ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΠΜΣ "Προηγμένα Συστήματα Πληροφορικής – Ανάπτυξη Λογισμικού και Τεχνητής Νοημοσύνης"

Τελική Εργασία:

Ευφυή Εικονικά Περιβάλλοντα

Τεχνητή Νοημοσύνη

Ανάπτυξη Παιχνιδιών και Εφαρμογών Εικονικής Πραγματικότητας

«The Chase Master»

Κώστας Σπυρόπουλος - Γιώργος Πάνου – Αγάπη Δαβράδου

Καθηγητής: Παναγιωτόπουλος Θεμιστοκλής

Πειραιάς 2020

ΠΕΡΙΕΧΟΜΕΝΑ

Εισαγωγή

Η παρούσα εργασία αφορά την ανάπτυξη ηλεκτρονικού παιχνιδιού κυνηγητού με την βοήθεια της μηχανής Unity. Η Unity είναι ένα Game Engine (Μηχανή Παιχνιδιών) που παρέχει δυνατότητες απεικόνισης 3D γραφικών αλλά και προσομοίωσης φυσικής.

Το παιχνίδι είναι ένα κλασσικό κυνηγητό με στόχο την αποφυγή του αντιπάλου και την ολοκλήρωση της πίστας. Το gameplay αφορά την συλλογή αντικειμένων που συμβάλουν στο ξεκλείδωμα χώρων για τον τερματισμό. Αποτελείται από τρείς πίστες αυξανόμενης δυσκολίας. Στον χαρακτήρα του παιχνιδιού έχουν δωθεί ικανότητες parkour, μπορεί να σκαρφαλώνει σε κτήρια και εμπόδια και να τα χρησιμοποιεί ώστε να αποφύγει τον αντίπαλο, και spawning εμποδίων ώστε να τον δυσκολέψει να τον πιάσει.

Για την δημιουργία του παιχνιδιού αναπτύχθηκαν scripts για την λογική του παιχνιδιού (gameplay) αλλά και για τον έλεγχο της κάμερας και του χαρακτήρα και της μουσικής και τέλος τροποποιήθηκε ο animation controller ώστε να καλύπτει τις ανάγκες του παιχνιδιού. Οι πίστες σχεδιάστηκαν και δημιουργήθηκαν από το μηδέν και είναι σύνθεσεις από building blocks τόσο του game engine όσο και από τρίτους δημιουργούς που διατίθενται online.

Για την ανάπτυξη χρησιμοποιήθηκαν πράκτορες τεχνητής νοημοσύνης Nav Mesh agents, 3rd party animations, 3D μοντέλα κτηρίων και χαρακτήρων και δημιουργήθηκε πρωτότυπη μουσική τόσο για τα μενού όσο και για την πίστα. Για την δημιουγία της μουσικής χρησιμοποιήθηκαν τα BandLab Cakewalk για την δημιουργία της μουσικής και το Audacity για την επεξεργασία. Για την ανάπτυξη των scripts χρησιμοποιήθηκε η γλώσσα C# και το Visual Studio ώς code editor.

1 Game Engines

Μια Game Engine παρέχει βασικές λειτουργίες που απαιτεί η ανάπτυξη ενός παιχνιδιού όπως: την απεικόνιση γραφικών (renderer), την παραγωγή ήχου, μηχανή φυσικής/φυσικών νόμων (physics), animations, διαχείριση πόρων του υπολογιστή του χρήστη από το πρόγραμμα και την παραγωγή (build) του εκτελέσιμου του παιχνιδιού για διαφορετικά λειτουργικά συστήματα.

1.1 Ιστορική αναδρομή

Οι μηχανές παιχνιδιών έκαναν την εμφάνισή τους στις αρχές της δεκαετίας του '90 ταυτόχορνα με την εμφάνιση των 3d γραφικών, τα οποία άλλαξαν την βιομηχανία των ηλεκτρονικών παιχνιδιών. Στα μέσα της σημαντικής αυτής δεκαετίας καθιερώθηκε ο όρος game engine ,με τα παιχνίδια "Doom" και "Quake" να κάνουν την επανάσταση στο video game development. Μετά την επιτυχία των δύο αυτών παιχνιδιών διάφοροι developers άρχισαν και αγόραζαν τα βασικά κομμάτια που τα στοιχειοθετούσαν και άρχισαν να προσθέτουν δικά τους κομμάτια κώδικα ώστε να τα εξελίξουν ή να τα τροποποιήσουν ανάλογα με τις απαιτήσεις. Έτσι σταδιακά αυτές οι συλλογές διαμορφόνονταν σε Game Engines.

Η πιο σημαντική εμπορικά διαθέσιμη στο λιανικό εμπόριο μηχανή παιχνιδιών ήταν η **Unreal Engine** που κυκλοφόρησε το 1998. Είχαν κυκλοφορήσει και στο παρελθόν και παλιότερα πρώιμες τέτοιες μηχανές. Μερικές από τις πιο γνωστές ήταν:

- Pinball Construction Set (1983)
- ASCII's War Game Construction Kit (1983)
- Adventure Construction Set (1984)
- Shooter-Up Construction Kit (1987)

2 Unity3D

Πέρα από Game Engine το Unity3D είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού (IDE) για την δημιουργία 3D και 2D βιντεοπαιχνιδιών ή άλλου περιεχομένου όπως αρχιτεκτονικές μοντελοποίησης ή 3D animations πραγματικού χρόνου. Είναι της φιλοσοφίας ότι ένα ολοκληρωμένο γραφικό περιβάλλον πρέπει να είναι το κύριο μέσο ανάπτυξης των βιντεοπαιχνιδιών και ελαχιστοποιεί τη χρήση προγραμματισμού μόνο στη συμπεριφορά των αντικειμένων του κόσμου του παιχνιδιού.

Η πλατφόρμα Unity προσφέρει ολοκληρωμένο περιβάλλον ανάπτυξης με ιεραρχικό και οπτικό editing και live preview του υπό ανάπτυξης παιχνιδιού. Τα αρχεία που φορτώνονται στο Unity (π.χ. υφές, μοντέλα) εισάγονται αυτόματα στο παιχνίδι και μάλιστα επανεισάγοντα αν χρειαστεί. Παρέχει την δυνατότητα συνθέσεις στοιχείων του παιχνιδιού να εξαχθούν ως templates (prefabs) ώστε να επαναχρησιμοποιηθούν με ευκολία στο μέλλον.

2.1.1 Πλεονεκτήματα

Το Unity 3D το χαρακτηρίζει η ιδαίτερη φιλικότητα προς το χρήστη, ενώ είναι αρκετά εύκολο στην εκμάθηση για κάποιον αρχάριο και παράλληλα προσφέρει πολλές δυνατότητες σε κάποιον έμπειρο χρήστη. Πιο αναλυτικά τα πλεονεκτήματα του είναι:

- Δημιουργία για πολλές πλατφόρμες: Ένα από τα κύρια πλεονεκτήματα του Unity σε σχέση με άλλα εργαλεία ανάπτυξης παιχνιδιών είναι η δυνατότητα που προσφέρει στον προγραμματιστή να αναπτύξει ένα παιχνίδι για οποιαδήποτε πλατφόρμα επιθυμεί. Ενδεικτικά επιτρέπει την ανάπτυξη κώδικα για:
    - Windows,
    - Android,
    - Linux,
    - Mac OS

  - Xbox
  - PS4
  - WebGL
  - Facebook

- Ύπαρξη μεγάλης και αναπτυσσόμενης κοινότητας χρηστών: Υπάρχουν πάνω απο 2 εκατομμύρια developers οι οποίοι χρησιμοποιούν το λογισμικό του Unity. Αυτό έχει ως αποτέλεσμα την ύπαρξη μιας μεγάλης ενεργής κοινότητας, η οποία προσφέρει σημαντική βοήθεια σε χρήστες όλων των επιπέδων. Εδώ αξίζει να αναφέρουμε πως μια ιδιαίτερα σημαντική πηγή βοήθειας, ειδικά για τους νέους χρήστες, είναι μια πληθώρα απο tutorials, τα οποία υπάρχουν, είτε από την επίσημη ομάδα του Unity στο http://unity3d.com/learn είτε απο απλούς χρήστες του λογισμικού.

- Εύκολη ενσωμάτωση scripts: Το Unity προσφέρει τη δυνατότητα ενσωμάτωσης scripts γραμμένα σε C# ή Javascript τα οποία μπορούν εύκολα να αντιστοιχηθούν με συγκεκριμένα αντικείμενα δίνοντάς τους ξεχωριστές ιδιότητες.

- Asset Store: Το Asset Store του Unity περιέχει packages, τα οποία μπορεί κάποιος να κάνει include στο project του. Τα πακέτα αυτά περιέχουν κάποια έτοιμα αντικείμενα, όπως κτίρια, χαρακτήρες, τα οποία ο developer του εκάστοτε παιχνιδιού μπορεί να επεξεργαστεί χωρίς να χρειάζεται να τα δημιουργήσει απο το μηδέν.

