποστολών.

Κεφάλαιο 4ο: Τελικό προϊόν

4.1 Εισαγωγή

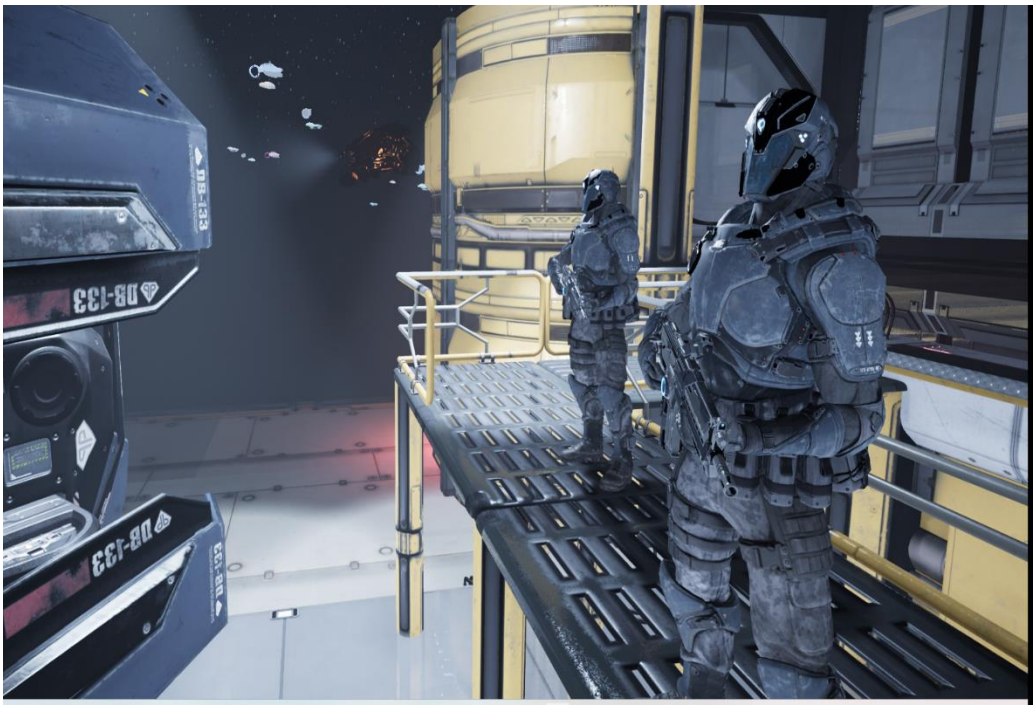
Μετά την ανάπτυξη και τον εκτενή έλεγχο λαθών, το βιντεοπαιχνίδι συμπιέζεται κατάλληλα σε ένα εκτελέσιμο αρχείο το οποίο αποτελεί το τελικό προϊόν. Ο εκάστοτε κατασκευαστής μπορεί πλέον να το διανέμει στην αγορά με ψηφιακή ή φυσική μορφή, μέσω κάποιας διαδικτυακής πλατφόρμας ή σε μορφή DVD δίσκου αντίστοιχα. Παρακάτω θα παρουσιαστεί συνοπτικά το τελικό παράγωγο του έργου καθώς και τα διαγράμματα που περιγράφουν την λειτουργικότητα του προγράμματος.

4.2 Αναφορικά με το βιντεοπαιχνίδι

Όπως κάθε βιντεοπαιχνίδι, έτσι και αυτό πρέπει να έχει ένα εξώφυλλο με συγκεντρωμένες όλες τις χρήσιμες πληροφορίες του όπως τη πλατφόρμα για την οποία πρόκειται να κυκλοφορήσει και τις απαιτήσεις συστήματος.

4.2.1 Βασικά στοιχεία

Εξώφυλλο:



Εικόνα 4.1: Εξώφυλλο

Τίτλος: Thesis Game

Παραγωγός: Σπυριδόπουλος Κωνσταντίνος

Είδος: Δράση πρώτου προσώπου

Ημερομηνία κυκλοφορίας: -

Πλατφόρμα: Windows

Προτεινόμενες απαιτήσεις συστήματος:

- Λειτουργικό σύστημα Windows έκδοσης 10 ή 11
- DirectX 11 συμβατή κάρτα γραφικών με 2GB VRAM και άνω
- Επεξεργαστής Intel i3 8ης γενιάς και άνω
- 8 GB RAM και άνω
- Τουλάχιστον 7Gb ελεύθερη χωρητικότητα στο δίσκο

4.2.2 Οδηγίες εγκατάστασης

Αφού μεταβούμε στο σύνδεσμο του Google Drive και κατεβάσουμε το εκτελέσιμο ThesisGameInstaller.exe, το ανοίγουμε με διπλό κλικ και επιλέγουμε τοποθεσία εξαγωγής των αρχείων. Η προεπιλεγμένη είναι στον δίσκο C:\Program Files (x86)\UE4ThesisGame. Έπειτα η εφαρμογή μπορεί να παιχτεί κανονικά ανοίγοντας το ThesisGame.exe από την τοποθεσία ή από το εικονίδιο που δημιουργείται στην επιφάνεια εργασίας με το όνομα Thesis Game.