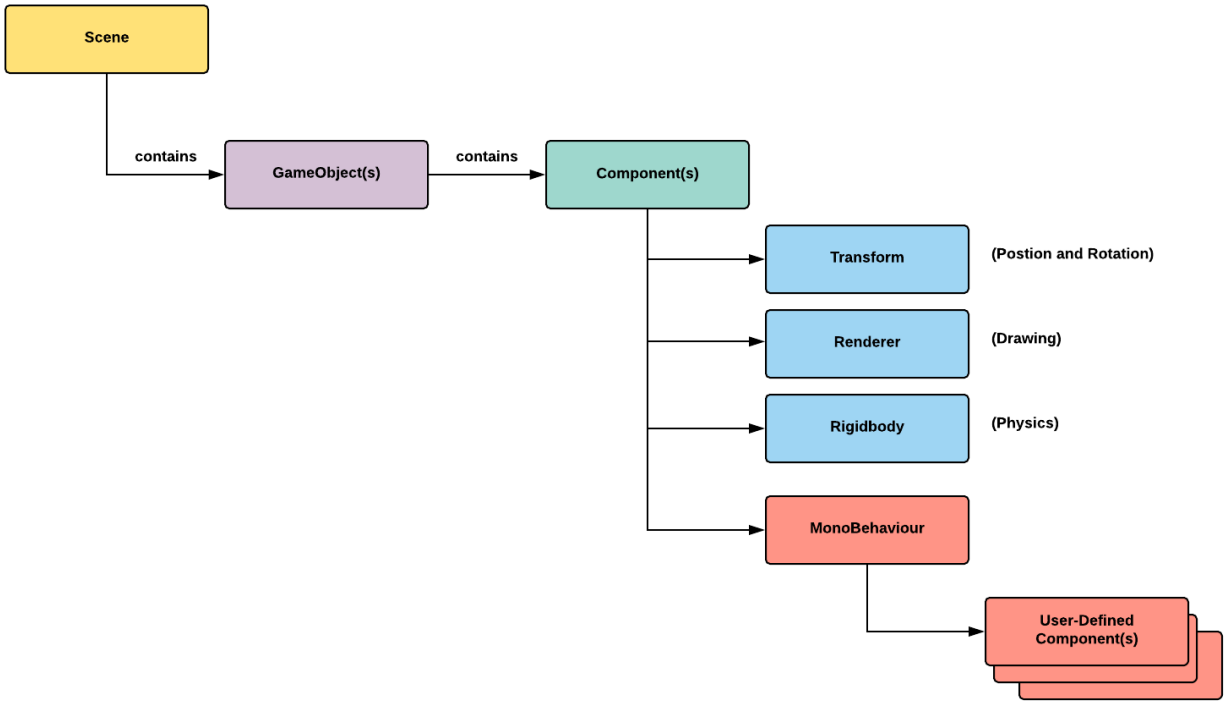
- Είναι δωρεάν: Στο Unity 3D υπάρχει μια Free version και μια Pro version. Φυσικά η Pro version περιέχει κάποια extra χαρακτηριστικά. Όμως, σε αντίθεση με άλλα πακέτα λογισμικού, η Free version προσφέρει ένα πλήρως λειτουργικό εργαλείο για την ανάπτυξη ενός εμπορικού παιχνιδιού.

2.1.2 Αγορά

Τα τελέυταία χρόνια η βιομηχανία των παιχνιδιών έχει αποκτήσει μια διαφορετική δομή σε σχέση με το παρελθόν και αυτό συμβαίνει από την δωρεάν διάθεση που υπάρχει σε Game Engines όπως η **Unity3D**. Πλατφόρμες σαν την Unity δίνουν την δυνατότητα σε μικρές ομάδες από προγραμματιστές και καλλιτέχνες να εκφραστούν και να δημιουργήσουν περιβάλλοντα εικονικής πραγματικότητας με σχεδόν μηδενικό budget. Η αγορά ηλεκτρονικών παιχνιδιών, τα τελευταία χρόνια, έχει αποκτήσει έναν άνευ προηγουμένου πλουραλισμό στους προσφερόμενους τίτλους και το Unity είναι ο κυρίαρχος λόγος πίσω από αυτό. Τα παιχνίδια αυτά – δημιουργημένα από μικρές ομάδες – χαρακτηρίζονται πλέον με το δικό τους genre όνομα: Indie. Ο όρος προκύπτει από το independent και θέλει να υποδείξει ότι είναι μια παραγωγή ανεξάρτητη από μεγάλυος διανομείς τίτλων.

Αυτό δεν σημαίνει ότι η Unity3D έχει κάτι να ζηλέψει από εμπορικές είτε από proprietary (in-house) μηχανές γραφικών. Πληθόρα τίτλων με αστρονομικά budget χρησιμοποιούν την ίδια μηχανή γραφικών που χρησιμοποιήσαμε και εμείς ως βάση για την ανάπτυξη των παιχνιδιών τους.

2.2 Unity Engine Architecture



Εικόνα 2.1: Αρχιτεκτονική Unity

Η αρχιτεκτονική του κώδικα όταν αναπτύσσεται μια εφαρμογή δεν μπορεί να διαφέρει από αυτή που έχει οριστεί από τους δημιουργούς του Unity Engine. Το μητρικό στοιχείο είναι μια σκηνή, έπειτα η σκηνή αποτελείται από μια αφηρημένη (abstract) κλάση η οποία λέγεται game object και περιλαμβάνει όλα τα αντικείμενα που συνθέτουν το παιχνίδι. Αυτά ονομάζονται components και ενδεικτικά τα πιο σημαντικά είναι τα :

- Transform, αποθηκεύει και διαχειρίζεται τις συντεταγμένες του αντικειμένου στον κόσμο
- Renderer, αναλαβμάνει να αποτυπώσει τα γραφικά του αντικειμένου, και τις αλληλεπιδράσεις με τον φωτισμό

- Rigidbody, καθορίζει την συμπεριφορά του αντικειμένου με βάση τους νόμους της φυσικής
- MonoBehaviour, επιτρέπει την εφαρμογή προγραμματιστικής λογικής από τον προγραμματιστή μέσω scripts ελέγχου όλων των υπόλοιπων Components.

2.2.1 Scenes

Το πρώτο αντικέιμενο στην ιεραρχία όπως ορίζεται από την αρχιτεκτονική του Unity είναι η σκηνή. Οτιδήποτε τρέχει στο παιχνίδι υπάρχει σε μια σκηνή. Όταν πακετάρεται το παιχνίδι για μια πλατφόρμα, το εκτελέσιμο που προκύπτει είναι μια συλλογή από μία ή περισσότερες σκηνές, που περιλαμβάνουν τον κώδικα που έχει δημιουργηθεί.

Μπορούμε να εισάγουμε όσες σκηνές θέλουμε σε ένα project. Μια σκηνή μπορεί να θεωρηθεί ως ένα επίπεδο σε ένα παιχνίδι, αν και είναι δυνατόν να υπάρχουν πολλαπλά επίπεδα σε μία σκηνή απλά μετακινώντας τον παίκτη / κάμερα σε διαφορετικά σημεία της σκηνής. Ένα αρχείο σκηνής είναι ένα μοναδικό αρχείο που περιέχει όλα τα είδη των αρχείων που χρησιμοποιούνται στο παιχνίδι για την τρέχουσα σκηνή.

Σε μια σκηνή, δεν μπορεούμε να δούμε τίποτα χωρίς ένα αντικείμενο κάμερας και δεν μπορούμε να ακούσουμε τίποτα χωρίς ένα Audio Listener εφαρμοσμένο πάνω σε κάποιο GameObject. Σημειώνεται ωστόσο ότι σε κάθε νέα σκηνή , το Unity δημιουργεί πάντα μια default κάμερα που διαθέτει και Audio Listener πάνω της.

2.2.2 GameObjects

Σχεδόν τα πάντα σε μια σκηνή είναι ένα GameObject. Είναι η βασική κλάση για όλα τα αντικείμενα σε μία σκηνή του Unity. Όλα τα αντικείμενα απορρέουν από ένα GameObject. ‘Ενα GameObject είναι αρκετά απλό, δεδομένου ότι εντάσσεται στο παράθυρο Inspector. Τα GameObjects by default δεν έχουν οπτικές ιδιότητες, εκτός από το widget που δείχνει το Unity όταν επιλεγεί το αντικείμενο.

Ένα GameObject έχει ένα όνομα, μια ετικέτα (Tag), ένα Layer και το Transform (ίσως η πιο σημαντική ιδιότητα του συνόλου). Το Transform είναι στην ουσία η θέση, η περιστροφή και η κλίμακα του κάθε GameObject. Το Unity χρησιμοποιεί ως τις συντεταγμένες το X (οριζόντια), Y (κάθετη) και Z συντεταγμένη (το βάθος, δηλαδή, με φορά σαν μπαίνει ή βγαίνει απο την οθόνη).

2.2.3 Components

Για να γίνει λειτουργικό ένα GameObject, γίνεται μέσω της προσθήκης διάφορων στοιχείων (Components). Οτιδήποτε προστεθεί είναι ένα Component και εμφανίζονται όλα στο παράθυρο του Inspector. Υπάρχουν MeshRender και SpriteRender Components, Components για τον ήχο και την λειτουργικότητα της κάμερας, Components που σχετίζονται με την Φυσική (Colliders και Rigidbodies) και πολλά ακόμα. Για να συνδέσουμε κώδικα με ένα GameObject χρησιμοποιούμε το script Component. Τα Components αυτά είναι του ζωντανεύουν ένα GameObject και του προσδίδουν οποιαδήποτε λειτουργικότητα.

2.2.4 Ήχος

Όσον αφορά τον ήχο, το Unity υποστηρίζει 2D και 3D ήχους. Οι 3D ήχοι αλλάζουν ένταση ανάλογα με την απόσταση, και αλλάζουν ανάλογα με την κίνηση τους σε σχέση με την κάμερα. Οι 2D ήχοι από την άλλη είναι πιό κατάλληλοι για χρήση ως μουσική παρασκηνίου καθώς κρατούν μια ομοιόμορφη ένταση. Για ήχους τώρα που παράγονται από γεγονότα χρησιμοποιούνται οι 3D ήχοι.

2.3 Physics

Το Unity έρχεται με μία ενσωματωμένη μηχανή φυσικής που επιτρέπει να οριστούν οι φυσικές ιδιότητες των αντικειμένων και αφήνει τις λεπτομέρειες της προσομοίωσης στον χρήστη. Γενικά αντί να προσπαθήσει κάποιος να “εφαρμόσει” την δική του φυσική, είναι απλούστερο και καλύτερο να χρησιμοποιήσει την μηχανή φυσικής του Unity όσο περισσότερο μπορεί.