4.2.3 Περίληψη

Βρισκόμαστε στο μέλλον, όπου τα διαγαλαξιακά ταξίδια πλέον είναι εφικτά και τα μη επανδρωμένα σκάφη με αυτοματοποιημένο πλήρωμα κάτι συνηθισμένο. Ο πρωταγωνιστής είναι ένας στρατιώτης του μέλλοντος με υψηλής τεχνολογίας εξάρτηση και είναι μέλος του Διαγαλαξιακού Στρατού. Τυχαίνει να τον ορίσουν υπεύθυνο σε μια αποστολή διάσωσης του στρατηγού της ομάδας με κωδικό όνομα Κόκκινος Αετός. Η επιχείρηση λαμβάνει χώρα σε ένα σκάφος ελεγχόμενο από μάχιμα ανδροειδή στο οποίο καταφέρνει να προσγειωθεί με ένα μικρό σκάφος που εκπέμπει ψεύτικα σήματα αναγνώρισης. Όλα ξεκινούν από τον θάλαμο συντήρησης του εχθρικού σκάφους.

4.2.4 Σκοπός

Σκοπός του βιντεοπαιχνιδιού είναι ο παίκτης να φέρει εις πέρας την αποστολή διάσωσης και να ελευθερώσει τον στρατηγό από τα ανδροειδή. Ο παίκτης μπορεί να χρησιμοποιήσει οποιοδήποτε μέσο κρίνει απαραίτητο ώστε να επιτύχει τον σκοπό του καθώς καλείται να εκπληρώσει τα βήματα της αποστολής.

4.2.5 Χειρισμός (Controls)

Πίνακας 4-1: Χειρισμός χαρακτήρα

Πλήκτρο	Ενέργεια
W	Βάδισμα εμπρός
S	Βάδισμα πίσω
A	Βάδισμα αριστερά
D	Βάδισμα δεξιά
Shift + W|A|S|D	Τρέξιμο/γρήγορο βάδισμα προς κάθε κατεύθυνση
Mouse Left click	Επίθεση/ Πυροβολισμός

Mouse Right click	Σκόπευση
Mouse Wheel up	Αλλαγή όπλου στο επόμενο
Mouse Wheel down	Αλλαγή όπλου στο προηγούμενο
1	Αλλαγή όπλου στο πρώτο
2	Αλλαγή όπλου στο δεύτερο
Space	Άλμα
E	Συλλογή αντικειμένου από το χώρο/ενεργοποίηση
R	Γέμισμα όπλου
Tab	Άνοιγμα/κλείσιμο μενού παύσης
Esc	Κλείσιμο μενού παύσης

Στον παραπάνω πίνακα φαίνονται οι ενέργειες που εκτελεί ο παίκτης εντός του βιντεοπαιχνιδιού αν πατήσει τα κατάλληλα πλήκτρα σε πληκτρολόγιο. Ο χειρισμός της κάμερας γίνεται με την κύλιση του ποντικιού προς την αντίστοιχη κατεύθυνση. Τα πλήκτρα δεν είναι επαναχαρτογραφήσιμα στο τρέχον έργο.

4.3 Αναφορά πακέτων τρίτων κατασκευαστών

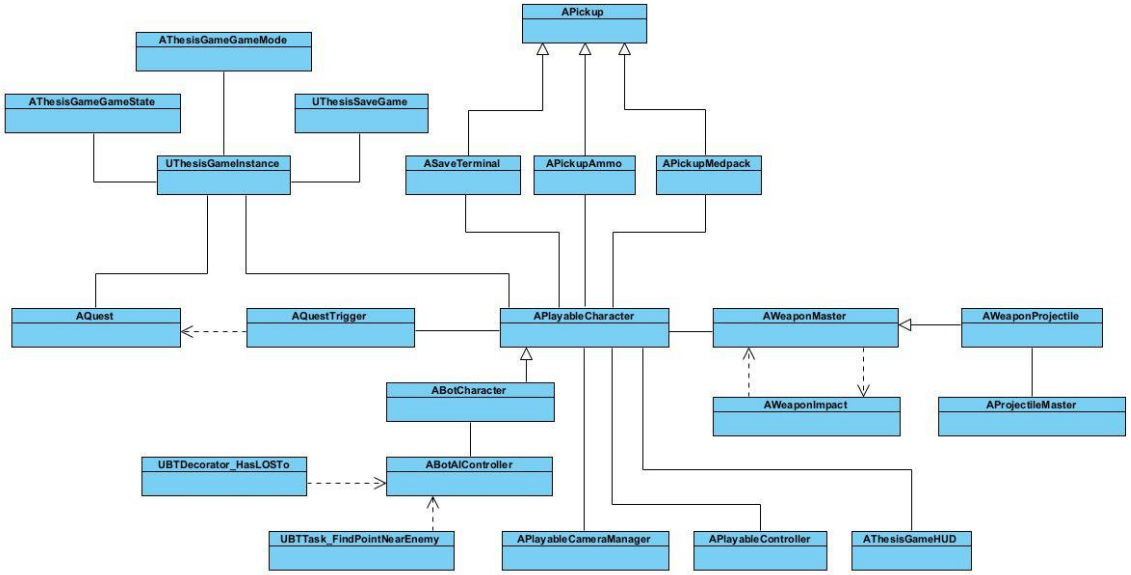
Πίνακας 4-2: Πακέτα γραφικών και κώδικα του Unreal Marketplace

Όνομα	Περιγραφή	Κατασκευαστής
Infiltrator Demo	Sci-Fi κινηματογραφικό πακέτο με χαρακτήρες, κινήσεις και εσωτερικούς χώρους στο διάστημα	Epic Games
Shooter Game	Πακέτο με ένα πρότυπο παιχνιδιού βολών πρώτου προσώπου	Epic Games
Modular Sci Fi Office	Πακέτο με εσωτερικούς χώρους γραφείων και διακοσμητικά	Kelheor
Modular SciFi Season 1 Starter Bundle	Πακέτο με εσωτερικούς χώρους διαστημόπλοιου	Jonathon Frederic
Modular SciFi Season 2 Starter Bundle	Πακέτο με εσωτερικούς χώρους διαστημόπλοιου	Jonathon Frederic
Spaceship Interior Environment Set	Πακέτο με εσωτερικούς χώρους διαστημόπλοιου	Denys Rutkovskyi

Στον παραπάνω πίνακα αναφέρονται τα ονόματα των πακέτων που χρησιμοποιήθηκαν κατά την ανάπτυξη αυτού του έργου.

4.4 Διάγραμμα κλάσεων

Παρακάτω παρατίθεται διάγραμμα με τις κλάσεις που αναπτύχθηκαν και χρησιμοποιήθηκαν σε αυτό το έργο και οι μεταξύ τους σχέσεις. Στο διάγραμμα δεν συμπεριέχονται οι προκαθορισμένες κλάσεις της μηχανής.



Εικόνα 4.2: Διάγραμμα κλάσεων βιντεοπαιχνιδιού

Στην κορυφή της ιεραρχίας δημιουργίας και εκτέλεσης των κλάσεων βρίσκεται το UThesisGameInstance καθώς δημιουργείται με το άνοιγμα του εκτελέσιμου, μαζί με τις υπόλοιπες προκαθορισμένες κλάσεις του Unreal.

Το Instance δημιουργεί με την σειρά του ένα AThesisGameGameMode, δηλαδή έναν τύπο παιχνιδιού, όποιο είχε οριστεί στον Unreal Editor για το χάρτη εκκίνησης. Έπειτα ένα AThesisGameGameState, δηλαδή μια κατάσταση παιχνιδιού για να αποθηκεύει δεδομένα σχετικά με την τρέχουσα κατάσταση της μηχανής. Δημιουργούνται δείκτες για το AThesisSaveGame, AThesisGameGameMode και AThesisGameGameState στο Instance και έπειτα το GameMode δημιουργεί αποστολές και τον χειριστή του παίκτη, τον APlayableController. Έπειτα τοποθετεί το πιόνι του παίκτη σε προκαθορισμένη θέση εκκίνησης.