2.3.1 Collisions

Συχνά, κατά την δημιουργία ενός παιχνιδιού, θα επιθυμούσαμε μια σύγκρουση δύο αντικειμένων να οδηγήσει σε κάποια αλλαγή κατάστασης στο κώδικα. Για να επιτευχθεί κάτι τέτοιο χρησιμοποιείται η μέθοδος ανίχνευσης σύγκρουσης (Collision Detection Method).

Αρχικά ένα τουλάχιστον από τα αντικείμενα στην σύγκρουση χρειάζεται να έχει ένα non-kinematic Rigidbody, ενώ και τα δύο αντικείμενα πρέπει να έχουν σωστούς Colliders αρχικοποιημένα σωστά. Η συνολική ταχύτητα και των δύο πρέπει να είναι αρκετά μικρή ώστε να υπάρχει εντοπισμός της σύγκρουσης και να μην περνάει το ένα μέσα από το αλλό χωρίς να εντοπίζεται η σύγκρουση. Αφού ελεγχθούν όλα αυτά, υπάρχει μία ειδική μέθοδος για ανίχνευση της σύγκρουσης μέσω ενός script που επισυνάπτεται στο αντικείμενο με το οποίο θα θέλαμε να ελέγξουμε την σύγκρουση.

2.3.2 Materials

Τα materials είναι objects που διαμορφώνουν παραμέτρους τριβής μεταξύ των επιφανειών των αντικειμένων. Με την δυνατότητα παραμετροποίησης μπορεί να αλλάξει η τριβή ενός υλικού ή η ελαστικότητα ενός αντικειμένου, αλλάζοντας τις συνθήκες κίνησης και την συμπεριφορά κατά την εφαρμογή δυνάμεων ή και κρούσεων εταξύ των αντικειμένων του ψηφιακού κόσμου. Τα materials μπορούμε να τα τροποποιούμε και να τα αποθηκεύουμε και έτσι η ανάπτυξη γίνεται εύκολα και σε ελάχιστο χρόνο.

2.3.3 Raycasting

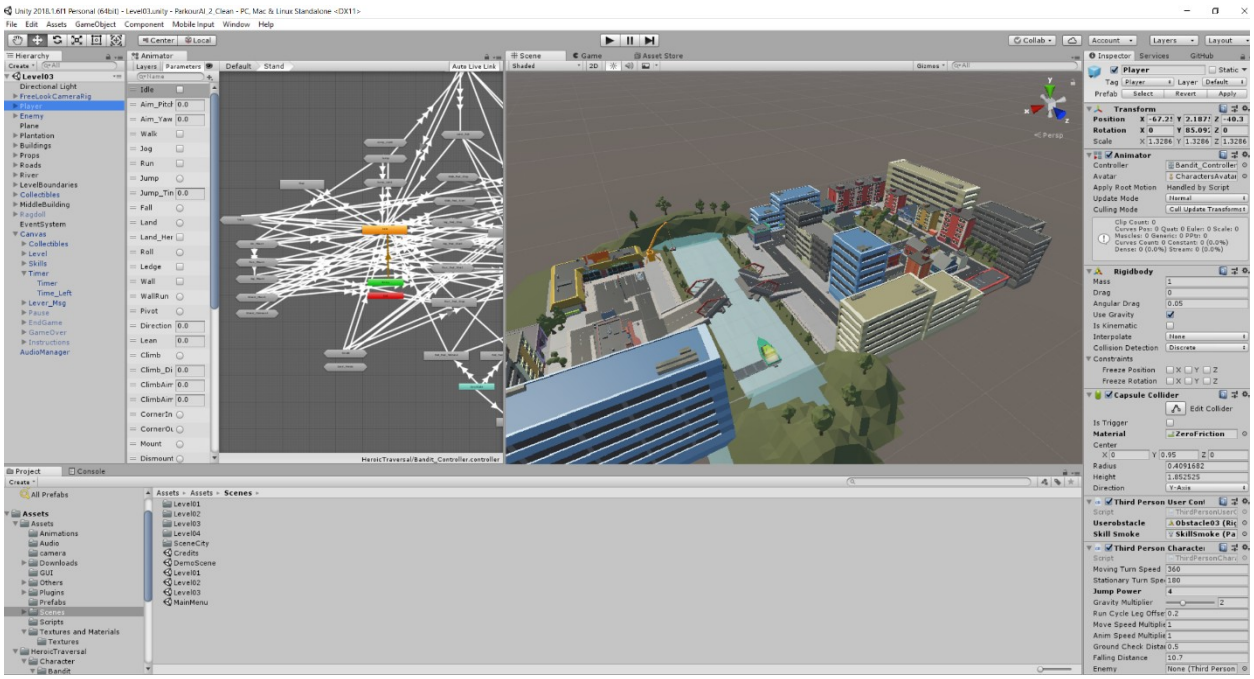
Ένας ακόμη βασικός μηχανισμός μεγάλης κρισιμότητας ειδικά για το δικό μας παιχνίδι αλλά και γενικότερα είναι το Raycasting. Με το Raycasting δίνεται η δυνατότητα να εντοπίσουμε «οπτική» επαφή μεταξύ αντικειμένων και να την διαχειριστούμε προγραμματιστικά μέσω ενός script.

Στην ουσία προβάλεται από ένα σημείο εκκίνησης μια ευθεία γραμμή (μπορούμε να το φανταστούμε σαν ακτίνα lazer- εξ ου και το όνομα) με μια κατεύθυνση και ένα μέγιστο

μήκος. Όταν αυτή η ακτίνα συνατνήσει ένα άλλο αντικείμενο και «συγκρουστεί» με αυτό επιστρέφεται από το Unity ένα αντικείμενο με περεταίρω πληροφορίες για την επαφή.

Προαιρετικά μπορούμε να αξιοποιήσουμε και τα layer tags που ορίζονται για κάθε αντικείμενο ώστε το Raycasting να γίνεται πάνω σε λογικές ομάδες αντικειμένων ορισμένες από τον προγραμματιστή.

2.4 Unity 3D IDE



Εικόνα 2.2: Το περιβάλλον ανάπτυξης Unity

Στην παραπάνω είκονα παρουσιάζεται το περιβάλλον ανάπτυξης Unity3D. Τα κύρια στοιχεία που διακρίνονται είναι:

- **Project**: Όλα τα αρχεία του project σας. Μπορούμε να κάνουμε drag and drop αρχεία απο τον Explorer μέσα στο project Unity.
- **Scene**: Η σκήνη την οποία δουλεύουμε αυτή τη στιγμή.

- **Hierarchy**: Όλα τα αντικείμενα του παιχνιδιού στην σκηνή.
- **Inspector**: Τα συστατικά (components) του επιλεγμένου αντικειμένου στη σκηνή.
- **Console**: Δείχνει την έξοδο από την μεταγλώττιση, τα λάθη, τις προειδοποιήσεις κτλ. Δείχνει επίσης τα μηνύματα σφαλμάτων που ενδέχεται να βάλαμε στον κώδικα (π.χ. Debug.Log)
- **Game**: Αυτή η καρτέλα ενεργοποιείται όταν πατηθεί το κουμπί play και το παιχνίδι αρχίζει να τρέχει σε αυτό το παράθυρο. Τότε το παιχνίδι μπαίνει σε play mode επιτρέποντας να γίνουν ζωντανές αλλαγές μέσω του Scene Tab.
- **Animator**: Παρέχει απεικόνιση του γράφου των animation states και μάλιστα έχει live λειτουργία με απεικόνιση σε πραγματικό χρόνο ώστε να διευκολύνει το debugging.

3 Γραφικά

Τα μοντέλα γραφικών που επιλέχθηκαν έχουν στύλ low polygon ώστε να είναι δυνατό το παιχνίδι να τρέξει σε παλιό ή χαμηλών προδιαγραφών hardware χωρίς προβλήματα, διατηρώντας παράλληλα ένα πολύ υψηλό επίπεδο αισθητικής. Η δημιουργία του εικονικού περιβάλλοντος που περιλαμβάνει τα κτήρια τις επιφάνειες τα props και τους χαρακτήρες έχει διαμορφωθεί ως collage του ίδιου στυλ μοντέλων από διαφορετικούς δημιουργούς διατηρόντας όμως έναν κοινό χαρακτήρα. Όλα τα levels αποτελούν πρωτότυπη δουλεία αφού έχουν σχεδιαστεί και δημιουργηθεί από το μηδέν.

4 Σκοπός του παιχνιδιού

Ο σκοπός του παιχνιδιού είναι να συλλέξει έναν αριθμό κλειδιών ώστε να ξεκλειδώσει το σημείο τερματισμού της κάθε πίστας. Τον χαρακτήρα κυνηγάει ένας εχθρός ελεγχόμενος από τον υπολογιστή (NPC) που με αλγόριθμους τεχνητής νοημοσύνης κυνηγάει τον χαρακτήρα. Αν τον ακουμπήσει το παιχνίδι τελειώνει και δίνεται η δυνατότητα επανάληψης του επιπέδου. Ο χαρακτήρας έχει ικανότητες parkour ώστε να ανεβαίνει πάνω σε ταράτσες και μπαλκόνια κτιρίων, μια ιδιότητα που αν χρησιμοποιηθεί σωστά του επιτρέπει να αποφύγει τον εχθρό.

Για να περάσει απιτυχώς ένα level ο χρήστης θα πρέπει να συλλέξει ένα αριθμό από κλειδιά τα οποία ανοίγουν μέσω script props κατάλληλα τοποθετημένα στην κάθε πίστα. Τα props αυτά είναι μια εικονική πύλη στην πρώτη πίστα μια πόρτα στην δεύτερη πίστα και στην τρίτη μια γέφυρα που ανοίγει την πρόσβαση στην απέναντι όχθη ενός ποταμού.