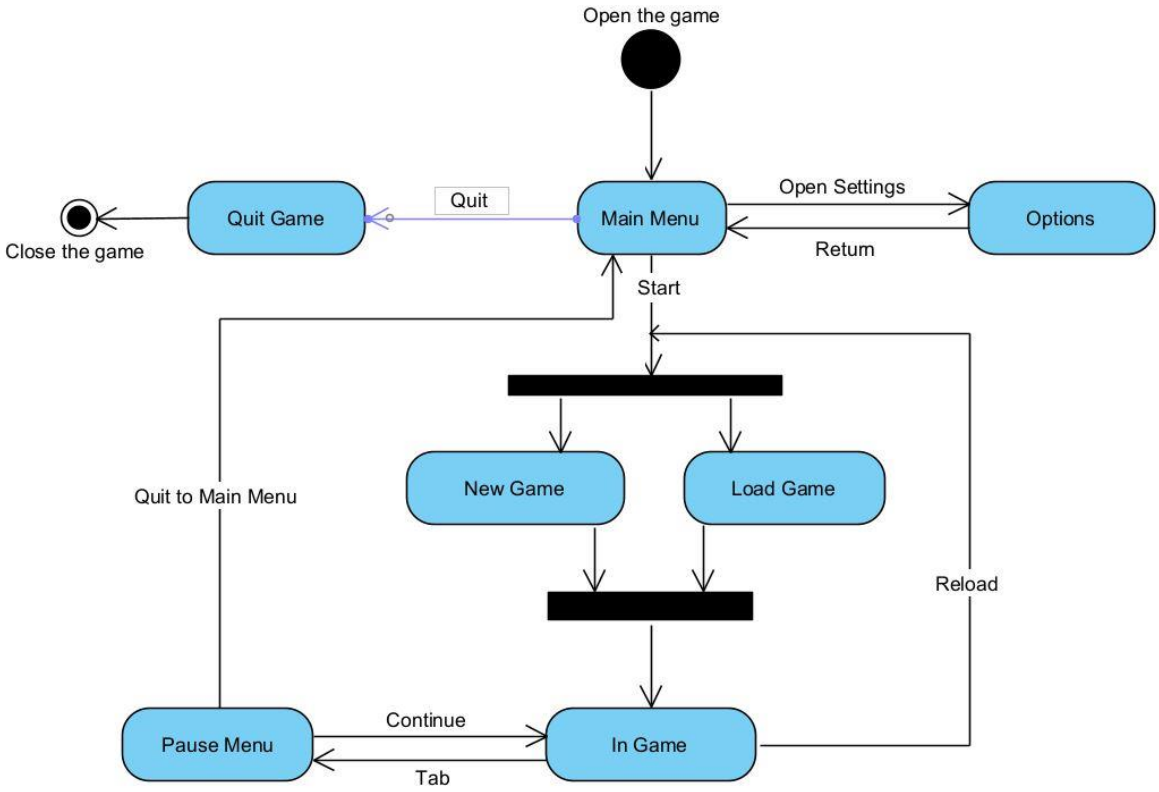
Ο χειριστής του παίκτη δημιουργεί έναν χειριστή κάμερας με την κλάση APlayableCameraManager. Το πιόνι του παίκτη που ανήκει στην κλάση APlayableCharacter δημιουργεί έναν πίνακα με τα προκαθορισμένα όπλα του, από την κλάση AWeaponMaster και παίρνει το πρώτο στη σειρά.

Οι αντίπαλοι ανήκουν στην κλάση ABotCharacter και ομοίως με τον παίκτη, δημιουργούν ο καθένας το δικό του οπλοστάσιο και παίρνουν το πρώτο όπλο της λίστας. Για λόγους απλότητας, ο κάθε αντίπαλος κρατάει ένα μόνο όπλο.

Τα Pickup είναι αντικείμενα που μπορεί να χρησιμοποιήσει ο παίκτης με διάφορους τρόπους. Υπάρχουν τρείς κλάσεις Pickup, μια για κάθε βασική λειτουργία και μπορούν να επεκταθούν με οποιονδήποτε τρόπο μέσω blueprint.

4.5 Διαγράμματα ροής

Παρακάτω παρατίθενται τα διαγράμματα ροής που αφορούν διάφορες διαδικασίες του παιχνιδιού.



Εικόνα 4.3: Διάγραμμα ροής μιας συνεδρίας.

Όπως φαίνεται και στην εικόνα 4.2, το βιντεοπαιχνίδι από την στιγμή που ανοίγουμε το εκτελέσιμο μεταβαίνει στο κεντρικό μενού αφού φορτώσει όλες τις απαραίτητες κλάσεις και καλέσει τα εσωτερικά της μηχανής του Unreal για αρχικοποίηση πεδίων. Από το κεντρικό μενού ο παίκτης έχει την δυνατότητα να αρχίσει ένα νέο παιχνίδι, να φορτώσει την προηγούμενη πρόοδο του, να μεταβεί στις ρυθμίσεις ή να κάνει έξοδο από την εφαρμογή. Αν μεταβεί στις ρυθμίσεις μπορεί να αλλάξει την ποιότητα των γραφικών, το μέγεθος του παραθύρου και την εσωτερική ανάλυση καθώς και την ένταση από τις διάφορες συλλογές ήχων. Έπειτα πατώντας το κουμπί Return μπορεί να περιηγηθεί στο κεντρικό μενού. Αν αρχίσει νέο παιχνίδι ή φορτώσει την προηγούμενη συνεδρία, το παιχνίδι αποφασίζει ποιο σετ συναρτήσεων θα χρησιμοποιήσει για να αρχικοποιήσει τα αντικείμενα και έπειτα μεταβαίνει στην κατάσταση «εντός παιχνιδιού». Όταν είναι σε αυτή τη κατάσταση, ο παίκτης μπορεί να κάνει παύση και έχει την επιλογή να φορτώσει μια προηγούμενη συνεδρία όπως και στο κεντρικό μενού, να συνεχίσει το παιχνίδι πατώντας το κουμπί Continue ή να βγει στο κεντρικό μενού.

4.6 Επίλογος

Σε αυτό το κεφάλαιο παρουσιάστηκε το παιχνίδι ως τελικό προϊόν. Όπως φάνηκε, η ροή του παιχνιδιού είναι μια αλληλουχία από κλήσεις μηχανής, συναρτήσεις και γεγονότα που ενορχηστρώνονται έτσι ώστε το εκτελέσιμο να λειτουργεί πάντα με τον καθορισμένο τρόπο. Η μηχανή γραφικών Unreal Engine καθιστά αυτού του είδους την διαδικασία σχετικά εύκολη καθώς δίνει πρόσβαση σε αυτοματοποιημένες

διαδικασίες και την βασική δομή πάνω στην οποία μπορούν να στηριχτούν οι προγραμματιστές για να δημιουργήσουν και να διανέμουν περιεχόμενο.

Κεφάλαιο 5ο: Συμπεράσματα και προτάσεις βελτίωσης

5.1 Συμπεράσματα

Τα βιντεοπαιχνίδια αποτελούν αναπόσπαστο κομμάτι της εξέλιξης της τεχνολογίας και θεωρούνται ως ένα ακόμη μέσο ψυχαγωγίας της σύγχρονης κοινωνίας. Μπορεί κανείς να διακρίνει την εξέλιξη του υλικού και του λογισμικού απλά παρατηρώντας, ανάμεσα σε άλλες εφαρμογές, βιντεοπαιχνίδια από διαφορετικές δεκαετίες.

Με την διαρκή ανάπτυξη λογισμικού απαιτούνταν όλο και πιο γρήγοροι ρυθμοί παραγωγής, οπότε ακόμα και στα βιντεοπαιχνίδια ενσωματώθηκαν διαδικασίες και πρότυπα οργάνωσης, ανάπτυξης κώδικα και διαχωρισμού καθηκόντων που μεγέθυναν σημαντικά τις απαιτήσεις σε ανθρώπινο δυναμικό, ειδικότητες και βοηθητικά προγράμματα. Όμως αυτό δεν εμποδίζει μικρότερες ομάδες και μεμονωμένα άτομα από το να δημιουργήσουν και να διανέμουν έργα πληροφορικής με χαρακτήρα βιντεοπαιχνιδιού.

Το Unreal Engine είναι μια μηχανή προσαρμοσμένη στις πολύπλοκες ανάγκες μιας εταιρείας παραγωγής βιντεοπαιχνιδιών και διαθέτει όλα τα σύγχρονα εργαλεία τα οποία θα χρειαστεί κανείς για να δημιουργήσει ένα βιντεοπαιχνίδι. Έχει ποικίλες εφαρμογές και πλήρως πολυμορφικό χαρακτήρα. Οτιδήποτε είναι ψηφιακό, μπορεί να δημιουργηθεί ή να τεθεί υπό επεξεργασία με αυτήν τη μηχανή. Απευθύνεται σε προγραμματιστές με αρκετή εμπειρία στην γλώσσα C++ αλλά το σύστημα προγραμματισμού με Blueprints την κάνει ελκυστική ακόμα και σε άτομα που δεν διαθέτουν τις κατάλληλες βάσεις για να προγραμματίσουν σε πιο χαμηλό επίπεδο.

Με τα ανωτέρω ως δεδομένα, η κατασκευή ενός παιχνιδιού μικρού μήκους έγινε πραγματικότητα χρησιμοποιώντας υλικό προς ελεύθερη (εντός των πλαισίων της μηχανής) χρήση, βιντεομαθήματα και το εγχειρίδιο χρήσης της μηχανής. Στην πορεία αποδείχθηκε πως το έργο σε C++ ήταν αρκετά δύσκολο διότι κάποια λογικά λάθη κατέληγαν σε σφάλμα του λειτουργικού συστήματος, γεγονός που κατέστησε την ανάπτυξη κώδικα χωρίς επαρκείς γνώσεις αρκετά επικίνδυνη.

5.2 Προτάσεις βελτίωσης

Παρόλο που το έργο στην παρούσα στιγμή θεωρείται ολοκληρωμένο, υπάρχουν αρκετά σημεία που θα μπορούσε να γίνει κάποια βελτίωση και γίνονται αισθητά κατά το τρέξιμο του παιχνιδιού.

Στην αρχική σκηνή ο χαρακτήρας βλέπει τον θάλαμο προσγείωσης αλλά οι αντίπαλοι δεν τον αντιλαμβάνονται ενώ είναι σε κατάλληλη απόσταση. Θα μπορούσαν να είναι ενεργοί αντίπαλοι με νοημοσύνη και αν τον αντιληφθούν να αποτυγχάνει η αποστολή.