Ο αριθμός των κλειδιών που πρέπει να συλλεγούν αλλά και ο βαθμός δυσκολίας της συλλογής τους αυξάνεται σε κάθε level. Επιπλέον ο αντίπαλος έχει ρυθμιστεί ώστε να τρέχει πιο γρήγορα σε κάθε επόμενο level. Τα στοιχεία αυτά προσθέτουν έναν χαρακτήρα αυξανόμενης δυσκολίας στο παιχνίδι με δύο κυρίαρχες επιπτώσεις στην εμπειρία χρήσης. Από τη μία η δυσκολία αυξάνει παράλληλα με την εξοικείωση του χρήστη με το περιβάλλον και τα game mechanics ώστε να μην απογοητεύει κάποιον πρωτάρη. Αυτό συμβαίνει διότι δίνει τον χρόνο που χρειάζεται κανείς ώστε να πετύχει την αποστολή χωρίς πολλές απαναλήψεις. Από την άλλη διατηρεί το ενδιαφέρον των χρηστών ζωντανό επειδή καθώς αυτοί εξοικειώνονται και βελτιώνονται ως παίκτες δεν σταματά να τους προκαλλεί να γίνουν ακόμη καλύτεροι ώστε να είναι σε θέση να το τερματίσουν.

4.1 Σενάριο

Ο κυρίως χαρακτήρας του παιχνιδιού είναι ο Adam. Ο Adam δούλευε σαν σεναριογράφος για μια εταιρεία παραγωγής παιχνιδιών για πάνω από μια δεκαετία. Μια βομβιστική επίθεση εκδηλώθηκε, και ο πρωταγωνιστής μας καθώς παρακολουθούσε τα νέα στο

έκτακτο δελτίο συνειδητοποίησε ότι η μέθοδος των βομβιστών είναι ολόιδια με μια σκηνή δράσης που είχε δημιουργήσει για την εταιρεία πριν μερικά χρόνια για ένα παιχνίδι fps.

Τη σκηνή την είχαν δει μόνο ο μεγαλομέτοχος της εταιρείας και ο σκηνοθέτης. Η σκηνή είχε κοπεί για την μεγάλη της πολυπλοκότητα και βαρβαρότητα. Δεν θα μπορούσε να είναι σύμπτωση. Η αναστάτωσή του κορυφώθηκε όταν άκουσε από την δημοσιογράφο ότι ανάμεσα στα θύματα βρισκόταν η αδερφή του! Ήταν σίγουρο ότι ο μεγαλομέτοχος Drake ήταν υπεύθυνος, ο σκηνοθέτης Abot είναι ένας φιλήσυχος καλλιτέχνης που δεν μπορεί να πειράξει μυρμηγκάκι, άσε που δεν έχει και πρόσβαση σε εξωτικά εκρηκτικά υλικά.

Ο Adam εξοργισμένος από την κατάσταση αποφάσισε να πάρει εκδίκηση. Θα επιχειρήσει να κλέψει πολύτιμες πληροφορίες από την εταιρεία που ενοχοποιούν τον ιδιοκτήτη και να τις μεταφέρει στην εισαγγελέα. Στο κατώπι του βρίσκεται η εκπαιδευμένη ninja Lynx. Θα τα καταφέρει?

5 Ανάπτυξη 3D Παιχνιδιού

Η κατασκευή του 3D παιχνιδού με χρήση του Unity έγινε πάνω σε δύο κύριους άξονες: το κομμάτι του γραφικού σχεδιασμού και το κομμάτι του κώδικα στα διάφορα scripts. Για το πρώτο κομμάτι, το Unity προσφέρει μια πληθώρα εργαλείων, με τα οποία, κατασκευάσαμε τα αντικείμενα που επιθυμούσαμε να περιλαμβάνει το παιχνίδι και τα τοποθετήσαμε στο επιυθμητό σημείο της σκηνής.

Το δεύτερο κομμάτι, όπως αναφέραμε, αποτελείται απο τα scripts. Δηλαδή αρχεία κώδικα τα οποία σχετίζονται με κάποια συγκεκριμένα αντικείμενα. Τα scripts προσδίδουν κάποια καθορισμένα χαρακτηριστικά σε αυτά τα αντικείμενα και ορίζουν τη συμπεριφορά τους σε συγκεκριμένες ενέργειες του χρήστη, όπως το πάτημα ενός πλήκτρου. Το Unity είναι σχεδιασμένο να δέχεται αρχεία κώδικα σε C#.

Αρχικά, το παιχνίδι αποτελείται από μία σκηνή η οποία είναι το κυρίως μενού και στην οποία επιστρέφει ο έλεγχος του προγράμματος μετά το πέρας κάθε πίστας ή με επιλογή του χρήστη. Οι πίστες αποτελούν σκηνές που εναλλάσσονται μεταξύ τους και αυτή η εναλλαγή πραγματοποιείται με τον εντοπισμό ενός προγραμματισμένου event (trigger) που έχει οριστεί σε ένα κατάλληλο script. Για την δημιουργία του συγκεκριμένου μηχανισμού απαιτείται το script που ελέγχει τα events να είναι κληρονομεί την κλάση MonoBehavior, το οποίο διασφαλίζει ότι θα υπάρχει μόνο ένα instance του αντικειμένου στον runtime περιβάλλον.

στον εισαγγελέα. Από τότε οι άνθρωποι του Drake τον κυνηγούν ανηλεώς να τον σκοτώσουν. Στο πόδι του αυτή τη φορά βρίσκεται η Lynx μια ninja εκπαιδευμένη από την Mosad. Η Lynx έχει μοναδικές ικανότητες εντοπισμού και μπορεί να τρέχει ακατάπαυστα. Ο Adam πρέπει να δραπετεύσει και να φτάσει ως τον εισαγγελέα μέσα σε τρεις μέρες, θα τα καταφέρει?

5.1 Scripting

Το scripting είναι δυνατό να υλοποιηθεί κληρωνομόντας την κλάση MonoBehaviour. Το MonoBehaviout είναι ένα programming interface με σαφείς ορισμούς και περιορισμούς που προκύπτουν από τον προκαθορισμένο τρόπο αλληλεπίδρασης με τα αντικείμενα.

Η αλληλεπίδαρση γίνεται μόνο μέσω class methods. Παρότι μπορεί να φαντάζει ότι οι περιορισμοί θα μπορούσαν να ελαχιστοποιήσουν τις επιλογές του προγραμματιστή, η ύπαρξη μεγάλης γκάμας μεθόδων και εργαλείων αλλά και το γεγονώς ότι οι κλάσεις και μέθοδοι αυτές είναι low level στην υλοποίησή τους, επιτρέπει στην ουσία οποιαδήποτε λειτουργία είναι πιθανό να χρειαστεί για την ανάπτυξη ενός παιχνιδιού.

Σε επόμενα κεφάλαια αναλύεται ο τρόπος που έγινε ανάπτυξη μέσω scripts για την λειτουργία του παιχνιδιού, τα game mechanics, το control, το gameplay, την κάμερα, τον animator και την μουσική.

5.2 Scenes

Το παιχνίδι αποτελείται από τέσσερις σκηνές, οι τρείς σκηνές είναι τα levels από τα οποία πρέπει να περάσει ο χρήστης για να τερματίσει το παιχνίδι και η τέταρτη είναι η σκηνή του κεντρικού μενού από όπου ο χρήστης κάνει navigate στις υπόλοιπες.

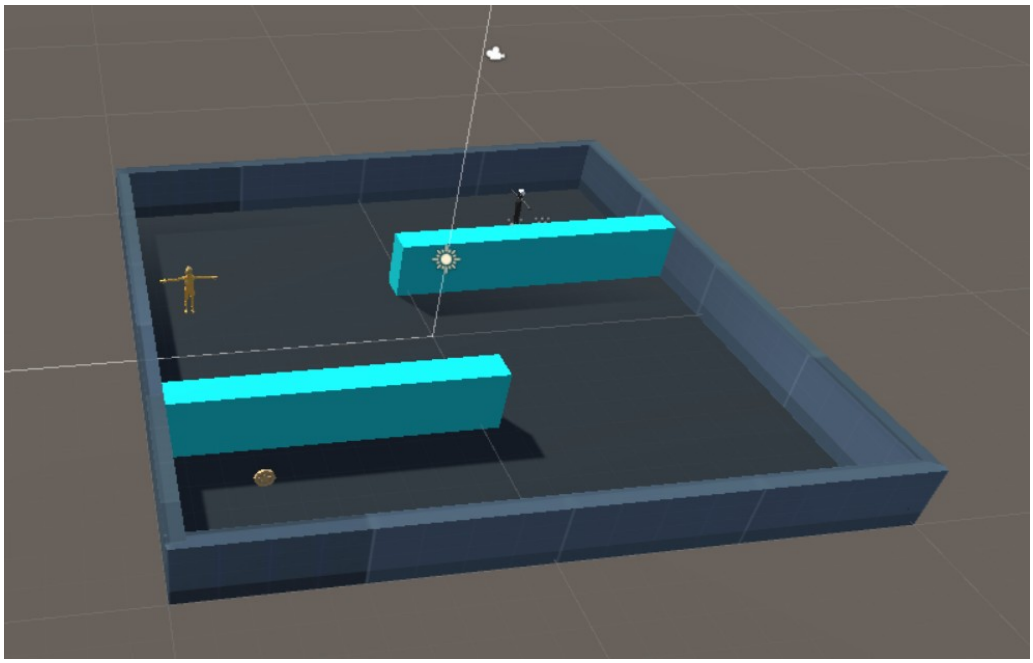
5.2.1 Main Menu

Το main menu είναι και αυτό μία σκηνή. Αποτελέι ένα κοινό σημέιο από το οποίο περνάει ο χρήστης μεταξύ των levels αλλά παρέχει και πληροφορίες για τον χειρισμό του παιχνιδιού. Ουσιαστικά είναι ένα τετράγνωνο 2D frame πάνω στο οποίο έχουν προστεθεί κουμπιά αλληλεπίδρασης. Η αλληλεπίδραση ελέγχεται μέσα παό scripts που έχουν οριστεί σε ένα game object της σκηνής.