Δεν υπάρχει ένδειξη για την ζωή των εχθρών και υποβιβάζει την εμπειρία του παίκτη καθώς δεν γνωρίζει πόση ζημιά έχει κάνει στους εχθρούς και είναι σε μια μόνιμη αβεβαιότητα κατά την διάρκεια της μάχης.

Τα ηχητικά εφέ θα μπορούσαν να είναι λίγο καλύτερα στους εσωτερικούς χώρους.

Όταν ο παίκτης πεθάνει δεν εμφανίζεται κάποιο μενού που να του ζητάει να κάνει επαναφόρτωση από την τελευταία συνεδρία, αντιθέτως τον επαναφέρει αυτόματα στο τελευταίο γνωστό σημείο. Θα μπορούσε να βγαίνει κάποιο μήνυμα ότι έχασε και να του ζητάει να ξαναπροσπαθήσει ή να βγει στο αρχικό μενού.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Jason Rutter and Jo Bryce, Understanding Digital Games, London: SAGE Publications Ltd, 2006

[2] Julian Alvarez, Jean-Pierre Jessel, Gilles Methel and Pierre Molinier, "Cyber Games and Interactive Entertainment 2006", Research Article, Volume 2008, Article ID 470350. [Online]. Available: https://doi.org/10.1155/2008/470350

[3] Esposito Nicolas, "A Short and Simple Definition of What a Videogame Is", member of collection DiGRA 2005: Changing Views: Worlds in Play, 2005 International Conference, 16 Απριλίου 2005

[4] Geoff King and Tanya Krzywinska, Computer Games / Cinema / Interfaces, UK: Brunel University, pp. 141-152

[5] Grant Tavinor, The Art of Videogames, Chichester, Wiley-Blackwell, 2009

[6] Gerald A. Voorhees, Josh Call and Katie Whitlock, Guns, Grenades and Grunts: First-Person Shooter Games, New York: A Bloomsbury Company, 2012, pp. 63-79

[7] René Weber, Katharina-Maria Behr, Ron Tamborini, Ute Ritterfeld and Klaus Mathiak, "What Do We Really Know about First-Person-Shooter Games? an Event-Related, High-Resolution Content Analysis", Journal of Computer-Mediated Communication, Vol. 14, Issue 4, pp. 1016–1037, 1 July 2009, [Online]. Available: https://doi.org/10.1111/j.1083-6101.2009.01479.x

[8] Vic Hood, Jordan Oloman and Malindy Hetfeld, "Best FPS games: essential first-person shooters", New York 130 W. 42nd St.: techradar, 12 Jan. 2023. [Online]. Available: https://www.techradar.com/news/best-fps-games?msclkid=cc837de7bee911ec9dc53eebdd5f4d21

[9] Ralph Edwards, "The Game Production Pipeline: Concept to Completion, What goes into making a game?", Los Angeles: IGN, www.ign.com, 16 Mar. 2006. [Online]. Available: https://www.ign.com/articles/2006/03/16/the-game-production-pipeline-concept-to-completion

[10] Dustin Tyler, "Video Game Mechanics for Beginners", St. Petersburg: GAMEDESIGN - GameDesigning.org, updated on 23 Feb. 2023. [Online]. Available: https://www.gamedesigning.org/learn/basic-game-mechanics/?msclkid=e7148bdfbf1111eca5335055c3916405

[11] Marc Schmalz, Aimee Finn and Hazel Taylor, "Risk Management in Video Game Development Projects", published in 2014 47th Hawaii International Conference on System Sciences, 6-9 Jan 2014, ISSN 1530-1605. Section 2. Available: https://doi.org/10.1109/HICSS.2014.534

[12] Grethe Mitchell and Andy Clarke, "VIDEOGAME ART: REMIXING, REWORKING AND OTHER INTERVENTIONS", Proceedings of the 2003 DiGRA International Conference: Level Up, Volume 2, ISSN 2342-9666, 2003

[13] Lucas Bertolini, Hands-On Game Development without Coding, UK: Packt Publishing Ltd, Nov. 2018, pp. 4-30.