Εικόνα 4.3: Main menu

5.2.2 Level 1

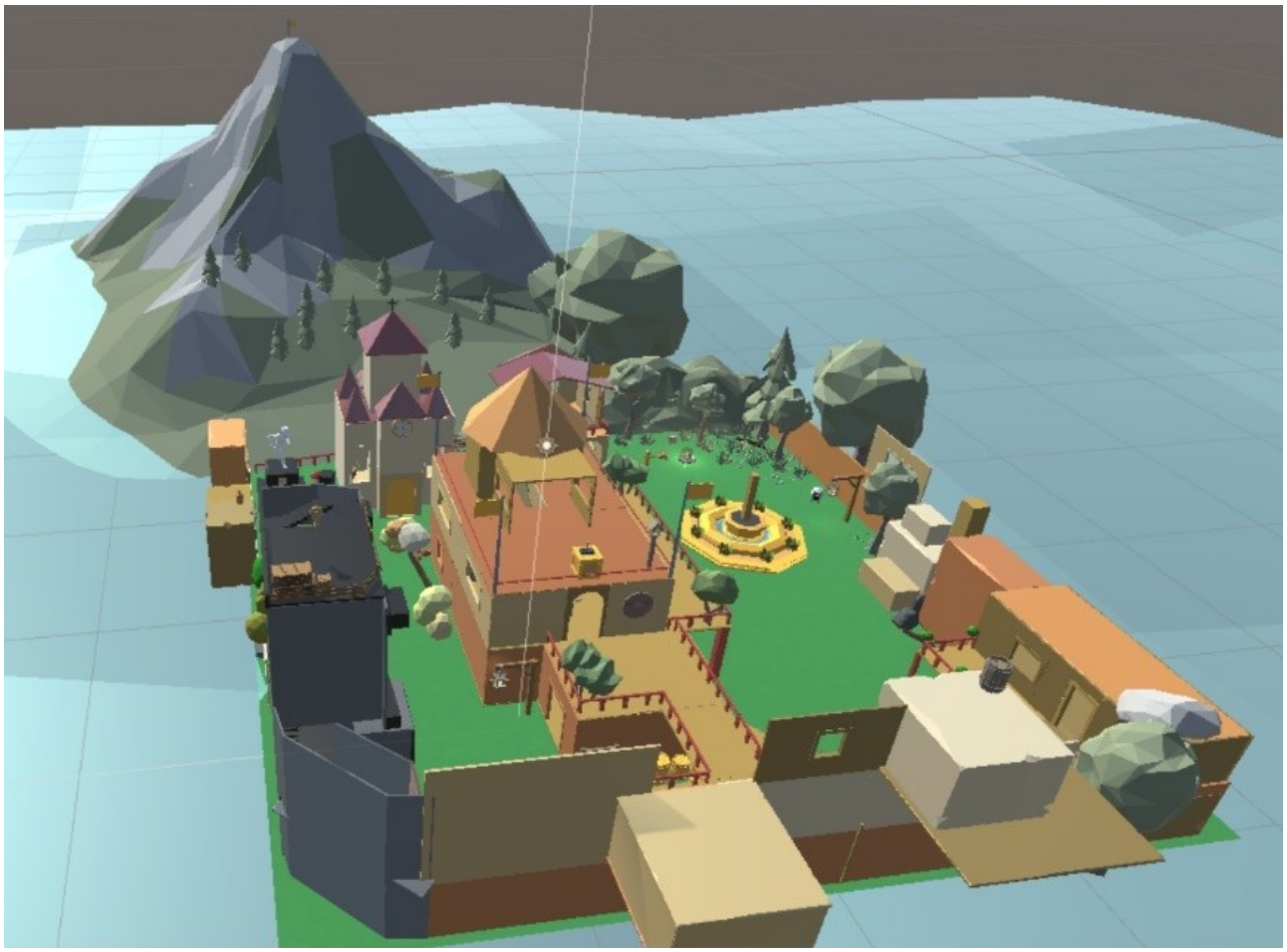
Το πρώτο level είναι εκπαιδευτικού χαρακτήρα και για αυτό το λόγο αποφασίστηκε να έχει πολύ μικρές διαστάσεις ο χώρος του. Ουσιαστικά δίνει την ευκαιρία στον χρήστη να εξοικειωθεί με τον χειρισμό του παιχνιδιού αλλά και με τα game mechanics και τα χαρακτηριστικά του εχθρού.



Εικόνα 4.4: Level 1

5.2.3 Level 2

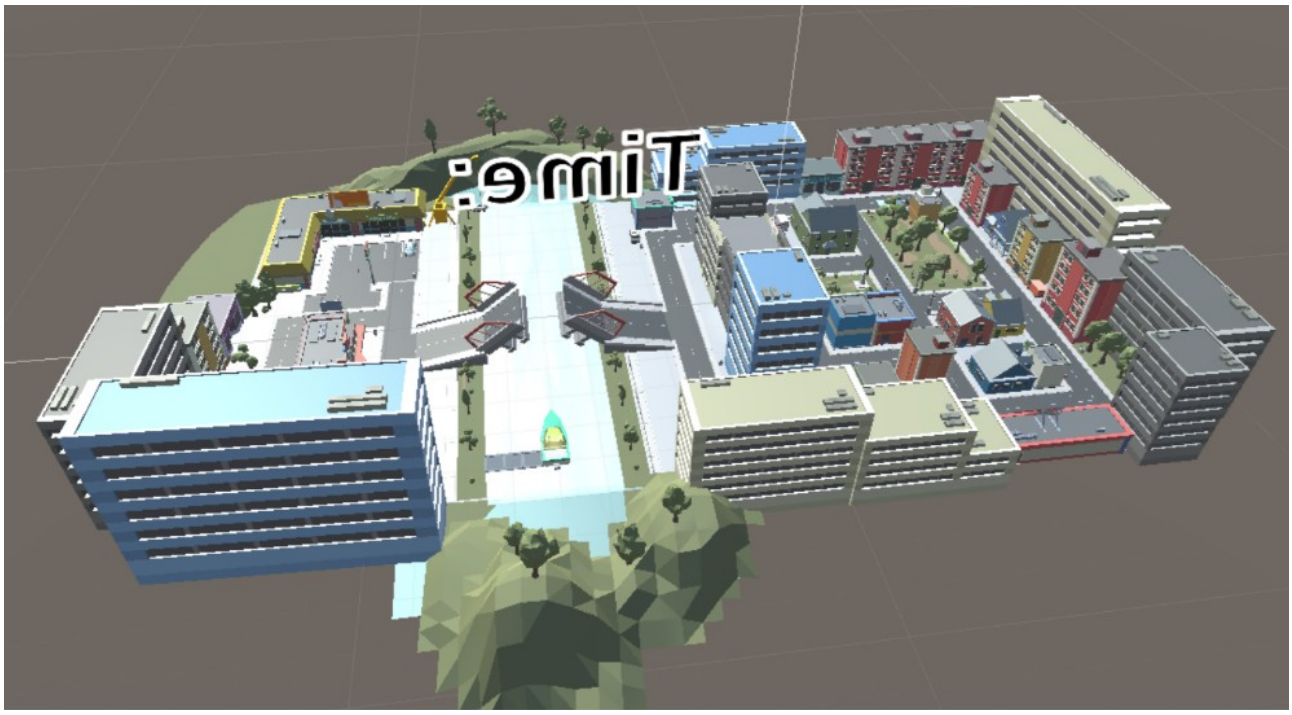
Το δεύτερο level είναι μια far-west τύπου πόλη, έχει μια εκκλησία ένα κεντρικό κτήριο με διαθέσιμο εσωτερικό χώρο και πολλούς ορόφους και διάφορα κτήρια περιμετρικά τα οποία και αυτά διαθέτουν προσβάσιμο εσωτερικό. Στην ταράτσα του κεντρικού κτήριου βρίσκεται ένας μοχλός ο οποίος ξεκλειδώνει το πέρασμα στο πιο ψηλό κτήριο της πίστας όπου βρίκεται ο τερματισμός. Τα κτήρια και η διαμόρφωσή τους είναι τέτεια ώστε να υπάρχουν πολλά σημεία για σκαρφάλωμα.



Εικόνα 4.5: Level 2

5.2.4 Level 3

Το τρίτο level έχει χαρακτήρα μεγαλούπολης. Αποτελείται από δύο μέρη που τα χωρίζει ποταμός. Η γέφυρα που ενώνει τις απέναντι όχθες είναι ανοιχτή και ο χρήστης δεν έχει πρόσβαση στο σημείο τερματισμού παρά μόνο αν έχει συλλέξει τα κλειδιά. Η πόλη διαθέτει δρόμους, αυτοκίνητα, χαμηλά σπίτια τύπου αμερικάνικων προαστίων αλλά και μεγάλα τύπου γραφείων. Περιμετρικά της πόλης έχουν τοποθετηθεί ψηλά κτήρια τα οποία χρησιμεύουν και ως οριοθέτηση της περιμέτρου της πίστας. Η πόλη προσφέρει πολλά σημεία για σκαρφάλωμα κυρίως στο κέντρο της.



Εικόνα 4.6: Level 3

5.3 Βασικά Δομικά Στοιχεία

Στο παρών κεφάλαιο περιγράφονται τα βασικά δομικά στοιχεία που σε συνέργεια συνθέτουν το παιχνίδι με όλες του τις λειτουργίες. Για την υλοποίηση του παιχνιδιού και του κόσμου δημιουργήθηκαν scripts και έγιναν κατάλληλες ρυθμίσεις ώστε ο χαρακτήρας να αλληλεπιδρά με το περιβάλλον, ο εχθρός να τον εντοπίζει και να τον κυνηγά, τα animations να επεκταθούν και να εκτελούνται την κατάλληλη στιγμή, η πίστα να μεταβάλλεται ανάλογα με την δράση του χρήστη, να επιτρέπεται η αναπαργωγή ηχητικών εφέ σε καθορισμένα events, να ελέγχεται η αναπαραγωγή μουσικής, να επιτρέπεται το navigation μεταξύ των levels και του κεντρικού μενού, και τέλος να τηρούνται στοιχεία του game progress.

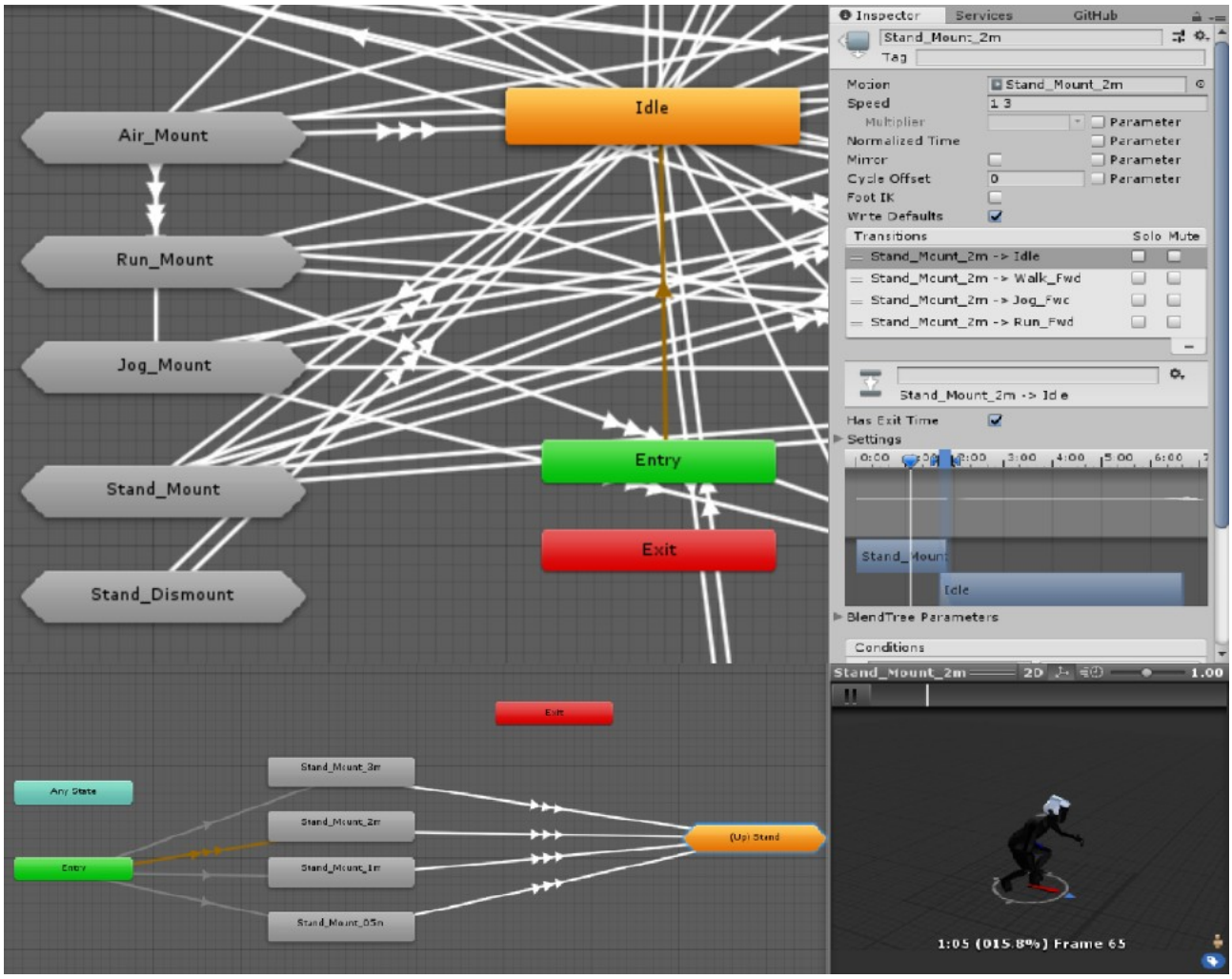
5.3.1 Player

Animator

Για την δημιουργία του παιχνιδιού αγοράστηκε από το AssetStore της Unity ένα πακέτο animations το οποίο παρείχε τα κατάληλλα animation clips στον χαρακτήρα μας ώστε αυτός να έιναι σε θέση να κάνει κινήσεις parkour. Το bundle είχε έναν animator που χρειάστηκε να επεκταθεί και να ρυθμιστεί κατάλληλα ώστε να εξυπηρετήσει τους σκοπούς της παρούσας εργασίας.

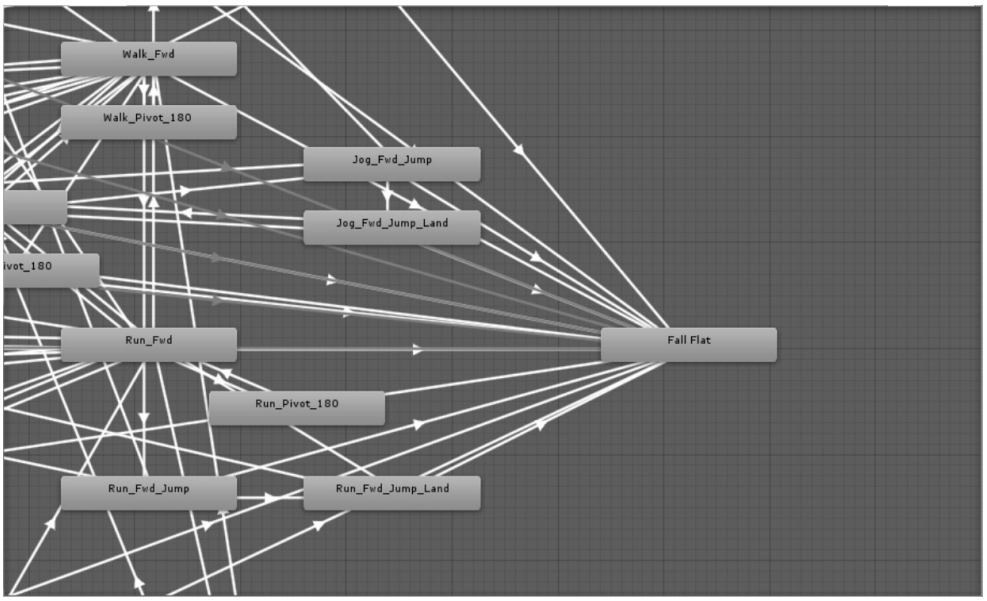
Τα animations του πακέτου υποδείκνυαν την χρήση του κάθε animation clip ως “ελεγκτή” της ταχύτητας του avatar του χαρακτήρα. Αυτό έχει σαν αποτέλεσμα πολύ πιο ομαλή κίνηση κάτι που σημαίνει ότι οπτικά το αποτέλεσμα είναι πολύ ανώτερο, όμως έχει το μειονέκτημα της μικρότερης δυνατότητας επέμβασης από τον προγραμματιστή αλλά και συνεπακόλουθους περιορισμούς στην αμεσότητα του χειρισμού. Για παράδειγμα, για την αλλαγή από ένα state running σε ένα state walking απαιτείται ο χαρακτήρας να ολοκληρώσει πρώτα το τρέχων clip ώστε να αρχίσει το επόμενο. Αυτό σημαίνει ότι ο χρήστης θα δεί την εντολή για μείωση στην ταχύτητα να εκτελείται μετά από 1 δευτερόλεπτο.

Στον animator αλλάχθηκαν τα default states που αφορούσαν το parkour ώστε να ταιριάζουν καλύτερα με τις συνθήκες της κίνησης στην υλοποίησή μας. Επίσης έγινε fine tuning στις παραμέτρους της ταχύτητας και της διάρκειας του clip.



Εικόνα 4.7: Parkour animations

Για τα animations προστέθηκε ένας επιπλέον κόμβος ώστε να υλοποιηθεί η περίπτωση ήττας όπου ο εχθρός πιάνει τον παίχτη. Αυτό το animation κατεβάστηκε από την πλατφόρμα Mixamo και έγινε import στον controller συνδέθηκε με όλα τα άλλα πιθανά states του παιχνιδιού μας και του δώθηκαν χαρακτηριστικά trigger ώστε να μπορούμε να το ενεργοποιοήσουμε από το gameplay script.



Εικόνα 4.8: Fall Flat (Game Over) animator node

Λειτουργία Εμποδίων

Η λειτουργία αυτή έχει σκοπό να δώσει ένα ανταγωνιστικό πλεονέκτημα στον παίκτη σε σχέση με τον εχθρό του. Του επιτρέπει πατώντας το πλήκτρο Q να κάνει spawn ένα βαρύ αντικείμενο πίσω του. Ο παίχτης έχει δικαίωμα για συνολικά 3 τέτοια spawns σε κάθε level και υπάρχει μηχανισμός cooldown που καθορίζει πόσο συχνά μπορεί να τα ενεργοποιήσει.