[14] Sean Gibson, "10 Things That Make an FPS Awesome", GamersDecide, 13 Jul 2015. [Online]. Available: https://gamersdecide.com/pc-game-news/10-things-make-fps-awesome?msclkid=cc839694bee911eca80fca730ef9f71f

[15] ScreenSkills Recruits, “Job profiles – Games”, Ibex House, 42-47, London, ScreenSkills, EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/

[16] ScreenSkills Recruits, “Games publisher”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/production/games-publisher/

[17] ScreenSkills Recruits, “Games producer”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/production/games-producer-games/

[18] Dustin Tyler, “What Does a Lead Video Game Designer Do?”, St. Petersburg: GAMEDESIGN – GameDesigning.org. [Online]. Available: https://www.gamedesigning.org/career/lead-game-designer/

[19] ScreenSkills Recruits, “Lead games designer”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/design/lead-games-designer/

[20] ScreenSkills Recruits, “Gameplay designer”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/design/gameplay-designer/

[21] ScreenSkills Recruits, “Writer (Games)”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/design/writer-games/

[22] Indeed Editorial Team, “How to Become a Video Game Writer”, indeed, updated 27 Jan 2023. [Online]. Available: https://www.indeed.com/career-advice/career-development/how-to-become-a-video-game-writer

[23] Pawel Ledwon, “What’s the role of a tech lead?”, hackernoon, 25 Jan 2018. [Online]. Available: https://hackernoon.com/whats-the-role-of-a-tech-lead-7725b47104b7

[24] Indeed Editorial Team, “7 Key Roles in Video Game Development”, updated 11 Mar. 2023. [Online]. Available: https://www.indeed.com/career-advice/finding-a-job/game-development-roles

[25] ScreenSkills Recruits, “Level Designer”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/design/level-designer/

[26] ScreenSkills Recruits, “User experience (UX) designer”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/design/user-experience-ux-designer/

[27] Oliver Franzke, “How to become a Graphics Programmer in the games industry”, Game Developer – www.gamedeveloper.com, 18 Jul 2014. [Online]. Available: https://www.gamedeveloper.com/programming/how-to-become-a-graphics-programmer-in-the-games-industry

[28] ScreenSkills Recruits, “Audio programmer”, Ibex House, 42-47, London, ScreenSkills EC3N 1DY. [Online]. Available: https://www.screenskills.com/job-profiles/browse/games/audio/audio-programmer/

[29] Tim Sweeney, “If You Love Something, Set It Free”, Epic Games Sarl - Unreal Engine Community, 2 Mar. 2015. [Online]. Available: https://www.unrealengine.com/en-US/blog/ue4-is-free

[30] Unreal Engine, “Frequently Asked Questions”, Epic Games. [Online]. Available: https://www.unrealengine.com/en-US/faq

[31] Unreal Engine, “Unreal Engine End User License Agreement”, Epic Games. [Online]. Available: https://www.unrealengine.com/en-US/eula/unreal

[32] Unreal Engine, “Marketplace”, Epic Games. [Online]. Available: https://www.unrealengine.com/marketplace/en-US/store

[33] Unreal Engine, “Working with Content”, Unreal Engine Documentation version 5.0. [Online]. Available: https://docs.unrealengine.com/5.0/en-US/working-with-content-in-unreal-engine/

[34] Unreal Engine Documentation, “Introduction to C++ Programming in UE4”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/IntroductionToCPP/

[35] Unreal Engine Documentation, “Handware and Software Specifications”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Basics/InstallingUnrealEngine/RecommendedSpecifications/

[36] Unreal Engine Documentation, “UnrealVS Extension”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/ProductionPipelines/DevelopmentSetup/VisualStudioSetup/UnrealVS/

[37] Unreal Engine Documentation, “Unreal Editor Interface”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Basics/UI/

[38] Unreal Engine Documentation, “Project Settings”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Basics/Projects/ProjectSettings/

[39] Unreal Engine Documentation, “Components”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Basics/Components/

[40] Unreal Engine Documentation, “Review Collision in Your Game”, Unreal Engine Documentation version 5.1, Epic Games. [Online]. Available: https://docs.unrealengine.com/5.1/en-US/review-collision-in-your-unreal-engine-game/

[41] Unreal Engine Documentation, “Review Collision in Your Game”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/Physics/Collision/Overview/

[42] Unreal Engine Documentation, “Actors”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/UnrealArchitecture/Actors/

[43] Unreal Engine Documentation, “Actor Mobility”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Basics/Actors/Mobility/

[44] Unreal Engine Documentation, “Static Lights”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/LightingAndShadows/LightMobility/StaticLights/

[45] Unreal Engine Documentation, “Stationary Lights”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/LightingAndShadows/LightMobility/StationaryLights/

[46] Unreal Engine Documentation, “Movable Lights”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/LightingAndShadows/LightMobility/DynamicLights/

[47] Unreal Engine Documentation, “Trigger Actors”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Basics/Actors/Triggers/

[48] Harrison McGuire, “Trigger Box”, Unreal Engine Tutorials - UnrealCPP.com, 01 Dec. 2017. [Online]. Available: https://unrealcpp.com/trigger-box/

[49] Unreal Engine Documentation, “Decals”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Resources/ContentExamples/Decals/1_1/

[50] Unreal Engine Documentation, “Implementing your Character”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/CPPTutorials/FirstPersonShooter/2/

[51] Unreal Engine Documentation, “Components and Colllision”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/CPPTutorials/Components/