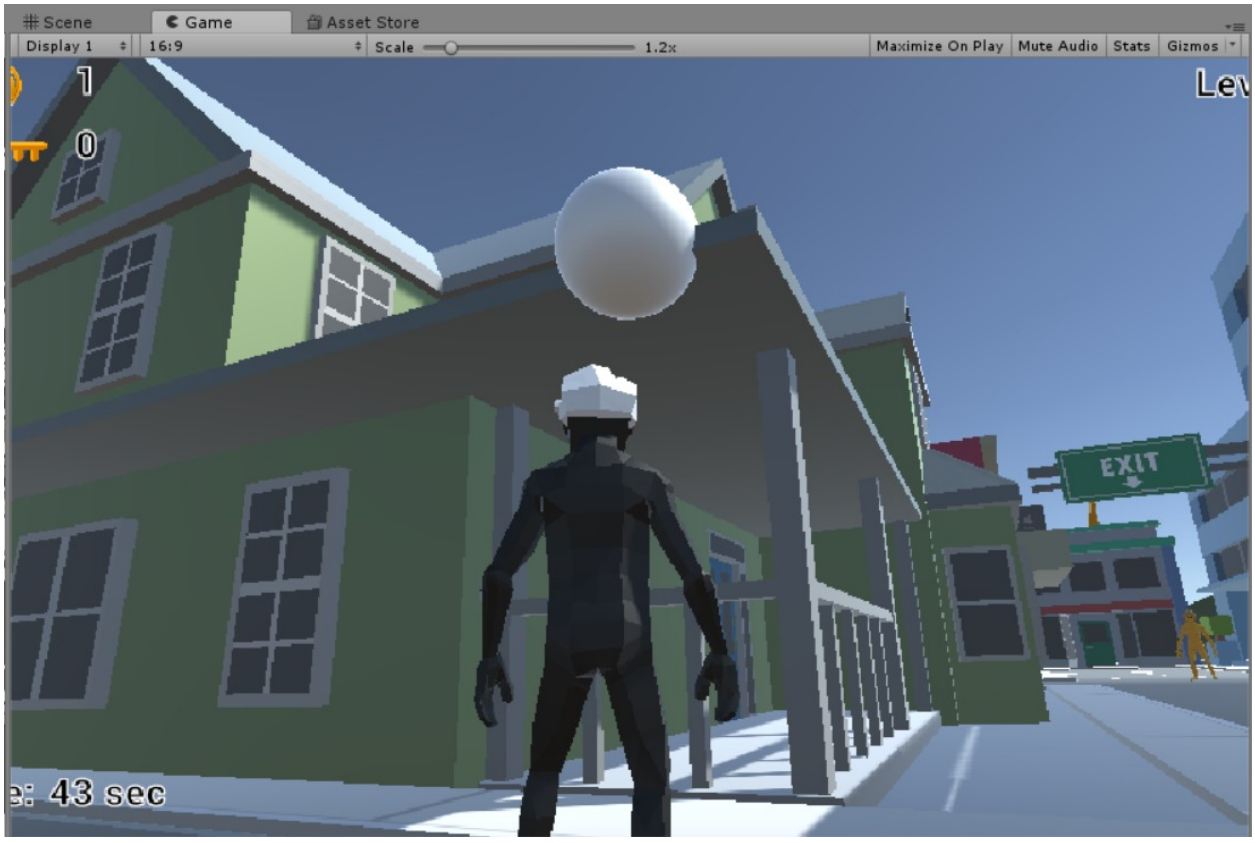
Λειτουργία Parkour

Έχει υλοποιηθεί λειτουργία parkour που επιτρέπει στον χαρακτήρα να σκαρφαλώνει σε αντικείμενα σε μεγάλο ύψος. Για την υλοποίηση χρησιμοποιήθηκαν spherecasts με layers, lerping και animation controls. Τα spherecasts επιτρέπουν τον εντοπισμό ενός σημείο στο οποίο θα ήταν δυνατή αυτή η ενέργεια. Τα spheres που προβάλλονται ξεκινούν από μεγαλύτερο ύψος από το κεφάλι του παίχτη και έχουν κατεύθυνση προς τα κάτω. Αν εντοπιστεί άλλο αντικείμενο και ταυτόχρονα ανοίκει στο layer group που έχει οριστεί τότε ενεργοποιείται η δυνατότητα για σκαρφάλωμα.

Το πλήκτρο του άλματος μετατρέπεται στιγμιαία σε πλήκτρο σκαρφαλώματος και ο χαρακτήρας εκτελεί το animation σκαρφαλώματος. Την ίδια στιγμή απενεργοποιείται το rigidbody ώστε να μην ισχύουν καθόλου κανόνες φυσικής πάνω στον χαρακτήρα και ξεκινάει μια μετατόπιση με χρήση της συνάρτησης lerp (linear interpolation) ώστε να μετατοπιστεί ο παίχτης στο σημείο επαφής της σφαίρας με την επιφάνεια.

Έχουν χρησιμοποιηθεί τρεις σφαίρες σε διαφορετική απόσταση μπροστά και πάνω από τον παίχτη ώστε να είναι πιο εύχρηστος ο χειρισμός ανάλογα με την ταχύτητα κίνησης. Τέλος έχει χρησιμοποιηθεί άλλη μια σφαίρα που ξεκινάει από το κεφάλι του παίχτη και εκτείνεται κάθετα προς τα πάνω ώστε να εντοπίζεται αν υπάρχει τυχόν εμπόδιο και να ακυρώνεται η δυνατότητα σκαρφαλώματος. Οι σφαίρες κινούνται μαζί με τον παίχτη «αντιγράφοντας» το transform του αντικειμένου του.

Στην παρακάτω εικόνα φαίνεται μια προβολή της σχετικής σφαίρας για σκοπούς debugging του μηχανισμού. Στη σφαίρα έχει δοθεί η τοποθεσία του hit του spherecast.



Εικόνα 4.9: Σημείο διαθέσιμου σκαρφαλώματος

Camera

Σαν βάση γαι την ανάπτυξη της κάμερας χρησιμοποιήθηκε το script της Unity που ανοίκει στα standard assets και είναι ελεγχόμενη από το mouse. Η κάμερα έχει επεκταθεί ώστε ο χειρισμός και η λειτουργία της να αντιστοιχούν στα σημερινά standards των 3rd person εμπορικών παιχνιδιών. Έχουν προστεθεί οι λειτουργίες:

- Player direction control
- Snap to obstacles
- Zoom in / Zoom out

Όσο αφορά το Player direction control, η κάμερα έχει τροποποιηθεί ώστε να παρεμβαίνει στον χειρισμό του χαρακτήρα ανεξάρτητα από το πληκτρολόγιο. Με αυτό τον τρόπο επιτρέπεται η χρήση του παιχνιδιού τόσο με Mouse & Keyboard όσο και αποκλειστικά από Gamepad.

Το rotation του χαρακτήρα αλλάζει ανάλογα με την κατεύθυνση της κάμερας στον 3D κόσμο του παιχνιδιού. Αυτό επιτυγχάνεται απομονώνοντας την περιστροφή της κάμερας γύρω από τον κάθετο άξονα και τροποποιόντας το rotation vector του χαρακτήρα με late update.

Η κάμερα ακολουθεί τον χαρακτήρα αντιγράφοντας την τοποθεσία του αλλά διατηρώντας δικό της ανεξάρτητο ύψος (άξονας Z):

```
Vector3 newSpot = new Vector3(xi, yi, zi);

newSpot = new Vector3(newSpot.x, yi + offsetv.y, newSpot.z);//keep original height

transform.position = target.position + newSpot;
```

Το σχετικό αρχείο είναι το mouselook.cs στον φάκελο των Scripts.

Για αποτραπεί το cliping άλλων αντικειμένων του περιβάλλοντος από την κάμερα υλοποιήθηκε μηχανισμός που αξιοποιεί το raycasting. Η μέθοδος που ακολουθήθηκε ήταν να προβάλλεται μία

σφαίρα με φορά αντίθετη από αυτή που κοιτάζει η κάμερα και να καταγράφεται το σημείο επαφής με τον κόσμο. Αν τυχόν έχουμε επαφή (hit) τότε η κάμερα τοποθετείται στο σημείο επαφής αντί το συνηθισμένο. Το συνολικό αποτέλεσμα είναι η κάμερα να εφάπτεται σε τοίχους ή τυχόν άλλα αντικείμενα όταν η by default θέση της θα ήταν μέσα σε ένα σντικείμενο.

```
// if the ray hits something the camera is moved to a new position

protecting = hitSomething;

m_CurrentDist = Mathf.SmoothDamp(m_CurrentDist, targetDist, ref m_MoveVelocity,m_CurrentDist
> targetDist ? clipMoveTime : returnTime);

m_CurrentDist = Mathf.Clamp(m_CurrentDist, closestDistance, m_OriginalDist);

m_Cam.localPosition = -Vector3.forward*m_CurrentDist;
```

Το σχετικό αρχείο είναι το ProtectCameraFromWallClip.cs στον φάκελο των Scripts.

5.3.2 Enemy

5.3.3 Nav Mesh

5.3.4 Gameplay Controller

5.3.5 GUI Overlay

5.3.6 Music Controller

Για την διαχείρηση της αναπαραγωγής της μουσικής χρειάστηκε η δημιουργία ενός Music controller. Αν και ο audio manager της Unity φάνηκε επαρκής για την διαχείρηση των εφέ μέσω scripts, το Unity δεν παρέχει δικό του υποσύστημα με αρκετά χαρακτηριστικά ώστε να επιτρέπει την εναλλαγή της μουσικής μεταξύ των σκηνών κατά το δοκούν.

Ο πυρήνας του script είναι ο εξής:

```
// Checking which scene is loaded (for handling music)

        currentScene = SceneManager.GetActiveScene().name;

        if (currentScene != lastScene && lastScene == "MainMenu")

        {

            ChangeSong();

            lastScene = currentScene;

        }

        if (currentScene != lastScene &&  currentScene == "MainMenu") {

            ChangeSong();

            lastScene = currentScene;

        }
```

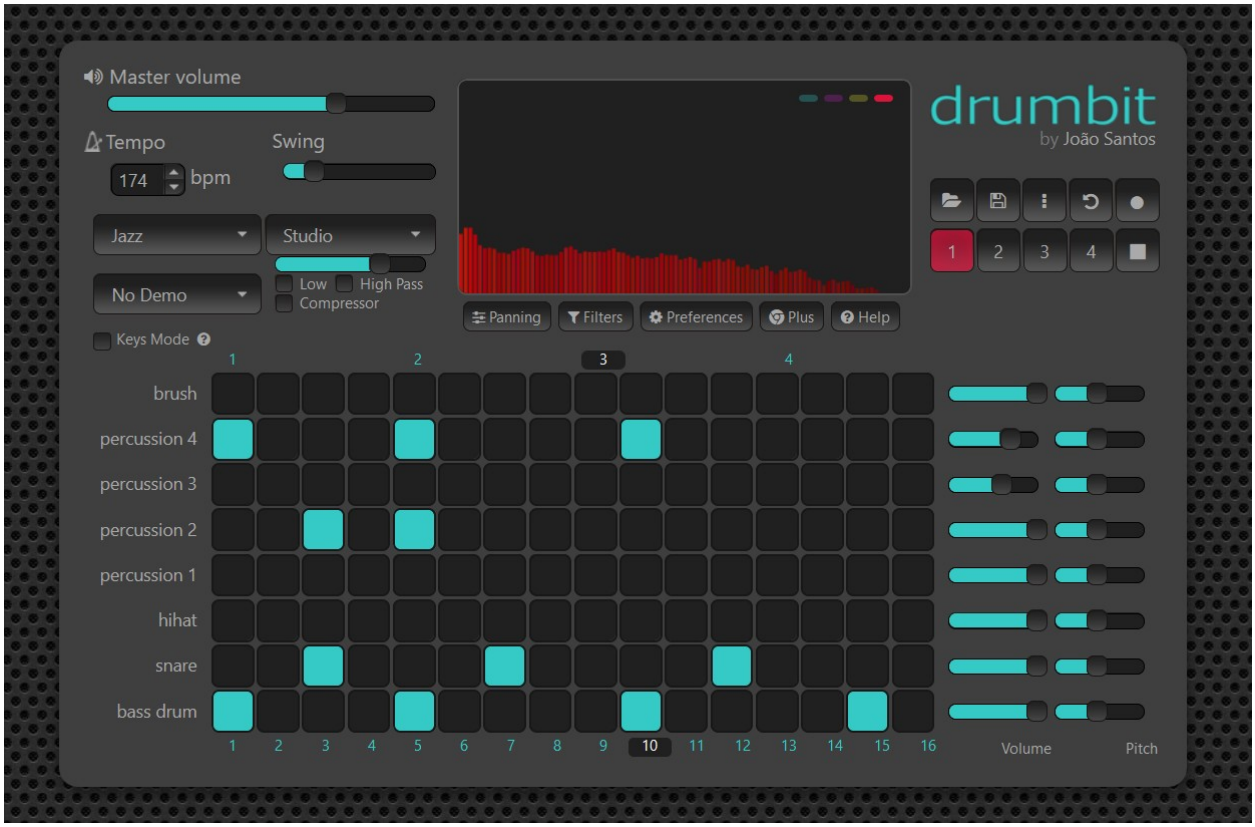
Ουσιαστικά το script έχει γνώση του ποιά ήταν η προηγούμενη σκηνή και εφαρμόζει με βάση αυτούς τους κανόνες την αναπαραγωγή του εκάστοτε κομματιού.

5.4 Music Production

Συνολικά για το παιχνίδι έχουν δημιουργηθεί 2 κομμάτια, ένα για τα μενού και ένα δράσης για το gameplay. Το κομμάτι των μενού χαρακτηρίζεται από πιο ήπιο ύφος χαμηλότερη

ένταση και σκοπός του είναι να χαλαρώσει τον χρήστη. Αντίθετα το κομμάτι της δράσης έχει πολύ πιο ταχύ ρυθμό έντονο ύφος και σκοπός του είναι να αυξήσει λιγάκι την αδρεναλίνη κατά την διάρκεια του παιχνιδιού.

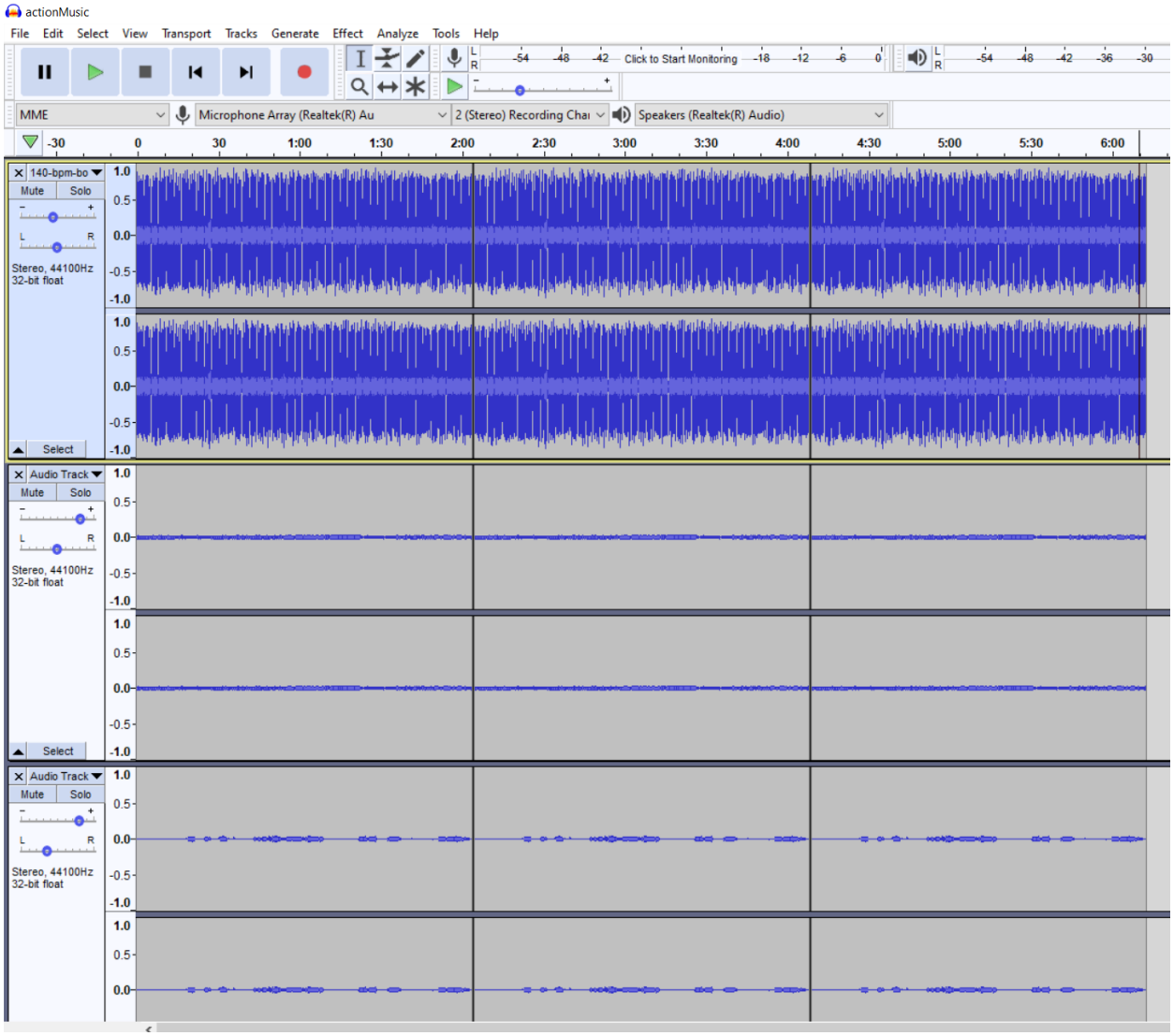
Για την δημιουργία της μουσκής χρησιμοποιήθηκε το drumbit, το audacity και ηλεκτρική κιθάρα. Πρώτα έγινε η δημιουργία του track των drums μέσω του drumbit, μετά με την χρήση του Audacity και της τεχνικής του overdub ηχογραφήθηκε η κιθάρα. Πρώτα ηχογραφήθηκε το ρυθμικό track και έπειτα το σόλο. Για την ηχογράφιση χρησιμοποιήθηκε κιθάρα Fender Stratocaster και ενισχυτής Fender Musicmaster και μικρόφωνο τοποθετημένο κοντά στο ηχείο.



Εικόνα 4.10: drumbit GUI

Το drumbit είναι ένα drum machine και επιτρέπει την δημιουργία ενός κομματίου drums. Είναι διαθέσιμο online μέσω web browser και παρέχει την δυνατότητα να κατευάσεις τα κομμάτια που δημιούργησες σε διάφορα φορματ.

Για την δημιουργία της μουσικής ο χρήστης ορίζει σε ποιο frame θα παίξει κάποιο sample και η μηχανή επαναλαμβάνει την ακολουθία σε loop. Μπορούν να ρυθμιστούς τα bpm, το swing effect το drum set που περιλαμβάνει τα samples αλλά και κάποια εφέ.



Εικόνα 4.11: Audacity Overdub

Με το Audacity είναι δυνατό να γίνει ηχογράφιση ασύγχρονα, δηλαδή αφού έχει δημιουργηθεί ένα όργανο μπορούμε να ηχογραφίσουμε «απο πάνω» του ένα δεύρτερο, τρίτο κ.ο.κ.. Στην παραπάνω εικόνα απεικονίζεται η κυρίως οθόνη δημιουργίας του κομματιού δράσης που αναπαράγεται κατά την διάρκεια του παιχνιδιού.

Η πρώτη γραμμή είναι τα drums και οι άλλες δύο η ρυθμική και σόλο κιθάρα. Η διαδικασία overdub που ακολουθήθηκε ήταν σε πρώτη φάση να ενεργοποιήσουμε τα drums να παίζουν καθώς γίνεται η ηχογράφιση της ρυθμικής κιθάρας και έπειτα χρησιμοποιήθηκε ο συνδυασμός τους για την ηχογράφιση της σόλο κιθάρας. Με αυτό τον τρόπο το κάθε track ήταν ηχογραφημένο σε σε πραγματικό χρόνο και ήταν δυνατό η μουσική όλων των οργάνων να είναι συγχρονισμένη.

6 Αναφορές

Unity:

https://create.unity3d.com/curricular-framework

https://unity.com/how-to/how-architect-code-your-project-scales

https://learn.unity.com/

https://docs.unity3d.com/2018.4/Documentation/Manual/UnityManual.html

Audacity Overdubbing:

https://manual.audacityteam.org/man/tutorial_multi_track_overdubbing.html

Heroic Traversal Manual:

https://5b9856b8-7b6c-4329-aebf-c15d37f61b0d.filesusr.com/ugd/f4fbd4_683bff8aac504ef9a09746118be3e750.pdf