[52] Unreal Engine Documentation, “UMG UI Designer Quick Start Quide”, Unreal Engine Documentation version 4.26, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.26/en-US/InteractiveExperiences/UMG/QuickStart/

[53] Alex Galuzin, “UE4: 16 Principles – How to Start Learning Unreal Engine 4”, World of Level Design – www.worldofleveldesign.com, 23 Sept. 2015. [Online]. Available: https://www.worldofleveldesign.com/categories/ue4/ue4-how-to-learn-unreal-engine4.php

[54] Unreal Engine Documentation, “Navmesh Map”, Unreal Engine Documentation version 4.26, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/Resources/ContentExamples/NavMesh/

[55] Orfeas Eleftheriou, “Implementing AI Hearing using C++”, *AI Programming, Unreal Engine 4 C++ Tutorials – www.orfsel.com*, 26 Mar. 2016. [Online]. Available: https://www.orfeasel.com/implement-ai-hearing-using-c/

[56] Unreal Engine Documentation, “Artificial Intelligence”, Unreal Engine Documentation version 4.26, Epic Games. [Online]. Available: https://docs.unrealengine.com/5.1/en-US/artificial-intelligence-in-unreal-engine/

[57] Unreal Engine Documentation, “Behavior Trees”, Unreal Engine Documentation version 4.26, Epic Games. [Online]. Available: https://docs.unrealengine.com/5.1/en-US/behavior-trees-in-unreal-engine/

[58] Nerivec, “Quest Framework”, old UE4 wiki at github.com, 9 Apr. 2020. [Online]. Available: https://github.com/Nerivec/old-ue4-wiki/blob/master/pages/duplicates/quest-framework.html

[59] Unreal Engine Documentation, “Saving and Loading Your Game”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/SaveGame/

[60] SolidSk, “Save Game causing crash”, Unreal Engine Forums, 20 Jul 2023. [Online]. Available: https://forums.unrealengine.com/t/save-game-causing-crash/147569/9

[61] Unreal Engine Documentation, “Game Mode and Game State”, Unreal Engine Documentation version 4.27, Epic Games. [Online]. Available: https://docs.unrealengine.com/4.27/en-US/InteractiveExperiences/Framework/GameMode/

[62] Yun-Kun, “What should I do in GameMode, GameState, and PlayerState”, Unreal Engine Forums, 17 Jun 2023. [Online]. Available: https://forums.unrealengine.com/t/what-should-i-do-in-gamemode-gamestate-and-playerstate/93584/6

[63] iorek01, “Simple Pause Menu”, Dev Community – forums.unrealengine.com, Jan 16 2023. [Online]. Available: https://forums.unrealengine.com/t/simple-pause-menu/53111

[64] Unreal Engine Documentation, “Visual Studio Tips and Tricks”, Unreal Engine Documentation version 5.0, Epic Games. [Online]. Available:https://docs.unrealengine.com/5.0/en-US/visual-studio-tips-and-tricks-in-unreal-engine/

[65] Chris McCole, “Logging in UE4 CPP (C++)”, Tutorial, 22 June 2022. [Online]. Available: https://www.chrismccole.com/blog/logging-in-ue4-cpp

[66] Unreal Engine Documentation, “Packaging Projects”, Unreal Engine Documentation version 4.26, Epic Games. [Online]. Available:https://docs.unrealengine.com/4.26/en-US/Basics/Projects/Packaging/

ΠΑΡΑΡΤΗΜΑ Α : ΣΥΝΔΕΣΜΟΙ

Παρατίθεται ο δημόσιος σύνδεσμος στο Google Drive για το παιχνίδι:

https://drive.google.com/file/d/1Djv2vSV3fXTf4KWhPDWZGn7bXzWut0gw/view?usp=sharing

Ο πηγαίος κώδικας στο GitHub: https://github.com/Lordkarnak/UE4_ThesisGame

Το πακέτο γραφικών Infiltrator: https://www.unrealengine.com/marketplace/en-US/product/infiltrator-demo

Το πακέτο γραφικών Shooter Game: https://www.unrealengine.com/marketplace/en-US/product/shooter-game

Το πακέτο γραφικών Modular Scifi Season 1 Starter Bundle: https://www.unrealengine.com/marketplace/en-US/product/modular-scifi-season-1-starter-bundle

Το πακέτο γραφικών Modular Scifi Season 2 Starter Bundle: https://www.unrealengine.com/marketplace/en-US/product/modular-scifi-season-2-starter-bundle

Το πακέτο γραφικών Modular Sci fi Office: https://www.unrealengine.com/marketplace/en-US/product/modular-sci-fi-office

Το πακέτο γραφικών Spaceship Interior Environment Set: https://www.unrealengine.com/marketplace/en-US/product/spaceship-interior-environment-set